```
// SW[0]:       reset signal
// SW[1]:       input signal (w)

// KEY[0]:      clock

// LEDR[2:0]:   current state
// LEDR[9]:     output (z)

module sequence_detector(SW, KEY, LEDR);
    input [9:0] SW;
    input [3:0] KEY;
    output [9:0] LEDR;

    wire w, clock, resetn, z;

    reg [2:0] y_Q, Y_D; // y_Q represents current state, Y_D represents next state

    localparam A = 3'b000, B = 3'b001, C = 3'b010, D = 3'b011, E = 3'b100, F = 3'b101, G = 3'b110;

    // Connect inputs and outputs to internal wires
    assign w = SW[1];
    assign clock = ~KEY[0];
    assign resetn = SW[0];
    assign LEDR[9] = z;
    assign LEDR[2:0] = y_Q;

    // State table
    // The state table should only contain the logic for state transitions
    // Do not mix in any output logic.  The output logic should be handled separately.
    // This will make it easier to read, modify and debug the code.
    always @(*)
    begin   // Start of state_table
        case (y_Q)
            A: begin
                    if(!w) Y_D = A;
                    else Y_D = B;
               end
            B: begin
                    if(!w) Y_D = A;
                    else Y_D = C;
               end
            C: begin
                    if(!w) Y_D = E;
                    else Y_D = D;
               end
            D: begin
                    if(!w) Y_D = E;
                    else Y_D = F;
               end
            E: begin
                    if(!w) Y_D = A;
                    else Y_D = G;
               end
            F: begin
                    if(!w) Y_D = E;
                    else Y_D = F;
               end
            G: begin
                    if(!w) Y_D = A;
                    else Y_D = C;
               end
            default: Y_D = A;
        endcase
    end     // End of state_table

    // State Register (i.e., FFs)
    always @(posedge clock)
    begin   // Start of state_FFs (state register)
        if(resetn == 1'b0)
            y_Q <= A;
        else
            y_Q <= Y_D;
    end     // End of state_FFs (state register)

    // Output logic
    // Set z to 1 to turn on LED when in relevant states
    assign z = ((y_Q == F)|| (y_Q == G));
endmodule
```