VLSI IMPLEMENTATION OF HYBRID MEDIAN FILTER USING MULTI-LEVEL DATA COMPARATORS

SHAIK.SUMAYYA SULTHANA, K.MADHAVI, T.VIJAY KUMAR
M.Tech Student, Assistant Professor, Associate Professor
Department of ECE
Dr.K V Subba Reddy Institute of Technology, Kurnool

ABSTRACT:

Filters are used to remove the different types of noises including salt and pepper, gaussian, and random noises from image. Therefore, the VLSI oriented hardware implementation of filters plays the crucial role in real time applications. However, the conventional hardware-based filters are failed to reduce the look-up-table (LUT)s, path delays, and power consumption. Therefore, this work is focused on implementation of Hybrid Median Filter (HMF) using Data Comparator (DC) logic. Initially, the multiplexer selection logic-based data comparator is used to identify the high and low values from two numbers. Then, data comparator is repeated for multiple number of times for nine pixels combinations, which identifies the median value from nine pixels. The subjective and objective evaluation shows that the proposed HMF-DC resulted in superior performance in terms reduced noise, hardware metrics like LUTs, delay, and power consumption as compared to state of art approaches.

I. INTRODUCTION:

Harsh environments, like space, are a challenge for electroniccircuits in general and for memories in particular. For example,radiation causes several types of errors that can disrupt the circuitfunctionality [1]. One common error for SRAM memories is softerrors that change the value of one or more memory cells [2].To avoid corruption in the data stored in the memory, error correctioncodes (ECCs) are commonly used
[3].ECCs add parity check bits to each memory word to detect andcorrect errors. This requires an encoder to compute those bits whenwriting to the memory and a decoder to detect and correct errorswhen reading from the memory. These elements increase the memoryarea and the power consumption, and can also reduce the accessspeed. These overheads increase with the error correction capabilityof the ECC. Traditionally, codes that can correct a single bit errorper word have been used. In particular, single error correction–doubleerror detection (SEC–DED) codes that can also detect double errorsare commonly used [4]. In recent years, the number of errors that affect more than onememory cell has increased significantly. This is due to the scalingof the memory cells and is projected to grow further [5]. Theseerrors, known as multiple cell upsets (MCUs), pose a challengefor SEC–DED codes. One solution to ensure that the MCU errorscan be corrected is to interleave the bits of different logical wordsso that an MCU affects one bit per word [6]. This is based onthe observation that the cells affected by an MCU are physicallyclose [7]. Interleaving, however, has a cost as it complicates thememory design [8]. In some space applications, there is an additionalissue as the number of errors is high, and SEC–DED codes may notbe sufficient when errors accumulate over time [9]. These issues haveled to an increased interest on the use of more advanced ECCs toprotect SRAM memories.As MCUs affect cells that are close together, a number of codesthat can correct doubleadjacent or triple-adjacent errors have beenrecently proposed [8], [10]–[12].

These codes, in many cases, do notrequire additional parity check bits and in the rest require only one ortwo additional bits. The decoding complexity increases but in manycases can still be implemented with limited impact on the memoryspeed. These codes are useful for applications in which the error rateis low, however, when the error rate is large, codes that can correcterrors on multiple independent bits are needed [9].Research for multibit ECCs has focused on reducing the decodinglatency as in many cases, the traditional decoders are serial andrequire several clock cycles. To some extent this can be done forsome traditional ECCs by using a parallel syndrome decoder [13] butthe decoder complexity explodes as the error correction capability orthe word size increases. Another approach is to use codes that canbe decoded with low delay, such as orthogonal Latin squares (OLSs)or difference set (DS) codes [14], [15]. In the case of OLS codes,the main issue is that they are not optimal in terms of the numberof parity check bits and thus require more memory overhead. TheDS codes are more competitive in terms of parity check bits but arestill not optimal for some word lengths. For example, the*(*21, 10*)* DS code can correct 2-bit errors while a code with

a similarblock size and code rate, and the (24,12) extended Golay code [16]can correct 3-bit errors. However, the Golay code requires a morecomplex decoder that needs several clock cycles [17].Namba *et al.* [18] have proposed a compromise solution forBose–Chaudhuri–Hocquenghem codes. The idea is that the mostcommon error patterns are decoded in parallel and the rest serially.In particular, single and double-adjacent errors are corrected in asingle clock cycle. This means that the most memory accesses canbe completed in a single clock cycle, and only a small percentage ofthe words in error require a full serial decoding. This can enable theuse of traditional ECCs that do not support fast parallel decoding toprotect SRAM memories In this brief, the use of the scheme in [18] is considered for the(24,12) Golay code. In more detail, an efficient parallel decoder capableof correcting the single and double-adjacent errors is presented.The decoder exploits the properties of the Golay code to reduce theimplementation cost. This results in a decoder that is simpler than a traditional SEC decoder but that can also correct all double-adjacenterrors and some triple-adjacent errors. The proposed decoder has beenimplemented in hardware description language and mapped to a 65-nm technology to show its benefits. The main contribution of thisbrief is to enable a fast and efficient parallel correction of the singleand double-adjacent errors in the (24,12) Golay code.

$$\begin{bmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&1&1&0&0&0&1&0&1\\
0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&1&1&0&0&0&1&1\\
0&0&1&0&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1&1&0&0&0&1\\
0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1&1&0&0&1\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1&1&0&1\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&1&1&0&1&1&1&1\\
0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&1&0&1&1&0&1&1&1\\
0&0&0&0&0&0&0&1&0&0&0&0&1&1&0&0&0&1&0&1&1&0&1&1\\
0&0&0&0&0&0&0&0&1&0&0&0&1&1&1&0&0&0&1&0&1&1&0&1\\
0&0&0&0&0&0&0&0&0&1&0&0&0&1&1&1&0&0&0&1&0&1&1&1\\
0&0&0&0&0&0&0&0&0&0&1&0&1&0&1&1&1&0&0&0&1&0&1&1\\
0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&1&1&1&1&0
\end{bmatrix}$$

Fig. 1. Parity check matrix of the (24,12) Golay code.

$$\begin{bmatrix}
1&1&0&1&0&0&0&1&0&1&0&1&0&0&0&0&0&0&1&0&0&0&1\\
0&0&1&1&0&1&0&0&0&1&0&1&0&1&0&0&0&0&0&0&0&1&0&1\\
0&1&0&0&1&1&0&1&0&0&0&1&0&1&0&1&0&0&0&0&0&0&0&1\\
0&0&0&1&0&0&1&1&0&1&0&0&0&1&0&1&0&1&0&0&0&0&0&1\\
0&0&0&0&0&1&0&0&1&1&0&1&0&0&0&1&0&1&0&1&0&0&0&1\\
0&0&0&0&0&0&0&1&0&0&1&1&0&1&0&0&0&1&0&1&0&1&0&1\\
0&1&0&0&0&0&0&0&0&1&0&0&1&1&0&1&0&0&0&1&0&1&0&1\\
0&1&0&1&0&0&0&0&0&0&0&1&0&0&1&1&0&1&0&0&0&1&0&1\\
0&1&0&1&0&1&0&0&0&0&0&0&0&1&0&0&1&1&0&1&0&0&0&1\\
0&0&0&1&0&1&0&1&0&0&0&0&0&0&0&1&0&0&1&1&0&1&0&1\\
0&1&0&0&0&1&0&1&0&1&0&0&0&0&0&0&0&1&0&0&1&1&0&1\\
0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&1&0
\end{bmatrix}$$

Fig. 2. Parity check matrix of the (24,12) Golay code with the proposed bit placement.

II.LITERATURE SURVEY

THE comparator is a very useful combinational circuit used for testing whether the binary number at one input is greater or less than to another binary number. An XOR gate can be used as an essential comparator. The Comparators are comprised of two types (a).Magnitude Comparator (b). Data Comparator. The former compares only the magnitude of the two binary numbers and the later gives the greater and a lesser data itself. The Magnitude comparator has two outputs to indicate whether first input is greater than second input or vice versa . Whereas data comparator can also be referred to two cell comparator, as it compares word X with word Y and gives out a Higher and lower value respectively A compressed, good quality cost, high performance, and low power comparator play a significant role in almost all hardware comparators. The work attempt to observe the features of certain comparator circuits which assure better performance compared to existing circuits. Karpagaabirami and Ramamoorthy [1] developed an Adaptive Rank Order filter (AROF) with VLSI implementation had been developed to remove impulse noise and pipelining with parallel processing in order to speed up filtering process. The advantage of Decision Rank Order Filter (DROF) consumes less area and also architecture is simple compared to Decision Tree
Based De-noising Method(DTBDM).The disadvantage of VLSI DTBDM involves too many architectures for detection of noise and reconstruction of noisy pixel.

Ayeesha et al [2] explored the design of high speed and low power comparator since it operates only with 1volt power and less propagation delay and its architecture includes two stage CMOS op-amp circuit. In this work, comparator is designed with cadence tool with 0.18micrometer technology.

Bharat et al [3] introduced a special types of comparators and these circuits are simulated with 1 Volt DC supply voltage in LTspice-IV using PTM 45nm technology. Unlike static and dynamic characteristics of all these comparators are considered and compared and it operates with higher speed and provide more stabilized output compare to 90nm and 180nm.

Mehamood et al [4] developed various new designs to reduce the area and power consumption as small saving in area and power of a circuit yield a large

overall saving. From the evaluation, found that fullcustom design saves about 50%in area and 35%in power consumption when compared to autogenerated design.

Vasanth et al [5] developed a parallel architecture and pipelined architecture for modified shear sorting. The method introduced an area efficient data comparator for sorting 9 elements. The basic processing element is area optimized two cell sorter. A group of three two cell sorter form a three cell sorter which in turn uses compare and swap approach for ordering the data sequence. Keeping chan [6] developed a VLSI algorithms and implementation architectures for a class of nonlinear filters. The class of filters contains all of the functions earlier defined by stack filters, where rank-order and median filters are special cases. The function of a stack filter can be realized in k-step recursive use of one binary processing circuit. Vasanth et al [7] introduced a novel Borrow Look Ahead Logic based Comparator (BLAC) and implemented the output.

III.PROPOSED METHOD

Noise is undesirable information which degrades image quality [1]. The image can be noisy because of dust present on the lens, electronic noise in camera, imperfection present in the image sensor or can be introduced when image data is transmitted over communication channel. The motive of image processing is to get rid of noise from a digital image while keeping its features unaltered. Image filter is the key blog of Image processing system. An impulsive noise can be added when image data transmitted over an insecure communication channel. [2]. It causes small size dots or dark/black spot on an image. Impulse noise is uniformly distributed and the most often mentioned noise in digital images. Further, Impulse noise can be divided into two parts. The first one is salt and pepper noise which is a type of impulse noise having noisy pixel intensity either 0 (minimum) or 255 (maximum) in the case of gray scale images. It appears as randomly scattered black or white dots over the images [3]. The second one is the random-valued shot noise which has arbitrary valued noisy pixels. To remove these noises, it is necessary that the acquired image must pass through an image pre-processing stage defined as a filter [4]. Spatial and frequency domain are two categories of the filtering operation. Generally, filters are implemented by MATLAB, OCTAVE software's in real time systems [5]. As it is a well-known fact that software implementation offers less processing speed in comparison to hardware implementation [6]. Hardware implementation has become better alternative after the boost in the VLSI technology. To reduce the power consumption in the systems, more cooling devices have to be incorporated results in the costly system. Keeping the same functional capabilities with the reduction in power factors are heavily demanding. Yet in that context, battery and power optimizing technology have not matured up to that target. Most of these products include embedded microprocessors, DSPs and ASICs [7]. It is a provoking undertaking to accomplish low force plan of any VLSI circuit. There are various degrees of advancement in VLSI configuration measure for low force applications. For battery operated portable products [8], power has been the main concern. As System-on-Chip (SoC) developing with more power transistors, it requires less power consumption. Power consumption reduction in highly integrated SoC cut down the heating problem. It reduces the cost of expensive packing and cooling mechanism [9]. In this work, VLSI architecture for noise reduction in different imaging applications is proposed to deal with the above issues of power and cost reduction, respectively. To achieve low resources, this work mainly focusing on Verilog based coding mechanisms with FPGA prototype [10]. Then, the subjective and objective image statistics are measured by using MATLAB environment. The major contributions of this work are as follows:

- Implementation of data comparator for identifying the high, low values using multiplexer selection logic.
- Implementation of multi-level network for selection of median value from nine pixels in a window.
- Implementation of HMF-DC for removal of different types of noises from image using hybrid switching of data blocks. Rest of the article is organized as follows: section 2 delas with literature survey, section 3 delas with the proposed HMF-DC implementation, section 4 delas with analysis of results with performance comparison, section 5 concludes the article with possible future directions.

Commotion is signal-subordinate and is hard to be eliminated without disabling image subtleties. Various sorts of error influence the image, like Gaussian, drive, dot and Rician commotions. In the image denoising measure, data about the sort of error present in the original image assumes a huge part. The image error can be delegated either added substance or multiplicative. The image is a 2-D function $f(x, y)$ of

light intensities, where f is amplitude at any spatial coordinate x and y. The beam of light falls on an object and reflected light reaches to eyes. It makes human to see the object. The smallest element of the image is pixels. Each pixel represents intensity value at a particular location. Mathematically, image can be represented as Equation (1).

$$F(x, y) = I(x, y).\, R(x, y) \quad (1)$$

Here, $I\,(x, y)$ is intensity of incident light on object, $R\,(x,\, y)$ is reflected light from object in intensity and $F(x,\, y)$ is intensity of resultant image. The image restoration is the process to denoising a image, which has been distorted by prior knowledge of degradation model. Once the degradation model is known, by applying inverse process to recover the desired imagery. image restoration is different than traditional image enhancement techniques. It is a subjective process, which produces more effective results to an observer with and without using degradation model. The image degradation process is shown in Figure 1. image degradation model in the spatial domain is achieved by performing the convolution between $f\,(x,\, y)$ and model function $(h(x, y))$.

$$F(x, y) = h(x, y) * f\,(x, y) + \eta\,(x, y) \quad (2)$$

Here, $\eta\,(x,\, y)$ represents the speckle noise. Further, degradation model in the frequency domain is achieved by applying the Fourier transform as follows:

$$F\,(u, v) = h\,(u, v) * f\,(u, v) + \eta(u, v) \quad (3)$$

Here, u, v represents the frequency domain coefficients.

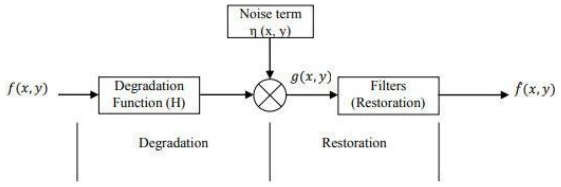

Figure : image degradation model The HMF-DC is a digital non-linear method used to eliminate noise, similar to that of the medium filter. However, by keeping valuable details in the image, it typically does better than the mean filter. This filter class belongs to the class of filter that preserves the edge. These filters smooth down the data while maintaining the details. The median is only the average of all the pixel values in the area. It doesn't correspond to the mean (or average), but the median is half bigger and half smaller in the neighborhood. The median is a "centre indication" stronger than the average. Like the medium, every pixel in the image is taken into account by the HMF-DC and its close neighbors are examined to determine if it is typical of their surroundings. It replaces the median value with those values instead of just replacing the pixel value by the mean of the next pixel value. Particularly better than the typical filter is to take away impulsive noise. The HMF-DC eliminates the noise as well as the fine details as the difference between them cannot be identified. Anything that is comparatively tiny in size with the area size will minimize and filter out the median value. In other words, the HMF-DC can differentiate between fine detail and noise.

3.1 Data comparator

Figure 2 shows the block diagram of data comparator, which is used to perform the selection of highest and lowest values from the given two input data. Further, the data comparator block contains inputs as A, B and outputs are High (H) and Low (L). Step 1: Initially, A < B condition is verified, if condition is satisfied selection line of multiplexer becomes one, else condition failed selection line becomes zero.

Step 2: Input-A is applied as Data-input-0 and InputB is applied as Data-input-1 to 2to1 multiplexer. If A value is smaller than B, then selection line becomes one and multiplexer generates H as input-B through selection switching. If A value is higher than B, then selection line becomes zero and multiplexer generates H as input-A through selection switching.

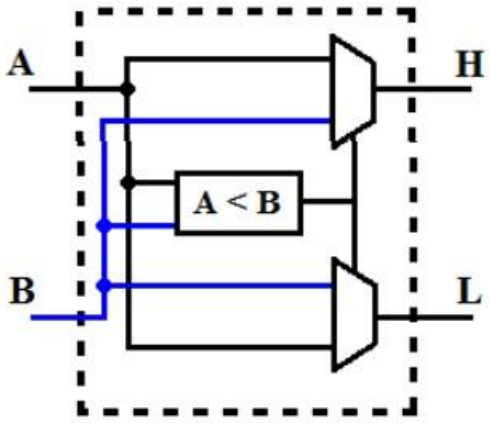

Figure . Block diagram of data comparator.

Step 3: Input-B is applied as Data-input-0 and InputA is applied as Data-input-1 to 2to1 multiplexer. If A value is smaller than B, then selection line becomes one and multiplexer generates L as input-A through selection switching. If A value is higher than B, then selection line becomes zero and multiplexer generates L as input-B through selection switching.

3.2 Hardware architecture of HMF

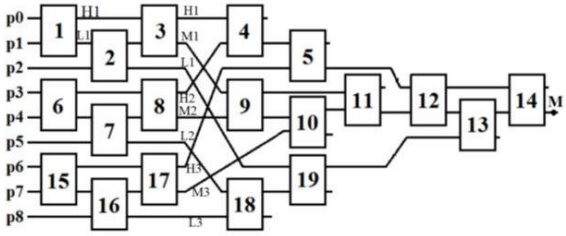

Figure . Hardware architecture of HMF-DC.

Figure 3 shows the hardware architecture of HMFDC, which contains the fourteen number of hardware resource blocks. Here, inputs P0, P1, P2, P3, P4, P5, P6, P7, and P8 are applied to HMF-DC, which generates the median value as M. Here, DC-1, DC-2, DC-3 are grouped together and performs the selection of high (H1), low (L1) and median (M1) values. Similarly, DC-6, DC-7, DC-8 and DC-15, DC-16, DC-17 performs the generation of high, low and median values. Further, DC-4 is used to select the lowest value from H1, H2, H3 outcomes.

Furthermore, DC-18 is used to select the highest value from L1, L2, L3 outcomes. Similarly, DC-9, DC-10, DC-11 are grouped together and performs the selection of high, low and median values. Like this, the process will continue and generates the median value (M) from DC-14 low outcome.

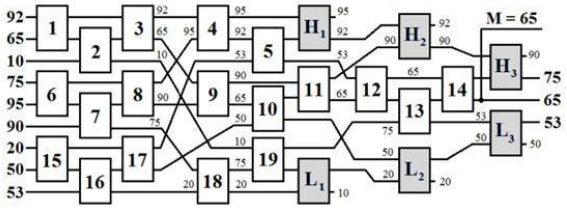

Figure . Example of HMF-DC. Figure 4 provides a numerical example that might help better illustrate how the HMF-DC system works. In this case, the median value is defined by the two non-median outputs that are located the closest to it. As can be seen in this diagram, the H1, H2, and H3 blocks are used to sort the four highest pixel values (95, 92, 90, and 75), which results in 75 being the upper range. At the same time, the three lower ranges (L1, L2, and L3) are used to sort the four lowest values (10, 20, and 50), which results in 53 being the lower range.

IV. RESULTS AND DISCUSSION

Xilinx ISE software was used to create all of the HMF-DC designs. This software programmed gives two types of outputs: simulation and synthesis. The simulation results provide a thorough examination of the HMF-DC architecture in terms of input and output byte level combinations. Decoding procedure approximated simply by applying numerous combinations of inputs and monitoring various outputs through simulated study of encoding correctness. The use of area in relation to the LUT count will be accomplished as a result of the synthesis findings. In addition, a time summary will be obtained with regard to various path delays, and a power summary will be prepared utilizing the static and dynamic power consumption. Further, MatlabR2020a software is used to evaluate the subjective performance of HMF-DC.

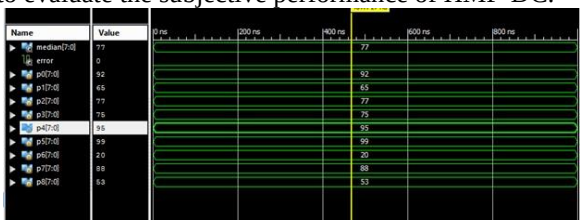

Figure . Simulation outcome of HMF-DC.

Figure 5 presents the simulation outcome of HMFDC. Here, P0, P1, P2, P3, P4, P5, P6, P7, P8 are the inputs to HMF-DC and median is the output value.

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice LUTs	437	303600	0%
Number of fully used LUT-FF pairs	0	437	0%
Number of bonded IOBs	81	700	11%

Figure . Design summary.

```
LUT6:I0->O            1    0.043   0.350  c14/a[7]_b[7]_LessThan_1_o22
LUT6:I5->O           14    0.043   0.422  c14/a[7]_b[7]_LessThan_1_o24
LUT4:I3->O            3    0.043   0.507  c14/a[7]_b[7]_LessThan_1_o25
LUT6:I3->O            1    0.043   0.613  H3/a[7]_b[7]_LessThan_1_o3 (H:
LUT6:I0->O            1    0.043   0.405  H3/a[7]_b[7]_LessThan_1_o4 (H:
LUT3:I1->O            2    0.043   0.410  H3/a[7]_b[7]_LessThan_1_o1_SW:
LUT5:I3->O            1    0.043   0.000  H3/a[7]_b[7]_LessThan_1_o1_G
MUXF7:I1->O           1    0.178   0.405  H3/a[7]_b[7]_LessThan_1_o1 (H:
LUT5:I3->O            6    0.043   0.631  H3/a[7]_b[7]_LessThan_1_o21 (H
LUT5:I0->O            1    0.043   0.613  H3/Mmux_13 (h31<2>)
LUT6:I0->O            1    0.043   0.405  median[7]_h31[7]_LessThan_1_o:
LUT5:I3->O            1    0.043   0.613  median[7]_h31[7]_LessThan_1_o:
LUT6:I0->O            1    0.043   0.339  error5 (error_OBUF)
OBUF:I->O                  0.000          error_OBUF (error)
----------------------------------------
Total                      20.375ns (1.726ns logic, 18.649ns route)
                                    (8.5% logic, 91.5% route)
```

Figure . Time summary**.** Figure 6 shows the design (area) summary of proposed method. Here, the proposed method utilizes the low area in terms of slice LUTs i.e., 437 out of available 303600. Figure 7 shows the time summary of proposed method. Here, the proposed method consumed total 20.375ns of time delay, where 1.726ns is logical delay, and 18.649ns is route delay. Figure 8 shows the power consumption report of proposed DCM-RTPG-BFD. Here, the proposed DCM-RTPG-BFD consumed power as 32.83 milli watts.

2. Summary
2.1. On-Chip Power Summary

On-Chip Power Summary				
On-Chip	Power (mW)	Used	Available	Utilization (%)
Clocks	1.30	3	---	---
Logic	0.00	10	11776	0
Signals	0.00	20	---	---
IOs	0.00	20	372	5
Quiescent	31.52			
Total	32.83			

Figure . Power summary.

Figure . Visuval performmace of HMF-DC. (a) original image, (b) noisy image, (c) SMF [18], (d) DMF [21], (e) AMF [25], (d) proposed HMF-DC.

Figure shows the filtering performance of various methods like SMF [18], DMF [21], AMF [25], and proposed HMF-DC. Here, SMF [18] and AMF [25] methods resulted outcome still contains the highher noises, DMF [21] method outcome contains the low level noises. But, the proposed HMF-DC method resulted outcome is looks similar to the original image. Table 1 compares the performance evaluation of proposed HMF-DC method. Here, the proposed HMF-DC resulted in superior (reduced) hardware performance in terms of LUTs, time-delay, and power consumption as compared to conventional approaches such as SMF [18], DMF [21], and AMF [25]. Further, the proposed HMF-DC resulted in improved subjective performance in terms of peak signal to noise ratio (PSNR), stctural similarity index metric (SSIM) as compared to conventional approaches such as SMF [18], DMF [21], and AMF [25]. Further, the graphical representation of performance comparison is presented in Figure 10.

Table 1. Performance evaluation.

Metric	SMF [18]	DMF [21]	AMF [25]	Proposed HMF-DC
LUTs	767	655	542	437
Time delay (ns)	51.927	43.837	32.735	20.37
Power consumption (mw)	82.61	73.41	58.26	32.83
PSNR (dB)	37.34	42.45	48.38	54.53
SSIM	0.827	0.893	0.927	0.992

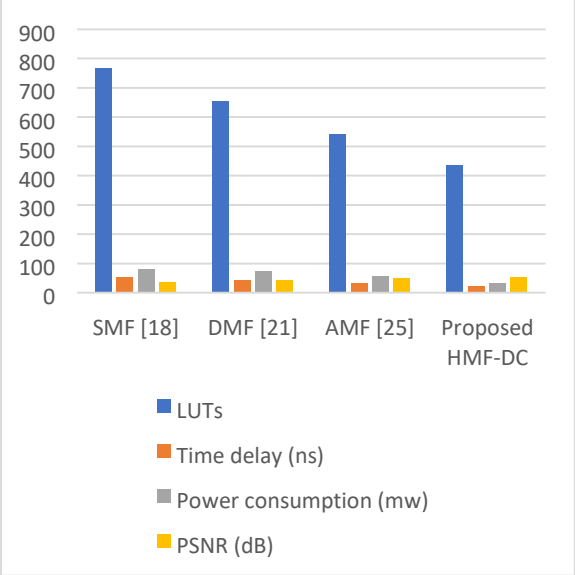

Figure . Graphical representation of performance.

V.CONCLUSION

The development of a Hybrid Median Filter by making use of Data Comparator logic is the primary emphasis of this study. In the beginning, a multiplexer selection logic-based data comparator is used in order to determine which of two numbers have high and low values. After then, the data comparator is carried out a number of times for the nine different possible combinations of pixels, which determines the median value for all nine of those values. The subjective and objective evaluations both reveal that the suggested HMF-DC resulted in greater performance when compared to the state-of-the-art techniques in terms of decreased noise, latency, and power consumption. Hardware metrics such as LUTs were also reduced and software metrics such as PSNR, SSIM are improved using the proposed HMFDC approach. Further, this

work can be extended with the hybrid adaptive filters for improved PSNR performance.

REFERENCES

[1]. Goel, Anish, M. Omair Ahmad, and M. N. S. Swamy. "Design of a 2D median 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 median 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 Median Filter and Contrast Stretching." *Advances in VLSI, Communication, and Signal Processing*. Springer, Singapore, 2021. 157178.

[7]. Rini, C., Perumal, B., & Rajasekaran, M. P. (2019). Eradication of Rician Noise in Orthopedic Knee MR Images Using Local Mean-Based Hybrid Median 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 Median Filter, Hybrid Median 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]. 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.

[12]. Goel, A., Ahmad, M. O., & Swamy, M. N. S. (2019, August). Design of a 2D median filter with a high throughput FPGA implementation. In *2019 IEEE 62nd International Midwest Symposium on Circuits and Systems (MWSCAS)* (pp. 10731076). IEEE.

[13]. 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.

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