Performance-Driven Hybrid Approximate Multiplier Design with Selective Exact Computation for Embedded Systems

Mr. K. NAGA SAI, P. MOUNIKA[2]
[1]Assistant Professor, Dept. of E.C.E, RV Institute of Technology, Guntur- 522212
[2]PG Scholar, Dept. of E.C.E, RV Institute of Technology, Guntur- 522212

Abstract: Approximate computing has emerged as an efficient design methodology for achieving significant improvements in power consumption, processing speed, and hardware utilization in error-tolerant applications such as convolutional neural networks, multimedia processing, biomedical signal analysis, and image processing systems. This paper presents an ultra-efficient N-bit approximate multiplier architecture integrated with an adaptive Error Compensation Module (ECM) and a high-speed Parallel Prefix Adder (PPA) for enhanced computational performance. In the proposed design, the least significant partial product region is approximated using a constant-truncated compensation technique, while the most significant partial products are generated accurately using reduction structures composed of half adders, full adders, and 4:2 compressors to preserve computational precision. The ECM effectively minimizes approximation errors with minimal area and power overhead. Furthermore, the final accumulation stage employs a Parallel Prefix Adder to reduce carry propagation delay and improve overall operating speed compared with conventional ripple-carry and carry-select adders. The proposed architecture achieves an optimized trade-off between accuracy, power consumption, delay, and hardware complexity. Simulation and synthesis results demonstrate improved energy efficiency, reduced latency, enhanced scalability, and higher operating frequency, making the proposed multiplier highly suitable for low-power, high-speed, and next-generation approximate computing applications.

Key Words: Approximate computing, digital arithmetic, error compensation, low power design, multiplier architecture

1. Introduction

The rapid growth of modern digital applications such as digital signal processing, artificial intelligence, machine learning, multimedia systems, and communication technologies has created a strong demand for high-speed and energy-efficient arithmetic circuits. Among various arithmetic components, multipliers are considered one of the most important and computation-intensive units because multiplication operations dominate overall system delay, power consumption, and hardware complexity. Conventional exact multipliers provide highly accurate results; however, they often require large silicon area, high power consumption, and increased

Doi:10.48047/ijiee.2026.16.2.58

computational delay, making them less suitable for modern low-power VLSI systems.

Approximate computing has emerged as an effective design approach for improving hardware efficiency by allowing controlled computational inaccuracies in error-tolerant applications such as image processing, neural networks, video compression, and signal processing systems. Approximate multipliers reduce hardware complexity and energy consumption by simplifying partial product generation and accumulation, particularly in the least significant bits where small errors have minimal impact on output quality. To improve the accuracy of approximate multiplication, Error Compensation Modules (ECMs) are introduced to estimate and reduce approximation errors while maintaining low power and area overhead.

In this work, an ultra-efficient approximate multiplier architecture integrated with an Error Compensation Module and a Parallel Prefix Adder (PPA) is proposed to achieve high-speed and low-power operation. The most significant partial products are computed accurately, while the least significant region is approximated to reduce hardware complexity and delay. The incorporation of a high-speed PPA in the final accumulation stage minimizes carry propagation delay and enhances computational throughput. The proposed architecture achieves an effective balance between accuracy, speed, power consumption, and hardware utilization, making it highly suitable for next-generation FPGA, VLSI, embedded, and AI-based processing applications.

2.Literature Survey

[1]Vasileios Leon, Georgios Zervakis, and Dimitrios Soudris proposed an approximate hybrid high-radix encoding technique for energy-efficient inexact multipliers. Their approach utilizes the error resilience property of modern applications to reduce power consumption and hardware complexity with minimal accuracy degradation. In the proposed method, the most significant bits are encoded using exact radix-4 encoding, while the least significant bits are approximated using higher radix encoding. The architecture achieves significant reductions in energy consumption and silicon area while maintaining acceptable computational accuracy. Experimental results demonstrated considerable improvements in energy efficiency and scalability compared with conventional and existing approximate multiplier architectures.

[2]Zhang Chip-Hong Chang, Jiangmin Gu, and Mingyan presented ultra-low voltage and low-power 4:2 and 5:2 compressor architectures for fast arithmetic circuits. The proposed compressors were implemented using optimized CMOS static logic techniques to reduce power dissipation and improve speed performance at very low supply voltages. A novel XOR-XNOR module and efficient carry generation structure were incorporated to enhance compressor performance and eliminate weak logic issues in pass transistor circuits. The proposed compressor architectures demonstrated improved power-delay product, reduced energy consumption, and reliable operation at low voltages compared with conventional compressor designs. These compressors are highly suitable for high-speed parallel multipliers and low-power VLSI arithmetic applications.

3.Existing System

Approximate computing is widely used to improve the energy efficiency and performance of modern digital systems where small computational errors are acceptable.

Doi:10.48047/ijiee.2026.16.2.58

Approximate multipliers reduce power consumption, propagation delay, hardware complexity, and silicon area by simplifying arithmetic operations. In most existing systems, approximation is mainly applied to the least significant bits to minimize hardware usage while maintaining acceptable output quality. Techniques such as truncation, approximate compressors, and simplified logic gates are commonly used to achieve faster and low-power multiplication. Partial products are generated using AND operations between input bits and are reduced using full adders and reduction trees. However, these methods often introduce approximation errors and unpredictable error distributions. Conventional final adders such as Ripple Carry Adders and Carry Select Adders increase carry propagation delay and hardware overhead. Some systems use error compensation circuits to improve accuracy, but these circuits may increase area and design complexity. Existing architectures also face scalability issues for higher bit-width operations such as 16-bit and 32-bit multiplication. Due to inefficient carry propagation and critical path delay, many approximate multipliers fail to achieve an optimal balance between speed, power efficiency, area utilization, and computational accuracy.

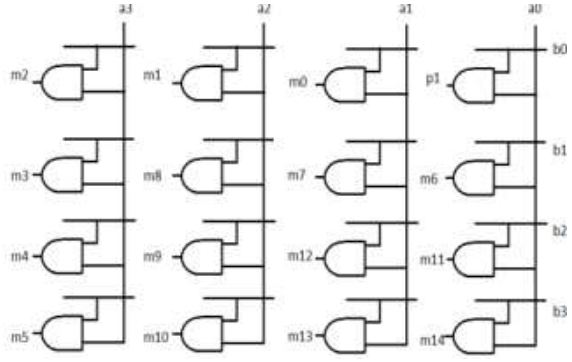

Fig 3.1: Product terms generated by a collection of AND gates.

The Ripple Carry Adder (RCA) is commonly used in multiplier architectures to perform multi-bit addition operations. It is constructed using multiple full adders connected in series, where the carry output of one adder is provided as the carry input to the next stage. Due to this sequential carry propagation, the carry signal "ripples" through each stage, resulting in increased propagation delay for higher bit-width operations. In the proposed RCA-based Wallace multiplier architecture, partial products generated during multiplication are reduced using layers of full adders. Each full adder combines three equally weighted input bits and produces sum and carry outputs. The carry outputs are forwarded to the next stage for further addition, while partial sums are accumulated to generate the final product bits. Although the RCA structure is simple and area-efficient, the sequential carry propagation limits overall computational speed in high-performance arithmetic systems.

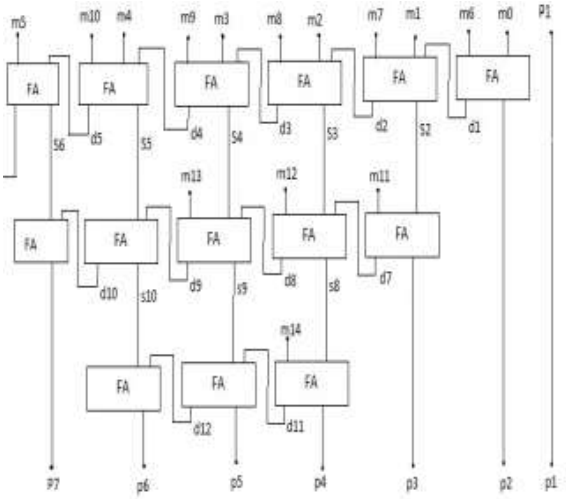

Fig 3.2:4x4 Wallace Multiplier Implementation.

- A Wallace tree multiplier is a high-speed parallel multiplication architecture that reduces partial products using a hierarchical tree of adders.
- It uses Carry Save Adders (CSAs) to compress multiple input bits in each column, thereby reducing the number of partial product rows efficiently.

Doi:10.48047/ijiee.2026.16.2.58

- In each reduction stage, bits are grouped in sets of three and converted into sum and carry outputs in parallel, improving computation speed.
- The reduction process is applied iteratively until only two final rows remain, significantly reducing propagation delay through logarithmic depth.
- The final two rows are combined using a fast adder to generate the final multiplication result with improved speed and efficiency.

To expedite the compacting of partial products, high-speed parallel multipliers are using the 4-2 Carry Save Adder. Traditional implementations of such a 4-2 Carry Save Adder use a cascade of two complete adders, as illustrated in Fig.4.6.

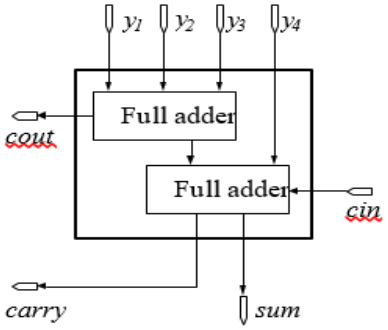

Fig.3.3: Implementation of a conventional 4-2 carry save adder.

It uses 5 inputs (y1, y2, y3, y4, and cin) and 3 outputs (y0, y1, and cin) with each 4-2 carry save adder (sum, cout, and carry).

4.PROPOSED SYSTEM

The proposed system presents an ultra-efficient approximate multiplier architecture designed for high-speed and low-power digital applications. The architecture combines approximate computing techniques, an Error Compensation Module (ECM), and a high-speed Parallel Prefix Adder (PPA) to achieve reduced delay, lower power consumption, and optimized hardware utilization. Approximation is mainly applied to the least significant partial products to minimize computational complexity, while the most significant partial products are generated accurately to maintain acceptable output precision. The ECM compensates for approximation errors with minimal hardware overhead, thereby improving computational accuracy. In addition, the Parallel Prefix Adder accelerates the final accumulation stage by reducing carry propagation delay and improving operating frequency. The proposed architecture is implemented using Verilog HDL and is highly suitable for FPGA, VLSI, image processing, machine learning, and embedded applications. If we compel the erroneous output to occur at specific input characteristics, we can remedy compressor errors with a simple logic gate. The method can be extended to scenarios with more than four faults of the total sixteen input patterns. A new area efficient 4-2 compressor is designed in an effort to make the sum of *S* (sum) and *C* (carry) as close to the exact result as possible and to improve the regularity of the Karnaugh map. Table I shows the proposed compressor's outputs *S* and *C*.

The *C* derived from OR as shown in equation (1). According to De Morgan's laws, the proposed expression of *S* is shown in equation (2) under the assumption of using the fewest number of logic gates possible. It is worth noting that NAND and NOR gates have better performance than AND and OR gates [3].

$$C = P1 + P2$$

$$S = (\overline{P1} \cdot \overline{P2} + P1 \cdot P2)(P3 + P4)$$
$$= \overline{(P1 \oplus P2) + \overline{(P3 + P4)}}$$

The architecture of the proposed 4-2 compressor is presented in Fig. 1. This compressor only has two NOR gates and one XOR gate and one OR gate, which makes the

Doi:10.48047/ijiee.2026.16.2.58

compressor consume very few transistors. So even though it has six errors out of sixteen input patterns, the design is still attractive and competitive.

Out of the three, the subsequent stage is the basic one in the plan of a multiplier since it dials back the activity and devours more force. In this manner, blowers are being utilized in the second stage which decreases the force utilization and velocities up the activity of the whole multiplier circuit. The multiplier structure with careful blowers is displayed in Fig.5.2

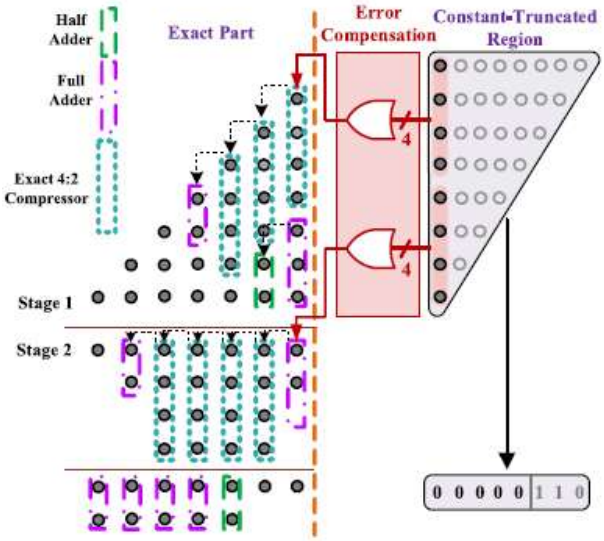

Fig 4.1: Proposed hybrid approximate multiplier structure

Step1: The module first analyses the impact of truncated bits on intermediate sum and carry behaviour to identify potential error sources in approximation. It focuses on statistically significant regions that contribute most to output error.

Step 2: An error signature is generated based on the observed truncation behaviour to estimate the magnitude and direction of deviation. This allows prediction of carry loss and bias without performing full recomputation.

Step 3: The system determines whether compensation is required and selects the most critical bits for correction. This ensures efficient operation by avoiding unnecessary switching activity and power consumption.

Step 4: Lightweight correction signals such as carry adjustments are generated using simple logic operations. These signals are designed for fast response and minimal propagation delay.

Step 5: The correction signals are injected into selected nodes of the reduction tree to minimize truncation error. This localized feedback improves accuracy while maintaining hardware efficiency.

Step 6: The corrected values propagate through the remaining computation stages to produce a stabilized approximate output. This balances reduced error with low power and compact design requirements.

The proposed system incorporates a lightweight and efficient Error Compensation Module (ECM) to reduce approximation errors generated in the lower-order partial products. Unlike conventional correction methods that increase hardware complexity and power consumption, the proposed ECM provides effective error correction with minimal area and delay overhead.

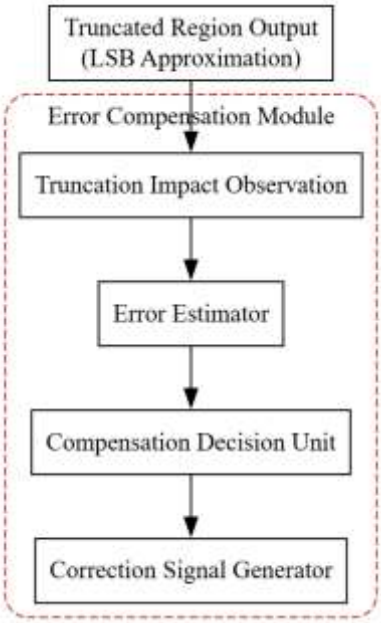

Fig 4.2: Proposed ECM.

Doi:10.48047/ijiee.2026.16.2.58

The compensation value generated by the ECM improves computational accuracy and enhances parameters such as Mean Error Distance (MED), Error Rate (ER), and Normalized Mean Error Distance (NMED). Partial products from both approximate and accurate regions are processed using an optimized reduction network that minimizes logic depth, carry propagation delay, and switching activity, thereby improving speed and reducing power consumption.

To further enhance performance, the proposed multiplier integrates a high-speed Parallel Prefix Adder (PPA) in the final accumulation stage. Unlike Ripple Carry Adders and Carry Select Adders, the PPA computes carry signals in parallel using prefix computation techniques, significantly reducing critical path delay and improving operating frequency. The combined use of selective approximation, optimized reduction logic, efficient error compensation, and parallel prefix addition enables the architecture to achieve lower power consumption, reduced hardware complexity, improved computational speed, and enhanced accuracy. Due to its scalability and high efficiency, the proposed multiplier is highly suitable for FPGA, VLSI, AI accelerators, IoT devices, image processing, and real-time embedded applications.

5.Results and Discussion

The proposed approximate multiplier improves performance by reducing area, power consumption, and delay through partial product approximation with error compensation. It achieves better accuracy and efficiency trade-offs, making it suitable for image processing, AI, and IoT applications.

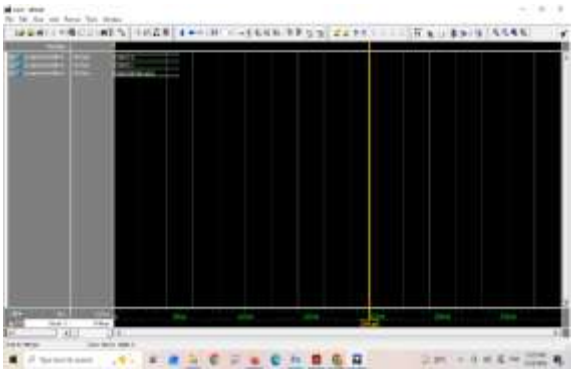

Fig 5.1: Simulation Result

The waveform verifies correct operation of the proposed multiplier, showing accurate output generation with effective approximation and proper error compensation during final accumulation.

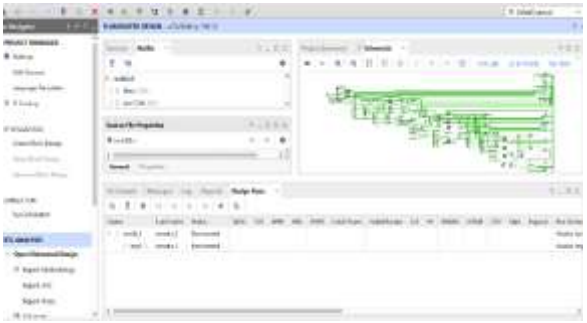

Fig 5.2: RTL schematic of ALU

The RTL schematic confirms the synthesized hardware structure of the multiplier, showing optimized integration of partial product generation, approximation, and error compensation stages.

Fig 5.3: Power Report

Fig 5.3 shows the power report of the proposed multiplier architecture obtained after synthesis. The report confirms that the design achieves low power consumption, making it suitable for energy-efficient and high-performance error-resilient applications.

Doi:10.48047/ijiee.2026.16.2.58

Fig 5.4: Delay Report

Fig 5.4 shows the delay report of the proposed multiplier architecture obtained after synthesis. The report confirms that the design achieves reduced propagation delay, enabling faster computation and improved performance for high-speed digital applications.

	Existing System	**Proposed System**
DELAY	13.514NS	13.096NS
POWER	8.850MW	9.922MW
AREA	68	58
SPEED	73.99MZ	76.35MZ

TABLE: Compression Table

The comparison results show that the proposed multiplier architecture achieves improved hardware efficiency and better overall performance compared with the existing design. The optimized structure reduces area utilization and computational delay while increasing processing speed, making the design more suitable for high-speed and error-resilient applications.

6.Conclusion and Future Scope

The proposed ultra-efficient approximate multiplier demonstrates a balanced trade-off between accuracy, speed, area, and power consumption. By integrating selective approximation with an effective error compensation module and a parallel prefix adder, the design significantly reduces delay and hardware complexity. Simulation and synthesis results confirm improved performance in terms of latency, energy efficiency, and resource utilization compared to conventional multipliers. The architecture maintains acceptable computational accuracy while achieving low-power operation. Its scalable and optimized structure makes it suitable for modern computing applications. Overall, the proposed design provides an efficient solution for high-performance approximate computing systems.

Future Scope: Future enhancements may include adaptive approximation techniques to further improve accuracy and energy efficiency. The design can also be extended to higher bit-width implementations and integrated into AI and DSP accelerators for real-time applications.

REFERENCES

[1] P. Harpe, E. Cantatore, and A. van Roermund, “A 2.2/2.7fJ/conversion-step 10/12b 40kS/s SAR ADC with data-driven noise reduction,” *IEEE ISSCC*, pp. 270–271, 2014.

[2] S. Babayan-Mashhadi and R. Lotfi, “Low voltage low-power double-tail comparator,” *IEEE TVLSI*, vol. 22, no. 2, pp. 343–352, 2014.

[3] K. Yoshioka and H. Ishikuro, “A 13b SAR ADC with eye-opening VCO-based comparator,” *IEEE ESSCIRC*, pp. 411–414, 2014.

[4] M. Brandolini et al., “5 GS/s 10b pipeline/SAR hybrid ADC,” *IEEE JSSC*, vol. 50, no. 12, pp. 2922–2934, 2015.

[5] B. Koziel, R. Azarderakhsh, M. Mozaffari Kermani, and D. Jao, “Post-quantum cryptography on FPGA based on isogenies on elliptic curve,” *IEEE TCAS I*, vol. 64, no. 1, pp. 86–99, 2017.

[6] B. Koziel, R. Azarderakhsh, and M. Mozaffari Kermani, “A high performance and scalable hardware architecture for isogeny-based cryptography,” *IEEE Trans. Computers*, vol. 67, no. 11, pp. 1594–1609, 2018.

Doi:10.48047/ijiee.2026.16.2.58

[7] A. Khorami and M. Sharifkhani, "Low-power high-speed comparator for precise applications," *IEEE TVLSI*, vol. 26, no. 10, pp. 2038–2049, 2018.
[8] S. Karunamurthi and V. K. Natarajan, "VLSI implementation of reversible logic gates cryptography with LFSR key," *Microprocessors and Microsystems*, vol. 69, pp. 68–78, 2019.
[9] H. Zhuang, H. Tang, and X. Liu, "Fully-dynamic time-interleaved noise shaping SAR ADC," *IEEE CICC*, 2020.
[10] G. Chandran, M. C. H. Mary, and G. Anjana, "VLSI implementation of image encryption and decryption using reversible logic gates," *IEEE PEREA*, 2020.
[11] N. Shahpari et al., "Early shutdown technique for dynamic comparators," *IEEE TCAS*, 2021.
[12] A. Lombardi Campos et al., "Low-power asynchronous SAR ADC in 65nm CMOS," *IEEE TCAS II*, 2021.
[13] A. Ihsan and N. Doğan, "Improved affine encryption algorithm for color images using LFSR and XOR encryption," *Multimedia Tools and Applications*, 2023.
[14] B. Murmann, "ADC Performance Survey 1997–2024," Stanford University, 2024.

Doi:10.48047/ijiee.2026.16.2.58