```
module DE1_SoC (CLOCK_50, HEX0, HEX1, HEX2, HEX3, HEX4, HEX5, KEY, LEDR, SW);
   input  logic CLOCK_50; // 50MHz clock.
   output logic [6:0] HEX0, HEX1, HEX2, HEX3, HEX4, HEX5;
   output logic [9:0] LEDR;
   input  logic [3:0] KEY; // True when not pressed, False when pressed
   input  logic [9:0] SW;

   // Generate clk off of CLOCK_50, whichClock picks rate.
   logic [31:0] clk;
   parameter whichClock = 25;
   clock_divider cdiv (.clock(CLOCK_50),.reset(reset),.divided_clocks(clk));

   // Hook up FSM inputs and outputs.
   logic reset;
   assign reset = ~KEY[0]; // Reset when KEY[0] is pressed.

   lights l (.clk(clk[whichClock]), .reset, .in(SW[1:0]), .out(LEDR[2:0]));

   // Show signals on LEDRs so we can see what is happening.
   assign LEDR[9] = clk[whichClock];
   assign LEDR[7] = reset;

endmodule

// divided_clocks[0] = 25MHz, [1] = 12.5Mhz, ... [23] = 3Hz, [24] = 1.5Hz, [25] = 0.75Hz, ...
module clock_divider (clock, reset, divided_clocks);
   input logic reset, clock;
   output logic [31:0] divided_clocks = 0;

   always_ff @(posedge clock) begin
      divided_clocks <= divided_clocks + 1;
   end
endmodule
```