```
module Top_Module_Main(
    input clkin,
    input btnR,
    input btnU,
    input btnC,
    output [15:0] led,
    output [3:0] an,
    output dp,
    output [6:0] seg
    );
    wire clk, digsel, qsec;
    wire [15:0] bit16out;
    lab5_clks sys_clock
(.clkin(clkin), .greset(btnR),
.clk(clk), .digsel(digsel),
.qsec(qsec));

    wire four_secs, two_secs, match,
delay, delay_4sec;

    wire show_num, reset_timer,
```

```
run_game, scored, flash_both,
flash_alt;
    wire edge_C, edge_U;
    Edge_Detector bC (.clk(clk),
.btn(btnC), .out(edge_C));
    Edge_Detector bU (.clk(clk),
.btn(btnU), .out(edge_U));
    StateMachine state_machine
(.Go(edge_C), .Stop(edge_U),
.FourSecs(four_secs),
.TwoSecs(two_secs), .Match(match),
.ShowNum(show_num),

.ResetTimer(reset_timer),
.RunGame(run_game), .Scored(scored),
.Delay(delay),

.Sec4Delay(delay_4sec),
.FlashBoth(flash_both),
.FlashAlt(flash_alt), .clk(clk));
```

```
    Two_Second_Delay  two_sec_delay
(.clk(clk), .CE(delay),
.Reset(run_game),  .Qsec(qsec),
.Signal(two_secs));
    Four_Second_Delay four_sec_delay
(.clk(clk), .CE(delay_4sec),
.Reset(run_game),  .Qsec(qsec),
.Signal(four_secs));


    wire [7:0] CMPUTR, USER;
    wire [7:0] rand_ints;
    LFSR rand_num (.clk(clk),
.CE(!show_num), .Q(rand_ints));

   wire btnC_ED;
   Edge_Detector ED (.clk(clk),
.btn(btnC), .out(btnC_ED));
   Game_Counter game_cnt (.CE(btnC_ED
& !show_num), .R(1'b0),
.in(rand_ints), .clk(clk),
```

```
.Q(CMPUTR));   //from LFSR changed to
reset_timer

    wire [7:0]equal;
    wire rstTimer;
    Edge_Detector timerst (.clk(clk),
.btn(run_game), .out(rstTimer));
    Time_Counter time_cnt (.CE(qsec),
.R(rstTimer), .run(run_game),
.clk(clk), .Q(USER)); //added a RESET
    assign equal[7] = !(CMPUTR[7] ^
USER[7]);
    assign equal[6] = !(CMPUTR[6] ^
USER[6]);
    assign equal[5] = !(CMPUTR[5] ^
USER[5]);
    assign equal[4] = !(CMPUTR[4] ^
USER[4]);
    assign equal[3] = !(CMPUTR[3] ^
USER[3]);
    assign equal[2] = !(CMPUTR[2] ^
```

```
USER[2]);
    assign equal[1] = !(CMPUTR[1] ^
USER[1]);
    assign equal[0] = !(CMPUTR[0] ^
USER[0]);
    assign match =
equal[0]&equal[1]&equal[2]&

equal[3]&equal[4]&equal[5]
                    &equal[6]&equal[7];

    assign bit16out[15:8] = CMPUTR;
    assign bit16out[7:0] = USER;


    wire [15:0]led_shift;
    wire scored_ED;
    Edge_Detector score_ED
(.clk(clk), .btn(scored),
.out(scored_ED));
    LED_Shifter led_lights
```

```
(.In(1'b1), .CE(scored_ED),
.clk(clk), .Q(led_shift));

    //FLASH ALT
    wire alt1, alt2, both1, both2;
    FDRE #(.INIT(1'b1) ) ffA1
(.C(clk), .R(1'b0),
.CE(qsec&flash_alt), .D(alt1),
.Q(alt2));
    FDRE #(.INIT(1'b0) ) ffA2
(.C(clk), .R(1'b0),
.CE(qsec&flash_alt), .D(alt2),
.Q(alt1));

    //FLASH_BOTH
    FDRE #(.INIT(1'b1) ) ffB1
(.C(clk), .R(1'b0),
.CE(qsec&flash_both), .D(both2),
.Q(both1));
    FDRE #(.INIT(1'b0) ) ffB2
(.C(clk), .R(1'b0),
```

```
.CE(qsec&flash_both), .D(both1),
.Q(both2));

    //below is from lab 4
    wire [3:0]Qring;
    RingCounter ring_cntr
(.digsel(digsel), .clk(clk),
.Q(Qring));

    assign led = led_shift;
    assign an[3] =
!(show_num&(Qring[3] & !flash_both &
!flash_alt) | (Qring[3] &alt1 &
!flash_both & flash_alt) | (Qring[3]
&both1 & flash_both & !flash_alt));
    assign an[2] =
!(show_num&(Qring[2] & !flash_both &
!flash_alt) | (Qring[2] &alt1 &
!flash_both & flash_alt) | (Qring[2]
&both1 & flash_both & !flash_alt));
    assign an[1] = !((Qring[1] &
```

```
!flash_both & !flash_alt) | (Qring[1]
&alt2 & !flash_both & flash_alt) |
(Qring[1] &both1 & flash_both &
!flash_alt));
    assign an[0] = !((Qring[0] &
!flash_both & !flash_alt) | (Qring[0]
&alt2 & !flash_both & flash_alt) |
(Qring[0] &both1 & flash_both &
!flash_alt));

    wire [3:0]sel;
    Selector select(.sel(Qring),
.N(bit16out), .H(sel));
    hex7seg segment_disp (.n(sel),
.seg(seg));

endmodule
```