ECEN 3763 – 1080p Video Controller

Week 6

Due Friday, February 25th, 11:59pm - 35 points

Summary:

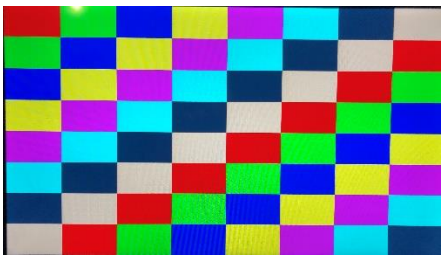
For the Week6 lab, you will convert your VGA 720p60 Controller design to support 1080p60, and the output will be a displayed checkboard pattern. You have one week to complete this lab.

The VGA details provided in the Week 4/5 lab document are not repeated here, so refer to the previous lab document as needed.

Lab Instructions:

1. Create a new project. Your Week 4/5 lab is an excellent starting point.

2. Modify the Week 4/5 design to support 1080p60 video. The specific design parameters are provided in this document. The displayed image will consist of an 8 x 8 checkboard pattern similar to the image below:



The selection of colors or gray scale is your choice.

3. Write a simple testbench that simulates one frame of video, and include all top level signals in your simulation screen capture. You may add additional signals to your simulation as desired. It is not necessary to provide multiple screen captures.

4. Your design for Week 6 only needs to support 1080p60 resolution. There is no requirement that your video controller switches between 720p and 1080p resolutions.

Submission and Grading:

Your submission will be a single .zip (or .7z or .tar) file uploaded to Canvas for Project/Lab Week 5. The submitted file should contain the following:

a) A Quartus archive of your project (.qar file). Test your .qar file prior to submission.

b) At least one screen capture of your Modelsim simulation waveforms, in focus and at a proper zoom level to convey whatever aspects of your design you are attempting to show. This file can be JPEG, PNG, PDF, or some common image format.

c) You may choose to include a brief text file describing issues with your final design (if any), an explanation of your simulation (if needed), and anything else you want me to know about your submission.

This lab will be graded as follows:

Quartus project that successfully displays the stripe pattern:	30 points
Modelsim simulation:	5 points

General Design Guidelines

1. Follow the same guidelines as the previous lab.

2. **Verify the bit widths of your horizontal and vertical counters are sufficient for the 1080p60 design.**

3. Terasic System Builder generates a file with a .sdc extension. Verify that this file is included as a source in your Quartus project. You do not need to make any edits to the .sdc file.

4. Since the pixel clock is 148.5 Mhz, it is important that Quartus has timing constraints for this design. Since the PLL file contains information about the input and output clock frequencies, you only need to include the .sdc file in your project.

5. You can modify your previous pixel generator SystemVerilog to generate the RGB values for the checkboard pattern, or you can use a memory block to store the RGB data. Refer to Lecture 11 regarding creation of the memory block and how to create the .mif file. You must have a .mif file created before generating the ROM memory block from the IP catalog. The .mif file can be empty, but must exist.

   It is possible to use the In-System Memory Content Editor (ISMCE) to create your checkerboard pattern, and then write out the resulting .mif file. The formatting of the .mif file is not very pretty, but this does work. There will be a homework assignment later in the semester discussing use of the ISMCE, work with the instructor or SA if you want to try this approach. Note you do not have to use the ISMCE to create a memory based pixel generator, you can create the .mif file by hand.

6. If you do use a ROM to store your RGB data, your pixel generator must generate the correct address pattern.

1080p60 Parameters:

```
1080p 1920 x 1080 60Hz, 148.5 Mhz pixel clock
2200 pixels per line, 1125 lines per frame
h_synch and v_synch are active high

parameter h_pixels = 1920;  // visible pixels per line
parameter h_fp = 88;        // horizontal front porch width
parameter h_bp = 148;       // horizontal back porch width
parameter h_synch = 44;     // horizontal synch pulse width
parameter v_lines = 1080;   // visible lines per frame
parameter v_fp = 4;         // vertical front porch width
parameter v_bp = 36;        // vertical back porch width
parameter v_synch = 5;      // vertical synch pulse width
```