INDIAN INSTITUTE OF INFORMATION TECHNOLOGY NAGPUR

Department of Electronics and Communications Engineering

HARDWARE DESCRIPTION LANGUAGES

PROJECT REPORT on "IMAGE PROCESSING USING VERILOG"

SUBMITTED TO: Dr. SUSHMITA DANDELIYA

SUBMITTED BY: G. AJAY KUMAR (BT22ECE053)

PIYUSH SHUKLA (BT22ECE050)

AZAD BABU (BT22ECE052)

SARTHAK DUBEY (BT22ECE042)

IMAGE PROCESSING USING VERILOG

INTRODUCTION:

In this project, we implemented image processing using Verilog. The input image was taken using a Python script, which converted the image into a format that Verilog could process. We focused on four key parameters to modify the image: contrast, brightness, threshold, and inverting. The contrast adjustment enhances the difference between light and dark areas. Brightness changes the overall light level of the image. The threshold operation converts the image into a binary form, distinguishing light and dark regions. Inverting reverses the colours, making light areas dark and vice versa. These operations were implemented using Verilog modules, demonstrating how hardware description languages can handle basic image processing tasks.

THEORY:

Contrast Manipulation:

Expanding the contrast range by assigning the darkest pixel value to black, the brightest value to white, and each of the others to linearly interpolated shades of Gray make good use of the display and enhances the visibility of features in the image.

Brightness Manipulation:

A dark region in an image may become brighter after the point operation and the commonly used point operation are increasing and decreasing of brightness. If an operator takes each pixel value and adds a constant number to it, then this point operation increases the brightness of the image and similar subtraction operator reduces the brightness

Inverting Images:

Inverting the contrast of an image creates a negative, like a photographic negative, which can highlight details better. For example, X-ray images are often viewed as negatives. If only part of the brightness is inverted, it creates an unusual effect that shows details in both dark and bright areas.

To invert an image, you simply reverse the pixel values. This can be done by multiplying each pixel value by -1 and then adding a constant to keep the values within the valid range. This method is simple and effective for enhancing specific details in images, especially when working with shadowed or overexposed regions.

Threshold Operation:

Thresholding operations are particularly interesting for segmentation in the process of isolating an object of interest from its background.

Thresholding an image means transforming all pixels in two values only. This is a special type of quantization comparing the pixel values with a given threshold value ath that is usually constant. That allows the separate the pixel values in two classes.

RESULTS:

Contrast Result:

Verilog result for Contrast Operation using threshold=127, value To Add = 10 and value To Subtract = 15 (right image)

Verilog result for Brightness Operation using sign = 1 and value = 60 (right image)

Verilog result for Threshold Operation using threshold=127 (right image)

Verilog result for Invert Operation (right image)

CONCLUSION:

This project demonstrated an efficient approach to image processing using Verilog. The input image was divided into individual pixels and converted into a hexadecimal format using Python. The processed hex data was then used in Verilog to apply four key parameters: contrast, brightness, threshold, and inverting. Each parameter was thoroughly verified, ensuring the desired output for all cases. This project highlights the synergy between software (Python) and hardware description languages (Verilog) for image processing tasks.

The applications of this project include hardware-based real-time image processing in embedded systems, image enhancement for medical imaging, and surveillance systems. Additionally, it can be integrated with FPGA for high-speed and low-latency processing. Future work could involve implementing advanced techniques like edge detection or filtering, extending the project to handle colour images, and optimizing for better hardware resource utilization. Real-time implementation on FPGA boards would further enhance its practicality for various applications.

REFERENCES:

1. Image Enhancement Methods Approach using Verilog Hardware Description Language

https://www.dasconference.ro/cd2012/data/papers/C60.pdf

2.FPGA Implementation of a Real-Time Audio Equalizer

https://www.researchgate.net/publication/306379854_FIR_Filter_for_Audio_Signals_Based_on_FPGA_Design_and_Implementation

3. Real-Time Audio Processing Using FPGA

https://www.researchgate.net/publication/306379854_FIR_Filter_for_Audio_Signals_Based_on_FPGA_Design_and_Implementation