```
`timescale 1ns / 1ps

module Top_Level(
    input clkin, btnR, btnU, btnD,
    output [15:0] led,
    output [3:0] an,
    output dp,
    output [6:0] seg
    );

    assign led [15:8] = 1'b0;
    assign dp = 1'b1;
    assign an[1] = 1'b1;
    assign an[2] = 1'b1;
    // State Machine Stuff
    wire Go, Pick, TimeUp, Match;   // Inputs
    wire RstTime, LDTarget, IncSc, DecSc, ShowTarget, FlashSc, FlashLED, LEDon;  //
Outputs

//    qsec_clk
    wire clk, digsel, qsec;
    qsec_clks slowit (.clkin(clkin), .greset(btnR), .clk(clk), .digsel(digsel),
.qsec(qsec));

    //flipflop for btnU and Go
    FDRE #(.INIT(1'b0)) JustGoAlready (.C(clk), .R(1'b0), .CE(1'b1), .D(btnU), .Q(Go)

    //Pick
    FDRE #(.INIT(1'b0)) JustPick (.C(clk), .R(1'b0), .CE(1'b1), .D(btnD), .Q(Pick));

//    LFSR
    wire [7:0] LFSRout;
    LFSR RandNumGen (.clk(clk), .Q(LFSRout));

//    FDRE3
    wire [2:0] target;
    FDRE #(.INIT(1'b0)) FDRE3[2:0] (.C({3{clk}}), .R(1'b0), .CE(LDTarget),
.D(LFSRout[2:0]), .Q(target));



//    Decoder
    wire [7:0] decOut;
    Decoder decode1 (.x(target), .y(decOut));

//    countUD4L with Din
    wire [3:0] Time;
    wire UTC_out;
    countUD4L countDin (.clk(clk), .btnU(qsec), .LD(RstTime), .Din(4'b0), .Q(Time),
.UTC(UTC_out));
    assign TimeUp = UTC_out;

//    RingCounter
    wire [3:0] Ring_out;
    Ring_Counter ringCount(.Advance(digsel), .clk(clk), .Ring_out(Ring_out));

//    Selector
    wire [3:0] Sel_out;
    wire [3:0] Score;
    Selector selector (.N({Score, 8'h0,1'b0,target}), .sel(Ring_out), .H(Sel_out));

//    hex7seg
    hex7seg Hex7Seggs (.n(Sel_out), .seg(seg));

//    countUD4L for State Machine
    wire Inc, Dec;
    countUD4L countSM (.clk(clk), .LD(1'b0), .btnU(Inc), .btnD(Dec), .Din(4'b0),
.Q(Score));

    // Edge detect
    wire SlowTime;
   Edge_Detector Edge (.btnU(Time[0]), .clk(clk), .Edge_out(SlowTime));

//    MoveLEDs
    wire [7:0] Moveleds_out;
    MoveLEDs Move_leds (.Dir(LFSRout[5]), .Shift(SlowTime), .clk(clk),
.Q(Moveleds_out));

    // Match
    assign Match = ~|(decOut ^ Moveleds_out);



    //    State Machine
    wire [4:0] next, prev;
    State_Machine THE_BRAIN (.clk(clk), .Go(Go), .Pick(Pick), .TimeUp(TimeUp),
.Match(Match),
                            .RstTime(RstTime), .LDTarget(LDTarget), .IncSc(Inc),
.DecSc(Dec),
                            .ShowTarget(ShowTarget), .FlashSc(FlashSc),
.FlashLED(FlashLED), .LEDon(LEDon)
                            ,. nextState(next), .prevState(prev)
                            );
    FDRE #(.INIT(1'b1)) FF_SM0 (.C(clk), .R(1'b0), .CE(1'b1), .D(next[0]),
.Q(prev[0]));
    FDRE #(.INIT(1'b0)) FF_SM1 (.C(clk), .R(1'b0), .CE(1'b1), .D(next[1]),
.Q(prev[1]));
    FDRE #(.INIT(1'b0)) FF_SM2 (.C(clk), .R(1'b0), .CE(1'b1), .D(next[2]),
.Q(prev[2]));
    FDRE #(.INIT(1'b0)) FF_SM3 (.C(clk), .R(1'b0), .CE(1'b1), .D(next[3]),
.Q(prev[3]));
    FDRE #(.INIT(1'b0)) FF_SM4 (.C(clk), .R(1'b0), .CE(1'b1), .D(next[4]),
.Q(prev[4]));


//    Temp for testing
    assign an[3] = ~Ring_out[3] | (FlashSc & ~Time[0]);
    assign an[0] = ~Ring_out[0] | ~ShowTarget;
    assign led[7:0] = ({8{LEDon}} & Moveleds_out)|(decOut  & {8{FlashLED}} &
{8{~Time[0]}});

endmodule
```