```
module register(clk, rst, d, q, load);
  input clk, rst, load;
  input [7:0] d;
  output [7:0] q;
  reg [7:0] q, qNext;

  always @ (posedge clk) begin
    if (rst)
      q <= 8'b0;
    else
      q <= qNext;
  end

  always @ (d or q or load) begin
    if (load)
      qNext = d;
    else
      qNext = q;
  end
endmodule

module fsmFred(clk, rst, i, Z1, Z2);
  parameter ST_A = 11'b00000000001;
  parameter ST_B = 11'b00000000010;
  parameter ST_C = 11'b00000000100;
  parameter ST_D = 11'b00000001000;
  parameter ST_E = 11'b00000010000;
  parameter ST_F = 11'b00000100000;
  parameter ST_G = 11'b00001000000;
  parameter ST_H = 11'b00010000000;
  parameter ST_I = 11'b00100000000;
  parameter ST_J = 11'b01000000000;
  parameter ST_K = 11'b10000000000;

  input clk, rst, i;
  output Z1, Z2;
  reg Z1, Z2;
  reg [10:0] state, nextState;

  always @ (posedge clk) begin
    if (rst)
      state <= ST_A;
    else
      state <= nextState;
  end

  always @ (i or state) begin
    Z1 = 0;
    Z2 = 0;
    case (state)
      ST_A: begin
        if (i)  nextState = ST_B;
        else    nextState = ST_C;
      end
      ST_B: begin
        if (i)  nextState = ST_F;
        else    nextState = ST_D;
      end
      ST_C: begin
        if (i)  nextState = ST_G;
        else    nextState = ST_E;
      end
      ST_D: begin
        if (i) begin
          nextState = ST_G;
```

```
          Z1 = 1;
        end
        else    nextState = ST_E;
      end
      ST_E: begin
        if (i)  nextState = ST_G;
        else    nextState = ST_E;
      end
      ST_F: begin
        if (i)  nextState = ST_F;
        else    nextState = ST_D;
      end
      ST_G: begin
        if (i) begin
          nextState = ST_H;
          Z2 = 1;
        end
        else    nextState = ST_D;
      end
      ST_H: begin
        if (i)  nextState = ST_H;
        else    nextState = ST_J;
      end
      ST_I: begin
        if (i) begin
          nextState = ST_H;
          Z2 = 1;
        end
        else    nextState = ST_J;
      end
      ST_J: begin
        if (i)  nextState = ST_I;
        else    nextState = ST_K;
      end
      ST_K: begin
        if (i)  nextState = ST_I;
        else    nextState = ST_K;
      end
    endcase
  end
endmodule

module fredTester();
  reg clk, rst, i;
  wire z1, z2;

  fsmFred fred1(clk, rst, i, z1, z2);

  always begin
    #10;
    clk = ~clk;
  end

  integer j, k;

  initial begin
    rst = 1;
    clk = 0;
    i = 0;
    #100;
    rst = 0;

    i = 1; #20;
    i = 0; #20;
    i = 1; #20;
```

```
    i = 0; #20;
    i = 1; #20;
    i = 0; #20;
    i = 0; #20;
    i = 0; #20;
    i = 1; #20;
    i = 1; #20;
    i = 1; #20;

    for (j = 0; j < 1024; j = j + 1) begin
      rst = 1; #20;
      rst = 0;
      for (k = 0; k < 10; k = k + 1) begin
        i = j[k]; #20;
      end
    end
  end

endmodule
```