Design of Finite State Machine (FSM) and Algorithmic State Machine (ASM)

Objective:

To learn the design of Finite State Machine (FSM) and Algorithmic State Machine (ASM) for any application in Verilog, the simulating and synthesizing using EDA tools.

Tools Required:

1. Synthesis tool: Xilinx ISE
2. Simulation tool: Modelsim simulator

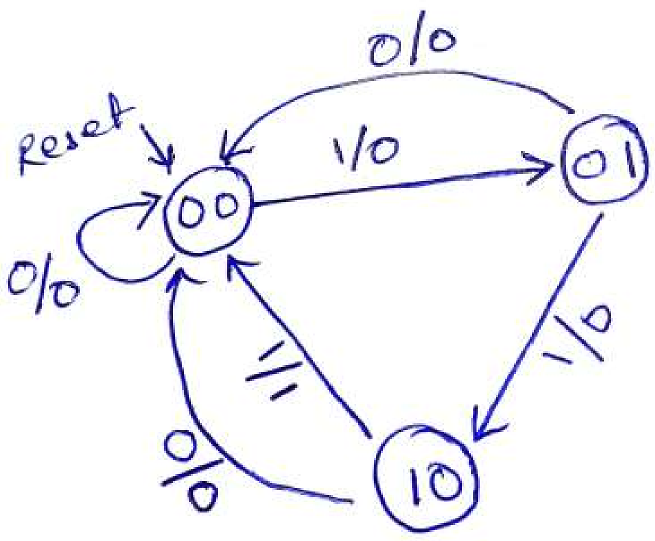
Prelab Questions:

1. Draw the simple model of FSM.
2. What is the basic Algorithm of Sequence Detector?
3. List the difference between Mealy and Moore Model.
4. What is ASM Chart and what are its main components?

Procedure:

1. Implement Sequence recognizer for detecting three successive 1's using Mealy Model (Behavioral Modeling).

Mealy model for 111 Detector (FSM):



Verilog code for Mealy Model FSM: (Behavioral Modelling)

```
module fsm_111_mealy(o,reset,a,clk);

output reg o;

input reset,a,clk;

reg[1:0]pre_s, nxt_s;

initial begin

pre_s=2’b00;

end

always@(posedge clk)

begin

if(reset==1)

begin pre_s=2’b00; o=0;end

else

begin

case(pre_s)

2’b00:begin if (a==1)begin o=0; nxt_s=2’b01;pre_s=nxt_s; end

else begin o=0; nxt_s=2’b00;pre_s=nxt_s; end end

2’b01:begin if (a==1)begin o=0; nxt_s=2’b10;pre_s=nxt_s; end

else begin o=0; nxt_s=2’b00;pre_s=nxt_s; end end

2’b10:begin if (a==1)begin o=1; nxt_s=2’b00;pre_s=nxt_s; end

else begin o=0; nxt_s=2’b00;pre_s=nxt_s; end end

default:pre_s2’b00;

endcase

end
```

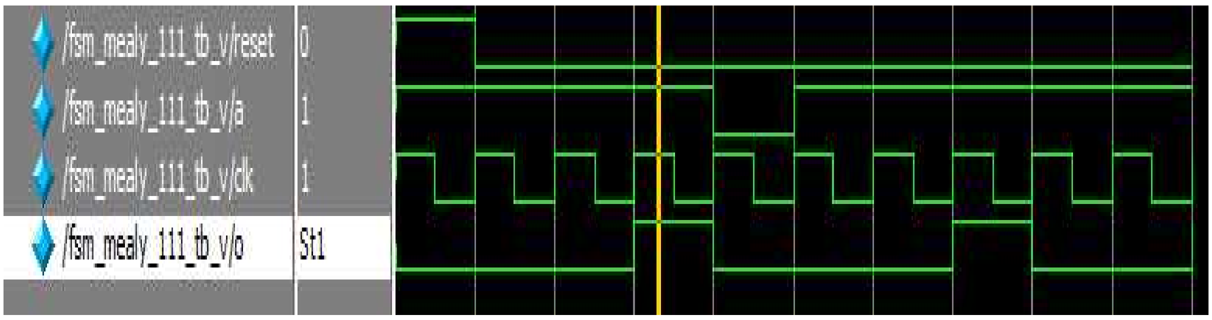
```
        end
        endmodule
```

Test Bench:

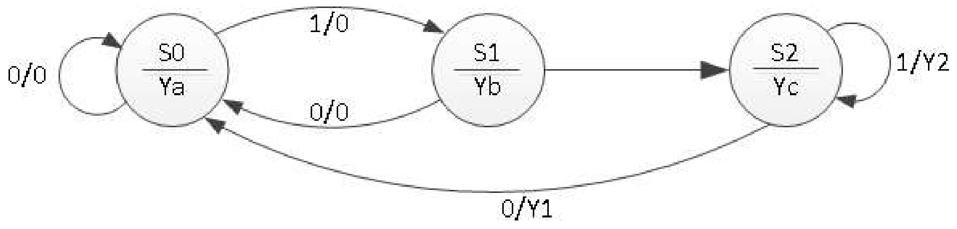
```
module fsm_mealy_111_tb_v;
// Inputs
reg reset;
reg a;
reg clk;
// Outputs
wire o;
// Instantiate the Unit Under Test (UUT)
fsm_mealy_111 uut (
.o(o),
.reset(reset),
.a(a),
.clk(clk)
);
initial begin
// Initialize Inputs
reset = 1;
a = 1;
clk = 1;
// Wait 100 ns for global reset to finish
#100;
reset = 0;
a = 1;
#100;
reset = 0;
a = 1;
#100;
reset = 0;
a = 1;
#100;
```

```
reset = 0;
a = 0;
#100;
reset = 0;
a = 1;
#100;
// Add stimulus here
end
always #50 clk=~clk;
endmodule
```

Simulation Result:

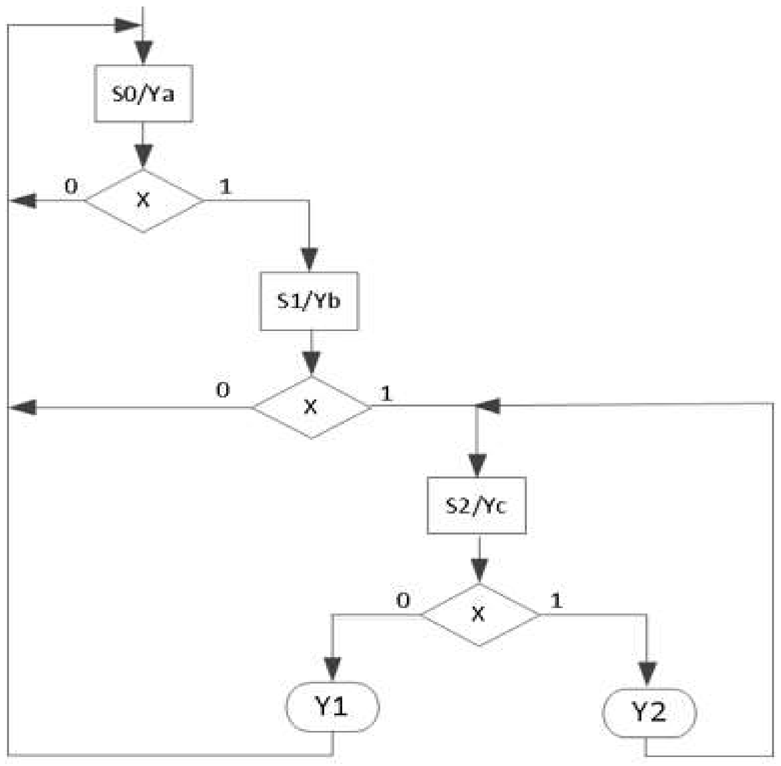


2. Consider the following state graph of a sequential network which has both Mealy and Moore outputs. The outputs Y1 and Y2 are the Mealy outputs and so should be conditional outputs. The Ya, Yb, and Yc are the Moore outputs so they should be part of state box. Input X can either be "0" or "1" and hence it should be part of the decision box.



Draw the ASM Chart for the above state graph and implement it using Verilog.

ASM Chart:



Verilog Program:

```
module asm (input clk, input x, output reg ya, output reg yb, output reg yc, output reg y1,
output reg y2);
     reg [1:0] state, nextstate;
     parameter [1:0] S0=0, S1=1, S2=2;
     always @(posedge clk)      // always block to update state
     state <= nextstate;
     always @(state or x) // always block to compute both Mealy output & next state
     begin
           y1 = 1'b0; y2 = 1'b0;
     case (state)
     S0: if(x)
          nextstate = S1;
                 else
          nextstate = S0;
     S1: if(x)
          nextstate = S2;
```

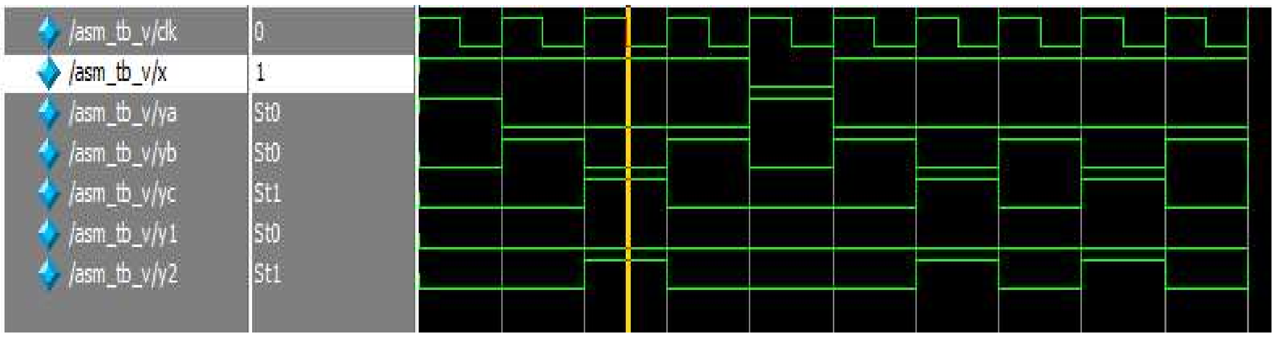
```
    else
       nextstate = S0;
S2: if(x)
begin
       y2 = 1'b1;
       nextstate = S1;
end
else
begin
        y1 = 1'b1;
        nextstate = S0;
 end
default:nextstate = S0;
endcase
end
always @(state) // always block to compute Moore output
begin
ya = 1'b0; yb = 1'b0; yc = 1'b0;
case(state)
//begin
S0: ya = 1'b1;
S1: yb = 1'b1;
 S2: yc = 1'b1;
 //end
default: begin
ya = 1'b0;
yb = 1'b0;
yc = 1'b0;
end
endcase
end
endmodule
```

Test Bench:

```
module asm_tb_v;

	// Inputs

	reg clk;

	reg x;

	// Outputs

	wire ya;

	wire yb;

	wire yc;

	wire y1;

	wire y2;

	// Instantiate the Unit Under Test (UUT)

	asm uut (

		.clk(clk),

		.x(x),

		.ya(ya),

		.yb(yb),

		.yc(yc),

		.y1(y1),

		.y2(y2)		);

	initial begin

		// Initialize Inputs

		x = 1;

		clk = 1;
```

```
                // Wait 100 ns for global reset to finish

                #100;

                x = 1;

                #100;

                x = 1;

                #100;

                x = 1;

                #100;

                x = 0;

                #100;

                // Add stimulus here

            end

  always #50 clk=~clk;

        // Add stimulus here

endmodule
```

Simulation Result:



Post lab Questions:

1. Write Verilog code to implement an FSM using Moore Machine.

Result:

All the FSM Logic and ASM logic were executed and verified.