Lab Assignment 10

ES 204: Digital Systems, Prof. Joycee Mekie

2nd March, 2024

Indian Institute of Technology, Gandhinagar

Aditya N. Mehta

22110017

Hrriday V. Ruparel

22110099

10010 Overlaping Pattern Detection (Moore Machine)

A) State Table:

Present State	Next State		Output (z)
	w = 0	w = 1	
A	A	B	0
B	C	B	0
C	D	B	0
D	A	E	0
E	F	B	0
F	D	A	1

B) Verilog Code:

```
// Moore Implimentation of 10010 overlaping sequence detection using FSM

`timescale 1ns/1ps

module FSM_Moore(clk, reset, w, z);
input clk, reset, w;
output z;
reg [2:0] y, Y;

parameter [2:0] A=3'b000, B=3'b001, C=3'b010, D=3'b011, E=3'b100, F=3'b101;

always@(w or y) begin
   // Assigning Next State
   case(y)
       A: if(w) Y = B;
          else Y = A;
       B:if(w) Y = B;
          else Y = C;
       C:if(w) Y = B;
          else Y = D;
       D:if(w) Y = E;
          else Y = A;
       E:if(w) Y = B;
          else Y = F;
       F:if(w) Y = A;
          else Y = D;
       default: Y = 3'bxxx;
   endcase
end

always@(negedge reset, posedge clk) begin
   // State Transition
   if(~reset) y <= A;
   else y <= Y;
end

// Output
assign z = (y == F);

endmodule
```

10010 Overlaping Pattern Detection (Mealy Machine)

A) State Table:

Present State	Next State		Output	
	w = 0	w = 1	w = 0	w =1
A	A	B	0	0
B	C	B	0	0
C	D	B	0	0
D	A	E	0	0
E	C	B	1	0

B) Verilog Code:

```
// Mealy Implimentation of 10010 overlaping sequence detection using FSM

`timescale 1ns/1ps

module FSM_Mealy(clk, reset, w, z);
input clk, reset, w;
output reg z;
reg [2:0] y, Y;

parameter [2:0] A=3'b000, B=3'b001, C=3'b010, D=3'b011, E=3'b100;

always@(w or y) begin
   // Assigning Next State
   case(y)
       A: if(w) begin Y = B; z = 0; end
           else begin Y = A; z = 0; end
       B:if(w) begin Y = B; z = 0; end
           else begin Y = C; z = 0; end
       C:if(w) begin Y = B; z = 0; end
           else begin Y = D; z = 0; end
       D:if(w) begin Y = E; z = 0; end
           else begin Y = A; z = 0; end
       E:if(w) begin Y = B; z = 0; end
           else begin Y = C; z = 1; end
       default: begin Y = 3'bxxx; z = 1'bx; end
   endcase
end

always@(negedge reset, posedge clk) begin
   // State Transition
   if(~reset) y <= A;
   else y <= Y;
end
endmodule
```

Common Test Bench and Simulation for both Machines

A) Verilog Code for Testbench:

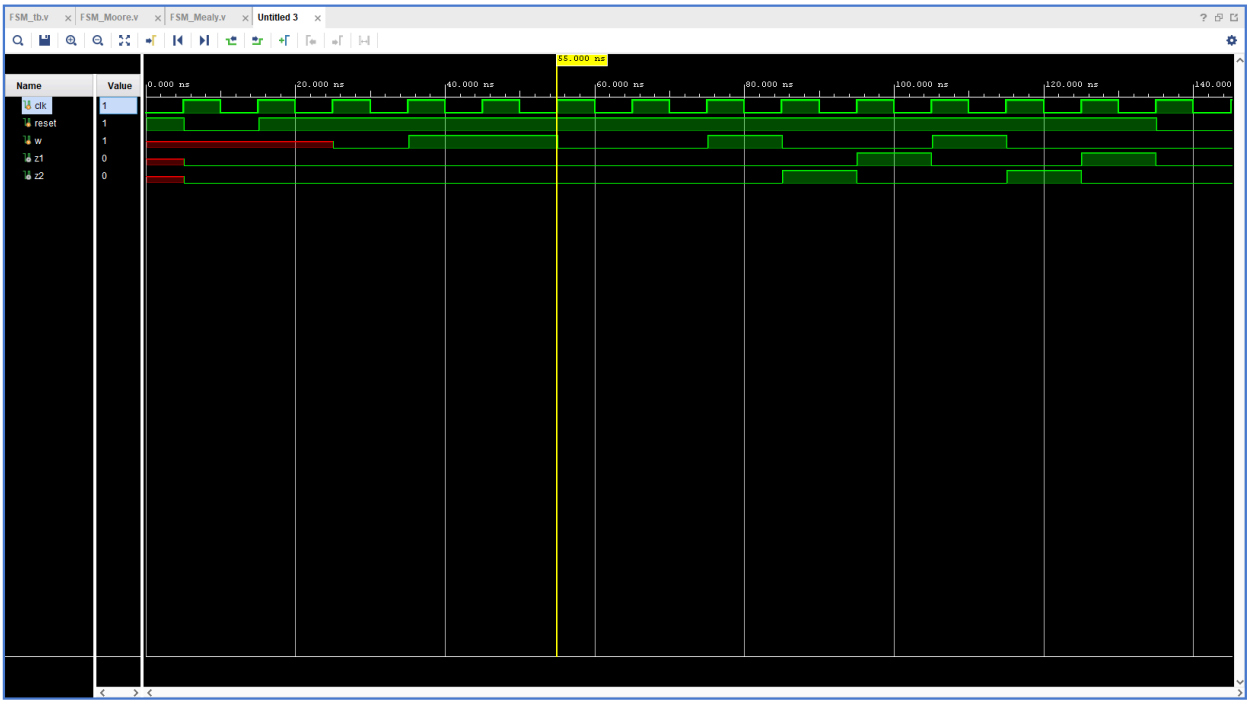
```
`timescale 1ns / 1ps

module FSM_tb();

reg clk;
reg reset;
reg w;
wire z1, z2;

FSM_Moore inst1(clk, reset, w, z1);
FSM_Mealy inst2(clk, reset, w, z2);

initial begin
   clk <= 0;
   forever begin
       #5 clk <= ~clk;
   end
end

initial begin
   reset = 1;
   #5.1;
   reset = 0;
   #10;
   reset = 1;
   #10;
   w = 0;
   #10;
   w = 1;
   #10;
   // Pattern Starts Now
   w = 1;
   #10;
   w = 0;
   #20;
   w = 1; // Overlapping Pattern
   #10;
   w = 0;
   #10;
   w = 0;
   #10;
   w = 1;
   #10;
   w = 0;
   #10;
   w = 0;
   #10;
   reset = 0; // Reset
   #10;


   $finish();
end
endmodule
```

B) Simulation Waveform:

w = Input

z1 = Output of Moore Machine

z2 = Output of Mealy Machine



Take Home FPGA Implementation:

A) Code for Moore Machine:

```
// Moore Implimentation of 10010 overlaping sequence detection using FSM
`timescale 1ns/1ps

module FSM_Moore(clk, reset, w, z, clk_tick, W, state);
input clk, reset, w;
output z;
reg [2:0] y, Y;
reg [27:0] slow_clk;
output reg clk_tick;
output reg W;
output reg [2:0] state;

parameter [2:0] A=3'b000, B=3'b001, C=3'b010, D=3'b011, E=3'b100, F=3'b101;

initial begin
   clk_tick <= 0;
end

always@(w or y) begin
   // Assigning Next State
   case(y)
       A: if(w) Y = B;
          else Y = A;
       B:if(w) Y = B;
          else Y = C;
       C:if(w) Y = B;
          else Y = D;
       D:if(w) Y = E;
          else Y = A;
       E:if(w) Y = B;
          else Y = F;
       F:if(w) Y = A;
          else Y = D;
       default: Y = 3'bxxx;
   endcase
end

always@(negedge reset, posedge clk) begin
   // State Transition
   if (slow_clk == 200_000_000) begin
       if(~reset) y <= A;
       else y <= Y;
       slow_clk <= 0;
       clk_tick <= ~clk_tick;
       W = w;
       state = y;
   end
   else begin
       slow_clk <= slow_clk + 1;
   end

end

// Output
assign z = (y == F);
endmodule
```

B) Code for Mealy Machine:

```
// Mealy Implimentation of 10010 overlaping sequence detection using FSM
`timescale 1ns/1ps

module FSM_Mealy(clk, reset, w, z, clk_tick, W, state);
input clk, reset, w;
output reg z;
reg [2:0] y, Y;
reg [28:0] slow_clk;
output reg clk_tick;
output reg W;
output reg [2:0] state;

parameter [2:0] A=3'b000, B=3'b001, C=3'b010, D=3'b011, E=3'b100;

initial begin
   clk_tick <= 0;
end

always@(w or y) begin
   // Assigning Next State
   case(y)
       A: if(w) begin Y = B; z = 0; end
          else begin Y = A; z = 0; end
       B:if(w) begin Y = B; z = 0; end
          else begin Y = C; z = 0; end
       C:if(w) begin Y = B; z = 0; end
          else begin Y = D; z = 0; end
       D:if(w) begin Y = E; z = 0; end
          else begin Y = A; z = 0; end
       E:if(w) begin Y = B; z = 0; end
          else begin Y = C; z = 1; end
       default: begin Y = 3'bxxx; z = 1'bx; end
   endcase
end

always@(negedge reset, posedge clk) begin
   // State Transition
   if (slow_clk == 400_000_000) begin
       if(~reset) y <= A;
       else y <= Y;
       slow_clk <= 0;
       clk_tick <= ~clk_tick;
       W = w;
       state = y;
   end
   else begin
       slow_clk <= slow_clk + 1;
   end
end
endmodule
```

C) **Constrains:**

```
set_property IOSTANDARD LVCMOS33 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports clk_tick]
set_property IOSTANDARD LVCMOS33 [get_ports reset]
set_property IOSTANDARD LVCMOS33 [get_ports w]
set_property IOSTANDARD LVCMOS33 [get_ports z]
set_property DRIVE 12 [get_ports clk_tick]
set_property DRIVE 12 [get_ports z]
set_property SLEW SLOW [get_ports clk_tick]
set_property SLEW SLOW [get_ports z]
set_property PACKAGE_PIN W5 [get_ports clk]
set_property PACKAGE_PIN R2 [get_ports reset]
set_property PACKAGE_PIN T1 [get_ports w]
set_property PACKAGE_PIN L1 [get_ports clk_tick]
set_property PACKAGE_PIN P1 [get_ports z]
set_property IOSTANDARD LVCMOS33 [get_ports W]
set_property IOSTANDARD LVCMOS33 [get_ports {state[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {state[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {state[0]}]
set_property PACKAGE_PIN N3 [get_ports W]
set_property PACKAGE_PIN P3 [get_ports {state[2]}]
set_property PACKAGE_PIN U3 [get_ports {state[1]}]
set_property PACKAGE_PIN W3 [get_ports {state[0]}]
```

D) **Video Link to Moore Implementation:** Click Here
E) **Video Link to Mealy Implementation:** Click Here