LAB 6
SARTHAK BHAGAT
2016189

DEBOUNCE

```
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

```
module clock_pulse(
  input inp,
  input cclk,
  input clr,
  output outp
  );

  reg delay1,delay2,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
```

CLOCK DIVISION

```
module clk_div(
    input mclk,
    output clk190
    );

    reg [18:0] q;
    always @(posedge mclk)
    begin
        q <= q + 1;
    end

    assign clk190 = q[18];

endmodule
```

MEALY

```
module fsm_meally_10101(
    input clk,
    input clr,
    input din,
    output reg dout,
    output [2:0] present_state
    );

    reg [2:0] next_state;
    parameter S0 = 3'b000, S1 = 3'b001, S2 = 3'b010, S3 = 3'b011, S4 = 3'b100;
```

```
always @(posedge clk or posedge clr)
begin
    if (clr == 1)
    begin
        dout <= 0;
        next_state <= S0;
    end
    else
    case (present_state)
        S0 :
            if (din == 0)
            begin
                next_state <= S0;
                dout <= 0;
            end
            else
            begin
                next_state <= S1;
                dout <= 0;
            end
        S1 :
            if (din == 0)
            begin
                next_state <= S2;
                dout <= 0;
            end
            else
            begin
                next_state <= S1;
                dout <= 0;
            end
        S2 :
            if (din == 0)
            begin
                next_state <= S0;
                dout <= 0;
            end
            else
            begin
                next_state <= S3;
                dout <= 0;
            end
        S3 :
            if (din == 0)
            begin
                next_state <= S4;
                dout <= 0;
            end
            else
            begin
                next_state <= S1;
                dout <= 0;
            end
        S4 :
            if (din == 0)
            begin
                next_state <= S0;
                dout <= 0;
            end
            else
            begin
                next_state <= S0;
                dout <= 1;
            end
        default :
            begin
                next_state <= S0;
                dout <= 0;
            end
    endcase
end

assign present_state = next_state;

endmodule
```

MOORE

```
module FSM_moore_10101(
    input wire clk,
    input wire clr,
    input wire din,
    output reg dout,
    output reg [2:0] present_state
    );
    reg [2:0] next_state;
    parameter S0 = 3'b000, S1 = 3'b001, S2 = 3'b010, S3 = 3'b011, S4 = 3'b100, S5 = 3'b101;
```

```
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 (din == 1)
               next_state <= S1;
            else
               next_state <= S0;
         S1: if (din == 0)
               next_state <= S2;
            else
               next_state <= S0;
         S2: if (din == 1)
               next_state <= S3;
            else
               next_state <= S0;
         S3: if (din == 0)
               next_state <= S4;
            else
               next_state <= S0;
         S4: if (din == 1)
               next_state <= S5;
            else
               next_state <= S0;
         S5: if (din == 1)
               next_state <= S1;
            else
               next_state <= S0;
      default next_state <= S0;
   endcase
end

always @ (*)
   begin
      if ( present_state == S5)
```

```
            dout = 1;
         else
            dout = 0;
      end
endmodule
```

TOP MEALY

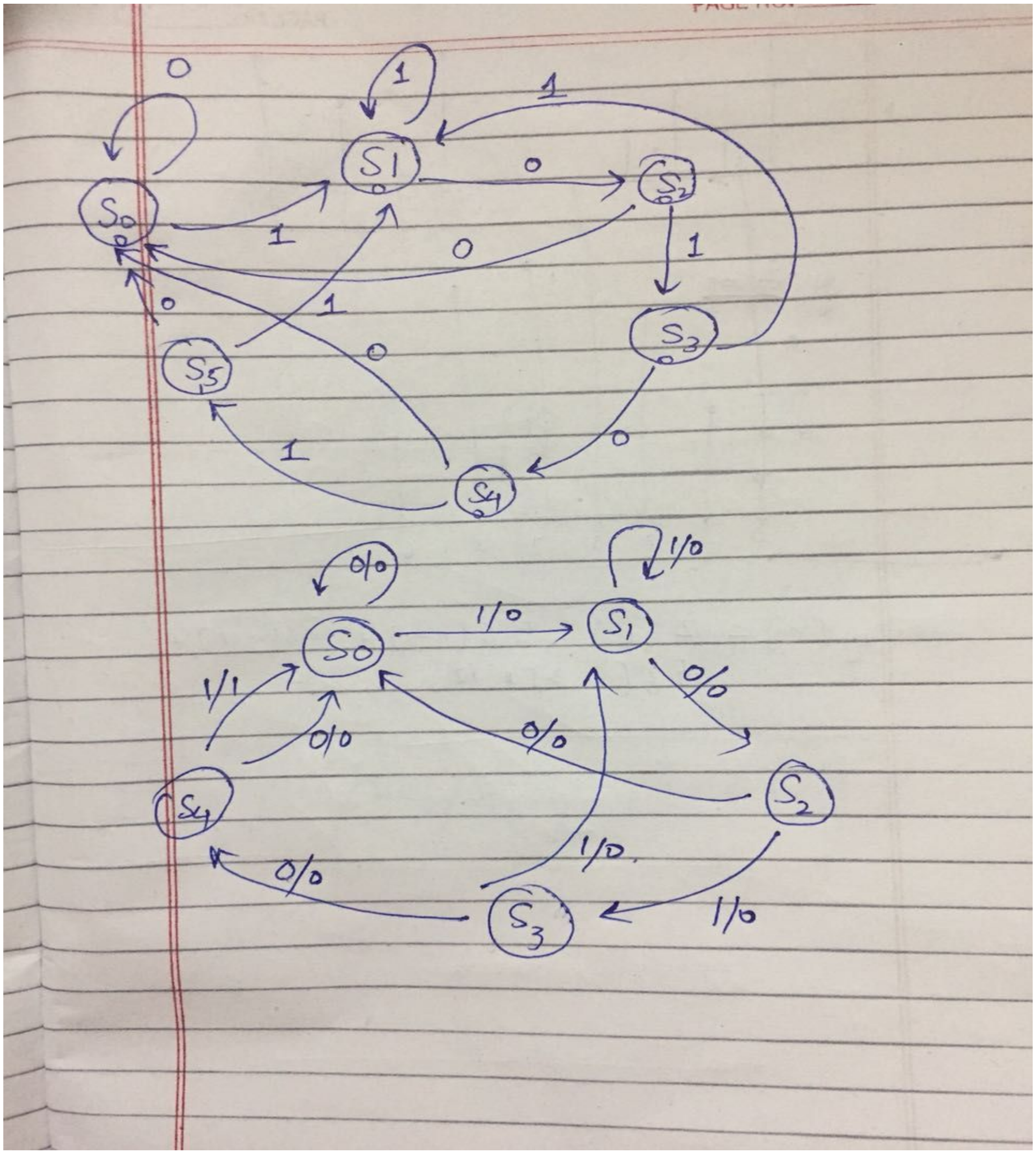
```
module top(
   input clk,
   input clr,
   input [2:0] btn,
   output led,
   output [2:0] present_state
   );

   wire inp,dout;

   assign inp = btn[0] || btn[1];

   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));
   fsm_meally_10101 meally
(.clk(out_pulse),.clr(clr_de),.din(btn[1]),.dout(led),.present_state(present_state));


endmodule
```

TOP MOORE

```
module top(
   input clk,
   input clr,
   input [1:0] btn,
   output led,
   output [2:0] present_state
   );

   wire inp,dout;

   assign inp = btn[0] || btn[1];
```

```
    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));
    FSM_moore_10101 m
(.clk(out_pulse),.clr(clr_de),.din(btn[1]),.dout(led),.present_state(present_state));


endmodule
```

FSM DIAGRAMS:


S0
S1
S2
S3
S4
S5
0
1
1/0
0/0
1/1
0/0
1/0
0/0
1/0
1/0
0/0
1/0