VLSI Design of Data Filter With Adaptive Neighbour hood Logic for Image Noise Reduction

Dadi Uday Kiran [1], Dadi Shiva Krishna [2], Kolli Ajay Kumar [3], Mr. K. Suresh [4]
[1,2,3] UG Scholar, Dept. of ECE, St. Martin's Engineering College, Secunderabad, Telangana, India-500100
[4]Assistant Professor, Dept. of ECE, St. Martin's Engineering College, Secunderabad, Telangana, India-500100
udaykiran4693@gmail.com

Abstract:

Image noise reduction is a critical aspect of modern digital processing, especially with 40% of digital images suffering from noise-related degradation due to sensor limitations and transmission errors. The demand for efficient image noise reduction techniques is growing at a rate of 20% annually, driven by applications in medical imaging, security, and telecommunications. However, traditional mean filters, which are commonly used for noise reduction, fail to preserve image details, leading to blurred images and loss of important features. Conventional mean filters struggle with edge preservation while reducing noise, making them unsuitable for images with high-detail regions. To address this, a novel Data Filter is proposed that integrates Data Comparator Decision-Based Multiplexer Logics, adaptive neighborhood logic, and multi-level decision blocks to enhance image quality. The proposed system incorporates a three-outcome decision (high, low, median) for effective noise removal while preserving image details. Additionally, sorting 9-pixel neighborhoods in multi-level decision blocks enables precise median filtering, ensuring superior performance in noisy environments.

***Keywords: data filters, VLSI based adaptive filters, dbmf, data comparators, multiplexers, adaptive neighbourhood logic, image noise reduction, Look-up tables, PSNR, MSE*.**

1. INTRODUCTION

The VLSI based adaptive filters are a class of electronic circuits designed to perform adaptive signal processing tasks efficiently and in real-time. These filters are widely used in various applications, including communications, audio processing, image processing, and control systems. Adaptive filters are essential in scenarios where the signal characteristics change over time, and traditional fixed filters may not be effective.

In general, a filter is a system that processes an input signal to obtain a desired output signal by emphasizing or suppressing certain frequency components. Fixed filters have predefined coefficients that remain constant throughout their operation, making them suitable for applications where the signal characteristics do not change significantly. However, in many real-world situations, the environment and input signals may vary, necessitating a dynamically adjustable filter, which is where adaptive filters come into play.

Adaptive filters can self-adjust their coefficients or parameters based on the input data and the desired output, allowing them to continuously adapt to changing signal conditions. They achieve this adaptability using adaptive algorithms, which are mathematical techniques that iteratively update the filter coefficients to minimize the error between the desired and actual output signals.

Adaptive filters are implemented in integrated circuits (ICs) using VLSI technology, which allows for the integration of many transistors and components on a single chip. This integration enables the realization of complex and high-performance adaptive filters that can operate in real-time and handle high data rates.Adaptive filters typically employ various adaptive algorithms such as the LMS (Least Mean Square), NLMS (Normalized Least Mean Square), RLS (Recursive Least Squares), or Kalman filtering algorithms. These algorithms determine how the filter coefficients should be updated at each iteration to minimize the error between the desired and actual outputs. Adaptive algorithms involve a considerable number of multiply and accumulate operations. Adaptive filters utilize dedicated MAC units to efficiently perform these operations in parallel.

Adaptive filters require data storage to hold the input signal, filter coefficients, and intermediate results during the filtering process. Data buffers and control logic ensure that the data is properly managed and processed. Adaptive filters need specialized circuits to update the filter coefficients based on the adaptive algorithms. These circuits implement the mathematical operations required to adjust the filter weights. Adaptive filters must monitor their convergence to ensure that they achieve the desired performance. Convergence monitoring circuits check for the stability and effectiveness of the adaptive algorithm. VLSI technology allows for the implementation of parallel processing elements, enabling simultaneous operations on multiple data samples. This parallelism contributes to real-time processing and faster convergence of adaptive filters.

Motivation

Image noise reduction is essential in many fields, from medical diagnostics to surveillance, where clarity and detail are crucial. In high-stakes applications like medical imaging or satellite data analysis, noise can distort essential features, leading to incorrect interpretations. Current filtering techniques like mean filters are not adequate for such tasks as they reduce noise but fail to retain fine image details. As images become more complex and noise issues increase with higher resolution sensors, an advanced solution for noise reduction is required.

An adaptive noise reduction system that can distinguish between noise and important details is needed to improve the clarity and accuracy of images. The proposed Data Filter using adaptive neighborhood logic and multi-level decision blocks addresses these challenges by dynamically adjusting filtering behavior based on the local neighborhood of each pixel. This method has the potential to improve noise reduction in images while retaining critical details, making it valuable for applications in healthcare, security, and advanced digital imaging systems.

Problem Statement

The rapid growth of technology and the increasing demand for real-time signal processing have motivated the development of adaptive filters in VLSI-based implementations. The goal of this project is to design and implement a high-performance Adaptive filter system capable of efficiently processing dynamic signals in real-time across various applications. The adaptive filter should continuously adjust its parameters to adapt to changing signal characteristics and achieve optimal performance. Design and implement a Adaptive filter system that addresses the following key challenges:

The primary issue with traditional mean filters lies in their inability to preserve important image details while removing noise. The uniform

application of noise reduction across the image leads to the degradation of edges and other high-frequency components, which are critical for interpreting the image correctly. In applications like medical imaging, this can result in the loss of vital diagnostic information. The lack of adaptability in existing noise reduction methods further complicates the problem, as different areas of an image may require different levels of filtering based on the type and intensity of noise present.

Manual tuning of noise reduction filters also presents significant challenges. It is a time-consuming process that often requires expert knowledge to achieve acceptable results. Additionally, these manual methods are not easily transferable between different types of images or noise conditions, limiting their usefulness in real-world applications where images and noise patterns vary widely. The need for a more adaptive, automated solution is evident to address these challenges.

Applications

Noise Cancellation in Audio and Speech Processing: Adaptive filters can be employed to remove background noise and interference from audio signals, enhancing the clarity of voice communications, audio recordings, and speech recognition systems.

Channel Equalization in Communication Systems: Adaptive filters can compensate for channel distortions in communication systems, such as wireless channels or fiber-optic links. By equalizing the channel, the system can recover transmitted data more accurately, leading to improved data rates and reliability.

Echo Cancellation in Telecommunications: Adaptive filters can eliminate acoustic echoes in telephony systems, preventing distracting feedback and enhancing the overall call quality for both wired and wireless communications.

Biomedical Signal Processing: In biomedical applications, adaptive filters can be used to clean up noisy signals from medical sensors, such as electrocardiograms (ECG) or electroencephalograms (EEG), improving the accuracy of medical diagnoses and monitoring.

Adaptive Noise Filtering in Image and Video Processing: Adaptive filters can be employed to denoise images and videos, reducing artifacts caused by compression or sensor noise and enhancing image quality.

Control Systems and System Identification: In control systems, adaptive filters can be used for identifying the dynamics of unknown systems, allowing the controller to adapt to changes and uncertainties in the plant.

Acoustic Echo Suppression in Hands-Free Systems: Adaptive filters are useful in hands-free communication systems (e.g., in cars) to suppress acoustic echoes arising from the speaker being too close to the microphone.

Active Noise Control (ANC): ANC systems utilize adaptive filters to generate anti-noise signals that cancel out external noise, commonly used in noise-canceling headphones and environments where noise reduction is critical.

Radar Signal Processing: Adaptive filters are employed in radar systems to filter out clutter and interference, enabling better detection and tracking of radar targets.

Sonar Signal Processing: In underwater applications, adaptive filters can be used to remove noise and echoes from sonar signals, improving target detection and underwater navigation.

Smart Antenna Systems: Adaptive filters are used in smart antenna systems to adaptively adjust antenna patterns and null out interference, leading to enhanced communication and increased capacity in wireless networks.

Real-Time Audio and Music Processing: Adaptive filters find applications in real-time audio processing, such as real-time equalization, dynamic range compression, and feedback cancellation in audio systems.

Environmental Noise Control in Buildings and Offices: Adaptive filters can be used to reduce ambient noise levels in buildings and offices, providing a more comfortable and productive environment.

Vibration Control in Mechanical Systems: Adaptive filters can be employed to reduce vibrations and noise in mechanical systems, such as in active vibration control in automotive suspensions or in industrial machines.

Adaptive Beamforming: In antenna arrays and radar systems, adaptive filters are used for adaptive beamforming to improve the spatial selectivity and directionality of the receiving or transmitting antennas.

2. LITERATURE SURVEY

The kernal generator is a combinational circuit that is used in the process of determining if the binary number that is present at one input is larger or less than the binary number that is present at another input. Kernal generators are critical components, and an XOR gate may serve this purpose. The Kernal generators is broken down into two distinct categories (a). Magnitude Kernal generator (b). Adaptive filter. In the first method, just the magnitude of the two binary integers is compared, but in the second method, both the larger and the lesser data are provided. The Magnitude kernal generator has two outputs, one of which indicates if the first input is larger than the second input, while the other output indicates the opposite. In contrast, a adaptive filter is also often called a two-cell kernal generator since it compares two words, X and Y, and returns a number that is either higher or lower depending on which word is being compared. A compact, high-quality, cost-effective, high-performance, and low power kernal generator plays a vital role in almost all hardware kernal generators. The purpose of this study is to investigate the characteristics of certain kernal generator circuits that, in comparison to those of existing circuits, provide superior performance.

An Adaptive Rank Order filter (AROF) with VLSI implementation has been developed by in [1] in order to get rid of impulsive noise and pipelining with parallel processing in order to get the filtering process done more quickly. This filter was developed specifically to cut down on impulsive noise. In contrast to the Decision Tree Based De-noising Method, the Decision Rank Order Filter (DROF) has the advantage of occupying less space and also having a design that is straightforward and simple to comprehend (DTBDM). These are the benefits of using the DROF. One of the drawbacks of VLSI DTBDM is that it requires an excessive number of designs for the detection of noise and the reconstruction of noisy pixel.

In [2] investigated the possibility of designing a kernal generator that is both high speed and low power. This is possible because the kernal generator runs using just 1volt of power, has a shorter propagation delay, and its architecture comprises a two-stage CMOS op-amp circuit. This study involves the development of a kernal generator using a cadence tool that has a technology of 0.18 micrometers.

In [3] presented a unique category of kernal generators, and the simulation of these circuits in LTspice-IV made use of PTM 45nm technology. The supply voltage for these circuits was set at 1 volt DC. It runs at a greater speed and provides a more stable output in comparison to 90nm and 180nm kernal generators, unlike the static and dynamic features of all of these kernal generators, which are studied and compared below.

In [4] produced a variety of different designs in order to lessen the amount of space and power that was used. This was done since even a little reduction in the amount of space and power that a circuit uses

may result in a significant overall cost reduction. When compared to an automatically produced design, full-custom design results in around a 50% reduction in area and a 35% reduction in power usage, according to the study.

In their work on improved shear sorting, In [5] established a parallel design in addition to a pipelined architecture. This approach included a space-saving adaptive filter for the classification of 9 different components. The area optimized two cell sorter is the fundamental component of the processing system. A three cell sorter is formed by the combination of a group of three two cell sorters, and this sorter employs a compare and swap strategy in order to arrange the data sequence.

In [6] created a set of VLSI algorithms and implementation designs for a family of nonlinear filters. Keeping chan was involved in this research. All of the functions that were previously specified by stack filters are included in the class of filters; the rank-order filter and the wiener filter are special examples of this class. Utilizing a single binary processing circuit in a recursive manner k times may allow the function of a stack filter to be accomplished. A unique Borrow Look Ahead Logic based Kernal generator (BLAC) was presented in [7] and its output was implemented. An architecture of modified shear sorting that is pipelined as well as parallel.

In [8] authors presented a VLSI design for decision-based asymmetrical trimmed midpoint filter that was based on a finite state machine. A innovative approach for locating the median value of a set was suggested, and it included employing modified selection sort. A brand new 8-bit adaptive filter that made use of carry select logic was presented in [9]. The bit-serial architecture that has been presented is one that lends itself extremely well to VLSI implementation.

We describe a unit that is capable of doing continuous-time hybrid approximation computing. In this model, both analog and digital signals are treated as functions of continuous time. In every analog and mixed-signal block, you'll see the utilization of digitally aided calibration. Because of technological scalability and significant usage of class-AB analog blocks, our device is capable of arbitrary nonlinearities and achieves power dissipation that is 16 times lower than that of the previous art. Depending on the particulars of the equation, the apparatus is typically capable of achieving a computational accuracy of about 0.5% to 5% RMS, solution times ranging from a fraction of one microsecond to several hundred microseconds, and total computational energy ranging from a fraction of one nanojoule to hundreds of nanojoules. In addition, the device can typically achieve these results with a range of energy that spans from a fraction of one nanojoule to several hundred nanojoules.

In the SC paradigm [11], logical processing is carried out on bit streams that have been randomly generated. In previous research, streams were formed using linear feedback shift registers. These registers were a substantial contributor to the overall cost of the hardware and used up a considerable amount of power. In this piece of research, a novel method for encoding signal values is presented: the calculation is carried out on analog periodic pulse signals. Adjusting the frequency and duty cycles of pulse width modulated (PWM) signals allows for the generation of time-encoded signals that correspond to certain values. This is accomplished via the use of pulse width modulation.

The capacity to gracefully [12] noise and the skew tolerance are the key benefits that come with using stochastic computing. Another advantage is the extremely basic hardware requirements that are needed to accomplish sophisticated operations. However, this paradigm's very significant latency presents a possible obstacle to its wider use, especially in circumstances that call for a high level of precision. This research presents a novel method that is high-speed while still being accurate for the implementation of stochastic circuits. This method makes use of synchronized analog pulses as a new method for encoding correlated stochastic numbers.

One of the most efficient oblivious parallel sorting [13] algorithms that has been discovered to date is called bitonic sort. The great degree of modularity that bitonic sort has enables it to be mapped to a variety of different connectivity networks. This article presents a mapping of the bitonic sort algorithm to the chained-cubic tree (CCT) connectivity network. The BSCCT algorithm's performance in terms of calculation time, communication cost, message latency, and key comparisons is evaluated with the assistance of simulation, which is developed and employed in the process. When 1024 processors were utilized to sort 32 million keys, simulation results indicated that the BSCCT algorithm achieved a speedup that was nearly 12-fold greater than a bitonic sort on a single CPU. This was the case when comparing the two methods for sorting data.

Sorting that is both high-throughput and low-latency [14] is a crucial need in many applications because of the vast volumes of data that they deal with. This study demonstrates effective strategies for the construction of sorting units that have a high throughput and a low latency. The modular design principles that we use for our sorting architectures allow us to develop larger sorting units by constructing them hierarchically from smaller building pieces. The sorting units have been optimized for circumstances in which only the M biggest values from N inputs are needed. This is a condition that commonly happens in many applications for scientific computing, data mining, network processing, digital signal processing, and high-energy physics. Since this is a situation that frequently arises, the sorting units have been optimized. This circumstance often arises in applications of this kind because of how they are designed. We make use of the approaches that we have given in order to develop sorting units that are pipelined, parameterized, and modular.

We provide two unique concurrent error-detection (CED) [15] algorithms for a class of sorting networks, including odd-even transposition, bitonic, and perfect shuffle sorting networks, in this summary. These strategies are described in more detail below. These strategies are designed to identify errors in the networks simultaneously. In order to do an analysis of the fault coverage, a probabilistic approach has been created, and the overhead of the hardware has been assessed. The first thing that we do is suggest a CED system that can identify any and all mistakes that are brought on by individual faults in a concurrent checking sorting network. This strategy is the first one that has been developed that uses a significant amount less hardware overhead than duplication while maintaining the same level of performance.

A cutting-edge hierarchical sorting network that can tolerate defects is given [16]. The area-time cost of the bitonic sort and the odd-even transposition sort are both taken into account in the design, which results in a balanced solution. It decreases the wiring complexity of the bitonic sorter in VLSI or WSI (wafer scale integration) implementation, and it consumes less hardware than a single-level odd-even transposition sorter. Because of the very regular nature of the hierarchical sorting network, it is not difficult to implement redundancy at each and every level of the hierarchy. In the process of implementing hierarchical reconfiguration, the faulty cells at the lowest level are initially replaced with spare cells. If there is not enough redundancy at the level that is currently being worked on, the process moves up to the next higher level to execute reconfiguration.

Literature survey on adaptive filters

When using spatial domain filtering, the values of the image's pixels are changed in such a way as to immediately accomplish the desired effect. The mean, order statistics, and adaptive filters are the several types of spatial domain filters that are available [17]. Image processing, satellite and remote sensing, medical and microscopic imaging, geographic image surveillance, and seismographic analysis

are just few of the many fields that benefit from the use of image filters. In the article [18], the authors presented a novel technique of wiener filter by sharing Common Boolean Logic (CBL), which substitutes the RCA-XOR gate and inverter that are utilized for the sum generation. Carry generation makes use of both the AND gate and the OR gate [19]. Both the needed sum and carry output are created by the multiplexer, and both are dependent on the precise carry input that is provided to it. The partial sharing of the circuit as well as the logical simplification both contribute to the suggested efficient design. This design results in a reduction in the total number of transistors while simultaneously reducing the amount of power that is lost. In the article [20], the authors presented the efficient Frequency Domain Denoising Filters by making use of a newly proposed new form of basic full adder. The authors of [21] explored the development of a Wiener filter by making use of fundamental logic gates. The fact that employing the fundamental gates results in no loss of power under optimal circumstances is the primary benefit of doing so. Modifications to the design are made in the fundamental gates in order to cut down on the amount of trash bits and constant inputs.

The authors of [22] presented a skilled and efficient Wiener filter that substitutes a BEC with standard Boolean logic in conventional CSLA. Specifically, the authors of [22] referred to this as a conventional CSLA. Through the use of the Common Boolean logic term [23], this study makes use of an efficient CSLA. In order to generate carry and sum signals, an OR gate and an inverter are used. In order to pick the desired output depending on the proper carry, a multiplexer is used. Increases in area result in reductions in both power and delay. A low-cost image denoising standard Wiener filter (SMF) approach employing CSLA without multiplexers was suggested by the authors in [24]. First, a carry input zero operation is carried out, and then a BEC adder operation is carried out. The BEC adder has been incorporated into the circuit in such a way that it takes the place of the conventional method's final MUX arrangement. When the MUX stage is replaced, there will be less wasted space and reduced latency, leading to much higher execution for the adder. The authors of [25] constructed a parallel pipeline median finder by making use of an extremely low power design in the near threshold area. The functioning of a sub threshold is quite similar to the operation of a minimum energy. The energy delay modeling framework that emerges in the mild, moderate, and strong inversion areas is the focus of this body of work. The functioning of the system below the point of minimal energy is also examined [26]. The findings of the trial indicate that there is a 20% boost in energy, which ultimately results in improved performance. The findings of our estimate method are presented using this notion, which allows for the comparison of adders based on the energy delay characteristics of each adder. The authors of [27] talk about the area, power, and delay performances of hybrid sorting-based dynamic wiener filtering (DMF) by using several CMOS logic designs. A fresh hybrid approach has been suggested for the creation of complete adders. Despite the fact that complete adders are employed in tree-structured arithmetic circuits [28], novel hybrid logic is used for simulation, and this logic is then used in the application environment. By following this approach, one may realize both a complete swing and a balanced output. This strategy results in a layout that makes effective use of the available space.

The authors of [29] devised a low power Wiener filter for the purpose of impulse noise reduction that was based on the fluctuation in supply voltage. The selection of the supply voltage was made based on the input vector pattern. Using this strategy will result in a significant reduction in the amount of electricity used [30]. This approach is discussed in relation to the 32-bit RCA prototype in this article. When compared to the power requirements of traditional RCA, the simulation findings reveal that there is a 29% reduction in power demand. Adaptive Wiener filters (AMF) using thresholding techniques were created by the authors of [31] for use in low power applications. Finding the best structures cannot be accomplished by designing them with a single supply voltage or by comparing them based on their gate count. Neither of these approaches is an appropriate strategy. Therefore, in order to achieve a lower amount of energy, high-performance structures have to be linked with the scaling of the supplementary voltage. These technological advancements provide new perspectives on the conventional design for low-power operation.

3. PROPOSED METHODOLOGY

The edge-preserving filtering technique known as guided image filtering, which is frequently shortened as GIF, is used extensively in both computer vision and digital video processing. This technique was developed very recently. The acronym for this technology is well understood by the general public. Image smoothing and enhancement, noise reduction, and the extraction of image structure are some of the applications that it may be used for. Relighting and tone mapping are two more areas of use. When compared to the edge-preserving bilateral filter [1] and the domain transform filter [2], the behavior of GIF images that are close to edges is much more desired. This is largely owing to the fact that GIF is more successful than other methods in avoiding gradient reversal artifacts, as explained in the major reference on GIF [3].

Proposed Methodology

Within this segment, a complete evaluation of the proposed filtering technique is provided. Presented in Figure 1 is the proposed operational method for the VLSI-primarily based DBMF platform.

Step 1: Plug in the Noisy Image: In the Matlab surroundings, begin with an image that is tormented by noise. Pixel values make up this noisy image, wherein each pixel represents a numerical price that encodes the color or intensity of a particular place in the image. This image is characterised with the aid of its inherent noise.

Step 2: Read and keep the pixel values. The pixel values from the noisy image are read by Matlab, and then they may be either saved in reminiscence or stored to a record. To proceed with this step, the image should be transformed right into a statistics format that Matlab is able to work with. This format is commonly a matrix or an array.

Step 3: DBMF Module based on VLSI generation: To start processing the pixel move, the VLSI-based totally DBMF module receives it as input after which begins processing it. This specific module is designed using VLSI technology, which suggests that it is applied as a hardware element this is capable of method data.

Step 4: Multi-Level Data Comparator Decision-Based Multiplexer Logic: Within the VLSI-based DBMF module, the pixel movement undergoes multi-level data comparator choice-primarily based multiplexer common sense. The choices regarding which pixel values must be saved and which should be filtered out are made by using this good judgment. These selections are based totally on sure standards that are specific to the situation. The criteria are determined by the noise reduction algorithm this is being applied.

Step 5: Adaptive Neighborhood Logics: This step uses adaptive community logics further to the selection-primarily based multiplexer common sense that is utilized by the VLSI module. The reasoning at the back of these logics maximum likely entails reading the pixels that surround each pixel inside the circulation to arrive at well-informed choices regarding noise discount. Therefore, due to the fact those logics are adaptive, they could modify their behavior according with the traits of the image and the noise.

Step 6: Row-Wise and Column-Wise Sorting: The DBMF module techniques the pixel moves in both row-clever and column-smart instructions. This is the third step inside the procedure. This indicates that it examines pixels in each a horizontal and vertical direction to acquire information approximately the context and relationships between neighboring pixels, that's essential for efficient noise reduction.

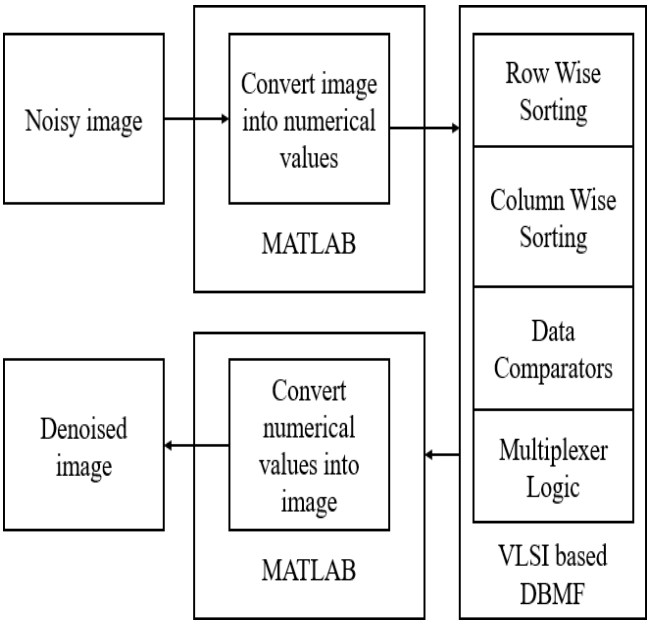

Figure 3.1: Proposed Image-Denoising diagram.

Step 7: Denoised Pixel Stream: Once the logic of the Step 4: Denoised Pixel Stream: Once the common sense of the DBMF module has been carried out to every pixel inside the movement, a denoised pixel movement ought to be obtained. This movement has been up to date to include pixel values that have been altered so that you can reduce noise while preserving the integrity of the favored image details.

Step 8: The denoised pixel stream is then carried out again to the Matlab environment, that is the fifth step in the manner of producing the denoised image. In Matlab, those numerical values are converted into a image format after being taken with the aid of the program. In this technique, the pixel values are mapped to coloration or depth values, and denoised image is created so that it is based totally at the denoised pixel stream. When compared to the authentic noisy image, this denoised image has substantially less noise, whilst on the same time retaining the important characteristics and information of the illustration.

Decision Based Multiplexer Function (DBMF) operation

The DBMF is a nonlinear filtering method that replaces the value of the current pixel with the median value that is determined from the values of the pixels that are in the vicinity of the specific pixel that is now being filtered. This value is determined from the values of the pixels in the neighbourhood of the current pixel. This value is the result of computing all the pixel values that are near the current pixel. Consider input image has the dimensions M by N and is of the form x. We will also assume that the value $x_{i,j}$ signifies the pixel in the image that has the pixel location i, j, where $i, jA = 1, \ldots, M1, \ldots, N$. Following this, we will provide a concise summary of the MF filter. We will be using $x_{i,j}$ to represent a window with a size of 2W+12W+1 that is cantered at i, j. The W is understood to mean:

$$x_{i,j}(W) = a, b: a - i \leq W, b - j \leq W, a, b \in A \quad (1)$$

The $x_{i,j}(W)$ is that it is the median value of all the pixel values, which have been ordered in ascending order. So, the DBMF operating concept might be stated as:

$$x_{i,j} = x_{i,j}^{med}(W) \quad (2)$$

An image includes one of three distinct sorts of pixels: light spot pixels, noise pixels, or irrelevant pixel types. Light spot pixels are the most common form of pixel. Most of the pixels that make up the image have pixel values that are either 0 or 255. This is because the size of the DBMF does not fluctuate. When the size of the DBMF is large, it not only has the capability to effectively eradicate the noise that is both around and inside the light spot.

Data Comparator Decision-Based Multiplexer Logics

A decision-based multiplexer logic is used to choose the highest (H) and lowest (L) values from two input data sources, which are labeled as A and B, respectively. Figure 3.2 provides a description of the functioning of this logic. In order to explain how this decision-based multiplexer works, the following processes are broken down into their respective categories:

Step 1: Condition Verification: In this step, the comparison A < B is made. If this condition is true (i.e., A is smaller than B), it sets the selection line of the multiplexer to one (1). Otherwise, if the condition is false (A is not smaller than B), the selection line becomes zero (0).

Step 2: Selection of Highest Value (H): The input A is connected to Data-input-0, and input B is connected to Data-input-1 of a 2-to-1 multiplexer. If A is smaller than B (as determined in Step 1), the selection line is set to one (1), and the multiplexer routes input-B to the output, generating H as the result. If A is larger than B (based on the condition in Step 1), the selection line is set to zero (0), and the multiplexer routes input-A to the output, again generating H as the result.

Step 3: Selection of Lowest Value (L): Like Step 2, in this step, the input B is connected to Data-input-0, and input A is connected to Data-input-1 of another 2-to-1 multiplexer. If A is smaller than B (as determined in Step 1), the selection line is set to one (1), and the multiplexer routes input-A to the output, generating L as the result. If A is larger than B (based on the condition in Step 1), the selection line is set to zero (0), and the multiplexer routes input-B to the output, again generating L as the result.

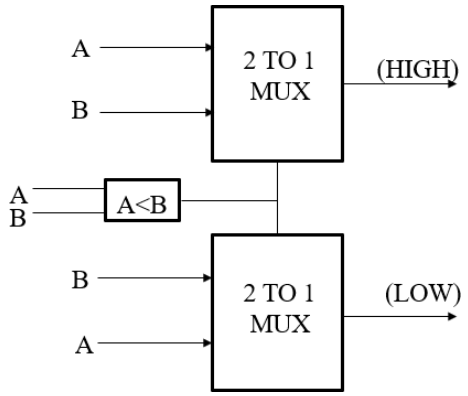

Figure 3.2: Proposed decision-based multiplexer logic.

Figure 2 shows the proposed adaptive neighborhood logic block diagram. It contains the three decision-based multiplexer logic modules, which are used to select high (H), low (L), and medium (M) values. Initially,and L1. Then, the L1 and C are applied to second decision-based module, which generates the H2, and L2.. Here, H3 considered as high value, L3 considered as median value, and L2 considered as low value.

VLSI based DBMF

Figure 3.3 shows the proposed DBMF implementation using VLSI environment, which contains multiple number of adaptive neighborhood, and decision blocks. These logics performs the row wise, column wise sorting and generates the final median outcome from nine-pixel combination. Here, the pixel values P0, P1, and P2

are applied to adaptive neighborhood block, which generates the H1, M1, L1 outcomes. Then, the pixel values P3, P4, and P5 are applied to adaptive neighborhood block, which generates the H2, M2, L2 outcomes.

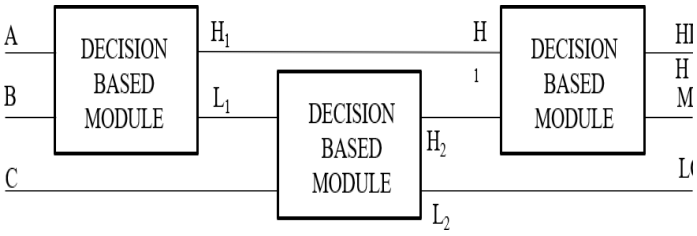

Figure 3.3: Proposed adaptive neighborhood logic.

In addition, the pixel values P6, P7, and P8 are applied to adaptive neighborhood block, which generates H3, M3, L3 outcomes. This process is called row wise processing or sorting. In addition, row wise sorted middle values (M1, M2, M3) are applied to adaptive neighborhood block, which generates the M4 outcome. Then, decision block1 considers H1, H2, which generates the L5 outcome as in Figure 3.4. The L5 and H3 are applied to the next decision block, which generates the L6 outcome. Similarly, decision block3 considers L2, L3, which generates the H5 outcome. The H5 and L1 are applied to the next decision block, which generates the H6 outcome. Finally, L6, M4, and H6 are applied to adaptive neighborhood block, which generates the final median outcome (M_{out}).

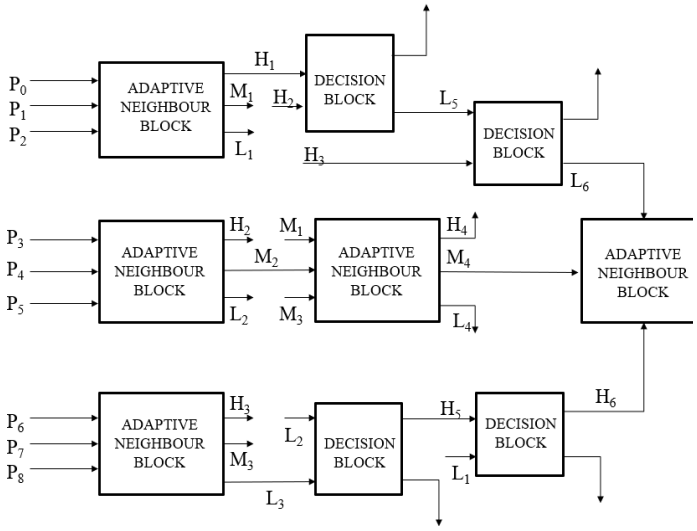

Figure 3.4: Proposed multi level decision blocks communication process.

Advantages

The advantages of this work is to propose a VLSI implementation of a DBMF for image scaling applications. The main goal is to overcome the limitations of traditional scaling methods and address issues such as blurring, ringing, and aliasing, particularly for significant scaling factors. The specific objectives of this research are as follows:

- **Improved Image Scaling Quality**: Develop an adaptive image scaling technique that can adjust the size of images while preserving their visual quality. The DBMF should adaptively estimate the optimal scaling kernel for each image region based on local characteristics, such as edges and textures, resulting in superior image quality compared to traditional scaling methods.
- **Reduction of Undesirable Artifacts**: Minimize undesirable artifacts introduced by traditional scaling methods, such as blurring, ringing, and aliasing, by utilizing the adaptive nature of the Wiener filter. The DBMF should demonstrate improved noise reduction and maintain image sharpness and fidelity during scaling.
- **Real-Time Processing of High-Resolution Images**: Design a VLSI-based DBMF architecture that efficiently computes the adaptive filter coefficients for each image region, enabling real-time processing of high-resolution images. The proposed hardware implementation should be capable of handling large-scale images with low latency.
- **Comparison with State-of-the-Art Techniques**: Conduct a comprehensive evaluation of the proposed DBMF technique against state-of-the-art image scaling methods. Perform objective and subjective evaluations to assess the performance of the DBMF in terms of decreased noise, latency, and power consumption.
- **Hardware Resource Optimization**: Optimize the hardware metrics of the VLSI-based DBMF implementation, such as look-up tables (LUTs), to ensure efficient resource utilization and reduce hardware complexity.
- **Generalization to Different Image Types and Scale Factors**: Validate the effectiveness of the DBMF across a wide range of images and scale factors. Ensure that the adaptive filtering approach is robust and applicable to various image processing applications.

4. EXPERIMENTAL ANALYSIS

Within this section, simulation analysis as well as subjective and objective results are presented. A representation of the simulation process is shown in Figure 4.1. This process includes the input signals P0, P1, P2, P3, P4, P5, P6, P7, and P8. The DBMF process is responsible for producing the median result, which is known as the noise-free median and does not contain any errors.

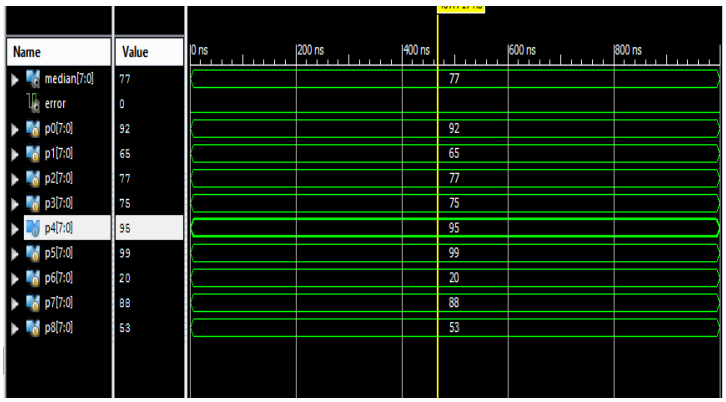

Figure 4.1: Proposed simulation outcome.

A comparison of three distinct scenarios is shown in Figure 4.2, which depicts the performance of noise removal in an image. The term "existing outcome" refers to the accomplishment of removing noise through the application of a method or technique that already exists. A comparison with the method that has been proposed can be made using it as a benchmark. Through the implementation of the DBMF suggested method for noise removal, the proposed outcome is obtained.

The area utilization for the proposed method is broken down in great detail in Figure 3, which includes the breakdown. There are 78,600 Look-Up Tables (LUTs) available, and it is specified that 393 of them are used out of the total number of LUTs that are available. In addition to this, it discusses the utilization of Input/Output Blocks

(IoBs), stating that out of the 250 IoBs that are available, 81 of them are being utilized. Through the use of this figure, we hope to demonstrate how effectively the proposed method makes use of these crucial components for the purpose of data processing and management. The values that are provided provide insights into the density of logic and input/output resources that are consumed, which assists designers in comprehending the spatial efficiency of the proposed method within the hardware or processing unit.

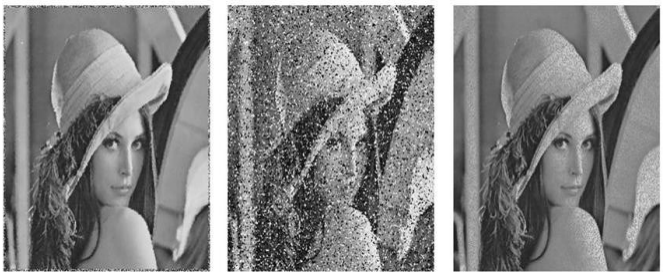

Figure 4.2: Noise removal performance. (a) original image, (b) existing outcome, (c) proposed outcome.

Resource	Estimation	Available	Utilization...
LUT	393	78600	0.50
IO	81	250	32.40

Figure 4.3: Area Utilization Summary of Proposed Method.

The power utilization of the proposed method is examined in Figure 4.3, which provides a comprehensive breakdown of the various power levels at the various components. It is measured that the power of the signal is 5.681 micro-watts, the power of the logic is 6.0132 micro-watts, and the power of the IoB is 8.009 micro-watts. 19.704 microwatts is the value that has been specified for the total dynamic power, which is a representation of the overall dynamic power consumption. On the other hand, the static power, which represents the amount of power that is consumed by the circuit when it is not being used, is reported to be 0.208 micro-watts. After taking everything into consideration, the total power comes out to 19.912 microwatts. When it comes to understanding the power characteristics and efficiency of the proposed method, these values are extremely important. Furthermore, they assist engineers in optimizing for both performance and energy consumption.

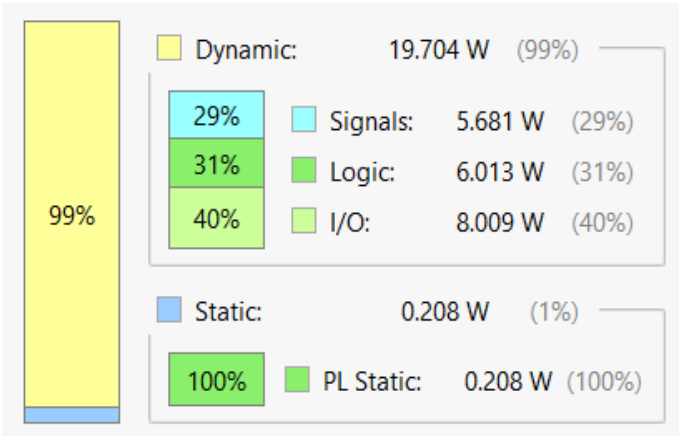

Figure 4.4: Power Utilization Summary of Proposed Method.

The summary of delay utilization reveals key metrics that are associated with the method that has been proposed are shown in the Figure 4.4. It has been determined that the total delay, which is the amount of time it takes for the method to finish its operation, is 23.64 nanoseconds. A further breakdown of this total delay reveals that the logic delay is 6.57 nanoseconds, and the net delay is 17.103 nanoseconds. Both of these delays are included in the total delay. For the purpose of evaluating the effectiveness and performance of the proposed method, these values are of the utmost importance, particularly in applications where timing is critically important.

In Table 4.1, a comparison of the area metrics of the proposed method and the method that is currently in use is presented. The current method makes use of 475 Look-Up Tables (LUTs) and 124 Input/Output Blocks (IoBs), whereas the proposed method demonstrates improvements by reducing the number of LUTs to 393 and the number of IoBs to 81, which is a significant reduction. The numbers presented here demonstrate that the proposed method is capable of achieving a more effective utilization of resources, as it calls for a reduced number of LUTs and IoBs to achieve the same or potentially improved functionality.

Table 4.1: Area Metrics Comparison of Various Methods.

Resource	**Existing Method**	**Proposed Method**
LUT	475	393
IoB	124	81

Table 4.2 presents a comparison of delay metrics between the methods that are currently in use and those that are being proposed. Compared to the current method, which has a total delay of 35.43 nanoseconds, the proposed method has a significantly shorter total delay of 23.64 nanoseconds at its disposal. In addition, the logic delay and the net delay are both decreased by the method that has been proposed, with the values coming in at 6.57 nanoseconds and 17.103 nanoseconds, respectively. Based on these findings, it appears that the method that was proposed demonstrates enhanced efficiency in terms of processing speed, with decreased delays in logic and overall net performance.

Table 4.2: Delay Metrics Comparison of Various Methods.

Metric	**Existing Method**	**Proposed Method**
Total Delay (ns)	35.43	23.64
Logic Delay (ns)	9.01	6.57
Net Delay (ns)	26.42	17.103

The comparison of the power metrics presented in Table 4.3 between the existing methods and the proposed methods is presented. The method that has been proposed demonstrates improvements in a number of different categories with regard to the amount of power that is consumed. Notably, the power of the signal is decreased from 7.81 microwatts to 5.681 microwatts, the power of the logic is decreased from 8.312 microwatts to 6.0132 microwatts, and the power of the IoB is decreased from 11.91 microwatts to 8.009 microwatts. Significant reductions have been made to the total dynamic power, which has been reduced from 26.704 micro-watts to 19.704 micro-watts. Additionally, the static power has been reduced from 1.8 micro-watts to 0.208 micro-watts.

Table 4.3: Power Metrics Comparison of Various Methods.

Power	Existing Method	Proposed Method
Signal Power (µw)	7.81	5.681
Logic Power (µw)	8.312	6.0132
IoB Power (µw)	11.91	8.009
Total Dynamic Power (µw)	26.704	19.704
Static Power (µw)	1.8	0.208
Total Power (µw)	28.504	19.912

The filtering metrics are the primary focus of Table 4.4, which compares the quality of the output produced by the proposed methods to that of the existing methods. With the proposed method, the Peak Signal-to-Noise Ratio (PSNR) is significantly improved, going from 34.58 dB to 45.84 dB. This represents a significant improvement. While the Mean Squared Error (MSE) has reduced from 0.00974 to zero.00477, computational performance however additionally improves the quality and fidelity of the output, making it a advanced choice in terms of filtering performance.

Table 4.4: Filtering Metrics Comparison of Various Methods.

Power	Existing Method	Proposed Method
PSNR (dB)	34.58	45.84
MSE	0.00974	0.00477
SSIM	0.7836	0.9816

5. CONCLUSION

This work at discusses the giant characteristic that filters play in image processing, highlighting the importance of filters in lowering quite a few noise types, which include random noise, salt and pepper noise, and Gaussian noise. Within the context of actual-time packages, specifically within the context of VLSI-oriented hardware implementations, the significance of filtering techniques which can be both efficient and effective is mainly emphasised. Traditional hardware-primarily based filters, however, have run into problems in terms of minimizing the amount of LUT necessities, route delays, and electricity consumption. In essence, this research now not only addresses the demanding situations that are encountered by way of traditional hardware-based totally filters, but it additionally gives a doable solution within the form of the DBMF that has been proposed. A significant improvement in noise reduction talents is executed thru the implementation of the progressive technique, which simultaneously optimizes key hardware metrics. This contributes to the advancement of image processing methodologies, in hardware implementations which can be oriented towards VLSI, and it promises solutions for real-time packages which are greater resource-friendly and efficient.

REFERENCES

[1]. Goel, Anish, M. Omair Ahmad, and M. N. S. Swamy. "Design of a 2D wiener filter with a high throughput FPGA implementation." *2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, 2019.

[2]. Ngo, Dat, Gi-Dong Lee, and Bongsoon Kang. "A 4K-Capable FPGA Implementation of Single Image Haze Removal Using Hazy Particle Maps." *Applied Sciences* 9.17 (2019): 3443.

[3]. Rahnama, O., Cavallari, T., Golodetz, S., Tonioni, A., Joy, T., Di Stefano, L., ... & Torr, P. H. (2019). Real-time highly accurate dense depth on a power budget using an fpga-cpu hybrid soc. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *66*(5), 773-777.

[4]. Sambamurthy, N., & Kamaraju, M. (2020). Power optimised hybrid sorting-based wiener filtering. *International Journal of Digital Signals and Smart Systems*, *4*(1-3), 80-86.

[5]. Arnal, J., & Súcar, L. (2019). Hybrid filter based on fuzzy techniques for mixed noise reduction in color images. *Applied Sciences*, *10*(1), 243.

[6]. Ahmad, Rais, and Som Pal Gangwar. "Rain Streaks Elimination Using Hybrid Wiener filter and Contrast Stretching." *Advances in VLSI, Communication, and Signal Processing*. Springer, Singapore, 2021. 157-178.

[7]. Rini, C., Perumal, B., & Rajasekaran, M. P. (2019). Eradication of Rician Noise in Orthopedic Knee MR Images Using Local Mean-Based Hybrid Wiener filter. In *Proceedings of the 2nd International Conference on Data Engineering and Communication Technology* (pp. 693-702). Springer, Singapore.

[8]. Banerjee, S., Sinha Chaudhuri, S., Mehra, R., & Misra, A. (2021). A Comprehensive Review of Wiener filter, Hybrid Wiener filter and Their Proposed Variants. *Advances in Smart Communication Technology and Information Processing*, 393-406.

[9]. Elsharif, R. I., Ibraheem, B. A., Mustafa, Z. A., & Abass, S. K. (2018). Wavelet decomposition–based speckle reduction method for ultrasound images by using speckle-reducing anisotropic diffusion and hybrid median. *Journal of Clinical Engineering*, *43*(4), 163-170.

[10]. Kannan, L. M., & Deepa, D. (2021). Low power very large scale integration (VLSI) design of finite impulse response (FIR) filter for biomedical imaging application. *DYNA-Ingeniería e Industria*, *96*(5).

[11]. Bevara, Vasudeva, et al. "Design of an efficient QCA-based wiener filter with energy dissipation analysis." *The Journal of Supercomputing* (2022): 1-21.

[12]. Mohan, V., Singla, D., Pulluri, T., Ussa, A., Gopalakrishnan, P. K., Sun, P. S., ... & Basu, A. (2022). EBBINNOT: A Hardware-Efficient Hybrid Event-Frame Tracker for Stationary Dynamic Vision Sensors. *IEEE Internet of Things Journal*.

[13]. Saritha, M., and Krupa Rasane. "Image Despeckling Using Hybrid Method." *ICCCE 2021*. Springer, Singapore, 2022. 93-101.

[14]. Lone, Mohd Rafi. "An FPGA Implementation of an Impulse Noise Reduction Algorithm in Visual Sensor Network." *Circuits, Systems, and Signal Processing* (2022): 1-16.

[15]. Xu, Xiaoli, Chenghua Fan, and Lijun Wang. "A deep analysis of the image and video processing techniques using nanoscale quantum-dots cellular automata." *Optik* 260 (2022): 169036.

[16]. Mathuria, R., Potla, V. V. K., Prajapati, P., Gupta, S., Kakkireni, N., & Darji, A. (2022, July). Hardware Co-Simulation of an Efficient Adaptive Filter based ECG Denoising System with Inbuilt Reference Generator. In *2022 IEEE Region 10 Symposium (TENSYMP)* (pp. 1-6). IEEE.

[17]. [17]Yang, C. H., & Lin, C. H. (2020, October). Combined Architectures of Denoising Filter and Local Binary Patterns Feature Extraction. In *2020 IEEE 9th Global Conference on Consumer Electronics (GCCE)* (pp. 152-153). IEEE.

[18]. Goel, A., Ahmad, M. O., & Swamy, M. N. S. (2019, August). Design of a 2D wiener filter with a high throughput FPGA implementation. In *2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)* (pp. 1073-1076). IEEE.

[19]. Pashaeifar, Masoud, Mehdi Kamal, Ali Afzali-Kusha, and Massoud Pedram. "Approximate reverse carry propagate adder for energy-efficient DSP applications." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 26, no. 11 (2018): 2530-2541.

[20]. Srinu, B., Bevara, S., & Kumar, M. N. (2019). FPGA Based Implementation of Wiener filter using Compare and Exchange Unit. *i-Manager's Journal on Digital Signal Processing*, *7*(1), 33.

[21]. Allwin Devaraj, S., Jacob, S., Darling Rosita, J., & Vijay, M. M. (2021). VLSI Implementation of Color Demosaicing Algorithm for Real-Time Image Applications. In *2nd EAI International Conference on Big Data Innovation for Sustainable Cognitive Computing* (pp. 317-329). Springer, Cham.

[22]. Kattekola, N., Jawale, A., Nath, P. K., & Majumdar, S. (2021). Efficient partial product reduction for image processing application using approximate 4: 2 compressor. *Circuit World*.

[23]. Sindhana Devi, M., and M. Soranamageswari. "Efficient impulse noise removal using hybrid neuro-fuzzy filter with optimized intelligent water drop technique." *International Journal of Imaging Systems and Technology* 29.4 (2019): 465-475.

[24]. Bose, Sudipta, Arijit De, and Indrajit Chakrabarti. "Area-Delay-Power Efficient VLSI Architecture of FIR Filter for Processing Seismic Signal." *IEEE Transactions on Circuits and Systems II: Express Briefs* 68.11 (2021): 3451-3455.

[25]. Abubakar, A., Zhao, X., Takruri, M., Bastaki, E., & Bermak, A. (2020). A hybrid denoising algorithm of BM3D and KSVD for Gaussian noise in DoFP polarization images. *IEEE Access*, *8*, 57451-57459.

[26]. Arjun, Jawale Amol, and Shubhankar Majumdar. "Development of approximate compressor based hybrid dadda multiplier for image de-noising applications." *2019 IEEE 16th India Council International Conference (INDICON)*. IEEE, 2019.

[27]. Chen, Wei-Ting, Pei-Yin Chen, Yu-Che Hsiao, and Shih-Hsiang Lin. "A low-cost design of 2D wiener filter." *IEEE Access* 7 (2019): 150623-150629.

[28]. Kumar, Rahul, et al. "A hybrid dehazing method and its hardware implementation for image sensors." *IEEE Sensors Journal* 21.22 (2021): 25931-25940.

[29]. Gollamandala, Udaykiran Bhargava, Vasujadevi Midasala, and Vallabhuni Rajeev Ratna. "FPGA implementation of hybrid recursive reversable box filter-based fast adaptive bilateral filter for image denoising." *Microprocessors and Microsystems* 90 (2022): 104520.

[30]. Lin, C., Chen, W. T., Chou, Y. C., & Chen, P. Y. (2019). A novel comparison-free 1-D wiener filter. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *67*(7), 1329-1333.

[31]. Najafi, M. H., Lilja, D. J., Riedel, M. D., & Bazargan, K. (2018). Low-cost sorting network circuits using unary processing. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, *26*(8), 1471-1480.