The goal of this lab is to modify your VGA controller from the last lab so that it can read pixel information from memory. The memory layout should be pixel-based true-color. Since you can only display eight colors, you need three bits to represent each pixel. The SRAM interface is 32-bits wide, meaning you could store up to 10 pixels per word. However, for simplicity, just store 8 pixels per word – one pixel for every four bits or nibble. The layout of the pixels within the word looks as follows:

```
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
```

Since your display is 640x480, you will need 38400 words to store all the pixels. You can store the pixels in the order you will read them. Thus row 0, columns 0-7 will be at address 0x00000000, row 0, columns 8-15 will be at address 0x00000001 ... row 1, columns 0-7 will be at address 0x00000050, and so on. Each row starts at a multiple of 0x00000050. The last row starts at 0x000095B0, and ends at 0x000095FF. Once you reach the end, your VGA controller can start reading pixels at 0 again.

The VGA controller should conform to the bus280 logic that you did for lab 4. Your memory writer will be very similar to what was done in lab4. The only difference is that the data you write to memory will be the pixels. In this case, you want to display a pixel pattern so that the colors alternate every 8 pixels.

Some things to be aware of:

You may need to insert wait states in the SRAM interface.

Your VGA controller will to read from memory ahead of when you actually display the data. There are two options. The first option is to read just one word or 8 pixels ahead. This requires doing the reading in the same process that you are displaying data. The second option is to read a whole line of data, i.e. 640 pixels or 80 words, during the hsync and front and back porches. This is much simpler, since the memory reading can be done in a separate process. However, it does require extra memory storage – 80 words vs. 1 word.