# Solutions PDF Generated from: solutions-openai-generated/labs/lab-04-solutions-set-02.json

## Question A

Design a Finite State Machine to mimic the performance of a 1965 Ford Thunderbirds taillights. There are three inputs  left, right, and hazard. If left is the input, the left set of LEDs should progressively light up and then go dark, repeating until the signal is no longer applied. Similar logic will apply to the right LEDs. If the hazard input switch is on, all of the lights should blink on and off until the input is changed. Write out a Next State Logic and Output Logic circuit. The three inputs will be the left, right, and hazard switches. The outputs will be the 6 or 8 LEDs that you choose to use for displaying the patterns.

```
Next State Logic:

IF Left = 1 THEN
State A
ELSE IF Right = 1 THEN
State B
ELSE IF Hazard = 1 THEN
State C

Output Logic :

State A :
LEDs 1-3 On then LEDs 1-3 Off (repeating until the Left input is no longer applied)

State B :
LEDs 4-6 On then LEDs 4-6 Off (repeating until the Right input is no longer applied)

State C :
All LEDs On then All LEDs Off (repeating until the Hazard input is no longer applied)
```

## Question B

Write Verilog code to realize the Finite State Machine for the taillights operation. There are three inputs left, right, and hazard. If left is the input, the left set of LEDs should progressively light up and then go dark, repeating until the signal is no longer applied. Similar logic will apply to the right LEDs. If the hazard input switch is on, all of the lights should blink on and off until the input is changed.

```
//Finite State Machine for Taillight Operation
module taillight_control(left, right, hazard, leftlight, rightlight);
  input left, right, hazard;
  output [3:0] leftlight, rightlight;
  reg [3:0] leftlight, rightlight;
  reg [1:0] mainstate, leftstate, rightstate;
  initial begin
```

```
  mainstate <= 0;
  leftstate <= 0;
  rightstate <= 0;
  leftlight <= 0;
  rightlight <= 0;
end

always @ (posedge left or posedge right or posedge hazard)
  begin
    if (hazard)
      mainstate <= 0;
    else
      mainstate <= 1;
  end

always @ (posedge left or posedge right or posedge hazard)
  begin
    case (mainstate)
      0: begin
           leftlight[3:0] <= 4'b0000;
           rightlight[3:0] <= 4'b0000;
         end
      1: begin
           if (left)
             leftstate <= leftstate + 1;
           if (right)
             rightstate <= rightstate + 1;
         end
    endcase
  end

always @ (posedge left or posedge right or posedge hazard)
  begin
    case (mainstate)
      0: begin
           for (int i = 0; i < 4; i++)
             begin
               if (i == 0)
                 leftlight[i] <= 1;
               else
                 leftlight[i] <= 0;
               if (i == 0)
                 rightlight[i] <= 1;
               else
                 rightlight[i] <= 0;
             end
         end
      1: begin
           case (leftstate)
             0:   leftlight[3:0] <= 4'b0000;
             1:   leftlight[3:0] <= 4'b0001;
             2:   leftlight[3:0] <= 4'b0011;
             3:   leftlight[3:0] <= 4'b0111;
             4:   leftlight[3:0] <= 4'b1111;
           endcase
           case (rightstate)
             0:   rightlight[3:0] <= 4'b0000;
             1:   rightlight[3:0] <= 4'b0001;
             2:   rightlight[3:0] <= 4'b0011;
```

```
            3:    rightlight[3:0] <= 4'b0111;
            4:    rightlight[3:0] <= 4'b1111;
          endcase
        end
    endcase
  end

endmodule
```

## Execution Time

0:00:28.777200

## OpenAI Parameters

Model: text-davinci-003, Max. Tokens: 1024, Temperature: 1, N: 1