Checkoff List

Digital Stereoscope

Joshua Jen Monzon
Tony Ng
Hao (Steven) Zhou

Modules

Tony

1. Triangle Corner and Color Memory
2. Perspective Projector
   a. Arctangent submodule
      Function: Calculates arctangent of a vector, returning a value between zero and 2pi. First three bits represents integer part and last 13 bits represents the decimals as a binary fraction.
      Test: Demonstrate binary fraction output on LED for pi/4, pi/2, and 3pi/2
   b. Projection module
      Test: Show projection of the three corners of a triangle on screen and change to different triangles and orientations using switches on labkit
3. User Position Controller
   Test: Show a point moving around on the screen responding to user input
4. Top-level connections to the other modules and labkit

Steve

1. Color LUT Module
   Function: given an eight bit input for color, it displays a 24 bit rgb value
   Test: use the 8 switches as controls for input, and see if the screen display of the 24 bit rgb value is correct.
2. Pixel Extractor
   Function: Given the corner location of the triangles for each eye, this module will loop through appropriate pixels which are held for two clock cycles at a time, and output the correct distance, and tocolor signals along with the pixel location.
   Test: Modelsim test bench
   a. TriangleArea submodule
      Function: Given the x, y, z coordinates of the three corners, output the area.
      Test: Modelsim test bench
3. Shading Module
   Function: Given the pixel location, pixel depth, and color, it will output the correct pixel address in memory, and alter the color according to the depth of that pixel
   Test: Modelsim test bench

Overall Test:

To test all of my modules as a whole, I will encompass all three of my modules in one module, and will wire the inputs for corner locations of the three corners to fixed values. Then, using the logic analyzer, I will test to see if the output for pixel address and pixel data is correct.

Josh

1. Buffer Interface Module
   Function: Comprised of 3 submodules, depth comparator, depth buffer initiator and the zbt interface module. It has an internal BRAM used to compare depth values synchronously. It interfaces with the onboard SRAM (ZBTs) to store the memory extensive color pixel values which are refreshed in the screen.
   Test: Demonstrate the whole module by using a rom graphic file to manually input address and pixel values.
   a. DepthComparator submodule
      Function: Accepts pixel values, depth values, and pixel addresses and compares incoming depth values with previously stored values and overwrites the old values if the incoming depth values are less deep than the old pixel's depth. Sends the **pixel value** and its corresponding **address** to the ZBT interface with an appropriate **write enable** signal depending on whether we have to overwrite the old pixel or not.
      Test: Testbench is sufficient
   b. DepthBufferIniator submodule
      Function: During reset or when switching buffers, this initiates default values for the buffer which is currently being written to. Refreshes the internal BRAM to hold maximum depth values and passes default black values to be written to the ZBTs.
      Test: Testbench
   c. ZBT interface submodule
      Function: Final processing of pixel data and pixel addresses and write enable signals before they are passed to the ZBTs. It routes these data to the appropriate ZBT inputs based on which buffer is being refreshed.
      Test: Testbench

2. Displayer Module
   Function: It accepts control signals from the XVGA (hcount, vcount, and blank) and outputs the **corresponding address** which maps to the hcount, vcount and blank signals to the ZBT. It accepts incoming data which correspond to the addresses it sent 2 clock cycles ago and passes this **pixel** data to the VGA display outputs in the labkit. It also outputs a **done display** signal every time it displays one screenful of data.
   Test: Demonstrate the whole module by displaying the stored rom graphic file in the ZBT in the screen.
3. BufferSelector submodule
   Function: It informs the whole system which buffer should be written. At reset, Buffer 0 should be written and switches every time a buffer is done refreshing AND the screen finish displaying.
   Test: Testbench should be enough

Expected Functionalities of Final Project

1. Make a box or any single object appear on two screens, which represent the image of the box seen from two perspectives.

2. Moving user and changing his/her viewing angle will display the object correctly. For example, moving farther shrinks the object.
3. Make a wall appear in our world with the box.
4. Moving user and changing his/her viewing angle will display the object and wall correctly.
5. Color of the object and wall gets brighter as user approaches the object. Proper shading behavior observed.
6. Make multiple objects and wall appear on the two screens. Proper overlapping behavior should be observed.
7. Moving user will display the objects correctly, maintaining proper overlapping behavior. Proper shading changes occur.

Possible Extensions

1. Make more complex objects with more triangles.