LAB Assignment #2 for ECE 443

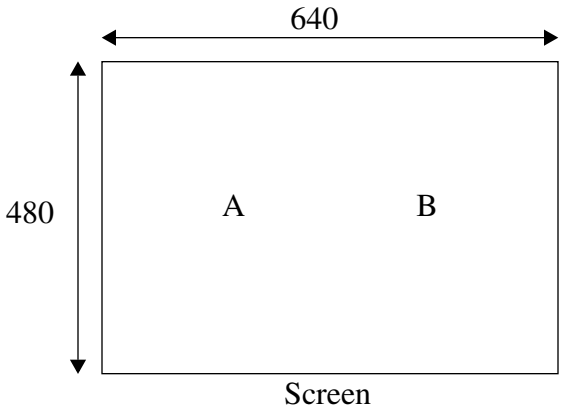
Assigned: Mon., Oct. 14, 2015
Due: Wed., Oct. 19, 2015

Description: Copy and Demo Video Driver Code from lecture notes, and then implement a fixed graphics as descibed below.

1) Copy the code from 'VGA documentation and VHDL code' pdf on slides 9-17, create a project USING THE CONSTRAINTS file on my website for lab2, program the FPGA, connect a VGA cable between the FPGA and video monitor in the lab and demonstrate that the screen color changes based on the 3 switches.

NOTE: You do NOT need to add a block diagram, i.e., you will NOT use the zynq processor or GPIO for this lab. Once the project is created, add three files (add sources), the VGA_const.xdc and VGA_test.vhd files from my lab2 files. The third file should be created by you from a copy of the code given on slides 9-17 (be sure to delete the 'comp_sync' entity and simple signal assignment statement from this code as discussed in class).

2) In reference to slide 21, you need to create a *tile-mapped* display. To keep this simple for now, you should only create two tiles of 8x8 pixels (64 pixels) for the letters 'A' and 'B' (later, you'll want to implement the other letters and numbers for your game). In reference to the object-mapped description on slides 22 on forward, you need to display these two letters at the 'approximate' positions as shown on the screen below:



You should keep the switches connected as descibed in 1) so that you can change the background color. You can choose 2 of the 8 colors used for the background display above as the background and foreground colors for your two tiles (or you can create a new color by controlling the individual bits of the 12-bit rgb signals on the Zedboards as we will discuss in class).

The VHDL construct for creating a 2D 'bitmap' is given on slide 32 as follows:

```
type rom_type is array(0 to 7) of std_logic_vector(0 to
7);
constant A_CHAR: rom_type:= (
"00111100",
```

```
        "01111110",
        "11111111", "11111111", "11111111", "11111111",
        "01111110",
        "00111100");
```

Change this bitmap to display the letter ‘A’. Create the B_CHAR in a similar fashion using a second ‘constant’ definition.

Clues on how to do the pixel addressing for these character bitmaps is given on slide 33 and 34. The slides show how a ‘ball’ bitmap is displayed and how data is read out of the ‘rom_type’ defined above.

The 8x8 bitmaps given above only indicate on or off (0 or 1) and therefore, these bitmaps can only be used to display black and while colors. To achieve this, you’ll need to read out each bit, one at a time, as determined by the current values of ‘pixel_x’ and ‘pixel_y’ and then assign the ‘0’ or ‘1’ from the bitmap to ALL TWELVE rgb signals (similar to the fanout example from VGA_test where each rgb_reg(0) signal drives all 4 vga_b1/b2/b3/b4 signals).

Laboratory Report Requirements:

Grading:
The grading from this lab will be based entirely on your in-class demo. Bonus points will be given to any implementation feature that goes above and beyond the requirements. Please print out and turn in a copy of your VHDL code.