```
// File: heartbeat.v
// John Hubbard, 16 Feb 205
// For HW #4 assignment of FPGA class.

//
// Heart beat display: problem 3.9.8
//

module choose_pattern_from_count(count, seg, an);
    input [2:0] count;
    output [6:0] seg; // abcdefg
    output [0:3] an;

    reg [6:0] seg;
    reg [0:3] an;

    always @(*)
        case (count)
            0:
            begin
                seg = 7'b0011100; // abgf
                an  = 4'b0111;    // leftmost
            end
            1:
            begin
                seg = 7'b0011100; // abgf
                an  = 4'b1011;    // left center
            end
            2:
            begin
                seg = 7'b0011100; // abgf
                an  = 4'b1101;    // right center
            end
            3:
            begin
                seg = 7'b0011100; // abgf
                an  = 4'b1110;    // rightmost
            end
            4:
            begin
                seg = 7'b1100010; // cdeg
                an  = 4'b1110;    // rightmost
            end
            5:
            begin
                seg = 7'b1100010; // cdeg
                an  = 4'b1101;    // right center
            end
            6:
            begin
                seg = 7'b1100010; // cdeg
                an  = 4'b1011;    // left center
            end
            7:
```

```
            begin
                seg = 7'b1100010; // cdeg
                an  = 4'b0111;    // leftmost
            end
        endcase

endmodule

module heartbreat_top_module(clk, btnU, seg, an);
    parameter C = 28; //27..0 counter
    parameter N7 = 7 ;
    parameter N4 = 4 ;
    parameter CRYSTAL = 100 ; // 100 MHZ
    parameter NUM_SEC = 1 ;
    parameter STOPAT = (CRYSTAL * 1_000_000 * NUM_SEC)- 1 ;

    input clk, btnU;
    output [0:6] seg; // abcdefg
    wire   [2:0] zero_to_8_counter;
    output [3:0] an;
    wire [C-1:0] big_counter;
    wire one_second_clock;

    mod_counter #(C,STOPAT) MOD_COUNTER(clk, btnU, big_counter, one_second_clock);
    counter #(3) COUNTER(clk, btnU, one_second_clock, zero_to_8_counter);

    choose_pattern_from_count PATTERN(zero_to_8_counter, seg, an);
endmodule
```