College of Engineering, Pune
Dept. of Electronics and Telecommunication Engineering

Course **: RTL Simulation and Synthesis Lab (FY VLSI & ES)**
Year : 2020-21

RTL Lab 4,5 Report :

Mis : 122035014
Name : Swapnil Hanmant Phalke

1011 sequence detector using moore machine non-overlapping

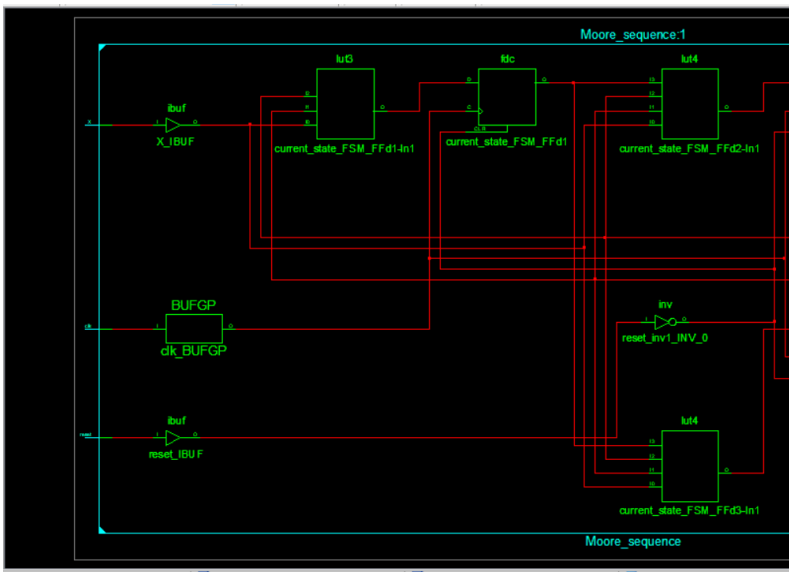
Code:-

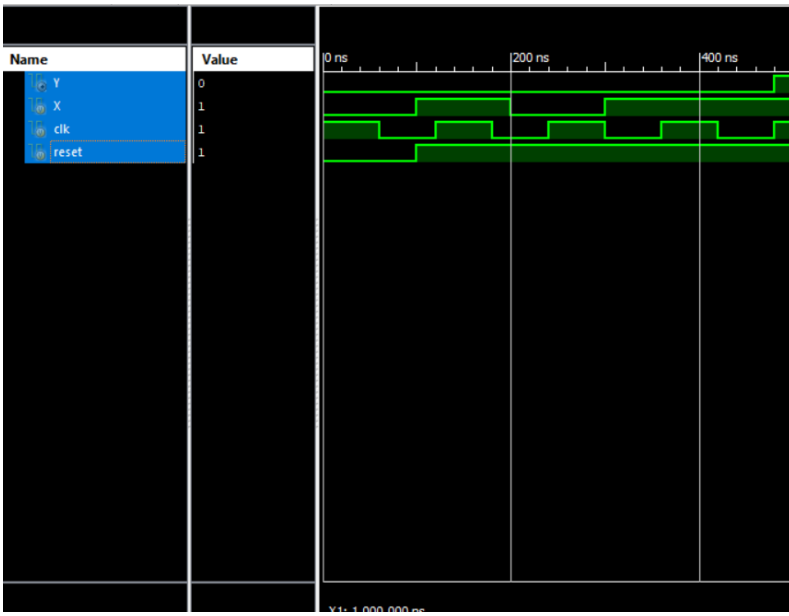
```
module Moore_sequence(X,clk,reset,Y);
input X,clk,reset;
output reg Y;
parameter  S0=3'b000,
  S1=3'b001,
  S2=3'b011,
  S3=3'b010,
  S4=3'b110;
reg [2:0] current_state, next_state;
always @(negedge reset,posedge clk)
begin
 if(!reset)
 current_state <= S0;
 else
 current_state <= next_state;
end
always @(current_state,X)
begin
 case(current_state)
 S0:begin
 if(X==1'b1)
          next_state = S1;
  else
   next_state = S0;
 end
 S1:begin
 if(X==1'b0)
          next_state = S2;
  else
   next_state = S1;
 end
 S2:begin
 if(X==1'b0)
          next_state = S0;
  else
   next_state = S3;
 end
 S3:begin
 if(X==1'b0)
          next_state = S2;
  else
   next_state = S4;
 end
 S4:begin
 if(X==1'b0)
          next_state = S0;
  else
   next_state = S1;

 end
 default:next_state = S0;
 endcase
end
always @(current_state)
begin
  case(current_state)
    S0:Y=0;
          S1:Y=0;
          S2:Y=0;
          S3:Y=0;
          S4:Y=1;
          default Y=0;
  endcase
end
endmodule
```

testbench:

```
module qw;

          // Inputs
          reg x;
          reg clk;
          reg rst;

          // Outputs
```

```
        wire y;

        // Instantiate the Unit Under Test
(UUT)
        mealy uut (
                .x(x),
                .clk(clk),
                .rst(rst),
                .y(y)
        );

        initial begin

        clk = 1;
forever #60 clk=~clk;
end
                initial begin// Initialiye
Inputs
                x = 0;

                rst = 0;

                // Wait 100 ns for global rst
to finish
                #100;
                // Add stimulus here
rst=1;
x = 1;
#100;
x=0;
#100;
x = 1;
#100;
x = 1;
#100;
                // Add stimulus here

        end

endmodule
```

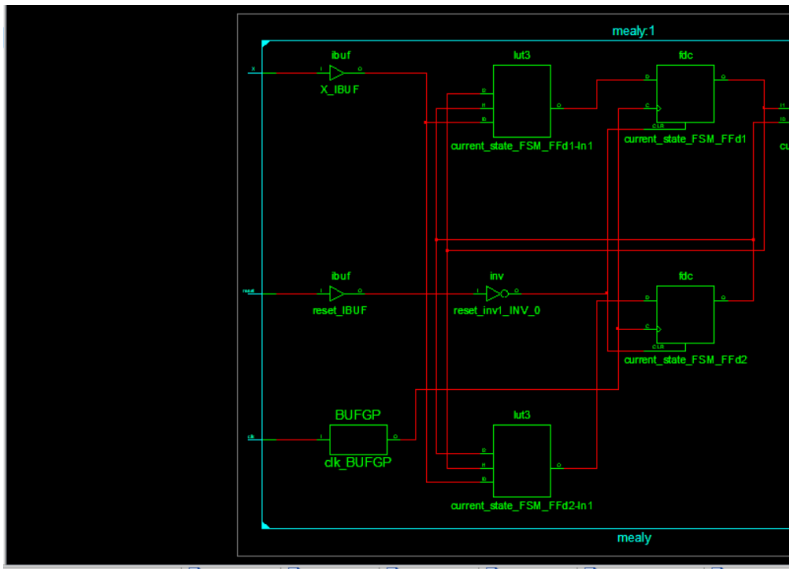


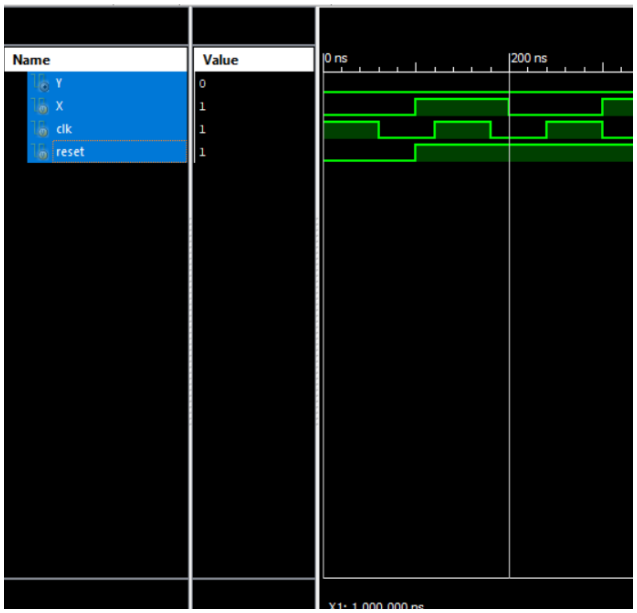


1011 Sequence detector using mealy machine

Code:-

```
module mealy(X,clk,reset,Y);
input X,clk,reset;
output reg Y;
parameter  S0=2'b00,
  S1=2'b01,
  S2=2'b10,
  S3=2'b11;
reg [1:0] current_state, next_state;
always @(negedge reset,posedge clk)
begin
 if(!reset)
 current_state <= S0;
 else
 current_state <= next_state;
end
always @(current_state,X)
begin
 case(current_state)
 S0:begin
 if(X==1'b1)
next_state = S1;
  else
   next_state = S0;
Y=0;
 end
 S1:begin
 if(X==1'b0)

  next_state = S2;
  else
   next_state = S1;
 Y=0;
 end
 S2:begin
 if(X==1'b0)

next_state = S0;
 else
  next_state = S3;
Y=0;
end
S3:begin
if(X==1'b0)

next_state = S2;
 else
  next_state = S0;
Y=1;
end
 default:next_state = S0;
endcase
end
endmodule
```



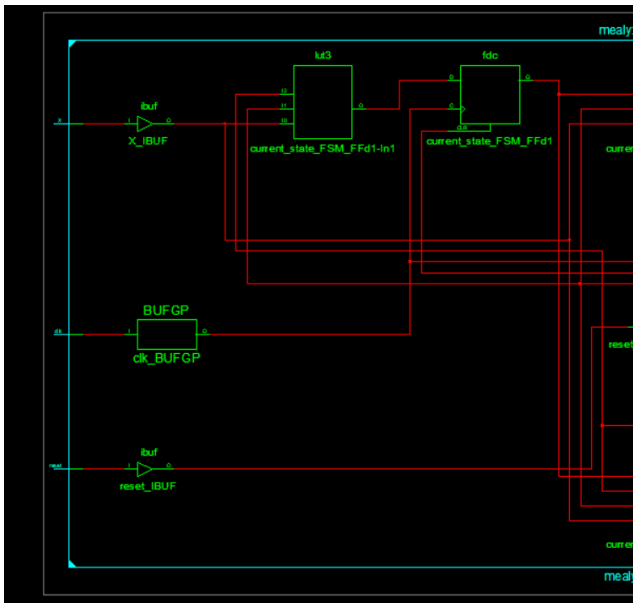


1011 Sequence detector using moore (overlap method)

Code-

```
module Moore_sequence(X,clk,reset,Y);
input X,clk,reset;
output reg Y;
parameter  S0=3'b000,
  S1=3'b001,
  S2=3'b011,
  S3=3'b010,
  S4=3'b110;
reg [2:0] current_state, next_state;
always @(negedge reset,posedge clk)
begin
 if(!reset)
 current_state <= S0;
 else
 current_state <= next_state;
end
always @(current_state,X)
begin
 case(current_state)
 S0:begin
 if(X==1'b1)

next_state = S1;
 else
  next_state = S0;
 end
 S1:begin
 if(X==1'b0)

next_state = S2;
 else
  next_state = S1;
 end
 S2:begin
 if(X==1'b0)

next_state = S0;
 else
  next_state = S3;
 end
 S3:begin
 if(X==1'b0)

next_state = S2;
 else
  next_state = S4;
 end
 S4:begin
 if(X==1'b0)

next_state = S2;
 else
  next_state = S1;


end
```

```
 default:next_state = S0;
 endcase
end
always @(current_state)
begin
  case(current_state)
    S0:Y=0;

 S1:Y=0;

 S2:Y=0;

 S3:Y=0;

 S4:Y=1;

 default Y=0;
  endcase
end
endmodule
```

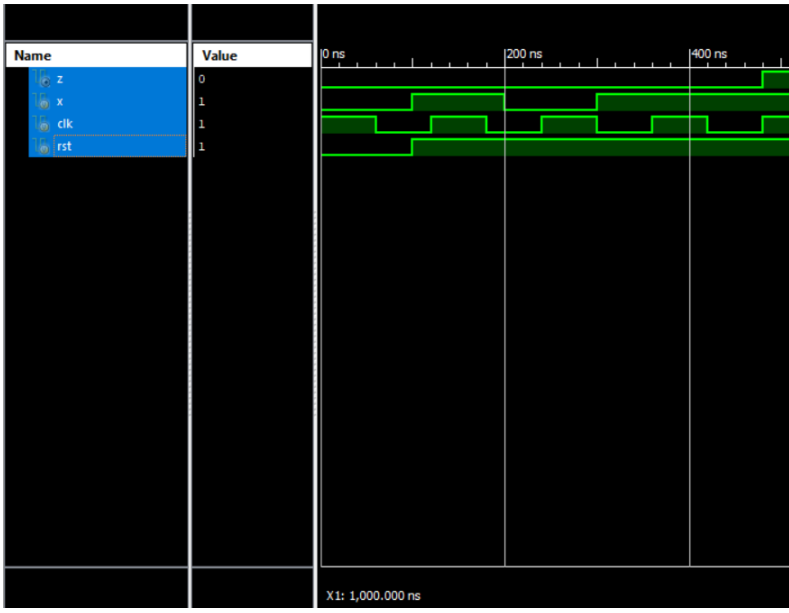


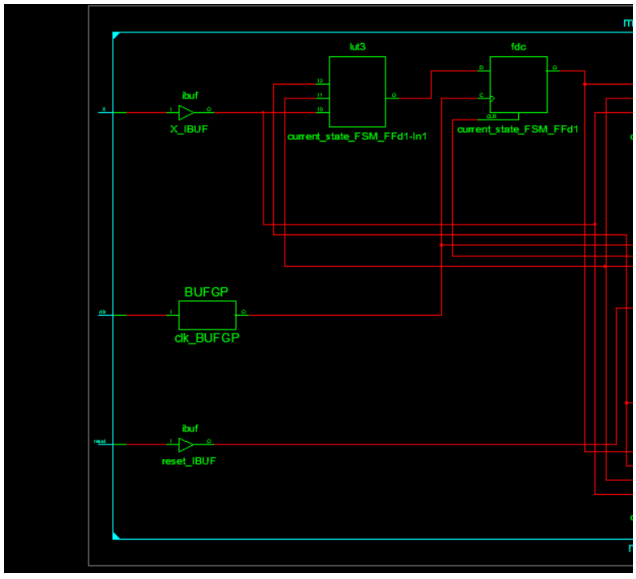
Write a code for Mealy FSM to avoid Glitches

Code:

```
module Mealy(clk,x,z,rst);
input clk,rst,x;
output reg z;
parameter A=2'b00,B=2'b01,C=2'b11,D=2'b10;
reg [1:0] ps,ns;
always @(posedge clk,negedge rst)
begin
if(!rst)
ps<=A;
else
ps<=ns;
end

always @(ps,x)
begin
case(ps)
A:
if(!x)
begin
ns<=A;
end
else
begin
ns<=B;
end
B:
if(!x)
begin
ns<=C;
end
else
begin
ns<=B;
end
C:
if(!x)
begin
ns<=A;

end
else
begin
ns<=D;
end
D:
if(!x)
begin
ns<=C;
end
else
begin
ns<=A;
end
default:
ns<=0;

endcase
end

always @(posedge clk ,negedge rst)
begin
if(!rst)
z<=1'b0;
else
begin
z<=1'b0;
case(ps)
A:
if(!x)
z<=0;
else
```

```
z<=0;
B:
if(!x)
z<=0;
else
z<=0;

C:
if(!x)
z<=0;
else
z<=0;

D:
if(!x)
z<=0;
else
z<=1;

endcase
end

end
endmodule
```

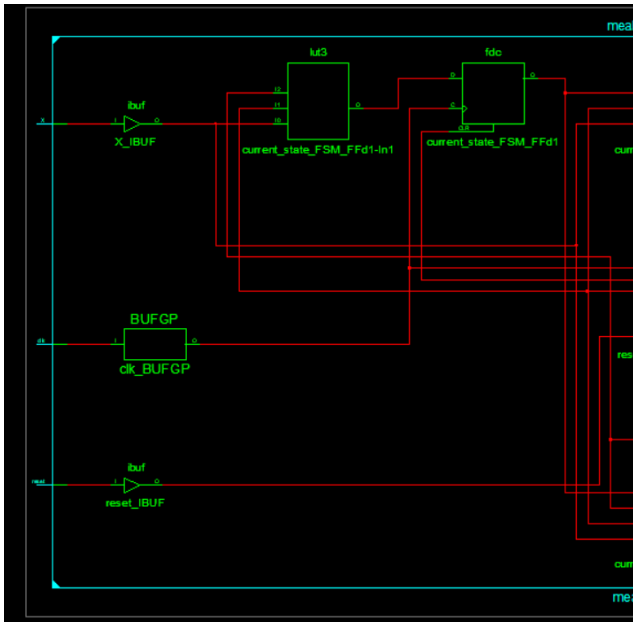


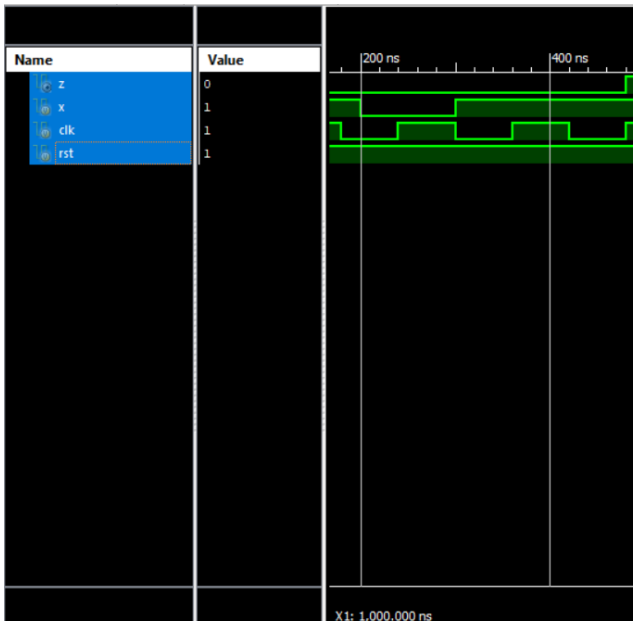


Mealy with one always block

```
module mealy_1alwaysblock(clk,x,z,rst);
input clk,rst,x;
output reg z;
parameter
A=2'b00,B=2'b01,C=2'b11,D=2'b10;
reg [1:0] ps,ns;
always @(posedge clk,negedge rst)
begin
if(!rst)
begin
ps=A;
z=0;
end
else
begin
case(ps)
A:   if(!x)
     begin
     ns=A;
     ps=ns;

  z=0;
  end
  else
  begin
  ns=B;
ps=ns;
  z=0;
  end
B:  if(!x)
     begin
     ns=C;
ps=ns;

  z=0;
  end
  else
  begin
  ns=B;
ps=ns;
  z=0;
  end
C:  if(!x)
     begin
     ns=A;
ps=ns;

  z=0;
  end
  else
  begin
  ns=D;
ps=ns;
  z=0;
  end
D: if(!x)
    begin
     ns=C;
ps=ns;

  z=0;
  end
  else
  begin
  ns=A;
ps=ns;
  z=1;
  end
default:
begin
ns=A;
z=0;
end
endcase
end
```

```
end
endmodule
```





Moore using 2 always blocks.

```
module moore_2alwaysblock(clk,rst,x,y);
input rst,clk,x;
output reg y;
reg [2:0]ps,ns;
parameter s0=3'b000, s1=3'b001 ,s2=3'b010,
s3=3'b011, s4=3'b100;
always @(posedge clk,negedge rst)
begin
if(!rst)
ps<=s0;
else
ps<=ns;
end
always @(ps,x)
begin
case(ps)
3'b000: begin
y<=1'b0;
if(!x)
 ns <=s0;
  else
  ns<=s1;
  end
3'b001: begin
y<=1'b0;
if(!x)
 ns <=s2;
  else
  ns<=s1;
  end
3'b010:  begin
y<=1'b0;
if(!x)
ns <=s0;
  else
ns<=s3;
end
3'b011:  begin
y<=1'b0;if(!x)
ns <=s2;
else
ns<=s4;
end
```

```
3'b100:  begin
y<=1'b1;
if(!x)
ns <=s0;
else
ns<=s1;
end
default:
begin
y<=1'b0;
ns<=s0;
end
endcase
end
endmodule
```