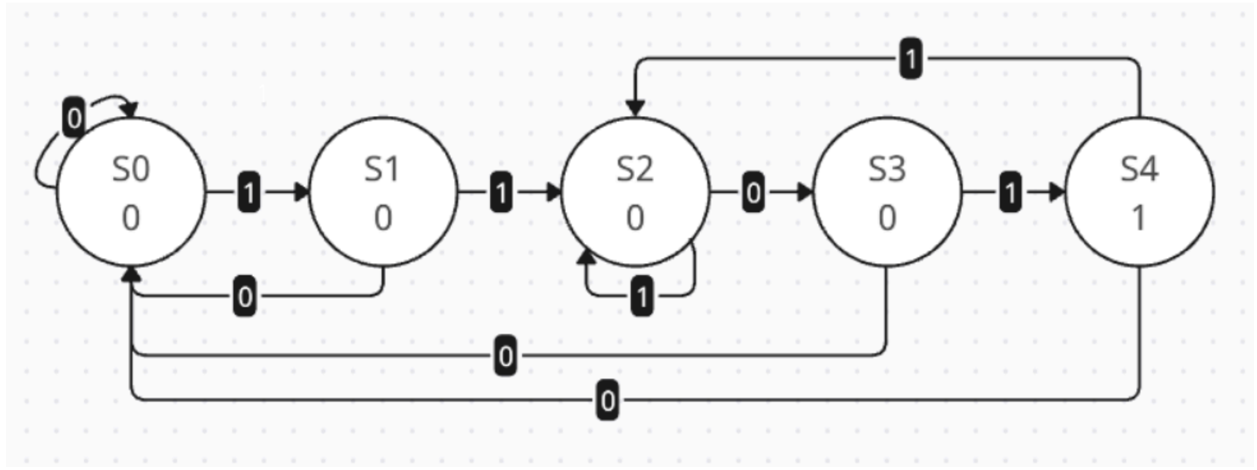
PHYS 234 Lab 5: Sequence Detector

Ethan Yang

A Mealy Machine is a finite-state machine whose output values are determined by both its current state and the current inputs, whereas a Moore Machine's output values are determined solely by its current state. They both detect the sequence 1101. We will draw the state diagram and develop the Verilog code for each machine. We will also create a Mealy Machine with a D flip-flop to make it behave like a Moore Machine.

Moore Machine

State Diagram:



S0 will be the starting point of the circuit. When the input bit is 1, the circuit advances to S1, and so on if the input sequence follows 1101. If an input bit doesn't follow the sequence , the machine resets back to S0, with an exception for S2. At S2, if the input is another 1, the machine stays at S2 because the sequence can overlap (e.g. 1101101). If the input is 1 at S4, the machine partially resets back to S2 for the same reason. Finally the circuit will output 1 if the circuit reaches S4, as shown by the number 1 below the S4 label.

Verilog Code:

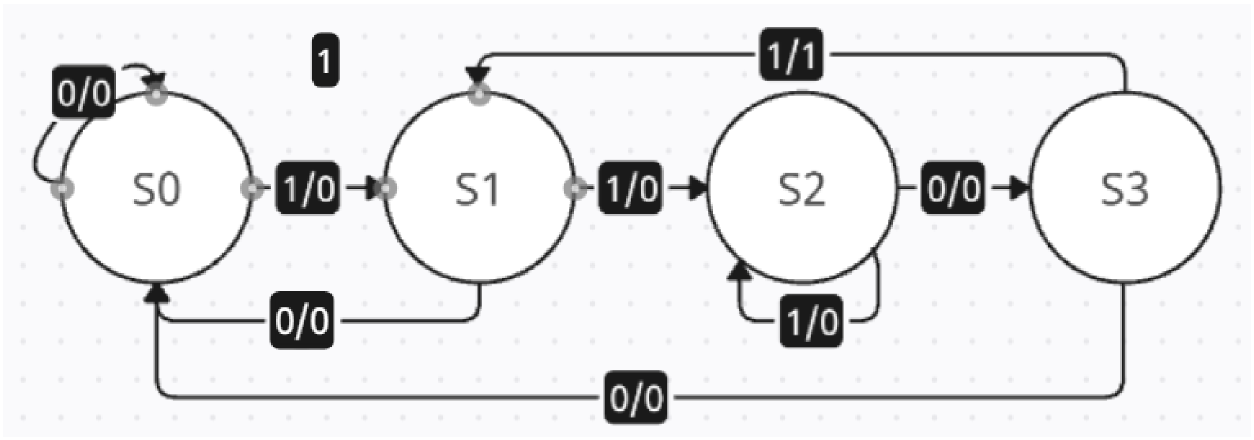
```
`timescale 1ns / 1ps

module eelab5(
    input in,
    input clk,
    output y,
    output [2:0] q
    );
reg [2:0] state, nextstate;
always @ (posedge clk) begin
```

```
          state <= nextstate;
end
always @ (*) begin
          case (state)
                    0: if (in) nextstate = 1;
                              else nextstate = 0;
                    1: if (in) nextstate = 2;
                              else nextstate = 0;
                    2: if (in) nextstate = 2;
                              else nextstate = 3;
                    3: if (in) nextstate = 4;
                              else nextstate = 0;
                    4: if (in) nextstate = 2;
                              else nextstate = 0;
                    default nextstate = 0;
          endcase
end
          assign y = (state==4);
          assign q = state;
endmodule
```

We used a case statement to determine how the state advances or resets as in the state diagram. At the rising edge of the clock, we update our current states. We also added an output q to help us determine our current states. The LED for the output y will only light up when the circuit reaches S4. For example, the LED would not light up if we input 0011, lights up if we input 001101, and dim again if we add another 0 to the input.

Mealy Machine

State Diagram:



Since a Mealy Machine is dependent on both the current state and the input, we included the outputs of the circuit to the right of the slashes on the arrows. Mealy machine works similarly to the Moore machine, just that it only outputs 1 if the circuit is at S3 AND if the input is also 1. Therefore, only the top arrow with the input at 1 returns 1, not the bottom arrow where the input is 0.

Verilog Code:

```verilog
`timescale 1ns / 1ps

module eelab5(
    input in,
    input clk,
    output y,
    output [1:0] q
    );
reg [2:0] state, nextstate;
always @ (posedge clk) begin
      state <= nextstate;
end
always @ (*) begin
      case (state)
            0: if (in) nextstate = 1;
                  else nextstate = 0;
            1: if (in) nextstate = 2;
                  else nextstate = 0;
            2: if (in) nextstate = 2;
                  else nextstate = 3;
            3: if (in) nextstate = 1;
                  else nextstate = 0;
            default nextstate = 0;
      endcase
end
      assign y = (in&(state==3));
      assign q = state;
endmodule
```

The Verilog code is identical to the Moore Machine, only with the addition on the "assign y" line, where we made sure that the circuit outputs 1 only if the input also is 1. As a result, the

LED only lights up only when the input switch is 1 and the correct sequence 1101 is inputted/the circuit is at S3.

Mealy Machine with a D Flip-flop
A D flip-flop is also known as a “data” flip-flop or a “delay” flip-flop. As its name suggests, it takes the state of the D input at the moment of a rising edge of the clock and delays it by one clock cycle.

Verilog Code:

```
`timescale 1ns / 1ps

module eelab5(
    input in,
    input clk,
    output reg y,
    output [1:0] q
    );
reg [2:0] state, nextstate;
wire out;
always @ (posedge clk) begin
      state <= nextstate;
end
always @ (*) begin
      nextstate = state;
      case (state)
            0: if (in) nextstate = 1;
                  else nextstate = 0;
            1: if (in) nextstate = 2;
                  else nextstate = 0;
            2: if (in) nextstate = 2;
                  else nextstate = 3;
            3: if (in) nextstate = 1;
                  else nextstate = 0;
            default nextstate = 0;
            endcase
      end
      assign out = (in&(state==3));
      assign q = state;
always @ (posedge clk) begin
```

```
      y <= out;
end
endmodule
```

By delaying the “out” variable, we can trick the Mealy machine into acting like a Moore machine. The output changes only after the clock instead of instantly with the input, thus the LED would light up even if the output is 0 at S3. As a result, the Mealy machine with a D flip-flop works like a Moore machine, lighting up the LED whenever the circuit is in S3.