California State University, Fullerton

Computer Engineering

**EGCP 446 – Advanced Digital Design using Verilog HDL**

**(Fall 2019)**

**Lab No 3: FSM**

1. **Lab Description**

**Part A: Moore State Machine**

****

Figure Robot eyes a) state table b) state diagram

Write a Verilog code creating the state machine shown in Figure 1 b). Use clk, reset as input, whereas W [3:0] as output. Write an appropriate testbench code to test your HDL code.

**Part B: Mealy State Machine**

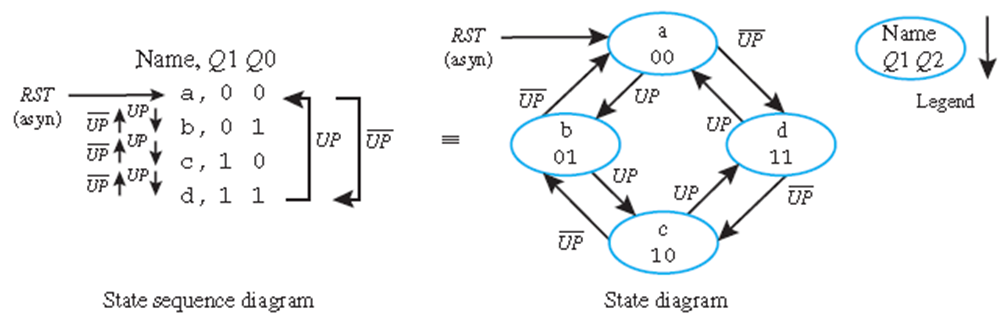


Figure . Binary Counter

Write a Verilog code creating the state machine shown in Figure 2. Use clk, reset, up as input, whereas Q [1:0] as output. Write an appropriate testbench code to test your HDL code.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submit your Verilog, Picture of Waveform, and testbench file

module moore(clk,rst,W);

input clk, rst;

output [3:0] W;

reg [4:0] state\_reg, state\_next;

parameter S0 = 5'b00001;

parameter S1 = 5'b00010;

parameter S2 = 5'b00100;

parameter S3 = 5'b01000;

parameter S4 = 5'b10100;

parameter S5 = 5'b10010;

always @(posedge clk, posedge rst)

if (rst) state\_reg <= S0;

else

state\_reg <= state\_next;

always @(\*)

case (state\_reg)

S0: state\_next = S1;

S1: state\_next = S2;

S2: state\_next = S3;

S3: state\_next = S4;

S4: state\_next = S5;

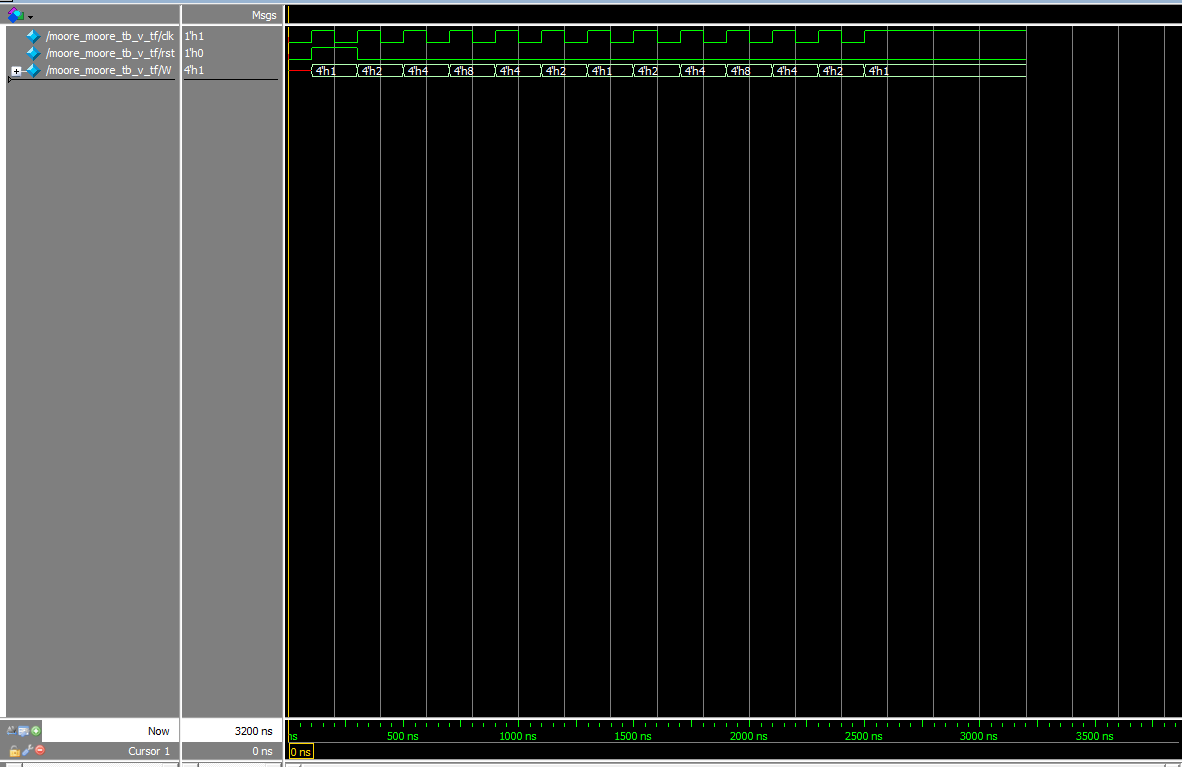
S5: state\_next = S0;

default: state\_next = S0;

endcase

assign W = state\_reg[3:0];

endmodule



module moore\_moore\_tb\_v\_tf();

reg clk;

reg rst;

wire [3:0] W;

moore uut (

.clk(clk),

.rst(rst),

.W(W)

);

initial begin

clk = 0;

rst = 0;

#100;

rst = 1;

clk = 1;

#100;

clk = 0;

#100;

rst = 0;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

end

endmodule

--------------------------------------------------------------------------------------------------------

module mealy(clk, rst, up, W);

input clk, rst, up;

output [1:0] W;

reg [1:0] state\_reg, state\_next;

parameter a = 2'b00;

parameter b = 2'b01;

parameter c = 2'b10;

parameter d = 2'b11;

always @(posedge clk, posedge rst)

if (rst) state\_reg <= a;

else

state\_reg <= state\_next;

always @(\*)

case (state\_reg)

a: if(up) state\_next = b;

else state\_next = d;

b: if(up) state\_next = c;

else state\_next = a;

c: if(up) state\_next = d;

else state\_next = b;

d: if(up) state\_next = a;

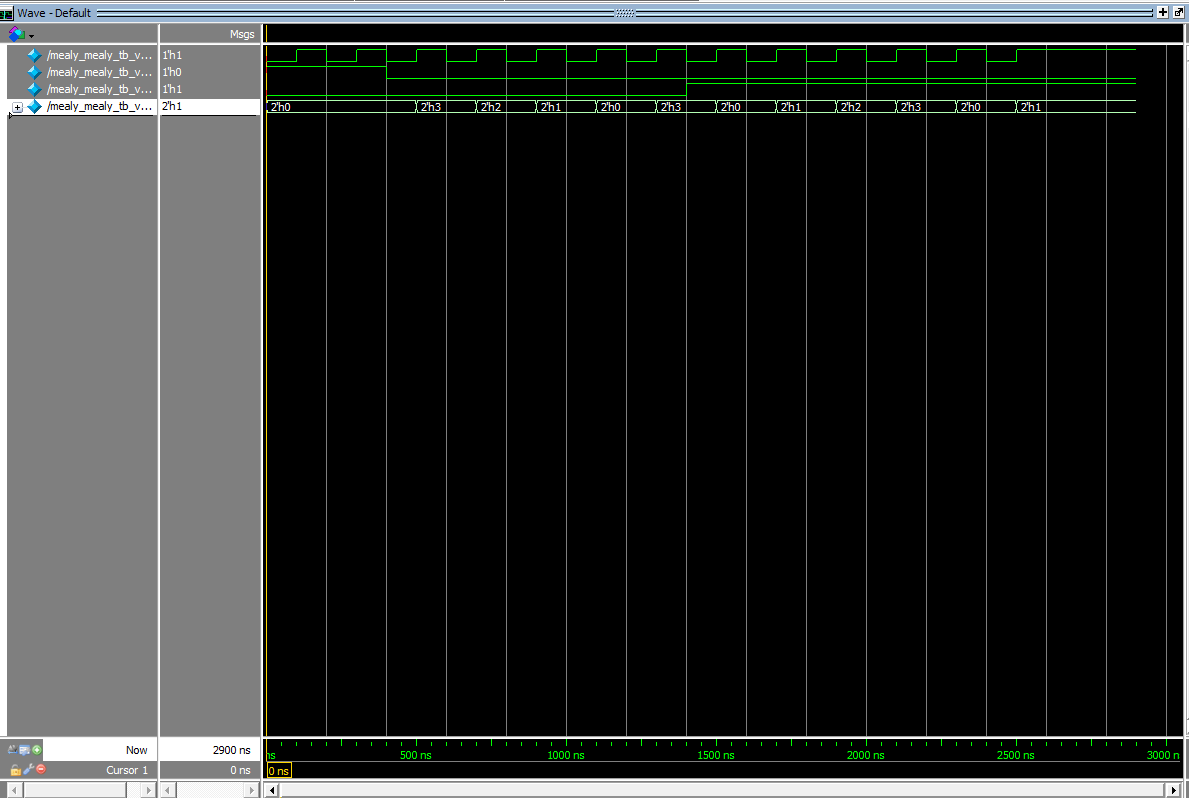
else state\_next = c;

default: state\_next = a;

endcase

assign W = state\_reg;

endmodule



module mealy\_mealy\_tb\_v\_tf();

reg clk;

reg rst;

reg up;

wire [1:0] W;

mealy uut (

.clk(clk),

.rst(rst),

.up(up),

.W(W)

);

initial begin

clk = 0;

rst = 1;

up = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

rst = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

up = 1;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

#100;

clk = 0;

#100;

clk = 1;

end

endmodule