Lab #7

Sequential Circuits in Verilog

Annelise Kezdy

Instructor: Dr. Christopher Miller

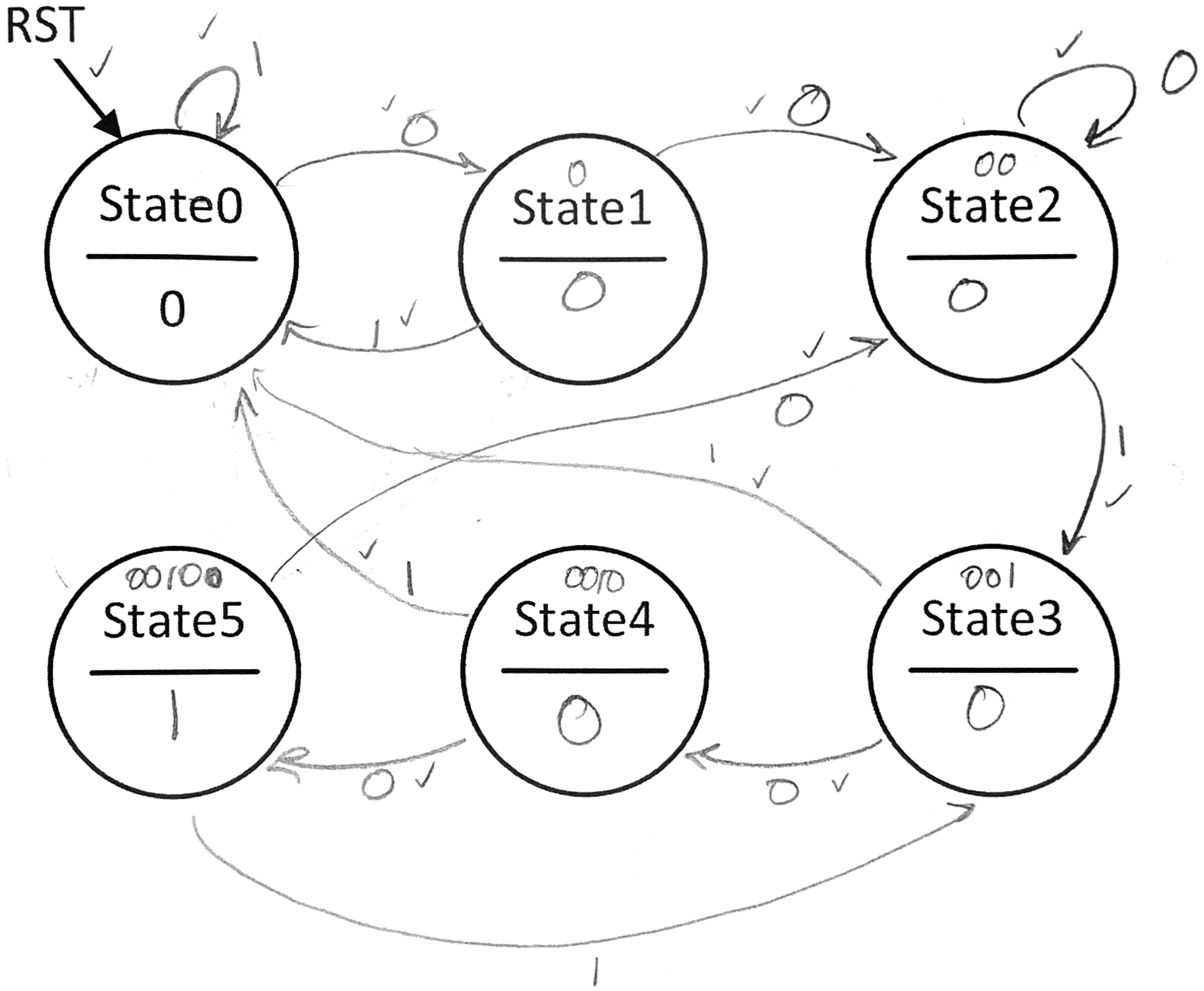
ECE 233-01

List of Attachments

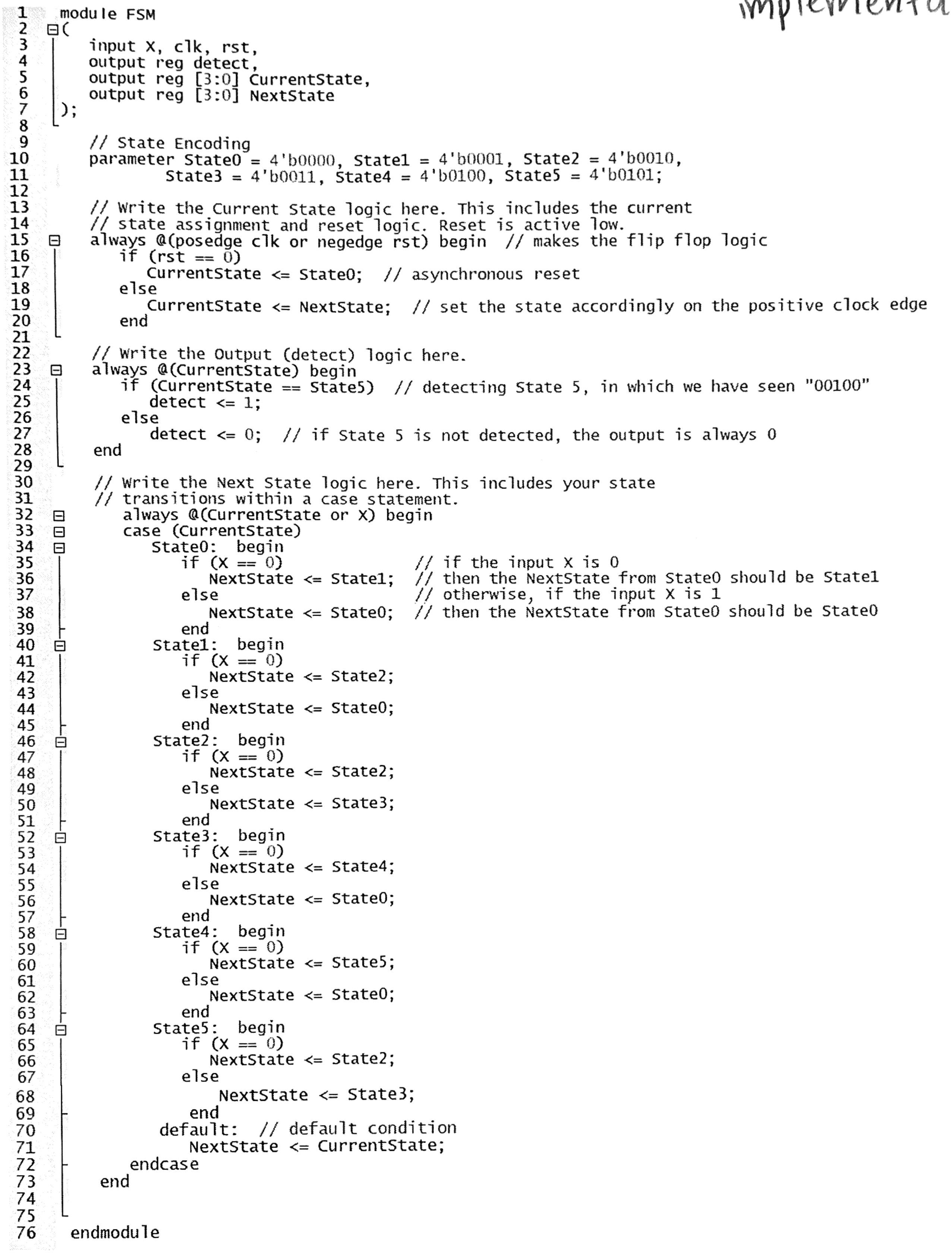
1. A cover with the lab number, lab name, your name, your instructor's name, and your lab section.
2. The state diagram on the last page of this document properly filled out.
3. The Verilog implementation and annotated functional waveform for your FSM.

Name(s): Annelise Kezdy

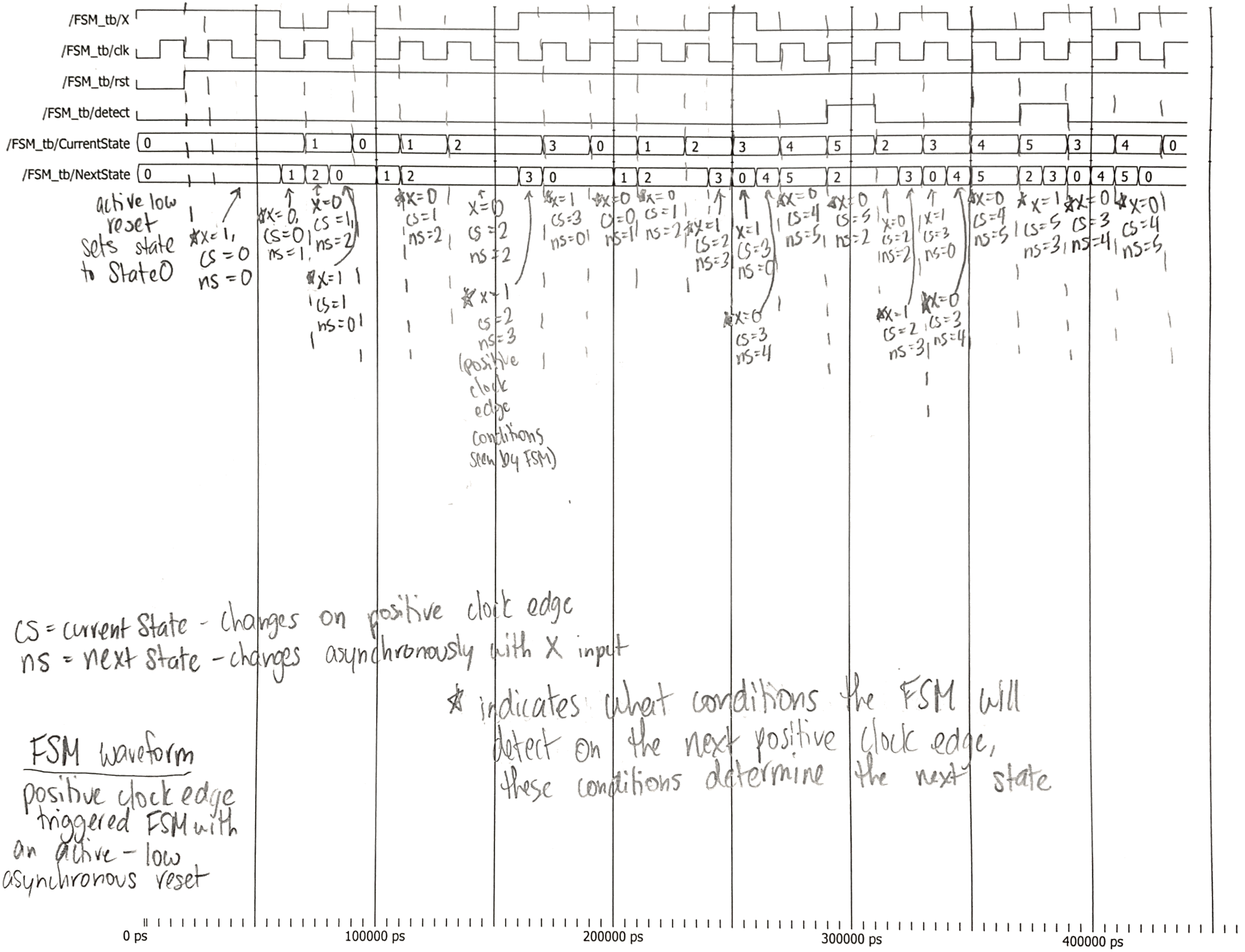
State Diagram for "00100" Pattern Detector



FSM Verilog implementation



```
module FSM
(
   input X, clk, rst,
   output reg detect,
   output reg [3:0] CurrentState,
   output reg [3:0] NextState
);

   // State Encoding
   parameter State0 = 4'b0000, State1 = 4'b0001, State2 = 4'b0010,
          State3 = 4'b0011, State4 = 4'b0100, State5 = 4'b0101;

   // Write the Current State logic here. This includes the current
   // state assignment and reset logic. Reset is active low.
   always @(posedge clk or negedge rst) begin  // makes the flip flop logic
      if (rst == 0)
         CurrentState <= State0;  // asynchronous reset
      else
         CurrentState <= NextState;  // set the state accordingly on the positive clock edge
      end

   // Write the Output (detect) logic here.
   always @(CurrentState) begin
      if (CurrentState == State5)  // detecting State 5, in which we have seen "00100"
         detect <= 1;
      else
         detect <= 0;  // if State 5 is not detected, the output is always 0
   end

   // Write the Next State logic here. This includes your state
   // transitions within a case statement.
      always @(CurrentState or X) begin
      case (CurrentState)
         State0:  begin
            if (X == 0)              // if the input X is 0
               NextState <= State1;  // then the NextState from State0 should be State1
            else                     // otherwise, if the input X is 1
               NextState <= State0;  // then the NextState from State0 should be State0
            end
         State1:  begin
            if (X == 0)
               NextState <= State2;
            else
               NextState <= State0;
            end
         State2:  begin
            if (X == 0)
               NextState <= State2;
            else
               NextState <= State3;
            end
         State3:  begin
            if (X == 0)
               NextState <= State4;
            else
               NextState <= State0;
            end
         State4:  begin
            if (X == 0)
               NextState <= State5;
            else
               NextState <= State0;
            end
         State5:  begin
            if (X == 0)
               NextState <= State2;
            else
                NextState <= State3;
             end
          default:  // default condition
             NextState <= CurrentState;
       endcase
    end


 endmodule
```



CS = current State - changes on positive clock edge
ns = next state - changes asynchronously with X input

* indicates what conditions the FSM will detect on the next positive clock edge, these conditions determine the next state

FSM waveform

positive clock edge triggered FSM with an active-low asynchronous reset

Entity:FSM_tb Architecture: Date: Wed May 19 09:27:58 EDT 2021 Row: 1 Page: 1