IMPLEMENTATION OF SEQUENCE DETECTORS BY FINITE STATE MACHINES USING VERILOG

PABBULETI VENU

Table of Contents

SINGLE PROCEDURAL BLOCK

DESIGN MODULE

```
`timescale 1ns / 1ps
module mealy_1010_seq_det_non_over(Clk,Rst,In,OP,state);
input Clk,Rst,In; // Input signals
output OP;        // Output signal
output [1:0]state;

reg op;    // register to store output
reg [1:0]State; // State register

parameter s0   = 0;  // State registers
parameter s1   = 1;
parameter s10  = 2;
parameter s101 = 3;

always @(posedge Clk,negedge Rst) begin
     if(!Rst)
     begin
          State <= s0;
          op <= 0;
     end

     else begin
          case(State)
               s0  : begin
                          State <= In ? s1:s0 ;
                          op <= 0;
                     end

               s1  : begin
```

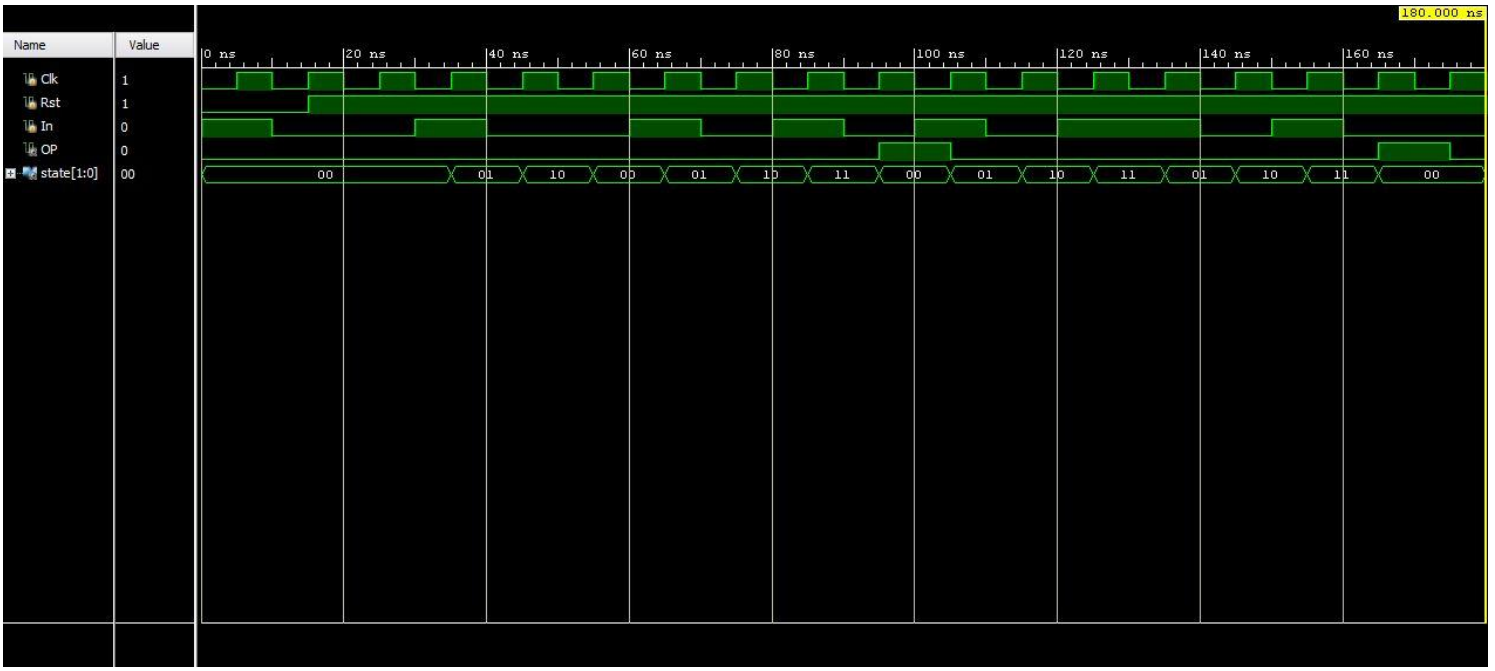
```
                        State <= In ? s1:s10 ;
                        op <= 0;
                    end

                s10 : begin
                        State <= In ? s101:s0 ;
                        op <= 0;
                    end

                s101: begin
                        State <= In ? s1:s0 ;
                        op <= In ? 0:1 ;
                    end

                default : begin
                            State <= s0 ;
                            op <= 0;
                            end
            endcase
        end

end

assign OP = op;
assign state = State;

endmodule
```

TESTBENCH MODULE

```
`timescale 1ns / 1ps
module mealy_1010_seq_det_non_over_tb ();
// Signal Declarations
reg Clk,Rst,In;
wire OP;
wire [1:0]state;
//instantiating design to test
mealy_1010_seq_det_non_over UUT(Clk,Rst,In,OP,state);

// Monitoring the simulation results
initial begin
     $monitor("Input = %b || State = %b || Output = %b",In,state,OP);
end

// clock signal generation
initial begin
     Clk = 1'b0;
     forever #5 Clk = ~Clk;
end

// input signals
initial begin
          Rst = 1'b0;
     #15  Rst = 1'b1; end

initial begin
         In  = 1'b1;
     #10 In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #10 In  = 1'b0;
     #10 In  = 1'b0;
```

```
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b0;
        #10 $finish();
end


endmodule
```

SIMULATION WAVEFORMS



SIMULATION OUTPUTS

```
Input = 1 || State = 00 || Output = 0
Input = 0 || State = 00 || Output = 0
Input = 1 || State = 00 || Output = 0
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 0 || State = 00 || Output = 0
Input = 1 || State = 00 || Output = 0
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 1 || State = 10 || Output = 0
Input = 1 || State = 11 || Output = 0
Input = 0 || State = 11 || Output = 0
Input = 0 || State = 00 || Output = 1
Input = 1 || State = 00 || Output = 1
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 1 || State = 10 || Output = 0
Input = 1 || State = 11 || Output = 0
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 1 || State = 10 || Output = 0
Input = 1 || State = 11 || Output = 0
Input = 0 || State = 11 || Output = 0
Input = 0 || State = 00 || Output = 1
Input = 0 || State = 00 || Output = 0
```

TWO PROCEDURAL BLOCK(METHOD-01)

DESIGN MODULE

```
`timescale 1ns / 1ps

module mealy_1010_seq_det_non_over(Clk,Rst,In,OP);
input Clk,Rst,In; // Input signals
output OP;        // Output signal

reg op;    // register to store output
reg [1:0]C_State,N_State; // state register

parameter   s0   = 0,  // state registers
            s1   = 1,
            s10  = 2,
            s101 = 3;



always @(posedge Clk,negedge Rst) begin
     if(!Rst)
     begin
          C_State <= s0;
          op <= 0;
     end

     else begin
        C_State <= N_State;
     end
end

always @(C_State,In) begin

          case(C_State)
```

```
                s0   :     begin N_State <= In ? s1:s0 ; op <= 0 ; end

                s1   :     begin N_State <= In ? s1:s10 ; op <= 0 ; end

                s10  :     begin N_State <= In ? s101:s0 ; op <= 0 ; end

               s101  :     begin  N_State <= In ? s1:s0 ; op <= In ? 0:1 ;
end

               default : begin N_State <= s0; op <= 0; end

          endcase

     end

assign OP = op;

endmodule
```

TESTBENCH MODULE

```
`timescale 1ns / 1ps

module mealy_1010_seq_det_non_over_tb ();
// Signal Declarations
reg Clk,Rst,In;
wire OP;

//instantiating design to test
mealy_1010_seq_det_non_over UUT(Clk,Rst,In,OP);

// Monitoring the simulation results
initial begin
```

```
      $monitor("CS = %d ||Input = %b || NS = %b || Output
= %b",UUT.C_State,In,UUT.N_State,OP);
end

// clock signal generation
initial begin
      Clk = 1'b0;
      forever #5 Clk = ~Clk;
end


// input signals
initial begin
          Rst = 1'b0;
      #15 Rst = 1'b1;
end


initial begin
      #10 In  = 1'b0;
      #10 In  = 1'b0;
      #10 In  = 1'b1;
      #10 In  = 1'b0;
      #10 In  = 1'b0;
      #10 In  = 1'b1;
      #10 In  = 1'b0;
      #10 In  = 1'b1;
      #7  In  = 1'b0;
      #13 In  = 1'b1;
      #8  In  = 1'b0;
      #10 In  = 1'b1;
      #10 In  = 1'b1;
      #10 In  = 1'b0;
      #10 In  = 1'b1;
      #10 In  = 1'b0;
```

```
        #10 In  = 1'b0;
        #10 $finish();
end


endmodule
```

SIMULATION WAVEFORMS

SIMULATION OUTPUTS

```
Input = x || State = 00 || Output = 0
Input = 0 || State = 00 || Output = 0
Input = 1 || State = 00 || Output = 0
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 0 || State = 00 || Output = 0
Input = 1 || State = 00 || Output = 0
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 1 || State = 10 || Output = 0
Input = 1 || State = 11 || Output = 0
Input = 0 || State = 11 || Output = 1
Input = 0 || State = 00 || Output = 0
Input = 1 || State = 00 || Output = 0
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 1 || State = 10 || Output = 0
Input = 1 || State = 11 || Output = 0
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 1 || State = 10 || Output = 0
Input = 1 || State = 11 || Output = 0
Input = 0 || State = 11 || Output = 1
Input = 0 || State = 00 || Output = 0
```

TWO PROCEDURAL BLOCK(METHOD-02)

DESIGN MODULE

```
`timescale 1ns / 1ps
module mealy_1010_seq_det_non_over(Clk,Rst,In,OP,sta);
input Clk,Rst,In; // Input signals
output OP;        // Output signal
output [1:0]sta;

reg op;    // register to store output
reg [1:0]state; // state register

parameter   s0   = 0,  // state registers
            s1   = 1,
            s10  = 2,
            s101 = 3;

reg [1:0]State;

always @(posedge Clk,negedge Rst) begin
     if(!Rst)
     begin
          State <= s0;
          op <= 0;
     end

     else begin

          case(State)

               s0  :     State <= In ? s1:s0 ;

               s1  :     State <= In ? s1:s10 ;
```

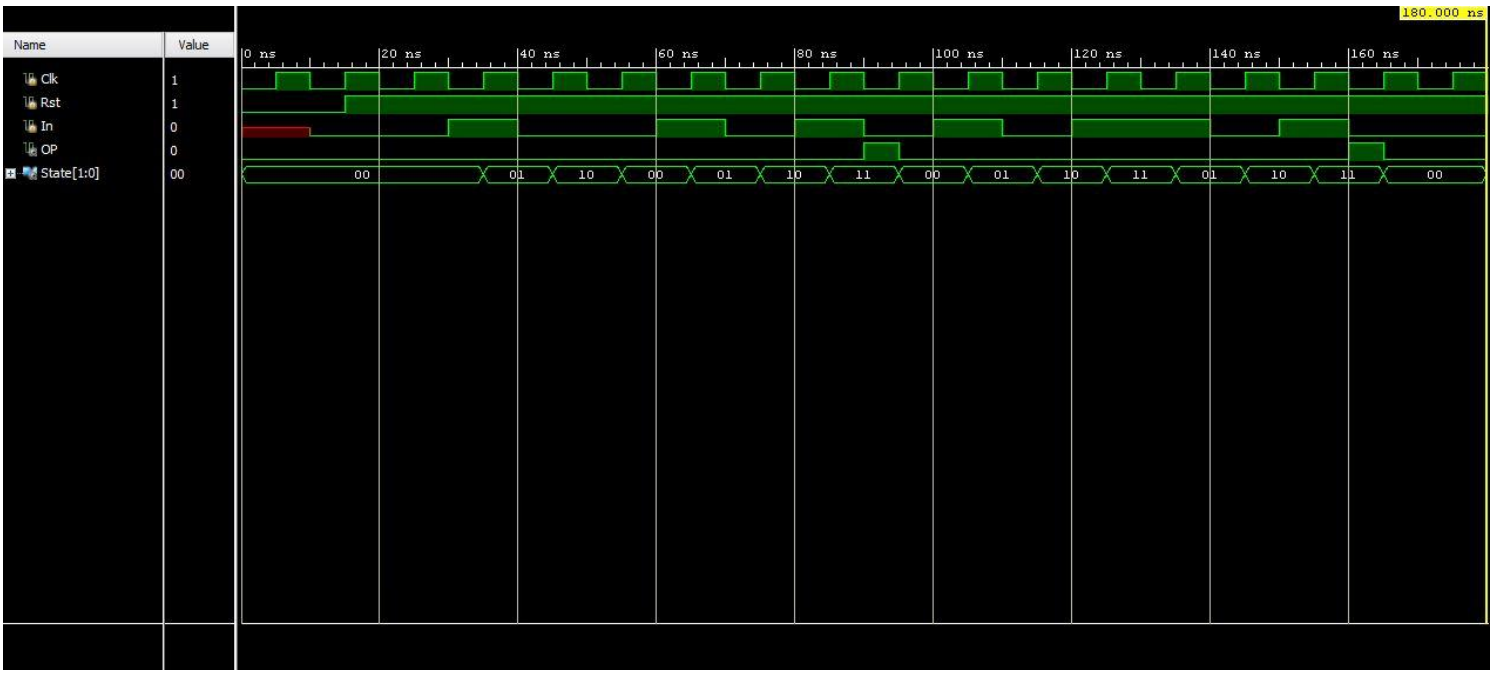
```
                s10 :     State <= In ? s101:s0 ;

               s101 :     State <= In ? s1:s0 ;

               default :  State <= s0;

          endcase

     end
end

always @(State,In) begin

     case(State)
          s0   :     op <= 0;

          s1   :     op <= 0;

          s10 :    op <= 0;

          s101:      op <= In ? 0:1 ;

          default : op <= 0;

     endcase

end

assign OP = op;
assign sta = State;

endmodule
```

TESTBENCH MODULE

```
`timescale 1ns / 1ps
module mealy_1010_seq_det_non_over_tb ();
// Signal Declarations
reg Clk,Rst,In;
wire OP;
wire [1:0]State;
//instantiating design to test
mealy_1010_seq_det_non_over UUT(Clk,Rst,In,OP,State);

// Monitoring the simulation results
initial begin
     $monitor("Input = %b || State = %b || Output = %b",In,State,OP);
end

// clock signal generation
initial begin
     Clk = 1'b0;
     forever #5 Clk = ~Clk;
end

// input signals
initial begin
          Rst = 1'b0;
     #15 Rst = 1'b1; end


initial begin
     #10 In  = 1'b0;
     #10 In    = 1'b0;
     #10 In  = 1'b1;
     #10 In    = 1'b0;
     #10 In  = 1'b0;
     #10 In    = 1'b1;
     #10 In  = 1'b0;
```

```
        #10 In     = 1'b1;
        #10 In   = 1'b0;
        #10 In     = 1'b1;
        #10 In   = 1'b0;
        #10 In   = 1'b1;
        #10 In     = 1'b1;
        #10 In   = 1'b0;
        #10 In     = 1'b1;
        #10 In   = 1'b0;
        #10 In     = 1'b0;
        #10 $finish();
end

endmodule
```

SIMULATION WAVEFORMS



SIMULATION OUTPUTS

```
Input = x || State = 00 || Output = 0
Input = 0 || State = 00 || Output = 0
Input = 1 || State = 00 || Output = 0
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 0 || State = 00 || Output = 0
Input = 1 || State = 00 || Output = 0
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 1 || State = 10 || Output = 0
Input = 1 || State = 11 || Output = 0
Input = 0 || State = 11 || Output = 1
Input = 0 || State = 00 || Output = 0
Input = 1 || State = 00 || Output = 0
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 1 || State = 10 || Output = 0
Input = 1 || State = 11 || Output = 0
Input = 1 || State = 01 || Output = 0
Input = 0 || State = 01 || Output = 0
Input = 0 || State = 10 || Output = 0
Input = 1 || State = 10 || Output = 0
Input = 1 || State = 11 || Output = 0
Input = 0 || State = 11 || Output = 1
Input = 0 || State = 00 || Output = 0
```

THREE PROCEDURAL BLOCK

DESIGN MODULE

```
`timescale 1ns / 1ps
module mealy_1010_seq_det_non_over(Clk,Rst,In,OP,cs,ns);
input Clk,Rst,In; // Input signals
output OP;        // Output signal
output [1:0]cs,ns;

reg op;    // register to store output
reg state; // state register

parameter   s0   = 0,  // state registers
            s1   = 1,
            s10  = 2,
            s101 = 3;

reg [1:0]C_State,N_State; // state declarations

// State register

always @(posedge Clk,negedge Rst) begin

     if(!Rst)
     begin
          C_State <= s0;
          op <= 0;
     end

     else begin
          C_State <= N_State;
     end
end
// Next state logic
```

```
always @(C_State,In) begin

     case(C_State)

          s0   : N_State <= In ? s1:s0 ;

          s1   : N_State <= In ? s1:s10 ;

          s10  : N_State <= In ? s101:s0 ;

          s101: N_State <= In ? s1:s0 ;

          default : N_State <= s0;

     endcase
end

// Output logic

always @(C_State,In) begin
     case(C_State)

          s0   : op <= 0;

          s1   : op <= 0;

          s10  : op <= 0;

          s101: op <= In ? 0:1 ;

          default : op <= 0;

     endcase
end
```

```
//
assign OP = op;
assign cs = C_State;
assign ns = N_State;

endmodule
```

TESTBENCH MODULE

```
`timescale 1ns / 1ps
module mealy_1010_seq_det_non_over_tb ();
// Signal Declarations
reg Clk,Rst,In;
wire OP;
wire [1:0]CS,NS;
//instantiating design to test
mealy_1010_seq_det_non_over UUT(Clk,Rst,In,OP,CS,NS);

// Monitoring the simulation results
initial begin
     $monitor("Current State = %b || Input = %b || Next State = %b ||
Output = %b",UUT.C_State,In,UUT.N_State,OP);
end

// clock signal generation
initial begin
     Clk = 1'b0;
     forever #5 Clk = ~Clk;
end

// input signals
initial begin
          Rst = 1'b0;
```

```
      #15 Rst = 1'b1; end
initial begin
      #10 In  = 1'b0;
      #10 In  = 1'b0;
      #10 In  = 1'b1;
      #10 In  = 1'b0;
      #10 In  = 1'b0;
      #10 In  = 1'b1;
      #10 In  = 1'b0;
      #10 In  = 1'b1;
      #7  In  = 1'b0;
      #13 In  = 1'b1;
      #8  In  = 1'b0;
      #10 In  = 1'b1;
      #10 In  = 1'b1;
      #10 In  = 1'b0;
      #10 In  = 1'b1;
      #10 In  = 1'b0;
      #10 In  = 1'b0;
      #10 $finish();
end


endmodule
```

SIMULATION WAVEFORMS

SIMULATION OUTPUTS

```
Current State = 00 || Input = x || Next State = 0x || Output = 0
Current State = 00 || Input = 0 || Next State = 00 || Output = 0
Current State = 00 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 0 || Next State = 10 || Output = 0
Current State = 10 || Input = 0 || Next State = 00 || Output = 0
Current State = 00 || Input = 0 || Next State = 00 || Output = 0
Current State = 00 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 0 || Next State = 10 || Output = 0
Current State = 10 || Input = 0 || Next State = 00 || Output = 0
Current State = 10 || Input = 1 || Next State = 11 || Output = 0
Current State = 11 || Input = 1 || Next State = 01 || Output = 0
Current State = 11 || Input = 0 || Next State = 00 || Output = 1
Current State = 00 || Input = 0 || Next State = 00 || Output = 0
Current State = 00 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 0 || Next State = 10 || Output = 0
Current State = 10 || Input = 0 || Next State = 00 || Output = 0
Current State = 10 || Input = 1 || Next State = 11 || Output = 0
Current State = 11 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 0 || Next State = 10 || Output = 0
Current State = 10 || Input = 0 || Next State = 00 || Output = 0
Current State = 10 || Input = 1 || Next State = 11 || Output = 0
Current State = 11 || Input = 1 || Next State = 01 || Output = 0
Current State = 11 || Input = 0 || Next State = 00 || Output = 1
Current State = 00 || Input = 0 || Next State = 00 || Output = 0
```

SEQUENCE DETECTOR 1010 USING MEALY MACHINE (NON OVERLAPPING)

DESIGN MODULE

```
`timescale 1ns / 1ps
module mealy_1010_seq_det_non_over(Clk,Rst,In,OP,cs,ns);
input Clk,Rst,In; // Input signals
output OP;        // Output signal
output [1:0]cs,ns; // To represent the present and next states


parameter   s0   = 0,  // state registers
            s1   = 1,
            s10  = 2,
            s101 = 3;

reg [1:0]C_State,N_State; // state declarations

// State register

always @(posedge Clk,negedge Rst) begin

     if(!Rst)
     begin
          C_State <= s0;
     end

     else begin
          C_State <= N_State;
     end
end


// Next state logic
```

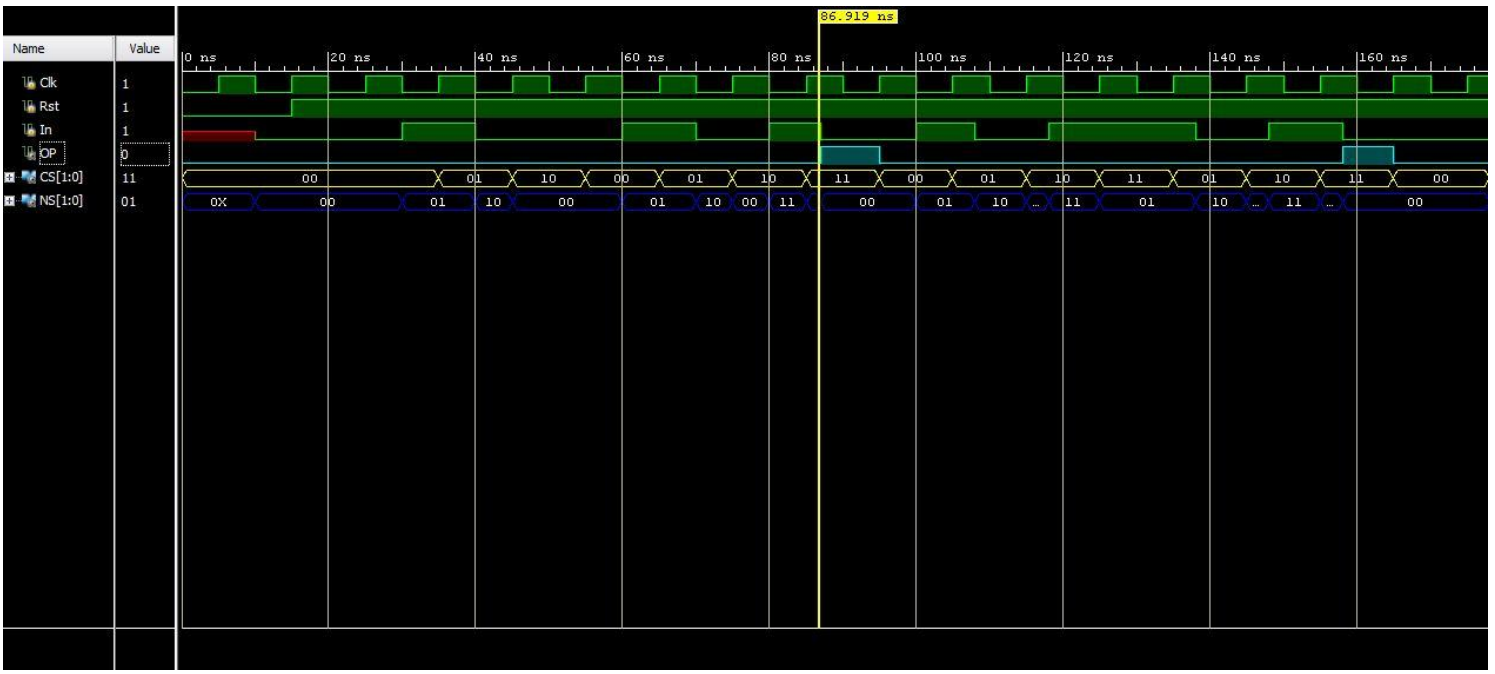
```
always @(C_State,In) begin

     case(C_State)

          s0  : N_State <= In ? s1:s0 ;

          s1  : N_State <= In ? s1:s10 ;

          s10 : N_State <= In ? s101:s0 ;

          s101: N_State <= In ? s1:s0 ;

          default : N_State <= s0;

     endcase
end

// Output logic

assign OP = ((C_State == s101 ) && (!In) );




assign cs = C_State;
assign ns = N_State;

endmodule
```

TESTBENCH MODULE

```
`timescale 1ns / 1ps
module mealy_1010_seq_det_non_over_tb ();
// Signal Declarations
reg Clk,Rst,In;
wire OP;
wire [1:0]CS,NS;
//instantiating design to test
mealy_1010_seq_det_non_over UUT(Clk,Rst,In,OP,CS,NS);

// Monitoring the simulation results
initial begin
     $monitor("Current State = %b || Input = %b || Next State = %b ||
Output = %b",UUT.C_State,In,UUT.N_State,OP);
end

// clock signal generation
initial begin
     Clk = 1'b0;
     forever #5 Clk = ~Clk;
end

// input signals
initial begin
          Rst = 1'b0;
     #15 Rst = 1'b1; end
initial begin
     #10 In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #10 In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #10 In  = 1'b0;
```

```
        #10 In  = 1'b1;
        #7  In  = 1'b0;
        #13 In  = 1'b1;
        #8  In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b0;
        #10 $finish();
end

endmodule
```

SIMULATION WAVEFORMS



SIMULATION OUTPUTS

```
Current State = 00 || Input = x || Next State = 0x || Output = 0
Current State = 00 || Input = 0 || Next State = 00 || Output = 0
Current State = 00 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 0 || Next State = 10 || Output = 0
Current State = 10 || Input = 0 || Next State = 00 || Output = 0
Current State = 00 || Input = 0 || Next State = 00 || Output = 0
Current State = 00 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 0 || Next State = 10 || Output = 0
Current State = 10 || Input = 0 || Next State = 00 || Output = 0
Current State = 10 || Input = 1 || Next State = 11 || Output = 0
Current State = 11 || Input = 1 || Next State = 01 || Output = 0
Current State = 11 || Input = 0 || Next State = 00 || Output = 1
Current State = 00 || Input = 0 || Next State = 00 || Output = 0
Current State = 00 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 0 || Next State = 10 || Output = 0
Current State = 10 || Input = 0 || Next State = 00 || Output = 0
Current State = 10 || Input = 1 || Next State = 11 || Output = 0
Current State = 11 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 1 || Next State = 01 || Output = 0
Current State = 01 || Input = 0 || Next State = 10 || Output = 0
Current State = 10 || Input = 0 || Next State = 00 || Output = 0
Current State = 10 || Input = 1 || Next State = 11 || Output = 0
Current State = 11 || Input = 1 || Next State = 01 || Output = 0
Current State = 11 || Input = 0 || Next State = 00 || Output = 1
Current State = 00 || Input = 0 || Next State = 00 || Output = 0
```

SEQUENCE DETECTOR 1010 USING MEALYMACHINE (OVERLAPPING)

DESIGN MODULE

```
`timescale 1ns / 1ps
module mealy_1010_seq_det_over(Clk,Rst,In,OP,CS,NS);
input Clk,Rst,In; // Input signals
output OP;        // Output signal
output [1:0]CS,NS;

parameter   s0   = 0,  // state registers
            s1   = 1,
            s10  = 2,
            s101 = 3;

reg [1:0]C_State,N_State; // state declarations

// State register

always @(posedge Clk,negedge Rst) begin

     if(!Rst)
     begin
          C_State <= s0;
     end

     else begin
          C_State <= N_State;
     end
end
```

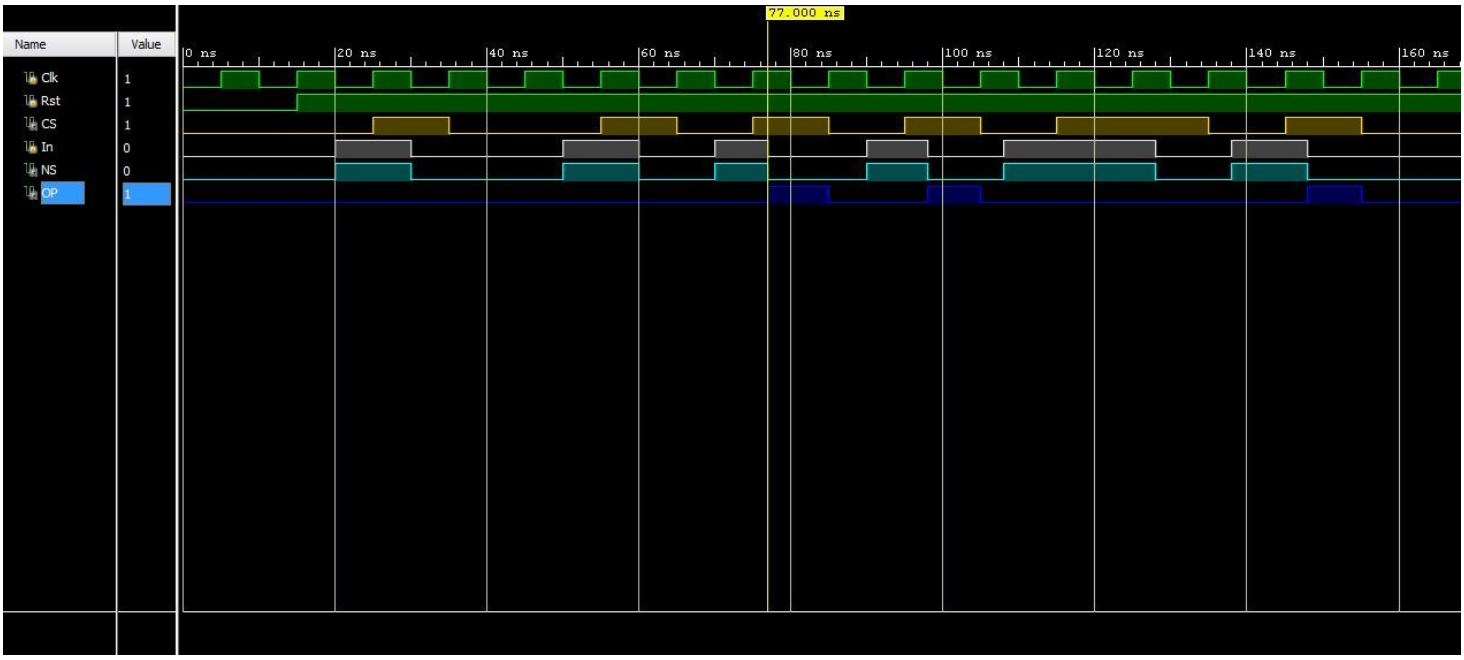
```
// Next state logic

always @(C_State,In) begin

     case(C_State)

          s0  : N_State <= In ? s1:s0 ;

          s1  : N_State <= In ? s1:s10 ;

          s10 : N_State <= In ? s101:s0 ;

          s101: N_State <= In ? s1:s10 ;

          default : N_State <= s0;

     endcase
end


// Output logic

assign OP = ((C_State == s101) && (!In));


//states
assign CS = C_State ;
assign NS = N_State ;

endmodule
```

TESTBENCH MODULE

```
`timescale 1ns / 1ps
module mealy_1010_seq_det_over_tb ();
// Signal Declarations
reg Clk,Rst,In;
wire OP;
wire CS,NS;

//instantiating design to test
mealy_1010_seq_det_over UUT(Clk,Rst,In,OP,CS,NS);

// Monitoring the simulation results
initial begin
     $monitor("Time = %0t ||Current State = %b || Input = %b || Next State
= %b || Output = %b",$time,CS,In,NS,OP);
end

// clock signal generation
initial begin
     Clk = 1'b0;
     forever #5 Clk = ~Clk;
end

// input signals
initial begin
          Rst = 1'b0;
     #15 Rst = 1'b1; end

initial begin
          In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #10 In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
```

```
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #7  In  = 1'b0;
        #13 In  = 1'b1;
        #8  In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b0;
        #10 $finish();
end

endmodule
```

SIMULATION WAVEFORMS



SIMULATION OUTPUTS

```
Time = 0 ||Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Time = 20000 ||Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Time = 25000 ||Current State = 1 || Input = 1 || Next State = 1 || Output = 0
Time = 30000 ||Current State = 1 || Input = 0 || Next State = 0 || Output = 0
Time = 35000 ||Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Time = 50000 ||Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Time = 55000 ||Current State = 1 || Input = 1 || Next State = 1 || Output = 0
Time = 60000 ||Current State = 1 || Input = 0 || Next State = 0 || Output = 0
Time = 65000 ||Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Time = 70000 ||Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Time = 75000 ||Current State = 1 || Input = 1 || Next State = 1 || Output = 0
Time = 77000 ||Current State = 1 || Input = 0 || Next State = 0 || Output = 1
Time = 85000 ||Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Time = 90000 ||Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Time = 95000 ||Current State = 1 || Input = 1 || Next State = 1 || Output = 0
Time = 98000 ||Current State = 1 || Input = 0 || Next State = 0 || Output = 1
Time = 105000 ||Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Time = 108000 ||Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Time = 115000 ||Current State = 1 || Input = 1 || Next State = 1 || Output = 0
Time = 128000 ||Current State = 1 || Input = 0 || Next State = 0 || Output = 0
Time = 135000 ||Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Time = 138000 ||Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Time = 145000 ||Current State = 1 || Input = 1 || Next State = 1 || Output = 0
Time = 148000 ||Current State = 1 || Input = 0 || Next State = 0 || Output = 1
Time = 155000 ||Current State = 0 || Input = 0 || Next State = 0 || Output = 0
```

SEQUENCE DETECTOR 1010 USING MOORE MACHINE (NON OVERLAPPING)

DESIGN MOUDLE

```
`timescale 1ns / 1ps
module moore_1010_seq_det_non_over(Clk,Rst,In,OP,CS,NS);
input Clk,Rst,In; // Input signals
output OP;        // Output signal
output [2:0]CS,NS;  // states

parameter   s0    = 0 ,  // state registers
            s1    = 1 ,
            s10   = 2 ,
            s101  = 3 ,
            s1010 = 4 ;

reg [2:0]C_State,N_State; // state declarations

// State register

always @(posedge Clk,negedge Rst) begin

     if(!Rst)
     begin
          C_State <= s0;
     end

     else begin
          C_State <= N_State;
     end
end
```

```
// Next state logic

always @(C_State,In) begin

     case(C_State)

          s0    : N_State <= In ? s1:s0 ;

          s1    : N_State <= In ? s1:s10 ;

          s10   : N_State <= In ? s101:s0 ;

          s101  : N_State <= In ? s1:s1010 ;

          s1010 : N_State <= In ? s1 : s0 ;

          default : N_State <= s0;

     endcase
end

// Output logic

assign OP = (C_State == s1010) ;



//
assign CS = C_State ;
assign NS = N_State ;

endmodule
```

TESTBENCH MODULE

```
`timescale 1ns / 1ps
module moore_1010_seq_det_non_over_tb ();
// Signal Declarations
reg Clk,Rst,In;
wire OP;
wire [2:0] CS,NS;

//instantiating design to test
moore_1010_seq_det_non_over UUT(Clk,Rst,In,OP,CS,NS);

// Monitoring the simulation results
initial begin
     $monitor("Time = %0t || Current State = %b || Input = %b || Next
State = %b || Output = %b",$time,CS,In,NS,OP);
end

// clock signal generation
initial begin
     Clk = 1'b0;
     forever #5 Clk = ~Clk;
end

// input signals
initial begin
         Rst = 1'b0;
     #15 Rst = 1'b1; end

initial begin
         In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #10 In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
```

```
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #7  In  = 1'b0;
        #13 In  = 1'b1;
        #8  In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b0;
        #10 $finish();
end


endmodule
```

SIMULATION WAVEFORMS

SIMULATION OUTPUTS

```
Time = 0 || Current State = 000 || Input = 0 || Next State = 000 || Output = 0
Time = 20000 || Current State = 000 || Input = 1 || Next State = 001 || Output = 0
Time = 25000 || Current State = 001 || Input = 1 || Next State = 001 || Output = 0
Time = 30000 || Current State = 001 || Input = 0 || Next State = 010 || Output = 0
Time = 35000 || Current State = 010 || Input = 0 || Next State = 000 || Output = 0
Time = 45000 || Current State = 000 || Input = 0 || Next State = 000 || Output = 0
Time = 50000 || Current State = 000 || Input = 1 || Next State = 001 || Output = 0
Time = 55000 || Current State = 001 || Input = 1 || Next State = 001 || Output = 0
Time = 60000 || Current State = 001 || Input = 0 || Next State = 010 || Output = 0
Time = 65000 || Current State = 010 || Input = 0 || Next State = 000 || Output = 0
Time = 70000 || Current State = 010 || Input = 1 || Next State = 011 || Output = 0
Time = 75000 || Current State = 011 || Input = 1 || Next State = 001 || Output = 0
Time = 77000 || Current State = 011 || Input = 0 || Next State = 100 || Output = 0
Time = 85000 || Current State = 100 || Input = 0 || Next State = 000 || Output = 1
Time = 90000 || Current State = 100 || Input = 1 || Next State = 001 || Output = 1
Time = 95000 || Current State = 001 || Input = 1 || Next State = 001 || Output = 0
Time = 98000 || Current State = 001 || Input = 0 || Next State = 010 || Output = 0
Time = 105000 || Current State = 010 || Input = 0 || Next State = 000 || Output = 0
Time = 108000 || Current State = 010 || Input = 1 || Next State = 011 || Output = 0
Time = 115000 || Current State = 011 || Input = 1 || Next State = 001 || Output = 0
Time = 125000 || Current State = 001 || Input = 1 || Next State = 001 || Output = 0
Time = 128000 || Current State = 001 || Input = 0 || Next State = 010 || Output = 0
Time = 135000 || Current State = 010 || Input = 0 || Next State = 000 || Output = 0
Time = 138000 || Current State = 010 || Input = 1 || Next State = 011 || Output = 0
Time = 145000 || Current State = 011 || Input = 1 || Next State = 001 || Output = 0
Time = 148000 || Current State = 011 || Input = 0 || Next State = 100 || Output = 0
Time = 155000 || Current State = 100 || Input = 0 || Next State = 000 || Output = 1
Time = 165000 || Current State = 000 || Input = 0 || Next State = 000 || Output = 0
```

SEQUENCE DETECTOR 1010 USING MOORE MACHINE (OVERLAPPING)

DESIGN MODULE

```
`timescale 1ns / 1ps
module moore_1010_seq_det_over(Clk,Rst,In,OP,CS,NS);
input Clk,Rst,In; // Input signals
output OP;        // Output signal
output [2:0]CS,NS ;


parameter   s0    = 0 ,  // state registers
            s1    = 1 ,
            s10   = 2 ,
            s101  = 3 ,
            s1010 = 4 ;

reg [2:0]C_State,N_State; // state declarations

// State register

always @(posedge Clk,negedge Rst) begin

     if(!Rst)
     begin
          C_State <= s0;
     end

     else begin
          C_State <= N_State;
     end
end
```

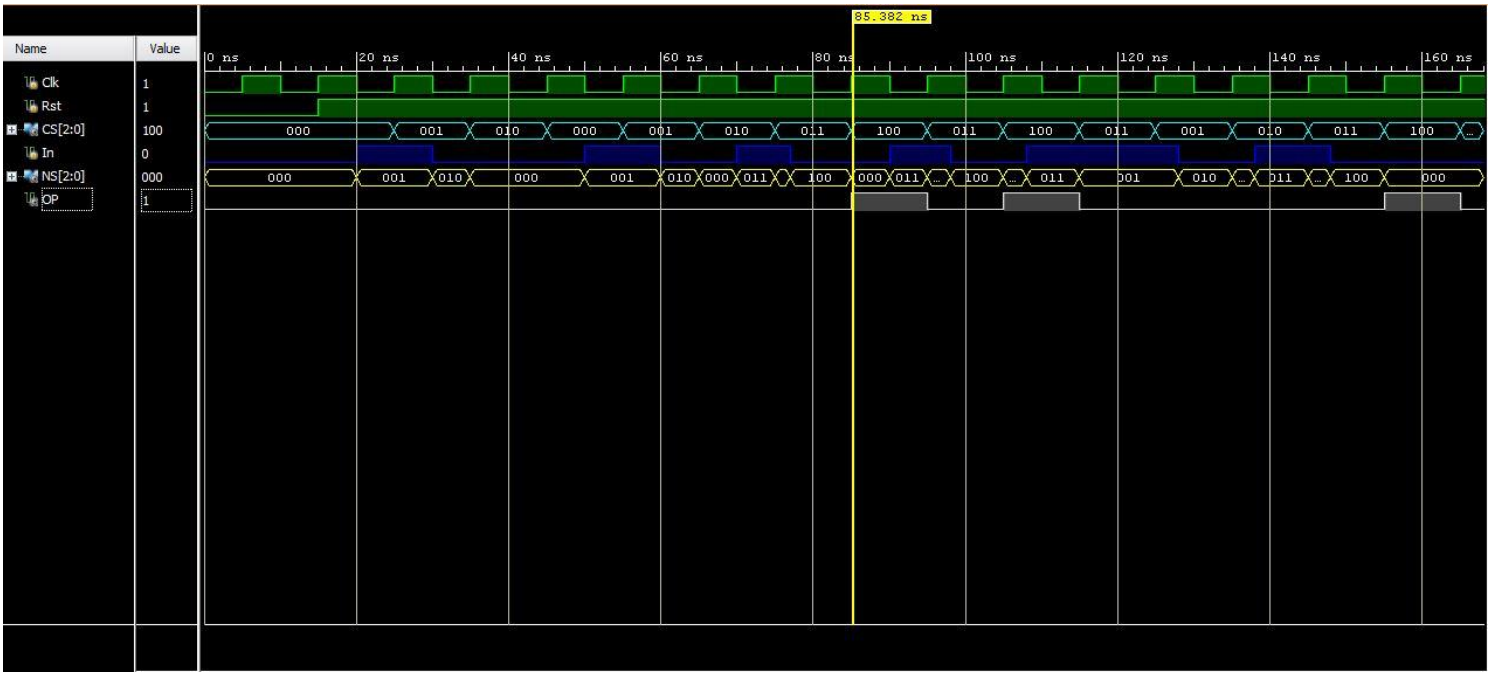
```
// Next state logic

always @(C_State,In) begin

     case(C_State)

          s0    : N_State <= In ? s1 : s0 ;

          s1    : N_State <= In ? s1 : s10 ;

          s10   : N_State <= In ? s101 : s0 ;

          s101  : N_State <= In ? s1 : s1010 ;

          s1010 : N_State <= In ? s101 : s0 ;

          default : N_State <= s0;

     endcase
end

// Output logic

assign OP = (C_State == s1010);

//
assign CS = C_State ;
assign NS = N_State ;

endmodule
```

TESTBENCH MODULE

```
`timescale 1ns / 1ps
module moore_1010_seq_det_over_tb ();
// Signal Declarations
reg Clk,Rst,In;
wire OP;
wire [2:0]CS,NS;

//instantiating design to test
moore_1010_seq_det_over UUT(Clk,Rst,In,OP,CS,NS);

// Monitoring the simulation results
initial begin
     $monitor("Time = %0t || Current State = %b || Input = %b || Next
State = %b || Output = %b",$time,CS,In,NS,OP);
end

// clock signal generation
initial begin
     Clk = 1'b0;
     forever #5 Clk = ~Clk;
end

// input signals
initial begin
         Rst = 1'b0;
     #15 Rst = 1'b1; end

initial  begin
         In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #10 In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
```

```
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #7  In  = 1'b0;
        #13 In  = 1'b1;
        #8  In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b0;
        #10 $finish();
end


endmodule
```

SIMULATION WAVEFORMS



SIMULATION OUTPUTS

```
Time = 0 || Current State = 000 || Input = 0 || Next State = 000 || Output = 0
Time = 20000 || Current State = 000 || Input = 1 || Next State = 001 || Output = 0
Time = 25000 || Current State = 001 || Input = 1 || Next State = 001 || Output = 0
Time = 30000 || Current State = 001 || Input = 0 || Next State = 010 || Output = 0
Time = 35000 || Current State = 010 || Input = 0 || Next State = 000 || Output = 0
Time = 45000 || Current State = 000 || Input = 0 || Next State = 000 || Output = 0
Time = 50000 || Current State = 000 || Input = 1 || Next State = 001 || Output = 0
Time = 55000 || Current State = 001 || Input = 1 || Next State = 001 || Output = 0
Time = 60000 || Current State = 001 || Input = 0 || Next State = 010 || Output = 0
Time = 65000 || Current State = 010 || Input = 0 || Next State = 000 || Output = 0
Time = 70000 || Current State = 010 || Input = 1 || Next State = 011 || Output = 0
Time = 75000 || Current State = 011 || Input = 1 || Next State = 001 || Output = 0
Time = 77000 || Current State = 011 || Input = 0 || Next State = 100 || Output = 0
Time = 85000 || Current State = 100 || Input = 0 || Next State = 000 || Output = 1
Time = 90000 || Current State = 100 || Input = 1 || Next State = 011 || Output = 1
Time = 95000 || Current State = 011 || Input = 1 || Next State = 001 || Output = 0
Time = 98000 || Current State = 011 || Input = 0 || Next State = 100 || Output = 0
Time = 105000 || Current State = 100 || Input = 0 || Next State = 000 || Output = 1
Time = 108000 || Current State = 100 || Input = 1 || Next State = 011 || Output = 1
Time = 115000 || Current State = 011 || Input = 1 || Next State = 001 || Output = 0
Time = 125000 || Current State = 001 || Input = 1 || Next State = 001 || Output = 0
Time = 128000 || Current State = 001 || Input = 0 || Next State = 010 || Output = 0
Time = 135000 || Current State = 010 || Input = 0 || Next State = 000 || Output = 0
Time = 138000 || Current State = 010 || Input = 1 || Next State = 011 || Output = 0
Time = 145000 || Current State = 011 || Input = 1 || Next State = 001 || Output = 0
Time = 148000 || Current State = 011 || Input = 0 || Next State = 100 || Output = 0
Time = 155000 || Current State = 100 || Input = 0 || Next State = 000 || Output = 1
Time = 165000 || Current State = 000 || Input = 0 || Next State = 000 || Output = 0
```

SEQUENCE DETECTOR 101X USING MEALY MACHINE (OVERLAPPING)

DESIGN MODULE

```
`timescale 1ns / 1ps
module mealy_101X_seq_det_over(Clk,Rst,In,OP,CS,NS);
input Clk,Rst,In; // Input signals
output OP;        // Output signal
output [1:0]CS,NS; // states

parameter   s0   = 0,  // state registers
            s1   = 1,
            s10  = 2,
            s101 = 3;

reg [1:0]C_State,N_State; // state declarations

// State register

always @(posedge Clk,negedge Rst) begin

     if(!Rst)
     begin
          C_State <= s0;
     end

     else begin
          C_State <= N_State;
     end
end
```

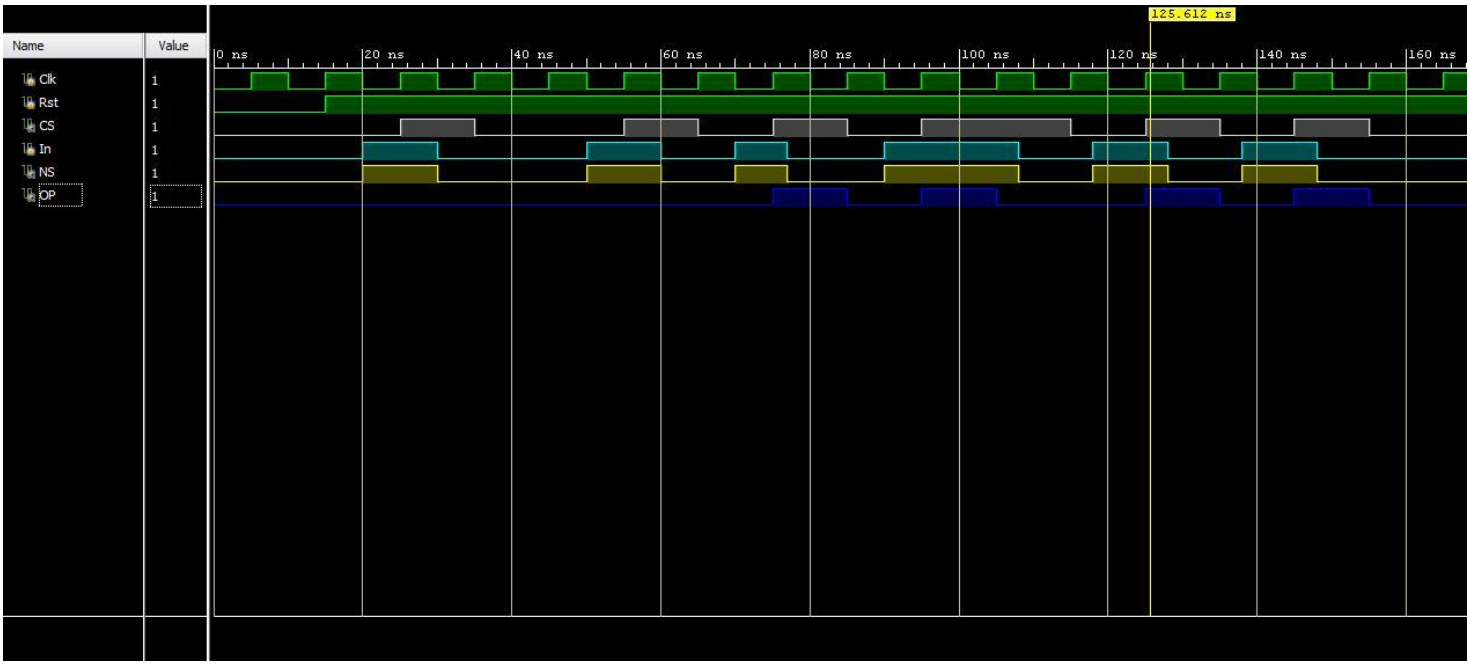
```
// Next state logic

always @(C_State,In) begin

     case(C_State)

          s0  : N_State <= In ? s1:s0 ;

          s1  : N_State <= In ? s1:s10 ;

          s10 : N_State <= In ? s101:s0 ;

          s101: N_State <= In ? s1:s10 ;

          default : N_State <= s0;

     endcase
end

// Output logic

assign OP = ((C_State == s101 ) ?(In ? 1:1 ):(In ? 0 : 0)) ;

//assing OP = (C_State == s101)  --> we can use it also for this sequence
Detector

//
assign CS = C_State ;
assign NS = N_State ;

endmodule
```

TESTBENCH MODULE

```
`timescale 1ns / 1ps
module mealy_101X_seq_det_over_tb ();
// Signal Declarations
reg Clk,Rst,In;
wire OP;
wire CS,NS;

//instantiating design to test
mealy_101X_seq_det_over UUT(Clk,Rst,In,OP,CS,NS);

// Monitoring the simulation results
initial begin
     $monitor("Current State = %b || Input = %b || Next State = %b ||
Output = %b",CS,In,NS,OP);
end

// clock signal generation
initial begin
     Clk = 1'b0;
     forever #5 Clk = ~Clk;
end

// input signals
initial begin
         Rst = 1'b0;
     #15 Rst = 1'b1; end

initial begin
         In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #10 In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
```

```
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #7  In  = 1'b0;
        #13 In  = 1'b1;
        # 8 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b1;
        #10 In  = 1'b0;
        #10 In  = 1'b0;
        #10 $finish();
end


endmodule
```

SIMULATION WAVEFORMS



SIMULATION OUTPUTS

```
Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Current State = 1 || Input = 1 || Next State = 1 || Output = 0
Current State = 1 || Input = 0 || Next State = 0 || Output = 0
Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Current State = 1 || Input = 1 || Next State = 1 || Output = 0
Current State = 1 || Input = 0 || Next State = 0 || Output = 0
Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Current State = 1 || Input = 1 || Next State = 1 || Output = 1
Current State = 1 || Input = 0 || Next State = 0 || Output = 1
Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Current State = 1 || Input = 1 || Next State = 1 || Output = 1
Current State = 1 || Input = 1 || Next State = 1 || Output = 0
Current State = 1 || Input = 0 || Next State = 0 || Output = 0
Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Current State = 1 || Input = 1 || Next State = 1 || Output = 1
Current State = 1 || Input = 0 || Next State = 0 || Output = 1
Current State = 0 || Input = 0 || Next State = 0 || Output = 0
Current State = 0 || Input = 1 || Next State = 1 || Output = 0
Current State = 1 || Input = 1 || Next State = 1 || Output = 1
Current State = 1 || Input = 0 || Next State = 0 || Output = 1
Current State = 0 || Input = 0 || Next State = 0 || Output = 0
```

SEQUENCE DETECTOR 10X0 USING MOORE MACHINE (OVERLAPPING)

DESIGN MODULE

```
`timescale 1ns / 1ps
module moore_10X0_seq_det_over(Clk,Rst,In,OP,CS,NS);
input Clk,Rst,In; // Input signals
output OP;        // Output signal
output [2:0]CS,NS; // states

parameter   s0    = 0 ,  // state registers
            s1    = 1 ,
            s10   = 2 ,
            s100  = 3 ,
            s101  = 4 ,
            s10X0 = 5 ;

reg x; // to know the 10X0--> X direction(1010/1000)

reg [2:0]C_State,N_State; // state declarations

// State register

always @(posedge Clk,negedge Rst) begin

     if(!Rst)
     begin
          C_State <= s0;
     end

     else begin
          C_State <= N_State;
     end
end
```

```
// Next state logic

always @(C_State,In) begin

     case(C_State)

          s0    : N_State <= In ? s1 : s0 ;

          s1    : N_State <= In ? s1 : s10 ;

          s10   : N_State <= In ? s101 : s100 ;

          s100  : begin N_State <= In ? s1 : s10X0 ;  x <= 0 ;end

          s101  : begin N_State <= In ? s1 : s10X0 ;  x <= 1 ; end

          s10X0 : N_State <= x ? (In ? s101 : s100 ) :(In ? s1 : s0 ) ;

          default : N_State <= s0;

     endcase
end

// Output logic

assign OP = (C_State == s10X0);


//
assign CS = C_State ;
assign NS = N_State ;

endmodule
```

TESTBENCH MODULE

```
`timescale 1ns / 1ps

module moore_10X0_seq_det_over_tb ();
// Signal Declarations
reg Clk,Rst,In;
wire OP;
wire [2:0]CS,NS;


//instantiating design to test
moore_10X0_seq_det_over UUT(Clk,Rst,In,OP,CS,NS);

// Monitoring the simulation results
initial begin
     $monitor("time = %0t || Current State = %b || Input = %b || Next
State = %b || Output = %b",$time,CS,In,NS,OP);
end

// clock signal generation
initial begin
     Clk = 1'b0;
     forever #5 Clk = ~Clk;
end

// input signals
initial begin
         Rst = 1'b0;
     #15 Rst = 1'b1; end

initial begin
         In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #10 In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #7  In  = 1'b0;
     #13 In  = 1'b1;
     #8  In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #10 In  = 1'b0;
     #10 In  = 1'b1;
     #10 In  = 1'b0;
     #10 In  = 1'b0;
     #10 $finish();
end
endmodule
```

SIMULATION WAVEFORMS

SIMULATION OUTPUTS

```
time = 0 || Current State = 000 || Input = 0 || Next State = 000 || Output = 0
time = 20000 || Current State = 000 || Input = 1 || Next State = 001 || Output = 0
time = 25000 || Current State = 001 || Input = 1 || Next State = 001 || Output = 0
time = 30000 || Current State = 001 || Input = 0 || Next State = 010 || Output = 0
time = 35000 || Current State = 010 || Input = 0 || Next State = 011 || Output = 0
time = 45000 || Current State = 011 || Input = 0 || Next State = 100 || Output = 0
time = 50000 || Current State = 011 || Input = 1 || Next State = 001 || Output = 0
time = 55000 || Current State = 001 || Input = 1 || Next State = 001 || Output = 0
time = 60000 || Current State = 001 || Input = 0 || Next State = 010 || Output = 0
time = 65000 || Current State = 010 || Input = 0 || Next State = 011 || Output = 0
time = 70000 || Current State = 010 || Input = 1 || Next State = 100 || Output = 0
time = 75000 || Current State = 100 || Input = 1 || Next State = 001 || Output = 1
time = 77000 || Current State = 100 || Input = 0 || Next State = 100 || Output = 1
time = 90000 || Current State = 100 || Input = 1 || Next State = 001 || Output = 1
time = 95000 || Current State = 001 || Input = 1 || Next State = 001 || Output = 0
time = 98000 || Current State = 001 || Input = 0 || Next State = 010 || Output = 0
time = 105000 || Current State = 010 || Input = 0 || Next State = 011 || Output = 0
time = 108000 || Current State = 010 || Input = 1 || Next State = 100 || Output = 0
time = 115000 || Current State = 100 || Input = 1 || Next State = 001 || Output = 1
time = 125000 || Current State = 001 || Input = 1 || Next State = 001 || Output = 0
time = 128000 || Current State = 001 || Input = 0 || Next State = 010 || Output = 0
time = 135000 || Current State = 010 || Input = 0 || Next State = 011 || Output = 0
time = 138000 || Current State = 010 || Input = 1 || Next State = 100 || Output = 0
time = 145000 || Current State = 100 || Input = 1 || Next State = 001 || Output = 1
time = 148000 || Current State = 100 || Input = 0 || Next State = 100 || Output = 1
```