```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 02/23/2020 09:32:18 PM
// Design Name:
// Module Name: VGA_Display
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module VGA_Display(
    input clk, blinkCar, twoSecs,
    input [23:0]seg0, seg1, seg2, seg3, seg4, seg5, seg6,
    input[47:0] car,
    input [11:0]width,
    //input[3:0] red, green, blue,
    output[11:0] hCount, vCount,
    output[3:0] vgaRed, vgaGreen, vgaBlue,
    output hSync, vSync, active, crash

    );

    wire [11:0]vOut, hOut;
    wire outOfActive;
    wire resetH, resetV;
    wire [11:0]vga;
   // hOut == 799
    assign resetH = (hOut == 10'd799);
    // vOut == 524
    assign resetV = (vOut == 10'd524);

    //reset when hOut = 799
    counterUD12L h(.clk(clk), .Up(1'b1), .R(resetH), .Q(hOut), .LD(1'b0), .Dw(1'b0));
    //reset when vOut = 524
```

```
    counterUD12L v(.clk(clk), .Up(resetH), .R(resetV), .Q(vOut), .LD(1'b0), .Dw(1'b0)

    //H: [655, 750] or H: [656, 751]
    assign hSync = (hOut < 10'd655) | (hOut > 10'd750);
    //V: [489, 490]
    assign vSync = (vOut < 10'd489) | (vOut > 10'd490);

    // v>479, h>639
    assign outOfActive = ((vOut > 10'd479) | (hOut > 10'd639));
    wire seg0Xrange, seg0Yrange, seg1Xrange, seg1Yrange, seg2Xrange, seg2Yrange,
seg3Xrange, seg3Yrange, seg4Xrange, seg4Yrange, seg5Xrange, seg5Yrange, seg6Xrange,
seg6Yrange;
    vga_seg_control zero(.seg(seg0), .width(width), .hCount(hCount),
.vCount(vCount), .segXrange(seg0Xrange), .segYrange(seg0Yrange));
    vga_seg_control one(.seg(seg1), .width(width), .hCount(hCount), .vCount(vCount),
.segXrange(seg1Xrange), .segYrange(seg1Yrange));
    vga_seg_control two(.seg(seg2), .width(width), .hCount(hCount), .vCount(vCount),
.segXrange(seg2Xrange), .segYrange(seg2Yrange));
    vga_seg_control three(.seg(seg3), .width(width), .hCount(hCount),
.vCount(vCount), .segXrange(seg3Xrange), .segYrange(seg3Yrange));
    vga_seg_control four(.seg(seg4), .width(width), .hCount(hCount),
.vCount(vCount), .segXrange(seg4Xrange), .segYrange(seg4Yrange));
    vga_seg_control five(.seg(seg5), .width(width), .hCount(hCount),
.vCount(vCount), .segXrange(seg5Xrange), .segYrange(seg5Yrange));
    vga_seg_control six(.seg(seg6), .width(width), .hCount(hCount), .vCount(vCount),
.segXrange(seg6Xrange), .segYrange(seg6Yrange));

    wire check0, check1, check2, check3, check4, check5, check6;
    crash_check cc0(.seg(seg0), .width(width), .hCount(hCount), .vCount(vCount),
.check(check0));
    crash_check cc1(.seg(seg1), .width(width), .hCount(hCount), .vCount(vCount),
.check(check1));
    crash_check cc2(.seg(seg2), .width(width), .hCount(hCount), .vCount(vCount),
.check(check2));
    crash_check cc3(.seg(seg3), .width(width), .hCount(hCount), .vCount(vCount),
.check(check3));
    crash_check cc4(.seg(seg4), .width(width), .hCount(hCount), .vCount(vCount),
.check(check4));
    crash_check cc5(.seg(seg5), .width(width), .hCount(hCount), .vCount(vCount),
.check(check5));
    crash_check cc6(.seg(seg6), .width(width), .hCount(hCount), .vCount(vCount),
.check(check6));
    assign crash = ~(check0 | check1 | check2 | check3 | check4 | check5 | check6);


    wire[11:0] carMinX, carMaxX, carMinY, carMaxY;
```

```
    wire carXrange, carYrange;
    assign carMinX = car[45:34];
    assign carMaxX = car[33:22];
    assign carMinY = car[21:12];
    assign carMaxY = car[11:0];
    //assign carXrange = (hCount >= carMinX) & (hCount <= carMaxX);
    //assign carYrange = (vCount >= carMinY) & (vCount <= carMaxY);
    assign carXrange = (hCount >= 10'd312) & (hCount <= 10'd328);
    assign carYrange = (vCount >= 10'd384) & (vCount <= 10'd400);
    // (12'hfff & {12{carXrange & carYrange}})
       //// assign vgaRed = (4'b1111 & {4{(seg1Xrange & seg1Yrange)}}) | (4'b1111 &
{4{(seg0Xrange & seg0Yrange)}});
       ////assign vgaGreen = (4'b0111 & {4{(seg1Xrange & seg1Yrange)}});
       assign vga = (12'hfff & (~{12{blinkCar}} | ({12{blinkCar}} & {12{twoSecs}}))
& {12{carXrange & carYrange}}) | (12'hf00 & {12{seg0Xrange & seg0Yrange}}) |
(12'hf70 & {12{(seg1Xrange & seg1Yrange)}}) | (12'hff0 & {12{(seg2Xrange &
seg2Yrange)}}) | (12'h0f0 & {12{(seg3Xrange & seg3Yrange)}}) | (12'h00f &
{12{(seg4Xrange & seg4Yrange)}}) | (12'h225 & {12{(seg5Xrange & seg5Yrange)}}) |
(12'h80f & {12{(seg6Xrange & seg6Yrange)}});
       assign vgaRed = vga[11:8];
       assign vgaGreen = vga[7:4];
       assign vgaBlue = vga[3:0];
       // testX: (hCount
       //
       // & (~blinkCar | blinkCar &
     //assign testRangeX = (hCount >= 10'd311) & (hCount <= 10'd329);
     // (testRangeX  & testRangeY
     //assign testRangeY = (vCount >= 10'd383) & (vCount <= 10'd401);


//    assign vgaRed = (4'b1111 & {4{(seg0Xrange & seg0Yrange)}}) | (4'b1111 &
{4{(seg3Xrange & seg3Yrange)}}) | (4'b1111 & {4{(seg6Xrange & seg6Yrange)}});
//    assign vgaGreen = (4'b1111 & {4{(seg1Xrange & seg1Yrange)}}) | (4'b1111 &
{4{(seg4Xrange & seg4Yrange)}});
//    assign vgaBlue = (4'b1111 & {4{(seg2Xrange & seg2Yrange)}}) | (4'b1111 &
{4{(seg5Xrange & seg5Yrange)}}) ;
    assign active = ~outOfActive;
    assign hCount = hOut;
    assign vCount = vOut;
endmodule
```