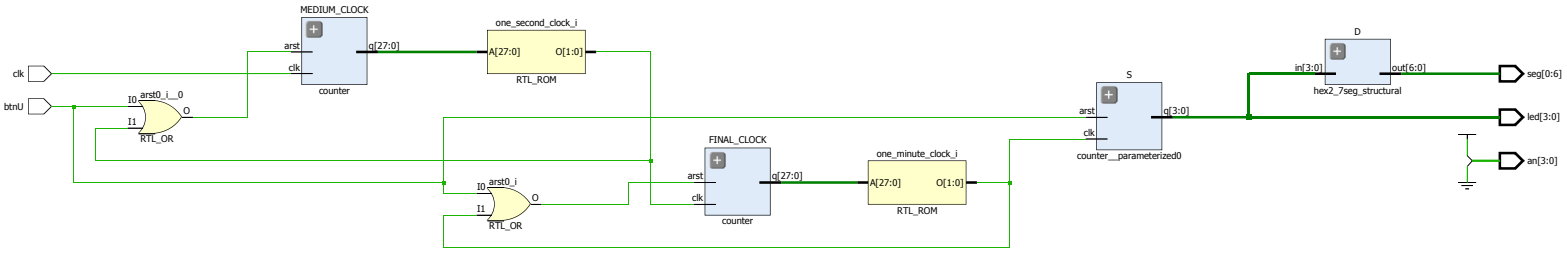
```
// file hex2_7seg.v
// Modified by John Hubbard, 31 Jan 2015
// Added hex2_7seg_structural module, and tested by instantiating into
// the switch-based test (and also the one-minute counter).
//
// For HW #3 assignment of FPGA class.

module hex2_7seg(in,out) ;
    input [3:0] in ; /*3210 */
    output [6:0] out ; /* abcdefg */
    reg [6:0] out ;

    always @(*)
    begin
        case (in)
             /*3210*/ /*abcdefg */
             0:  out = 7'b0000001 ;
             1:  out = 7'b1001111 ;
             2:  out = 7'b0010010 ;
             3:  out = 7'b0000110 ;
             4:  out = 7'b1001100 ;
             5:  out = 7'b0100100 ;
             6:  out = 7'b0100000 ;
             7:  out = 7'b0001111 ;
             8:  out = 7'b0000000 ;
             9:  out = 7'b0000100 ;
             10: out = 7'b0001000 ;
             11: out = 7'b1100000 ;
             12: out = 7'b0110001 ;
             13: out = 7'b1000010 ;
             14: out = 7'b0110000 ;
             15: out = 7'b0111000 ;
            default: out = 7'bx ;
         endcase
     end
endmodule

module hex2_7seg_structural(in,out) ;
    input [3:0] in ; /*3210 */
    output [6:0] out ; /* abcdefg */
    wire [6:0] out ;

    LUT4 #(16'h1083) L0(out[0], in[0], in[1], in[2], in[3]);
    LUT4 #(16'h208e) L1(out[1], in[0], in[1], in[2], in[3]);
    LUT4 #(16'h02ba) L2(out[2], in[0], in[1], in[2], in[3]);
    LUT4 #(16'h8492) L3(out[3], in[0], in[1], in[2], in[3]);
    LUT4 #(16'hd004) L4(out[4], in[0], in[1], in[2], in[3]);
    LUT4 #(16'hd860) L5(out[5], in[0], in[1], in[2], in[3]);
    LUT4 #(16'h2812) L6(out[6], in[0], in[1], in[2], in[3]);
endmodule

module a7_hex2_7seg(sw,seg,an) ;
    input [3:0] sw ; /* 3210 */
    output [0:6] seg; /* abcdefg */
```

```
    output [3:0] an ; /* 3210 */

    assign an = 4'b0010 ; /* ON ON OFF ON */
    // hex2_7seg U(.in(sw),.out(seg) );
    hex2_7seg_structural U(.in(sw),.out(seg) );
endmodule
```

```
 //File onesecond_bad.v
 //hex2_7seg.v onesecond_bad.v basys3.xdc
// Modified by John Hubbard, 31 Jan 2015
// Added an intermediate clock, so that there is one for seconds, and that
// clock cascades into the per-minute clock. This avoids overflowing the
// parameter (which appears to be a 32-bit value).
//
// For HW #3 assignment of FPGA class.

module counter(clk,arst,q) ;
     parameter N = 7 ;
     input clk,arst ;
     output [N-1:0] q ;
     reg [N-1:0] q ;

    always @(posedge clk or posedge arst)
        if (arst == 1'b1)
            q <= 0 ;
        else
            q <= q + 1 ;
endmodule

module onesecond_bad(clk, btnU, seg, an, led) ;
    parameter C = 28 ; //28-1 = 27. Note counter is also N-1
    parameter N7 = 7 ;
    parameter N4 = 4 ;
    parameter CRYSTAL = 100 ; // 100 MHZ

    parameter STOPAT_SEC_PER_MIN = 60;
    parameter STOPAT_ONE_SEC = (CRYSTAL * 1_000_000 * 1)- 1;

    input clk, btnU;
    output [0:N7-1] seg ;
    output [N4-1:0] an ;
    output [N4-1:0] led ;
    wire [C-1:0] min_counter ;
    wire [C-1:0] sec_counter ;
    wire [N4-1:0] zero_to_f_counter ;
    wire one_second_clock ;
    wire one_minute_clock ;

    assign an = 4'b0010 ; /* ON ON OFF ON */
    assign one_second_clock = (sec_counter == STOPAT_ONE_SEC) ? 1: 0 ;
    assign one_minute_clock = (min_counter == STOPAT_SEC_PER_MIN) ? 1: 0 ;

    counter #C FINAL_CLOCK(one_second_clock,(btnU|| one_minute_clock),min_counter) ;
    counter #C MEDIUM_CLOCK(clk,(btnU|| one_second_clock),sec_counter) ;
    counter #N4 S(one_minute_clock,btnU,zero_to_f_counter) ;

    hex2_7seg D(zero_to_f_counter,seg);

    assign led = zero_to_f_counter;
endmodule
```


MEDIUM_CLOCK
arst
clk
q[27:0]
counter
one_second_clock_i
A[27:0]
O[1:0]
RTL_ROM
clk
btnU
I0
arst0_i_0
I1
O
RTL_OR
FINAL_CLOCK
arst
clk
q[27:0]
counter
one_minute_clock_i
A[27:0]
O[1:0]
RTL_ROM
I0
arst0_i
I1
O
RTL_OR
S
arst
clk
q[3:0]
counter__parameterized0
D
in[3:0]
out[6:0]
hex2_7seg_structural
seg[0:6]
led[3:0]
an[3:0]