```
1
 2
     Name Lamin Jammeh
 3
     CLass: EE417 Summer 2024
     Lesson 04 HW Question 1
 5
    Group: Ron Kalin/ Lamin Jammeh
 6
     Project Description: This is the main module for a Mealy FSM that receives a
7
     set of bits and checks if they are equa to Z1=010 or Z2=100 and takes adecision
     on whether to reset the machine of keeps look receiving data and comparing them
8
9
10
      module Sequence_100_Detector_Mealy (z2, z1, clk, S_reset, x_in);
11
12
      //define the inputs and outputs of the system
13
      output reg z_2, z_1;
14
                 clk, S_reset;
      input
15
      input
                 x_in;
16
17
      //define the states
18
      reg [2:0] present_state, next_state;
19
20
      //define the possible parameter
      parameter S_000 = 3'b000;
21
      parameter S_001 = 3'b001;
22
23
      parameter S_010 = 3'b010;
                                       //z1
24
      parameter S_{011} = 3'b011;
25
      parameter S_100 = 3'b100;
                                       //z2
26
      parameter S_101 = 3'b101;
27
      parameter S_110 = 3'b110;
28
      parameter S_111 = 3'b111;
29
30
      //sequenctial logic updating the state register (flip flop)
31
      always @(posedge clk)
32
           if (S_reset) present_state <= S_000;</pre>
33
           else
                      present_state <= next_state;</pre>
34
35
     //combination logic determining the next_state and the output state
36
                                    //trigger the block when there is any change in the signals
     always @ *
     used in the block
37
           case (present_state)
38
           S_000
                    : begin
                       z1 = 1'b0;
39
40
                       z2 = 1'b0;
41
                       if (x_in) next_state = S_001;
42
                       else next_state = S_000;
43
                    end
44
           s 001
                    : begin
45
                       z1 = 1'b0;
46
                       z2 = 1'b0;
47
                       if (x_in) next_state = S_010;
48
                       else
                                next_state = S_001;
49
                    end
                    : begin
50
           S_010
51
                       z1 = 1'b1;
                       z2 = 1'b0;
52
53
                       if (x_in) next_state = s_011;
54
                                 next_state = S_010;
                       else
55
                    end
56
           S_011
                    : begin
57
                       z1 = 1'b0;
                       z2 = 1'b0;
58
59
                       if (x_in) next_state = S_100;
60
                                 next_state = S_011;
                        else
61
                    end
62
           S_100
                    : begin
63
                       z1 = 1'b0;
64
                       z2 = 1'b1;
                       if (x_in) next_state = S_101;
65
66
                                  next_state = S_100;
                       else
67
                    end
68
           S_101
                    : begin
```

```
69
                              z1 = 1'b0;
70
                              z2 = 1'b0;
                              if (x_in) next_state = S_110;
71
72
                              else
                                          next_state = S_100;
73
                          end
74
              S_110
                          : begin
                              z1 = 1'b0;

z2 = 1'b0;

if (x_in) next_state = S_111;
75
76
77
                              else
78
                                          next_state = S_110;
79
                          end
80
              S_111
                          : begin
                             z1 = 1'b0;
z2 = 1'b0;
if (x_in) next_state = S_111;
else next_state = S_111;
81
82
83
84
85
                          end
86
              endcase
87
      endmodule
```