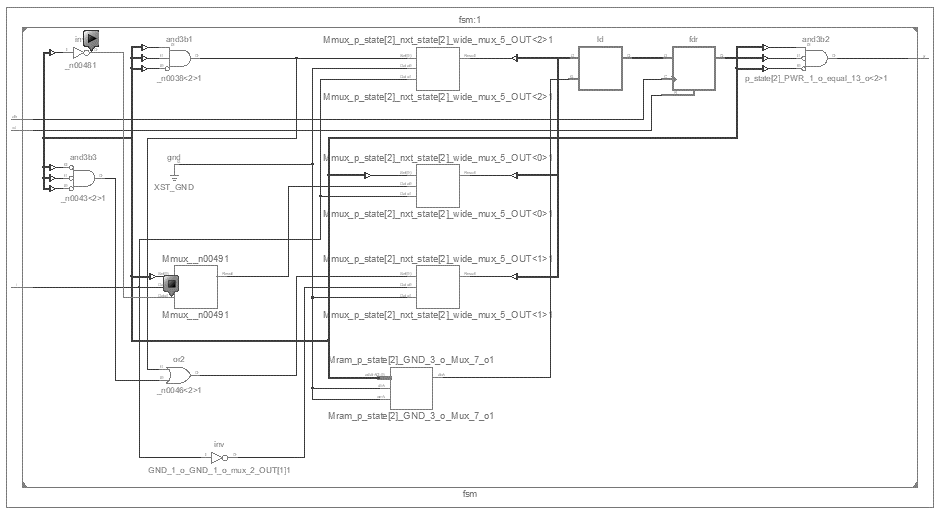
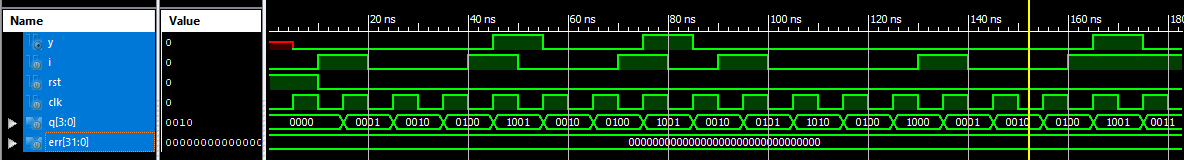
**RTL Diagram:**

****

**Output Waveform:**

****

**Experiment-5**

**Objective:**

To design a Moore FSM that detects the sequence 1001. The test bench should check the generated output with the expected output and prints pass/fail messages.

**Tool Used:**

Xilinx ISE.

**Theory:**

A FSM detects the matching of input sequence and triggers the output, a self checking test bench is stimulated using shift registers as 4 bit memory and the output is verified.

**DUT Code:**

module fsm(input i,rst,clk, output y);

    parameter idle=3'b000, s1=3'b001, s10=3'b010, s100=3'b011, s1001=3'b100;

    reg [2:0] p\_state,nxt\_state;

    always@(\*)begin

        case(p\_state)

            idle    : nxt\_state <= i ? s1   : idle;

            s1      : nxt\_state <= i ? s1   : s10   ;

            s10 : nxt\_state <= i ? s1   : s100;

            s100    : nxt\_state <= i ? s1001: idle;

            s1001 : nxt\_state <= i ? s1 : s10   ;

        endcase

    end

    always@(posedge clk)begin

        if(rst) p\_state <= idle;

        else p\_state <= nxt\_state;

    end

    assign y = (p\_state == s1001) ? 1 : 0;

endmodule

**TB Code:**

module tb;

    reg i=0,rst,clk=0;

    wire y;

    fsm uut (i,rst,clk,y);

    reg [3:0]q=0    ;

    integer err = 0;

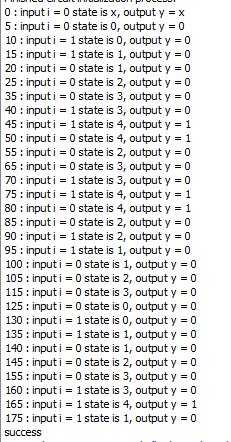
    initial forever #5 clk = !clk;

    initial forever @(posedge clk) q <= {q[2:0],i};

    initial begin

        $monitor("%0d : input i = %b state is %d, output y = %b ",$time,i,uut.p\_state,y);

**Simulation Output:**

****

rst = 1'b1;

        #10;

        rst = 1'b0;

        @(negedge clk) i = 1;

        @(negedge clk) i = 0;

        @(negedge clk) i = 0;

        @(negedge clk) i = 1;

        @(negedge clk) i = 0;

        @(negedge clk) i = 0;

        @(negedge clk) i = 1;

        @(negedge clk) i = 0;

        @(negedge clk) i = 1;

        @(negedge clk) i = 0;

        @(negedge clk) i = 0;

        @(negedge clk) i = 0;

        @(negedge clk) i = 1;

        @(negedge clk) i = 0;

        @(negedge clk) i = 0;

        @(negedge clk) i = 1;

        #100;

        if(err) $display("error"); else $display("success");

        #20;

        $finish;

    end

    always@(q)begin

        if(q == 4'b1001)begin

            @(negedge clk);

            if(!y) err = err+1;

        end

    end

endmodule

**Result:**

The simulation output and the RTL diagram is observed and found to be valid.