```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 02/23/2020 06:32:21 PM
// Design Name:
// Module Name: top_level
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module top_level(
    input clkin, btnC, btnR, btnL,
    input[15:0] sw,
    //input crash,
    //input start,
    output [15:0] led,
    output Hsync,
    output Vsync,

    output [3:0] vgaRed, vgaGreen, vgaBlue,
    output [6:0] seg,
    output dp,
    output [3:0] an
    );


    wire clk, digsel;
    wire frame;
    wire active;
    wire [11:0]posX0, posY0, posY1, posX1, posY2, posX2, posY3, posX3, posY4, posX4,
posY5, posX5, posY6, posX6, carMinX, carMaxX, carMinY, carMaxY;
    wire[11:0]width;

    wire[11:0] hCount, vCount;
```

```
    assign width = 10'd8 + (sw[6:4] * 16);
    lab7_clks not_so_slow (.clkin(clkin), .greset(sw[0]), .clk(clk), .digsel(digsel))
    edge_detector ed(.btn(((hCount == 10'd655) & (vCount == 10'd491)) | ((hCount ==
10'd659) & (vCount == 10'd49))), .clk(clk), .clkout(frame));
    // ((hCount == 10'd655) & (vCount == 10'd491)) | ((hCount == 10'd659) & (vCount
== 10'd49))
    wire[3:0] garbage, offSet;
    LFSR rndm(.clk(clk), .rnd({garbage, offSet}));

    //wire offSet[2:0] = ((rndOut[2:0] == 3'b001) & 10'd0) | ((rndOut[2:0] ==
3'b010) & 10'd8) | ((rndOut[2:0] == 3'b011) & 10'd16) | ((rndOut[2:0] == 3'b100) &
10'd24) | ((rndOut[2:0] == 3'b101) & 10'd32) | ((rndOut[2:0] == 3'b110) & 10'd40) |
((rndOut[2:0] == 3'b111) & 10'd48);



    wire [7:0] timeCount;
    wire [15:0] scoreCount;
    wire started, crash, starting, resetTimer, loadRoad, countScore, resetScore,
blinkScore, blinkCar, resetLED;
    wire frameO = frame & started;
    wire decL = btnL & frameO;
    wire incR = btnR & frameO;
    //(12'd310 & {12{loadRoad}}) | ({12{(posY0>559)} & ((~{12{negative}} * (posX6 +
offSet) + ({12{negative}} * (posX6 - offSet)))
    roadSegs seg0(.posX(posX0), .prevX(posX6), .offSet(offSet), .posY(posY0),
.width(width), .clk(clk), .decL(decL), .incR(incR), .LD(loadRoad), .R(1'b0),
.referencePointX(12'd310), .referencePointY(12'd480), .frameSignal(frameO));
    roadSegs seg1(.posX(posX1), .prevX(posX0),  .offSet(offSet),  .posY(posY1),
.width(width), .clk(clk), .decL(decL), .incR(incR), .LD(loadRoad), .R(1'b0),
.referencePointX(12'd300), .referencePointY(12'd400), .frameSignal(frameO));
    roadSegs seg2(.posX(posX2), .prevX(posX1),  .offSet(offSet),  .posY(posY2),
.width(width), .clk(clk), .decL(decL), .incR(incR), .LD(loadRoad), .R(1'b0),
.referencePointX(12'd260), .referencePointY(12'd320), .frameSignal(frameO));
    roadSegs seg3(.posX(posX3), .prevX(posX2),  .offSet(offSet),  .posY(posY3),
.width(width), .clk(clk), .decL(decL), .incR(incR), .LD(loadRoad), .R(1'b0),
.referencePointX(12'd300), .referencePointY(12'd240), .frameSignal(frameO));
    roadSegs seg4(.posX(posX4), .prevX(posX3),  .offSet(offSet),  .posY(posY4),
.width(width), .clk(clk), .decL(decL), .incR(incR), .LD(loadRoad), .R(1'b0),
.referencePointX(12'd345), .referencePointY(12'd160), .frameSignal(frameO));
    roadSegs seg5(.posX(posX5), .prevX(posX4),  .offSet(offSet),  .posY(posY5),
.width(width), .clk(clk), .decL(decL), .incR(incR), .LD(loadRoad), .R(1'b0),
.referencePointX(12'd320), .referencePointY(12'd80), .frameSignal(frameO));
    roadSegs seg6(.posX(posX6), .prevX(posX5),  .offSet(offSet),  .posY(posY6),
.width(width), .clk(clk), .decL(decL), .incR(incR), .LD(loadRoad), .R(1'b0),
.referencePointX(12'd316), .referencePointY(12'd0), .frameSignal(frameO));
    car c(.carMinX(carMinX), .carMaxX(carMaxX), .carMinY(carMinY), .carMaxY(carMaxY))
```

```
    counterUD16L score(.clk(clk), .R(resetScore), .Up(countScore & timeCount[4]),
.Dw(1'b0), .Q(scoreCount));
    counterUD8L timer(.clk(clk), .R(resetTimer | (started & timeCount[4])),
.CE(1'b1), .Up(frame), .Dw(1'b0), .Q(timeCount), .LD(1'b0));
    state_machine sm(.clk(clk), .start(btnC), .resetScore(resetScore),
.countScore(countScore), .crash(crash), .twoSecs(timeCount[7]), .started(started),
.starting(starting), .blinkCar(blinkCar), .resetTimer(resetTimer),
.loadRoad(loadRoad), .blinkScore(blinkScore), .resetLED(resetLED));
    wire start = timeCount[8];

    wire [3:0] ringOut, selectorOut;
    ring_counter r(.clk(clk), .digsel(digsel), .Q(ringOut));
    Selector s(.sel(ringOut), .N(scoreCount), .H(selectorOut));
    hex7seg hs(.n(selectorOut), .seg(seg));
    //assign an[3:0] = {~ringOut[3], ~ringOut[2], ~ringOut[1], ~ringOut[0]};
    assign an[3] = ~(ringOut[3] & (~blinkScore | (blinkScore & timeCount[5])));
    assign an[2] = ~(ringOut[2] & (~blinkScore | (blinkScore & timeCount[5])));
    assign an[1] = ~(ringOut[1] & (~blinkScore | (blinkScore & timeCount[5])));
    assign an[0] = ~(ringOut[0] & (~blinkScore | (blinkScore & timeCount[5])));


    VGA_Display vga (.clk(clk), .hCount(hCount), .vCount(vCount), .hSync(Hsync),
.vSync(Vsync), .active(active), .seg0({posX0, posY0}), .seg1({posX1, posY1}),
.seg2({posX2, posY2}), .seg3({posX3, posY3}), .seg4({posX4, posY4}), .seg5({posX5,
posY5}), .seg6({posX6, posY6}), .car({carMinX, carMaxX, carMinY, carMaxY}),
.width(width), .vgaRed(vgaRed), .vgaGreen(vgaGreen), .vgaBlue(vgaBlue),
.crash(crash), .blinkCar(blinkCar),  .twoSecs(timeCount[5]));

    wire [7:0] ledCount;
    counterUD8L ledC(.clk(clk), .R(~started | (ledCount[2])), .CE(started),
.Up(timeCount[5]), .Dw(1'b0), .Q(ledCount), .LD(1'b0));
    wire [3:0] ledRCount;
    wire ledThree, ledTwo, ledOne, ledZero;
    //ring_counter LED(.clk(clk), .digsel(started & timeCount[4]), .R(resetLED),
.Q({led[2], led[1], led[0], ledOne}));
    ring_counter LED(.clk(clk), .digsel(started & timeCount[4]), .R(), .Q({ledZero,
ledOne, ledTwo, ledThree}));
    //assign led[5] = blinkScore;


//    assign led[2] = (~ledCount[1] & ledCount[0]);
//    assign led[1] = (ledCount[1] & ~ledCount[1]);
//    assign led[0] = (ledCount[1] & ledCount[0]);
//    assign led[6] = (ledCount[2:0] == 3'b000) & started;
//    assign led[5] = (ledCount[2:0] == 3'b001) & started;
```

```
//    assign led[4] = (ledCount[2:0] == 3'b010) & started;
//    assign led[10] = (ledCount[2:0] == 3'b000) & started;
//    assign led[9] = (ledCount[2:0] == 3'b001) & started;
//    assign led[8] = (ledCount[2:0] == 3'b010) & started;
//    assign led[14] = (ledCount[2:0] == 3'b000) & started;
//    assign led[13] = (ledCount[2:0] == 3'b001) & started;
//    assign led[12] = (ledCount[2:0] == 3'b010) & started;

    assign led[15] = (starting & (timeCount > 10'd32)) | (ledThree & started);
    assign led[14] = ledTwo & started;
    assign led[13] = ledOne & started;
    assign led[12] = ledZero & started;
    assign led[11] = (starting & (timeCount > 10'd56)) | (ledThree & started);
    assign led[10] = ledTwo & started;
    assign led[9] = ledOne & started;
    assign led[8] = ledZero & started;
    assign led[7] = (starting & (timeCount > 10'd80)) | (ledThree & started);
    assign led[6] = ledTwo & started;
    assign led[5] = ledOne & started;
    assign led[4] = ledZero & started;
    assign led[3] = (starting & (timeCount > 10'd104)) | (ledThree & started);
    assign led[2] = ledTwo & started;
    assign led[1] = ledOne & started;
    assign led[0] = ledZero & started;


endmodule
```