Digital Design Lab 8

Name:Dhruv

Roll Number: B20EE016

Task 1: 8 bit MLS counter using LFSR

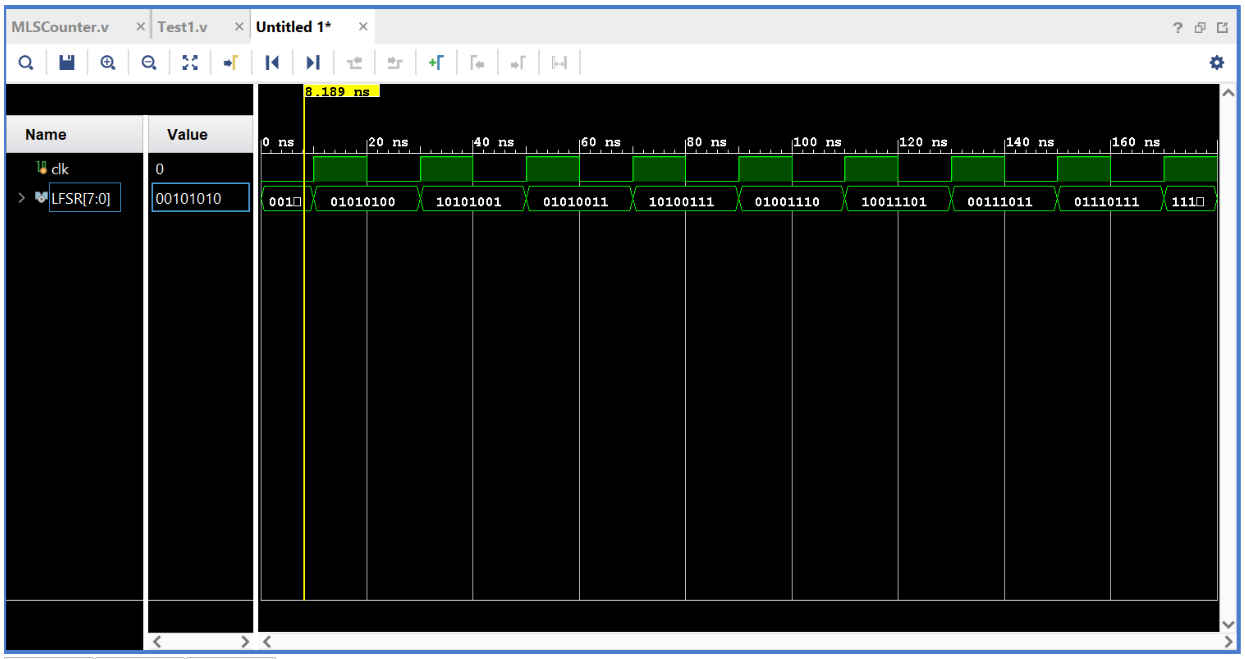
Code:

```
`timescale 1ns / 1ps

module MLSCounter(clk, LFSR);
input clk;
output reg [7:0] LFSR = 42;
wire w1 = LFSR[3];
wire w2 = LFSR[2];
wire w3 = LFSR[1];
wire w4 = LFSR[7];
always@(posedge clk)
 begin
 LFSR[7:1] <= LFSR[6:0];
 LFSR[0] <= LFSR[3]^LFSR[4]^LFSR[5]^LFSR[7];
end
endmodule
```

Test Bench:

```
module MLSCountertest;
reg clk;
wire [7:0] LFSR;
MLSCounter mls(clk, LFSR);
initial begin
 clk = 0;
end
always #10 clk = ~clk;
initial #180 $finish;
endmodule
```

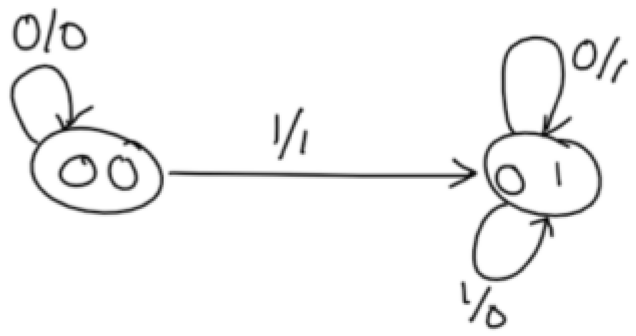
Waveform:



Name	Value	Waveform
clk	0	
LFSR[7:0]	00101010	001□, 01010100, 10101001, 01010011, 10100111, 01001110, 10011101, 00111011, 01110111, 111□

Task 2: 2's complement of a running bitstream using

a) Mealy Machine



Code:

```
module melay_machine(clk, reset, a, out, ps, ns);
input a;
input clk, reset;
```

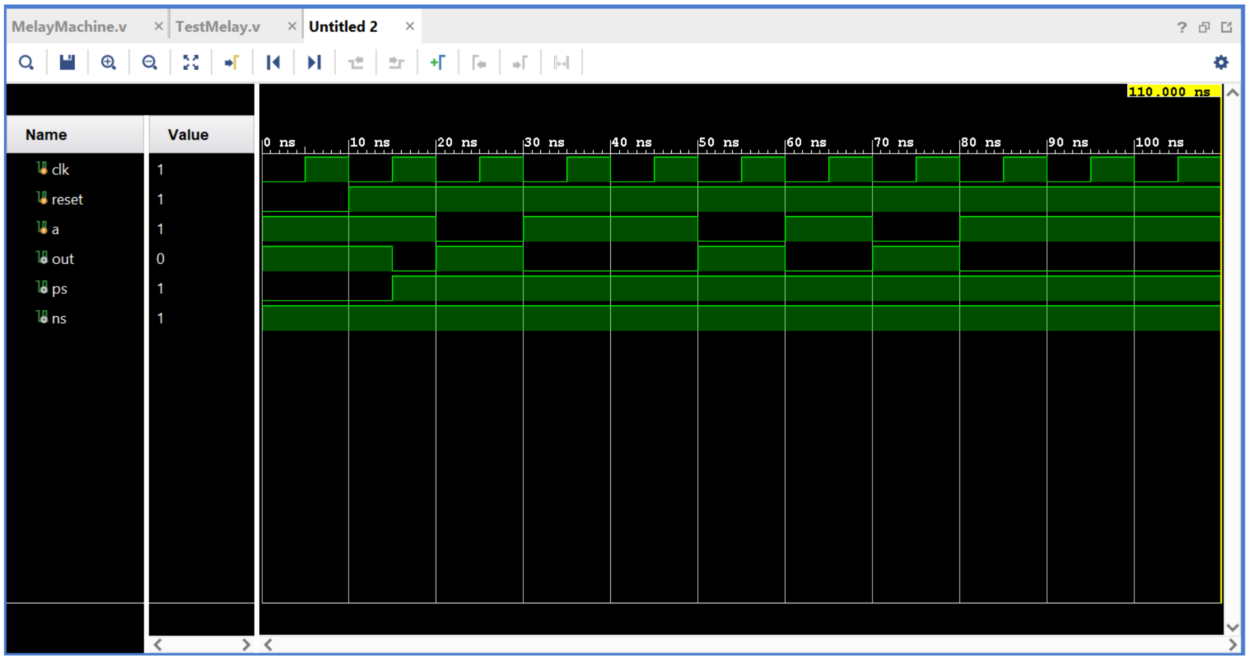
```
output reg out;
output reg ps, ns;
parameter s0 = 1'b0;
parameter s1 = 1'b1;
always@(posedge clk, negedge reset)
begin
 if(!reset)
 ps <= s0;
 else
 ps <= ns;
end
always@(ps, a)
begin
 case(ps)
 s0: if(a)
 begin
 ns <= s1;
 out <= 1;
 end
 else
 begin
 ns<=s0;
 out<=0;
 end
 s1: if(a)
 begin
 out <= 0;
 ns <= s1;
 end
 else
 begin
 out <= 1;
 ns <= s1;
 end
 endcase
end
endmodule
```

Test Bench:

```
module test_melay2_complement;
reg clk, reset;
reg a;
wire out;
wire ps, ns;
melay_machine mc(clk, reset, a, out, ps, ns);
always #5 clk = ~clk;
initial
 begin
```

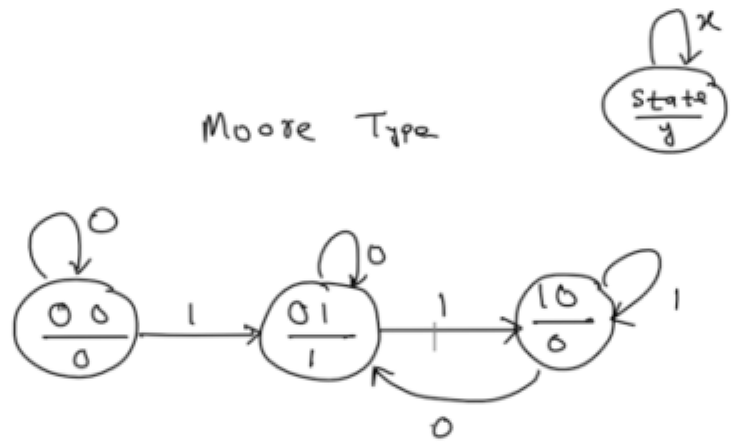
```
  clk = 0;
  reset = 0;
  a = 1;
  #10 reset = 1;
  #10 a = 0;
  #10 a = 1;
  #10 a = 1;
  #10 a = 0;
  #10 a = 1;
  #10 a = 0;
  #10 a = 1;
  #10 a = 1;
  #10 a = 1;
  end

 initial #110 $finish;
 endmodule
```

Waveform:



Name	Value
clk	1
reset	1
a	1
out	0
ps	1
ns	1

b) Moore Machine



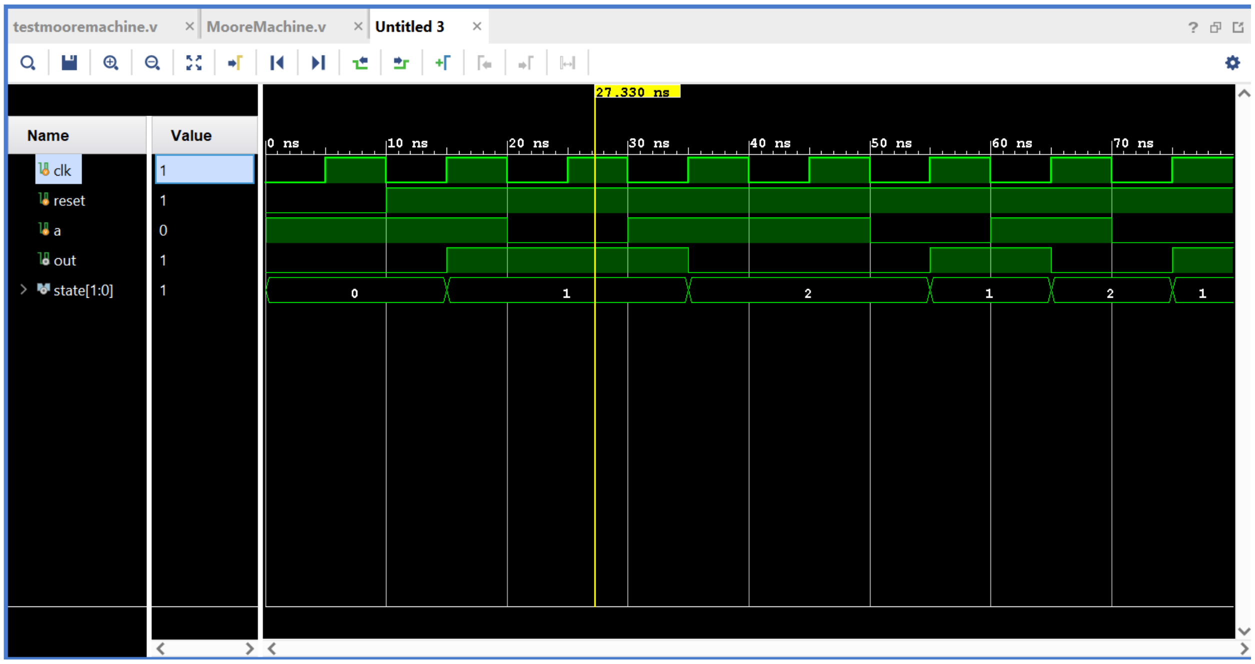
Code:

```
module moore_machine(clk, reset, a, out, state);
parameter s0 = 2'b00, s1 = 2'b01, s2 = 2'b10;
integer i;
input clk, reset, a;
output reg out;
output reg [1:0] state;


always @(posedge clk, negedge reset)
begin
 if(!reset)
 state<=s0;
 else
 begin
 case(state)
 s0:if(!a)
 state<=s0;
 else
 state<=s1;
 s1:if(!a)
 state<=s1;
 else
 state<=s2;
 s2:if(!a)
 state<=s1;
 else
 state<=s2;
 endcase
 end
end
always @(state)
 begin
 case(state)
 s0: out<=0;
 s1: out<=1;
 s2: out<=0;
 endcase
 end
endmodule
```

Test Bench:

```
module test_moore_machine_2_complement();
reg clk ,reset;
reg a;
wire out;
wire [1:0] state;
moore_machine mo(clk, reset, a, out, state);
always #5 clk = ~clk;
initial begin
 clk = 0;
 reset = 0;
 a = 1;
 #10 reset = 1;
 #10 a = 0;
 #10 a = 1;
 #10 a = 1;
 #10 a = 0;
 #10 a = 1;
 #10 a = 0;
end
initial #80 $finish;
endmodule
```

Waveform:



Name	Value
clk	1
reset	1
a	0
out	1
state[1:0]	1