CSC 347/ENS 211

Title and Experiment #	Lab9: ALU and Seven-Segment Display
Name	Gianna Galard
Date Performed	30-Nov-21
Date Submitted	07-Dec-21

The student pledges this work to be their own *Gianna Galard*

Link: https://www.edaplayground.com/x/eypm

Objective:

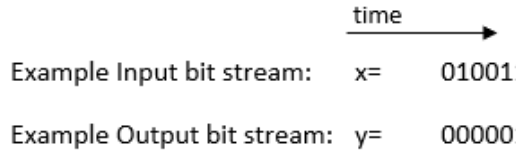
The objective of this week's lab was to design a synchronous sequence detector that detects a bit-pattern “110”.

Language and Compiler used:

- Verilog
- Edaplayground.com

Design Procedure:

We want to design a circuit which detects (110) sequences in a string of bits coming through an input line (i.e., the input is a serial bit stream). Once the (110) sequence is detected, output becomes (1), otherwise it stays as (0). A sample input and output bit streams (sequence) are given below. First bit coming to the input is the one shown on the far left.



The diagram of the sequence detector is shown below. Beside the clock, input x and output y, it has a reset input to force the detector into the initial state (“00”). The circuit also output the present state.

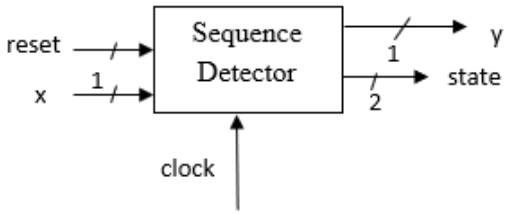
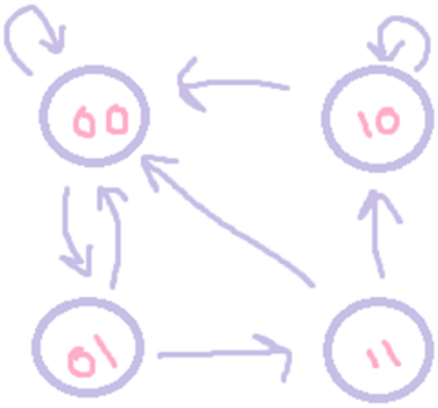


Figure 1. State Diagram



In the figure above, the student derived the state diagram for a Moore "110" sequence detector. Then, the student wrote verilog code to implement the sequence detector.

Figure 2. design.sv

```
// Author: Gianna Rose
// Lab 11
// https://www.edaplayground.com/x/eypm

module sequence_detector(clock, reset, x, y, state);
    // input
    input clock, reset, x;
    // output
    output y;
    output [1:0] state; // state is 2-bit
    reg [1:0] state;

  always @( posedge clock, posedge reset ) // state machine
     if (reset)
       state <= 2'b00; // reset state
  else
```

```
      case (state)
        2'b00: if(x)state <= 2'b01;   else state <= 2'b00;
        2'b01: if(x)state <= 2'b10;   else state <= 2'b00;
        2'b10: if(x)state <= 2'b10;   else state <= 2'b11;
        2'b11: if(x)state <= 2'b01;   else state <= 2'b00;
      endcase

  // output y at state "11"
  assign  y = state[1] & state[0]; // state[1] is the MSB
endmodule
```

The figure below shows the students design.sv for a bitstream sequence detector of "110". Then, the student created a testbench to test their bitstream sequence detector.

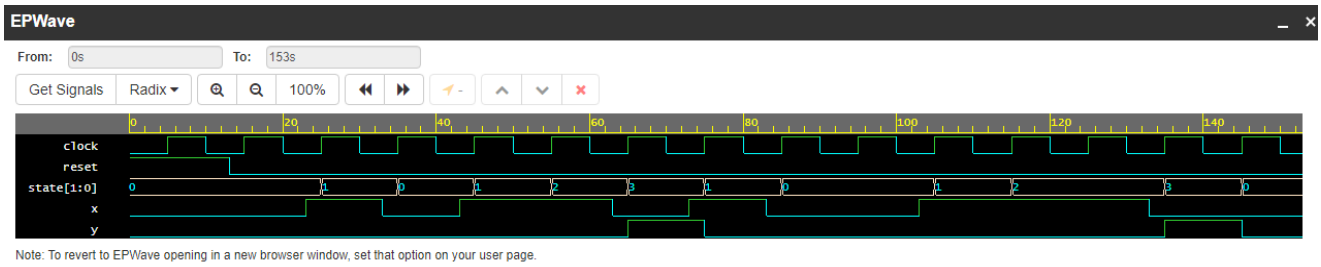
Figure 3. testbench.sv

```
// Author: Gianna Rose
// Lab 11
// https://www.edaplayground.com/x/eypm

module test;
  reg clock, x, reset;
  wire y;
  wire [1:0] state;

  // instantiate the uut
  sequence_detector uut(clock, reset, x, y, state);

  //generating the clock signal
  initial
    begin
      clock = 0; // clock starts low
      forever #5 clock = ~clock; // toggle clock
      #200 $finish; // stop simulation after 200 clock cycles
```

```
   end

  initial
    begin
      $dumpfile("dump.vcd"); $dumpvars(1,test);
      // monitor the state, x, and y signals
      $monitor("state = %b x = %b y = %b ", state,x,y);

      // Initalize inputs

      x = 0; reset = 1;

      #13  reset = 0; // hold reset low for 13 clock cycles
      #10   x = 1; // set x high for 10 clock cycles
      #10   x = 0; // set x low for 10 clock cycles
      #10   x = 1;
      #10   x = 1;
      #10   x = 0;
      #10   x = 1;
      #10   x = 0;
      #10   x = 0;
      #10   x = 1;
      #10   x = 1;
      #10   x = 1;
      #10   x = 0;
      #10   x = 0;
      #10 $finish;  // stop simulation after 200 clock cycles
    end
endmodule

//  OUTPUT
// state = 00 x = 0 y = 0
// state = 00 x = 1 y = 0
// state = 01 x = 1 y = 0
```

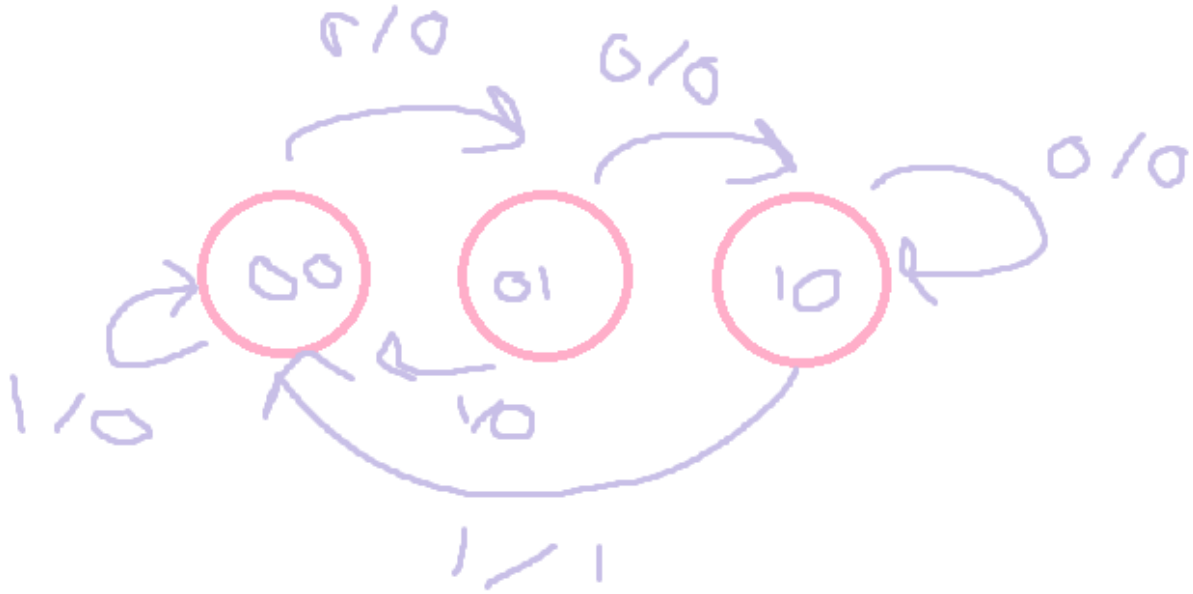
```
// state = 01 x = 0 y = 0
// state = 00 x = 0 y = 0
// state = 00 x = 1 y = 0
// state = 01 x = 1 y = 0
// state = 10 x = 1 y = 0
// state = 10 x = 0 y = 0
// state = 11 x = 0 y = 1
// state = 11 x = 1 y = 1
// state = 01 x = 1 y = 0
// state = 01 x = 0 y = 0
// state = 00 x = 0 y = 0
// state = 00 x = 1 y = 0
// state = 01 x = 1 y = 0
// state = 10 x = 1 y = 0
// state = 10 x = 0 y = 0
// state = 11 x = 0 y = 1
// state = 00 x = 0 y = 0
```

Figure 4. Waveform EP



Homework: design a FSM machine for detecting sequence “001” by deriving a state diagram and implement it using Verilog.

Link: https://www.edaplayground.com/x/9Bn4



design.sv

```
// Author: Gianna Rose
// Lab 11 hw
// https://www.edaplayground.com/x/9Bn4

module sequence_detector(clock, reset, x, y, state);
  input clock, reset, x;
  output reg y = 0;
  output reg [1:0] state;

  always @(posedge clock, posedge reset)
    if(reset)
      state <= 2'b00;
```

```
 else
  case(state)
    2'b00: if(x)begin
      state <= 2'b00;
      assign y = 0;
      end
    else
      begin
      state <= 2'b01;
        assign y = 0;
    end
    2'b01: if(x)
      begin
        state <= 2'b00;
        assign y = 0;
        end
    else
      begin
      state <= 2'b10;
      assign y = 0;
    end
    2'b10: if(x)
      begin
        state <= 2'b00;
        assign y = 1;
      end
        else
          begin
          state <= 2'b10;
                    assign y = 0;
          end
      endcase

endmodule
```

testbench.sv

```
// Author: Gianna Rose
// Lab 11 hw
// https://www.edaplayground.com/x/9Bn4

module test;
  reg clock, x, reset;
  wire y;
  wire [1:0] state;

  sequence_detector uut(clock, reset, x, y, state);

  //generating the clock signal
  initial
    begin
      clock = 0;
      forever #5 clock = ~clock;
      #200 $finish;
    end

  initial
    begin
      $dumpfile("dump.vcd"); $dumpvars(1,test);
      $monitor("state = %b x = %b y = %b ", state,x,y);

      // Initalize inputs

      x = 0; reset = 1;

      #13  reset = 0;
      #10   x = 1;
      #10   x = 0;
      #10   x = 1;
```

```
    #10   x = 1;
    #10   x = 0;
    #10   x = 1;
    #10   x = 0;
    #10   x = 0;
    #10   x = 1;
    #10   x = 1;
    #10   x = 0;
    #10   x = 0;
    #10   x = 1;
    #10   x = 0;
    #10 $finish;
  end
endmodule
```