```verilog
/*-----------------------------------------------------------------------------
Name Lamin Jammeh
CLass: EE417 Summer 2024
Lesson 04 HW Question 1
Group: Ron Kalin/ Lamin Jammeh
Project Description: This is the main module for a Mealy FSM that receives a
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
-------------------------------------------------------------------------------*/
 module Sequence_100_Detector_Mealy (z2, z1, clk, S_reset, x_in);

 //define the inputs and outputs of the system
 output reg z2, z1;
 input      clk, S_reset;
 input      x_in;

 //define the states
 reg [2:0] present_state, next_state;

 //define the possible parameter
 parameter  S_000 = 3'b000;
 parameter  S_001 = 3'b001;
 parameter  S_010 = 3'b010;     //z1
 parameter  S_011 = 3'b011;
 parameter  S_100 = 3'b100;     //z2
 parameter  S_101 = 3'b101;
 parameter  S_110 = 3'b110;
 parameter  S_111 = 3'b111;

 //sequenctial logic updating the state register (flip flop)
 always @(posedge clk)
      if (S_reset) present_state <= S_000;
      else       present_state <= next_state;

//combination logic determining the next_state and the output state
always @ *                    //trigger the block when there is any change in the signals used in the block
      case (present_state)
      S_000    : begin
                  z1 = 1'b0;
                  z2 = 1'b0;
                  if (x_in) next_state = S_001;
                  else      next_state = S_000;
               end
      S_001    : begin
                  z1 = 1'b0;
                  z2 = 1'b0;
                  if (x_in) next_state = S_010;
                  else      next_state = S_001;
               end
      S_010    : begin
                  z1 = 1'b1;
                  z2 = 1'b0;
                  if (x_in) next_state = S_011;
                  else      next_state = S_010;
               end
      S_011    : begin
                  z1 = 1'b0;
                  z2 = 1'b0;
                  if (x_in) next_state = S_100;
                  else      next_state = S_011;
               end
      S_100    : begin
                  z1 = 1'b0;
                  z2 = 1'b1;
                  if (x_in) next_state = S_101;
                  else      next_state = S_100;
               end
      S_101    : begin
```

```
                  z1 = 1'b0;
                  z2 = 1'b0;
                  if (x_in) next_state = S_110;
                  else      next_state = S_100;
               end
      S_110    : begin
                  z1 = 1'b0;
                  z2 = 1'b0;
                  if (x_in) next_state = S_111;
                  else      next_state = S_110;
               end
      S_111    : begin
                  z1 = 1'b0;
                  z2 = 1'b0;
                  if (x_in) next_state = S_111;
                  else      next_state = S_111;
               end
      endcase
endmodule
```