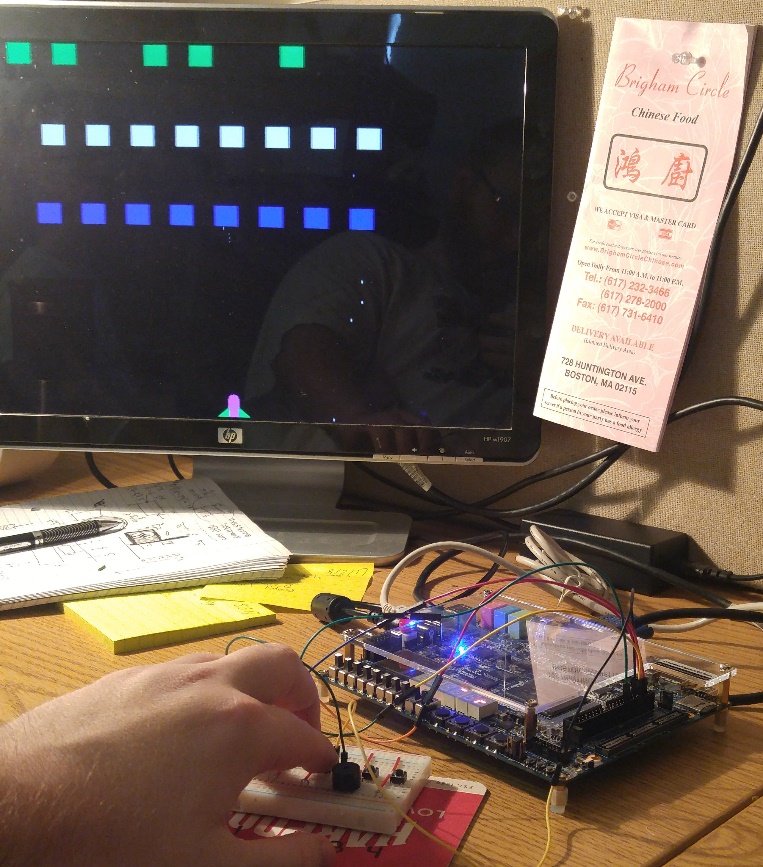
Gagala

ELEC 3200 – Final Report



By:

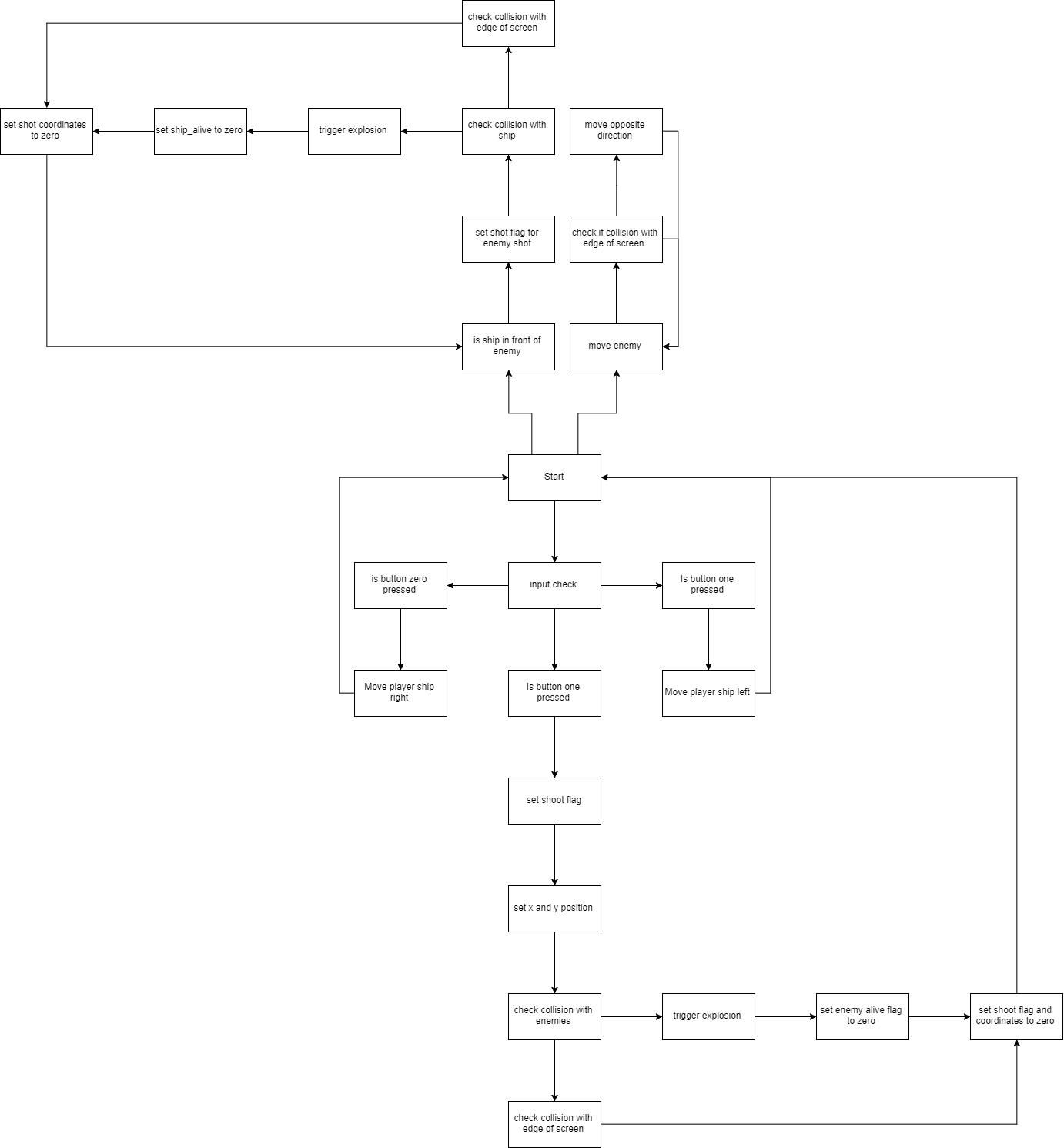
Sean Copp

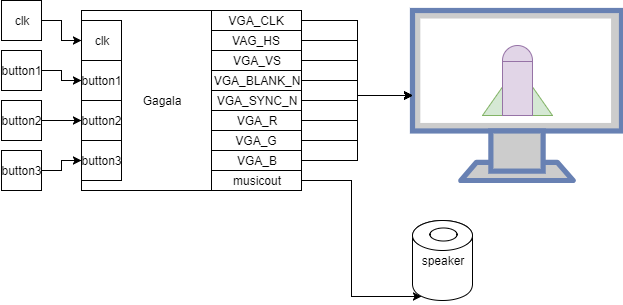
Michael Hickey

1. Brief Description of the Assignment

To utilize all the knowledge, we have obtained from the classes and labs throughout the semester and apply them to topic of our choosing

.

1. Flowchart
   1. My overall block diagram
   2. My technical block diagram



1. Step in solving the problem

The first problem that had to be tackled was drawing the player model. It was decided that the main rectangle that made the body was going to be made first and that everything was going to be based off those coordinates. After drawing the main rectangle, the circle at the top was the next problem that had to be tackled. To do that, we used the equation for a circle . Where, x is the VGA\_X used for the horizontal position of the cathode ray, and y is the VGA\_Y used for the vertical position of the cathode ray. The variables h and k are both offsets for the x and y variables respectively. r represents the desired length of the radius of the circle. Finally, the wings of the ship were constructed out of triangles. These used the same principle as the rectangles but they also required a slope value that could be moved without climbing up the screen as the player moved. . This allowed all the values under the curve to be colored appropriately. After drawing the player model, an basic enemy was drawn using a simple rectangle, to make practicing moving using the on board buttons easier. This model was kept because if a more complicated model was used there was a risk of running out of logic units. After getting movement to work on a simple rectangle that same concept was applied to the four shapes that made up the player model.

After figuring out how the player and the enemies were going to look, actual game mechanics had to figured out. To do this, a flag was made to note when the player pressed the shoot button When this flag was set the players shot was draw, as a small rectangle, and as long as the rectangle did not hit an enemy or the upper limit of the display it would stay on the screen and move one pixel up the screen on the positive edge of the clock. If the shot did hit an enemy then a hit flag was set and an explosion is set off to give the player visual feedback that an enemy was hit. The explosion was created using the same circle equation from above but the radius was increased to allow the explosion to expand. Also, as the explosion is triggered the enemy simultaneously disappear based on a unique alive flag for each enemy.

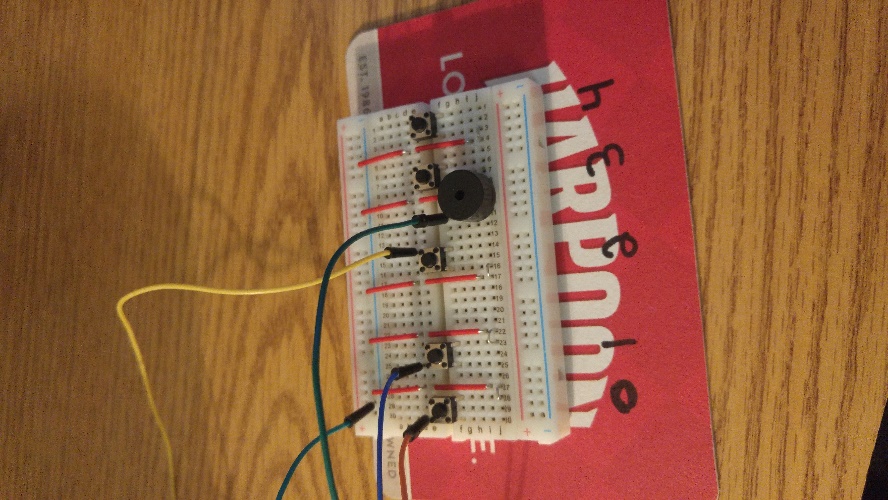
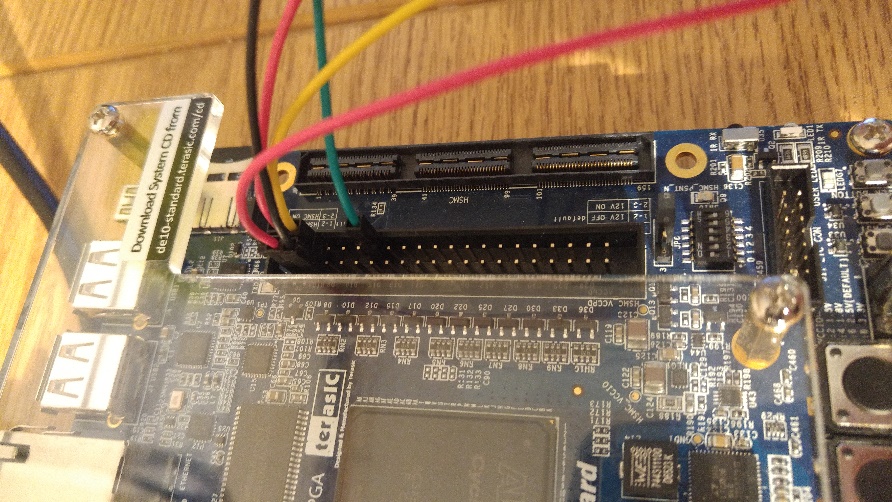
After giving the player the ability to shoot, the player needed some things to shoot at. Since it’s no fun to shoot at a still target, the enemy needed to move. Enemy movement was based on the same timer as everything else which added or subtracted one based on the last edge that was hit. After enemy movement was finished, 23 more enemies were added to up the challenge of the game a little bit.

After added more enemies, the win and lose screen were created. The letter for these screens were made using rectangles like the rectangle that makes up the ship. The screen was broken up into two rows of ten. This means that each character received a plot that is sixty-four by two-hundred-forty pixels. Then rectangles were arranged in a way that looked like the characters that were needed to display, “YOU WIN” and “YOU DIED”.

After creating the lose screen, the player needed a way to lose. Due to time constraints, the original attack pattern used in Galaga could not be used. Instead the enemies in Gagala were made to shoot at the player character when the enemy is directly over the player. Each enemy has a unique shoot flag, that allows for up to 24 enemy shots to be on screen at once.

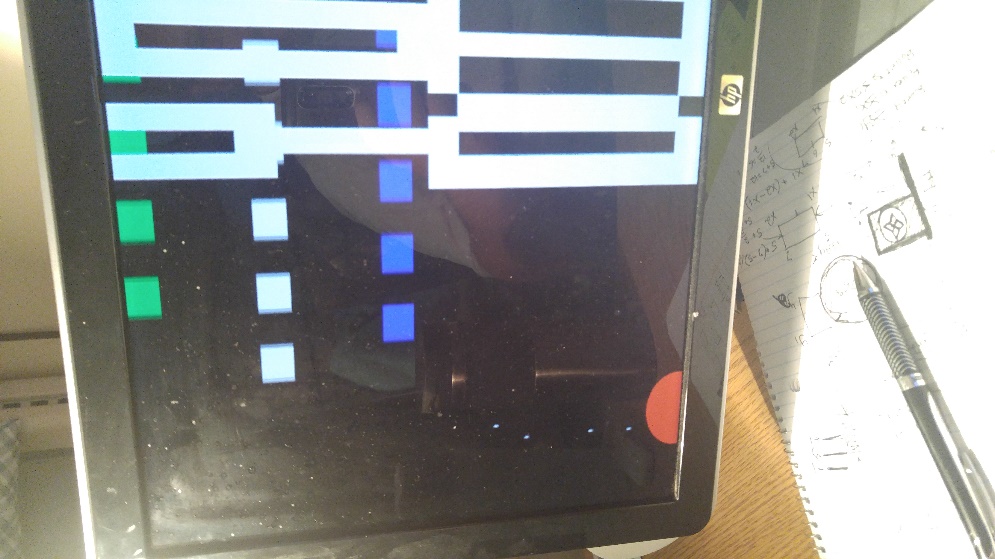
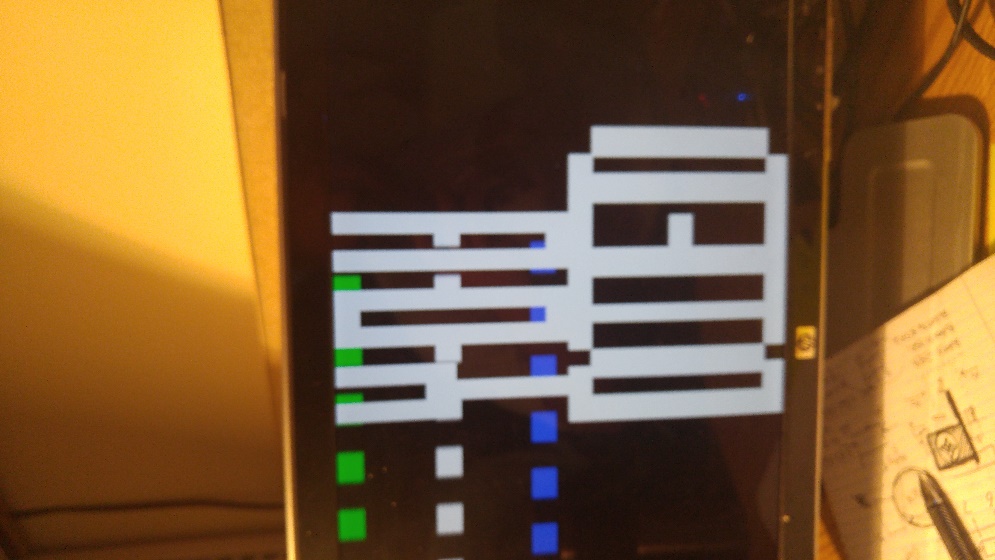
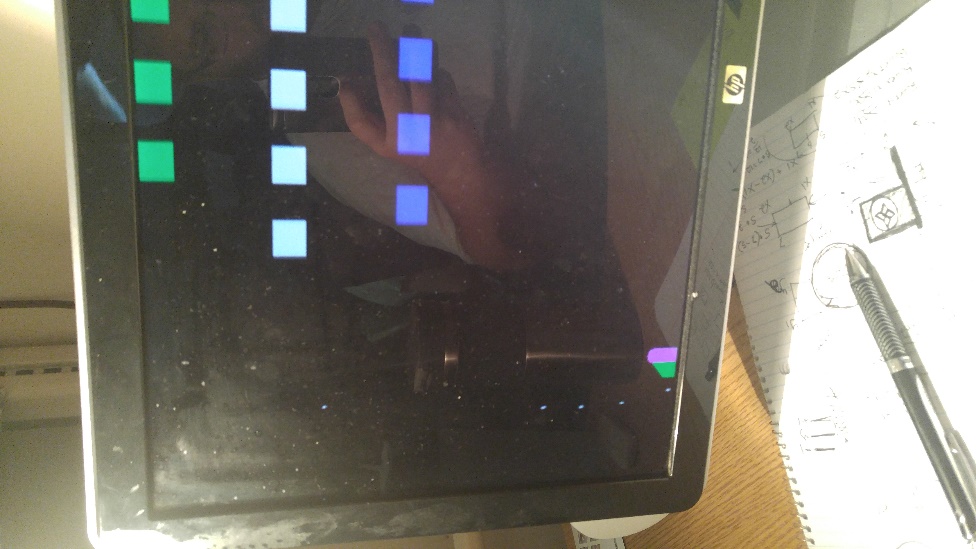
Now that there was a way to win and a way to lose, the last thing left was set the mood for the player by providing a moving background and background music. The moving background used a moving point on screen and on that point, it would draw a white rectangle to give the appearance of a star, but with the way the point on the screen it gave the appearance of rapid movement through the void of space. The background music was created using the previous speaker lab and translating some sheet music to the scale that was created during that lab.

1. Pictures of your Final Project



This is an image of the pinout of the GPIO that is attached to the controller.

This is a picture of the external controller prototype that can be used with the project.



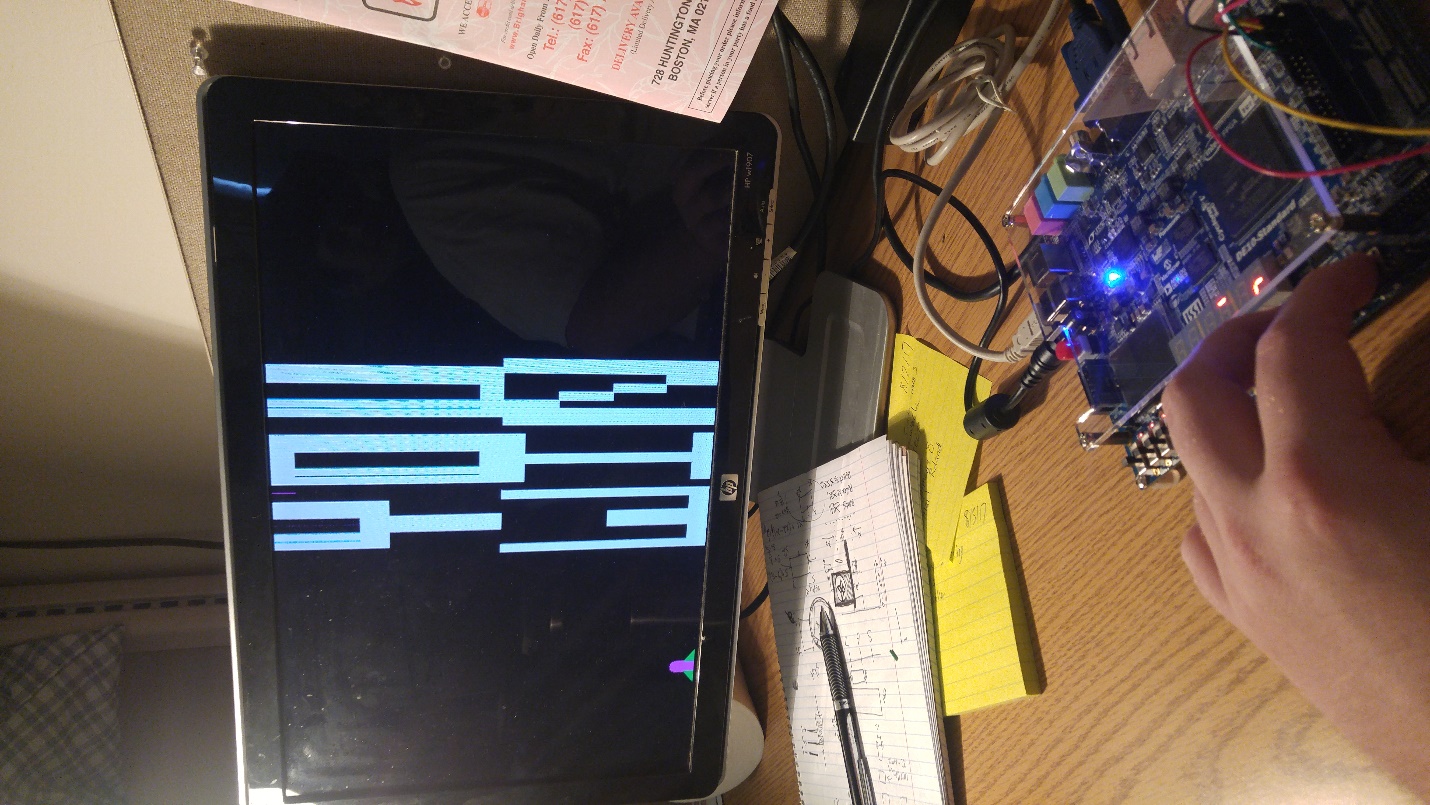
Pictured Right:

Player module exploding and death screen.

Pictured Right:

Better view of death screen.

To the left and right are images of graphical glitches that occur when uploading .sof file to board after the board has been powered off.



Pictured:

The Player model next to the victory screen.

1. New Acquired Skills/Knowledge

We learned about mission creep. Now we know not to add too many extra components to a project. We learned how to apply mathematical equations to Verilog to draw shapes on a display. We learned that sine and cosine are very difficult in Verilog. We learned to keep in mind the limitations of the device we are trying to implement our project on. We learned it’s hard to keep a train of thought with an excessively long compile time.

1. References

Copp, Sean. Lab 4.

This was used as a base for creating the music for this lab.

Copp, Sean. Lab 7.

This was used as a base for creating all of the graphics in the project.

DESMOS Inc., “DESMOS,” *Explore Math With Desmos*. DESMOS Inc. [www.desmos.com/.Acknowledgement](http://www.desmos.com/.Acknowledgement)

This was used to draft shapes, such as circles and slopes pior to implementing them in Verilog.

Terasic, “DE10-SoC User Manual.” Terasic, Taiwan, 2017.

This was used for assigning all inputs and outputs on the board.

1. Acknowledgements

We would like to thank our moms for giving birth to us.

We would like to than Dr. Marpaung for teaching us how to use the FPGA boards properly.

We would like to thank our fellow students for being supportive of our project.

We would like to thank ourselves for having the mental fortitude to actually completing the assignment.

1. Code with Heavy Comments on a separate file

module gagala (

//input

clk, key, debugLED, sw, gamepad, musicOut,

//output

VGA\_CLK, VGA\_HS, VGA\_VS, VGA\_BLANK\_N, VGA\_SYNC\_N, VGA\_R, VGA\_G, VGA\_B,

);

input clk;

input [3:0] key;

input [9:0] sw;

input [2:0] gamepad;

///////// VGA /////////

output VGA\_CLK, VGA\_HS, VGA\_VS, VGA\_BLANK\_N, VGA\_SYNC\_N;

output [ 7: 0] VGA\_R, VGA\_G, VGA\_B;

output reg [9:0] debugLED;

// For VGA Controller

reg [9:0] mRed;

reg [9:0] mGreen;

reg [9:0] mBlue;

wire [10:0] VGA\_X;

wire [10:0] VGA\_Y;

wire VGA\_Read; // VGA data request

wire DLY2;

// VGA Controller

wire [9:0] vga\_r10;

wire [9:0] vga\_g10;

wire [9:0] vga\_b10;

assign VGA\_R = vga\_r10[9:2];

assign VGA\_G = vga\_g10[9:2];

assign VGA\_B = vga\_b10[9:2];

///MUSIC\\\

output musicOut;

///Ship Variable Initialization///

//alive//

reg ship\_alive;

//For convenience//

integer xmid;

integer ymid;

//position//

integer ship\_x;

integer ship\_y;

//tanks//

integer ship\_fin\_left; //rear ship component left

integer ship\_fin\_right; //rear ship component right

//Hull//

integer ship\_hull\_x1; //ship main component top left

integer ship\_hull\_x2; //ship main component top right

integer ship\_hull\_y1; //ship main component bottom left

integer ship\_hull\_y2; //ship main component bottom right

//Cockpit//

integer ship\_cockpit; //top component of ship

//ship\_color//

reg [9:0] ship\_red; //ship red value

reg [9:0] ship\_green; //ship green value

reg [9:0] ship\_blue; //ship blue value

///Enemy Variable Initialization///

//Enemy 1//

//alive//

reg alive;

//hull//

integer en\_hull\_x1; //enemy top left

integer en\_hull\_x2; //enemy top right

integer en\_hull\_y1; //enemy bottom left

integer en\_hull\_y2; //enemy bottom right

//Enemy Color//

reg [9:0] en\_red; //enemy red value

reg [9:0] en\_green; //enemy green value

reg [9:0] en\_blue; //enemy blue value

//Enemy Shot//

reg en\_shot;

integer en\_shot\_x1, en\_shot\_x2;

integer en\_shot\_y1, en\_shot\_y2;

//enemy movement variables

integer eox, eoy; //enemy offset x and enemy offset y

integer xFlag, yFlag;

integer bt1x, bt2x;// the b in y=mx + b for triangle 1 left triangle

integer bt1y, bt2y;

//Enemy 2//

//alive//

reg alive2;

//hull//

integer en\_hull2\_x1;

integer en\_hull2\_x2;

integer en\_hull2\_y1;

integer en\_hull2\_y2;

//color//

reg [9:0] en2\_red;

reg [9:0] en2\_green;

reg [9:0] en2\_blue;

//Enemy Shot//

reg en\_shot2;

integer en\_shot2\_x1, en\_shot2\_x2;

integer en\_shot2\_y1, en\_shot2\_y2;

//movement// This might not be necessary

//Enemy 3//

//alive//

reg alive3;

//hull//

integer en\_hull3\_x1;

integer en\_hull3\_x2;

integer en\_hull3\_y1;

integer en\_hull3\_y2;

//color//

reg [9:0] en3\_red;

reg [9:0] en3\_green;

reg [9:0] en3\_blue;

//Enemy Shot//

reg en\_shot3;

integer en\_shot3\_x1, en\_shot3\_x2;

integer en\_shot3\_y1, en\_shot3\_y2;

//Enemy 4//

//alive//

reg alive4;

//hull//

integer en\_hull4\_x1; //enemy top left

integer en\_hull4\_x2; //enemy top right

integer en\_hull4\_y1; //enemy bottom left

integer en\_hull4\_y2; //enemy bottom right

//Enemy Color//

reg [9:0] en4\_red; //enemy red value

reg [9:0] en4\_green; //enemy green value

reg [9:0] en4\_blue; //enemy blue value

//Enemy Shot//

reg en\_shot4;

integer en\_shot4\_x1, en\_shot4\_x2;

integer en\_shot4\_y1, en\_shot4\_y2;

//Enemy 5//

//alive//

reg alive5;

//hull//

integer en\_hull5\_x1; //enemy top left

integer en\_hull5\_x2; //enemy top right

integer en\_hull5\_y1; //enemy bottom left

integer en\_hull5\_y2; //enemy bottom right

//Enemy Color//

reg [9:0] en5\_red; //enemy red value

reg [9:0] en5\_green; //enemy green value

reg [9:0] en5\_blue; //enemy blue value

//Enemy Shot//

reg en\_shot5;

integer en\_shot5\_x1, en\_shot5\_x2;

integer en\_shot5\_y1, en\_shot5\_y2;

//Enemy 6//

//alive//

reg alive6;

//hull//

integer en\_hull6\_x1; //enemy top left

integer en\_hull6\_x2; //enemy top right

integer en\_hull6\_y1; //enemy bottom left

integer en\_hull6\_y2; //enemy bottom right

//Enemy Color//

reg [9:0] en6\_red; //enemy red value

reg [9:0] en6\_green; //enemy green value

reg [9:0] en6\_blue; //enemy blue value

//Enemy Shot//

reg en\_shot6;

integer en\_shot6\_x1, en\_shot6\_x2;

integer en\_shot6\_y1, en\_shot6\_y2;

//Enemy 7//

//alive//

reg alive7;

//hull//

integer en\_hull7\_x1; //enemy top left

integer en\_hull7\_x2; //enemy top right

integer en\_hull7\_y1; //enemy bottom left

integer en\_hull7\_y2; //enemy bottom right

//Enemy Color//

reg [9:0] en7\_red; //enemy red value

reg [9:0] en7\_green; //enemy green value

reg [9:0] en7\_blue; //enemy blue value

//Enemy Shot//

reg en\_shot7;

integer en\_shot7\_x1, en\_shot7\_x2;

integer en\_shot7\_y1, en\_shot7\_y2;

//Enemy 8//

//alive//

reg alive8;

//hull//

integer en\_hull8\_x1; //enemy top left

integer en\_hull8\_x2; //enemy top right

integer en\_hull8\_y1; //enemy bottom left

integer en\_hull8\_y2; //enemy bottom right

//Enemy Color//

reg [9:0] en8\_red; //enemy red value

reg [9:0] en8\_green; //enemy green value

reg [9:0] en8\_blue; //enemy blue value

//Enemy Shot//

reg en\_shot8;

integer en\_shot8\_x1, en\_shot8\_x2;

integer en\_shot8\_y1, en\_shot8\_y2;

//Enemy 9//

//alive//

reg alive9;

//hull//

integer en\_hull9\_x1;

integer en\_hull9\_x2;

integer en\_hull9\_y1;

integer en\_hull9\_y2;

//color//

reg [9:0] en9\_red;

reg [9:0] en9\_green;

reg [9:0] en9\_blue;

//Enemy Shot//

reg en\_shot9;

integer en\_shot9\_x1, en\_shot9\_x2;

integer en\_shot9\_y1, en\_shot9\_y2;

//Enemy 10//

//alive//

reg alive10;

//hull//

integer en\_hull10\_x1;

integer en\_hull10\_x2;

integer en\_hull10\_y1;

integer en\_hull10\_y2;

//color//

reg [9:0] en10\_red;

reg [9:0] en10\_green;

reg [9:0] en10\_blue;

//Enemy Shot//

reg en\_shot10;

integer en\_shot10\_x1, en\_shot10\_x2;

integer en\_shot10\_y1, en\_shot10\_y2;

//Enemy 11//

//alive//

reg alive11;

//hull//

integer en\_hull11\_x1;

integer en\_hull11\_x2;

integer en\_hull11\_y1;

integer en\_hull11\_y2;

//color//

reg [9:0] en11\_red;

reg [9:0] en11\_green;

reg [9:0] en11\_blue;

//Enemy Shot//

reg en\_shot11;

integer en\_shot11\_x1, en\_shot11\_x2;

integer en\_shot11\_y1, en\_shot11\_y2;

//Enemy 12//

//alive//

reg alive12;

//hull//

integer en\_hull12\_x1;

integer en\_hull12\_x2;

integer en\_hull12\_y1;

integer en\_hull12\_y2;

//color//

reg [9:0] en12\_red;

reg [9:0] en12\_green;

reg [9:0] en12\_blue;

//Enemy Shot//

reg en\_shot12;

integer en\_shot12\_x1, en\_shot12\_x2;

integer en\_shot12\_y1, en\_shot12\_y2;

//Enemy 13//

//alive//

reg alive13;

//hull//

integer en\_hull13\_x1;

integer en\_hull13\_x2;

integer en\_hull13\_y1;

integer en\_hull13\_y2;

//color//

reg [9:0] en13\_red;

reg [9:0] en13\_green;

reg [9:0] en13\_blue;

//Enemy Shot//

reg en\_shot13;

integer en\_shot13\_x1, en\_shot13\_x2;

integer en\_shot13\_y1, en\_shot13\_y2;

//Enemy 14//

//alive//

reg alive14;

//hull//

integer en\_hull14\_x1;

integer en\_hull14\_x2;

integer en\_hull14\_y1;

integer en\_hull14\_y2;

//color//

reg [9:0] en14\_red;

reg [9:0] en14\_green;

reg [9:0] en14\_blue;

//Enemy Shot//

reg en\_shot14;

integer en\_shot14\_x1, en\_shot14\_x2;

integer en\_shot14\_y1, en\_shot14\_y2;

//Enemy 15//

//alive//

reg alive15;

//hull//

integer en\_hull15\_x1;

integer en\_hull15\_x2;

integer en\_hull15\_y1;

integer en\_hull15\_y2;

//color//

reg [9:0] en15\_red;

reg [9:0] en15\_green;

reg [9:0] en15\_blue;

//Enemy Shot//

reg en\_shot15;

integer en\_shot15\_x1, en\_shot15\_x2;

integer en\_shot15\_y1, en\_shot15\_y2;

//Enemy 16//

//alive//

reg alive16;

//hull//

integer en\_hull16\_x1;

integer en\_hull16\_x2;

integer en\_hull16\_y1;

integer en\_hull16\_y2;

//color//

reg [9:0] en16\_red;

reg [9:0] en16\_green;

reg [9:0] en16\_blue;

//Enemy Shot//

reg en\_shot16;

integer en\_shot16\_x1, en\_shot16\_x2;

integer en\_shot16\_y1, en\_shot16\_y2;

//Enemy 17//

//alive//

reg alive17;

//hull//

integer en\_hull17\_x1;

integer en\_hull17\_x2;

integer en\_hull17\_y1;

integer en\_hull17\_y2;

//color//

reg [9:0] en17\_red;

reg [9:0] en17\_green;

reg [9:0] en17\_blue;

//Enemy Shot//

reg en\_shot17;

integer en\_shot17\_x1, en\_shot17\_x2;

integer en\_shot17\_y1, en\_shot17\_y2;

//Enemy 18//

//alive//

reg alive18;

//hull//

integer en\_hull18\_x1;

integer en\_hull18\_x2;

integer en\_hull18\_y1;

integer en\_hull18\_y2;

//color//

reg [9:0] en18\_red;

reg [9:0] en18\_green;

reg [9:0] en18\_blue;

//Enemy Shot//

reg en\_shot18;

integer en\_shot18\_x1, en\_shot18\_x2;

integer en\_shot18\_y1, en\_shot18\_y2;

//Enemy 19//

//alive//

reg alive19;

//hull//

integer en\_hull19\_x1;

integer en\_hull19\_x2;

integer en\_hull19\_y1;

integer en\_hull19\_y2;

//color//

reg [9:0] en19\_red;

reg [9:0] en19\_green;

reg [9:0] en19\_blue;

//Enemy Shot//

reg en\_shot19;

integer en\_shot19\_x1, en\_shot19\_x2;

integer en\_shot19\_y1, en\_shot19\_y2;

//Enemy 20//

//alive//

reg alive20;

//hull//

integer en\_hull20\_x1;

integer en\_hull20\_x2;

integer en\_hull20\_y1;

integer en\_hull20\_y2;

//color//

reg [9:0] en20\_red;

reg [9:0] en20\_green;

reg [9:0] en20\_blue;

//Enemy Shot//

reg en\_shot20;

integer en\_shot20\_x1, en\_shot20\_x2;

integer en\_shot20\_y1, en\_shot20\_y2;

//Enemy 21//

//alive//

reg alive21;

//hull//

integer en\_hull21\_x1;

integer en\_hull21\_x2;

integer en\_hull21\_y1;

integer en\_hull21\_y2;

//color//

reg [9:0] en21\_red;

reg [9:0] en21\_green;

reg [9:0] en21\_blue;

//Enemy Shot//

reg en\_shot21;

integer en\_shot21\_x1, en\_shot21\_x2;

integer en\_shot21\_y1, en\_shot21\_y2;

//Enemy 22//

//alive//

reg alive22;

//hull//

integer en\_hull22\_x1;

integer en\_hull22\_x2;

integer en\_hull22\_y1;

integer en\_hull22\_y2;

//color//

reg [9:0] en22\_red;

reg [9:0] en22\_green;

reg [9:0] en22\_blue;

//Enemy Shot//

reg en\_shot22;

integer en\_shot22\_x1, en\_shot22\_x2;

integer en\_shot22\_y1, en\_shot22\_y2;

//Enemy 23//

//alive//

reg alive23;

//hull//

integer en\_hull23\_x1;

integer en\_hull23\_x2;

integer en\_hull23\_y1;

integer en\_hull23\_y2;

//color//

reg [9:0] en23\_red;

reg [9:0] en23\_green;

reg [9:0] en23\_blue;

//Enemy Shot//

reg en\_shot23;

integer en\_shot23\_x1, en\_shot23\_x2;

integer en\_shot23\_y1, en\_shot23\_y2;

//Enemy 24//

//alive//

reg alive24;

//hull//

integer en\_hull24\_x1;

integer en\_hull24\_x2;

integer en\_hull24\_y1;

integer en\_hull24\_y2;

//color//

reg [9:0] en24\_red;

reg [9:0] en24\_green;

reg [9:0] en24\_blue;

//Enemy Shot//

reg en\_shot24;

integer en\_shot24\_x1, en\_shot24\_x2;

integer en\_shot24\_y1, en\_shot24\_y2;

//====================\\

//used for the movement

integer en\_high\_x1;

integer en\_high\_x2;

integer en\_mid\_x1;

integer en\_mid\_x2;

integer en\_low\_x1;

integer en\_low\_x2;

integer quarter, offset, moveCounter;//used for movement

integer direction, direction\_low, direction\_high;//either forwards or backwards 1 / 0

integer increment;

//Shot Stuff//

reg shot\_act;

integer shotx1;

integer shotx2;

integer shoty1;

integer shoty2;

integer shottest; initial shottest = 0;

//enemy shot stuff//

reg high\_shot, mid\_shot, low\_shot;

integer high\_shot\_x1, mid\_shot\_x1, low\_shot\_x1;

integer high\_shot\_x2, mid\_shot\_x2, low\_shot\_x2;

integer high\_shot\_y1, mid\_shot\_y1, low\_shot\_y1;

integer high\_shot\_y2, mid\_shot\_y2, low\_shot\_y2;

//Explosion variables//

reg exp\_live;

integer exp\_x1;

integer exp\_y1;

integer exp\_radius;

reg [9:0] exp\_red;

reg [9:0] exp\_green;

reg [9:0] exp\_blue;

initial begin

dir = 1;

increment = 1;

bt1x = 526;

bt2x = 390;

alive = 1'b1;

debugLED = 10'h000;

//Enemy Shot intialization

high\_shot = 1'b0;

mid\_shot = 1'b0;

low\_shot = 1'b0;

///Ship Vars///

//alive//

ship\_alive = 1'b1;

//convenience//

xmid = (ship\_hull\_x2- ship\_hull\_x1) - ((ship\_hull\_x2-ship\_hull\_x1)/2);

ymid = (ship\_hull\_y2 -ship\_hull\_y1) - ((ship\_hull\_y2-ship\_hull\_y1)/2);

//Ship Position//

ship\_x = 0;

ship\_y = 0;

//Hull//

ship\_hull\_x1 = 62; //62

ship\_hull\_x2 = 74; //74

ship\_hull\_y1 = 452; //452

ship\_hull\_y2 = 476; //476

//ship\_color//

ship\_red = 10'hFFF; //Should Truncate

ship\_green = 10'h000; //Should TRuncate

ship\_blue = 10'hFFF; //Should truncate

///Explosions///

//consider having multiple concentric circles after finishing this one

exp\_live = 1'b0; //This there an explosion or not

exp\_x1 = 0; //This needs to be the x value in the center of an enemy

exp\_y1 = 0; //This needs to be the y value in the center of an enemy

//Next line describes a circle that should appear in the center of an enemy as its alive state

//goes from 1 to 0, then the circle grows to a point then disappears

exp\_radius = 36;

//This should be an orange red value

exp\_red = 10'hFFF;

exp\_green = 10'h880;

exp\_blue = 10'h000;

///Enemy///

//alive

alive = 1'b1;

//hull//

en\_hull\_x1 = 1;

en\_hull\_x2 = 25;

en\_hull\_y1 = 100;

en\_hull\_y2 = 125;

//Enemy Color//

en\_red = 10'hFFF;

en\_green = 10'hFFF;

en\_blue = 10'hFFF;

//Enemy Shot//

en\_shot = 1'b0;

//Enemy 2//

//alive//

alive2 = 1'b1;

//hull//

en\_hull2\_x1 = en\_hull\_x1 + 50;

en\_hull2\_x2 = en\_hull\_x2 + 50;

en\_hull2\_y1 = en\_hull\_y1;

en\_hull2\_y2 = en\_hull\_y2;

//color//

en2\_red = 10'hFFF;

en2\_green = 10'hFFF;

en2\_blue = 10'hFFF;

//Enemy Shot//

en\_shot2 = 1'b0;

///Enemy 3///

//alive

alive3 = 1'b1;

//hull//

en\_hull3\_x1 = en\_hull2\_x1 + 50;

en\_hull3\_x2 = en\_hull2\_x2 + 50;

en\_hull3\_y1 = en\_hull\_y1;

en\_hull3\_y2 = en\_hull\_y2;

//Enemy Color//

en3\_red = 10'hFFF;

en3\_green = 10'hFFF;

en3\_blue = 10'hFFF;

//Enemy Shot//

en\_shot3 = 1'b0;

///Enemy 4///

//alive

alive4 = 1'b1;

//hull//

en\_hull4\_x1 = en\_hull3\_x1 + 50;

en\_hull4\_x2 = en\_hull3\_x2 + 50;

en\_hull4\_y1 = en\_hull\_y1;

en\_hull4\_y2 = en\_hull\_y2;

//Enemy Color//

en4\_red = 10'hFFF;

en4\_green = 10'hFFF;

en4\_blue = 10'hFFF;

//Enemy Shot//

en\_shot4 = 1'b0;

///Enemy 5///

//alive

alive5 = 1'b1;

//hull//

en\_hull5\_x1 = en\_hull4\_x1 + 50;

en\_hull5\_x2 = en\_hull4\_x2 + 50;

en\_hull5\_y1 = en\_hull\_y1;

en\_hull5\_y2 = en\_hull\_y2;

//Enemy Color//

en5\_red = 10'hFFF;

en5\_green = 10'hFFF;

en5\_blue = 10'hFFF;

//Enemy Shot//

en\_shot5 = 1'b0;

///Enemy 6///

//alive

alive6 = 1'b1;

//hull//

en\_hull6\_x1 = en\_hull5\_x1 + 50;

en\_hull6\_x2 = en\_hull5\_x2 + 50;

en\_hull6\_y1 = en\_hull\_y1;

en\_hull6\_y2 = en\_hull\_y2;

//Enemy Color//

en6\_red = 10'hFFF;

en6\_green = 10'hFFF;

en6\_blue = 10'hFFF;

//Enemy Shot//

en\_shot6 = 1'b0;

///Enemy 7///

//alive

alive7 = 1'b1;

//hull//

en\_hull7\_x1 = en\_hull6\_x1 + 50;

en\_hull7\_x2 = en\_hull6\_x2 + 50;

en\_hull7\_y1 = en\_hull\_y1;

en\_hull7\_y2 = en\_hull\_y2;

//Enemy Color//

en7\_red = 10'hFFF;

en7\_green = 10'hFFF;

en7\_blue = 10'hFFF;

//Enemy Shot//

en\_shot7 = 1'b0;

///Enemy 8///

//alive

alive8 = 1'b1;

//hull//

en\_hull8\_x1 = en\_hull7\_x1 + 50;

en\_hull8\_x2 = en\_hull7\_x2 + 50;

en\_hull8\_y1 = en\_hull\_y1;

en\_hull8\_y2 = en\_hull\_y2;

//Enemy Color//

en8\_red = 10'hFFF;

en8\_green = 10'hFFF;

en8\_blue = 10'hFFF;

//Enemy Shot//

en\_shot8 = 1'b0;

///Enemy 9///

//alive

alive9 = 1'b1;

//hull//

en\_hull9\_x1 = 25;

en\_hull9\_x2 = en\_hull9\_x1 + 25;

en\_hull9\_y1 = en\_hull\_y1 + 100;

en\_hull9\_y2 = en\_hull\_y2 + 100;

//Enemy Color//

en9\_red = 10'h000;

en9\_green = 10'h000;

en9\_blue = 10'hFFF;

//Enemy Shot//

en\_shot9 = 1'b0;

///Enemy 10///

//alive

alive10 = 1'b1;

//hull//

en\_hull10\_x1 = en\_hull9\_x1 + 50;

en\_hull10\_x2 = en\_hull10\_x1 + 25;

en\_hull10\_y1 = en\_hull\_y1 + 100;

en\_hull10\_y2 = en\_hull\_y2 + 100;

//Enemy Color//

en10\_red = 10'h000;

en10\_green = 10'h000;

en10\_blue = 10'hFFF;

//Enemy Shot//

en\_shot10 = 1'b0;

///Enemy 11///

//alive

alive11 = 1'b1;

//hull//

en\_hull11\_x1 = en\_hull10\_x1 + 50;

en\_hull11\_x2 = en\_hull11\_x1 + 25;

en\_hull11\_y1 = en\_hull\_y1 + 100;

en\_hull11\_y2 = en\_hull\_y2 + 100;

//Enemy Color//

en11\_red = 10'h000;

en11\_green = 10'h000;

en11\_blue = 10'hFFF;

//Enemy Shot//

en\_shot11 = 1'b0;

///Enemy 12///

//alive

alive12 = 1'b1;

//hull//

en\_hull12\_x1 = en\_hull11\_x1 + 50;

en\_hull12\_x2 = en\_hull12\_x1 + 25;

en\_hull12\_y1 = en\_hull\_y1 + 100;

en\_hull12\_y2 = en\_hull\_y2 + 100;

//Enemy Color//

en12\_red = 10'h000;

en12\_green = 10'h000;

en12\_blue = 10'hFFF;

//Enemy Shot//

en\_shot12 = 1'b0;

///Enemy 13///

//alive

alive13 = 1'b1;

//hull//

en\_hull13\_x1 = en\_hull12\_x1 + 50;

en\_hull13\_x2 = en\_hull13\_x1 + 25;

en\_hull13\_y1 = en\_hull\_y1 + 100;

en\_hull13\_y2 = en\_hull\_y2 + 100;

//Enemy Color//

en13\_red = 10'h000;

en13\_green = 10'h000;

en13\_blue = 10'hFFF;

//Enemy Shot//

en\_shot13 = 1'b0;

///Enemy 14///

//alive

alive14 = 1'b1;

//hull//

en\_hull14\_x1 = en\_hull13\_x1 + 50;

en\_hull14\_x2 = en\_hull14\_x1 + 25;

en\_hull14\_y1 = en\_hull\_y1 + 100;

en\_hull14\_y2 = en\_hull\_y2 + 100;

//Enemy Color//

en14\_red = 10'h000;

en14\_green = 10'h000;

en14\_blue = 10'hFFF;

//Enemy Shot//

en\_shot14 = 1'b0;

///Enemy 15///

//alive

alive15 = 1'b1;

//hull//

en\_hull15\_x1 = en\_hull14\_x1 + 50;

en\_hull15\_x2 = en\_hull15\_x1 + 25;

en\_hull15\_y1 = en\_hull\_y1 + 100;

en\_hull15\_y2 = en\_hull\_y2 + 100;

//Enemy Color//

en15\_red = 10'h000;

en15\_green = 10'h000;

en15\_blue = 10'hFFF;

//Enemy Shot//

en\_shot15 = 1'b0;

///Enemy 16///

//alive

alive16 = 1'b1;

//hull//

en\_hull16\_x1 = en\_hull15\_x1 + 50;

en\_hull16\_x2 = en\_hull16\_x1 + 25;

en\_hull16\_y1 = en\_hull\_y1 + 100;

en\_hull16\_y2 = en\_hull\_y2 + 100;

//Enemy Color//

en16\_red = 10'h000;

en16\_green = 10'h000;

en16\_blue = 10'hFFF;

//Enemy Shot//

en\_shot16 = 1'b0;

///Enemy 17///

//alive

alive17 = 1'b1;

//hull//

en\_hull17\_x1 = en\_hull\_x1 + 50;

en\_hull17\_x2 = en\_hull17\_x1 + 25;

en\_hull17\_y1 = en\_hull\_y1 - 100;

en\_hull17\_y2 = en\_hull\_y2 - 100;

//Enemy Color//

en17\_red = 10'h000;

en17\_green = 10'hFFF;

en17\_blue = 10'h000;

//Enemy Shot//

en\_shot17 = 1'b0;

///Enemy 18///

//alive

alive18 = 1'b1;

//hull//

en\_hull18\_x1 = en\_hull17\_x1 + 50;

en\_hull18\_x2 = en\_hull18\_x1 + 25;

en\_hull18\_y1 = en\_hull\_y1 - 100;

en\_hull18\_y2 = en\_hull\_y2 - 100;

//Enemy Color//

en18\_red = 10'h000;

en18\_green = 10'hFFF;

en18\_blue = 10'h000;

//Enemy Shot//

en\_shot18 = 1'b0;

///Enemy 19///

//alive

alive19 = 1'b1;

//hull//

en\_hull19\_x1 = en\_hull18\_x1 + 50;

en\_hull19\_x2 = en\_hull19\_x1 + 25;

en\_hull19\_y1 = en\_hull\_y1 - 100;

en\_hull19\_y2 = en\_hull\_y2 - 100;

//Enemy Color//

en19\_red = 10'h000;

en19\_green = 10'hFFF;

en19\_blue = 10'h000;

//Enemy Shot//

en\_shot19 = 1'b0;

///Enemy 20///

//alive

alive20 = 1'b1;

//hull//

en\_hull20\_x1 = en\_hull19\_x1 + 50;

en\_hull20\_x2 = en\_hull20\_x1 + 25;

en\_hull20\_y1 = en\_hull\_y1 - 100;

en\_hull20\_y2 = en\_hull\_y2 - 100;

//Enemy Color//

en20\_red = 10'h000;

en20\_green = 10'hFFF;

en20\_blue = 10'h000;

//Enemy Shot//

en\_shot20 = 1'b0;

///Enemy 21///

//alive

alive21 = 1'b1;

//hull//

en\_hull21\_x1 = en\_hull20\_x1 + 50;

en\_hull21\_x2 = en\_hull21\_x1 + 25;

en\_hull21\_y1 = en\_hull\_y1 - 100;

en\_hull21\_y2 = en\_hull\_y2 - 100;

//Enemy Color//

en21\_red = 10'h000;

en21\_green = 10'hFFF;

en21\_blue = 10'h000;

//Enemy Shot//

en\_shot21 = 1'b0;

///Enemy 22///

//alive

alive22 = 1'b1;

//hull//

en\_hull22\_x1 = en\_hull21\_x1 + 50;

en\_hull22\_x2 = en\_hull22\_x1 + 25;

en\_hull22\_y1 = en\_hull\_y1 - 100;

en\_hull22\_y2 = en\_hull\_y2 - 100;

//Enemy Color//

en22\_red = 10'h000;

en22\_green = 10'hFFF;

en22\_blue = 10'h000;

//Enemy Shot//

en\_shot22 = 1'b0;

///Enemy 23///

//alive

alive23 = 1'b1;

//hull//

en\_hull23\_x1 = en\_hull22\_x1 + 50;

en\_hull23\_x2 = en\_hull23\_x1 + 25;

en\_hull23\_y1 = en\_hull\_y1 - 100;

en\_hull23\_y2 = en\_hull\_y2 - 100;

//Enemy Color//

en23\_red = 10'h000;

en23\_green = 10'hFFF;

en23\_blue = 10'h000;

//Enemy Shot//

en\_shot23 = 1'b0;

///Enemy 24///

//alive

alive24 = 1'b1;

//hull//

en\_hull24\_x1 = en\_hull23\_x1 + 50;

en\_hull24\_x2 = en\_hull24\_x1 + 25;

en\_hull24\_y1 = en\_hull\_y1 - 100;

en\_hull24\_y2 = en\_hull\_y2 - 100;

//Enemy Color//

en24\_red = 10'h000;

en24\_green = 10'hFFF;

en24\_blue = 10'h000;

//Enemy Shot//

en\_shot24 = 1'b0;

//movement

en\_high\_x1 = en\_hull17\_x1;

en\_high\_x2 = en\_hull24\_x2;

en\_mid\_x1 = en\_hull\_x1;

en\_mid\_x2 = en\_hull8\_x2;

en\_low\_x1 = en\_hull9\_x1;

en\_low\_x2 = en\_hull16\_x2;

offset = 32'h00000000;

moveCounter = 32'h00000000;

direction\_high = 1;

direction = 1;

direction\_low = 1;

end

integer y1, y2, timer, count, dir, arr\_r, arr\_l, i1, i2, arr\_b1, arr\_x;

reg clk25;

always @( posedge clk ) begin

clk25 = !clk25;

timer = timer + 1;

///Background///

//This if statement should add start every 32nd pixel

//counting from left to right

i1 = i1 + 120;

i2 = i2 + 1;

if (i1 >= 640) begin i1 = i1 - 640; end

if (i2 >= 480) begin i2 = 0; end

mRed = (VGA\_X >= i1 - 1) && (VGA\_X <= i1 + 1) && (VGA\_Y >= i2 - 1) && (VGA\_Y <= i2 + 1) ? 10'hFFF : 10'b0000000000;

mGreen = (VGA\_X >= i1 - 1) && (VGA\_X <= i1 + 1) && (VGA\_Y >= i2 - 1) && (VGA\_Y <= i2 + 1) ? 10'hFFF : 10'b0000000000;

mBlue = (VGA\_X >= i1 - 1) && (VGA\_X <= i1 + 1) && (VGA\_Y >= i2 - 1) && (VGA\_Y <= i2 + 1) ? 10'hFFF : 10'b0000000000;//(VGA\_X == i1) && (VGA\_Y == i2)

//updating the ship location

ship\_hull\_x1 = 62+ship\_x;

ship\_hull\_x2 = 74+ship\_x;

ship\_hull\_y1 = 452+ship\_y;

ship\_hull\_y2 = 476+ship\_y;

if ( timer >= 1250000/2) begin

timer = 0;

//=== Enemy Movement ===\\

//if ( en\_hull8\_x2 >= 640 || en\_hull\_x1 <= 0) begin //The bound should probably be their own variable

if ( en\_mid\_x2 >= 640 || en\_mid\_x1 <= 0) begin

direction = -direction;

end

en\_hull\_x1 = en\_hull\_x1 + direction;

en\_hull\_x2 = en\_hull\_x2 + direction;

en\_hull2\_x1 = en\_hull2\_x1 + direction;

en\_hull2\_x2 = en\_hull2\_x2 + direction;

en\_hull3\_x1 = en\_hull3\_x1 + direction;

en\_hull3\_x2 = en\_hull3\_x2 + direction;

en\_hull4\_x1 = en\_hull4\_x1 + direction;

en\_hull4\_x2 = en\_hull4\_x2 + direction;

en\_hull5\_x1 = en\_hull5\_x1 + direction;

en\_hull5\_x2 = en\_hull5\_x2 + direction;

en\_hull6\_x1 = en\_hull6\_x1 + direction;

en\_hull6\_x2 = en\_hull6\_x2 + direction;

en\_hull7\_x1 = en\_hull7\_x1 + direction;

en\_hull7\_x2 = en\_hull7\_x2 + direction;

en\_hull8\_x1 = en\_hull8\_x1 + direction;

en\_hull8\_x2 = en\_hull8\_x2 + direction;

if ( en\_low\_x2 >= 640 || en\_low\_x1 <= 0) begin

direction\_low = -direction\_low;

end

en\_hull9\_x1 = en\_hull9\_x1 - direction\_low;

en\_hull9\_x2 = en\_hull9\_x2 - direction\_low;

en\_hull10\_x1 = en\_hull10\_x1 - direction\_low;

en\_hull10\_x2 = en\_hull10\_x2 - direction\_low;

en\_hull11\_x1 = en\_hull11\_x1 - direction\_low;

en\_hull11\_x2 = en\_hull11\_x2 - direction\_low;

en\_hull12\_x1 = en\_hull12\_x1 - direction\_low;

en\_hull12\_x2 = en\_hull12\_x2 - direction\_low;

en\_hull13\_x1 = en\_hull13\_x1 - direction\_low;

en\_hull13\_x2 = en\_hull13\_x2 - direction\_low;

en\_hull14\_x1 = en\_hull14\_x1 - direction\_low;

en\_hull14\_x2 = en\_hull14\_x2 - direction\_low;

en\_hull15\_x1 = en\_hull15\_x1 - direction\_low;

en\_hull15\_x2 = en\_hull15\_x2 - direction\_low;

en\_hull16\_x1 = en\_hull16\_x1 - direction\_low;

en\_hull16\_x2 = en\_hull16\_x2 - direction\_low;

if ( en\_high\_x2 >= 640 || en\_high\_x1 <= 0) begin

direction\_high = -direction\_high;

end

en\_hull17\_x1 = en\_hull17\_x1 - direction\_high;

en\_hull17\_x2 = en\_hull17\_x2 - direction\_high;

en\_hull18\_x1 = en\_hull18\_x1 - direction\_high;

en\_hull18\_x2 = en\_hull18\_x2 - direction\_high;

en\_hull19\_x1 = en\_hull19\_x1 - direction\_high;

en\_hull19\_x2 = en\_hull19\_x2 - direction\_high;

en\_hull20\_x1 = en\_hull20\_x1 - direction\_high;

en\_hull20\_x2 = en\_hull20\_x2 - direction\_high;

en\_hull21\_x1 = en\_hull21\_x1 - direction\_high;

en\_hull21\_x2 = en\_hull21\_x2 - direction\_high;

en\_hull22\_x1 = en\_hull22\_x1 - direction\_high;

en\_hull22\_x2 = en\_hull22\_x2 - direction\_high;

en\_hull23\_x1 = en\_hull23\_x1 - direction\_high;

en\_hull23\_x2 = en\_hull23\_x2 - direction\_high;

en\_hull24\_x1 = en\_hull24\_x1 - direction\_high;

en\_hull24\_x2 = en\_hull24\_x2 - direction\_high;

en\_high\_x1 = alive17 ? en\_hull17\_x1 : //if enemy 1 is alive dont let it go off screen to the left

alive18 ? en\_hull18\_x1 : //if enemy 2 is alive dont let it go off screen to the left

alive19 ? en\_hull19\_x1 : //if enemy 3 is alive dont let it go off screen to the left

alive20 ? en\_hull20\_x1 : //if enemy 4 is alive dont let it go off screen to the left

alive21 ? en\_hull21\_x1 : //if enemy 5 is alive dont let it go off screen to the left

alive22 ? en\_hull22\_x1 : //if enemy 6 is alive dont let it go off screen to the left

alive23 ? en\_hull23\_x1 : //if enemy 7 is alive dont let it go off screen to the left

alive24 ? en\_hull24\_x1 : //if enemy 8 is alive dont let it go off screen

0; //I dont really know what this should be

en\_high\_x2 = alive24 ? en\_hull24\_x2 : //If enemy 8 is alive dont let it go off screen to the right

alive23 ? en\_hull23\_x2 : //If enemy 7 is alive dont let it go off screen to the right

alive22 ? en\_hull22\_x2 : //If enemy 6 is alive dont let it go off screen to the right

alive21 ? en\_hull21\_x2 : //If enemy 5 is alive dont let it go off screen to the right

alive20 ? en\_hull20\_x2 : //If enemy 4 is alive dont let it go off screen to the right

alive19 ? en\_hull19\_x2 : //If enemy 3 is alive dont let it go off screen to the right

alive18 ? en\_hull18\_x2 : //If enemy 2 is alive dont let it go off screen to the right

alive17 ? en\_hull17\_x2 : //If enemy 1 is alive dont let it go off screen to the right

0; //I dont really know what to make this value either

en\_mid\_x1 = alive ? en\_hull\_x1 : //if enemy 1 is alive dont let it go off screen to the left

alive2 ? en\_hull2\_x1 : //if enemy 2 is alive dont let it go off screen to the left

alive3 ? en\_hull3\_x1 : //if enemy 3 is alive dont let it go off screen to the left

alive4 ? en\_hull4\_x1 : //if enemy 4 is alive dont let it go off screen to the left

alive5 ? en\_hull5\_x1 : //if enemy 5 is alive dont let it go off screen to the left

alive6 ? en\_hull6\_x1 : //if enemy 6 is alive dont let it go off screen to the left

alive7 ? en\_hull7\_x1 : //if enemy 7 is alive dont let it go off screen to the left

alive8 ? en\_hull8\_x1 : //if enemy 8 is alive dont let it go off screen

0; //I dont really know what this should be

en\_mid\_x2 = alive8 ? en\_hull8\_x2 : //If enemy 8 is alive dont let it go off screen to the right

alive7 ? en\_hull7\_x2 : //If enemy 7 is alive dont let it go off screen to the right

alive6 ? en\_hull6\_x2 : //If enemy 6 is alive dont let it go off screen to the right

alive5 ? en\_hull5\_x2 : //If enemy 5 is alive dont let it go off screen to the right

alive4 ? en\_hull4\_x2 : //If enemy 4 is alive dont let it go off screen to the right

alive3 ? en\_hull3\_x2 : //If enemy 3 is alive dont let it go off screen to the right

alive2 ? en\_hull2\_x2 : //If enemy 2 is alive dont let it go off screen to the right

alive ? en\_hull\_x2 : //If enemy 1 is alive dont let it go off screen to the right

0; //I dont really know what to make this value either

en\_low\_x1 = alive9 ? en\_hull9\_x1 : //if enemy 9 is alive dont let it go off screen to the left

alive10 ? en\_hull10\_x1: //if enemy 10 is alive dont let it go off screen to the left

alive11 ? en\_hull11\_x1: //if enemy 11 is alive dont let it go off screen to the left

alive12 ? en\_hull12\_x1: //if enemy 12 is alive dont let it go off screen to the left

alive13 ? en\_hull13\_x1: //if enemy 13 is alive dont let it go off screen to the left

alive14 ? en\_hull14\_x1: //if enemy 14 is alive dont let it go off screen to the left

alive15 ? en\_hull15\_x1: //if enemy 15 is alive dont let it go off screen to the left

alive16 ? en\_hull16\_x1: //if enemy 16 is alive dont let it go off screen

0; //I dont really know what this should be

en\_low\_x2 = alive16 ? en\_hull16\_x2: //if enemy 16 is alive dont let it go off screen

alive15 ? en\_hull15\_x2: //if enemy 15 is alive dont let it go off screen to the left

alive14 ? en\_hull14\_x2: //if enemy 14 is alive dont let it go off screen to the left

alive13 ? en\_hull13\_x2: //if enemy 13 is alive dont let it go off screen to the left

alive12 ? en\_hull12\_x2: //if enemy 12 is alive dont let it go off screen to the left

alive11 ? en\_hull11\_x2: //if enemy 11 is alive dont let it go off screen to the left

alive10 ? en\_hull10\_x2: //if enemy 10 is alive dont let it go off screen to the left

alive9 ? en\_hull9\_x2: //if enemy 9 is alive dont let it go off screen to the left

0; //I dont really know what this should be

if(!key[0] || !gamepad[1])begin

//right

//xFlag = 1;

if ( ship\_hull\_x2 + 12 == 640) begin

//Do nothing

end else begin

ship\_x = ship\_x + 2;

//eox = eox + 1;

bt1x = bt1x + 2;

bt2x = bt2x - 2;

end

end

if(!key[1] || !gamepad[0])begin

//left

if (ship\_hull\_x1 - 12 == 0) begin

//Do nothing

end else begin

ship\_x = ship\_x - 2;

//eox = eox - 1;

bt1x = bt1x - 2;

bt2x = bt2x + 2;

end

end

if(!key[2] || !gamepad[2])begin

//down

// eoy = eoy + 1;

// bt1y = bt1y + 1;

// bt2y = bt1y - 1;

if (!shot\_act) begin

shot\_act = 1;

shotx1 = ship\_hull\_x1 + 5;

shotx2 = ship\_hull\_x1 + 6;

shoty1 = ship\_hull\_y1 - 2;

shoty2 = ship\_hull\_y1 + 3;

end

end

if(!key[3])begin

//up

// eoy = eoy - 1;

// bt1y = bt1y - 1;

// bt2y = bt2y + 1;

end

if ( exp\_live ) begin

exp\_radius = exp\_radius + 10;

end

end

if ( (!sw[9]) && (!sw[8]) && (!sw[7]) && (sw[6]) && (!sw[5]) && (!sw[4]) && (!sw[3]) && (sw[2]) && (!sw[1]) && (sw[0]) ) begin

///Draw Cocket Ship///

//hull//

if ( (VGA\_X >= ship\_hull\_x1) && (VGA\_X <= ship\_hull\_x2) && (VGA\_Y >= ship\_hull\_y1) && (VGA\_Y <= ship\_hull\_y2) && (ship\_alive) ) begin

mRed = ship\_red;

mGreen = ship\_green;

mBlue = ship\_blue;

end

//MIKES SIR-CLE//

if ( ((VGA\_X - (ship\_hull\_x1+12/2))\*\*2 + (VGA\_Y - ship\_hull\_y1)\*\*2 <= (12/2)\*\*2) && ship\_alive ) begin

mRed = ship\_red;

mGreen = ship\_green;

mBlue = ship\_blue;

end

//Spherical Fuel Tank Left//

if ( ((VGA\_X - (ship\_hull\_x1))\*\*2 + (VGA\_Y - ship\_hull\_y2)\*\*2 <= (12/2)\*\*2) && ship\_alive ) begin

mRed = ship\_red;

mGreen = ship\_green;

mBlue = ship\_blue;

end

//Spherical Fuel Tank Right//

if ( ((VGA\_X - (ship\_hull\_x2))\*\*2 + (VGA\_Y - ship\_hull\_y2)\*\*2 <= (12/2)\*\*2) && ship\_alive ) begin

mRed = ship\_red;

mGreen = ship\_green;

mBlue = ship\_blue;

end

end else begin

///Draw Ship///

//hull//

if ( ((VGA\_X >= ship\_hull\_x1) && (VGA\_X <= ship\_hull\_x2) && (VGA\_Y >= ship\_hull\_y1) && (VGA\_Y <= ship\_hull\_y2)) && ship\_alive ) begin

mRed = ship\_red;

mGreen = ship\_green;

mBlue = ship\_blue;

end

///MIKES TRIANGLE LEFT///

if ( ((VGA\_X >= ship\_hull\_x1 - 12) && (VGA\_X <= ship\_hull\_x1) && (VGA\_Y >= -VGA\_X + bt1x + bt1y) && (VGA\_Y <= ship\_hull\_y2)) && ship\_alive ) begin

mRed = 10'h0000;

mGreen = 10'hFFFF;

mBlue = 10'h000;

end

//MIKES TRIANGLE RIGHT//

if ( ((VGA\_X >= ship\_hull\_x2) && (VGA\_X <= ship\_hull\_x2 + 12) && (VGA\_Y >= VGA\_X + bt2x + bt2y) && (VGA\_Y <= ship\_hull\_y2)) && ship\_alive )begin

mRed = 10'h0000;

mGreen = 10'hFFFF;

mBlue = 10'h000;

end

//MIKES SIR-CLE//

if ( ((VGA\_X - (ship\_hull\_x1+12/2))\*\*2 + (VGA\_Y - ship\_hull\_y1)\*\*2 <= (12/2)\*\*2) && ship\_alive ) begin

mRed = ship\_red;

mGreen = ship\_green;

mBlue = ship\_blue;

end

end

///Shot///

if ( ((VGA\_X >= shotx1) && (VGA\_X <= shotx2) && (VGA\_Y >= shoty1) && (VGA\_Y <= shoty2)) && (shot\_act) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((shoty1 >= en\_hull\_y1) && (shoty1 <= en\_hull\_y2) && (shotx1 >= en\_hull\_x1) && (shotx2 <= en\_hull\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull\_x1 + ((en\_hull\_x2 - en\_hull\_x1)/2);

exp\_y1 = en\_hull\_y2;

exp\_live = 1'b1;

end

alive = 1'b0;

en\_hull\_x1 = 0;

en\_hull\_x2 = 0;

en\_hull\_y1 = 0;

en\_hull\_y2 = 0;

en\_red = 10'h000;

en\_green = 10'h000;

en\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull2\_y1) && (shoty1 <= en\_hull2\_y2) && (shotx1 >= en\_hull2\_x1) && (shotx2 <= en\_hull2\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull2\_x1 + ((en\_hull2\_x2 - en\_hull2\_x1)/2);

exp\_y1 = en\_hull2\_y2;

exp\_live = 1'b1;

end

alive2 = 1'b0;

en\_hull2\_x1 = 0;

en\_hull2\_x2 = 0;

en\_hull2\_y1 = 0;

en\_hull2\_y2 = 0;

en2\_red = 10'h000;

en2\_green = 10'h000;

en2\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull3\_y1) && (shoty1 <= en\_hull3\_y2) && (shotx1 >= en\_hull3\_x1) && (shotx2 <= en\_hull3\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull3\_x1 + ((en\_hull3\_x2 - en\_hull3\_x1)/2);

exp\_y1 = en\_hull3\_y2;

exp\_live = 1'b1;

end

alive3 = 1'b0;

en\_hull3\_x1 = 0;

en\_hull3\_x2 = 0;

en\_hull3\_y1 = 0;

en\_hull3\_y2 = 0;

en3\_red = 10'h000;

en3\_green = 10'h000;

en3\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull4\_y1) && (shoty1 <= en\_hull4\_y2) && (shotx1 >= en\_hull4\_x1) && (shotx2 <= en\_hull4\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull4\_x1 + ((en\_hull4\_x2 - en\_hull4\_x1)/2);

exp\_y1 = en\_hull4\_y2;

exp\_live = 1'b1;

end

alive4 = 1'b0;

en\_hull4\_x1 = 0;

en\_hull4\_x2 = 0;

en\_hull4\_y1 = 0;

en\_hull4\_y2 = 0;

en4\_red = 10'h000;

en4\_green = 10'h000;

en4\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull5\_y1) && (shoty1 <= en\_hull5\_y2) && (shotx1 >= en\_hull5\_x1) && (shotx2 <= en\_hull5\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull5\_x1 + ((en\_hull5\_x2 - en\_hull5\_x1)/2);

exp\_y1 = en\_hull5\_y2;

exp\_live = 1'b1;

end

alive5 = 1'b0;

en\_hull5\_x1 = 0;

en\_hull5\_x2 = 0;

en\_hull5\_y1 = 0;

en\_hull5\_y2 = 0;

en5\_red = 10'h000;

en5\_green = 10'h000;

en5\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull6\_y1) && (shoty1 <= en\_hull6\_y2) && (shotx1 >= en\_hull6\_x1) && (shotx2 <= en\_hull6\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull6\_x1 + ((en\_hull6\_x2 - en\_hull6\_x1)/2);

exp\_y1 = en\_hull6\_y2;

exp\_live = 1'b1;

end

alive6 = 1'b0;

en\_hull6\_x1 = 0;

en\_hull6\_x2 = 0;

en\_hull6\_y1 = 0;

en\_hull6\_y2 = 0;

en6\_red = 10'h000;

en6\_green = 10'h000;

en6\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull7\_y1) && (shoty1 <= en\_hull7\_y2) && (shotx1 >= en\_hull7\_x1) && (shotx2 <= en\_hull7\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull7\_x1 + ((en\_hull7\_x2 - en\_hull7\_x1)/2);

exp\_y1 = en\_hull7\_y2;

exp\_live = 1'b1;

end

alive7 = 1'b0;

en\_hull7\_x1 = 0;

en\_hull7\_x2 = 0;

en\_hull7\_y1 = 0;

en\_hull7\_y2 = 0;

en7\_red = 10'h000;

en7\_green = 10'h000;

en7\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull8\_y1) && (shoty1 <= en\_hull8\_y2) && (shotx1 >= en\_hull8\_x1) && (shotx2 <= en\_hull8\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull8\_x1 + ((en\_hull8\_x2 - en\_hull8\_x1)/2);

exp\_y1 = en\_hull8\_y2;

exp\_live = 1'b1;

end

alive8 = 1'b0;

en\_hull8\_x1 = 0;

en\_hull8\_x2 = 0;

en\_hull8\_y1 = 0;

en\_hull8\_y2 = 0;

en8\_red = 10'h000;

en8\_green = 10'h000;

en8\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull9\_y1) && (shoty1 <= en\_hull9\_y2) && (shotx1 >= en\_hull9\_x1) && (shotx2 <= en\_hull9\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull9\_x1 + ((en\_hull9\_x2 - en\_hull9\_x1)/2);

exp\_y1 = en\_hull9\_y2;

exp\_live = 1'b1;

end

alive9 = 1'b0;

en\_hull9\_x1 = 0;

en\_hull9\_x2 = 0;

en\_hull9\_y1 = 0;

en\_hull9\_y2 = 0;

en9\_red = 10'h000;

en9\_green = 10'h000;

en9\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull10\_y1) && (shoty1 <= en\_hull10\_y2) && (shotx1 >= en\_hull10\_x1) && (shotx2 <= en\_hull10\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull10\_x1 + ((en\_hull10\_x2 - en\_hull10\_x1)/2);

exp\_y1 = en\_hull10\_y2;

exp\_live = 1'b1;

end

alive10 = 1'b0;

en\_hull10\_x1 = 0;

en\_hull10\_x2 = 0;

en\_hull10\_y1 = 0;

en\_hull10\_y2 = 0;

en10\_red = 10'h000;

en10\_green = 10'h000;

en10\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull11\_y1) && (shoty1 <= en\_hull11\_y2) && (shotx1 >= en\_hull11\_x1) && (shotx2 <= en\_hull11\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull11\_x1 + ((en\_hull11\_x2 - en\_hull11\_x1)/2);

exp\_y1 = en\_hull11\_y2;

exp\_live = 1'b1;

end

alive11 = 1'b0;

en\_hull11\_x1 = 0;

en\_hull11\_x2 = 0;

en\_hull11\_y1 = 0;

en\_hull11\_y2 = 0;

en11\_red = 10'h000;

en11\_green = 10'h000;

en11\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull12\_y1) && (shoty1 <= en\_hull12\_y2) && (shotx1 >= en\_hull12\_x1) && (shotx2 <= en\_hull12\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull12\_x1 + ((en\_hull12\_x2 - en\_hull12\_x1)/2);

exp\_y1 = en\_hull12\_y2;

exp\_live = 1'b1;

end

alive12 = 1'b0;

en\_hull12\_x1 = 0;

en\_hull12\_x2 = 0;

en\_hull12\_y1 = 0;

en\_hull12\_y2 = 0;

en12\_red = 10'h000;

en12\_green = 10'h000;

en12\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull13\_y1) && (shoty1 <= en\_hull13\_y2) && (shotx1 >= en\_hull13\_x1) && (shotx2 <= en\_hull13\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull13\_x1 + ((en\_hull13\_x2 - en\_hull13\_x1)/2);

exp\_y1 = en\_hull13\_y2;

exp\_live = 1'b1;

end

alive13 = 1'b0;

en\_hull13\_x1 = 0;

en\_hull13\_x2 = 0;

en\_hull13\_y1 = 0;

en\_hull13\_y2 = 0;

en13\_red = 10'h000;

en13\_green = 10'h000;

en13\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull14\_y1) && (shoty1 <= en\_hull14\_y2) && (shotx1 >= en\_hull14\_x1) && (shotx2 <= en\_hull14\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull14\_x1 + ((en\_hull14\_x2 - en\_hull14\_x1)/2);

exp\_y1 = en\_hull14\_y2;

exp\_live = 1'b1;

end

alive14 = 1'b0;

en\_hull14\_x1 = 0;

en\_hull14\_x2 = 0;

en\_hull14\_y1 = 0;

en\_hull14\_y2 = 0;

en14\_red = 10'h000;

en14\_green = 10'h000;

en14\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull15\_y1) && (shoty1 <= en\_hull15\_y2) && (shotx1 >= en\_hull15\_x1) && (shotx2 <= en\_hull15\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull15\_x1 + ((en\_hull15\_x2 - en\_hull15\_x1)/2);

exp\_y1 = en\_hull15\_y2;

exp\_live = 1'b1;

end

alive15 = 1'b0;

en\_hull15\_x1 = 0;

en\_hull15\_x2 = 0;

en\_hull15\_y1 = 0;

en\_hull15\_y2 = 0;

en15\_red = 10'h000;

en15\_green = 10'h000;

en15\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull16\_y1) && (shoty1 <= en\_hull16\_y2) && (shotx1 >= en\_hull16\_x1) && (shotx2 <= en\_hull16\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull16\_x1 + ((en\_hull16\_x2 - en\_hull16\_x1)/2);

exp\_y1 = en\_hull16\_y2;

exp\_live = 1'b1;

end

alive16 = 1'b0;

en\_hull16\_x1 = 0;

en\_hull16\_x2 = 0;

en\_hull16\_y1 = 0;

en\_hull16\_y2 = 0;

en16\_red = 10'h000;

en16\_green = 10'h000;

en16\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull17\_y1) && (shoty1 <= en\_hull17\_y2) && (shotx1 >= en\_hull17\_x1) && (shotx2 <= en\_hull17\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull17\_x1 + ((en\_hull17\_x2 - en\_hull17\_x1)/2);

exp\_y1 = en\_hull17\_y2;

exp\_live = 1'b1;

end

alive17 = 1'b0;

en\_hull17\_x1 = 0;

en\_hull17\_x2 = 0;

en\_hull17\_y1 = 0;

en\_hull17\_y2 = 0;

en17\_red = 10'h000;

en17\_green = 10'h000;

en17\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull18\_y1) && (shoty1 <= en\_hull18\_y2) && (shotx1 >= en\_hull18\_x1) && (shotx2 <= en\_hull18\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull18\_x1 + ((en\_hull18\_x2 - en\_hull18\_x1)/2);

exp\_y1 = en\_hull18\_y2;

exp\_live = 1'b1;

end

alive18 = 1'b0;

en\_hull18\_x1 = 0;

en\_hull18\_x2 = 0;

en\_hull18\_y1 = 0;

en\_hull18\_y2 = 0;

en18\_red = 10'h000;

en18\_green = 10'h000;

en18\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull19\_y1) && (shoty1 <= en\_hull19\_y2) && (shotx1 >= en\_hull19\_x1) && (shotx2 <= en\_hull19\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull19\_x1 + ((en\_hull19\_x2 - en\_hull19\_x1)/2);

exp\_y1 = en\_hull19\_y2;

exp\_live = 1'b1;

end

alive19 = 1'b0;

en\_hull19\_x1 = 0;

en\_hull19\_x2 = 0;

en\_hull19\_y1 = 0;

en\_hull19\_y2 = 0;

en19\_red = 10'h000;

en19\_green = 10'h000;

en19\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull20\_y1) && (shoty1 <= en\_hull20\_y2) && (shotx1 >= en\_hull20\_x1) && (shotx2 <= en\_hull20\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull20\_x1 + ((en\_hull20\_x2 - en\_hull20\_x1)/2);

exp\_y1 = en\_hull20\_y2;

exp\_live = 1'b1;

end

alive20 = 1'b0;

en\_hull20\_x1 = 0;

en\_hull20\_x2 = 0;

en\_hull20\_y1 = 0;

en\_hull20\_y2 = 0;

en20\_red = 10'h000;

en20\_green = 10'h000;

en20\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull21\_y1) && (shoty1 <= en\_hull21\_y2) && (shotx1 >= en\_hull21\_x1) && (shotx2 <= en\_hull21\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull21\_x1 + ((en\_hull21\_x2 - en\_hull21\_x1)/2);

exp\_y1 = en\_hull21\_y2;

exp\_live = 1'b1;

end

alive21 = 1'b0;

en\_hull21\_x1 = 0;

en\_hull21\_x2 = 0;

en\_hull21\_y1 = 0;

en\_hull21\_y2 = 0;

en21\_red = 10'h000;

en21\_green = 10'h000;

en21\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull22\_y1) && (shoty1 <= en\_hull22\_y2) && (shotx1 >= en\_hull22\_x1) && (shotx2 <= en\_hull22\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull22\_x1 + ((en\_hull22\_x2 - en\_hull22\_x1)/2);

exp\_y1 = en\_hull22\_y2;

exp\_live = 1'b1;

end

alive22 = 1'b0;

en\_hull22\_x1 = 0;

en\_hull22\_x2 = 0;

en\_hull22\_y1 = 0;

en\_hull22\_y2 = 0;

en22\_red = 10'h000;

en22\_green = 10'h000;

en22\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull23\_y1) && (shoty1 <= en\_hull23\_y2) && (shotx1 >= en\_hull23\_x1) && (shotx2 <= en\_hull23\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull23\_x1 + ((en\_hull23\_x2 - en\_hull23\_x1)/2);

exp\_y1 = en\_hull23\_y2;

exp\_live = 1'b1;

end

alive23 = 1'b0;

en\_hull23\_x1 = 0;

en\_hull23\_x2 = 0;

en\_hull23\_y1 = 0;

en\_hull23\_y2 = 0;

en23\_red = 10'h000;

en23\_green = 10'h000;

en23\_blue = 10'h000;

shot\_act = 0;

end else if ((shoty1 >= en\_hull24\_y1) && (shoty1 <= en\_hull24\_y2) && (shotx1 >= en\_hull24\_x1) && (shotx2 <= en\_hull24\_x2)) begin

if (!exp\_live) begin

exp\_x1 = en\_hull24\_x1 + ((en\_hull24\_x2 - en\_hull24\_x1)/2);

exp\_y1 = en\_hull24\_y2;

exp\_live = 1'b1;

end

alive24 = 1'b0;

en\_hull24\_x1 = 0;

en\_hull24\_x2 = 0;

en\_hull24\_y1 = 0;

en\_hull24\_y2 = 0;

en24\_red = 10'h000;

en24\_green = 10'h000;

en24\_blue = 10'h000;

shot\_act = 0;

end else if (shoty1 == 0) begin

shot\_act = 0;

end else begin

shoty1 = shoty1 - 1;

shoty2 = shoty2 - 1;

end

end

//Setting Bullet position//

if ( alive && (en\_hull\_x1 <= ship\_hull\_x1) && (en\_hull\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot != 1'b1 ) begin

en\_shot = 1'b1;

en\_shot\_x1 = (en\_hull\_x1 + en\_hull\_x2)/2;

en\_shot\_x2 = (en\_hull\_x1 + en\_hull\_x2)/2 + 1;

en\_shot\_y1 = en\_hull\_y2;

en\_shot\_y2 = en\_hull\_y2 + 1;

end

end

if ( alive2 && (en\_hull2\_x1 <= ship\_hull\_x1) && (en\_hull2\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot2 != 1'b1 ) begin

en\_shot2 = 1'b1;

en\_shot2\_x1 = (en\_hull2\_x1 + en\_hull2\_x2)/2;

en\_shot2\_x2 = (en\_hull2\_x1 + en\_hull2\_x2)/2 + 1;

en\_shot2\_y1 = en\_hull2\_y2;

en\_shot2\_y2 = en\_hull2\_y2 + 1;

end

end

if ( alive3 && (en\_hull3\_x1 <= ship\_hull\_x1) && (en\_hull3\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot3 != 1'b1 ) begin

en\_shot3 = 1'b1;

en\_shot3\_x1 = (en\_hull3\_x1 + en\_hull3\_x2)/2;

en\_shot3\_x2 = (en\_hull3\_x1 + en\_hull3\_x2)/2 + 1;

en\_shot3\_y1 = en\_hull3\_y2;

en\_shot3\_y2 = en\_hull3\_y2 + 1;

end

end

if ( alive4 && (en\_hull4\_x1 <= ship\_hull\_x1) && (en\_hull4\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot4 != 1'b1 ) begin

en\_shot4 = 1'b1;

en\_shot4\_x1 = (en\_hull4\_x1 + en\_hull4\_x2)/2;

en\_shot4\_x2 = (en\_hull4\_x1 + en\_hull4\_x2)/2 + 1;

en\_shot4\_y1 = en\_hull4\_y2;

en\_shot4\_y2 = en\_hull4\_y2 + 1;

end

end

if ( alive5 && (en\_hull5\_x1 <= ship\_hull\_x1) && (en\_hull5\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot5 != 1'b1 ) begin

en\_shot5 = 1'b1;

en\_shot5\_x1 = (en\_hull5\_x1 + en\_hull5\_x2)/2;

en\_shot5\_x2 = (en\_hull5\_x1 + en\_hull5\_x2)/2 + 1;

en\_shot5\_y1 = en\_hull5\_y2;

en\_shot5\_y2 = en\_hull5\_y2 + 1;

end

end

if ( alive6 && (en\_hull6\_x1 <= ship\_hull\_x1) && (en\_hull6\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot6 != 1'b1 ) begin

en\_shot6 = 1'b1;

en\_shot6\_x1 = (en\_hull6\_x1 + en\_hull6\_x2)/2;

en\_shot6\_x2 = (en\_hull6\_x1 + en\_hull6\_x2)/2 + 1;

en\_shot6\_y1 = en\_hull6\_y2;

en\_shot6\_y2 = en\_hull6\_y2 + 1;

end

end

if ( alive7 && (en\_hull7\_x1 <= ship\_hull\_x1) && (en\_hull7\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot7 != 1'b1 ) begin

en\_shot7 = 1'b1;

en\_shot7\_x1 = (en\_hull7\_x1 + en\_hull7\_x2)/2;

en\_shot7\_x2 = (en\_hull7\_x1 + en\_hull7\_x2)/2 + 1;

en\_shot7\_y1 = en\_hull7\_y2;

en\_shot7\_y2 = en\_hull7\_y2 + 1;

end

end

if ( alive8 && (en\_hull8\_x1 <= ship\_hull\_x1) && (en\_hull8\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot8 != 1'b1 ) begin

en\_shot8 = 1'b1;

en\_shot8\_x1 = (en\_hull8\_x1 + en\_hull8\_x2)/2;

en\_shot8\_x2 = (en\_hull8\_x1 + en\_hull8\_x2)/2 + 1;

en\_shot8\_y1 = en\_hull8\_y2;

en\_shot8\_y2 = en\_hull8\_y2 + 1;

end

end

if ( alive9 && (en\_hull9\_x1 <= ship\_hull\_x1) && (en\_hull9\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot9 != 1'b1 ) begin

en\_shot9 = 1'b1;

en\_shot9\_x1 = (en\_hull9\_x1 + en\_hull9\_x2)/2;

en\_shot9\_x2 = (en\_hull9\_x1 + en\_hull9\_x2)/2 + 1;

en\_shot9\_y1 = en\_hull9\_y2;

en\_shot9\_y2 = en\_hull9\_y2 + 1;

end

end

if ( alive10 && (en\_hull10\_x1 <= ship\_hull\_x1) && (en\_hull10\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot10 != 1'b1 ) begin

en\_shot10 = 1'b1;

en\_shot10\_x1 = (en\_hull10\_x1 + en\_hull10\_x2)/2;

en\_shot10\_x2 = (en\_hull10\_x1 + en\_hull10\_x2)/2 + 1;

en\_shot10\_y1 = en\_hull10\_y2;

en\_shot10\_y2 = en\_hull10\_y2 + 1;

end

end

if ( alive11 && (en\_hull11\_x1 <= ship\_hull\_x1) && (en\_hull11\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot11 != 1'b1 ) begin

en\_shot11 = 1'b1;

en\_shot11\_x1 = (en\_hull11\_x1 + en\_hull11\_x2)/2;

en\_shot11\_x2 = (en\_hull11\_x1 + en\_hull11\_x2)/2 + 1;

en\_shot11\_y1 = en\_hull11\_y2;

en\_shot11\_y2 = en\_hull11\_y2 + 1;

end

end

if ( alive12 && (en\_hull12\_x1 <= ship\_hull\_x1) && (en\_hull12\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot12 != 1'b1 ) begin

en\_shot12 = 1'b1;

en\_shot12\_x1 = (en\_hull12\_x1 + en\_hull12\_x2)/2;

en\_shot12\_x2 = (en\_hull12\_x1 + en\_hull12\_x2)/2 + 1;

en\_shot12\_y1 = en\_hull12\_y2;

en\_shot12\_y2 = en\_hull12\_y2 + 1;

end

end

if ( alive13 && (en\_hull13\_x1 <= ship\_hull\_x1) && (en\_hull13\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot13 != 1'b1 ) begin

en\_shot13 = 1'b1;

en\_shot13\_x1 = (en\_hull13\_x1 + en\_hull13\_x2)/2;

en\_shot13\_x2 = (en\_hull13\_x1 + en\_hull13\_x2)/2 + 1;

en\_shot13\_y1 = en\_hull13\_y2;

en\_shot13\_y2 = en\_hull13\_y2 + 1;

end

end

if ( alive14 && (en\_hull14\_x1 <= ship\_hull\_x1) && (en\_hull14\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot14 != 1'b1 ) begin

en\_shot14 = 1'b1;

en\_shot14\_x1 = (en\_hull14\_x1 + en\_hull14\_x2)/2;

en\_shot14\_x2 = (en\_hull14\_x1 + en\_hull14\_x2)/2 + 1;

en\_shot14\_y1 = en\_hull14\_y2;

en\_shot14\_y2 = en\_hull14\_y2 + 1;

end

end

if ( alive15 && (en\_hull15\_x1 <= ship\_hull\_x1) && (en\_hull15\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot15 != 1'b1 ) begin

en\_shot15 = 1'b1;

en\_shot15\_x1 = (en\_hull15\_x1 + en\_hull15\_x2)/2;

en\_shot15\_x2 = (en\_hull15\_x1 + en\_hull15\_x2)/2 + 1;

en\_shot15\_y1 = en\_hull15\_y2;

en\_shot15\_y2 = en\_hull15\_y2 + 1;

end

end

if ( alive16 && (en\_hull16\_x1 <= ship\_hull\_x1) && (en\_hull16\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot16 != 1'b1 ) begin

en\_shot16 = 1'b1;

en\_shot16\_x1 = (en\_hull16\_x1 + en\_hull16\_x2)/2;

en\_shot16\_x2 = (en\_hull16\_x1 + en\_hull16\_x2)/2 + 1;

en\_shot16\_y1 = en\_hull16\_y2;

en\_shot16\_y2 = en\_hull16\_y2 + 1;

end

end

if ( alive17 && (en\_hull17\_x1 <= ship\_hull\_x1) && (en\_hull17\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot17 != 1'b1 ) begin

en\_shot17 = 1'b1;

en\_shot17\_x1 = (en\_hull17\_x1 + en\_hull17\_x2)/2;

en\_shot17\_x2 = (en\_hull17\_x1 + en\_hull17\_x2)/2 + 1;

en\_shot17\_y1 = en\_hull17\_y2;

en\_shot17\_y2 = en\_hull17\_y2 + 1;

end

end

if ( alive18 && (en\_hull18\_x1 <= ship\_hull\_x1) && (en\_hull18\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot18 != 1'b1 ) begin

en\_shot18 = 1'b1;

en\_shot18\_x1 = (en\_hull18\_x1 + en\_hull18\_x2)/2;

en\_shot18\_x2 = (en\_hull18\_x1 + en\_hull18\_x2)/2 + 1;

en\_shot18\_y1 = en\_hull18\_y2;

en\_shot18\_y2 = en\_hull18\_y2 + 1;

end

end

if ( alive19 && (en\_hull19\_x1 <= ship\_hull\_x1) && (en\_hull19\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot19 != 1'b1 ) begin

en\_shot19 = 1'b1;

en\_shot19\_x1 = (en\_hull19\_x1 + en\_hull19\_x2)/2;

en\_shot19\_x2 = (en\_hull19\_x1 + en\_hull19\_x2)/2 + 1;

en\_shot19\_y1 = en\_hull19\_y2;

en\_shot19\_y2 = en\_hull19\_y2 + 1;

end

end

if ( alive20 && (en\_hull20\_x1 <= ship\_hull\_x1) && (en\_hull20\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot20 != 1'b1 ) begin

en\_shot20 = 1'b1;

en\_shot20\_x1 = (en\_hull20\_x1 + en\_hull20\_x2)/2;

en\_shot20\_x2 = (en\_hull20\_x1 + en\_hull20\_x2)/2 + 1;

en\_shot20\_y1 = en\_hull20\_y2;

en\_shot20\_y2 = en\_hull20\_y2 + 1;

end

end

if ( alive21 && (en\_hull21\_x1 <= ship\_hull\_x1) && (en\_hull21\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot21 != 1'b1 ) begin

en\_shot21 = 1'b1;

en\_shot21\_x1 = (en\_hull21\_x1 + en\_hull21\_x2)/2;

en\_shot21\_x2 = (en\_hull21\_x1 + en\_hull21\_x2)/2 + 1;

en\_shot21\_y1 = en\_hull21\_y2;

en\_shot21\_y2 = en\_hull21\_y2 + 1;

end

end

if ( alive22 && (en\_hull22\_x1 <= ship\_hull\_x1) && (en\_hull22\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot22 != 1'b1 ) begin

en\_shot22 = 1'b1;

en\_shot22\_x1 = (en\_hull22\_x1 + en\_hull22\_x2)/2;

en\_shot22\_x2 = (en\_hull22\_x1 + en\_hull22\_x2)/2 + 1;

en\_shot22\_y1 = en\_hull22\_y2;

en\_shot22\_y2 = en\_hull22\_y2 + 1;

end

end

if ( alive23 && (en\_hull23\_x1 <= ship\_hull\_x1) && (en\_hull23\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot23 != 1'b1 ) begin

en\_shot23 = 1'b1;

en\_shot23\_x1 = (en\_hull23\_x1 + en\_hull23\_x2)/2;

en\_shot23\_x2 = (en\_hull23\_x1 + en\_hull23\_x2)/2 + 1;

en\_shot23\_y1 = en\_hull23\_y2;

en\_shot23\_y2 = en\_hull23\_y2 + 1;

end

end

if ( alive24 && (en\_hull24\_x1 <= ship\_hull\_x1) && (en\_hull24\_x2 >= ship\_hull\_x2) ) begin

if ( en\_shot24 != 1'b1 ) begin

en\_shot24 = 1'b1;

en\_shot24\_x1 = (en\_hull24\_x1 + en\_hull24\_x2)/2;

en\_shot24\_x2 = (en\_hull24\_x1 + en\_hull24\_x2)/2 + 1;

en\_shot24\_y1 = en\_hull24\_y2;

en\_shot24\_y2 = en\_hull24\_y2 + 1;

end

end

//Enemy Shootin//

if ( ((VGA\_X >= en\_shot\_x1) && (VGA\_X <= en\_shot\_x2) && (VGA\_Y >= en\_shot\_y1) && (VGA\_Y <= en\_shot\_y2)) && (en\_shot) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot\_y1 >= ship\_hull\_y1) && (en\_shot\_y2 <= ship\_hull\_y2) && (en\_shot\_x1 >= ship\_hull\_x1) && (en\_shot\_x2 <= ship\_hull\_x2)) begin

if (!exp\_live) begin

exp\_x1 = (ship\_hull\_x1 + ship\_hull\_x2)/2;

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot = 0;

end else if (en\_shot\_y1 == 480) begin

en\_shot = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot\_y1 = en\_shot\_y1 + 1;

en\_shot\_y2 = en\_shot\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot2\_x1) && (VGA\_X <= en\_shot2\_x2) && (VGA\_Y >= en\_shot2\_y1) && (VGA\_Y <= en\_shot2\_y2)) && (en\_shot2) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot2\_y1 >= ship\_hull\_y1) && (en\_shot2\_y2 <= ship\_hull\_y2) && (en\_shot2\_x1 >= ship\_hull\_x1) && (en\_shot2\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot2 = 0;

end else if (en\_shot2\_y1 == 480) begin

en\_shot2 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot2\_y1 = en\_shot2\_y1 + 1;

en\_shot2\_y2 = en\_shot2\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot3\_x1) && (VGA\_X <= en\_shot3\_x2) && (VGA\_Y >= en\_shot3\_y1) && (VGA\_Y <= en\_shot3\_y2)) && (en\_shot3) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot3\_y1 >= ship\_hull\_y1) && (en\_shot3\_y2 <= ship\_hull\_y2) && (en\_shot3\_x1 >= ship\_hull\_x1) && (en\_shot3\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot3 = 0;

end else if (en\_shot3\_y1 == 480) begin

en\_shot3 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot3\_y1 = en\_shot3\_y1 + 1;

en\_shot3\_y2 = en\_shot3\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot4\_x1) && (VGA\_X <= en\_shot4\_x2) && (VGA\_Y >= en\_shot4\_y1) && (VGA\_Y <= en\_shot4\_y2)) && (en\_shot4) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot4\_y1 >= ship\_hull\_y1) && (en\_shot4\_y2 <= ship\_hull\_y2) && (en\_shot4\_x1 >= ship\_hull\_x1) && (en\_shot4\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot4 = 0;

end else if (en\_shot4\_y1 == 480) begin

en\_shot4 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot4\_y1 = en\_shot4\_y1 + 1;

en\_shot4\_y2 = en\_shot4\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot5\_x1) && (VGA\_X <= en\_shot5\_x2) && (VGA\_Y >= en\_shot5\_y1) && (VGA\_Y <= en\_shot5\_y2)) && (en\_shot5) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot5\_y1 >= ship\_hull\_y1) && (en\_shot5\_y2 <= ship\_hull\_y2) && (en\_shot5\_x1 >= ship\_hull\_x1) && (en\_shot5\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot2 = 0;

end else if (en\_shot5\_y1 == 480) begin

en\_shot5 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot5\_y1 = en\_shot5\_y1 + 1;

en\_shot5\_y2 = en\_shot5\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot6\_x1) && (VGA\_X <= en\_shot6\_x2) && (VGA\_Y >= en\_shot6\_y1) && (VGA\_Y <= en\_shot6\_y2)) && (en\_shot6) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot6\_y1 >= ship\_hull\_y1) && (en\_shot6\_y2 <= ship\_hull\_y2) && (en\_shot6\_x1 >= ship\_hull\_x1) && (en\_shot6\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot6 = 0;

end else if (en\_shot6\_y1 == 480) begin

en\_shot6 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot6\_y1 = en\_shot6\_y1 + 1;

en\_shot6\_y2 = en\_shot6\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot7\_x1) && (VGA\_X <= en\_shot7\_x2) && (VGA\_Y >= en\_shot7\_y1) && (VGA\_Y <= en\_shot7\_y2)) && (en\_shot7) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot7\_y1 >= ship\_hull\_y1) && (en\_shot7\_y2 <= ship\_hull\_y2) && (en\_shot7\_x1 >= ship\_hull\_x1) && (en\_shot7\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot7 = 0;

end else if (en\_shot7\_y1 == 480) begin

en\_shot7 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot7\_y1 = en\_shot7\_y1 + 1;

en\_shot7\_y2 = en\_shot7\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot8\_x1) && (VGA\_X <= en\_shot8\_x2) && (VGA\_Y >= en\_shot8\_y1) && (VGA\_Y <= en\_shot8\_y2)) && (en\_shot8) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot8\_y1 >= ship\_hull\_y1) && (en\_shot8\_y2 <= ship\_hull\_y2) && (en\_shot8\_x1 >= ship\_hull\_x1) && (en\_shot8\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot8 = 0;

end else if (en\_shot8\_y1 == 480) begin

en\_shot8 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot8\_y1 = en\_shot8\_y1 + 1;

en\_shot8\_y2 = en\_shot8\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot9\_x1) && (VGA\_X <= en\_shot9\_x2) && (VGA\_Y >= en\_shot9\_y1) && (VGA\_Y <= en\_shot9\_y2)) && (en\_shot9) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot9\_y1 >= ship\_hull\_y1) && (en\_shot9\_y2 <= ship\_hull\_y2) && (en\_shot9\_x1 >= ship\_hull\_x1) && (en\_shot9\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot9 = 0;

end else if (en\_shot9\_y1 == 480) begin

en\_shot9 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot9\_y1 = en\_shot9\_y1 + 1;

en\_shot9\_y2 = en\_shot9\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot10\_x1) && (VGA\_X <= en\_shot10\_x2) && (VGA\_Y >= en\_shot10\_y1) && (VGA\_Y <= en\_shot10\_y2)) && (en\_shot10) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot10\_y1 >= ship\_hull\_y1) && (en\_shot10\_y2 <= ship\_hull\_y2) && (en\_shot10\_x1 >= ship\_hull\_x1) && (en\_shot10\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot10 = 0;

end else if (en\_shot10\_y1 == 480) begin

en\_shot10 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot10\_y1 = en\_shot10\_y1 + 1;

en\_shot10\_y2 = en\_shot10\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot11\_x1) && (VGA\_X <= en\_shot11\_x2) && (VGA\_Y >= en\_shot11\_y1) && (VGA\_Y <= en\_shot11\_y2)) && (en\_shot11) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot11\_y1 >= ship\_hull\_y1) && (en\_shot11\_y2 <= ship\_hull\_y2) && (en\_shot11\_x1 >= ship\_hull\_x1) && (en\_shot11\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot11 = 0;

end else if (en\_shot11\_y1 == 480) begin

en\_shot11 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot11\_y1 = en\_shot11\_y1 + 1;

en\_shot11\_y2 = en\_shot11\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot12\_x1) && (VGA\_X <= en\_shot12\_x2) && (VGA\_Y >= en\_shot12\_y1) && (VGA\_Y <= en\_shot12\_y2)) && (en\_shot12) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot12\_y1 >= ship\_hull\_y1) && (en\_shot12\_y2 <= ship\_hull\_y2) && (en\_shot12\_x1 >= ship\_hull\_x1) && (en\_shot12\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot12 = 0;

end else if (en\_shot12\_y1 == 480) begin

en\_shot12 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot12\_y1 = en\_shot12\_y1 + 1;

en\_shot12\_y2 = en\_shot12\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot13\_x1) && (VGA\_X <= en\_shot13\_x2) && (VGA\_Y >= en\_shot13\_y1) && (VGA\_Y <= en\_shot13\_y2)) && (en\_shot13) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot13\_y1 >= ship\_hull\_y1) && (en\_shot13\_y2 <= ship\_hull\_y2) && (en\_shot13\_x1 >= ship\_hull\_x1) && (en\_shot13\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot13 = 0;

end else if (en\_shot13\_y1 == 480) begin

en\_shot13 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot13\_y1 = en\_shot13\_y1 + 1;

en\_shot13\_y2 = en\_shot13\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot14\_x1) && (VGA\_X <= en\_shot14\_x2) && (VGA\_Y >= en\_shot14\_y1) && (VGA\_Y <= en\_shot14\_y2)) && (en\_shot14) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot14\_y1 >= ship\_hull\_y1) && (en\_shot14\_y2 <= ship\_hull\_y2) && (en\_shot14\_x1 >= ship\_hull\_x1) && (en\_shot14\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot14 = 0;

end else if (en\_shot14\_y1 == 480) begin

en\_shot14 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot14\_y1 = en\_shot14\_y1 + 1;

en\_shot14\_y2 = en\_shot14\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot15\_x1) && (VGA\_X <= en\_shot15\_x2) && (VGA\_Y >= en\_shot15\_y1) && (VGA\_Y <= en\_shot15\_y2)) && (en\_shot15) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot15\_y1 >= ship\_hull\_y1) && (en\_shot15\_y2 <= ship\_hull\_y2) && (en\_shot15\_x1 >= ship\_hull\_x1) && (en\_shot15\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot15 = 0;

end else if (en\_shot15\_y1 == 480) begin

en\_shot15 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot15\_y1 = en\_shot15\_y1 + 1;

en\_shot15\_y2 = en\_shot15\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot6\_x1) && (VGA\_X <= en\_shot6\_x2) && (VGA\_Y >= en\_shot6\_y1) && (VGA\_Y <= en\_shot6\_y2)) && (en\_shot6) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot6\_y1 >= ship\_hull\_y1) && (en\_shot6\_y2 <= ship\_hull\_y2) && (en\_shot6\_x1 >= ship\_hull\_x1) && (en\_shot6\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot6 = 0;

end else if (en\_shot6\_y1 == 480) begin

en\_shot6 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot6\_y1 = en\_shot6\_y1 + 1;

en\_shot6\_y2 = en\_shot6\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot17\_x1) && (VGA\_X <= en\_shot17\_x2) && (VGA\_Y >= en\_shot17\_y1) && (VGA\_Y <= en\_shot17\_y2)) && (en\_shot17) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot17\_y1 >= ship\_hull\_y1) && (en\_shot17\_y2 <= ship\_hull\_y2) && (en\_shot17\_x1 >= ship\_hull\_x1) && (en\_shot17\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot17 = 0;

end else if (en\_shot17\_y1 == 480) begin

en\_shot17 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot17\_y1 = en\_shot17\_y1 + 1;

en\_shot17\_y2 = en\_shot17\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot18\_x1) && (VGA\_X <= en\_shot18\_x2) && (VGA\_Y >= en\_shot18\_y1) && (VGA\_Y <= en\_shot18\_y2)) && (en\_shot18) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot18\_y1 >= ship\_hull\_y1) && (en\_shot18\_y2 <= ship\_hull\_y2) && (en\_shot18\_x1 >= ship\_hull\_x1) && (en\_shot18\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot18 = 0;

end else if (en\_shot18\_y1 == 480) begin

en\_shot18 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot18\_y1 = en\_shot18\_y1 + 1;

en\_shot18\_y2 = en\_shot18\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot19\_x1) && (VGA\_X <= en\_shot19\_x2) && (VGA\_Y >= en\_shot19\_y1) && (VGA\_Y <= en\_shot19\_y2)) && (en\_shot19) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot19\_y1 >= ship\_hull\_y1) && (en\_shot19\_y2 <= ship\_hull\_y2) && (en\_shot19\_x1 >= ship\_hull\_x1) && (en\_shot19\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot19 = 0;

end else if (en\_shot19\_y1 == 480) begin

en\_shot19 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot19\_y1 = en\_shot19\_y1 + 1;

en\_shot19\_y2 = en\_shot19\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot20\_x1) && (VGA\_X <= en\_shot20\_x2) && (VGA\_Y >= en\_shot20\_y1) && (VGA\_Y <= en\_shot20\_y2)) && (en\_shot20) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot20\_y1 >= ship\_hull\_y1) && (en\_shot20\_y2 <= ship\_hull\_y2) && (en\_shot20\_x1 >= ship\_hull\_x1) && (en\_shot20\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot20 = 0;

end else if (en\_shot20\_y1 == 480) begin

en\_shot20 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot20\_y1 = en\_shot20\_y1 + 1;

en\_shot20\_y2 = en\_shot20\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot21\_x1) && (VGA\_X <= en\_shot21\_x2) && (VGA\_Y >= en\_shot21\_y1) && (VGA\_Y <= en\_shot21\_y2)) && (en\_shot21) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot21\_y1 >= ship\_hull\_y1) && (en\_shot21\_y2 <= ship\_hull\_y2) && (en\_shot21\_x1 >= ship\_hull\_x1) && (en\_shot21\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot21 = 0;

end else if (en\_shot21\_y1 == 480) begin

en\_shot21 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot21\_y1 = en\_shot21\_y1 + 1;

en\_shot21\_y2 = en\_shot21\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot22\_x1) && (VGA\_X <= en\_shot22\_x2) && (VGA\_Y >= en\_shot22\_y1) && (VGA\_Y <= en\_shot22\_y2)) && (en\_shot22) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot22\_y1 >= ship\_hull\_y1) && (en\_shot22\_y2 <= ship\_hull\_y2) && (en\_shot22\_x1 >= ship\_hull\_x1) && (en\_shot22\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot22 = 0;

end else if (en\_shot22\_y1 == 480) begin

en\_shot22 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot22\_y1 = en\_shot22\_y1 + 1;

en\_shot22\_y2 = en\_shot22\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot23\_x1) && (VGA\_X <= en\_shot23\_x2) && (VGA\_Y >= en\_shot23\_y1) && (VGA\_Y <= en\_shot23\_y2)) && (en\_shot23) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot23\_y1 >= ship\_hull\_y1) && (en\_shot23\_y2 <= ship\_hull\_y2) && (en\_shot23\_x1 >= ship\_hull\_x1) && (en\_shot23\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot23 = 0;

end else if (en\_shot23\_y1 == 480) begin

en\_shot23 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot23\_y1 = en\_shot23\_y1 + 1;

en\_shot23\_y2 = en\_shot23\_y2 + 1;

shottest = 0;

end

end

end

if ( ((VGA\_X >= en\_shot24\_x1) && (VGA\_X <= en\_shot24\_x2) && (VGA\_Y >= en\_shot24\_y1) && (VGA\_Y <= en\_shot24\_y2)) && (en\_shot24) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

if ((en\_shot24\_y1 >= ship\_hull\_y1) && (en\_shot24\_y2 <= ship\_hull\_y2) && (en\_shot24\_x1 >= ship\_hull\_x1) && (en\_shot24\_x2 <= ship\_hull\_x2)) begin //

if (!exp\_live) begin

exp\_x1 = ship\_hull\_x1 + ((ship\_hull\_x2 - ship\_hull\_x1)/2);

exp\_y1 = ship\_hull\_y2;

exp\_live = 1'b1;

end

ship\_alive = 1'b0;

ship\_hull\_x1 = 0;

ship\_hull\_x2 = 0;

ship\_hull\_y1 = 0;

ship\_hull\_y2 = 0;

ship\_red = 10'h000;

ship\_green = 10'h000;

ship\_blue = 10'h000;

en\_shot24 = 0;

end else if (en\_shot24\_y1 == 480) begin

en\_shot24 = 1'b0;

end else begin

if(shottest >= 1250000/16) begin

en\_shot24\_y1 = en\_shot24\_y1 + 1;

en\_shot24\_y2 = en\_shot24\_y2 + 1;

shottest = 0;

end

end

end

//MAKES ENEMIES SHOOT. DO NOT DELETE SHOTTEST

shottest = shottest + 4;

//Draw Enemy Shot//

if ( en\_shot && (VGA\_X >= en\_shot\_x1) && (VGA\_X <= en\_shot\_x2) && (VGA\_Y <= en\_shot\_y2) && (VGA\_Y >= en\_shot\_y1) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

///Draw Enemy///

if ( (VGA\_X <= en\_hull\_x2) && (VGA\_X >= en\_hull\_x1) && (VGA\_Y <= en\_hull\_y2) && (VGA\_Y >= en\_hull\_y1) && (alive) ) begin

mRed = en\_red;

mGreen = en\_green;

mBlue = en\_blue;

end

///Draw Enemy 2///

if ( (VGA\_X <= en\_hull2\_x2) && (VGA\_X >= en\_hull2\_x1) && (VGA\_Y <= en\_hull2\_y2) && (VGA\_Y >= en\_hull2\_y1) && (alive2) ) begin

mRed = en2\_red;

mGreen = en2\_green;

mBlue = en2\_blue;

end

///Draw Enemy 3///

if ( (VGA\_X <= en\_hull3\_x2) && (VGA\_X >= en\_hull3\_x1) && (VGA\_Y <= en\_hull3\_y2) && (VGA\_Y >= en\_hull3\_y1) && (alive3) ) begin

mRed = en3\_red;

mGreen = en3\_green;

mBlue = en3\_blue;

end

///Draw Enemy 4///

if ( (VGA\_X <= en\_hull4\_x2) && (VGA\_X >= en\_hull4\_x1) && (VGA\_Y <= en\_hull4\_y2) && (VGA\_Y >= en\_hull4\_y1) && (alive4) ) begin

mRed = en4\_red;

mGreen = en4\_green;

mBlue = en4\_blue;

end

///Draw Enemy 5///

if ( (VGA\_X <= en\_hull5\_x2) && (VGA\_X >= en\_hull5\_x1) && (VGA\_Y <= en\_hull5\_y2) && (VGA\_Y >= en\_hull5\_y1) && (alive5) ) begin

mRed = en5\_red;

mGreen = en5\_green;

mBlue = en5\_blue;

end

///Draw Enemy 6///

if ( (VGA\_X <= en\_hull6\_x2) && (VGA\_X >= en\_hull6\_x1) && (VGA\_Y <= en\_hull6\_y2) && (VGA\_Y >= en\_hull6\_y1) && (alive6) ) begin

mRed = en6\_red;

mGreen = en6\_green;

mBlue = en6\_blue;

end

///Draw Enemy 7///

if ( (VGA\_X <= en\_hull7\_x2) && (VGA\_X >= en\_hull7\_x1) && (VGA\_Y <= en\_hull7\_y2) && (VGA\_Y >= en\_hull7\_y1) && (alive7) ) begin

mRed = en7\_red;

mGreen = en7\_green;

mBlue = en7\_blue;

end

///Draw Enemy 8///

if ( (VGA\_X <= en\_hull8\_x2) && (VGA\_X >= en\_hull8\_x1) && (VGA\_Y <= en\_hull8\_y2) && (VGA\_Y >= en\_hull8\_y1) && (alive8) ) begin

mRed = en8\_red;

mGreen = en8\_green;

mBlue = en8\_blue;

end

///Draw Enemy 9///

if ( (VGA\_X <= en\_hull9\_x2) && (VGA\_X >= en\_hull9\_x1) && (VGA\_Y <= en\_hull9\_y2) && (VGA\_Y >= en\_hull9\_y1) && (alive9) ) begin

mRed = en9\_red;

mGreen = en9\_green;

mBlue = en9\_blue;

end

///Draw Enemy 10///

if ( (VGA\_X <= en\_hull10\_x2) && (VGA\_X >= en\_hull10\_x1) && (VGA\_Y <= en\_hull10\_y2) && (VGA\_Y >= en\_hull10\_y1) && (alive10) ) begin

mRed = en10\_red;

mGreen = en10\_green;

mBlue = en10\_blue;

end

///Draw Enemy 11///

if ( (VGA\_X <= en\_hull11\_x2) && (VGA\_X >= en\_hull11\_x1) && (VGA\_Y <= en\_hull11\_y2) && (VGA\_Y >= en\_hull11\_y1) && (alive11) ) begin

mRed = en11\_red;

mGreen = en11\_green;

mBlue = en11\_blue;

end

///Draw Enemy 12///

if ( (VGA\_X <= en\_hull12\_x2) && (VGA\_X >= en\_hull12\_x1) && (VGA\_Y <= en\_hull12\_y2) && (VGA\_Y >= en\_hull12\_y1) && (alive12) ) begin

mRed = en12\_red;

mGreen = en12\_green;

mBlue = en12\_blue;

end

///Draw Enemy 13///

if ( (VGA\_X <= en\_hull13\_x2) && (VGA\_X >= en\_hull13\_x1) && (VGA\_Y <= en\_hull13\_y2) && (VGA\_Y >= en\_hull13\_y1) && (alive13) ) begin

mRed = en13\_red;

mGreen = en13\_green;

mBlue = en13\_blue;

end

///Draw Enemy 14///

if ( (VGA\_X <= en\_hull14\_x2) && (VGA\_X >= en\_hull14\_x1) && (VGA\_Y <= en\_hull14\_y2) && (VGA\_Y >= en\_hull14\_y1) && (alive14) ) begin

mRed = en14\_red;

mGreen = en14\_green;

mBlue = en14\_blue;

end

///Draw Enemy 15///

if ( (VGA\_X <= en\_hull15\_x2) && (VGA\_X >= en\_hull15\_x1) && (VGA\_Y <= en\_hull15\_y2) && (VGA\_Y >= en\_hull15\_y1) && (alive15) ) begin

mRed = en15\_red;

mGreen = en15\_green;

mBlue = en15\_blue;

end

///Draw Enemy 16///

if ( (VGA\_X <= en\_hull16\_x2) && (VGA\_X >= en\_hull16\_x1) && (VGA\_Y <= en\_hull16\_y2) && (VGA\_Y >= en\_hull16\_y1) && (alive16) ) begin

mRed = en16\_red;

mGreen = en16\_green;

mBlue = en16\_blue;

end

///Draw Enemy 17///

if ( (VGA\_X <= en\_hull17\_x2) && (VGA\_X >= en\_hull17\_x1) && (VGA\_Y <= en\_hull17\_y2) && (VGA\_Y >= en\_hull17\_y1) && (alive17) ) begin

mRed = en17\_red;

mGreen = en17\_green;

mBlue = en17\_blue;

end

///Draw Enemy 18///

if ( (VGA\_X <= en\_hull18\_x2) && (VGA\_X >= en\_hull18\_x1) && (VGA\_Y <= en\_hull18\_y2) && (VGA\_Y >= en\_hull18\_y1) && (alive18) ) begin

mRed = en18\_red;

mGreen = en18\_green;

mBlue = en18\_blue;

end

///Draw Enemy 19///

if ( (VGA\_X <= en\_hull19\_x2) && (VGA\_X >= en\_hull19\_x1) && (VGA\_Y <= en\_hull19\_y2) && (VGA\_Y >= en\_hull19\_y1) && (alive19) ) begin

mRed = en19\_red;

mGreen = en19\_green;

mBlue = en19\_blue;

end

///Draw Enemy 20///

if ( (VGA\_X <= en\_hull20\_x2) && (VGA\_X >= en\_hull20\_x1) && (VGA\_Y <= en\_hull20\_y2) && (VGA\_Y >= en\_hull20\_y1) && (alive20) ) begin

mRed = en20\_red;

mGreen = en20\_green;

mBlue = en20\_blue;

end

///Draw Enemy 21///

if ( (VGA\_X <= en\_hull21\_x2) && (VGA\_X >= en\_hull21\_x1) && (VGA\_Y <= en\_hull21\_y2) && (VGA\_Y >= en\_hull21\_y1) && (alive21) ) begin

mRed = en21\_red;

mGreen = en21\_green;

mBlue = en21\_blue;

end

///Draw Enemy 22///

if ( (VGA\_X <= en\_hull22\_x2) && (VGA\_X >= en\_hull22\_x1) && (VGA\_Y <= en\_hull22\_y2) && (VGA\_Y >= en\_hull22\_y1) && (alive22) ) begin

mRed = en22\_red;

mGreen = en22\_green;

mBlue = en22\_blue;

end

///Draw Enemy 23///

if ( (VGA\_X <= en\_hull23\_x2) && (VGA\_X >= en\_hull23\_x1) && (VGA\_Y <= en\_hull23\_y2) && (VGA\_Y >= en\_hull23\_y1) && (alive23) ) begin

mRed = en23\_red;

mGreen = en23\_green;

mBlue = en23\_blue;

end

///Draw Enemy 24///

if ( (VGA\_X <= en\_hull24\_x2) && (VGA\_X >= en\_hull24\_x1) && (VGA\_Y <= en\_hull24\_y2) && (VGA\_Y >= en\_hull24\_y1) && (alive24) ) begin

mRed = en24\_red;

mGreen = en24\_green;

mBlue = en24\_blue;

end

//Explosion//

if ( (exp\_live) && ((VGA\_X - (exp\_x1))\*\*2 + (VGA\_Y - exp\_y1)\*\*2 <= exp\_radius)) begin //remember to add the variables used here up above, there is going to need to be either a max radius bound or a max x and y bound

mRed = exp\_red;

mGreen = exp\_green;

mBlue = exp\_blue;

if (exp\_radius >= 1000) begin

exp\_radius = 0;

exp\_live = 1'b0;

end

end

///GAME OVER SCREEN///

//Check if ship is alive, if not set ship stuff to 0 and and bring up this screen

if ( !ship\_alive ) begin

//Y

if ( (VGA\_X >= 264) && (VGA\_X <= 280) && (VGA\_Y >= 0) && (VGA\_Y <= 120) ) begin //Top left post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 295) && (VGA\_X <= 311) && (VGA\_Y >= 0) && (VGA\_Y <= 120) ) begin //Top right post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 264) && (VGA\_X <= 311) && (VGA\_Y >= 90) && (VGA\_Y <= 120) ) begin //Crossbar

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 280) && (VGA\_X <= 296) && (VGA\_Y >= 90) && (VGA\_Y <= 240) ) begin //Middle Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

//O

if ( (VGA\_X >= 328) && (VGA\_X <= 375) && (VGA\_Y >= 0) && (VGA\_Y <= 24) ) begin //Top

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 328) && (VGA\_X <= 344) && (VGA\_Y >= 0) && (VGA\_Y <= 240) ) begin //Left Side

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 359) && (VGA\_X <= 375) && (VGA\_Y >= 0) && (VGA\_Y <= 240) ) begin //Right Side

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 328) && (VGA\_X <= 375) && (VGA\_Y >= 216) && (VGA\_Y <= 240) ) begin //Bottom

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

//U

if ( (VGA\_X >= 392) && (VGA\_X <= 408) && (VGA\_Y >= 0) && (VGA\_Y <= 240) ) begin //Left Side

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 423) && (VGA\_X <= 439) && (VGA\_Y >= 0) && (VGA\_Y <= 240) ) begin //Right Side

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 392) && (VGA\_X <= 423) && (VGA\_Y >= 216) && (VGA\_Y <= 240) ) begin //Bottom

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

//D

if ( (VGA\_X >= 256) && (VGA\_X <= 304) && (VGA\_Y >= 240) && (VGA\_Y <= 264) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 256) && (VGA\_X <= 304) && (VGA\_Y >= 456) && (VGA\_Y <= 480) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 256) && (VGA\_X <= 278) && (VGA\_Y >= 240) && (VGA\_Y <= 480) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 295) && (VGA\_X <= 319) && (VGA\_Y >= 264) && (VGA\_Y <= 456) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

//I

if ( (VGA\_X >= 320) && (VGA\_X <= 383) && (VGA\_Y >= 240) && (VGA\_Y <= 264) ) begin //Top

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 344) && (VGA\_X <= 358) && (VGA\_Y >= 240) && (VGA\_Y <= 480) ) begin //Bottom

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 320) && (VGA\_X <= 383) && (VGA\_Y >= 456) && (VGA\_Y <= 480) ) begin //Middle Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

//E

if ( (VGA\_X >= 384) && (VGA\_X <= 447) && (VGA\_Y >= 240) && (VGA\_Y <= 264) ) begin //Top Bar

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 384) && (VGA\_X <= 436) && (VGA\_Y >= 348) && (VGA\_Y <= 372) ) begin //Middle Bar

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 384) && (VGA\_X <= 447) && (VGA\_Y >= 456) && (VGA\_Y <= 480) ) begin //Bottom Bar

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 384) && (VGA\_X <= 408) && (VGA\_Y >= 240) && (VGA\_X <= 480) ) begin //Left Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

//D

if ( (VGA\_X >= 448) && (VGA\_X <= 472) && (VGA\_Y >= 240) && (VGA\_Y <= 480) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 472) && (VGA\_X <= 488) && (VGA\_Y >= 240) && (VGA\_Y <= 264) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 472) && (VGA\_X <= 488) && (VGA\_Y >= 456) && (VGA\_Y <= 480) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 487) && (VGA\_X <= 511) && (VGA\_Y >= 264) && (VGA\_Y <= 456) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

end

///YOU WIN SCREEN///

//Check if all enemies are dead then bring up this screen

if ( !alive && !alive2 && !alive3 && !alive4 && !alive5 && !alive6 && !alive7 && !alive8 &&

!alive9 && !alive10 && !alive11 && !alive12 && !alive13 && !alive14 && !alive15 && !alive16 &&

!alive17 && !alive18 && !alive19 && !alive20 && !alive21 && !alive22 && !alive23 && !alive24 ) begin

//Y

if ( (VGA\_X >= 264) && (VGA\_X <= 280) && (VGA\_Y >= 0) && (VGA\_Y <= 120) ) begin //Top left post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 295) && (VGA\_X <= 311) && (VGA\_Y >= 0) && (VGA\_Y <= 120) ) begin //Top right post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 264) && (VGA\_X <= 311) && (VGA\_Y >= 90) && (VGA\_Y <= 120) ) begin //Crossbar

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 280) && (VGA\_X <= 296) && (VGA\_Y >= 90) && (VGA\_Y <= 240) ) begin //Middle Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

//O

if ( (VGA\_X >= 328) && (VGA\_X <= 375) && (VGA\_Y >= 0) && (VGA\_Y <= 24) ) begin //Top

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 328) && (VGA\_X <= 344) && (VGA\_Y >= 0) && (VGA\_Y <= 240) ) begin //Left Side

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 359) && (VGA\_X <= 375) && (VGA\_Y >= 0) && (VGA\_Y <= 240) ) begin //Right Side

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 328) && (VGA\_X <= 375) && (VGA\_Y >= 216) && (VGA\_Y <= 240) ) begin //Bottom

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

//U

if ( (VGA\_X >= 392) && (VGA\_X <= 408) && (VGA\_Y >= 0) && (VGA\_Y <= 240) ) begin //Left Side

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 423) && (VGA\_X <= 439) && (VGA\_Y >= 0) && (VGA\_Y <= 240) ) begin //Right Side

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 392) && (VGA\_X <= 423) && (VGA\_Y >= 216) && (VGA\_Y <= 240) ) begin //Bottom

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

//W

if ( (VGA\_X >= 256) && (VGA\_X <= 264) && (VGA\_Y >= 240) && (VGA\_Y <= 480) ) begin //Left Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 311) && (VGA\_X <= 319) && (VGA\_Y >= 240) && (VGA\_Y <= 480) ) begin //Right Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 280) && (VGA\_X <= 296) && (VGA\_Y >= 360) && (VGA\_Y <= 480) ) begin //Middle Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 256) && (VGA\_X <= 319) && (VGA\_Y >= 456) && (VGA\_Y <= 480) ) begin //Bottom

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

//I

if ( (VGA\_X >= 328) && (VGA\_X <= 375) && (VGA\_Y >= 240) && (VGA\_Y <= 264) ) begin //Top

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 346) && (VGA\_X <= 355) && (VGA\_Y >= 240) && (VGA\_Y <= 480) ) begin //Bottom

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 328) && (VGA\_X <= 375) && (VGA\_Y >= 456) && (VGA\_Y <= 480) ) begin //Middle Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

//N

if ( (VGA\_X >= 384) && (VGA\_X <= 400) && (VGA\_Y >= 240) && (VGA\_Y <= 480) ) begin //Left Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 424) && (VGA\_X <= 436) && (VGA\_Y >= 420) && (VGA\_Y <= 480) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 416) && (VGA\_X <= 424) && (VGA\_Y >= 360) && (VGA\_Y <= 420) ) begin //Middle Left Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 408) && (VGA\_X <= 416) && (VGA\_Y >= 300) && (VGA\_Y <= 360) ) begin //Middle Right Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 400) && (VGA\_X <= 408) && (VGA\_Y >= 240) && (VGA\_Y <= 300) ) begin

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

if ( (VGA\_X >= 434) && (VGA\_X <= 447) && (VGA\_Y >= 240) && (VGA\_Y <= 480) ) begin //Right Post

mRed = 10'hFFF;

mGreen = 10'hFFF;

mBlue = 10'hFFF;

end

end

end

Lab4 music(.signalout(musicOut), .clk(clk));

VGA\_Ctrl u9 ( // Host Side

.iRed(mRed),

.iGreen(mGreen),

.iBlue(mBlue),

.oCurrent\_X(VGA\_X),

.oCurrent\_Y(VGA\_Y),

.oRequest(VGA\_Read),

// VGA Side

.oVGA\_R(vga\_r10 ),

.oVGA\_G(vga\_g10 ),

.oVGA\_B(vga\_b10 ),

.oVGA\_HS(VGA\_HS),

.oVGA\_VS(VGA\_VS),

.oVGA\_SYNC(VGA\_SYNC\_N),

.oVGA\_BLANK(VGA\_BLANK\_N),

.oVGA\_CLOCK(VGA\_CLK),

// Control Signal

.iCLK(clk25),

.iRST\_N(1) );

endmodule

module VGA\_Ctrl ( // Host Side

iRed,

iGreen,

iBlue,

oCurrent\_X,

oCurrent\_Y,

oAddress,

oRequest,

// VGA Side

oVGA\_R,

oVGA\_G,

oVGA\_B,

oVGA\_HS,

oVGA\_VS,

oVGA\_SYNC,

oVGA\_BLANK,

oVGA\_CLOCK,

// Control Signal

iCLK,

iRST\_N );

// Host Side

input [9:0] iRed;

input [9:0] iGreen;

input [9:0] iBlue;

output [21:0] oAddress;

output [10:0] oCurrent\_X;

output [10:0] oCurrent\_Y;

output oRequest;

// VGA Side

output [9:0] oVGA\_R;

output [9:0] oVGA\_G;

output [9:0] oVGA\_B;

output reg oVGA\_HS;

output reg oVGA\_VS;

output oVGA\_SYNC;

output oVGA\_BLANK;

output oVGA\_CLOCK;

// Control Signal

input iCLK;

input iRST\_N;

// Internal Registers

reg [10:0] H\_Cont;

reg [10:0] V\_Cont;

////////////////////////////////////////////////////////////

// Horizontal Parameter

parameter H\_FRONT = 16;

parameter H\_SYNC = 96;

parameter H\_BACK = 48;

parameter H\_ACT = 640;

parameter H\_BLANK = H\_FRONT+H\_SYNC+H\_BACK;

parameter H\_TOTAL = H\_FRONT+H\_SYNC+H\_BACK+H\_ACT;

////////////////////////////////////////////////////////////

// Vertical Parameter

parameter V\_FRONT = 11;

parameter V\_SYNC = 2;

parameter V\_BACK = 31;

parameter V\_ACT = 480;

parameter V\_BLANK = V\_FRONT+V\_SYNC+V\_BACK;

parameter V\_TOTAL = V\_FRONT+V\_SYNC+V\_BACK+V\_ACT;

////////////////////////////////////////////////////////////

assign oVGA\_SYNC = 1'b1; // This pin is unused.

assign oVGA\_BLANK = ~((H\_Cont<H\_BLANK)||(V\_Cont<V\_BLANK));

assign oVGA\_CLOCK = ~iCLK;

assign oVGA\_R = iRed;

assign oVGA\_G = iGreen;

assign oVGA\_B = iBlue;

assign oAddress = oCurrent\_Y\*H\_ACT+oCurrent\_X;

assign oRequest = ((H\_Cont>=H\_BLANK && H\_Cont<H\_TOTAL) &&

(V\_Cont>=V\_BLANK && V\_Cont<V\_TOTAL));

assign oCurrent\_X = (H\_Cont>=H\_BLANK) ? H\_Cont-H\_BLANK : 11'h0 ;

assign oCurrent\_Y = (V\_Cont>=V\_BLANK) ? V\_Cont-V\_BLANK : 11'h0 ;

// Horizontal Generator: Refer to the pixel clock

always@(posedge iCLK or negedge iRST\_N)

begin

if(!iRST\_N)

begin

H\_Cont <= 0;

oVGA\_HS <= 1;

end

else

begin

if(H\_Cont<H\_TOTAL)

H\_Cont <= H\_Cont+1'b1;

else

H\_Cont <= 0;

// Horizontal Sync

if(H\_Cont==H\_FRONT-1) // Front porch end

oVGA\_HS <= 1'b0;

if(H\_Cont==H\_FRONT+H\_SYNC-1) // Sync pulse end

oVGA\_HS <= 1'b1;

end

end

// Vertical Generator: Refer to the horizontal sync

always@(posedge oVGA\_HS or negedge iRST\_N)

begin

if(!iRST\_N)

begin

V\_Cont <= 0;

oVGA\_VS <= 1;

end

else

begin

if(V\_Cont<V\_TOTAL)

V\_Cont <= V\_Cont+1'b1;

else

V\_Cont <= 0;

// Vertical Sync

if(V\_Cont==V\_FRONT-1) // Front porch end

oVGA\_VS <= 1'b0;

if(V\_Cont==V\_FRONT+V\_SYNC-1) // Sync pulse end

oVGA\_VS <= 1'b1;

end

end

endmodule

module Lab4(signalout, clk);

input clk;

//input [1:0] sw;

output signalout;

reg signalout;

reg [31:0] mycounter1, mycounter2, myonesecond, countlow, counthigh;

reg [31:0] progress, lastnote;

wire clk2;

assign clk2 = clk/2;

initial begin

mycounter1 = 0;

mycounter2 = 0;

signalout = 0;

myonesecond = 50000000/8;

progress = 4'b0001;

end

always @(posedge clk) begin

case(progress)

default: begin

end

4'b0000: begin

mycounter1 = mycounter1 + 1'b1;

mycounter2 = mycounter2 + 1'b1;

if(mycounter1 < countlow) begin

signalout = 0;

end

if((mycounter1 >= countlow) && (mycounter1 < counthigh)) begin

signalout = 1;

end

if(mycounter1 >= counthigh) begin

signalout = 0;

mycounter1 = 0;

if(mycounter2 >= myonesecond) begin

progress = lastnote;

mycounter2 = 0;

end

end

end

4'b0001: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 2;

end

2: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 3;

end

3: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 4;

end

4: begin//STALL

countlow = 0;

counthigh = 0;

progress = 0;

lastnote = 5;

end

5: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 6;

end

6: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 7;

end

7: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 8;

end

8: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 9;

end

9: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 10;

end

10: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 11;

end

11: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 12;

end

12: begin//STALL

countlow = 0;

counthigh = 0;

progress = 0;

lastnote = 13;

end

13: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 14;

end

14: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 15;

end

15: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 16;

end

16: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 17;

end

17: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 18;

end

18: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 19;

end

19: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 20;

end

20: begin//STALL

countlow = 0;

counthigh = 0;

progress = 0;

lastnote = 21;

end

21: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 22;

end

22: begin //F6

countlow = 17896;

counthigh = 35793;

progress = 0;

lastnote = 23;

end

23: begin//STALL

countlow = 0;

counthigh = 0;

progress = 0;

lastnote = 24;

end

24: begin //F6

countlow = 17896;

counthigh = 35793;

progress = 0;

lastnote = 25;

end

25: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 26;

end

26: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 27;

end

27: begin //C6

countlow = 23889;

counthigh = 47778;

progress = 0;

lastnote = 28;

end

28: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 29;

end

29: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 30;

end

30: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 31;

end

31: begin//STALL

countlow = 0;

counthigh = 0;

progress = 0;

lastnote = 32;

end

32: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 33;

end

33: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 34;

end

34: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 35;

end

35: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 40;

end

40: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 41;

end

41: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 42;

end

42: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 43;

end

43: begin//STALL

countlow = 0;

counthigh = 0;

progress = 0;

lastnote = 44;

end

44: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 45;

end

45: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 46;

end

46: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 47;

end

47: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 48;

end

48: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 49;

end

49: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 51;

end

51: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 52;

end

52: begin//STALL

countlow = 0;

counthigh = 0;

progress = 0;

lastnote = 53;

end

53: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 54;

end

54: begin //F6

countlow = 17896;

counthigh = 35793;

progress = 0;

lastnote = 55;

end

55: begin//STALL

countlow = 0;

counthigh = 0;

progress = 0;

lastnote = 56;

end

56: begin //F6

countlow = 17896;

counthigh = 35793;

progress = 0;

lastnote = 57;

end

57: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 58;

end

58: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 59;

end

59: begin //C6

countlow = 23889;

counthigh = 47778;

progress = 0;

lastnote = 60;

end

60: begin //C6

countlow = 23889;

counthigh = 47778;

progress = 0;

lastnote = 61;

end

61: begin //C6

countlow = 23889;

counthigh = 47778;

progress = 0;

lastnote = 62;

end

62: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 63;

end

63: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 64;

end

64: begin //F6

countlow = 17896;

counthigh = 35793;

progress = 0;

lastnote = 65;

end

65: begin //G5

countlow = 31888;

counthigh = 63776;

progress = 0;

lastnote = 66;

end

66: begin //G5

countlow = 31888;

counthigh = 63776;

progress = 0;

lastnote = 67;

end

67: begin //F6

countlow = 17896;

counthigh = 35793;

progress = 0;

lastnote = 68;

end

68: begin //B5

countlow = 25309;

counthigh = 50619;

progress = 0;

lastnote = 69;

end

69: begin //B5

countlow = 25309;

counthigh = 50619;

progress = 0;

lastnote = 70;

end

70: begin //A6

countlow = 28409;

counthigh = 56818;

progress = 0;

lastnote = 71;

end

71: begin //G5

countlow = 31888;

counthigh = 63776;

progress = 0;

lastnote = 72;

end

72: begin //A6

countlow = 28409;

counthigh = 56818;

progress = 0;

lastnote = 73;

end

73: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 74;

end

74: begin //E5

countlow = 37921;

counthigh = 75843;

progress = 0;

lastnote = 75;

end

75: begin //F6

countlow = 17896;

counthigh = 35793;

progress = 0;

lastnote = 76;

end

76: begin //C6

countlow = 23889;

counthigh = 47778;

progress = 0;

lastnote = 77;

end

77: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 78;

end

78: begin //F6

countlow = 17896;

counthigh = 35793;

progress = 0;

lastnote = 79;

end

79: begin //A6

countlow = 28409;

counthigh = 56818;

progress = 0;

lastnote = 80;

end

80: begin //B6

countlow = 25309;

counthigh = 50619;

progress = 0;

lastnote =81;

end

81: begin //C6

countlow = 23889;

counthigh = 47778;

progress = 0;

lastnote = 82;

end

82: begin //G5

countlow = 31888;

counthigh = 63776;

progress = 0;

lastnote = 83;

end

83: begin//STALL

countlow = 0;

counthigh = 0;

progress = 0;

lastnote = 84;

end

84: begin //C5

countlow = 47778;

counthigh = 95556;

progress = 0;

lastnote = 85;

end

85: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 86;

end

86: begin //F6

countlow = 17896;

counthigh = 35793;

progress = 0;

lastnote = 87;

end

87: begin //A6

countlow = 28409;

counthigh = 56818;

progress = 0;

lastnote = 88;

end

88: begin //G5

countlow = 31888;

counthigh = 63776;

progress = 0;

lastnote = 89;

end

89: begin//STALL

countlow = 0;

counthigh = 0;

progress = 0;

lastnote = 90;

end

90: begin //G5

countlow = 31888;

counthigh = 63776;

progress = 0;

lastnote = 91;

end

91: begin //F6

countlow = 17896;

counthigh = 35793;

progress = 0;

lastnote = 92;

end

92: begin //D5

countlow = 42565;

counthigh = 85131;

progress = 0;

lastnote = 1;

end

endcase

end

endmodule

1. Peer Evaluation

Peer Evaluation Form

Your First Name: Sean

Your Last Name: Copp

ELEC 3200: Section 1

Lab Section: Thursday

Fill out the table below using the following scale:

1-4 (1=strongly disagree; 2=disagree; 3=agree; 4=strongly agree)

|  |  |  |  |
| --- | --- | --- | --- |
| Evaluation Criteria\Name | Sean Copp | Mike Hickey | Irrelevant |
|  |  |  |  |
| Arrives on time | 4 | 4 |  |
| Attends meetings | 4 | 4 |  |
| Provides good quality of work | 4 | 4 |  |
| Contributes to the overall success of project/lab | 4 | 4 |  |
| Total | 16 | 16 |  |

% Grade Distribution:

Sean Copp: \_\_\_\_60%\_\_\_

Mike Hickey: \_\_\_40%\_\_\_

Total: \_\_\_\_100%\_\_\_\_\_

Peer Evaluation Form

Your First Name: Michael

Your Last Name: Hickey

ELEC 3200: Section 1

Lab Section: Thursday

Fill out the table below using the following scale:

1-4 (1=strongly disagree; 2=disagree; 3=agree; 4=strongly agree)

|  |  |  |  |
| --- | --- | --- | --- |
| Evaluation Criteria\Name | Sean Copp | Mike Hickey | Irrelevant |
|  |  |  |  |
| Arrives on time | 4 | 4 |  |
| Attends meetings | 4 | 4 |  |
| Provides good quality of work | 4 | 4 |  |
| Contributes to the overall success of project/lab | 4 | 4 |  |
| Total | 16 | 16 |  |

% Grade Distribution:

Sean Copp: \_\_\_\_60%\_\_\_

Mike Hickey: \_\_\_40%\_\_\_

Total: \_\_\_\_100%\_\_\_\_\_