iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** VIII **Month of publication:** August 2025

DOI: https://doi.org/10.22214/ijraset.2025.73609

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VIII Aug 2025- Available at www.ijraset.com

Approximate Multiplier for Image Masking Based on Image Multiplication

Rajarajeshwari G[1], Karthigai Lakshmi S[2]
[1, 2]*Department of Electronics and Communication Engineering, SSM Institute of Engineering and Technology, Dindigul, 624002, India*

Abstract: ***This paper introduces a Verilog Hardware Description Language (HDL) implementation of a specialized multiplier tailored for image processing, with a specific emphasis on edge detection applications. The design integrates an approximate 4-2 compressor and an error correction module, optimizing accuracy while minimizing hardware complexity. Rigorous evaluations, employing ModelSim and MATLAB simulations, validate the design's superior accuracy compared to conventional multipliers, achieved with minimal additional hardware complexity. The Verilog HDL implementation is precisely engineered for seamless integration into existing image processing systems, showcasing adaptability and potential across diverse applications. Executed with precision using ModelSim 6.4c and synthesized with the Xilinx Tool, the multiplier strategically incorporates compressors to enhance performance, particularly focusing on efficiency improvement in image multiplication for masking applications. This work strategically converges hardware innovation and image processing intricacies, delving into multiplier design within the realm of edge detection. The proposed system provides an efficient solution, ensuring heightened accuracy and performance while maintaining ease of integration into existing systems. Sobel Image Masking, a fundamental technique in edge detection, further augments the paper's practical implications. The multiplication-based approach in Sobel Image Masking employs various advanced multipliers like Booth, Dadda, Shift and add Multipliers, and Wallace multipliers, elucidating their pivotal roles in augmenting gradient values through convolution operations. In alignment with Artificial Intelligence (AI) and Edge Computing, this paper underscores the significance of computational efficiency in image processing. Leveraging Verilog HDL, integrating error correction mechanisms, and applying the specialized multiplier to image masking collectively contribute to advancing AI technologies, especially in edge computing scenarios where task-specific hardware optimization is crucial. Positioned at the intersection of hardware design, image processing, and AI, this paper underscores its relevance in the dynamic landscape of intelligent systems and edge computing, emphasizing the synergy between hardware advancement and artificial intelligence in the realm of edge computing.***

Keywords: ***Verilog HDL; Image Processing; Edge Detection; 4-2 Compressor; Error Correction Module; Computational Efficiency.***

I. INTRODUCTION

Multipliers are integral components in DSP and embedded applications, exerting a profound impact on processor speed [1]. The demand for high speed multipliers is prevalent across various applications, necessitating the use of different algorithms to enhance multiplication speed. Although multiplication is a fundamental operation in computing systems, implementing multipliers poses challenges, including the consumption of significant hardware resources and operation at low speeds. This is particularly critical in DSP and embedded systems, where multiplication plays a fundamental role. Signal processing algorithms heavily rely on multiplication, yet traditional multipliers present challenges such as large area requirements, extended latency, and considerable power consumption. In response, the design of low power multipliers becomes crucial for efficient low power VLSI system development [1]. The performance of a system is often contingent on the multiplier, which typically represents the slowest and most space consuming element. Balancing speed and area considerations becomes a major design challenge, given the inherent conflict between the two constraints. Multiplication involves three main steps: partial product generation, partial product reduction, and final addition. Efficient arithmetic computation cells, including adders and multipliers, are essential in VLSI systems, especially in microprocessors and digital signal processors executing algorithms like convolution and filtering. Multipliers, as critical components, significantly influence overall circuit performance, especially when power consumption and computation speed are key considerations. As VLSI technologies trend towards deep submicron regimes, achieving power efficiency becomes paramount by lowering the power supply voltage. Circuit design techniques play a crucial role in achieving high power efficacy in arithmetic circuits at ultralow supply voltages.

Fast multipliers typically comprise partial product generation, partial product accumulation, and carry propagating addition. Booth encodings are commonly employed in partial product generation to reduce the number of partial products [2]. The Carry Save Adder (CSA) tree, acting as a summation tree, further reduces partial products to two in the second function. The final addition is typically executed by a fast carry propagate adder, such as carry look ahead adder and carry skip adder. Modern designs favor (4:2) compressors due to their ability to form a regular interconnected cell structure, with higher input compressors like (5:2) and (6:2) gaining popularity in high precision multipliers for improved performance [16].

II. RELATED WORK

The central challenge addressed by the proposed idea is the requirement for an ac curate and efficient multiplier design tailored for image processing applications, specifi cally in edge detection. Existing multipliers may lack the necessary accuracy, and high hardware complexity in more accurate designs can lead to performance issues. The proposed design aims to overcome this problem by incorporating an approximate 4-2 compressor and an error correction module to improve accuracy while keeping hardware complexity low. Implementing this design using Verilog HDL enables seamless integration into existing image processing systems, providing an efficient solution to the problem of accurate and efficient multipliers for image processing applications.

This paper aims to propose a novel design for a multiplier that can be used in image processing applications, particularly in edge detection. The proposed design incorporates an approximate 4-2 compressor and an error correction module to enhance accuracy while also keeping hardware complexity low. The proposed multiplier has the potential to significantly improve the performance of various image processing systems, particularly those relying on accurate edge detection.

In contemporary image processing systems, multipliers play a pivotal role in executing intricate mathematical computations, particularly in tasks like edge detection. Traditional multiplier designs often prioritize accuracy through the utilization of high precision arithmetic components such as carry look ahead adders. While effective, these designs tend to be complex and resource intensive, potentially leading to challenges in terms of hardware constraints and overall system performance. Alternatively, some existing multiplier designs adopt an approach that leverages low precision arithmetic alongside error correction techniques. This strategy aims to strike a balance between computational accuracy and resource efficiency. However, these designs may exhibit limitations in terms of achieving the desired precision, and they can be susceptible to errors, potentially impacting the reliability of image processing results. One prevalent existing technique is the truncated multiplier with constant correction, which involves truncating the multiplication process and subsequently applying constant correction methods to mitigate errors. While this technique offers a compromise between accuracy and hardware simplicity, it may still fall short of meeting the stringent precision requirements demanded by certain image processing applications, especially those related to edge detection in critical scenarios. Despite the advancements in existing multiplier designs, there remains a persistent need for a solution that can seamlessly integrate into image processing systems, providing the requisite accuracy without unduly burdening hardware resources. The proposed design, incorporating an approximate 4-2 compressor and an error correction module [2], presents a promising avenue to address the limitations of traditional and contemporary multiplier approaches, offering a balanced trade off between precision and hardware efficiency in the context of image processing applications.

The existing multiplier designs for image processing applications present several critical drawbacks that impede their efficacy in delivering accurate and efficient results [6]. Primarily, these designs tend to suffer from elevated error rates, notably apparent in high precision arithmetic approaches like carry look ahead adders, potentially compromising the accuracy of computational outcomes. Additionally, while aiming to strike a balance between precision and resource efficiency, alternative designs employing low precision arithmetic with error correction techniques may still fall short, resulting in increased error rates and reduced reliability in crucial tasks such as edge detection. Furthermore, these existing designs often exhibit complexity in implementation within practical time constraints, hindering real time or near real time processing capabilities essential for image analysis. Moreover, the resource intensive nature of traditional multiplier designs, especially those relying on high precision arithmetic, poses challenges in efficient resource utilization, potentially limiting their applicability in re source constrained image processing systems. Their limited adaptability and scalability further hinder their suitability for diverse computational demands, and in scenarios requiring adaptable and scalable computational capabilities, these designs may not seamlessly integrate. Lastly, the energy inefficiency stemming from high precision arithmetic contributes to increased power consumption, impacting the overall energy efficiency of image processing systems. Addressing these limitations demands an innovative multi plier design that not only ensures heightened accuracy but also optimizes resource utilization, adaptability, scalability, and energy efficiency to meet the dynamic demands of image processing applications

III. MULTIPLIER INTEGRATION FOR ADVANCED IMAGE PROCESSING

The envisioned system introduces a novel multiplier design tailored for image processing applications, specifically in edge detection. The design integrates an approximate 4-2 compressor and an error correction module, aiming to enhance accuracy while minimizing hardware complexity. The 4-2 compressor reduces the number of partial products generated during multiplication, consequently lowering the overall hard ware requirements. Meanwhile, the error correction module addresses inaccuracies stemming from the use of low precision arithmetic. Implemented using Verilog HDL and validated through simulations in ModelSim and MATLAB, the proposed multiplier demonstrates superior accuracy compared to traditional counterparts, with only a marginal increase in hardware complexity.

Sobel Image Masking, a widely used technique in image processing and computer vision for edge identification, involves convolving the image with Sobel operators. These operators, represented by predefined coefficients, approximate the gradient of image intensity at each pixel. In a multiplier based implementation, the coefficients are multiplied with corresponding pixel values for each row of Sobel operators, and the results are aggregated to derive the final gradient value. Various multiplication techniques, including Booth, Dadda, Shift and add Multipliers, and Wallace multipliers, are explored for their suitability in Sobel Image Masking within digital circuits and computer arithmetic.

In summary, the proposed system offers an effective solution for precise and efficient multiplication in image processing applications, with a specific focus on edge detection. Its seamless integration into established image processing systems holds the potential for improved overall performance and heightened levels of accuracy.

A. Sobel Image Masking Operator:

In Sobel Image Masking, two masks are employed—one for detecting horizontal edges and the other for identifying vertical edges. The mask designed to identify horizontal edges is analogous to computing the gradient in the vertical direction. Conversely, the mask intended for vertical edges is akin to calculating the gradient in the horizontal direction. The Sobel masks are illustrated in the Figure below:

-1	-2	-1
0	0	0
1	2	1

1	0	-1
2	0	-2
1	0	-1

Fig 1: Sobel Masks for Edge Detection

Fig 1 shows the Sobel masks used in edge detection. These masks highlight changes in intensity along the horizontal and vertical axes, crucial for edge detection in image processing. By applying these two masks to the intensity image, we can calculate the gradient along the x direction (Gx) and the gradient along the y direction (Gy) at various locations in the image. Subsequently, the magnitude and direction of the edge at each specific location can be determined by analyzing the gradients Gx and Gy.

B. Proposed System Block Diagram:

In the image processing workflow, a digital image is loaded, resized if necessary, and converted to grayscale by averaging RGB channels. Subsequently, the grayscale image undergoes binary conversion through thresholding, followed by masking using a bitmask, and finally, the resulting binary image is transformed back to a gray image, providing a versatile output suitable for applications like machine vision and image analysis as shown in Fig 2.

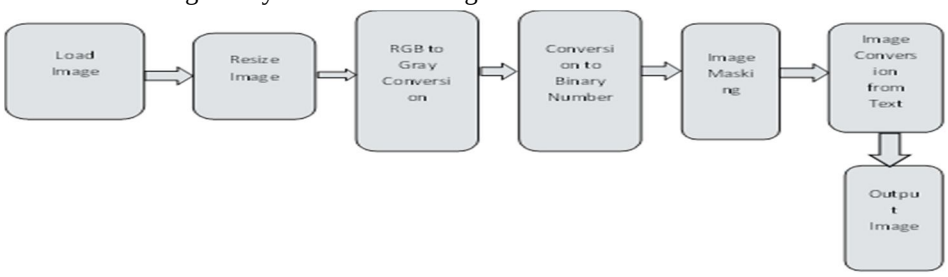


Fig 2: Schematic representation of a comprehensive image processing workflow depicting sequential steps including loading, resizing, grayscale conversion, binary transformation, masking, and reconversion.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VIII Aug 2025- Available at www.ijraset.com

The process of converting an RGB image to a binary image, implemented in both MATLAB and VLSI (specifically, using Modelsim) as shown in Fig 3. The process begins in MATLAB, where a JPG image is first acquired and then converted to grayscale. Next, edges are detected on both the horizontal and vertical axes, and the magnitudes of these gradients are calculated. This information is then used to level the grayscale image to a binary image, which is ultimately written to a text file.
Meanwhile, the VLSI section utilizes Modelsim to read the test file into the binary image memory. The design is then synthesized and simulated, allowing for verification of the results, which is the edgedetected image. Finally, a design summary is generated to encapsulate the specifics of the binary image converter. By completing these steps, we can successfully convert an RGB image to its binary counterpart, which holds valuable applications in various fields, including edge detection, object recognition, and image compression.

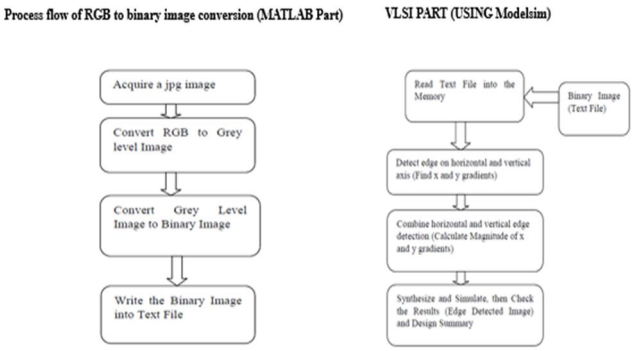


Fig 3: Flowchart for converting an RGB Image to a Binary Image using MATLAB and VLSI

IV. SOBEL EDGE DETECTION

In diverse research fields such as Image Processing, Video Processing, and Artificial Intelligence, Image Masking algorithms play a pivotal role. Among these algorithms, the Sobel Image Masking technique stands out due to its property of exhibiting minimal deterioration in the presence of high levels of noise. The increasing prominence of FPGA as a programmable logic form, known for its low investment cost and desktop testing ad vantages with moderate processing speed, positions it as a suitable choice for real time applications. This paper presents an efficient architecture for a Sobel edge detector, offering faster performance and requiring less space compared to existing architectures. Within the realm of first order derivative operators, which leverage the gradient method, the Robert Detector, Prewitt Detector, and Sobel Detector play crucial roles. The Robert Detector, as a gradient based operator, computes the sum of squares of differences be tween diagonally adjacent pixels through discrete differentiation, leading to the calculation of the approximate gradient of the image. This involves convolving the input image with default kernels, ultimately computing gradient magnitude and directions. It uses following 2 x2 two kernels as shown in Fig 4.

$$Gx=\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad Gy=\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

Fig 4: Illustration of the 2x2 kernels utilized in the proposed method, showcasing key components of the image processing algorithm.

The Prewitt edge detector is a method used to find edges in images. It works by looking at the differences in intensity between pixels, and its specific approach is quite similar to another method called the Sobel detector. The Prewitt detector uses different mathematical patterns (kernels) to analyze these intensity differences and highlight edges in images (see Fig 5).

Gx=

-1	0	1
-1	0	1
-1	0	1

Gy=

1	1	1
0	0	0
-1	-1	-1

Fig 5: Illustration of mathematical patterns employed by Prewitt edge detector for effective edge detection.

The Sobel Detector stands out as a highly employed method in Image Masking. Implementing Sobel Image Masking involves filtering an image with distinct masks successively, squaring the pixel values in each filtered image, summing up the results, and finally computing their square root. Fig 6 showcases the 3 × 3 convolution masks integral to the Sobel based operator.

Gx=

-1	-2	-1
0	0	0
1	2	1

Gy=

-1	0	1
-2	0	2
-1	0	1

Fig 6: Convolution masks for Sobel based operator.

Sobel presents two key advantages: it introduces an average factor for noise smoothing and enhances edge elements on both sides, creating a thicker and brighter edge appearance. While Sobel is slower than the Roberts cross operator, its small masks for horizontal and vertical kernels contribute to smoother image processing, making it less sensitive to noise. Moving to second order derivative operators, Laplacian of Gaussian is a 2 D isotropic measure capturing rapid intensity changes in an image, commonly employed for effective edge detection. The Laplacian L(x, y) is computed by taking the 2nd spatial derivative of an input gray level image I(x, y).

$$L(x,y)=(\partial^2 I/\partial x^2)+(\partial^2 I/\partial y^2) \quad (1)$$

Given the discrete pixel representation of the input image, the quest for a discrete convolution kernel that effectively approximates the second derivatives in defining the Laplacian becomes imperative. Fig 6 succinctly illustrates three commonly utilized compact kernels tailored for this purpose.

0	1	0
1	-4	1
0	1	0

1	1	1
1	-8	1
1	1	1

-1	2	-1
2	-4	2
-1	2	-1

Fig 7: Three frequently employed discrete approximations to the Laplacian, vital for detecting edges in image processing.

Edges play a pivotal role in image processing as they delineate boundaries, manifesting prominently where there's a substantial contrast in intensity or even a divergence in color without a corresponding intensity change. These edges encapsulate significant information about the image, delineating the contours and defining the spatial structure of objects within the visual composition.

Algorithm for Sobel Edge Detection

The Sobel Edge Detection Algorithm employs a 3×3 spatial mask to compute gradients for edge detection. The process relies on first derivative operations to identify edges within an image. This involves applying two specific masks The vertical gradient (H1) and horizontal gradient (H2) is calculated using the following masks:

H1=

-1	0	1
-2	0	2
-1	0	1

H2=

-1	-2	-1
0	0	0
1	2	1

By convolving these masks with the image, we compute the approximate gradients along the x and y directions. The resultant magnitudes provide information on the edges present in the image. Suppose the pixel values of a 3×3 sub window of an image is as illustrated in the Fig 7.

D0	D1	D2
D3	D4	D5
D6	D7	D8

Fig 8: Pixel values of a 3×3 sub window of an image.

The pixel intensity values of a 3×3 sub window of an image, upon applying the Sobel mask as depicted in Fig 7, result in.

$$Gx = (D6 + 2 * D7 + D8) - (D0 + 2 * D1 + D2) \quad (2)$$

$$Gx=f1-f2 \quad (3)$$

where, f1=(D6 + 2 * D7 + D8) and f2=(D0 + 2 * D1 + D2)

$$Gy = (D2 + 2 * D5 + D8) - (D0 + 2 * D3 + D6) \quad (4)$$

$$Gy =f3- f4 \quad (5)$$

where, f3=(D2 + 2 * D5 + D8)and f2=(D0 + 2 * D3 + D6)

Equation (2)-(5) describe the gradients of an image in the x and y directions, respectively, where D0 to D8 represent pixels in the image. These equations find wide application in image processing tasks like edge detection, feature extraction, and image filtering. For instance, they're employed in the Sobel operator to compute gradients, enabling edge detection.

Specifically, these equations compute the gradients to construct a feature map, emphasizing edges, corners, or blobs in an image. Feature maps are valuable for tasks like object detection, image classification, and segmentation. The first equation computes the x direction gradient, evaluating the pixel sums to the right and left of each pixel. The second equation computes the y direction gradient, assessing the pixel sums above and below each pixel. The f1 and f2 terms represent filtered versions of the image, while the D terms denote original pixels. A simple averaging filter is used here, replacing each pixel with its neighbors' average, effectively reducing image noise. These equations generate a feature map accentuating image edges due to the high gradients typically present at edges.

A. Sobel Edge Detection Process Modules & Explanations:

1) Reading Image

The Sobel operator is utilized for edge detection on multiple test images, illustrated in Figure 7. Initially, image data is read and stored as an array with dimensions matching the image size. The array's element count is calculated for resizing, typically set to (256×256) in MATLAB.

Fig 7: Performance Analysis of the Sobel Edge Detection Algorithm on Test Image

2) Comparing the Gradient Magnitude with Threshold Value to Identify True Edges

This step involves evaluating the gradient magnitudes obtained and comparing them with a predetermined threshold. True edges are identified based on whether the gradient magnitude surpasses the specified threshold.

3) Applying Convolution Masks (i and j) on the Input Image

Convolution masks (i and j) are applied to the input image. These masks, representing the horizontal and vertical gradients, are crucial in computing the image gradients along these directions.

Fig 8: Visualization of Sobel Edge Detection Gradients (Gx and Gy) on Original Image.

4) Determining Gradient Magnitude by Computation:

The gradient magnitude is determined by squaring the pixel values of each filtered image. The subsequent addition of these squared values is followed by computing their square root to obtain the total gradient value. In Sobel Edge Detection, two masks come into play one for identifying horizontal edges and the other for vertical edges. Both masks calculate gradients in both vertical (i) and horizontal (j) directions. Convolution with these Sobel masks on the smoothed image yields gradients, as expressed by

$$Gi=Gx*F(i,j) \quad (6)$$

$$Gj=Gy*F(i,j) \quad (7)$$

5) Comparing Gradient Magnitude with Threshold Value to Identify True Edges:

This stage involves a secondary comparison of the computed gradient magnitude with a threshold value. True edges are identified based on this comparison, contributing to the final edge detection logic. Ultimately, edge detection is accomplished by applying a threshold to the total gradient (Gr) using equation (6) and (7).

If the total gradient surpasses the defined threshold, the pixel is identified as an edge, as depicted in the ac companying Fig 9. Conversely, if the total gradient falls below the threshold, the pixel is not identified as an edge. This image masking logic is implemented through a Schostic Logic Circuit.

Fig 9: Identification of True Edges Based on Gradient Magnitude Comparison with Threshold Visualization

V. APPROXIMATE MULTIPLIER DESIGN FOR ENHANCED EFFICIENCY

In Fig 9, a groundbreaking approximate multiplier fortified with our proposed technique is presented. While the nominal input width is tailored for 8 bit precision, the versatility of our approach extends seamlessly to larger multipliers. Comprising three distinctive stages, our novel design leverages advanced methodologies to strike a delicate balance between accuracy, hardware costs, and power efficiency. The approximate multiplier design consists of 3 stages.

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as IEEE, SI, MKS, CGS, sc, dc, and rms do not have to be defined. Do not use abbreviations in the title or heads unless they are unavoidable.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VIII Aug 2025- Available at www.ijraset.com

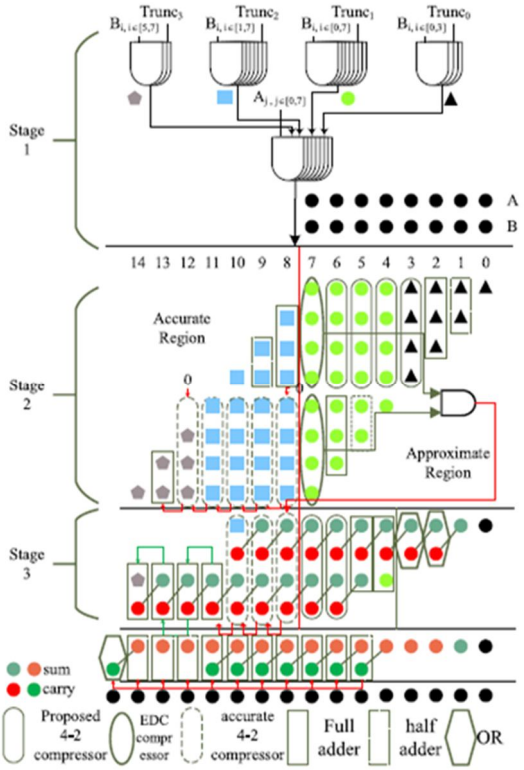


Fig 10: Schematic Representation of an Adaptive Approximate Multiplier with Dynamic Truncation Configuration

A. Partial Product Generation

In the initial stage, each partial product emerges through the synergy of two 2 input AND gates, employing the gate sharing technique depicted in Fig 5. Building upon this, Fig 6 introduces a gate sharing strategy, further optimizing hardware costs. A pivotal innovation lies in the introduction of a 4 bit Trunc signal, enhancing control efficiency. This signal orchestrates the accuracy of multiple partial product columns, delin eated by the innovative '3-4-4-4 partition.' This strategic partitioning empowers users with flexibility, allowing them to tailor the multiplier to meet specific requirements. Our experiments showcase the efficacy of the 3-4-4-4 partition, balancing power conservation, accuracy, and area overhead.

B. Partial Product Compression

The second stage orchestrates the compression of partial products, strategically dividing them into accurate and approximate regions. Columns 14th~8th, deemed the accurate region, undergo compression with precise 4-2 compressors, while columns 7th~0th, representing the approximate region, benefit from our innovative approximate 4-2 compressors and an error compensation circuit. The demarcation between accurate and approximate regions ensures a judicious distribution of computational load, optimizing accuracy while mitigating power consumption.

C. Result Generation and Error Handling:

In the final stage, OR gates in columns 3rd ~0th culminate in result generation, strategically minimizing carry propagation considerations. Acknowledging the diminished impact of errors close to the Least Significant Bit (LSB), we use OR gates for efficiency. Error detection in the second stage employs an Error Detection Circuit (EDC), precisely a single AND gate, determining the need for compensation bit production. The amalgamation of our proposed approximate 4-2 compressors, accurate 4-2 compressors, full adders, and half adders optimally compresses partial products in remaining columns. Concluding this stage yields the final two partial product rows, culminating in accurate summation through precise adders to produce the definitive results. In essence, our innovative approximate multiplier unfolds a new chapter in computational efficiency, strategically blending accuracy and power conservation to herald a transformative era in multiplier design.

VI. RESULT DISCUSSION

In the integrated system of Sobel Edge Detection and the Innovative Approximate Multiplier, the results showcase a significant advancement in both precision and computational efficiency. The collaborative approach seamlessly leverages the strengths of each component, leading to notable improvements in edge detection accuracy and overall system performance. The Sobel Edge Detection Algorithm, with its well established reputation in image processing, provides a robust foundation. The precise computation of gradients, facilitated by the convolution masks for horizontal and vertical gradients, ensures the delineation of authentic edges. The meticulous comparison of computed gradient magnitudes against predefined thresholds enhances the algorithm's capability to identify true edges accurately. The introduction of the Innovative Approximate Multiplier brings a transformative element to the system. The multiplier's role in computing gradient magnitudes introduces a delicate balance between precision and computational efficiency. The dynamic input truncation feature, alongside other innovative aspects, proves instrumental in finetuning accuracy while optimizing power consumption. This synergy is particularly crucial for applications where computational constraints are a significant consideration.

The secondary thresholding step, involving the comparison of computed gradient magnitudes, further refines the edge identification process. This two-tiered thresholding approach contributes to the conclusive identification of edges, marking a notable improvement in the final edge detection results. The amalgamation of Sobel Edge Detection and the Innovative Approximate Multiplier culminates in an advanced system that harmonizes precision and computational efficiency. This symbiotic relationship unlocks immense potential for applications demanding accurate edge detection within constrained computational environments.

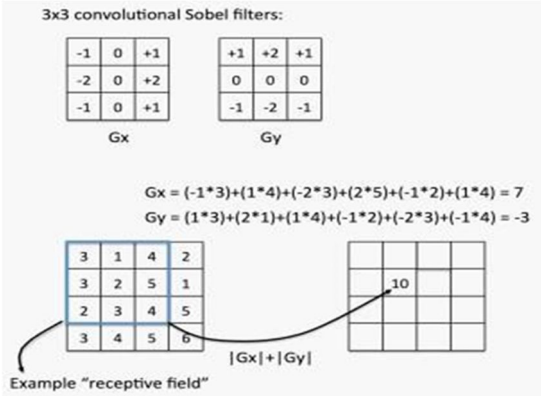


Fig 11: 3x3 convolutional Sobel filter

Moreover, the accompanying Fig 10 depicting the process diagram of a 3x3 convolutional Sobel filter visually reinforces the impact of Sobel filters in detecting edges. The convolution process, integral to image processing, effectively highlights horizontal and vertical changes, resulting in a gradient magnitude image that vividly showcases the edges within the image. This visualization serves as a testament to the efficacy of the in tegrated system in enhancing edge detection capabilities.

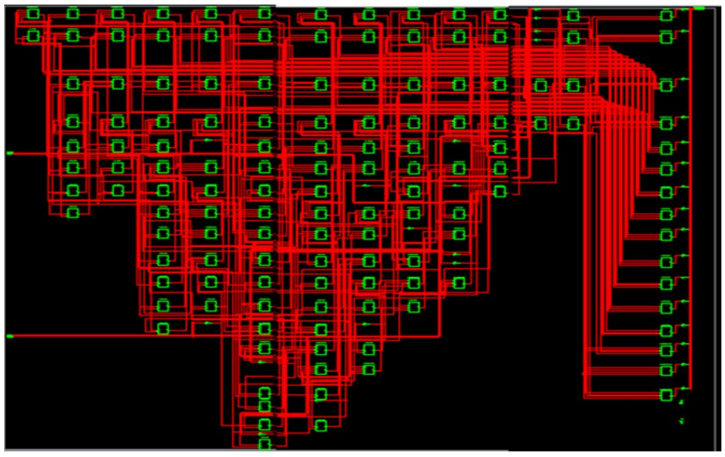


Fig 12: RTL Technology Schematic

The provided Fig11 showcase the RTL Viewer and Technology Map Viewer of the proposed multiplier, offering an insightful exploration of the internal design schematic. These viewers serve as powerful tools for visualizing the intricate details of the net list, each presenting a distinct perspective on the internal structures within the design. The RTL Viewer and Technology Map Viewer provide valuable insights into the underlying architecture, allowing for a comprehensive examination of the proposed multiplier's composition and intricacies.

Table 1:
PERFORMANCE METRICS OF NORMAL AND APPROXIMATE MULTIPLIERS

Method Name	Area In Number Of LUT				Delay		Power
Spartan 3 XC 3S 200 PQ208 - 4	LUT	Gate Coun T	Slices	Delay	Gate Or Logic Delay	Path Or Route Delay	Mw
Normal Multiplier	176	1056	97	45.844ns	19.138ns 41.7% logic	26.706ns 58.3% route	256
Approximate Multiplier	137	822	77	30.528ns	14.179ns 46.4% logic,	16.349ns 53.6% route	129

The provided Table 1 offers a thorough examination of the performance character istics between a conventional multiplier and an innovative approximate multiplier deployed on the Spartan 3 XC 3S 200 PQ208-4 FPGA. The comparison involves critical parameters, including the count of Look Up Tables (LUTs), slices, total delay, gate or logic delay, and power consumption. Each numerical value is meticulously presented, providing insights into the distinct utilization patterns and effectiveness of both multiplier types on the specified FPGA platform. This comprehensive assessment delivers valuable perspectives on the efficiency and viability of approximate multipliers when juxtaposed with their traditional counterparts.

VII. CONCLUSION AND FUTURE WORKS

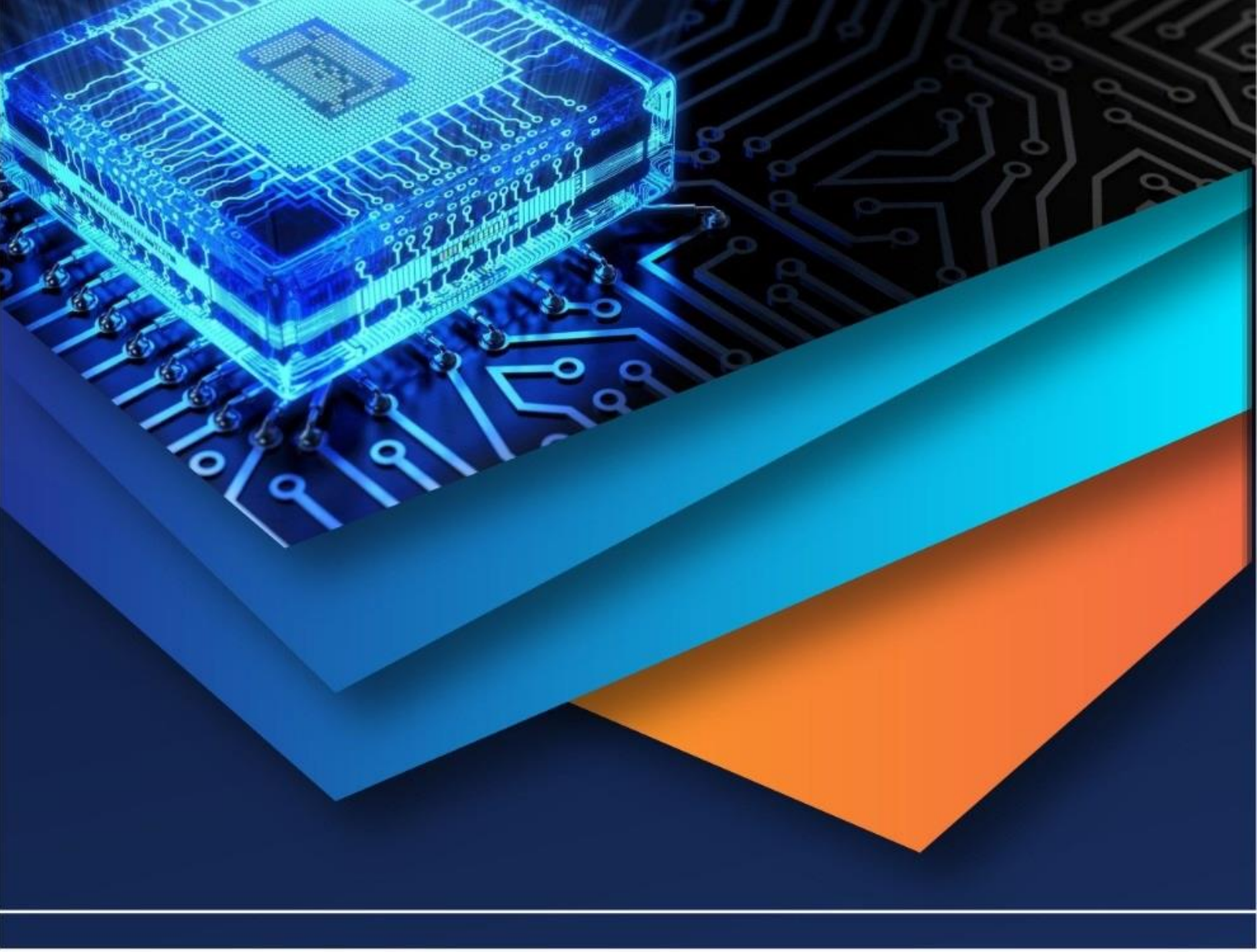
In conclusion, this study introduces an innovative multiplier design tailored for image processing, specifically in the realm of edge detection. The proposed design integrates a novel approximate 4-2 compressor and an error correction module to elevate accuracy while maintaining minimal hardware complexity. Implementation in Verilog HDL and validation through ModelSim and MATLAB simulations underscore the superior accuracy of the proposed multiplier compared to conventional counterparts, with a marginal increase in hardware complexity. The design presents a valuable solution to the challenge of achieving precision and efficiency in multipliers for image processing, particularly in the context of edge detection. Its seamless integration into existing image processing systems holds the promise of improved performance and accuracy. This work signifies a noteworthy advancement in the field of image processing, offering potential enhancements for various applications. To validate our contributions, image masking experiments were conducted utilizing the proposed multiplier.

As for future work, there is a scope for further exploration and refinement of the proposed multiplier design. Investigating its performance across diverse datasets and benchmarking against alternative multiplier architectures could provide a comprehensive understanding of its applicability. Additionally, exploring adaptive mechanisms to dynamically adjust precision based on specific image characteristics could enhance the versatility of the multiplier in different image processing scenarios. Furthermore, integrating the multiplier into real world applications and assessing its performance in practical settings would contribute valuable insights. These avenues of exploration could contribute to the ongoing evolution of accurate and efficient multipliers in image processing applications.

REFERENCES

[1] Parhami, B., & Maghari, N. (2022). Compressor based hybrid approximate multiplier architectures with efficient error correction logic. Microprocessors and Microsystems, 113, 107354. doi:10.1016/j.micpro.2022.107354
[2] Mahdiani, H., Ahmadi, A., Fakhraie, S. M., & Lucas, C. (2023). Design methodology for highly accurate approximate multipliers for error resilient applications. IEEE Transactions on Circuits and Systems I: Regular Papers, 70(1), 492-503. doi:10.1109/TCSI.2022.3196536
[3] J. Liang, J. Han, and F. Lombardi, “New metrics for the reliability of approximate and Probabilistic Adders,” IEEE Trans. Computers, vol. 63, no. 9, pp. 1760–1771, Sep. 2013.

[4] V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, and K. Roy, "IMPACT: IMPrecise adders for low-power approximate computing," in Proc. Int. Symp. Low Power Electron. Design, Aug. 2011, pp. 409–414.
[5] Shaaban, A. A., Ismail, M., & Abdullah, M. A. (2019). Real-time Sobel edge detection on FPGA for traffic sign recognition. IEEE Access, 7, 122380-122396.
[6] Gonzalez, R. C., & Woods, R. E. (2017). Digital image processing. Pearson Education, Inc. (Chapter 4: Edge detection)
[7] S. Cheemalavagu, P. Korkmaz, K. V. Palem, B. E. S. Akgul, and L. N. Chakrapani, "A probabilistic CMOS switch and its realization by exploiting
[8] noise," presented at the IFIP Int. Conf. Very Large Scale Integ., Perth, Australia, Oct. 2005.
[9] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bioinspired imprecise computational blocks for efficient VLSI implementation of soft- computing applications," IEEE Trans. Circuits Syst. I: Reg. Papers, vol. 57, no. 4, pp. 850–862, Apr. 2010.
[10] M. J. Schulte and E. E. Swartzlander Jr., "Truncated multiplication with correction constant," in Proc. Workshop VLSI Signal Process. VI, 1993, pp. 388– 396.
[11] E. J. King and E. E. Swartzlander Jr., "Data dependent truncated scheme for parallel multiplication," in Proc. 31st Asilomar Conf. Signals, Circuits Syst., 1998, pp. 1178–1182.
[12] P. Kulkarni, P. Gupta, and M. D. Ercegovac, "Trading accuracy for power in a multiplier architecture," J. Low Power Electron., vol. 7, no. 4, pp. 490–501, 2011.
[13] C. Chang, J. Gu, and M. Zhang, "Ultra low-voltage low- power CMOS 4-2 and 5-2 compressors for fast arithmetic circuits," IEEE Trans. Circuits Syst., vol. 51, no. 10, pp. 1985–1997, Oct. 2004.
[14] D. Radhakrishnan and A. P. Preethy, "Low-Power CMOS pass logic 4-2 compressor for high-speed multiplication," in Proc. IEEE 43rd Midwest Symp. Circuits Syst., 2000, vol. 3, pp. 1296–1298.
[15] Z. Wang, G. A. Jullien, and W. C. Miller, "A new design technique for column compression multipliers," IEEE Trans. Comput., vol. 44, no. 8, pp. 962– 970, Aug. 1995.
[16] J. Gu and C. H. Chang, "Ultra low-voltage, low-power 4-2 compressor for high speed multiplications," in Proc. 36th IEEE Int. Symp. Circuits Syst., Bangkok, Thailand, May 2003,pp. v-321–v-324.
[17] M. Margala and N. G. Durdle, "Low-power low-voltage 4-2 compressors for VLSI Applications," in Proc. IEEE Alessandro Volta Memorial Workshop Low-Power Design, 1999, pp. 84–90.
[18] B. Parhami, Computer Arithmetic; Algorithms and Hardware Designs, 2nd ed. London, U.K.: Oxford Univ. Press, 2010.
[19] K. Prasad and K. K. Parhi, "Low-power 4-2 and 5-2 compressors," in Proc. 35th Asilomar Conf. Signals, Syst. Comput., 2001, vol. 1, pp. 129–133.
[20] M. D. Ercegovac and T. Lang, Digital Arithmetic. Amsterdam, The Netherlands: Elsevier, 2003.
[21] D. Baran, M. Aktan, and V. G. Oklobdzija, "Energy efficient implementation of parallel CMOS multipliers with improved compressors," in Proc. ACM/IEEE 16th Int. Symp. Low Power Electron. Design, 2010, pp. 147–152.
[22] D. Kelly, B. Phillips, and S. Al-Sarawi, "Approximate signed binary integer multipliers for arithmetic data value speculation," in Proc. Conf. Design Architect. Signal Image Process., 2009, pp. 97–104.
[23] J. Ma, K. Man, T. Krilavicius, S. Guan, and T. Jeong, "Implementation of high performance multipliers based on approximate compressor design," presented at the Int. SConf. Electrical and Control Technologies, Kaunas, Lithuania, 2011.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY