University of Essex

School of Computer Science and Electronic Engineering

CE264 Digital Systems Design – Autumn Term 2019

Laboratory Design Work – Stage 2

This document outlines the second stage of design work that you are required to do as the main coursework component of CE264. In this part, you will design a sub-system to output raster drive signals on the VGA port (to drive a VGA computer monitor).

VGA driver

You will need to understand how a VGA monitor works (in terms of its input signals and the concept of the video 'raster'). The VGA interface is analogue, but it uses timing signals that can be generated digitally, and the Basys3 board has digital outputs from the FPGA that allow you to output simple graphics to the VGA port, and thence to a display. There is a description of this in the Basys3 Reference Manual, section 7, and you may also find other explanations in books or online.

Wikipedia has an article on VGA, covering the IBM graphics architecture of the same name (https://en.wikipedia.org/wiki/Video_Graphics_Array). This article has some relevant parts, but some of it relates to computer graphics hardware, which is somewhat different to what we need to do to drive the Basys3 board.

To make your reading easier, here is a simple account to start you off. VGA displays were originally based on cathode-ray tubes (not the LCD flat panels that we use now). Although modern displays don't use the same technology, the VGA signals still have the same format. (Recent modern displays also have digital or DVI video inputs too, and that is what most modern computers drive.) CRTs had an internal electron beam that scanned the inside face of a vacuum tube (the screen) in a specific pattern to display visible dots of light (by striking a phosphor coating on the inside of the glass). The amplitude of the electron beam was modulated as it scanned to control the brightness of individual dots (called **pixels**). Even when using LCD displays, you have to imagine that electron beam scanning, because the signals that drive the VGA display are analogue signals based on that electron beam scan. (This is a classic case where modern technology works in a way that was originally developed for a different technology, but it has to maintain compatibility at the interface.)

The scan pattern (known as a *raster*) starts at the top left of the display and scans lines from left to right, working down the display. At the end of each horizontal line, the electron beam was switched off, and scanned back across to the left-hand side (faster than the left to right scan with the beam on). At the end of the last horizontal line, the beam was switched off, and scanned back to the top left (again fast). Two **signals** called vertical sync (VSYNC) and horizontal sync (HSYNC) control the raster. VSYNC is a low-going pulse occurring typically 50-60 times per second. It indicates the start of a raster, and **commands** the beam to move to the top left of the display. HSYNC is also a low-going pulse, and it indicates the start of each scan line. The sync pulses are derived from a PIXEL CLOCK which defines the timing of one pixel on the display – find out what this means by reading the documentation. Also find out what the **blanking** signal is for. You will need to use it.

The basic principle of generating graphics in hardware is to count lines down the screen, and pixels across, using counters. This can give a position on the screen, to generate a graphic, we need some logic or a memory to output the actual pixels which make up the graphic. The pixel values are analogue voltages, but the Basys3 board provides a simple digital interface to create these analogue voltage levels. Read about it in section 7 of the Basys3 Reference Manual.

Your design should use a 640 × 480 raster, because this requires a 25 MHz pixel clock, which is easily generated from the Basys3 100 MHz clock by dividing by four. (Division by powers of two or powers of ten is easy with counters.)

Here are some questions to prompt you on what you need to know:

1. What timing is required for each of VYSNC and HSYNC? You need detail here in terms of time and pixel clock cycles.
2. How does the Basys3 generate analogue voltages for the red, green and blue outputs from digital values?
3. How many bits of red, green and blue data are output from the FPGA?
4. What is **blanking**?

(Students on CE264 have often been surprised to find that their exam paper covers topics like these, and are unable to construct good answers. Therefore: don't make the mistake of ignoring these questions.)

There are VHDL designs for the circuitry needed to drive the VGA monitor, but there seems to be no simple way to create a block in Multisim and use existing VHDL to provide the implementation. However, it is not too difficult to create the circuitry required – all we need is a suitable clock and some counters.

Your VGA controller design (a **hierarchical block**, of course) should have the following inputs and outputs:

Inputs:

1. Pixel clock. This will be a 25 MHz clock, divided by four from the Basys3 clock, OUTSIDE your VGA design. (Use the clock divider that you created in Stage 1.)

Outputs:

1. VSYNC – vertical synchronisation signal.
2. HSYNC – horizontal synchronisation signal.
3. BLANK – this should be high when the video signal is to be blanked.
4. VCOUNT – 9 bits of unsigned binary representing a row number between 0 and 479.
5. HCOUNT – 10 bits of unsigned binary representing a column number between 0 and 639.

The counters that produce VCOUNT and HCOUNT are not the same counters that will be used to generate HSYNC and VSYNC. Keep clear in your mind the distinction between **generating the timing** for HSYNC and VSYNC and **counting lines down and pixels across** the display to create graphics coordinates. These things are at two different **abstraction levels**.

Why do we need only 9 bits for VCOUNT, but 10 for HCOUNT?

Hierarchy

Don't forget that you are working on a sub-system (a hierarchical block). Do not fall into the trap of designing your circuit in such a way that you can't easily use it as part of a larger system in Stage 3. If necessary go back and re-read the material about hierarchy, as it may make more sense to you now after doing some more significant design.

Think carefully about the **internal** hierarchy within your VGA driver. VSYNC and HSYNC can be generated by **two copies of the same circuit**, if you design it correctly. Each copy will need a different clock (HSYNC requires the pixel clock, VSYNC requires HSYNC as its clock), and each copy will need different input parameters to control the number of clock cycles that are counted, **but the structure of the circuit can be exactly the same**, and therefore represented using only one hierarchical block.

Testing

It is possible to test the VSYNC and HSYNC signals alone by driving a monitor with them, and outputting black (digital zero) on all the RGB pins. If the SYNC signals are correct, a display will show a blank screen and the menu on the monitor can be used to check that the raster parameters are correct. It is also easy to check the timing of the SYNC signals on an oscilloscope, and you should do this, **carefully**, to be sure that all the timings are correct. (The timings are given in the Basys3 Reference Manual, Figure 14, page 19.)

Once you have VYSNC and HSYNC correct, you can drive the red/green/blue outputs from the switches on the board, and thus control the colour of the display (but you will need to use the blanking signal to cut off the RGB signals when the display is supposed to be blanked, otherwise the monitor may fail to display any colours at all).

After that, you can create VCOUNT and HCOUNT and use them to generate simple patterns on the display that will really confirm that your circuit is working properly. Example: use the most significant bit of VCOUNT to drive an RGB signal and display two colours, splitting the screen horizontally at some point (exactly where isn't important).

Dr Steve Sangwine
CE264 Module Supervisor
Version 4
1 October 2019