```verilog
`timescale 1ns / 1ns // `timescale time_unit/time_precision

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

            wire w, clock, resetn, out_light;
            // y_Q represents current state, Y_D represents next state
            reg [3:0] y_Q, Y_D;
            localparam  A = 4'b0000, B = 4'b0001, C = 4'b0010, D = 4'b0011,
                        E = 4'b0100, F = 4'b0101, G = 4'b0110;

            /*
             * SW[0] reset when 0
             * SW[1] input signal
             * KEY[0] clock signal
             * LEDR[3:0] displays current state
             * LEDR[9] displays output
            */

            assign w = SW[1];
            assign clock = ~KEY[0];
            assign resetn = SW[0];

            always@(*) // state table
            begin
               // logic for state transitions
               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 // state_table

            // state Registers
            always @(posedge clock)
            begin: state_FFs
              if(resetn == 1'b0)
                  y_Q <=  A; // reset to state A
              else
                  y_Q <= Y_D;
            end // state_FFS

            // output logic
            assign out_light = ((y_Q == F) | (y_Q == G));

            assign LEDR[9] = out_light;
            assign LEDR[3:0] = y_Q;
            assign LEDR[8:4] = 5'd0;

endmodule // sequence_detector
```