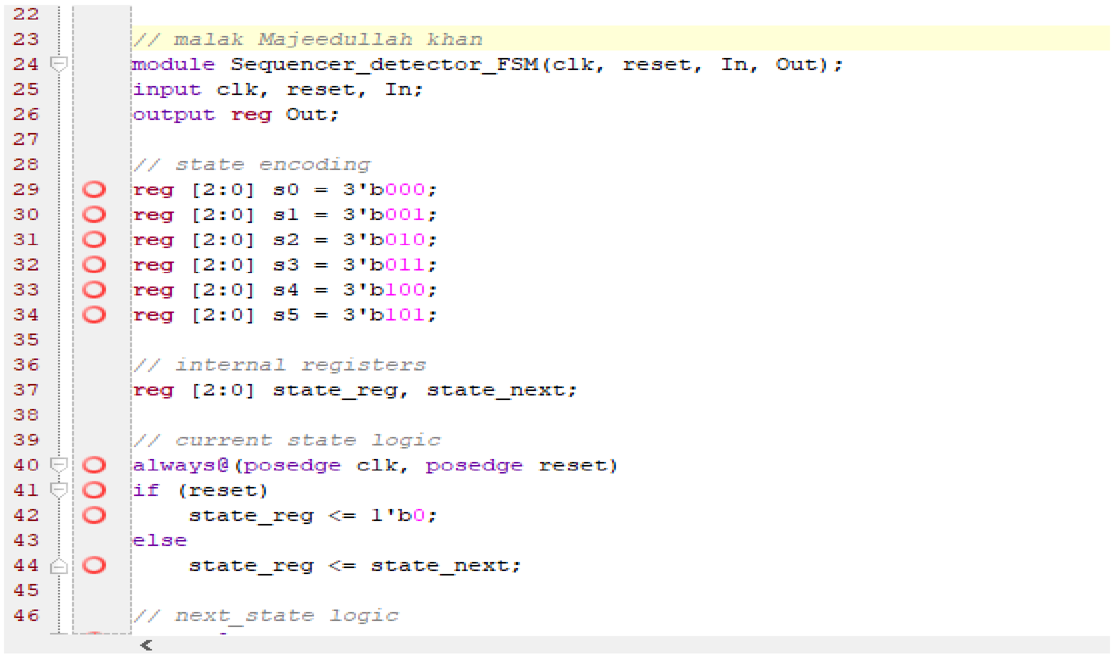
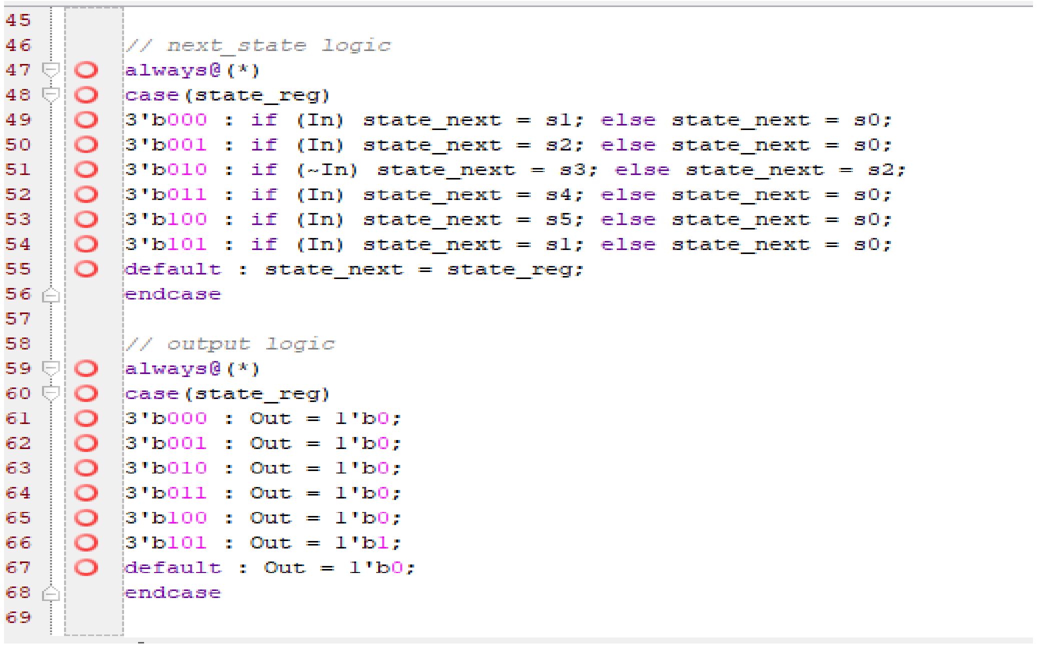
Lab 06

Design a sequence detector implementing a Mealy state machine using three always blocks. The Mealy state

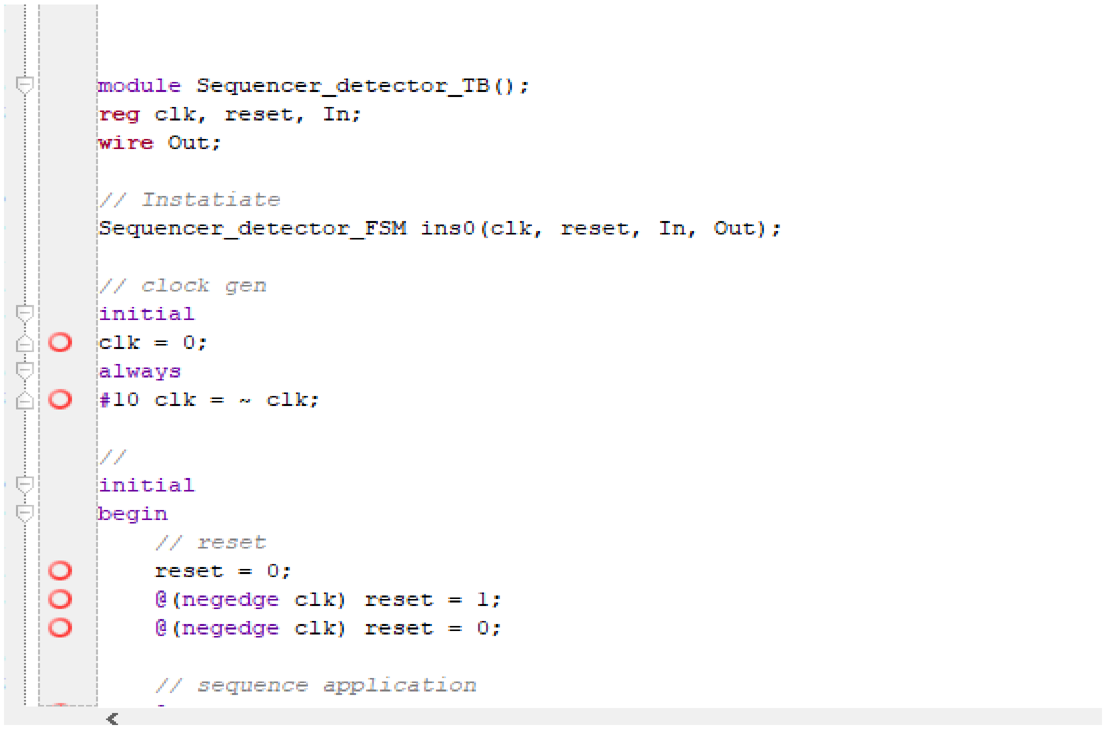


```
// malak Majeedullah khan
module Sequencer_detector_FSM(clk, reset, In, Out);
input clk, reset, In;
output reg Out;

// state encoding
reg [2:0] s0 = 3'b000;
reg [2:0] s1 = 3'b001;
reg [2:0] s2 = 3'b010;
reg [2:0] s3 = 3'b011;
reg [2:0] s4 = 3'b100;
reg [2:0] s5 = 3'b101;

// internal registers
reg [2:0] state_reg, state_next;

// current state logic
always@(posedge clk, posedge reset)
if (reset)
    state_reg <= 1'b0;
else
    state_reg <= state_next;

// next_state logic
```

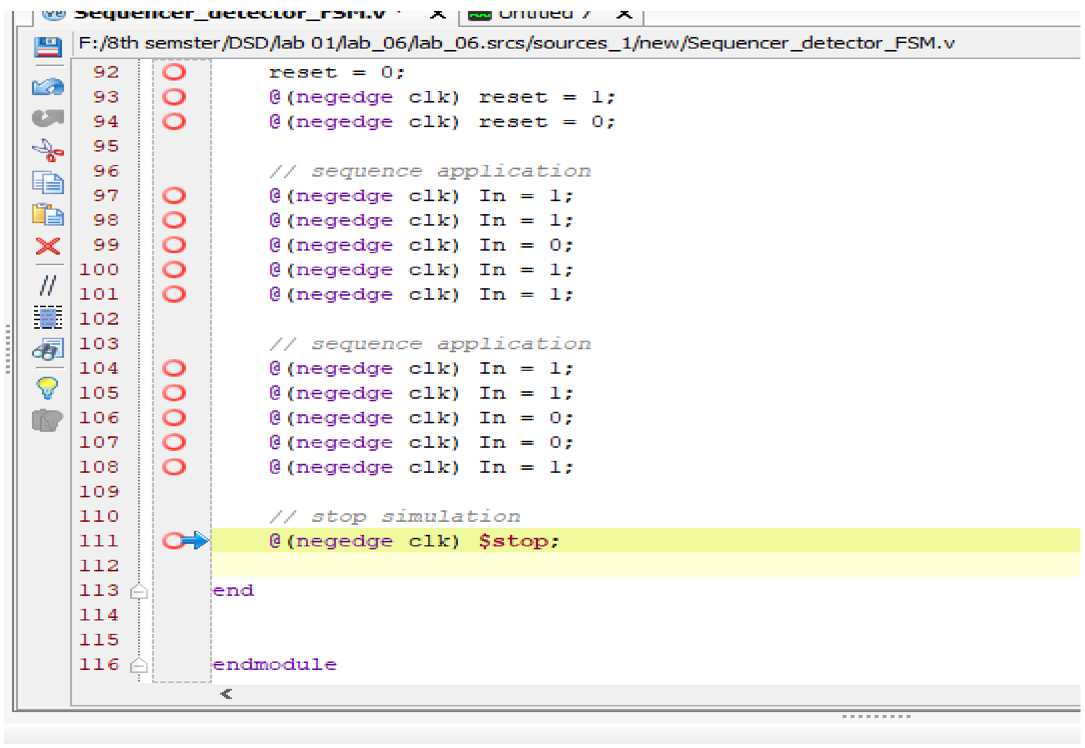


```

// next_state logic
always@(*)
case(state_reg)
3'b000 : if (In) state_next = s1; else state_next = s0;
3'b001 : if (In) state_next = s2; else state_next = s0;
3'b010 : if (~In) state_next = s3; else state_next = s2;
3'b011 : if (In) state_next = s4; else state_next = s0;
3'b100 : if (In) state_next = s5; else state_next = s0;
3'b101 : if (In) state_next = s1; else state_next = s0;
default : state_next = state_reg;
endcase

// output logic
always@(*)
case(state_reg)
3'b000 : Out = 1'b0;
3'b001 : Out = 1'b0;
3'b010 : Out = 1'b0;
3'b011 : Out = 1'b0;
3'b100 : Out = 1'b0;
3'b101 : Out = 1'b1;
default : Out = 1'b0;
endcase

```

Test bench

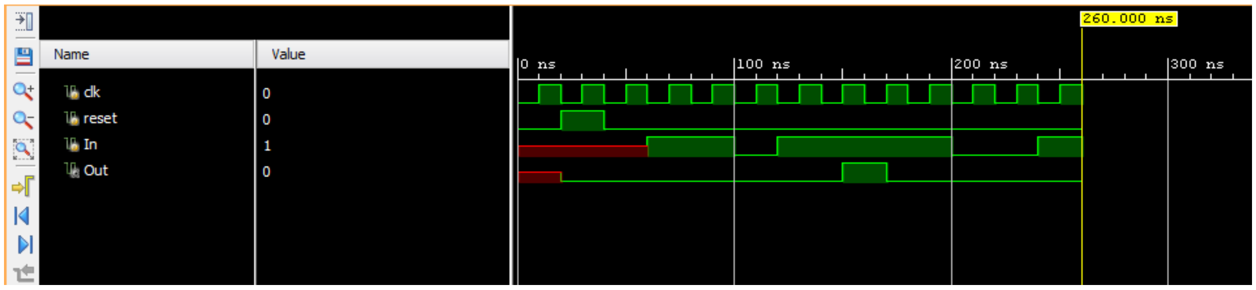


```
module Sequencer_detector_TB();
reg clk, reset, In;
wire Out;

// Instatiate
Sequencer_detector_FSM ins0(clk, reset, In, Out);

// clock gen
initial
clk = 0;
always
#10 clk = ~ clk;

//
initial
begin
    // reset
    reset = 0;
    @(negedge clk) reset = 1;
    @(negedge clk) reset = 0;

    // sequence application
```



F:/8th semster/DSD/lab 01/lab_06/lab_06.srcs/sources_1/new/Sequencer_detector_FSM.v

```
        reset = 0;
        @(negedge clk) reset = 1;
        @(negedge clk) reset = 0;

        // sequence application
        @(negedge clk) In = 1;
        @(negedge clk) In = 1;
        @(negedge clk) In = 0;
        @(negedge clk) In = 1;
        @(negedge clk) In = 1;

        // sequence application
        @(negedge clk) In = 1;
        @(negedge clk) In = 1;
        @(negedge clk) In = 0;
        @(negedge clk) In = 0;
        @(negedge clk) In = 1;

        // stop simulation
        @(negedge clk) $stop;

end


endmodule
```


260.000 ns
Name
Value
clk
0
reset
0
In
1
Out
0
0 ns
100 ns
200 ns
300 ns