Manual - Image Visualisation with Spartan3 (Starter Board)

Authors: *Alessandro De Palma (alessandrodepalma93@gmail.com)*

Francesco Pierri (frapierri@hotmail.it)

With supervision of:

Prof. P. Montuschi, Prof. L. Sterpone.

Politecnico di Torino a.a. 2013/2014

Tecnical Characteristics:

Input.

- 115200 Baud Rate, easily editable by the user (one has to change some constants both on the C program and in the block "*uart2BusTop*").
- Pixel sending frequency: (Baud Rate)/10 Hz.
- Maximum storable image has a product of dimensions equal to 256*256.

Output.

- Screen refresh rate: 60Hz.
- Screen resolution: 640*480.
- Number of colours which can be visualized: 8 (due to hardware limitations).

TUTORIAL

Run the "*ImageTransferOpenCores.exe*" and follow the on-screen instructions:

1. Write the image folder path when requested (just drag the image file and release it on the console).
2. Write down Height and Length accordingly.
3. Check the COM port in the "Control Panel"->"Device Manager" and write

its number when requested. Set in the same menu, before continuing, "115200" (or the chosen one) as "Baud Rate", "No Parity", No "Flow Control" and "1 Stop Bit". In "Advanced options" disable "FIFO buffer".

4. Move up "switch1" (all switches should be at zero in the default configuration) and send dimensions to the board.

5. Set "switch1" back to the Low State and rise "switch0" up. At this point press "Enter" to send the image.

6. Set "switch0" to zero again and enjoy the view.

7. Set "switch2" up to visualise the negative of the image. Act on "switch6" and "switch7" to change visualisation position (read the description of "*ImgPositionNew*" for further info).

Please notice that you can visualize new images without resetting the board: just repeat the procedure above!

Components description:

C program.

The C program called "*ImageTransferOpenCores*" works on a *.ppm file, ASCII encoding, chosen by the user: this has to be either produced by "*GIMP*" (an open source Image Editing program) or formatted in the same way. The correct format is:

```
4 useless lines (actually comments by the program)
pixel(1R)
pixel(1G)
pixel(1B)
pixel(2R)
pixel(2G)
pixel(2B)
[…]
pixel(nR)
pixel(nG)
pixel(nB)
```

where `pixel(iCOLOR)` is an unsigned char (ranging from 0 to 255).

This file describes an image in which each pixel is represented by three bytes, one per colour channel (RGB).

The board VGA employs three bits per pixel. The image is thus converted accordingly with the simplest possible algorithm: each pixel is stored into an unsigned char `pixel = 00000RGB` where R, G and B are 1 `if (pixel(iCOLOR) > 127)`.

Eventually, the image is put into a buffer formatted as required by the receiving serial block on the board ("*uart2BusTop*") - for further info about this please consult the pdf "*UART to Bus Core Specifications*", section 4.2 - and sent through the COM port specified by the user.

The maximum dimension of the image must have the product Height*Length smaller or equal to 256*256 due to the limitations on the BRAM of the board.

(To change the Baud rate modify the value 115200 in the function "*RS232_OpenComport*").

VideoGenerator.

The block works in a very simple way:

Reading Mode: it just checks if the blank signal is 1 or 0 and either drives output signals "R", "G", "B" (which go to the monitor) according to data coming from the memory `if blank = 0` or it displays black pixels `if blank = 1`.
When switch "colSwitch" is High the negative of the current pixel is sent to the screen.

Write Mode: black pixels are displayed.

Remark: we added a "*D-FLIP FLOP*" between the previous block "*ImgPositionNew*" and the "*VideoGenerator*" to account for delays in the synchronization between "*VideoGenerator*" and data read from "*Memory*".

SortingOffice.

A trivial block that just picks the useful data in the 8 bits coming from the "*Uart2BusTop*" (the three LSB are the RGB values needed) and discards the rest.

Clock25.

Since the VGA requires a 25MHz clock for the 640x480 visualization, this blocks halves the 50MHz clock of the board and sends it to the "*VgaRefComp*".

AddressManager2.

This block checks whether the user is sending dimensions, sending the image (Write Mode) or reading it from the memory (Read Mode).

In the first case, if the "switch1" is High (and "switch0" Low), the first 4 bytes sent correspond to image's "Height" and "Length"; these consist of 10 bits that are "extracted" in the following way:
- the whole first byte plus the two LSB of the second byte make up the Height;
- analogously, the entire third byte plus the two LSB of the fourth byte give the Length;

(Please note that Height and Length are not saved into the BRAM, but stored in this block.).

In Write Mode the block receives the correct address from the "*Uart2BusTop*" (address incrementation is done automatically by this module): AddressManager simply forwards the address to the "Memory" from "*Uart2BusTop*". The board is in Write Mode when "switch0" is in High state and "switch1" in Low State.

Eventually, moving down "switch0" to Low State (assuming you are in Write Mode) the block goes to Read Mode and starts scanning continuously all the addresses in which something has been previously stored (a maximum address value is set when the image is written) sending them to the "*Memory*" block in order to read data correctly.

ImgPositionNew.

This block receives Height and Length of the image from the block "*AddressManager2*" and, depending on the position of the switches on the board, it generates a signal called "*display*". This is sent to the block "*VideoGenerator*", allowing to visualize the image correctly on screen in the corner chosen by the user. The display signal enables visualization only in those regions in which the VGA is able to display.
For further info please consult "*VgaRefComp*" *.pdf.

```
Legend:

     Switch6 = 0, Switch7 = 0  -   Upper left corner
     Switch6 = 1, Switch7 = 0  -   Lower left corner
     Switch6 = 0, Switch7 = 1  -   Upper right corner
     Switch6 = 1, Switch7 = 1  -   Lower right corner
```

Note: due to synchronization issues the first column of pixels on screen can't be used and the position indicated by Hcounter and Vcounter in the "*VgaRefComp*" is displayed one pixel to the right with respect to the indicated one.

ResetManager.

This module provides reset signals for the "*VgaRefComp*" and the "*AddressManager2*" any time the user switches between Writing and Reading Modes or he decides to visualize the image in another position.
The reset signal prevents synchronisation errors from occurring in these occasions, restarting Image Reading and Visualization procedure.

VgaRefComp, uart2BusTop, Memory.

If you need additional info concerning these blocks, please consult the relative *.pdf files attached to the present Manual.

For anything else, contact us at the e-mail addresses specified at the beginning.