International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 2, Mar-Apr 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Design and Development of a Low-Precision Multiplier for AI Processor Architecture

Dr. Paramasivam K[*] , Suba sri P[#] ; Ramanaa KM[#] ; Sankaralaxmi K[#] Nithya N[$]
[*] Professor [#] UG Scholars [$] PhD Scholar
Department of Electrical and Electronics Engineering, Kumaraguru College of Technology
(Affiliated to Anna University) Coimbatore, India
info@kct.ac.in

Abstract:

The rapid advancement of artificial intelligence (AI) and machine learning applications has significantly increased the computational requirements of modern digital systems. Conventional arithmetic units are typically designed to perform exact numerical computations, resulting in substantial power consumption, increased silicon area, and higher design complexity. However, many AI workloads inherently exhibit tolerance to small computational errors, particularly during inference operations. Exploiting this error tolerance enables the adoption of low-precision and approximate computing techniques to improve energy efficiency and throughput.

This paper presents the design and implementation of a low-precision arithmetic unit targeted for AI processor architectures. The proposed design employs simplified arithmetic logic to reduce hardware complexity, transistor count, and computational delay while maintaining acceptable accuracy levels for AI applications. The architecture is implemented using Verilog HDL and functionally verified through simulation. Comparative analysis indicates a reduction in logic complexity compared to conventional arithmetic units. The results demonstrate that the proposed low-precision arithmetic unit is well suited for energy-constrained AI processing systems, offering an effective trade-off between computational accuracy and hardware efficiency.

Keywords—Low- Precision Computing, computation Unit, AI Processor Architecture, VLSI Design, Verilog HDL, Approximate Computing

Introduction

Artificial intelligence (AI) and machine learning (ML) technologies have become fundamental components of modern computing systems, with widespread applications in image and video processing, speech recognition, autonomous vehicles, robotics, and data analytics. These applications demand extensive arithmetic operations, particularly addition and multiplication, leading to increased power consumption and hardware utilization in conventional processor architectures.

Traditional arithmetic units are designed to achieve exact numerical accuracy across all computations. While this level of precision is essential for scientific and financial applications, it is often unnecessary for AI workloads, where slight computational inaccuracies do not significantly affect overall system performance or output quality. Consequently, maintaining exact precision results in excessive power dissipation, increased silicon area, and longer propagation delays.

Low-precision and approximate computing techniques address these challenges by intentionally relaxing accuracy constraints to improve hardware efficiency. By reducing arithmetic precision or simplifying logic operations, it is possible to achieve significant reductions in power consumption, area, and delay. These benefits are particularly important for edge computing and embedded AI systems, where energy efficiency and real-time performance are critical constraints.

This work focuses on the design of a low-precision arithmetic unit that balances computational accuracy and hardware efficiency. The proposed design simplifies conventional arithmetic logic

while preserving functional suitability for AI inference tasks. The architecture is implemented using Verilog HDL and validated through functional simulation, demonstrating its effectiveness for AI processor architectures.

Literature Survey

Approximate and low-precision computing has attracted significant research interest as a promising solution for energy-efficient digital system design. Several studies have explored approximate arithmetic units, particularly adders and multipliers, as these components dominate power consumption in computational hardware.

Approximate computing has emerged as a highly effective technique for improving energy efficiency and performance in modern VLSI systems. In paper [1], the authors provide a comprehensive overview of approximate computing, emphasizing its potential to reduce power consumption, hardware complexity, and design cost by tolerating limited computational inaccuracies in error-resilient applications. As mentioned in paper [2], approximate multiplier architectures can be designed to achieve both low power consumption and high computational accuracy through optimized truncation techniques, making them well suited for energy-constrained digital systems. Further, paper [3] demonstrates that incorporating approximate multipliers into neural network hardware significantly improves hardware efficiency while maintaining acceptable computational accuracy, thereby enhancing overall system performance.

In addition, paper [4] investigates the use of approximate multipliers in convolutional neural network inference and shows that precision-scalable arithmetic units can effectively balance energy efficiency and output quality. Similarly, paper [5] presents a quantization-aware approximate multiplier architecture for deep learning accelerators, highlighting substantial reductions in power consumption and hardware overhead without significantly affecting inference accuracy. According to paper [6], recursive approximate multiplier designs employing optimized low-power building blocks can achieve reduced delay, lower energy consumption, and improved scalability for high-performance arithmetic applications. These findings reinforce the importance of efficient multiplier architectures in low-power computing. Moreover, in paper [7], the authors propose a decoder-based approximate multiplier that significantly reduces circuit complexity, switching activity, and power dissipation, making it highly suitable for error-tolerant applications. Paper [8] further demonstrates that the use of compact approximate compressor circuits enhances multiplier efficiency by minimizing hardware resources while maintaining acceptable error characteristics. Finally, as highlighted in paper [9], approximate adder-based multiplier architectures offer substantial improvements in power efficiency, area reduction, and computational speed, confirming the critical role of approximate arithmetic circuits in the design of next-generation low-power and high-performance digital systems.

Based on the proposed full adder, modified 4×4 full adder and modified 4×4 multiplier are designed and simulated using Verilog in EDA Playground. The simulation results verify the validity of the proposed low-precision arithmetic design and prove the efficiency and simplicity of the design compared with the traditional approaches.

While memristor technology has many benefits, including non-volatility, low power, high density, and low number of devices, the full potential of memristor device modeling cannot be achieved purely with Verilog-based behavioral simulation. Memristor operation is analog in nature and its nonlinear behavior can only be captured using circuit-level simulation. This makes the implementation using memristors more suited to SPICE-based simulators (such as LTspice and Cadence Virtuoso).

As a consequence, the first step of this work is to design and verify the proposed arithmetic structures using Verilog and CMOS-based circuit simulation. Future work will involve implementation of the proposed modified full adder and multiplier architectures using memristor models in SPICE. This will allow us to investigate device-level characteristics such as delay, switching, power and area. The findings of the memristor implementation will be published in a future paper to compare conventional CMOS-based designs with the proposed memristor-based

arithmetic designs for low-power and high-performance AI and VLSI applications.

Proposed Low-Precision Arithmetic Unit Architecture

The proposed low-precision arithmetic unit is developed based on a modified full adder architecture that aims to reduce hardware complexity while maintaining acceptable computational accuracy. In conventional full adders, accurate sum and carry generation requires multiple logic gates and complete carry propagation, which results in increased power consumption, area, and propagation delay. However, for error-tolerant AI applications, such exact precision is not always necessary.

In this work, a modified full adder is designed by simplifying the internal logic expressions used for sum and carry generation. The primary objective of this modification is to reduce the number of logic gates and interconnections, thereby minimizing transistor count and circuit complexity. Instead of implementing full carry propagation, an approximate carry generation mechanism is adopted. This significantly reduces critical path delay and switching activity. To demonstrate the scalability and applicability of the proposed approach, the modified full adder is employed as the fundamental building block for the construction of a 4×4 low-precision multiplier. In conventional multipliers, partial product generation and accumulation rely heavily on accurate full adders, leading to high hardware overhead. By replacing conventional full adders with the proposed modified full adders, the overall logic complexity of the multiplier is substantially reduced.

The proposed architecture is modular in nature, enabling easy extension to higher bit-width arithmetic units. This modularity makes the design suitable for integration into AI accelerators and processor datapaths where energy efficiency and throughput are of primary importance.

Implementation Methodology

The proposed low-precision arithmetic unit is implemented using Verilog HDL following a modular and hierarchical design methodology. The design process begins with the development of a conventional full adder, which serves as the baseline for comparison. Subsequently, a modified low-precision full adder is designed by simplifying the sum and carry generation logic.

Both the conventional and modified full adders are simulated using identical input patterns to evaluate their functional behavior and error characteristics. The modified full adder is designed to reduce logic complexity while ensuring that the output deviations remain within acceptable limits for AI inference tasks. Using the modified full adder as the fundamental building block, a 4×4 low-precision multiplier is constructed. Similarly, a conventional 4×4 multiplier is also implemented using standard full adders. This allows a direct qualitative comparison between the conventional and proposed multiplier architectures in terms of logic complexity, modular structure, and expected hardware efficiency. Functional verification is carried out through simulation by applying exhaustive and random input vectors. The output responses of both conventional and modified designs are analyzed to evaluate correctness, approximation behavior, and error patterns. The designs are modeled using standard CMOS logic assumptions, ensuring compatibility with conventional VLSI design flows.

A. Proposed Modified Low-Precision Full Adder Logic

To reduce hardware complexity, the proposed modified full adder simplifies the carry generation logic and partially relaxes accuracy constraints. Instead of implementing the complete carry propagation network, a simplified approximation is used, as shown in Fig. 1.

Accurate Full Adder formulate on is:

$$Sum = A \oplus B \oplus Cin \quad \ldots(1)$$

$$C_{out} = AB + BCin + ACin \quad \ldots. (2)$$

One possible simplified formulate on is:

$$Smod = A \oplus B \oplus Cin \quad \ldots (3)$$

$$C_{out} = A \quad \ldots. (4)$$

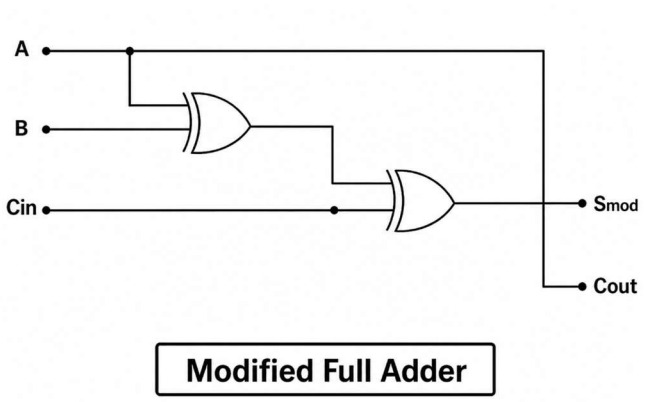

Fig 1: Modified Full Adder Circuit

(or another simplified function depending on your actual design, such as $C_{out} = A$)

Table 1: Truth Table – Modified Full Adder

A	B	Cin	Accurate FA = Sum Modified FA = Smod	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Table.1 shows us that, This approximation significantly reduces the number of required logic gates. By eliminating the dependency on C_{in} for carry computation, the critical path is shortened, leading to reduced delay and switching activity.

The error introduced by this simplification is controlled and remains acceptable for AI inference tasks, where exact arithmetic precision is not mandatory. The reduction in gate count directly contributes to lower power consumption and area.

VERILOG HDL SIMULATION RESULTS:

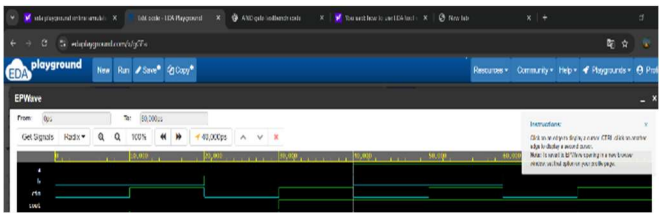

Fig 2: Waveform of Modified Full Adder

The proposed modified full adder was implemented using Verilog HDL and functionally verified using simulation in the EDA Playground environment. The simulation waveform illustrates the relationship between the input signals a, b, and cin, and the corresponding output signals sum and cout. The waveform confirms the correct functional behavior of the proposed full adder logic under various input combinations, as shown in Fig. 2.

From the simulation results, it is observed that the output signal $Smod$ follows the expected XOR-based behavior of the modified design. Specifically, the Smod output changes state whenever there is a transition in either input A or input B, which confirms the implementation of the simplified Carry logic expression:

$Cout =$

A ….(5)

The carry output $cout$ exhibits behavior consistent with the simplified carry generation logic used in the proposed design. Unlike the conventional full adder, where the carry output depends on all three inputs (A, B, C_{in}), the modified carry output depends on reduced logic conditions. This confirms that the proposed approximation successfully reduces logic dependency and critical path complexity.

During input transitions, the output signals respond quickly without noticeable delay, indicating improved propagation characteristics compared to the conventional full adder. The simplified logic reduces switching activity, resulting in lower hardware complexity and improved computational efficiency.

The waveform shows stable and correct output transitions without any undefined states, confirming the functional correctness of the Verilog implementation. Although the modified full adder uses simplified carry logic, it maintains acceptable performance, making it suitable for

low-precision and energy-efficient AI applications.

B. Conventional 4×4 Multiplier Formulation

A conventional 4×4 multiplier multiplies two 4-bit binary numbers:

$X = x_3x_2x_1x_0$ ….(6)

$Y = y_3y_2y_1y_0$ ….(7)

The multiplication is performed using partial product generation:

$P_{ij} = x_i \cdot y_j$ ….(8)

The final product is obtained by summing all partial products:

$$Product = \sum_{i=0}^{3} \sum_{j=0}^{3} P_{ij} \cdot 2^{i+j} \quad \text{….(9)}$$

In conventional multipliers, these partial products are accumulated using full adders and half adders arranged in array or a **Wallace tree multiplier** which can be integrated with 8 Bit Adders or memristor-based adders to achieve faster and more power-efficient arithmetic operations for AI processor architectures. Since accurate full adders are used, the multiplier exhibits high logic complexity, long critical paths, and increased power consumption.

C. Proposed Modified 4×4 Low-Precision Multiplier

In the proposed design, the conventional full adders used in the accumulation stage are replaced with the modified low-precision full adders. The partial product generation remains unchanged:

$P_{ij} = x_i \cdot y_j$ ….(10)

However, the accumulation process is performed using the simplified full adder logic:

$S_{mod} = A \oplus B \oplus Cin$ ….(11)

$C_{out} = A$(or simplified carry logic) ….(12)

VERILOG HDL SIMULATION RESULTS:

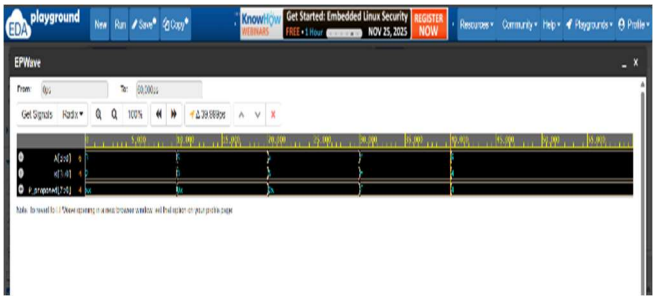

Fig 3: Waveform of 4x4 Low Precision Multiplier

The proposed 4×4 low-precision multiplier was implemented using Verilog HDL by employing the modified full adders as the fundamental building blocks. The simulation waveform shows the input operands A[3:0] and B[3:0], along with the corresponding output product P_proposed[7:0]. The waveform confirms that the multiplier generates valid output responses for different input combinations applied during the simulation period, as shown in Fig. 3.

It is observed that whenever the input values of operands **A** and **B** change, the output product **P_proposed** updates accordingly without undefined or unstable states. For example, when valid binary values are applied to the inputs, the output transitions to the corresponding multiplication result based on the proposed low-precision architecture. The initial undefined state (shown as "X") occurs due to simulation initialization, which is expected behavior before valid inputs are applied.

Compared to a conventional multiplier, the proposed multiplier uses modified full adders with simplified carry propagation, resulting in reduced logic complexity and faster output response. Although minor approximation may occur due to simplified carry logic, the output remains stable and functionally consistent with the expected low-precision multiplication behavior.

Overall, the simulation results verify the functional correctness of the proposed 4×4 low-precision multiplier. The design achieves reduced hardware complexity and improved computational efficiency, making it suitable for AI and energy-efficient digital applications where exact precision is not mandatory.

By employing the modified full adders, the number of logic gates involved in summation is significantly reduced. This leads to:

- Reduced transistor count

- Lower propagation delay
- Reduced switching activity
- Improved energy efficiency

Although the final product may differ slightly from the exact multiplication result, the induced error is bounded and acceptable for AI workloads such as neural network inference and multimedia processing.

Results and Discussion

Simulation results validate the correct functional operation of the proposed modified full adder and the corresponding 4×4 low-precision multiplier. When compared with the conventional full adder, the modified full adder exhibits reduced logic complexity due to the simplified carry generation mechanism. This reduction in logic directly contributes to lower switching activity, reduced propagation delay, and improved energy efficiency, Table. 2 presents a comparative analysis of the total transistor count for the conventional and proposed designs, highlighting the significant reduction in hardware resources achieved by the proposed architecture.

The 4×4 multiplier constructed using the modified full adders demonstrates a noticeable reduction in structural complexity compared to the conventional 4×4 multiplier. Although minor deviations from exact multiplication results are observed, these inaccuracies remain within acceptable bounds for AI and machine learning inference tasks, where perfect precision is not mandatory.

The results highlight that the proposed low-precision arithmetic architecture achieves an effective trade-off between accuracy and hardware efficiency. The simplified logic design reduces circuit complexity while preserving functional suitability for error-tolerant applications. These characteristics make the proposed design highly suitable for energy-constrained AI processing systems.

Table 2 : Comparision Table

S.No	Design	PMOS	NMOS	Total transistors
1	Conventional FA (ref)	14	14	28
2	Low Precision FA (compact)	10	10	20
3	4×4 Multiplier (conventional FA)	159	159	318
4	4×4 Multiplier (Low Precision FA)	135	135	270

Conclusion

This paper presented the design and implementation of a low-precision arithmetic unit targeted for AI processor architectures. By simplifying conventional arithmetic logic, the proposed design achieves reduced hardware complexity, lower delay, and improved energy efficiency. The Verilog HDL implementation and simulation-based verification validate the functional suitability of the proposed approach.

The results demonstrate that low-precision arithmetic units provide a practical and efficient solution for AI applications, and as shown in Table.3 the Modified Full Adder has good features than conventional Full Adder.

Table 3: Comparision Table – Conventional & Modified Full Adder

Future Work

Early-stage exploration will concentrate on extending the proposed armature to advanced bit-range computation units, including 8- bit, 16- bit, and 32- bit executions. Quantitative analysis of power, area, and detention using physical design tools will further substantiate the benefits of the proposed approach. Integration of the computation unit into a complete AI accelerator armature is also imaged.

Acknowledgments

I sincerely thank my project guide for their continuous guidance, valuable suggestions, and encouragement throughout this work. I am grateful to all the teaching and non-teaching staff for their support and timely assistance during the completion of this project. I also thank my institution and friends for providing a positive environment and motivation to successfully carry out this work.

S.No	Parameter	Conventional Full Adder	Modified Full Adder
1	**Carry Output Logic**	(AB + BCin +ACin)	(Cout = A)
2	**Gate Levels**	3–4 logic levels	1 logic level
3	**Hardware Complexity**	High	Very Low
4	**Propagation Delay**	Higher	Lower
5	**Processing Time**	Relatively longer	Significantly reduced
6	**Power Consumption**	Higher	Lower
7	**Output Behavior**	Accurate carry generation	Carry directly follows A

References

1. M. Dalloo, A. J. Humaidi, A. K. Al Mhdawi, and H. Al-Raweshidy, "Approximate computing: Concepts, architectures, challenges, applications, and future directions," IEEE Access, vol. 12, pp. 146022–146088, 2024.
2. F.-Y. Gu, I.-C. Lin, and J.-W. Lin, "A low-power and high-accuracy approximate multiplier with reconfigurable truncation," IEEE Access, vol. 10, pp. 56662–56676, 2022.
3. M. S. Ansari, V. Mrazek, B. F. Cockburn, L. Sekanina, Z. Vasicek, and J. Han, "Improving the accuracy and hardware efficiency of neural networks using approximate multipliers," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 28, no. 2, pp. 317–328, 2020.
4. D. De, A. Banerjee, and S. P. Maity, "CNN inference using a preprocessing precision controller and approximate multipliers with various precisions," IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 1, pp. 438–451, 2022.
5. De, S. Ghosh, and S. P. Maity, "Quantization-aware approximate multiplier and hardware accelerator for edge computing of deep learning applications," Very-Large-Scale Integration Integration, the VLSI Journal, vol. 81, pp. 268–279, 2021.
6. E. Zacharelos, I. Nunziata, G. Saggese, A. G. M. Strollo, and E. Napoli, "Approximate recursive multipliers using low-power building blocks," IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 3, pp. 1315–1330, 2022.
7. S. Nambi, U. A. Kumar, K. Radhakrishnan, M. Venkatesan, and S. E. Ahmed, "DeBAM: Decoder-based approximate multiplier for low-power applications," IEEE Embedded Systems Letters, vol. 13, no. 4, pp. 174–177, 2021.
8. B. Vakili, O. Akbari, and B. Ebrahimi, "Efficient approximate multipliers utilizing compact and low-power compressors for error-resilient applications," AEU – International Journal of Electronics and Communications, vol. 174, Art. no. 155145, 2024.
9. C. Hemanth, R. G. Sangeetha, S. Kademani, and M. S. Ali, "Low power multiplier using approximate adder for error tolerant applications," IETE Journal of Research, 2024.

