Watermarking and Image Processing using Verilog on Nexys 2
Digital Systems- ES 203 Project

Akhilesh Ravi(16110007), S. Vinu Sankar(16110143), Shivji Bhagat(16110149), Shubhranshu Singh(16110154)

Abstract In this project, we have watermarking and few other implemented image processing operations on a given image through FPGA Nexys 2. We send a given image in binary form to the FPGA Block RAM, with the help of Xilinx, and then perform some specific image processing applications depending user's choice in the FPGA itself and store the resultant image in another Block RAM and then display it through a VGA display. We use Verilog as the hardware description language and python for converting the given digital image into binary form.

Index Terms— Block RAM, FPGA, Image processing, Nexys 2, Verilog, VGA, Watermark, Grayscale, Sepia, Colour Filters.

I. INTRODUCTION

Image processing is a very powerful and a useful tool and currently has a widespread application. In the project, we have used FPGA to perform some image processing on a given image and produce the resultant image using VGA display monitor. The image processing operations we have performed are: watermarking, grayscale conversion, pixel effect and basic filter effects. The picture used was of 50x50 dimension and each pixel of the digital picture had twenty four bit data.
FPGA used- NEXYS 2
HDL used- Verilog HDL

Watermarking: Watermarking is the technology of superimposing some information on a signal or data which may be a video, a text or an image. It is generally used for authentication of the data, and claiming ownership of the data. Digital watermarking is the process of hiding some data in a carrier signal. The size of the signal does not change on adding a watermark. It is generally done for security, copyright reasons and it can be used for authentication of banknotes. It is similar to physical watermarking in a way that some algorithms have to be applied before we will actually be able to see the watermark (in a physical watermark due to different thicknesses of paper or different shades, the watermark would be visible only in certain lighting conditions and certain orientations).

Watermarking on image is currently very widely used by professional photographers to claim copyright. Watermarking on an image can be of two type:

Visible watermarking: In this, the image superimposed over the actual data is visible through naked eyes. The superimposed image is generally translucent and covers a small portion of the actual image. It is also used in advertisements. We have opted for visible watermarking in our project.

Invisible Watermarking: This involves superimposing the image on the original image in such way that it is not visible with naked eyes but can be extracted using suitable softwares. It provides more security to the data. There are different algorithms which can be used to achieve invisible watermarking.

Grayscale: The first form of photography produced grayscale images. The intensity of incident light was used to determine the amount of black and white. Nowadays, it is expressed in terms of bits. In the RGB

representation, there are 8 bits to represent each colour (with values from 0 to 255) and 24 bits in total. If it is a grayscale image, each pixel would ideally require only 8 bits to represent the intensity of light with white having a value of 255. However, if it is represented using RGB, then the values for R, G and B are all equal, that is, the same 8 bits are given to each colour.

Photographic print toning: Toning is the method by which a warmthness is added to the grayscale image of the actual image. The grayscale image depicts the brightness of each pixel. Toning is adding different colors to depict the same. There are many filter effects on images which are based on toning.

Sepia: Sepia is a photographic toning effect made using green and red signals. Sepia became a very widely used effect when it was introduced in the 19th century. The chemicals which were used to make this effect increased the lifetime of the images.

Other color filters: The color filters are also a type of photographic toning. Each of these effects are made by manipulating the red, blue and green data bit signals. We have cyan (blue and green), magenta (red and blue), blue and green filters. These colors adds the respective warmth to the images. We use these algorithms to beautify images and for some other vision applications depending on the use.

Brightness of the images: To get different brightness levels of the images, a completely white image is used. This image is superimposed on the original image to different extents. The extent of its contribution in the final image will determine the brightness of the image.

Pixel effect: This effect is very common nowadays and is widely used for beautifying purposes. This algorithm groups up a set of pixels and make them look like a single big pixel block. Here, we are grouping every 4 pixels, with alternative groups displaying the image pixels and black pixels respectively.

II. PROCEDURES

The following steps were broadly involved in our project :

a. Converting image into raw binary format.
   An image of 50 x 50 dimension was taken. It was converted into binary format and stored in '.coe' format using python and OpenCV . In the binary data, every twenty four bits represented a pixel. Hence, for each pixel, the R, G, B (Red, Green, Blue) values were represented by eight bits each and thus ranging from 0 to 255. For instance, if the .txt file begins as 1111000……..24 elements, then the first eight bits are the binary representation of the R value of first pixel and so on.
b. Sending the binary image file to the Block RAM. Block RAMs are the RAM memory blocks available in an FPGA board. We have used the Block RAMs to dump the main image and the watermark image data signals into the FPGA.They can be accessed and written within the FPGA using verilog codes. They take the input data from the computer as .coe file containing 24x2500 data bits for our project. One block RAM for our project uses about 7.5 kB of data. They are very useful in digital signal processing on FPGAs to store data on the board itself. They are made using an IP core generator using Xilinx software.
c. Retrieving the image from the Block RAM: The image has 2500 (50 x 50) pixels. Each pixel has 24 bits and is stored in a line. Each line is read one by one and the operations are performed on each pixel separately.
d. Converting the image to grayscale: The image is taken in binary form. Each pixel is stored as 24 bits. The R, G and B can have values from 0 to 255 (in stored in binary form). The pixels (24

bits) are read one by one and the following operation is done: the R,G, B values of each pixel are added and then divided by four to get new image data for the equivalent grayscale image and given as the new RGB values. Hence, for the grayscale image the new RGB values for each pixel would be the ratio of sum of initial R,G, B values and four.

e. Pixel Effect: The entire image is divided into a sets of eight pixels which are stored contiguously. Every alternate set of pixels is made black by giving it R, G, B values of 0, 0, 0.
f. Brightness: For changing the brightness of the image we add 255 to R,G,B of each pixel and divide by a 2,4,8,.. depending on the user's choice. Hence, the R,G,B values of the new image with different brightness are ½,¼,⅛, so on of that of initial image.
g. Watermark: Another image of dimensions 20 x 20 is embedded onto the original image. The square of 20 x 20 pixels on the lower right corner is taken and the RGB values of that square are being averaged with the other image. Ultimately a new image is blended with a portion of the main image.
h. Filter Effect: Different filter effects are added to the original image. One of the R, G, B of the resultant image is zero (eight bit zero) and the other two have the value equal to the sum of previous R,G,B values divided by four.
i. VGA interfacing: The pixel clock operates at a frequency of 25 MHz. The VGA screen refreshes at a frequency of 60 Hz. The resolution of the screen used is 480x640. The input to VGA has only 8 bits.3 bits each for red and green, 2 bits for blue.Pixels are displayed on the VGA screen, with proper vertical and horizontal time synchronisations.

III. Results and Discussions

The following images were used as inputs from the computer and the resultant images after performing the various operations on them follow these images.


Image 1


Image 2


Image 3


Image 4


Image 5

1. Watermarking: A watermark is added to the image: (Applied to all output images)

Image 1

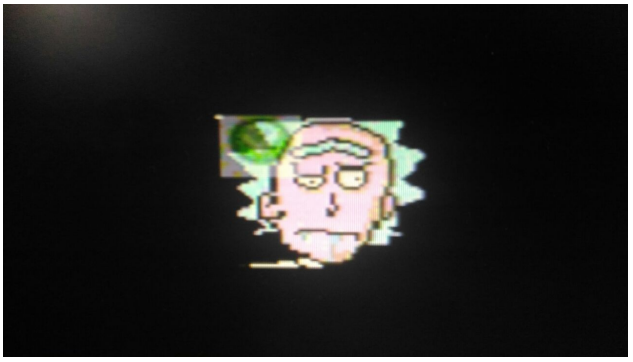


Image 2

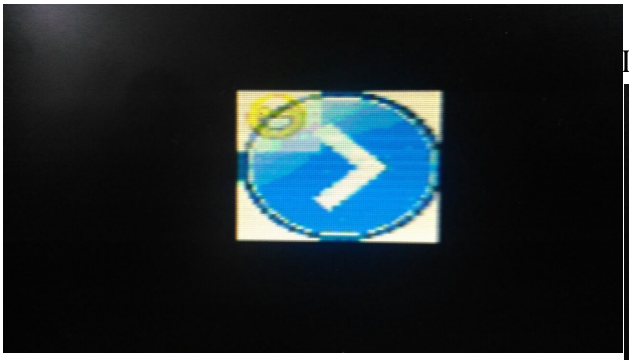


Image 3

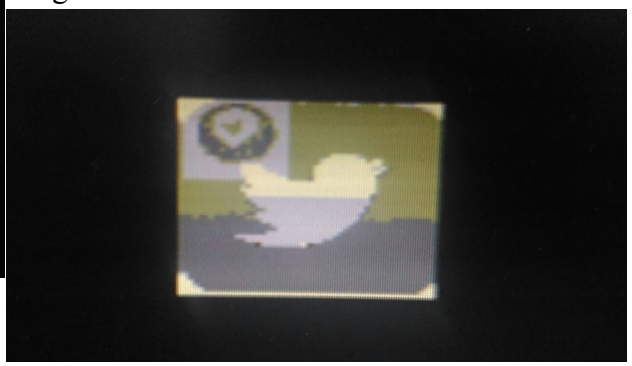
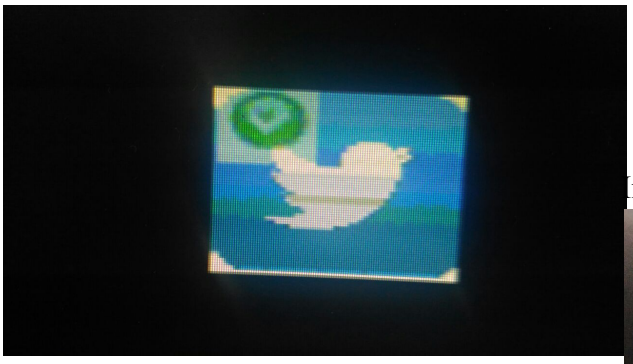
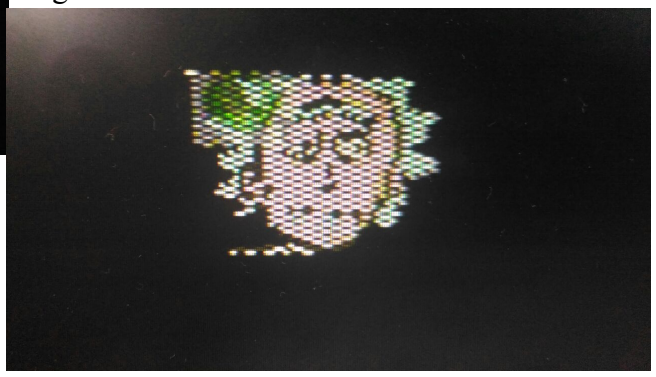


Image 3



3. Pixel Effect:(Choice bits are 000)

Image 1



2. Grayscale: The resultant images after converting the input images to grayscale is as follows: (Choice bits are 001)

Image 2

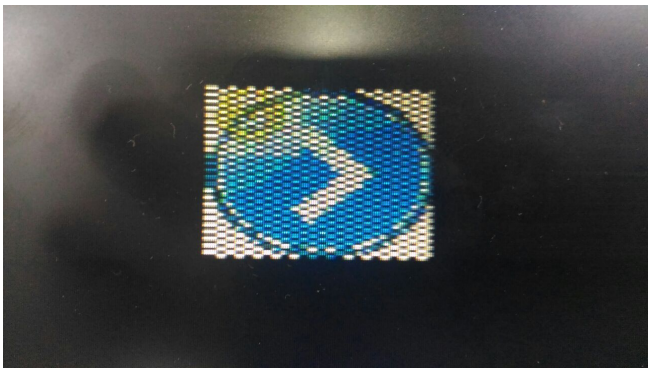


Image 1

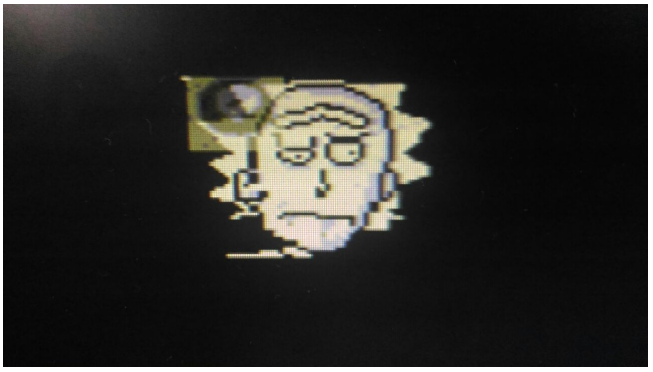


Image 3

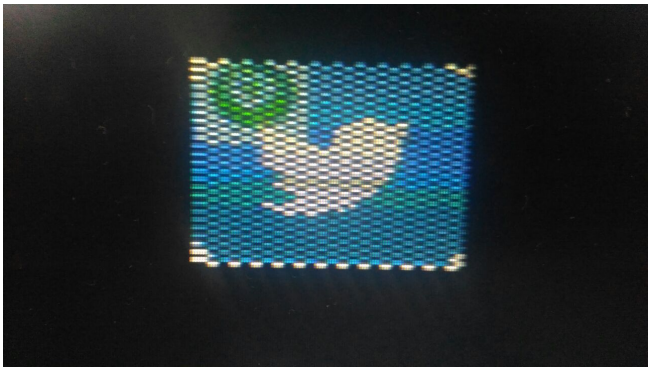


Image 2

4. Brightness: The brightness of the images have been changed by two levels:
(Choice bits are 010)

Image 1

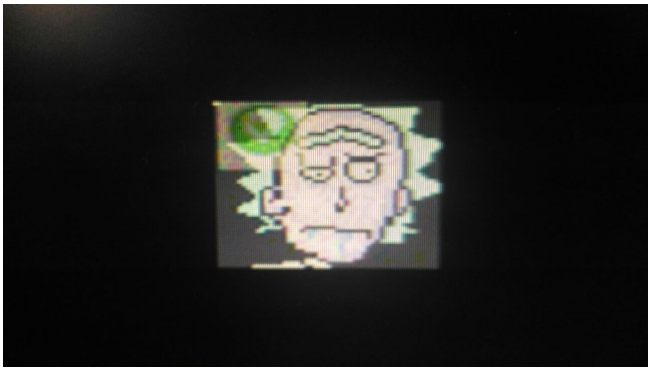


Image 2

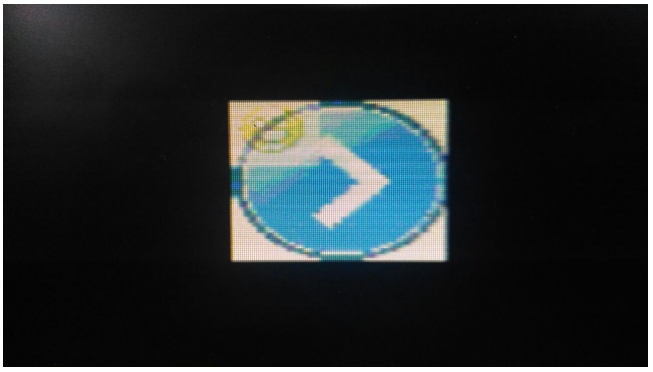
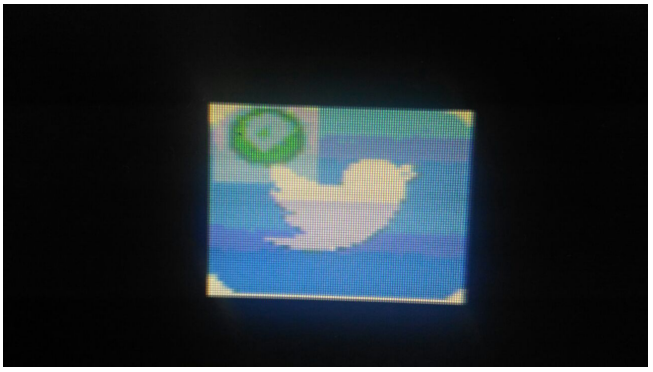
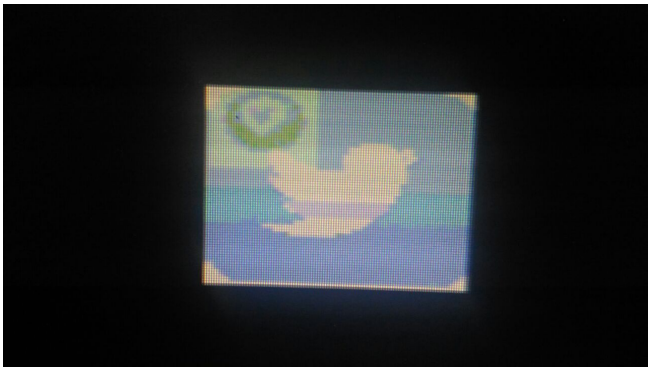


Image 3

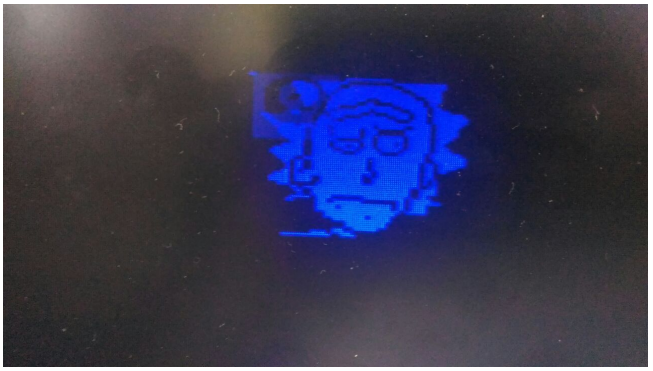


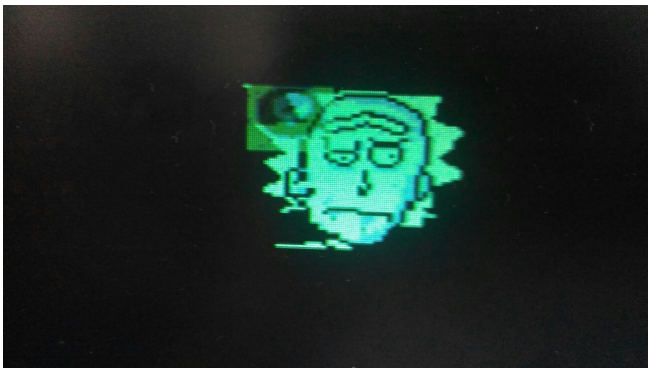


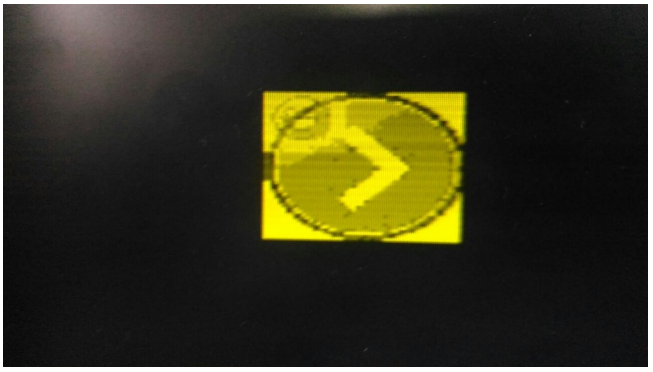
5. Filter Effects: Filters of blue, cyan, magenta, sepia and green have been applied separately.

(Choice bits are 011 for sepia, 100 for blue, 101 for magenta, 110 for green and 111 for cyan)

Image 1







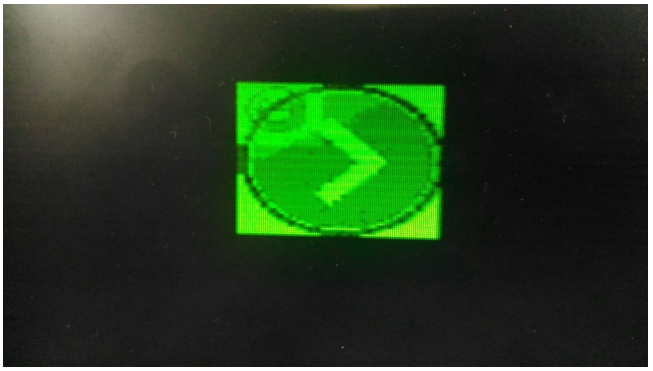


Image 3

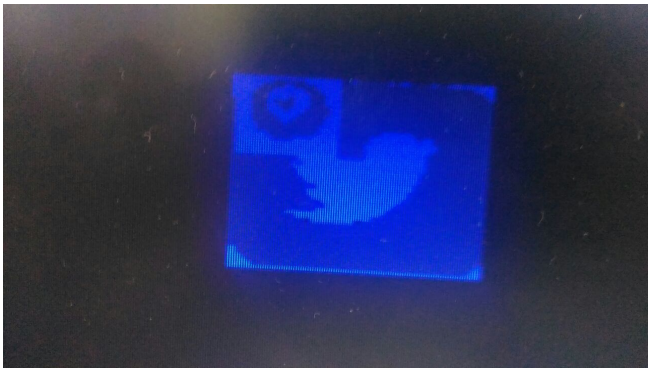
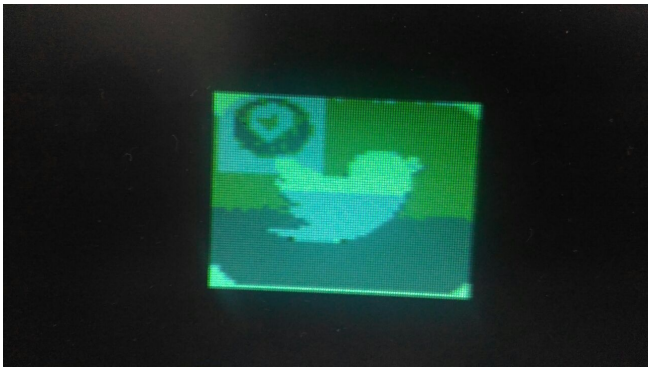
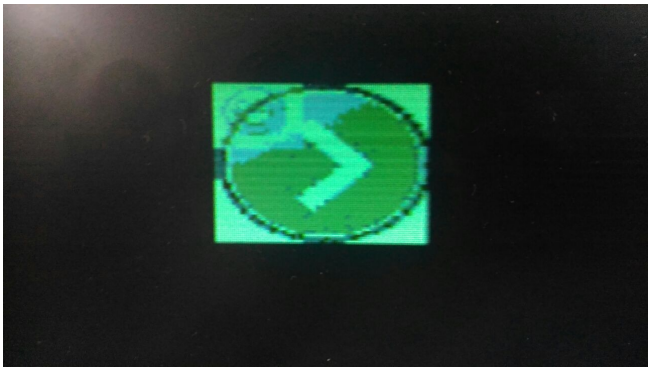
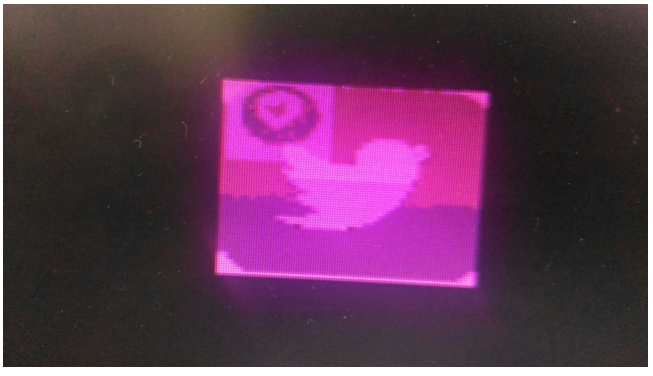
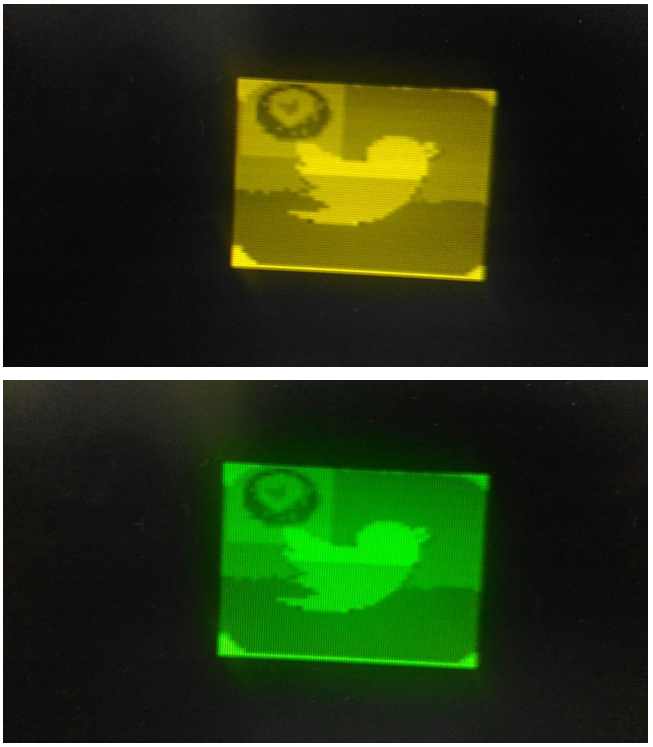


Image 2









IV. Conclusion

The entire process behind image processing using Verilog has been a challenging and interesting one. Starting from the image processing functions themselves and doing the conversion of the images into binary form, sending it to the FPGA using a Block RAM, processing the binary data, displaying the final image on a monitor using a VGA interface and more were very interesting. On the whole, the online resources for Verilog HDL were limited and in many cases we had to find the errors on our own. This helped us learn Verilog more in depth. Thus, we were able to implement various functions that comprised adding a watermark to the images, converting the images to grayscale, changing the brightness of the images by different levels, adding a pixel effect and adding various colour filters. On the whole, the module was an efficient module to implement various image processing functions.

V. Future Expansions

We would like to extend the project for real time videos. The image processing on the FPGA hardware was pretty fast, except for the displaying part on the VGA screen. The output images can be displayed using an HDMI interface, which will make the output image of 24 bits for each pixel and would retain the same image quality. We also want to extend it by adding Universal Asynchronous Receiver-Transmitter to analyze the output image from the FPGA hardware and compare it with the Python equivalent code output. The input image size can also be increased and a camera module interface can also be attached to the FPGA to take it to a practical application level. Watermarking is a trending research topic. We can implement different types of watermarking, both visible and invisible watermarks on images to increase security on copyrighted images.

VI. Acknowledgment

Performing the image processing tasks using FPGA is a challenging task in itself. It took us lot of research to understand how to go about in our project. We would like to acknowledge the continuous assistance and guidance provided by Prof. Joycee Mekie. We are indeed grateful to her for providing the opportunity to work in this project. We would also like to convey our gratitude to our course associate and project supervisor, Mr. Abhishek for guiding us from the beginning till end of the project. We would also like to convey our regards to Mr. Chandan for introducing extremely important concepts to us which proved to be really helpful for our project. We are greatly indebted to the electrical and electronics lab, IITGN, for providing us with all the necessary equipments required for our project.

VII. References

[1] https://www.researchgate.net/publication/262297066_Digital_Watermarking_Techniques_In_Image_Processing
[2] https://bytescout.com/products/enduser/watermarking/digital_watermark_types.html
[3] http://ieeexplore.ieee.org/document/4382120/?reload=true
[4] http://www-mtl.mit.edu/Courses/6.111/labkit/vga.shtml
[5] http://web.mit.edu/6.111/www/s2007/LECTURES/ramrom.pdf
[6] https://reference.digilentinc.com/reference/programmable-logic/nexys-2/reference-manual