EE 310 Hardware Description Languages
Spring 2018

Laboratory Assignment #4

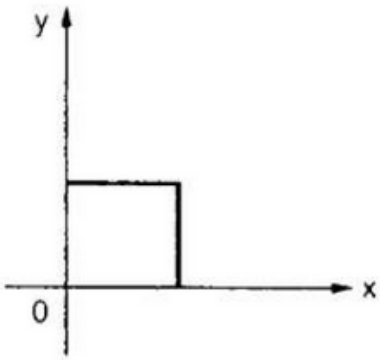
Due Date: 30 May 2018

In this lab, you will design and implement an image shear transform hardware and load this design into DIGILENT Spartan-3E starter development board. Your hardware should apply shear transform to a 128x128 input image and display the transformed 256x128 output image on a monitor connected to the FPGA board.

Shear transform slants the shape of an object. In this lab, you will implement X-Shearing which means X coordinates are changed and Y coordinates are preserved. To apply X-Shearing to an input image, each pixel coordinate must be multiplied with the shear transform matrix as shown below and pixel values of original pixel coordinates (x, y) must be assigned to resulting new pixel coordinates (x', y').

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix}$$

An image shear transform example is shown below.



Original image

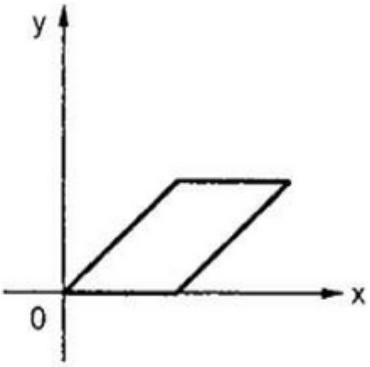


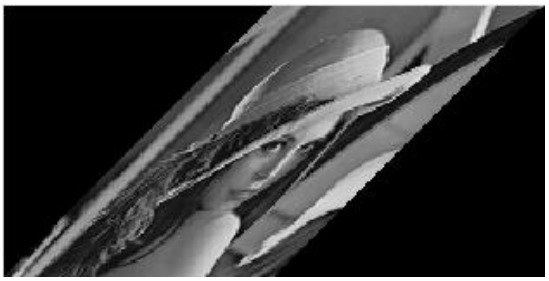
Image after shear transform

In DIGILENT Spartan-3E starter development board, one FPGA pin for each color channel (R, G, B) is connected to the VGA port. As shown in the below table, 8 different colors can be displayed on a monitor using these 3 VGA signals. In this lab, you will work with grayscale input images, and display the input and output images on a monitor as black and white. In order to display a grayscale image on a monitor as black and white, you should assign "1" to R, G and B channels for the pixels with a value greater than or equal to 128, and assign "0" to R, G and B channels for the pixels with a value less than 128.

VGA_RED	VGA_GREEN	VGA_BLUE	Resulting Color
0	0	0	Black
0	0	1	Blue
0	1	0	Green
0	1	1	Cyan
1	0	0	Red
1	0	1	Magenta
1	1	0	Yellow
1	1	1	White

The following figures show example input and output images. Images in the first row are grayscale (each pixel is 8 bits) lena input image and shear transformed output image. Images in the second row are black and white (each pixel is 1 bit) lena input image and shear transformed output image that will be seen on the monitor connected to the FPGA board.

Original Grayscale Image



Shear Transformed Grayscale Image

Original Image on Screen

Shear Transformed Image on Screen

You will be given “lena_input.v” module which preloads the grayscale lena image shown above into a BlockRAM. Therefore, the BlockRAM will contain 128x128 = 16384 8-bit pixels. In order to use this module, you have to add (right click -> add source) lena_input.v file to your project. The other files related to lena_input (lena_input.cgp, lena_input.mif, lena_input.ngc) should also be in the same project folder. You will be given “vga_controller_640_60.v” file which generates the VGA control signals needed for displaying an image on a monitor. You will also be given “display_template.v” file which displays a black and white image on a monitor using vga_controller_640_60 module. This file also shows how the BlockRAMs are instantiated and used.

First, you should reset your hardware with the reset button. Then, your hardware should wait until BTN1 (the button on the west side of the knob) is pressed. When BTN1 is pressed, it should display the input image as black and white on the left side of the monitor, apply the shear transform, store the output image into a BlockRAM and lit LED1 (rightmost LED). Your hardware should then wait until BTN2 (the button on the east side of the knob) is pressed. When BTN2 is pressed, it should display the output image as black and white on the right side of the monitor (while the input image is still on the left side of the monitor), and lit LED2 (next to the rightmost one).

In order to simulate a module that uses lena_input, you have to add lena_input.v file into your Xilinx ISE project. The other files related to lena_input (lena_input.cgp, lena_input.mif, lena_input.ngc) should also be in the same project folder. Do not try to simulate vga_controller or display_template files in Xilinx ISE, because they depend on hs and vs signals generated by the FPGA.

Your UCF file should contain the following lines.

```
NET "BTN_EAST" LOC = "H13" | IOSTANDARD = LVTTL | PULLDOWN ;
NET "BTN_WEST" LOC = "D18" | IOSTANDARD = LVTTL | PULLDOWN ;
NET "LED2" LOC = "E12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "LED1" LOC = "F12" | IOSTANDARD = LVTTL | SLEW = SLOW | DRIVE = 8 ;
NET "reset" LOC = "H13" | IOSTANDARD = LVTTL | PULLDOWN ;
NET "R" LOC = "H14" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST ;
NET "G" LOC = "H15" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST ;
NET "B" LOC = "G15" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST ;
NET "hs" LOC = "F15" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST ;
NET "vs" LOC = "F14" | IOSTANDARD = LVTTL | DRIVE = 8 | SLEW = FAST ;
NET "clk" LOC = "C9" | IOSTANDARD = LVCMOS33 ;
```

Your design should work at 50 MHz. Synthesize your RTL code using Xilinx XST synthesis tool targeting Xilinx Spartan XC3S500E FG320 FPGA with speed grade “-4”. Next, place and route the resulting netlist into Xilinx Spartan FPGA and generate the FPGA configuration bitstream using Xilinx ISE. Then, download the bitstream into Xilinx Spartan FPGA and verify your design on the board.

Finally, put all of your Verilog modules, Testbench, UCF file and FPGA bitstream into one “.zip” file named Lab4_Partner1Lastname_Partner2Lastname.zip (e.g. *Lab4_Azgin_Hamzaoglu.zip*) and submit this zip file using EE310 SUCourse website.