Real-Time Video Noise Reduction Techniques Implementation on FPGA

Usama Ibrar

Department of Electronics

Quaid-i-Azam University, Islamabad, Pakistan

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Philosophy in Electronics

September, 2023

DEPARTMENT OF ELECTRONICS
QUAID-I-AZAM UNIVERSITY
ISLAMABAD, PAKISTAN

A thesis entitled *Real-Time Video Noise Reduction Techniques Implementation on FPGA* by Usama Ibrar in partial fulfilment of the requirements for the degree of Master of Philosophy, has been approved and accepted by the following,

Supervisor
Dr. Arshad Hussain
Assistant Professor
Department of Electronics
Quaid-i-Azam University
Islamabad, Pakistan

Chairman
Dr. Qaisar Abbas Naqvi
Professor
Department of Electronics
Quaid-i-Azam University
Islamabad, Pakistan

Allah will say, 'This day truthfulness shall benefit the truthful. For them there will be gardens with streams running in them, to remain in them forever. Allah is pleased with them and they are pleased with Him. That is a great success(Paradise).

(Al-Mā'idah : 119)

dedicated to my parents and my teachers...

Acknowledgements

I am grateful to Almighty Allah for providing health, thoughts, loving parents, excellent instructors, supportive friends, and the chance to contribute to the ever-evolving human knowledge.

The final prophet of Allah, Muhammad (peace be upon him), pushed his followers to pursue knowledge throughout life and set a bright example for mankind as the best Quranic disciple.

My research motivation comes from the Holy Quran, which invites humanity to reflect on their own and the entire creation.

I would like to express my gratitude towards:

- My supervisor Dr.Arshad Hussain for their patience, inspiration, valuable guidance, dilation, and discussions in helping me to deal with my problems.

- My mentor, my cousin Dr.Nadeem Afzal for his valuable time, criticism, advice, and correction for refining my work from beginning to end.

- My parents, who supported me financially and emotionally, deserve special thanks. I've always known you supported me and wished the best for me.

- My brothers (Tauseef, Haris, Samiullah), my sister (Ayman) for their support, unconditional love and always there to encourage me.

- Last but not least, my friends Mudasir, Nadia Afzal, and Nadia Liaquat, for all the support and encouragement you give me and made my life pleasant during this time.

Abstract

The Gaussian filter and the Median filter are two types of noise reduction filters, and this thesis provides a complete presentation and analysis of each of these filters. This was a conscious choice that was motivated by the large amount of study that was undertaken on FPGA implementation for grayscale photos, in contrast to the minimal attention that was paid to color images. The filters have been particularly developed to work on grayscale image formats. Because of the rise in data dimensions, the processing of color images necessitates much more complex computations. Furthermore, the similarities in edge characteristics between grayscale and color pictures have led to color image processing being a field that has received comparatively less research than grayscale image processing.

This research makes a substantial contribution to the development of FPGA-based video processing by delivering efficient noise reduction capabilities as well as a unique camera interface solution, both of which are backed by a specific VLSI architecture. The aforementioned technological improvements have substantial ramifications in a variety of fields, including robotics, surveillance, and automation, all of which place a premium on being able to interpret video in real-time.

Contents

List of Figures

List of Tables

Abbreviations

AP-SoC	All Programmable-SoC
CMOS	Complementary Metal-Oxide Semiconductor
DDR	Double Data Rate
FMC	FPGA Mezzanine Card
FPGA	Field Programmable Gate Array
HDMI	High-Definition Multimedia Interface
OLED	Organic Light-Emitting Diode
OV	OmniVision
PL	Programmable Logic
Pmod	Peripheral Module
PS	Processing System
RTOS	Real Time Operating System
SCCB	Serial Camera Control Bus
SPI	Serial Peripheral Interface
SoC	System on Chip
VGA	Video Graphic Array
VHDL	Very high-speed integrated circuit Hardware Description Language.
XADC	Xilinx Analog-To-Digital Converter

Chapter 1

Motivation and Problem Statement

1.1 Research Objective

The goal of this research work is to conduct a comprehensive study focused on assessing the FPGA-based (Field Programmable Gate Array) implementation of real-time noise subtraction for camera data logging solutions. The primary purpose is to investigate and understand how to effectively use FPGA technology to remove unwanted noise from data captured by cameras in real-time. This noise can distort the quality of the captured images or videos. By implementing noise removal techniques using FPGA (discuss in Chapter 3), we aim to enhance the clarity and accuracy of the captured data. This is particularly important for applications where accurate and clear data is crucial.

Another essential goal is to engage with FPGA technology, thereby acquiring a deep understanding of its intricate complexities. This research endeavor also aims to streamline RTL (Register-Transfer Level) operations and economize the utilization of FPGA resources effectively.

1.2 Problem Statement

This thesis is going to concentrate on the very important problem of reducing noise in the video data that is taken by cameras. The presence of unwelcome noise significantly lowers the overall quality of the information that is recorded. The integrity of the picture data may be affected by several undesired kinds of noise, including interference noise, noise caused by equipment, and salt-and-pepper noise.

The combination of these many sources of noise creates a major barrier that must be overcome before image processing algorithms can perform as intended. As a direct consequence of this, the outcomes produced by these algorithms are often unsatisfactory. This issue further restricts our capability to draw meaningful inferences from the processed data, which necessitates a full examination of noise removal techniques and their incorporation into image processing algorithms to increase the efficiency of these algorithms and the overall quality of the findings.

The incorporation of pre-image processing methods, such as the Noise Reduction algorithm, has significant importance within the collection of image processing approaches previously described. The first stage is crucial since it is essential to deal with pictures that are free from any kind of noise interference before proceeding to more complex image processing tasks such as object recognition or motion analysis.

In order to address this issue, noise reduction filters are used. In the present context, two notable filters, namely the Gaussian filter and the Median filter, have been chosen based on their shown efficacy. The use of these filters is of great significance in the improvement of picture quality via the efficient reduction or elimination of unwanted noise artifacts.

1.3 Motivation

Video processing finds extensive utility across diverse applications, encompassing but not limited to, object detection, video surveillance, coin detection, and face detection, as referenced in [1]. These systems exhibit versatility and serve various purposes, including surveillance, agricultural applications, monitoring of parking facilities, and supervision of children and elderly individuals, among others.

The utilization of noise reduction algorithms, such as the Gaussian and Median filters, is motivated by various compelling factors. The presence of noise, in its various manifestations, can have a substantial detrimental impact on the overall quality of digital images. This is primarily due to the introduction of undesirable pixel fluctuations, which in turn obscure crucial details within the image. Through the effective reduction of noise, these algorithms not only restore clarity to images but also enhance their visual appeal. Additionally, they fulfill a crucial function in upholding the precision of subsequent image processing endeavors, such as the identification of objects and the analysis of motion, through the elimination of distortions induced by noise.

In disciplines such as medicine, where precision is of utmost importance, the necessity for imaging that is devoid of noise becomes crucial in order to ensure precise diagnoses and effective planning of treatment. Furthermore, the implementation of noise reduction techniques plays a significant role in enhancing the overall visual appeal of multimedia content, thereby contributing to its aesthetic quality and captivating the audience. In the context of resource-constrained and real-time applications, the utilization of these techniques aims to optimize computational resources, thereby facilitating efficient processing.

The applicability of noise reduction techniques extends beyond the scope of video processing, encompassing various domains such as photography and industrial quality control. Noise reduction algorithms demonstrate their value in a wide range of situations, including improving the visual clarity of a cherished photograph and ensuring the quality of products in a manufacturing environment.

In summary, the utilization of noise reduction algorithms, such as Gaussian and Median filters, has a significant influence on multiple aspects of image and video processing. The significance of these technologies in the contemporary digital landscape is underscored by their capacity to improve image quality, preserve accuracy in subsequent tasks, and demonstrate utility across a wide range of domains.

1.4 Thesis Organization

This thesis is organized as follows:

Chapter 2, provides an introductory insight into the preliminary implementation of noise reduction algorithms, specifically applied to real-time data acquired by the camera interfaced with an FPGA.

Chapter 3, provides a thorough analysis of noise reduction techniques, with a special emphasis on the Gaussian filter and the Median filter. This discourse not only expounds upon the underlying concepts of these filters but also offers a comprehensive examination of their operational processes.

Chapter 4, offers a comprehensive exploration of the experimental hardware employed throughout this research, with detailed explanations and insights provided.

Chapter 5, serves as a comprehensive resource detailing the different modules integral to the functionality of the overall hardware setup. Within this chapter, we delve into detailed simulations, provide a comprehensive discussion of module results, and thoroughly investigate the utilization of FPGA resources.

Chapter 6, is dedicated to presenting the conclusions drawn from this work. Furthermore, it serves as a platform for the identification and discussion of emerging trends and prospects for the future.

Chapter 2

Introduction

In recent times, the rapid progression of digital technology has catapulted several disciplines to unparalleled heights of creativity and efficacy. The Field-Programmable Gate Array (FPGA) is a technology that has attracted considerable interest throughout the ongoing digital revolution. Field-programmable devices, such as Field-Programmable Gate Arrays (FPGAs), have seen a significant increase in their level of acceptance and use due to their inherent versatility in facilitating the implementation of digital circuitry [2]. FPGAs are distinguished among these devices for their notable benefits, including rapid production turnaround, low initial costs, and particularly, design freedom. The aforementioned characteristics have made Field-Programmable Gate Arrays (FPGAs) a compelling option for a diverse array of applications, with a particular emphasis on the domain of image processing [3].

Image processing is a fundamental component of computer vision and digital signal processing. It involves the alteration of pictures in order to extract significant information, improve visual quality, and enable automated analysis. The increasing need for real-time image processing in many sectors has established FPGAs as a versatile technology capable of rapidly and effectively performing image processing algorithms. The image processing workflow encompasses a systematic sequence of operations aimed at manipulating and analyzing digital photos in a structured manner[4–6].

The procedure commences with the capture of images using a range of equipment, including cameras, scanners, or sensors. Following the acquisition phase, the obtained pictures undergo a preprocessing stage aimed at enhancing their quality in preparation for further analysis. This refinement involves fundamental tasks like noise reduction, contrast improvement, and scaling. After the initial preprocessing stage, image enhancement methods are used to optimize the visual quality of the picture by modifying factors such as brightness, contrast, and sharpness.

The main focus of this thesis has been the use of filtering strategies for incoming camera video streams or real-time video feeds in order to develop noise reduction techniques. The technique of reducing noise is a key aspect of image processing, which plays a critical role in enhancing the quality and dependability of digital photographs. The presence of noise, which arises due to many variables such as limits in sensors or challenges in transmission, may add distortions that hinder the process of visual interpretation. Noise reduction solutions are carefully crafted to effectively tackle these difficulties. The procedure involves the use of filters that selectively mitigate irregular pixel values caused by noise while retaining essential picture characteristics.

Spatial-domain methods, such as Median and Gaussian filtering, are used to intentionally alter the values of surrounding pixels in order to mitigate the effects of noise. Spatial filtering is a crucial component in a wide range of image processing applications, including tasks such as noise reduction, blurring, edge enhancement, and feature extraction[7].

Real-time processing is an essential need in several sectors, ranging from medical imaging to driverless cars, within the domain of image processing. this thesis has explored the complex realm of FPGA-based image processing and its use in the context of real-time video streams. FPGAs in addressing the increasing requirements of image processing in many sectors have been underscored. Moreover, the prioritization of noise reduction approaches highlights the need to obtain high-quality and dependable picture data throughout the period of digital transition. FPGA technology is at the forefront of driving advancements in image processing, providing a multifaceted, economically viable, and adaptable solution for academics and engineers engaged in this rapidly evolving domain.

Chapter 3

Image Noise Subtraction Algorithms

This chapter discusses several noise subtraction methods and algorithms. It is a major problem for the researchers to find an effective way to remove noise from the photos before processing them for additional analysis. At the time of image capture or transmission, noise might impair the image. Noise removal from the photos is performed with the utmost attention prior to using image processing tools on the image.

Image noise can be classified into a number of different categories, such as Impulse noise (also known as salt-and-pepper noise), Amplifier noise (also known as Gaussian noise), Shot noise (also known as poison noise), Quantization noise (also known as uniform noise), Multiplicative noise (also known as Speckle noise), and Periodic noise [8].

The following are the main causes of noise in digital images:

- The environment may have an impact on the imaging sensor during image capture.
- Image noise could appear due to insufficient light levels and sensor temperature.
- The transmission channel's interference may potentially cause image disruption.

- The image may also contain noise if there are dust particles on the scanner screen.

Different filters can be used to cope with these sorts of noise, however, in this thesis, we focus on two filters in particular:

- Gaussian Filter
- Median Filter

3.1 Gaussian Filter

By employing filters, the enhancement and noise reduction of images can be achieved. Within the domains of image processing and computer vision, the Gaussian filter stands as a form of linear filter specifically designed to execute tasks encompassing blurring, noise reduction, and feature extraction. The Gaussian filter's functionality revolves around the convolution of an image with a Gaussian kernel— 2D matrix (see Figure 3.1(a)) with values that resemble the properties of a Gaussian distribution. This kernel effectively computes an average of nearby pixel values when it is applied to specific pixels throughout the image. Notably, the importance of the central pixel grows, and as one moves away from the center, the significance of that pixel gradually decreases. This results in a blurring or smoothing of the image (see Figure 3.1(b)).

To design an effective image processing algorithm, harnessing FPGA stream-based processing is crucial. This method, known as **"Sliding Window"** or **"Windowing Operators"** maximizes FPGA capabilities (see Figure 3.1(c)). Common in tasks like object detection, image classification, and segmentation, the sliding window approach involves moving a smaller window across an image at various scales to identify objects. The architecture typically includes line buffers storing image rows and 2D window buffers containing localized regions essential for pixel computations. For instance, a Gaussian filter with a 3x3 kernel demands three line buffers and a matching window buffer. The architecture's structure is illustrated in Figure 3.1(c). This strategy optimizes FPGA's parallel processing, essential for real-time image analysis.

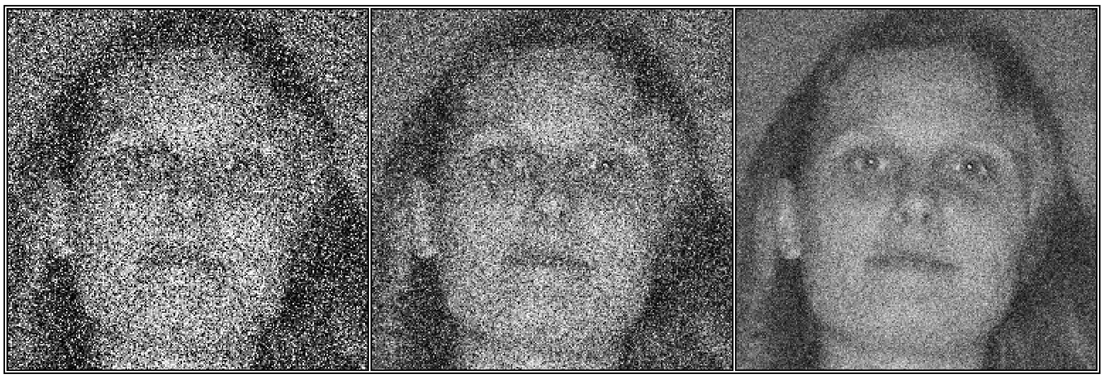
3.2 Median Filter

The Median filter primarily serves to eliminate irrelevant details, particularly salt-and-pepper noise in images (see Figure 3.2(a)) [9]. In contrast to the Gaussian filter, it excels at preserving structural edges [10]. This nonlinear technique is widely used in image processing to mitigate "salt and pepper" noise while maintaining edges. Unlike convolution, the Median filter excels in both noise reduction and edge preservation [11]. Employing a constant window size, it processes neighboring pixels to produce outcomes. The Median filter's consistent application across the entire image impacts both noisy and noise-free pixels, ensuring a comprehensive modification.

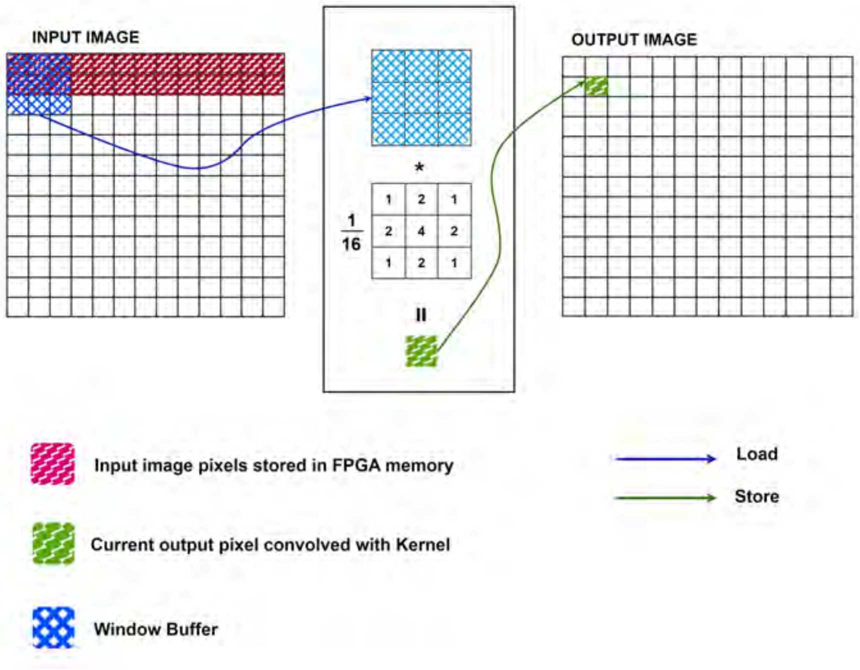
The Median filter algorithm leverages the sliding window concept previously discussed. Input image pixels are received and stored in line buffers, as established. Within each window buffer, pixel values undergo sorting in ascending order to extract the median value the midpoint in a sorted array. This calculated median value replaces the original value of the window buffer's central pixel. This operation recurs across all windows throughout the image, constituting the algorithm's iterative process for noise reduction and enhancement. The architecture's structure is illustrated in Figure 3.2(b) . [12, 13]

$$\frac{1}{16}\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline\end{array}$$

(a)Gaussian kernel— 2D matrix

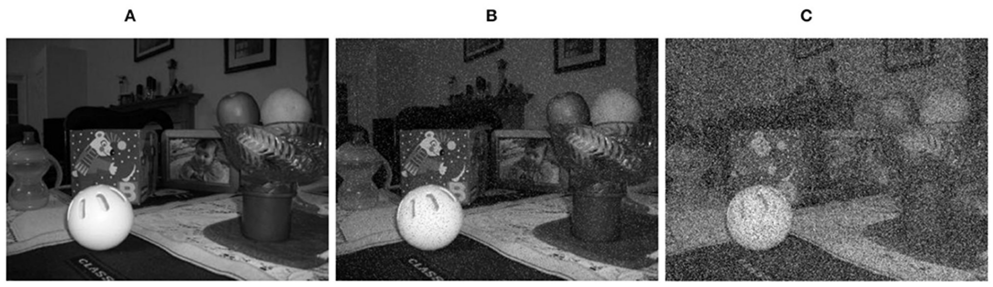


(b)Noise Reduction in Image using Gaussian Filter

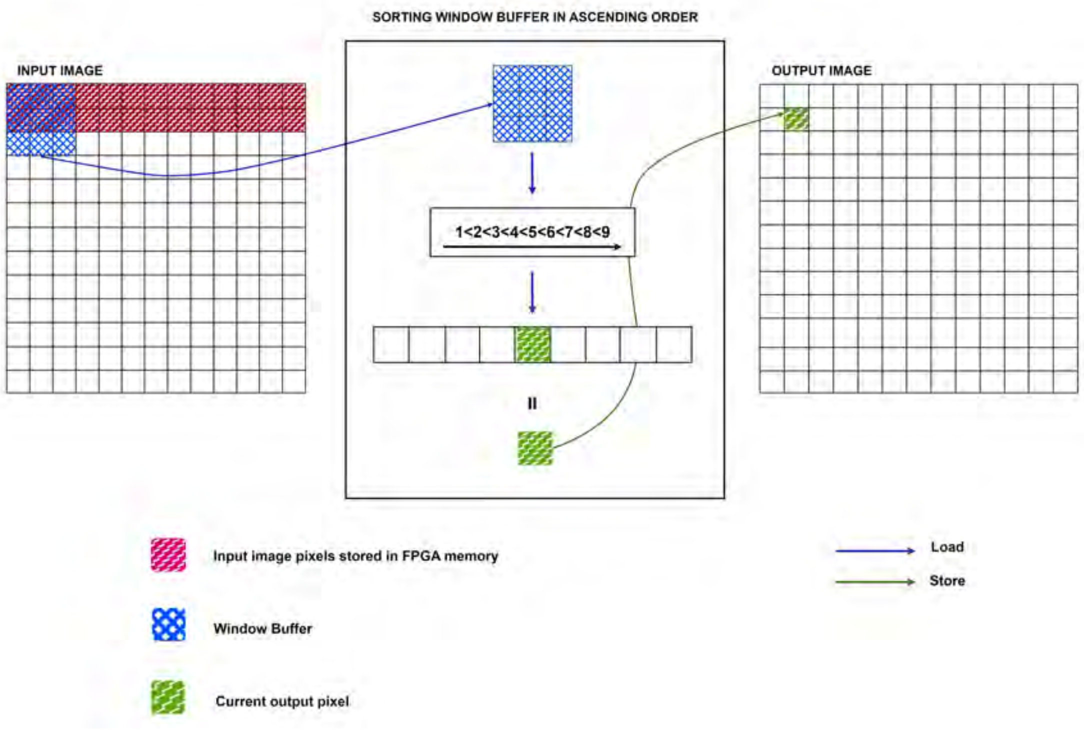

INPUT IMAGE
OUTPUT IMAGE
*
1
16
1 2 1
2 4 2
1 2 1
=
Input image pixels stored in FPGA memory
Current output pixel convolved with Kernel
Window Buffer
Load
Store

(c)Sliding Window Architecture

Figure 3.1: Implementation of Gaussian Filter Algorithm



(a): (A) original image; (B) salt and pepper noise with an intensity of 5; (C) salt and pepper noise with an intensity of 40.



(b)Sliding Window Architecture

Figure 3.2: Implementation of Median Filter Algorithm

Chapter 4

Experimental Hardware Overview

This chapter comprehensively addresses the hardware components employed within the experimental scope of my thesis work. I employed the ZedBoard FPGA, a VGA display, and the CMOS OV7670 Camera Module in my study to achieve real-time video capture. Each of these components will be individually elaborated upon in subsequent sections.

4.1 Board Introduction and Specifications

The ZedBoard is an evaluation and development board (see figure 4.1) based on the Xilinx ZynqT-7000 All Programmable SoC (AP SoC). The Zynq-7000 AP SoC, which has a dual Cortex-A9 Processing System (PS) and 85,000 Series-7 Programmable Logic (PL) cells, can be used in a wide range of applications. The ZedBoard is a great platform for designers of all skill levels because it provides a robust combination of onboard peripherals and expansion options.

4.1.1 Features

The ZedBoard incorporates the xc7z020clg484-1 Zynq device, featuring a robust set of components tailored for versatile embedded system development. At its core, the board is equipped with a dual-core Arm Cortex-A9 processor, providing substantial processing power. Complementing this, the ZedBoard

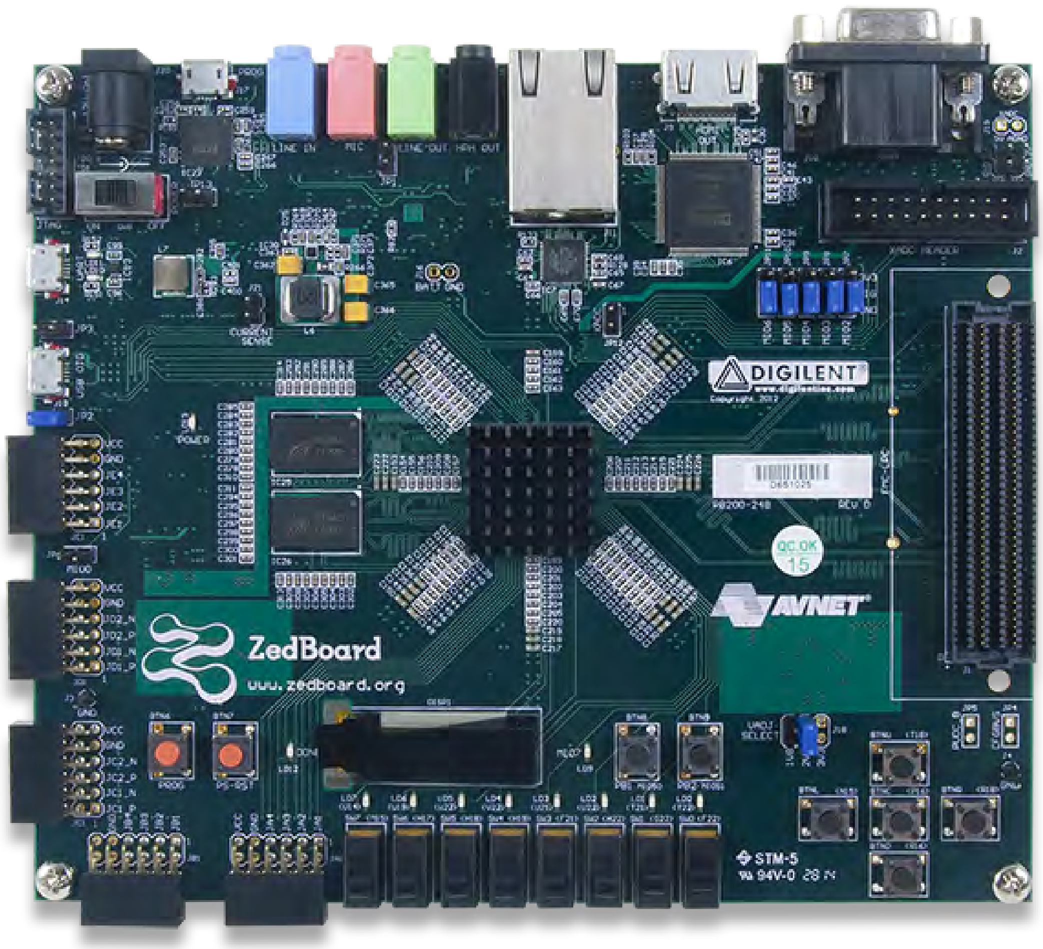


Figure 4.1: Zedboard (**Z**ynq **E**valuation and **D**evelopment Board)

includes 512 MB of DDR3 RAM, ensuring efficient memory utilization for diverse applications. Storage capabilities are addressed through a 256 MB quad-SPI Flash and a 4 GB SD card, offering flexibility for program storage and data handling.

Facilitating seamless connectivity, the ZedBoard integrates an onboard USB-JTAG for programming convenience, alongside USB OTG 2.0 and USB-UART interfaces for efficient communication. Networking needs are met with a 10/100/1000 Ethernet port, enabling high-speed data exchange. Additionally, the board features comprehensive audio interfaces, including Line-in, Line-out, headphone, and microphone inputs. This well-rounded set of features positions the Zed-

Board as an ideal platform for a wide range of embedded system development projects.

4.1.2 Application

- Video processing
- Reconfigurable computing
- Motor controls
- Software acceleration
- Embedded Linux
- RTOS development

4.2 Image Sensor (Camera Module)

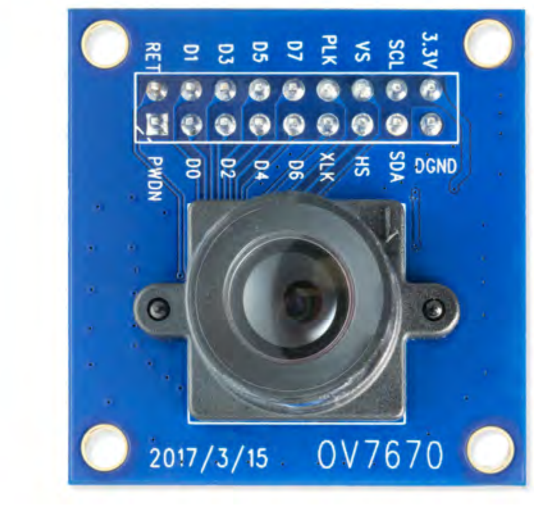


Figure 4.2: Image Sensor CMOS OV7670 Camera

The OmniVision OV7670 color camera chip is an advanced CMOS image sensor designed for low-voltage operation, encompassing full VGA camera functionality and image processing capabilities within a singular integrated chip.

The OV7670 camera yields 8-bit, full-frame images. Its operation is governed by the Serial Camera Control Bus (SCCB), an I2C interface, enabling precise sensor control. Operating within VGA parameters, this product achieves a remarkable 30fps image array performance, accompanied by extensive user command over image quality, format, and data output. The SCCB interface enables the programmability of fundamental image processing functions including exposure, gamma correction, white balance, color saturation, and hue control. Operating at 640×480 resolution, equivalent to 0.3 Megapixels, the OV7670 stands as a versatile imaging solution.

4.2.1 Core Specification

This section outlines several specifications of the OV7670 Camera Module:

- Resolution: 640×480 VGA
- Provides output compatibility for Raw RGB (GRB 4:2:2, RGB565/555/444), as well as YUV (4:2:2) and YCbCr (4:2:2) formats.
- High sensitivity for low-light operation
- Automatic Gain Control (AGC), Automatic White Balance (AWB), Automatic Black-Level Calibration (ABLC), Automatic Band Filter (ABF), and Automatic Exposure Control (AEC)
- Sharpness (edge enhancement), color saturation, hue, gamma, and anti-blooming are all aspects of an image's quality controls.
- Lens shading correction
- Flicker (50/60 Hz) auto detection

4.2.2 Pinout Description

In this section, we present an illustration of the camera module's pin arrangement (refer to Figure 4.3), along with a detailed description of the specific functions attributed to each pin (outlined in Table 4.1).

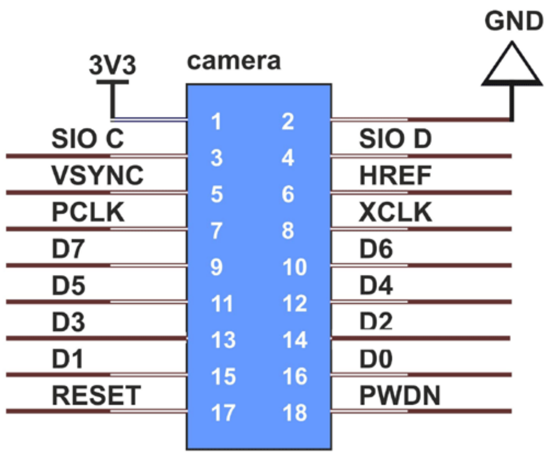


Figure 4.3: CMOS OV7670 Camera Module Pinout

PIN NAME	DESCRIPTION
VCC	3.3v Power Supply
GND	Ground
SIOC	Serial Interface Clock
SIOD	Serial Interface Data I/O
VSYNC	Vertical Synchronization ,Active high , Indicate active frames
HREF	Horizontal Reference , Active high , Indicate active pixels
PCLK	Pixel Clock Output from Sensor
XCLK	Master Clock into Sensor
PWDN	Power Down Mode Selection
RESET	Reset All Register to their Default value, Active Low
D0:D7	8-bits Data Pixel Output

Table 4.1: Pin Description of CMOS OV7670 Camera

4.2.3 Camera Controller

The camera Controller basically a The pins on the OV7670 camera can be classified into two main categories:

Input pins **(XCLK, PWDN, RESET, SIOD, and SIOC)** originating from an FPGA and used for camera configuration. These pins play a crucial role in configuring the internal registers of the camera to optimize its performance.

Of these pins, **SIOD** and **SIOC** hold particular importance. They are utilized to transmit register configuration data to the camera using the I2C standard, a widely-used communication protocol. The **SIOD** pin is responsible for sending data, while the **SIOC** pin provides the clock signal required for synchronous communication.

Another significant input pin is **XCLK**, which supplies an external clock signal to the camera. This clock operates at a frequency of 25MHz, ensuring synchronized operation between the camera and the controlling FPGA.

PWDN serves as a power-down mode selection pin with two distinct modes: normal mode (bit set to 0) and power-down mode (bit set to 1). This pin is pivotal in managing the camera's power state and operation.

RESET is an active low 1-bit pin that holds the capability to reset all internal registers to their default values when set to 0. Conversely, when set to 1, the camera operates in its normal mode.

All of the aforementioned functions are managed through the **OV7670 Controller** block, depicted in Figure 4.4. This controller serves as the intermediary between the FPGA-driven input pins and the camera's internal configuration. Through a coordinated interplay of these pins, the OV7670 camera can be optimally configured for its intended imaging tasks.

The output pins, namely **PCLK, VSYNC, HREF, DATA[7:0]**, are integral components emerging from the camera module and interfacing with the FPGA, specifically within the OV7670 Capture module depicted in Figure 4.4. These pins collectively contribute to the image capture process and subsequent data transfer.

The **PCLK** pin serves as the pixel clock, operating at a frequency of 24MHz. This clock signal synchronizes the timing of pixel data transmission, ensuring orderly and consistent image capture.

VSYNC is attributed to vertical synchronization, a crucial role in framing images. This signal manages the count of lines within each frame, facilitating the coherent arrangement of image data.

The **HREF** pin handles horizontal synchronization duties. It keeps track of pixel counts within each line of the frame, maintaining precise alignment of image data.

The **DATA[7:0]** pins collectively carry an 8-bit stream of captured image data. This data stream is then stored within the memory system of the FPGA for further processing.

The specialized **OV7670 Capture** module, shown in Figure 4.4, makes it easier to manage these output pins. This module creates a smooth connection between the FPGA's processing capability and the camera's output pins, making it possible to record, store, and manipulate picture data quickly and effectively.

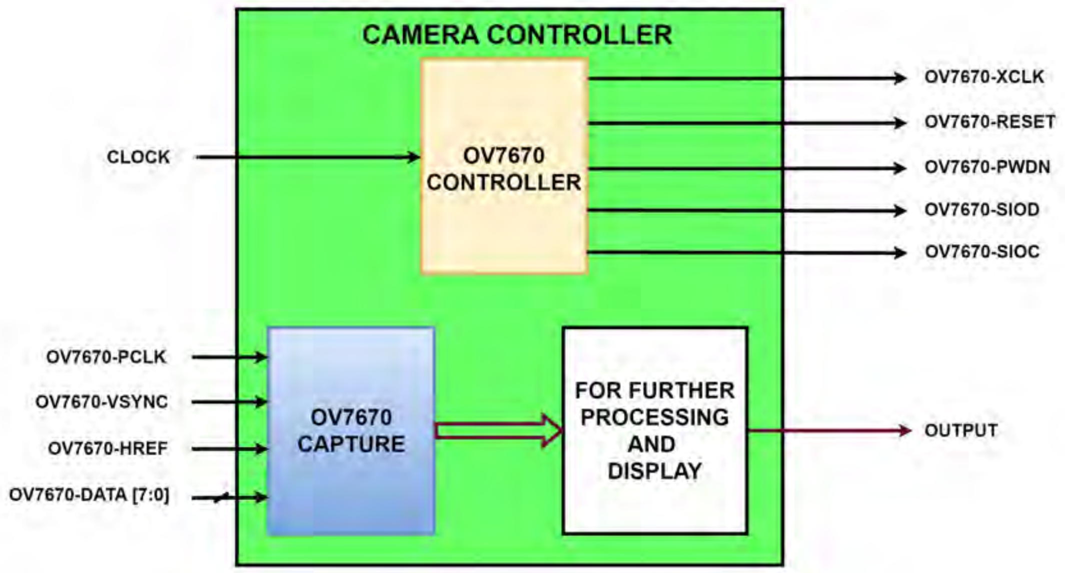


Figure 4.4: Camera Controller RTL Module

4.2.4 Application

The OV7670 camera module finds utility in various domains, including:

- Object detection
- Surveillance cameras
- Machine vision systems
- Gesture recognition
- Various image or video processing.

4.3 VGA Display Interface

To display the filtering results utilize a VGA interface display. The VGA interface, being a standardized communication channel, has gained extensive application across various domains. VGA is designed to operate within specific display modes, accommodating diverse resolutions and refresh rates[14]. In this context, a resolution of 640×480 is employed, aligning with the default resolution of the OV7670 camera.

4.3.1 VGA Interface Signal

The VGA interface serves as a standard for analog signals in computer displays. It encompasses two types of signals: data signals and control signals (as detailed in Table 4.2).

When implementing VGA control with an FPGA, our attention is directed toward five key signals:Red data, green data, blue data, horizontal synchronization, and vertical synchronization. By transmitting these signals from the FPGA to the VGA interface. The Zedboard produces a pin configuration of 12 bits, allocating 4-bit to each color channel (RGB). Before sending the data signal to VGA, the Zedboard initiates a conversion process, utilizing its integrated 4-bit Resistor-Ladder Digital-to-Analog Converter (DAC) for each color channel, to convert it into an analog signal, establish precise control over the display output(see figure4.5).

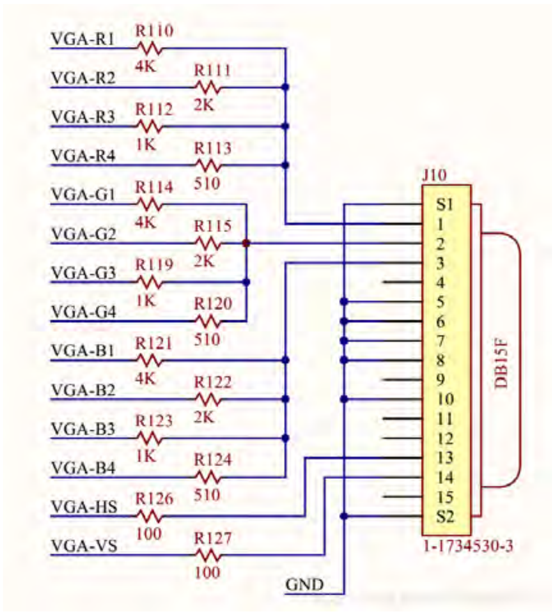


Figure 4.5: Pin Connection of VGA Port

DATA SIGNAL	RED
	GREEN
	BLUE
CONTROL SIGNAL	HORIZONTAL SYNC
	VERTICAL SYNC

Table 4.2: VGA Interface Signal

4.3.2 VGA Timing Specification

The VGA interface operates using highly accurate timing signals, and these particular standards are established and upheld by the reputable VESA organization (www.vesa.org). The VGA encompasses distinct display standards, including resolutions such as 640×480, 720p, 1080p, 4K and more. Each of these resolutions adheres to specific timing specifications in accordance with

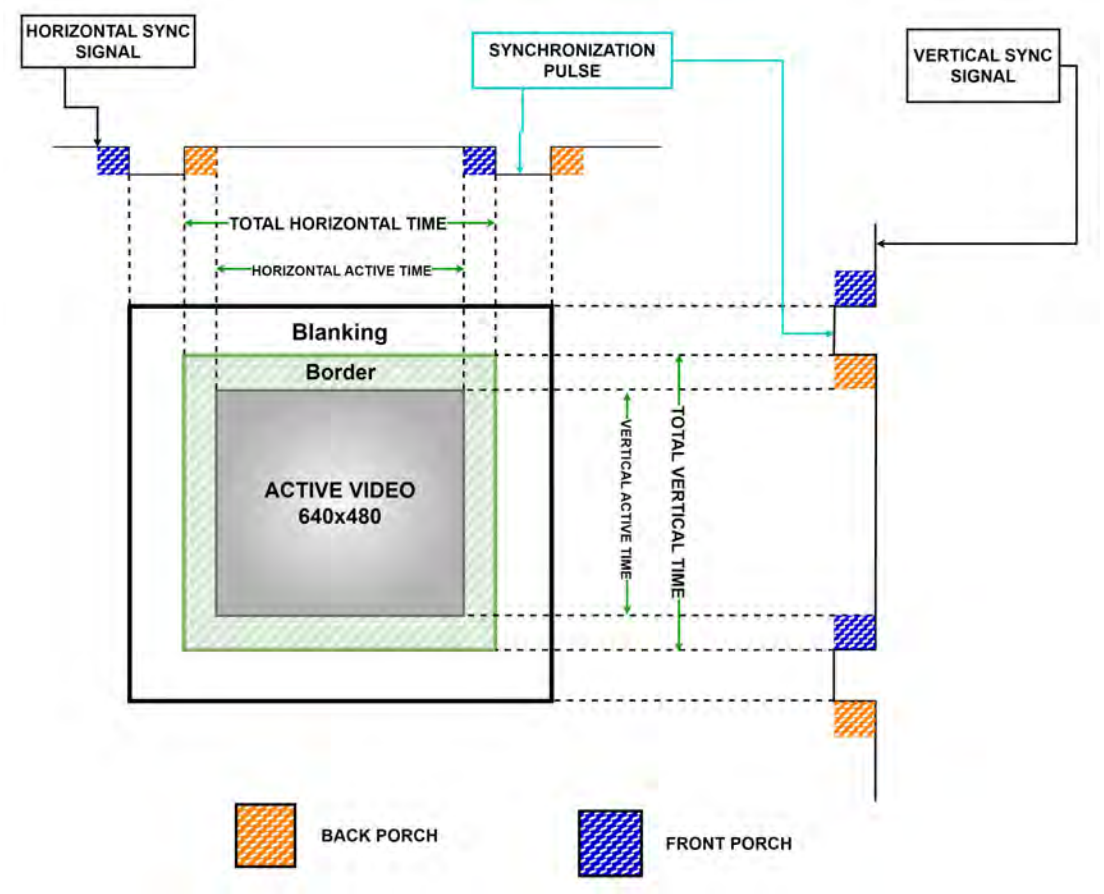


Figure 4.6: VGA Display Timing Configuration

their respective standards. However, a unifying characteristic across these diverse standards is the presence of common parameters, namely Sync pulse, Front porch, and Back porch. However, it's important to note that the values of these parameters vary according to the specific resolution standard being employed.

Presented here is a brief overview of the VGA display operating at a resolution of 640×480@60Hz. The illustration can be found in Figure 4.6, and further detail of the timing specification for the 640×480 resolution, set at a 60Hz frame rate, detailed in Table 4.3. We need to generate these signals(Hsync, Vsync) that will be comprehensively addressed in the forthcoming VGA Controller section.

DESCRIPTION	**HORIZONTAL PIXEL**	**VERTICAL PIXEL**
FRONT PORCH	16	10
SYNC PULSE	96	2
BACK PORCH	48	33
DISPLAY PIXEL	640	480
TOTAL PIXELS	800	525

Table 4.3: VGA Signal Timing According to Pixels

4.3.3 VGA Controller

The VGA interface forms a pixel grid, with 480 rows and 640 pixels per row and operates in a serial manner, sequentially transmitting color information for each pixel. In order to synchronize the supply of VGA visual data (RGB) based on the pixel clock, a VGA controller circuit must create the Hsync and Vsync timing signals.The amount of time that can be used to display one pixel of data is defined by the pixel clock. To calculate the pixel clock frequency, it is necessary to multiply the Total Horizontal Pixels (800) by the Total Vertical Pixels (525) and the Frame Rate (@60Hz).

$$= 800 \times 525 \times 60 = 25MHz(approx)$$

Through calculations, we determine that the VGA controller circuit necessitates a pixel clock frequency of 25MHz. This frequency is responsible for generating Hsync and Vsync signals. Figure 4.7 illustrates two counters: one for tracking pixel count within each line (horizontal counter) and another for monitoring the number of lines within a frame (vertical counter).

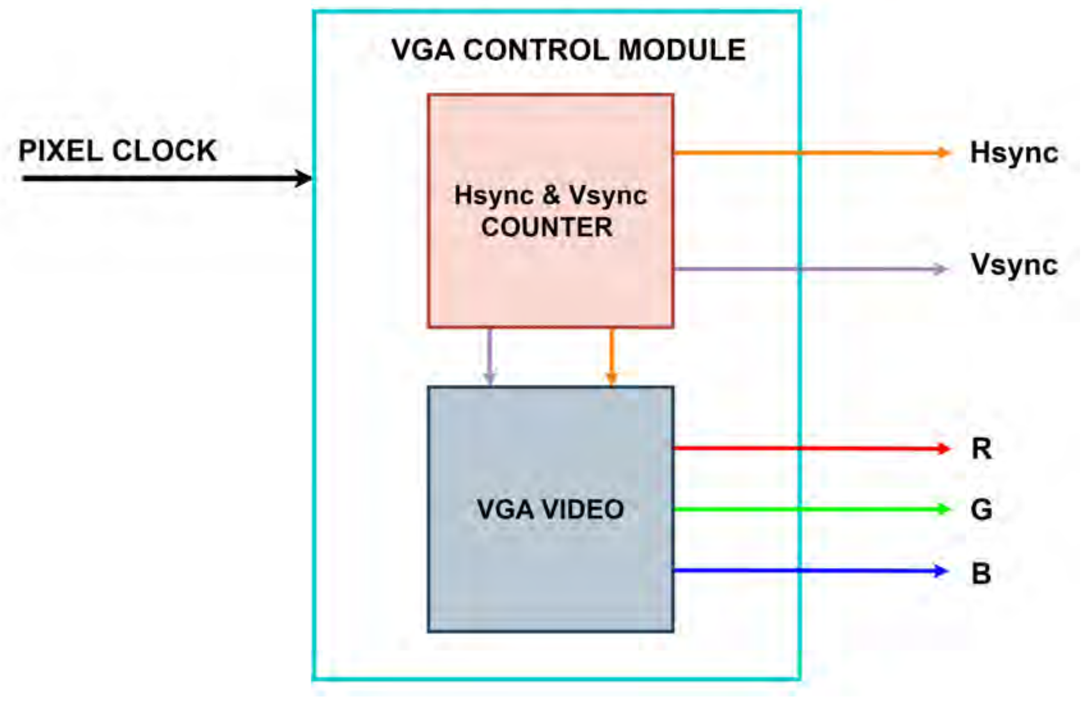


Figure 4.7: VGA Controller RTL Module

Chapter 5

Result and Discussion

5.1 Hardware Setup

Within the hardware setup, a visual representation of the experimental product configuration utilizing the proposed architecture is presented in Figure

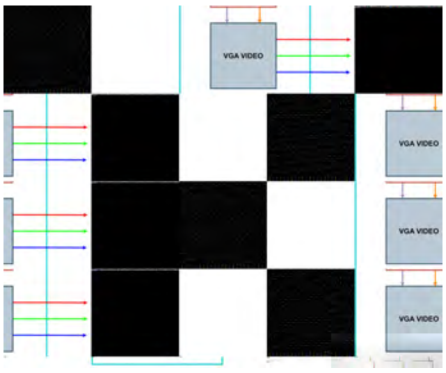


Figure 5.1: Hardware Setup for Experiment

This illustration showcases the interconnection of the OV7670 camera and VGA display with the Zedboard Zynq-7000 FPGA.

This integration is achieved through the utilization of **Pmod** (Peripheral Module) ports. These ports serve as the physical link, connected by standard general-purpose jumper wires, ensuring a structured and effective connection (in figure5.2). The **Pmod** interface, featuring a 2x6 pins connector, offers a structured and organized approach to this interconnection.

Notably, the Zedboard boasts five Pmod connectors, each serving as a General Purpose Input/Output (GPIO) interface, as defined by Digilent Inc[16]. For this setup, two specific connectors, JA1 and JB1, are strategically utilized, marking significant points of interaction and integration. Additionally accepts the video sequences, processes them, and then sends the output to the available VGA port to display.

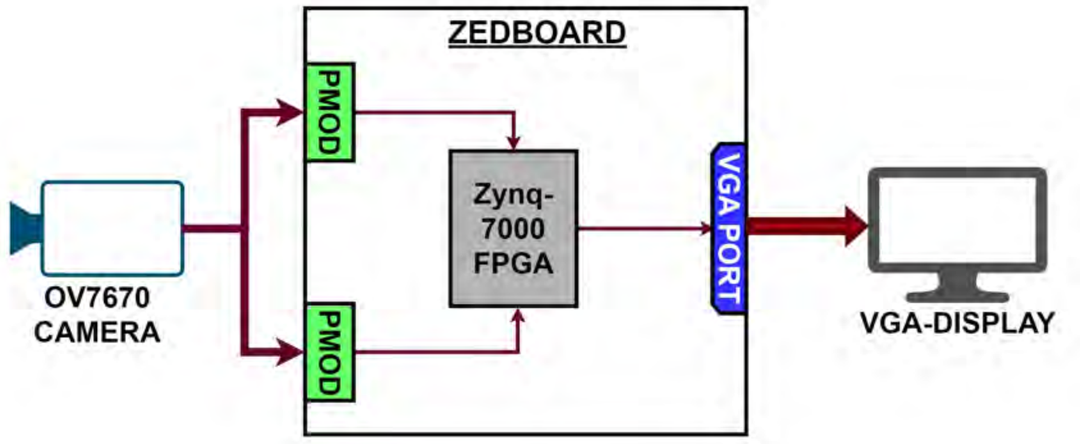


Figure 5.2: Block Diagram of Hardware Setup

5.2 Software Tool

In this thesis, all of the RTL modules presented have been precisely developed using VHDL and Verilog HDL, a Hardware Description Language that forms the foundation of the entire project's design. This choice of language ensures a comprehensive and systematic approach to module creation. Subsequently, the synthesis and implementation phases seamlessly unfold through the adept utilization of the Xilinx VIVADO Design Suite Edition, Version 2019.1. A seamless integration of theory and practice is demonstrated by this advanced software platform, which is a key tool in turning the project's conception into real-world outcomes.

5.3 Top Module

The proposed architecture has been put into practice and meticulously tested to verify its functionality[17]. The top-level block diagram, displayed in Figure 5.3, offers a comprehensive visual representation of the architecture's structure. This diagram encompasses a variety of pivotal modules, including those previously explained, such as the camera controller and VGA controller modules.

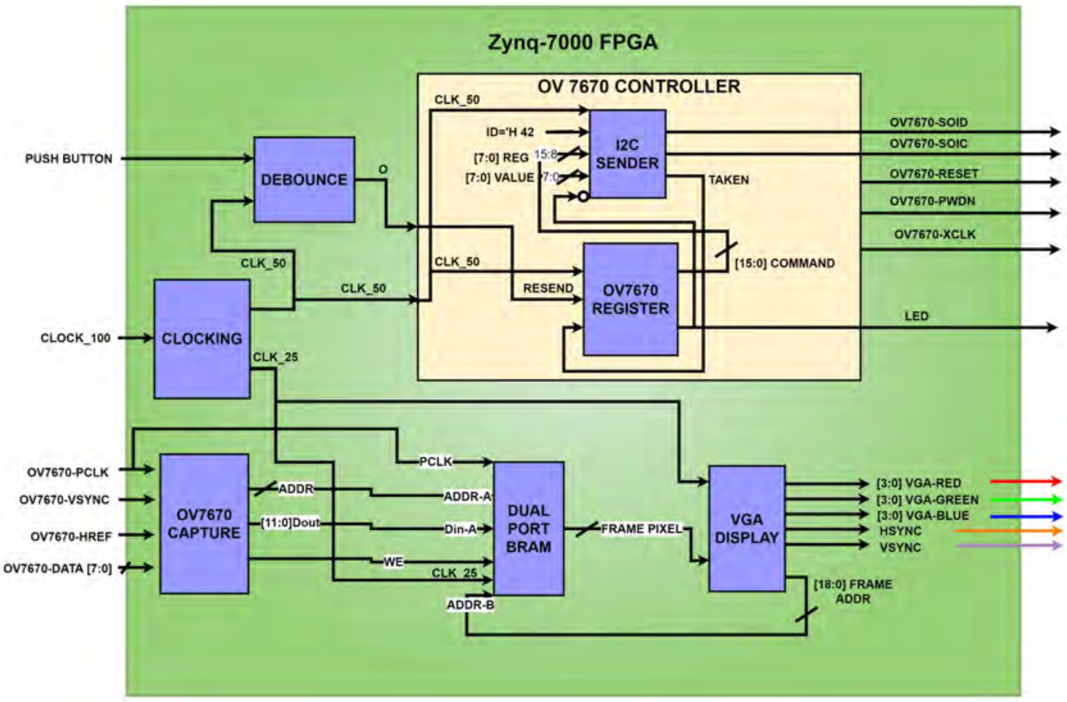


Figure 5.3: Top RTL Module of Project

At the core of this architecture lies the clock controller module, responsible for furnishing the necessary clock frequencies to the entire system, ensuring precise synchronization.

Furthermore, the architecture incorporates two crucial modules: the debounce module and the memory block module. These modules hold significant roles in enhancing the system's overall performance. In the upcoming section, we delve deeper into the specifics of each module, unraveling their distinct functionalities and contributions within the broader framework of the architecture.

5.3.1 Clock Source

Within the Zynq-7000, an internal 100 MHz oscillator serves as the source for the clock input to the PL subsystem. However, our proposed architecture necessitates specific clock frequencies for different modules. The Camera Controller Module, known as the OV7670 Controller, relies on a clock frequency of 50 MHz. Meanwhile, the VGA Controller module operates optimally at 25 MHz. To accommodate these distinct clock requirements, it becomes imperative to divide the system clock into the prescribed frequencies of 25 MHz and 50 MHz (Figure 5.4). This division of clock frequencies ensures seamless synchronization and effective operation of the respective modules within the architecture.

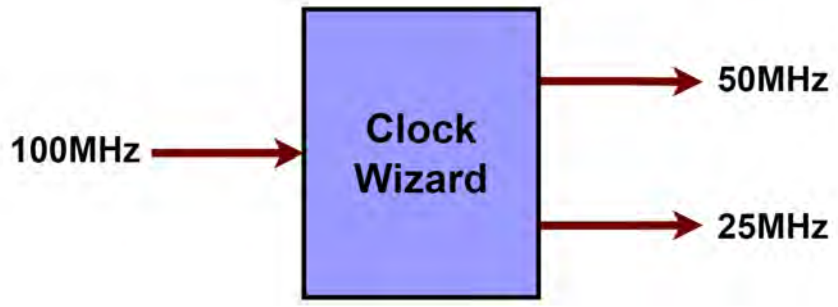


Figure 5.4: Clock Source Module

5.3.2 Memory Block

The camera produces a video stream at a frequency of 24MHz, while the VGA controller operates at 25MHz. This discrepancy in clock frequencies results in a mismatch between the incoming video stream and the intended display. To address this issue, a solution involving a Dual Port Memory block is required. Moreover, the two distinct tasks—writing (referred to as BRAM-PORTA) and reading (referred to as BRAM-PORTB) from the memory—necessitate independent clock signals for optimal performance (figure5.5).

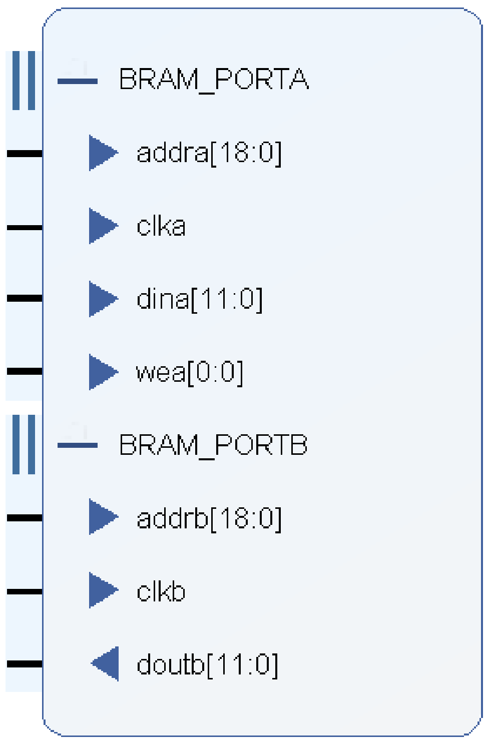


Figure 5.5: Memory Block For Store pixels

The writing process, involving the Block RAM, mandates a clock rate of 24MHz. Conversely, the reading process, which involves retrieving data from the Block RAM for display on the monitor, necessitates a clock rate of 25MHz. The deployment of these discrete clock signals facilitates efficient coordination between the processes, mitigating the mismatch between the video stream's incoming rate and the VGA controller's operation.

The bit count of incoming and outgoing video data varies across different scenarios, yet the buffer's depth remains consistent for a 640×480 resolution.

Upon calculation, the buffer encompasses 307,200 locations—resulting from the multiplication:

$$= 640 \times 480 = 307200$$

Consequently, to accommodate the numerous locations within the Block RAM (BRAM), an address length of 19-bit is necessary for the efficient storage and retrieval of data from these locations.

5.4 RGB Display Module

Within this section, we will elucidate the core functionality of the display module.

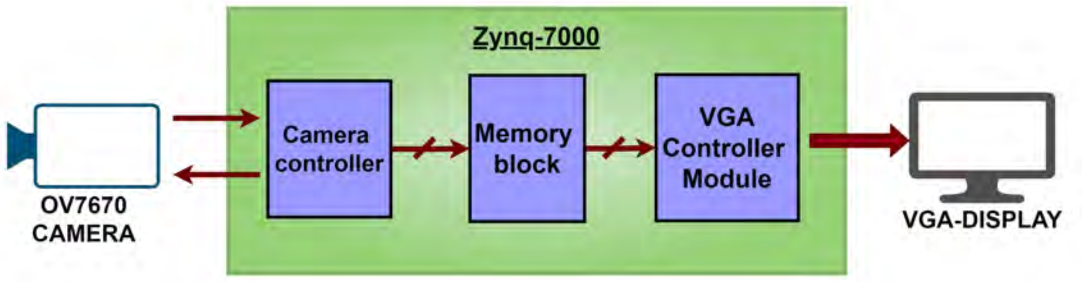


Figure 5.6: RGB Display Module

The camera controller assumes the responsibility of creating 12-bit RGB pixels and then storing them in the Block RAM (BRAM). Simultaneously, the VGA controller obtains the 12-bit pixels from the BRAM. The process of synchronization guarantees a constant transmission of pixels to the display, resulting in a consistent and reliable presentation of images. This module incorporates two continuously active counters, each of which serves a distinct role in addressing read and write operations for the BRAM. These counters play a crucial role in precisely determining the addresses necessary for interacting with the BRAM. One counter manages addresses for write operations, while the other is responsible for addresses related to read operations. Together, they form the backbone of the display module, ensuring the seamless storage and retrieval of data within the BRAM.

The outcomes of the Display module are illustrated in figure 5.7. Please refer to the resource utilization report within the table 5.1 (comparison section) for further insights.

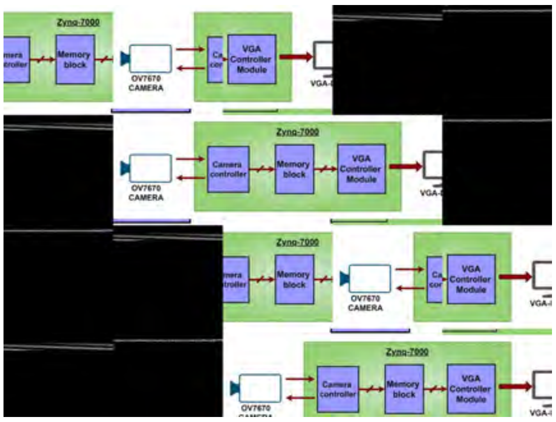


Figure 5.7: RGB Module Output

5.5 RGB to Grayscale Conversion

Within this section, the aim is to transform the incoming RGB image format from the camera into a grayscale image format. This conversion is motivated by the efficient utilization of resources and memory in FPGA, which often presents limitations in this regard (refer to the provided comparison). Converting to grayscale is a strategic move to streamline image data handling and minimize the computational complexities involved in various image processing tasks.

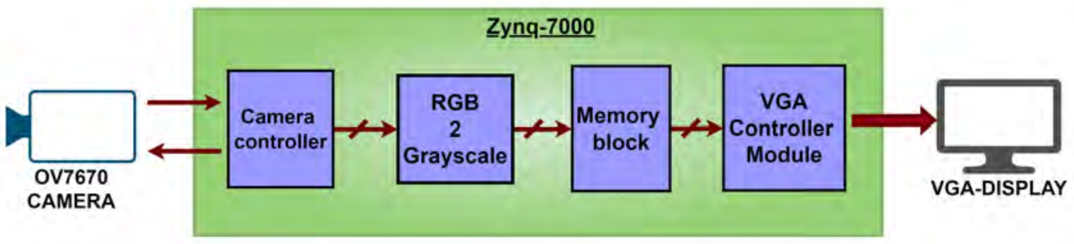


Figure 5.8: RGB to Grayscale Conversion Module

To execute this conversion, an RGB-to-Grayscale module will be seamlessly integrated immediately following the camera controller module. This specialized module is designed to process the RGB data and output grayscale representations. In this configuration, the RGB-to-Grayscale module will store the resulting 4-bit pixel data in the Block RAM (BRAM). This adaptation not only

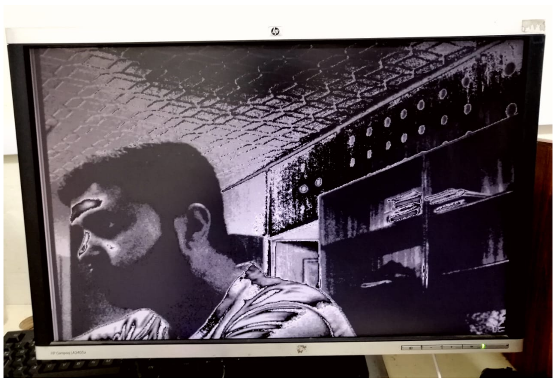


Figure 5.9: Grayscale Module Output

conserves memory but also aligns the image data with the requirements of subsequent image processing algorithms, which are typically more straightforward to implement and comprehend when working with grayscale imagery.

5.5.1 Comparison RGB image vs Grayscale Image

This comparison focuses on resource utilization within an FPGA when processing RGB and Grayscale image formats in table 5.1.

Resource	**Available**	**Color Image Utilize**	**Grayscale Image Utilize**
BRAM	140	104	38
LUT	53200	497	177
FF	106400	257	257
IO	200	33	33
MMCM	4	1	1

Table 5.1: Resource Utilization Report of RGB and Grayscale image

5.6 Gaussian Filter Implementation

After the conversion of images into grayscale, we seamlessly integrate a Gaussian filter block into the processing pipeline. This block is responsible for applying the Gaussian filter to the grayscale images and subsequently storing the filtered results in the Block RAM (BRAM). These filtered images are then displayed on the monitor.

It's essential to note that both the input and output of the Gaussian filter block are configured to use a 4-bit format, ensuring consistency in data representation throughout this image processing workflow.

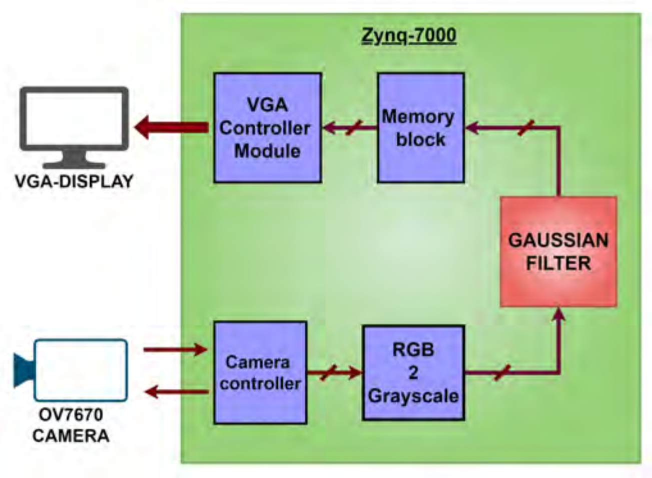


Figure 5.10: RTL Blocks including Gaussian Filter Module

5.7 Median Filter Implementation

In the continuous improvement of our image processing pipeline, we have chosen to eliminate the Gaussian filter block and, in its place, incorporate the Median filter. This transition preserves the uniformity of input and output data, both configured as 4-bit, while effectively addressing the demands of our image processing tasks.

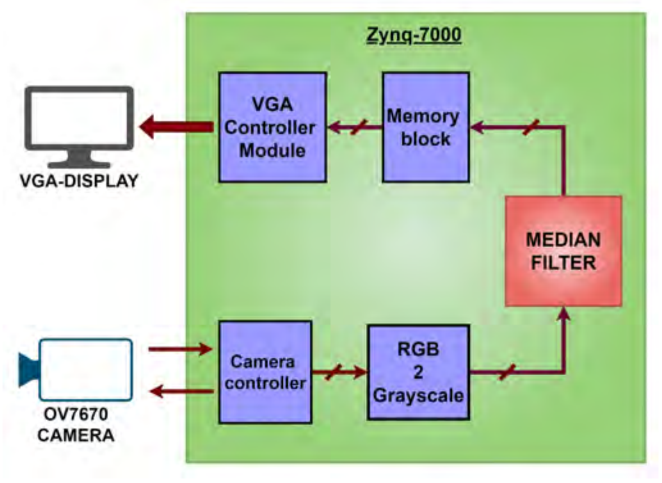


Figure 5.11: RTL Blocks including Median Filter Module

5.8 Results and discussion

In this dedicated section, we undertake a thorough comparison between the implementations of the Gaussian filter and the Median filter. Our evaluation encompasses an analysis of their respective efficiency levels and delves into how these implementations impact the utilization of FPGA resources. To provide a comprehensive view, we have compiled all synthesis results in this section for your reference.

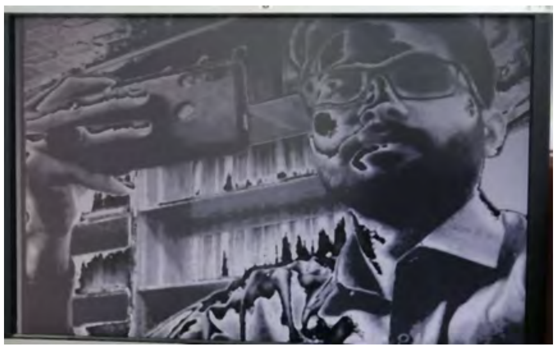


Figure 5.12: Median Filter Output

In the domain of FPGA utilization, a fundamental trade-off emerges encompassing performance, power consumption, and the inherent limitations of available resources [18]. The suggested approach has several notable benefits,

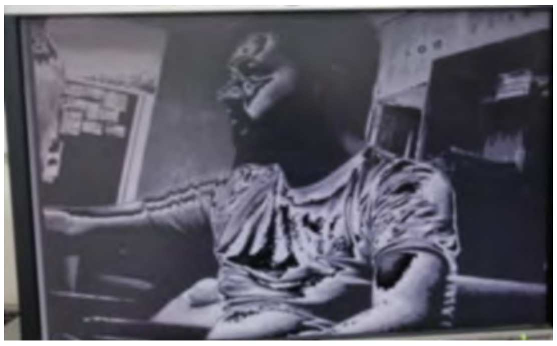


Figure 5.13: Gaussian Filter Output

Resource	Available	Gaussian Filter Utilize	Median Filter Utilize
BRAM	140	38.5	38.5
LUTRAM	17400	721	721
LUT	53200	1419	1489
FF	106400	445	502
I/O	200	34	34
BUFG	32	4	4
MMCM	4	1	1

Table 5.2: Resource Utilization Report Median Filter and Gaussian Filter

such as decreased hardware complexity, cost efficiency, and improved accuracy in comparison to previous methodologies.

The effectiveness of FPGA solutions and the allocation of resources are heavily influenced by several criteria that need careful consideration throughout the programming stage. It is important to acknowledge that the integration of floating-point operations in FPGA implementations often leads to inefficiencies and a significant demand for resources. Therefore, it is recommended to exercise care and explore alternate ways wherever feasible.

5.9 Comparison

In this section, a comparative analysis is conducted between my research and two referenced papers. Specifically, my work is implemented on the Zedboard platform and is juxtaposed against two distinct research endeavors. The first, presented by [19], involves the application of a Gaussian filter on the Virtex-II platform. The second study, conducted by [20], focuses on the implementation of a Median filter on the Zedboard. This comparative examination serves to highlight the unique contributions and differentiating aspects of my research within the context of the methodologies and outcomes presented in the referenced papers.

	My work implementation on Zedboard			**Gaussian Filter Implement on VIRTEX-4**		**Median Filter Implement on Zedboard**	
Resource	**Available**	**Gaussian Filter Utilize**	**Median Filter Utilize**	**Available**	**Utilization**	**Available**	**Utilization**
BRAM	140	38.5	38.5	648	-	140	23
LUTRAM	17400	721	721	5472	1132	999	369
LUT	53200	1419	1489	10944	6451	53200	751
FF	106400	445	502	10944	1132	106400	617
I/O	200	34	34	320	105	200	33
BUFG	32	4	4	-	-	32	4
MMCM	4	1	1	4	1	4	1

Table 5.3: Utilization Report Comparison With Previous Work

Chapter 6

Conclusion and Future Directions

This thesis introduces a hardware architecture that has been specifically developed to facilitate the integration of low-cost digital cameras with FPGA platforms in a highly efficient manner. This architectural design is specially optimized for the efficient execution of real-time video capture and processing applications. In order to get maximum efficiency, a parallel architectural methodology has been used, namely in the optimization of the Image Capture and VGA Signal Generator components. Furthermore, the integration of an RGB-to-Grayscale module aims to optimize the allocation of hardware resources.

In our pursuit of improving real-time video quality, we introduce noise reduction techniques through the implementation of Gaussian and median filters. The camera used in this context operates with an internal image matrix of dimensions 640 × 480 pixels. Furthermore, we have adopted specific image formats like RGB444 to reduce memory utilization.

In the future, we anticipate the integration of a real-time edge detection method for the purpose of object inspection in industrial settings as an extension of our work. Furthermore, our objective is to enhance compatibility with advanced cameras that possess better resolutions such as HD and 4K, as well as improved frame rates. The objective of incorporating more advanced processing algorithms is to produce photos of superior quality, hence enhancing the functionalities of our camera interface architecture based on FPGA technology.

Bibliography

[1] Wang Aihua, Liu Dong, and Wang Zhen. The design of vga data communication based on fpga. In *2011 IEEE International Symposium on IT in Medicine and Education*, volume 1, pages 649–651, 2011. doi: 10.1109/ITiME.2011.6130921.

[2] Pierre Chalimbaud and François Berry. Design of an imaging system based on fpga technology and cmos imager. pages 407 – 411, 01 2005. ISBN 0-7803-8651-5. doi: 10.1109/FPT.2004.1393311.

[3] Oleg Shipitko and Anton Grigoryev. Gaussian filtering for fpga-based@incollection. 06 2018.

[4] Sanjay Singh, Anil Kumar Saini, Ravi Saini, AS Mandal, Chandra Shekhar, and Anil Vohra. Area optimized fpga-based implementation of the sobel compass edge detector. *International Scholarly Research Notices*, 2013, 2013.

[5] Dr. Rajesh Mehra and Rupinder Verma. Area efficient fpga implementation of sobel edge detector for image processing applications. *International Journal of Computer Applications*, 56:7–11, 10 2012. doi: 10.5120/8973-3086.

[6] D. Datta S. Nandy, B. Datta. Implementation sobel edge detector on fpga. *International Journal of Computer Sciences and Engineering*, 6:196–200, 2 2018. ISSN 2347-2693. doi: https://doi.org/10.26438/ijcse/v6i2.196200. URL `https://www.ijcseonline.org/full_paper_view.php?paper_id=1722`.

[7] D. Arbel and N.S. Kopeika. Optical processing systems. In Robert D. Guenther, editor, *Encyclopedia of Modern Optics*, pages 69–77. Else-

vier, Oxford, 2005. ISBN 978-0-12-369395-2. doi: https://doi.org/10.1016/B0-12-369395-0/00729-6. URL `https://www.sciencedirect.com/science/article/pii/B0123693950007296`.

[8] Rohit Verma and J. Ali. A comparative study of various types of image noise and efficient noise removal techniques. *International Journal of Advanced Research in Computer Science and Software Engineering*, 3:617–622, 01 2013.

[9] Yunyun Jiang, Hefei Wang, Yi Cai, and Bo Fu. Salt and pepper noise removal method based on the edge-adaptive total variation model. *Frontiers in Applied Mathematics and Statistics*, 8, 2022. ISSN 2297-4687. doi: 10.3389/fams.2022.918357. URL `https://www.frontiersin.org/articles/10.3389/fams.2022.918357`.

[10] *Sub-window median-like filter in constant time*, volume 11515, 2020. SPIE. doi: 10.1117/12.2566249.

[11] Ming-Yi Tai, Wei-Chih Tu, and Shao-Yi Chien. Vlsi architecture design of layer-based bilateral and median filtering for 4k2k videos at 30fps. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–4, 2017. doi: 10.1109/ISCAS.2017.8050703.

[12] Vineet Kumar, Abhijit Asati, and Anu Gupta. Low-latency median filter core for hardware implementation of 5×5 median filtering. *IET Image Processing*, 11(10):927–934, 2017. doi: https://doi.org/10.1049/iet-ipr.2016.0737. URL `https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-ipr.2016.0737`.

[13] Qingxiong Yang, Narendra Ahuja, and Kar-Han Tan. Constant time median and bilateral filtering. *International Journal of Computer Vision*, 112: 307–318, 2015.

[14] Guohui Wang, Yong Guan, and Yan Zhang. Designing of vga character string display module based on fpga. In *2009 International Symposium on Intelligent Ubiquitous Computing and Education*, pages 499–502, 2009. doi: 10.1109/IUCE.2009.12.

[15] S. Navaneethan, Chenimineni Thanusha, M. Amirdha Varshini, and Ankireddygari Anil. A hardware efficient real-time video processing on fpga with ov 7670 camera interface and vga. In *2022 6th International Conference on Electronics, Communication and Aerospace Technology*, pages 348–352, 2022. doi: 10.1109/ICECA55336.2022.10009129.

[16] INC DIGILENT. Digilent pmod interface specification, 2011.

[17] Miroslav Hagara, Radovan Stojanović, Tomáš Bagala, Peter Kubinec, and Oldřich Ondráček. Grayscale image formats for edge detection and for its fpga implementation. *Microprocessors and Microsystems*, 75:103056, 2020.

[18] P. C. Bhaskar and Vikas K. Jathar. Development of processor engine for fpga based clock gating and performing power analysis. *2016 International Conference on Computing Communication Control and automation (ICCUBEA)*, pages 1–5, 2016. URL `https://api.semanticscholar.org/CorpusID:17333387`.

[19] Jorge Hiraiwa, Enrique Vargas, and Sergio Toral. An fpga based embedded vision system for real-time motion segmentation. In *Proceedings of 17th International Conference on Systems, Signals and Image Processing. Brazil*, 2010.

[20] Chaitannya Supe. Real time implementation of spatial filtering on fpga, 2015.