International Journal of Computational and Experimental Science and ENgineering (IJCESEN)

Vol. 11-No.2 (2025) pp. 1716-1726
http://www.ijcesen.com

ISSN: 2149-9144

Copyright © IJCESEN

Research Article

Enhanced Design of Resource-Efficient Approximate Multipliers with Truncation and Error Compensation for Multimedia Applications

Talla Srinivasa Rao[1]*, Ch Srinivasu[2], K. Babulu[3]

[1]Department of Electronics and Communication Engineering, Jawaharlal Nehru Technological University Gurajada Vizianagaram (JNTU-GV), Vizianagaram, Andhra Pradesh, India
* **Corresponding Author Email:** srinivasarao.t@adityauniversity.in - **ORCID:** 0000-0002-6472-1259

[2]Department of Electronics and Communication Engineering, Raghu Engineering College Visakhapatnam, Andhra Pradesh, India
Email: principal@raghuenggcollege.in - **ORCID:** 0009-0005-9000-8771

[3]Department of Electronics and Communication Engineering, Jawaharlal Nehru Technological University Gurajada Vizianagaram (JNTU-GV), Vizianagaram, Andhra Pradesh, India
Email: kapbbl.ece@jntugvcev.ac.in - **ORCID:** 0000-0003-1097-9390

Article Info:

DOI: 10.22399/ijcesen.485
Received : 07 October 2024
Accepted : 21 March 2025

Keywords :

8x8 Approximate multiplier,
Higher-order compressors,
Error compensation,
Scalable 16x16 multiplier.

Abstract:

Approximate computing offers an effective strategy for reducing resource utilization in multiplier design, particularly in applications where tolerance for errors is essential, such as multimedia and signal processing. This paper presents a systematic approach for designing an efficient 8x8 approximate multiplier by integrating truncation, approximation, and exact computation at various stages of partial product generation. This approach employs approximate compressors to streamline partial product decomposition, achieving significant reductions in hardware complexity, power consumption, and latency while maintaining acceptable accuracy levels. Additionally, error compensation techniques are applied to minimize discrepancies arising from approximations and truncations. A key innovation is the use of single-stage decomposition with higher-order compressors, which reduces operating time by handling all partial product decompositions in a single stage. For scalability, a 16x16 multiplier is constructed using a four-row parallel arrangement of proposed 8x8 multipliers, combining exact and approximate designs for enhancing efficiency. The proposed 16x16 multiplier design achieves a 30% reduction in delay, 15% lower power consumption, a 40.3% reduction in Power-Delay Product (PDP), and occupies 14.4% less area compared to the nearest existing designs, underscoring its superior efficiency and performance. The proposed design also significantly outperforms existing designs in error analysis metrics, showcasing improved accuracy and quality while ensuring high efficiency. This design methodology offers an optimal balance between accuracy and resource efficiency, making it an effective solution for resource-constrained, error-tolerant applications.

1. Introduction

In contemporary digital systems, especially those involved in multimedia processing, signal processing, and other computationally intensive applications, the efficiency of hardware components is crucial. The ever-increasing demands for higher performance and lower power consumption necessitate innovative approaches to multiplier design. Approximate computing, a paradigm that accepts a controlled level of inaccuracy to gain resource efficiency, presents a compelling solution. This paper explores a novel systematic approach to designing an approximate multiplier, specifically an 8x8 multiplier, that balances accuracy with resource optimization. Traditional exact multipliers, while reliable, come with significant drawbacks in terms of power consumption, hardware complexity, and latency. In high-precision applications, these exact methods require extensive resources, which can be impractical as data scales and processing speeds increase. Approximate computing offers a promising alternative by relaxing precision requirements to achieve reductions in hardware and power costs,

making it particularly suitable for applications where some level of error is permissible. The core challenge in designing approximate multipliers lies in managing the optimal balance between accuracy and resource efficiency. Multipliers are integral to many arithmetic operations in digital circuits, involving the generation and summation of partial products. Traditional designs achieve exact results but at the cost of increased hardware complexity and power consumption. Approximate designs aim to simplify this process, but careful consideration is needed to ensure that the introduced approximations do not exceed acceptable error thresholds for practical applications. This paper proposes a systematic approach to designing an 8x8 approximate multiplier that incorporates truncation, approximation, and exact computation across different stages of partial product generation. By utilizing approximate compressors, the proposed architecture streamlines the partial product decomposition process, thereby reducing hardware complexity and power consumption. The integration of error compensation techniques further ensures that the errors introduced by approximations and truncations are minimized, maintaining acceptable accuracy levels. A notable innovation in this approach is the use of single-stage decomposition with higher-order compressors. Traditional multipliers often require multiple stages of partial product generation and summation, which can increase latency and hardware requirements. The proposed design addresses this issue by consolidating all partial product decompositions into a single stage, thus reducing operating time and enhancing performance. To extend the applicability of the design, a 16x16 multiplier is developed by cascading two optimized 8x8 multipliers. This method leverages the efficiency of the 8x8 design while scaling to larger data sizes. The combination of exact and approximate components in the 16x16 multiplier ensures enhanced efficiency and performance. The proposed architecture is implemented using Verilog, a hardware description language widely used for designing and simulating digital circuits. Hardware verification is conducted using Cadence tools on 90nm technology, providing a robust platform for validating the design's functionality and performance. The use of Verilog allows for precise and flexible design specifications, while Cadence technology ensures that the design is thoroughly verified and optimized for modern hardware constraints. The main objective of this work as follows:

1. Design an efficient 8x8 approximate multiplier incorporating a single stage of partial product decomposition with higher-order compressors (8:2, 7:2, 6:2, and 5:2) to reduce latency. This design integrates truncation at the least significant bit (LSB) position, approximation in the middle stages, and exact computation at the most significant bit (MSB) position to balance accuracy and efficiency. Additionally, error compensation circuitry is included to mitigate inaccuracies introduced by approximations and truncations.
2. Develop a scalable 16x16 multiplier by cascading our proposed 8x8 multipliers, combining exact and approximate techniques to enhance overall efficiency and performance in larger-scale applications.
3. Verify the performance of the proposed 16x16 multiplier by implementing it in the image processing application for image edge detection. Assess the effectiveness of the multiplier by calculating performance metrics such as Normalized Mean Error Deviation (NMED), Mean Relative Error Difference (MRED), Peak Signal-to-Noise Ratio (PSNR), and Number of Effective Bits (NoEB), and Mean Structural Similarity Index Measure (MSSIM).

The structure of the paper is outlined as detailed below: Section II offers a concise review of pertinent literature. Section III explores recent related studies. Details of the proposed architectures and their implementation are covered in Section IV. Section V describes the findings and validations of the research. The paper wraps up with a summary and discussion in Section VI. The Bibliography section provides a list of references.

2. Related Work

This section examines recent progress and strategies in the design of approximate multipliers, with a special emphasis on innovations in 4:2 compressors and their effects on multiplier performance. It provides an overview of significant advancements in partial product reduction, focusing on the shift from exact to approximate methods to balance accuracy, power efficiency, and hardware complexity. The discussion also includes new approaches for integrating error correction and optimization within 8x8 multipliers, featuring a detailed analysis of a recently developed structure for the 8x8 approximate multiplier.

2.1 Literature Review

The article provides a comprehensive review and evaluation of approximate arithmetic circuits, examining their characteristics through functional simulation, performance and area-optimized circuit synthesis, and their use in image processing and machine learning applications [1]. Innovative designs for two approximate 4-2 compressors are

introduced and applied in the reduction modules of four approximate multipliers. These compressors demonstrate considerable reductions in transistor count, power consumption, and delay compared to exact designs [2]. The research presents an approximate Wallace-Booth multiplier that offers notable improvements in power efficiency and delay reduction, accompanied by a slight trade-off in accuracy. The design utilizes approximate Booth encoders, approximate 4-2 compressors, and approximate trees within the lower n/2 least significant bits (LSBs) of the multiplier [3]. Four Dual Quality 4:2 Compressors are designed to switch between exact and approximate operating modes. In approximate mode, these compressors achieve higher speeds and reduced power consumption, with a trade-off in accuracy [4]. A high-speed, low-power approximate multiplier is developed, utilizing an innovative radix-4 partial product generation technique and a new inexact 4-2 compressor. Its effectiveness is assessed through image processing tasks, such as sharpening and smoothing, showing that it can be applied to these applications while preserving overall image quality [5]. A novel 4-2 compressor design is introduced for the partial product compression phase in binary multipliers. This design involves a minor adjustment to an existing 4-2 compressor, resulting in errors that consistently produce slightly reduced values compared to exact results. This predictable error pattern allows for easy detection using a 2-input AND gate. An error recovery module can be employed to address these errors if necessary. The approach supports the development of an approximate multiplier with separate accurate and approximate regions, utilizing the new compressor in the approximate region and applying error recovery modules at the most significant bit positions [6]. The proposed approximate compressors utilize only AND-OR gates, avoiding the need for XOR gates, and demonstrate improved precision and reduced hardware complexity compared to previous designs. An algorithm is introduced to incorporate these compressors into approximate multipliers. The approach uses the compressors in the early stages of the partial product reduction process, applying them to the less-significant parts of the partial product matrix and integrating them into more significant parts only when required, in order to minimize the overall approximation error [7]. Various approaches to majority logic-based approximate adders and multipliers are explored. Studies highlight the design, analysis, and performance of these components. Notably, the majority logic-based full adders available in 1-bit, 2-bit, and multi-bit variants are discussed for their advantages in reducing circuit complexity and delay compared to exact designs. These approximate adders achieve these improvements with only a minimal loss in accuracy [8]. A new design for an approximate 4:2 compressor is introduced, along with a modified Dadda Multiplier architecture. This modification enhances the effectiveness of the proposed compressor and aims to reduce output errors [9]. This brief outline a method for enhancing the energy efficiency of the dynamic truncation technique used in developing energy-quality scalable multipliers. The suggested method involves an intelligent arrangement of the partial products, which minimizes energy consumption in the non-truncated sections of the multiplier. Additionally, a straightforward and effective hardware correction technique is presented, which can be adjusted based on the truncation settings [10]. Two innovative 4:2 compressor architectures are introduced. The first design emphasizes high speed and area efficiency, resulting in significant reductions in area, delay, and power consumption. The second design, a modified dual-stage architecture, further optimizes area, delay, and power while preserving accuracy [11]. An energy-efficient approximate multiplier is designed using a new 4-2 compressor combined with an error recovery module. This compressor, derived from modifications to existing advanced circuits, has a low error probability and enables easy error detection with a 3-input AND gate. It is integrated into the partial product reduction phase of the multiplier, where the error recovery module addresses any detected errors [12]. Three innovative approximate 4-2 compressors are introduced for use in 8-bit multipliers. An error-correcting module is also presented to improve the error performance of these multipliers. Additionally, the design reduces the number of outputs from the approximate 4-2 compressors to one, which enhances energy efficiency [13]. A novel approximate 5:2 compressor and a 4:1 compressor is presented. Furthermore, three approximate multipliers are developed by replacing AND gates with NAND gates and integrating different combinations of the newly proposed compressors [14]. An unsigned approximate multiplier architecture is proposed, consisting of three segments. The least significant segment is replaced with a constant compensation term to reduce hardware usage without greatly impacting accuracy. The middle segment employs a new 4:2 approximate compressor to simplify partial products, with an error correction module mitigating the resulting approximation errors. The most significant segment is implemented with exact logic to prevent substantial errors that could arise from approximation [15]. The approach involves extracting segments of mmm contiguous bits from

each n-bit operand and processing these segments with a small m× m internal multiplier, with the output then adjusted through shifting to produce the final result. The investigation covers both signed and unsigned multipliers, with a novel segmentation method proposed for the unsigned variant. Furthermore, effective correction techniques are presented to significantly reduce approximation errors while keeping hardware costs low [16]. An innovative approximate multiplier with integrated error compensation is presented. This design incorporates a region with fixed truncation, an error correction mechanism, and a precise computation segment. The lower portion of the product is treated as a constant compensation factor, while the more significant portion is computed with full accuracy to achieve a balance between hardware efficiency and precision. Furthermore, an easy-to-implement yet highly effective error correction module is included to substantially enhance accuracy [17]. A series of 4-bit approximate multipliers is introduced, along with a method for scaling these to larger bit lengths by utilizing smaller bit-width multipliers. To develop a higher bit-length multiplier, the approach decomposes it into four sub-multipliers, each with half the bit width of the target multiplier. Partial products from these sub-multipliers are then combined using several approximate techniques, each offering varying levels of accuracy [18]. Two multipliers are introduced that employ error compensation techniques and approximate 4:2 compressors to achieve lower energy consumption. The designs, termed “M00” and “M01,” offer A beneficial equilibrium between accuracy and energy efficiency. M00 reduces transistor count significantly, leading to lower energy usage while still providing adequate accuracy for areas including image processing and neural networks. Meanwhile, M01 achieves high precision with reduced energy consumption compared to other multipliers [19].

2.2 8x8 Approximate Dadda Multipliers

The 8x8 Dadda multiplier is a high-speed binary multiplication technique designed to optimize the reduction of partial products while minimizing the number of adders required as shown in Figure 1. It starts by generating an 8x8 grid of partial products through bitwise AND operations between the multiplicand and multiplier, resulting in 64 partial products. These are then reduced in stages using full adders, which take three inputs, and half adders, which take two inputs, to progressively condense the number of rows. At each stage, the number of bits per column is minimized to 1, 2, or 3, effectively reducing the rows until only two remain. In the final stage, these two rows are summed using a fast adder, such as a carry-lookahead or ripple carry adder, to produce the final product. This method is highly efficient because it reduces the number of addition stages compared to alternative methods like the Wallace tree, resulting in faster overall computation.

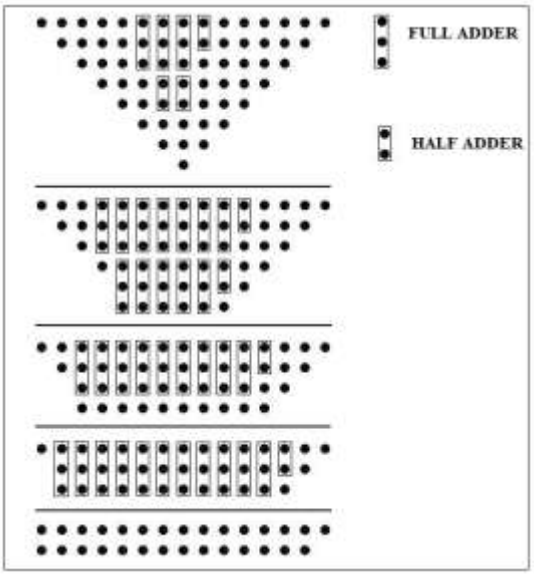

Figure 1. *Basic structure of 8x8 Dadda approximate multiplier.*

The key step is the reduction of partial products. In the early stages of research, half adders and full adders were used for this decomposition process. While this method yields accurate results, it incurs high resource utilization, including significant power consumption, area usage, and delay. To address this, 4:2 compressors were introduced to reduce resource utilization to some extent. However, the utilization remains high, and such a high degree of accuracy is unnecessary for error-resilient applications like multimedia and image processing. At this stage, approximate compressors, were employed and this approach significantly reduces hardware requirements, power consumption, and latency. However, the trade-off is the introduction of errors, leading to reduced accuracy. To mitigate this, approximate designs are used only in the least significant positions of partial products, while exact compressors and adders are reserved for the most significant bits. This strategy helps in decomposing the higher-order partial products while balancing accuracy and hardware efficiency. Further research introduced innovative methods to enhance both hardware utilization and accuracy. For example, researchers have integrated techniques like truncation in the least significant stages, approximation in the middle stages, and exact designs at higher levels. Error compensation techniques were also introduced to recover accuracy without increasing resource utilization. A recently

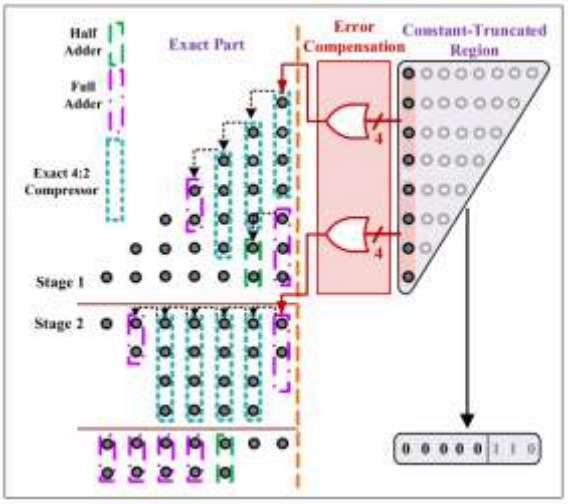

Figure 2. *Recently developed 8x8 approximate multiplier*

developed 8X8 approximate multiplier as shown in Figure 2. It shows a new multiplier design with three main parts: a fixed-truncation region, a fault mitigation system, and an exact computation unit. This design improves performance by skipping partial product generation in the eight least significant columns of the constant region and using a fixed 8-bit product "00000110" instead. The error correction module plays a key role by fixing errors in the left part of the fixed-truncation region. It uses two 4-input OR gates to detect and correct errors in high-probability situations. The module's output is then used as a Carry input for the next stage of compressors, which helps to address negative error distances (EDs). This is particularly useful for applications involving neural networks, where the ReLU activation function is common. To balance efficiency and accuracy, the design replaces the three least significant bits with a constant correction term "110". This term, which is based on averaging input sets in the lower order columns, enhances accuracy while simplifying the hardware needed. The reduction part of the multiplier uses a two-stage process. In the first stage, a half adder, two full adders, and three 4:2 exact compressors are used. The second stage includes two full adders and four 4:2 exact compressors. The final result is produced by a Ripple Carry Adder (RCA), which consists of two half adders and four full adders.

3. Proposed Work

In this paper, proposes novel methods for designing approximate multipliers. In previous designs, partial products were decomposed in multiple stages, leading to increased latency. In contrast, the proposed design decomposes all partial products in a single stage using higher-order compressors such as 8:2, 7:2, 6:2, and 5:2 in the design of 8X8 approximate multipliers. This hybrid method integrates truncation at the least significant positions, approximation in the middle stages, and exact circuits at the higher-order positions of partial products. An effective error recovery circuit is also employed to compensate for errors arising from both truncation and approximation. Additionally, to design higher-order multipliers from lower-order ones, a scaling method is utilized, which involves cascading lower-order multipliers in various combinations. It is detailed by designing 16X16 multiplier by using 8X8 multiplier. These innovative and effective modifications to existing designs result in a tremendous improvement in terms of balancing hardware utilization and accuracy. Hardware utilization is significantly reduced without compromising accuracy.

3.1 Design of proposed compressors

Compressors are circuits designed to sum multiple binary inputs with fewer operations compared to traditional adders. Common examples include 3:2 compressors (similar to a full adder) and 4:2 compressors. The traditional exact compressor, reduces four inputs to two using two exact full adders, which consume significant resources and time to perform the operation. In contrast, the approximate 4:2 compressor is designed using two approximate full adders, with the logic diagram of the proposed full adders as shown in Figure 3. The Boolean expressions of Sum and Carry-outs for the proposed full adder are provided in equation 1, and those for the proposed half adder are given in equation 2.

$$Sum = A + B + C_{in} \qquad Cout = C_{in} \qquad (1)$$

$$Sum = A + B \qquad Cout = C_{in} \qquad (2)$$

Figure 4. illustrates the design of both exact and approximate 5:2 compressors. The exact 5:2 compressor is constructed using an exact 4:2 compressor along with an exact full adder. Similarly, the approximate 5:2 compressor is built using an approximate 4:2 compressor paired with an approximate full adder. The exact 6:2 compressor was designed using two exact 4:2 compressors, whereas the approximate 6:2 compressor was designed using two approximate 4:2 compressors and one approximate full adder, as illustrated in Figure 5. Two compressors of 6:2 and 4:2 is sufficient and no need of full adder. The exact 7:2 compressor was created by combining one exact 5:2 compressor and one exact 4:2 compressor, whereas the approximate 7:2 compressor was created by

combining one approximate 5:2 compressor and one approximate 4:2 compressor, as illustrated in Figure 6. The exact 8:2 compressor designed by using one exact 6:2 compressor and one exact 4:2 compressor whereas the approximate 8:2 compressor designed by using one approximate 6:2 compressor and one approximate 4:2 compressor as shown in Figure 7.

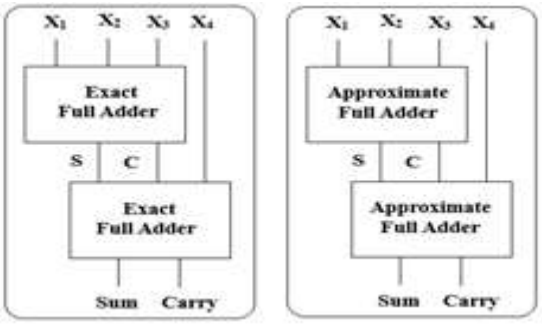

Figure 3. *Exact and Inexact 4:2 compressor*

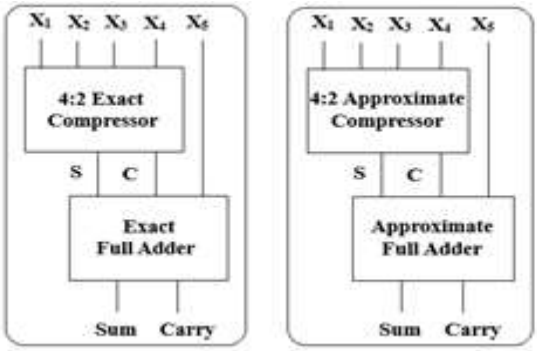

Figure 4. *Exact and Inexact 5:2 compressor*

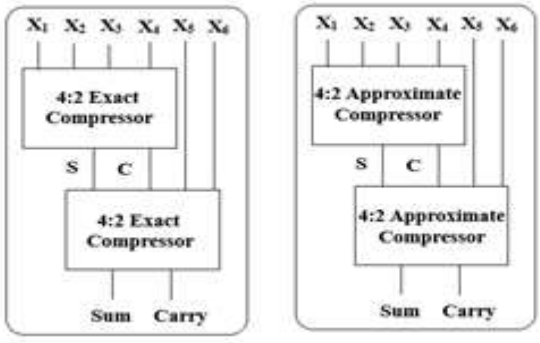

Figure 5. *Exact and Inexact 6:2 compressor*

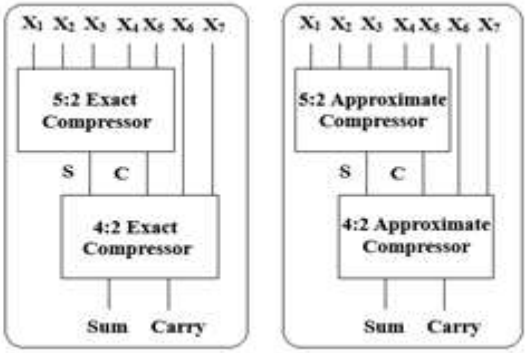

Figure 6. *Exact and Inexact 7:2 compressor*

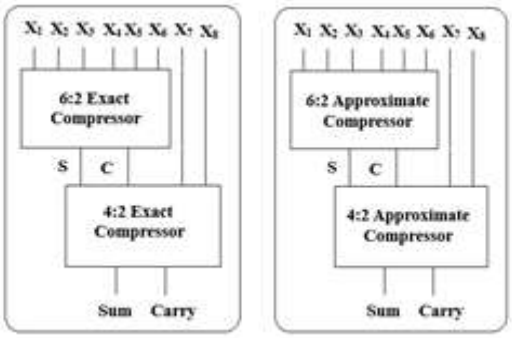

Figure 7. *Exact and Inexact 8:2 compressor*

3.2 Design of proposed 8x8 multiplier

The proposed 8x8 Dadda multiplier is structured into three main regions for partial product decomposition, as depicted in Figure 8. Here is a breakdown of each region:

First Region: Truncated Computation:

In the first region, columns 0 to 5 (the least significant 6 bits) are truncated. Instead of generating partial products for these columns, a constant "000110" is assigned to the product bits Y[5:0]. This approach eliminates the need for generating partial products for these columns and saves 15 AND gates and the hardware needed to combine these partial products. To handle the error from truncation, at column 5, the first three rows' bits are ORed and propagated to column 6, while bits 4, 5, and 6 are ORed and propagated to column 8.

Second Region: Approximate Computation:

In the second region, columns 6 to 8 use approximate computing. Each column utilizes an approximate 8:2 approximate compressor. In Column 6, the 8:2 compressor processes 7 partial product bits and the ORed error compensation output from truncation to produce the sum bit Y[6]. At Column 7, another approximate 8:2 compressor handles 8 partial product bits to produce the sum bit Y[7]. At Column 8, another approximate 8:2 compressor processes 7 partial product bits along with the ORed error compensation output from truncation to produce the sum bit Y[8]. Additionally, errors introduced by approximation are compensated by ORing the carry outputs from all three 8:2 compressors and then propagating this combined result to column 9.

Third Region: Exact Computation:

The third region covers columns 9 to 15 and uses exact computation. In Column 9, a 7:2 compressor combines 6 partial product bits with the carry from the previous error compensation for approximation to produce the sum bit Y[9] and the carry is propagated to column 10. At Column 10, a 6:2 compressor processes 5 partial product bits and the carry from column 9 to produce the sum bit Y[10], with the carry propagated to column 11. In Column

11, a 5:2 compressor adds 4 partial product bits and the carry from column 10 to produce the sum bit Y[11], with the carry propagated to column 12. In Column 12, a 4:2 compressor processes 3 partial product bits and the carry from column 11 to produce the sum bit Y[12], with the carry propagated to column 13. At Column 13, a full adder adds 2 partial product bits and the carry from column 12 to produce the sum bit Y[13], with the carry propagated to column 14. In Column 14, an half adder processes 1 partial product bit and the carry from column 13 to produce the sum bit Y[14] and the final carry from this operation becomes the last product bit Y[15].

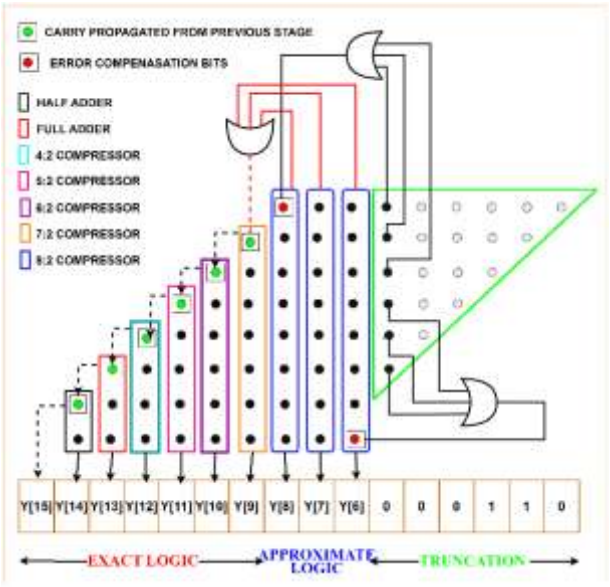

Figure 8. *Architecture of proposed 8X8 approximate multiplier*

3.3 Design of proposed 16x16 multiplier

The 16x16 multiplier can be constructed by cascading four 8x8 multipliers, as illustrated in Figure 9. The first 8x8 multiplier handles the multiplication of the lower 8 bits, A[7:0] and B[7:0], producing a 16-bit product that forms the first row. The second 8x8 multiplier multiplies the higher eight bits A[15:8] with the lower eight bits B[7:0], and the resulting product is placed in the second row, shifted 8 bit positions to the left relative to the first row. Similarly, the third 8x8 multiplier performs multiplication on the lower 8 bits A[7:0] and the higher 8 bits B[15:8], placed in third and aligning its product with second row. Finally, the fourth 8x8 multiplier multiplies the higher 8 bits A[15:8] and B[15:8], placing this product in the last row, shifted 8 bits to the left relative to the third row. This structured approach ensures that all partial products are correctly aligned to generate the final 32-bit result of the 16x16 multiplication. The detailed architecture of the 16x16 multiplier, as shown in Figure 10, generates a 32-bit product Y[31:0], which corresponds to the 0^{th} to 31^{st} columns. In this design, the first eight columns directly produce Y[7:0], representing the lower eight bits of the first 8x8 multiplier. The partial products in the 8^{th} column are combined using an approximate full adder to generate Y[8], with the carry propagated to the next column. In the 9^{th} column, three partial product bits along with the carry from the previous column are processed using an approximate 4:2 compressor, resulting in Y[9] while carrying the output forward. This process continues through to the 15^{th} column, generating product bits from Y[10] to Y[15]. From the 16^{th} to the 23^{rd} columns, the same approach is followed, but exact versions of the 4:2 compressors are used to ensure accuracy, yielding product bits from Y[17] to Y[23]. Notably, at the 23^{rd} column, the carry-out from the approximate 4:2 compressor is not propagated to the 24^{th} column; instead, it is propagated to the 30^{th} column. The product bits from the 24^{th} to the 29^{th} columns are generated directly from the fourth 8x8 multiplier, producing Y[24] to Y[29]. Finally, the product bits from the 30^{th} and 31^{st} columns are combined with the carry from the 23^{rd} column using an exact full adder, resulting in Y[30] and Y[31]. This design ensures that higher-order partial products (from the 16th to the 31^{st} columns) are processed with exact circuits to maintain accuracy and minimize precision loss.

4. Results and Discussions

To assess the performance of the proposed approximate multipliers (8x8 and 16x16), the architectures were modeled using Verilog HDL and synthesized with a 45-nm CMOS Cell Library using Cadence Design Compiler. This allowed for the measurement of circuit delays, power consumption, and area. Next, post-synthesis simulations were conducted with 500,000 randomly generated test vectors, capturing signal transitions in VCD files. After synthesis, the collected data were integrated into the netlists. Furthermore, the performance of the proposed 8x8 and 16x16 multipliers was verified by implementing them in the image processing application for image edge detection, providing a practical evaluation of their effectiveness in real-world scenarios. For this, Monte Carlo simulations were run with 500,000 uniformly distributed input patterns to evaluate various performance metrics. The analysis emphasized both performance metrics, including area utilization, power consumption, and delay time as well as quality metrics such as NMED, MRED, PSNR, NoEB and MSSIM. A comprehensi ve comparative analysis was conducted against state-of-the-art techniques, incorporating all relevant static values and graphical representations.

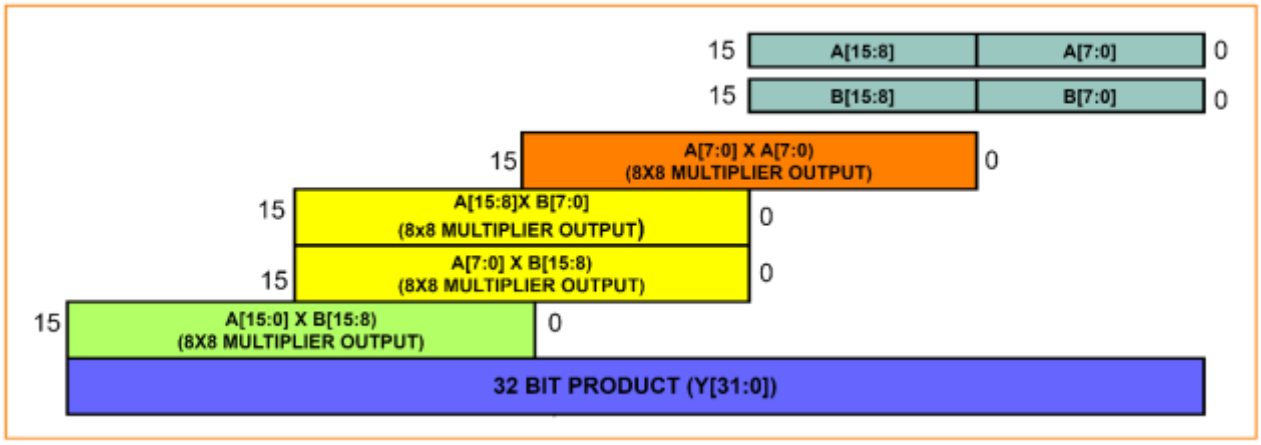

Figure 9. *Structure of proposed 16x16 approximate multiplier*

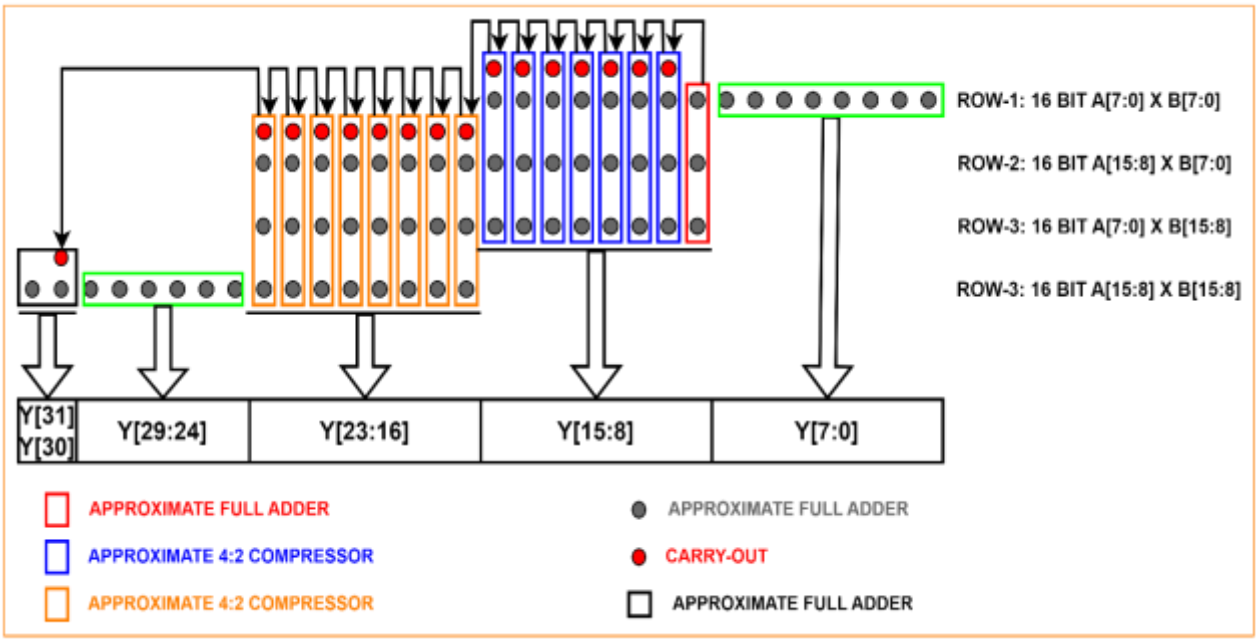

Figure 10. *Detailed architecture of proposed 16x16 approximate multiplier*

4.1 8x8 multiplier

The proposed 8x8 multiplier, modeled in Verilog and implemented in a 45-nm CMOS technology library using the Cadence compiler, demonstrates significant improvements in performance metrics, including power, area, and delay, as summarized in Table 1. The proposed design features an area of 320.6 µm², making it the most compact option compared to other leading designs, which range from 374.2 µm² to 699.5 µm². It also exhibits the fastest delay of 1.947 ns, outperforming competitors whose delays span from 2.374 ns to 3.248 ns. Power consumption is minimal at 38.80 µW, while other designs consume significantly more, with the highest at 95.20 µW. This optimal combination of low power and fast operation results in the lowest Power-Delay Product (PDP) of 75.54 fJ, demonstrating a substantial efficiency advantage over other designs, which range from 120.89 fJ to 309.21 fJ. The quality metrics of the proposed 8x8 multiplier are also summarized in Table 1, indicating that the design excels in quality as well. It achieves the lowest Normalized Mean Error Distortion (NMED) of 10.3 × 10^{-4} and a Mean Relative Error Distortion (MRED) of 2.3 × 10^{-4}, demonstrating superior error performance. Additionally, the design records a high Peak Signal-to-Noise Ratio (PSNR) of 26 dB and a Maximum Structural Similarity Index (MSSIM) of 0.9986, underscoring its exceptional image quality. Furthermore, it maintains a high number of edges (NoEB) at 9.2. Overall, the proposed 8x8 multiplier design stands out not only for its impressive performance in speed, power efficiency, and area but also for its superior quality metrics. The organized graphical representation of the statistical values of performance and quality metrics are illustrated in Figure 11.

4.2 16x16 multiplier

The proposed 16x16 multiplier design demonstrates substantial advancements in both performance and quality metrics, as summarized in Table 2. With a delay of 2.627 ns, it is the fastest design compared to others, which range from 3.374 ns to 4.583 ns. Additionally, it exhibits the lowest power consumption at 266.04 µW, while competing designs consume up to 652.8 µW. This combination results in a PDP of 697.02 fJ, significantly lower than other designs, which range from 1152.54 fJ to 2989.82 fJ. The proposed design also occupies only 1980 µm² of area, making it more compact than other designs that range from 2100 µm² to 5400 µm². In terms of quality metrics, the proposed design features a NMED of 19.47 × 10^{-4}, outperforming designs with values up to 35.8 × 10^{-4}. It has a MRED of 5.12 × 10^{-4}, which is competitive among the other designs, and achieves a PSNR of 25dB. The NoEB stands at 8.9, with an impressive MSSIM of 0.9915, indicating superior perceptual quality. Overall, the proposed design excels in performance, power efficiency, hardware simplicity, area efficiency, and image quality, positioning it as a highly optimized and competitive solution for high-performance multipliers. The organized graphical representations of the statistical values from Table 2 are depicted in Figure 12.

Table 1. *Comparison of performance and quality metrics of 8x8 multiplier design*

Design	Area (µm²)	Power (µW)	Delay (ns)	PDP (fJ)	NMED (X10^{-4})	MRED (X10^{-4})	PSNR (dB)	NoEB	MSSIM
Proposed	320.6	38.80	1.947	75.54	10.3	2.3	26	9.2	0.9986
[19]	374.2	45.60	2.651	120.89	12.4	3.13	25	9	0.9894
[18]	389.5	49.88	2.374	118.42	16	3.68	23	8.3	0.9493
[17]	537.8	64.18	2.855	183.23	21.8	5.49	16	6.1	0.8914
[14]	699.5	95.20	3.248	309.21	17.2	2.83	23	8.2	0.9193
[7]	665.7	83.80	3.169	265.56	9.8	2.1	27	9.7	0.9997

Table 2. *Comparison of performance and quality metrics of 16x16 multiplier design*

Design	Area (µm²)	Power (µW)	Delay (ns)	PDF (fJ)	NMED (X10^{-4})	MRED (X10^{-4})	PSNR (dB)	NoEB	MSSIM
Proposed	2244.2	266.04	2.627	697.02	19.47	5.12	25	8.9	0.9915
[19]	2619.4	312.66	3.745	1169.35	21.4	6.98	24	8.6	0.9892
[18]	2726.5	342.00	3.374	1152.54	24.7	8.20	22	7.8	0.9290
[17]	3764.6	440.04	3.926	1724.96	30.1	12.24	16	5.4	0.8920
[14]	4896.5	652.8	4.583	2989.82	35.8	6.31	22	7.7	0.9191
[7]	4659.9	574.62	4.369	2505.34	19.05	4.68	26	9.2	0.9996

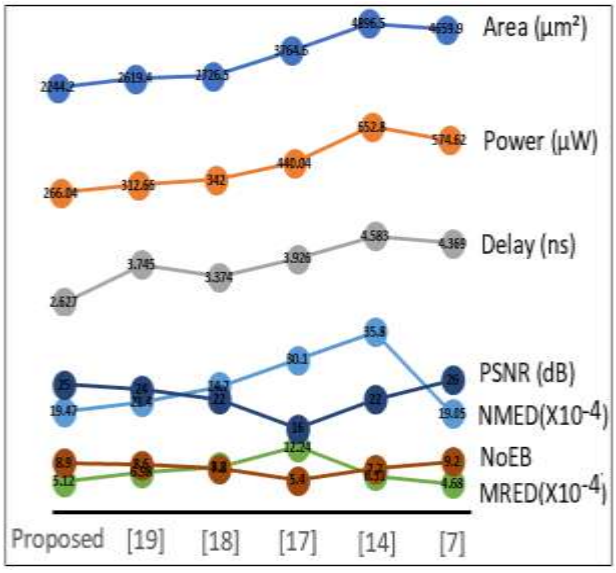

Figure 11. *Graphical representation of Comparison of 8x8 multipliers*

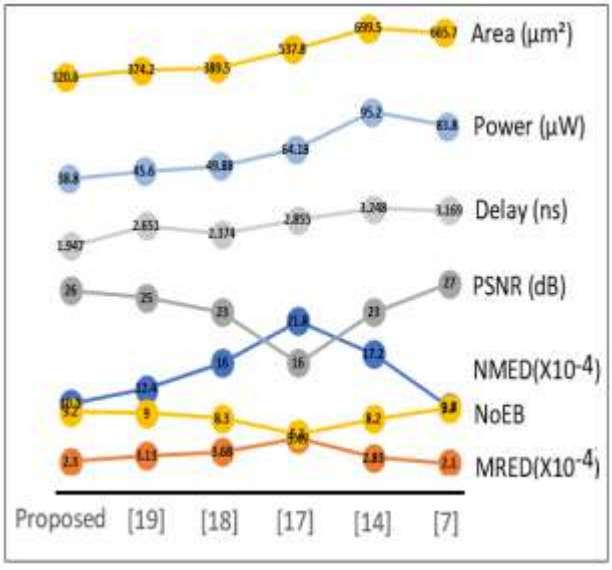

Figure 12. *Graphical representation of Comparison of 16x16 multipliers*

5. Conclusions

This work addresses the challenges of resource utilization and error tolerance in multiplier design, particularly in applications like multimedia processing and signal handling. To tackle these issues, we propose efficient 8x8 and 16x16 approximate multipliers that integrate advanced techniques such as truncation, approximation, and exact computation, optimizing both accuracy and efficiency. The primary objectives of this study include designing a high-performance 8x8 approximate multiplier using single-stage partial product decomposition with higher-order compressors, developing a scalable 16x16 multiplier by cascading four optimized 8x8 designs, and verifying their performance in image processing applications. The performance metrics for the proposed multipliers are impressive. The 8x8 multiplier achieves a delay of 1.947 ns, power consumption of 38.8 μW, and a strong balance between performance and efficiency. The 16x16 multiplier further excels with a delay of 2.627 ns and the lowest power consumption at 266.04 μW, combined with minimal error rates. Together, these designs offer significant improvements in speed, power efficiency, and error minimization, making them highly effective solutions for resource-constrained, error-tolerant applications.

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

References

[1]Ahmadinejad, M., & Moaiyeri, M. H. (2022). Energy- and Quality-Efficient Approximate Multipliers for Neural Network and Image Processing Applications. *IEEE Transactions on Emerging Topics in Computing*, *10*(2), 1105–1116. https://doi.org/10.1109/TETC.2021.3072666

[2]Akbari, O., Kamal, M., Afzali-Kusha, A., & Pedram, M. (2017). Dual-Quality 4:2 Compressors for Utilizing in Dynamic Accuracy Configurable Multipliers. *IEEE Transactions on Very Large-Scale Integration (VLSI) Systems*, *25*(4), 1352–1361. https://doi.org/10.1109/TVLSI.2016.2643003

[3]Anil Kumar, U., Chatterjee, S. K., & Ahmed, S. E. (2022). Low-Power Compressor-Based Approximate Multipliers with Error Correcting Module. *IEEE Embedded Systems Letters*, *14*(2), 59–62. https://doi.org/10.1109/LES.2021.3113005

[4]Antonio G.M. Strollo, Davide De Caro, Ettore Napoli, Nicola Petra, & Gennaro Di Meo. (2020). Low-Power Approximate Multiplier With Error Recovery Using a New Approximate 4-2 Compressor. IEEE.

[5]Edavoor, P. J., Raveendran, S., & Rahulkar, A. D. (2020). Approximate multiplier design using novel dual-stage 4:2 compressors. *IEEE Access*, *8*, 48337–48351. https://doi.org/10.1109/ACCESS.2020.2978773

[6]Esposito, D., Strollo, A. G. M., Napoli, E., De Caro, D., & Petra, N. (2018). Approximate multipliers based on new approximate compressors. *IEEE Transactions on Circuits and Systems I: Regular Papers*, *65*(12), 4169–4182. https://doi.org/10.1109/TCSI.2018.2839266

[7]Frustaci, F., Perri, S., Corsonello, P., & Alioto, M. (2020). Approximate Multipliers with Dynamic Truncation for Energy Reduction via Graceful Quality Degradation. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *67*(12), 3427–3431. https://doi.org/10.1109/TCSII.2020.2999131

[8]Ha, M., & Lee, S. (2018). Multipliers with Approximate 4-2 Compressors and Error Recovery Modules. *IEEE Embedded Systems Letters*, *10*(1), 6–9. https://doi.org/10.1109/LES.2017.2746084

[9]Jiang, H., Javier, F., Santiago, H., Mo, H., Liu, L., & Han, J. (2015). Approximate Arithmetic Circuits: A Survey, Characterization and Recent Applications.

[10]Khosravi, S., & Kamran, A. (2024). Iterative construction of energy and quality-efficient approximate multipliers utilizing lower bit-length counterparts. *Journal of Supercomputing*, *80*(13), 19210–19247. https://doi.org/10.1007/s11227-024-06212-8

[11]Liangyu Qian, Weiqiang Liu, Fabrizio Lombardi, & Jie Han. (2016). Design and Evaluation of An Approximate Wallace-Booth Multiplier. IEEE.

[12]Liu, W., Zhang, T., McLarnon, E., Orneill, M., Montuschi, P., & Lombardi, F. (2021). Design and Analysis of Majority Logic-Based Approximate Adders and Multipliers. *IEEE Transactions on Emerging Topics in Computing*, *9*(3), 1609–1624. https://doi.org/10.1109/TETC.2019.2929100

[13]Manikantta Reddy, K., Vasantha, M. H., Nithin Kumar, Y. B., & Dwivedi, D. (2019). Design and analysis of multiplier using approximate 4-2 compressor. *AEU - International Journal of*

Electronics and Communications, *107*, 89–97. https://doi.org/10.1016/j.aeue.2019.05.021

[14]Minaeifar, A., Abiri, E., Hassanli, K., & Darabi, A. (2024). A High-Accuracy Low-Power Approximate Multipliers with New Error Compensation Technique for DSP Applications. *Circuits, Systems, and Signal Processing*, *43*(1), 526–544. https://doi.org/10.1007/s00034-023-02487-z

[15] Momeni, A., Han, J., Montuschi, P., & Lombardi, F. (2015). Design and analysis of approximate compressors for multiplication. *IEEE Transactions on Computers*, *64*(4), 984–994. https://doi.org/10.1109/TC.2014.2308214

[16]Pei, H., Yi, X., Zhou, H., & He, Y. (2021). Design of Ultra-Low Power Consumption Approximate 4-2 Compressors Based on the Compensation Characteristic. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *68*(1), 461–465. https://doi.org/10.1109/TCSII.2020.3004929

[17]Sabetzadeh, F., Moaiyeri, M. H., & Ahmadinejad, M. (2023). An Ultra-Efficient Approximate Multiplier With Error Compensation for Error-Resilient Applications. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *70*(2), 776–780. https://doi.org/10.1109/TCSII.2022.3215065

[18] Strollo, A. G. M., Napoli, E., De Caro, D., Petra, N., Saggese, G., & Di Meo, G. (2022). Approximate Multipliers Using Static Segmentation: Error Analysis and Improvements. *IEEE Transactions on Circuits and Systems I: Regular Papers*, *69*(6), 2449–2462. https://doi.org/10.1109/TCSI.2022.3152921

[19]Xiaoting Sun, Yi Guo, Z. L., & Shinji Kimura. (2018). A Radix-4 Partial Product Generation-Based Approximate Multiplier for High-speed and Low-speed power Digital Signal Processing. Conference: *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)* DOI:10.1109/ICECS.2018.8617854