STU
SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA

Journal of Electrical Engineering, Vol. 77, No. 4, 2026, pp. 384-391
https://reference-global.com/journal/JEE

Radix-4 carry-save adder based accumulator for high-performance MAC units in factored systolic array accelerators

Komathy Vanitha Krishnan[1], Angeline Felicia Moses[1], Sowmya Pravin Sishu[1], Prediksha Dorai[1], Rajesh Kana Santharaj[2]

Machine learning workloads impose extreme computational demands on hardware, driving the adoption of domain-specific accelerators built around Systolic Arrays (SAs). Factored Systolic Arrays (FSAs) based on Radix-8 Booth multiplications reduce multiplier complexity by relocating the encoding logic to the array periphery, thereby simplifying each Processing Element (PE). Despite this multiplier-side enhancement, the Multiply-Accumulate (MAC) unit's accumulation stage continues to impose a critical bottleneck when implemented with a standard Ripple Carry Adder (RCA), whose carry-propagation delay scales linearly with the accumulator width. This paper presents a Radix-4 Carry-Save Adder (CSA) accumulator within the Radix-8 FSA MAC and evaluates it on the Xilinx Spartan-3E FPGA design suite 14.7. The proposed architecture significantly achieves 22.5%, 21.3%, 24.5% and 4.3% reductions in critical-path delay, LUT, Slice utilization and Power consumption relative to the RCA baseline. Further, as approximate computing has emerged as a promising solution for enhanced efficiency, a lower-part OR Adder variant is evaluated, achieving the best power-delay product (2204 pJ). Optimization within FSA MACs enables high-throughput Machine Learning acceleration.

Keywords: factored systolic arrays, machine learning acceleration, Radix-8 booth multiplication, processing element, Radix-4 carry-save adder, MAC, accumulator design, power

1 Introduction

The rapid proliferation of Machine Learning (ML) and Deep Neural Networks (DNNs) across scientific, industrial and commercial domains has fundamentally transformed the computing landscape [1]. These models achieve remarkable predictive accuracy, yet their practical deployment is constrained by immense computational demands where modern networks execute billions of Multiply-Accumulate (MAC) operations per inference pass, even for moderately scaled architectures [2, 3]. Conventional processors such as CPUs and GPUs struggle to sustain these workloads within the power, bandwidth and latency budgets imposed by edge and data-centre environments [4]. This has driven the development of domain-specific hardware accelerators designed to exploit the regular, data-parallel structure inherent to ML computations.

Systolic Arrays (SAs), first proposed by Kung [5], offer a particularly effective architectural template for this purpose. A SA consists of a regular grid of simple Processing Elements (PEs), each performing a MAC operation and passing data to its neighbours in a rhythmic, pipeline fashion. This structure enables high arithmetic throughput while achieving extensive data reuse for both weights and activations. SAs substantially reduce the pressure on external memory bandwidth, the dominant performance constraint in most ML workloads.

Consequently, the SA paradigm serves as the computational core of leading ML accelerators, most notably Google's Tensor Processing Unit [6]. An adaptive-precision SA framework for Transformer-based AI computation was introduced to accelerate matrix multiplication using reconfigurable PEs [7]. For Convolutional Neural Network-Recurrent Neural Network (CNN-RNN) based deep learning acceleration, an SA topology integrated with a modified Hybrid Kogge Stone Adder was developed to achieve enhanced computational throughput with optimized inter-processing-element data transfer [8]. A power-efficient SA system integrating a Dadda Tree Multiplier and Brent-Kung adder, for an efficient MAC unit, was designed using decomposition-based optimization techniques. Reduced Look-Up Table (LUT) usage and improved execution efficiency for machine learning applications were achieved [9].

Significant research effort has been directed at improving the arithmetic efficiency of the MAC units within each PE. Modified Booth encoding reduces the number of partial products (PPs) in binary multi-

[1] Department of Electronics and Communication Engineering, CSI College of Engineering, Ketti, The Nilgiris, Tamil Nadu, India
[2] Department of Mechanical Engineering, CSI College of Engineering, Ketti, The Nilgiris, Tamil Nadu, India
Corresponding author: vanitha.kkv@gmail.com

https://doi.org/10.2478/jee-2026-0037, Print (till 2015) ISSN 1335-3632, On-line ISSN 1339-309X

plication. Radix-8 Booth encoding reduces the PP count to approximately N/3 for an N-bit operand, yielding smaller and faster partial-product reduction trees [10]. The Factored Systolic Array (FSA) addresses the complexity of hard-multiple generation by relocating the shared Booth encoding logic to the array periphery, feeding pre-computed control signals to simplified PEs [11] as shown in Fig. 1.

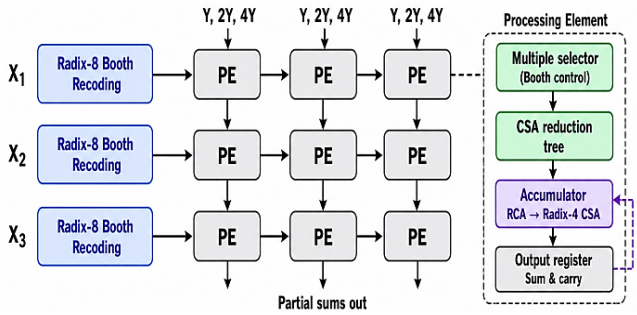

Fig. 1. Overview of a 3×3 factored systolic array

Despite the gains from FSA-based multiplier evolution, the accumulation stage of the MAC unit remains a critical constraint [12]. When implemented with a conventional RCA, the carry must propagate serially from the Least Significant Bit (LSB) to the Most Significant Bit (MSB), incurring a delay that grows linearly with the accumulator word width. The ripple delay dominates the MAC critical path and directly limits the maximum operating frequency (F_{max}) of the entire SA.

To overcome the above limitation, this paper proposes and evaluates a high-performance accumulator based on a Radix-4 Carry-Save Adder (CSA) configuration, integrated within the Radix-8 FSA MAC. Unlike an RCA, the CSA avoids carry propagation during intermediate accumulation steps by maintaining the running total in a redundant representation of a sum vector and a carry vector. It is updated in the delay of a single full adder, independent of word width. A lower-part OR Adder (LOA) is also evaluated as an approximate-computing alternative, providing a broader design-space comparison.

The remainder of the paper is organized as follows: Section 2 reviews prior work. Section 3 describes the baseline MAC with RCA accumulation. Section 4 presents the proposed Radix-4 CSA accumulator. Section 5 reports and analyses the FPGA synthesis results. Section 6 concludes the paper.

2 Literature review

2.1 Radix-8 multiplication in factored systolic arrays

The growing demand for real-time ML inference has accelerated the transition from general-purpose processors to custom silicon accelerators. SAs occupy a prominent position in this landscape because their regular, tiled structure maps directly onto matrix computations that dominate DNN workloads. While their nearest-neighbour data movement minimizes global interconnect cost [13]. Within each PE, the MAC unit is the fundamental computational kernel. Radix-4 Booth encoding examines overlapping 3-bit windows of the multiplier operand and generates at most N/2 partial products for an N-bit operand. Radix-8 Booth encoding decreases the number of PPs by recoding the multiplier into larger bit groups, resulting in fewer reduction stages [14]. This enables a smaller, faster PP reduction tree and translates into a lower area and shorter critical-path delay. The cost is the need to generate hard multiples such as ±3X and ±5X, which require dedicated logic not found in Radix-4 encoding [15].

The FSA eliminates the replication of this hard multiple-generation logic by hoisting the Booth encoding to the array periphery. FSA based on radix-8 multipliers, embedding Kogge-Stone parallel prefix adders and factoring techniques were utilized to achieve improved logic-level refinement in systolic arrays [16].

Though high-speed adder architectures, including Carry Look-Ahead Adders, Carry Select Adders and Parallel Prefix Adders (e.g., Kogge–Stone), provide fast addition for two operands, their performance deteriorates when multiple operands are accumulated due to increased carry-propagation overhead. While these adders reduce carry propagation delay compared to Ripple Carry Adders (RCAs), they often incur higher routing complexity, larger fan-out, and greater FPGA resource utilization. When addition is implemented using an RCA, the accumulation stage within each PE continues to constitute the critical timing path of the design. FPGA benchmarking confirms that RCAs exhibit significantly higher delays than alternative adder architectures at the bit-widths typical of ML MAC units [17].

In contrast, carry-save adders (CSAs) reduce multiple operands into sum and carry vectors without immediate carry propagation, making them well-suited for high-speed multi-operand arithmetic computations [18, 19]. The proposed Radix-4 CSA accumulator leverages this approach to minimize accumulation latency while maintaining lower hardware complexity within the FSA MAC implementation. The specific performance gains from integrating a Radix-4 CSA accumulator into a Radix-8 FSA MAC constitute the primary contribution of this paper.

2.2 Approximate computing and the lower-part OR adder

Approximate computing (AC) has emerged as an exciting field of research recently. AC offers active design methodologies that leverage the fact that digital processing systems and applications can tolerate a trivial loss of accuracy in computational output. AC exploits the inherent error tolerance of vast multimedia applications such as image processing, sensor fusion and neural network inference to enhance speed, area and power efficiency [20]. The AC paradigm has been successfully applied to adders, multipliers and MAC units across signal processing, big-data analytics and neuromorphic computing platforms [21]. An approximate integrated MAC (AIMAC) based systolic array was introduced for matrix multiplication and image processing applications. The model demonstrated reduced power consumption and latency while maintaining competitive image quality [22]. A SA structure featuring exact and approximate PEs, built with energy-aware partial-product cells was used for kernel-based and CNN-assisted edge detection tasks. Notable energy reduction was observed while preserving superior output quality in DCT and edge detection processes [23].

Among approximate adder designs, the Lower-part OR Adder (LOA) has emerged as an efficient and widely adopted design due to its trade-off between computational accuracy and hardware complexity. As demonstrated in Fig. 2, an *n*-bit LOA employs a partitioned structure of its operands into two segments. The upper k bits are processed by a conventional precise adder, producing an exact partial sum with carry-out. The lower (*n-k*) bits are handled by an array of OR gates, producing the approximate sum without any carry generation or propagation between bits [24]. The complete elimination of carry logic in the lower segment removes all inter-bit dependencies, reducing propagation delay and lowering switching activity at the cost of a bounded approximation error confined to the least significant bits.

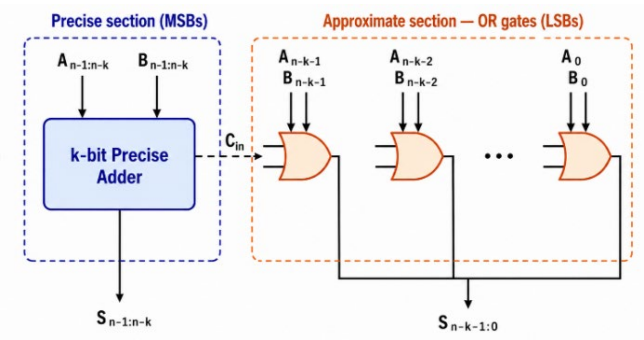

Fig. 2. Structure of the lower-part OR adder

In this work, the LOA is evaluated as a second accumulator variant within the FSA PE, referred to as the proposed PE (LOA) in the results tables alongside the primary Radix-4 CSA. This parallel evaluation allows direct comparison between an exact high-speed accumulator (Radix-4 CSA) and an approximate low-power accumulator (LOA) within the same FSA arrangement.

3 Factored systolic array MAC with RCA accumulation

The baseline architecture examined in this study is a MAC unit designed for integration into a Radix-8 Factored Systolic Array. It performs the Multiply-Accumulate operation using externally generated Radix-8 Booth control signals and employs a conventional RCA for the accumulation process. The complete datapath of this baseline PE is illustrated in Fig. 3.

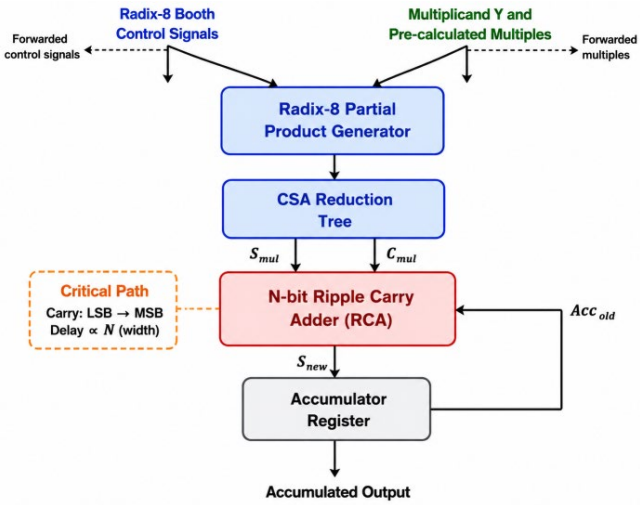

Fig. 3. Datapath of the baseline FSA PE with RCA accumulation

In accordance with the FSA principle, the PE does not house the Radix-8 Booth encoding logic internally. Instead, it receives five control signals s_i, d_i, t_i, q_i and n_i as represented in Eqns. (1) to (5) derived by external peripheral logic from overlapping 4-bit windows of the multiplier operand X, delivered via horizontal interconnects. These signals determine the multiplicand multiple to be selected for each of the N/3 partial-product rows according to the standard Radix-8 Booth encoding scheme summarized in Tab. 1. The Boolean expressions for the control signals are given as follows:

$$si = (X_3(i+1) - 1 \oplus X_3(i+1) \wedge (X_3 i \oplus X_3(i+1)) \quad (1)$$

$$di = \left(X_3 i \oplus X_3(i+1)\right) \wedge X_3 i \oplus X_3(i-1)) \quad (2)$$

$$ti = (X_3(i+1) - 1 \oplus X_3(i+1)) \wedge (X_3 i \oplus X_3(i-1)) \quad (3)$$

$$qi = ((X_3(i+1) - 1 \oplus X_3(i+1)) \wedge (X_3(i) \oplus X_3(i+1) \wedge (X_3 i \oplus X_3(i-1)) \quad (4)$$

$$ni = (X_3(i+1) - 1) \quad (5)$$

Table 1. Radix-8 Booth encoding: Multiplier bit patterns and selected multiples

X_{3i+2}	X_{3i+1}	X_{3i}	X_{3i-1}	Multiple selected	n_i
0	0	0	0	0	0
0	0	0	1	+Y	0
0	0	1	0	+Y	0
0	0	1	1	+2Y	0
0	1	0	0	+2Y	0
0	1	0	1	+3Y	0
0	1	1	0	+3Y	0
0	1	1	1	+4Y	0
1	0	0	0	−4Y	1
1	0	0	1	−3Y	1
1	0	1	0	−3Y	1
1	0	1	1	−2Y	1
1	1	0	0	−2Y	1
1	1	0	1	−Y	1
1	1	1	0	−Y	1
1	1	1	1	0	1

In addition to control signals, the PE receives the multiplicand Y and pre-calculated hard multiples (2Y, 3Y, 4Y) through vertical interconnect. These partial products are compressed by a CSA reduction tree into a two-row residue: a sum vector S_{mult} and a carry vector C_{mult}. The accumulation stage adds this result to the running total accumulated result stored in the feedback register. In the baseline design, a standard N-bit RCA whose critical-path delay scales linearly with N is explained. For the 16-bit accumulator evaluated here, this yields 8.791 ns, making the RCA the major delay contributor.

4 Proposed method: FSA MAC with Radix-4 CSA accumulator

The proposed MAC design targets the reduction of accumulation latency identified in the baseline architecture. Operating within the same Radix-8 FSA, the proposed MAC replaces the N-bit RCA with an accumulator built using a Radix-4 CSA configuration, as illustrated in Fig 4. The partial products generated by the Radix-8 Booth encoder result in sum and carry vectors. These vectors are accumulated by the proposed Radix-4 CSA accumulator, where separate sum and carry registers eliminate intermediate carry propagation. Finally, a Carry Propagate Adder (CPA) is used only in the last cycle to produce the final product, resulting in reduced delay and enhanced hardware efficiency.

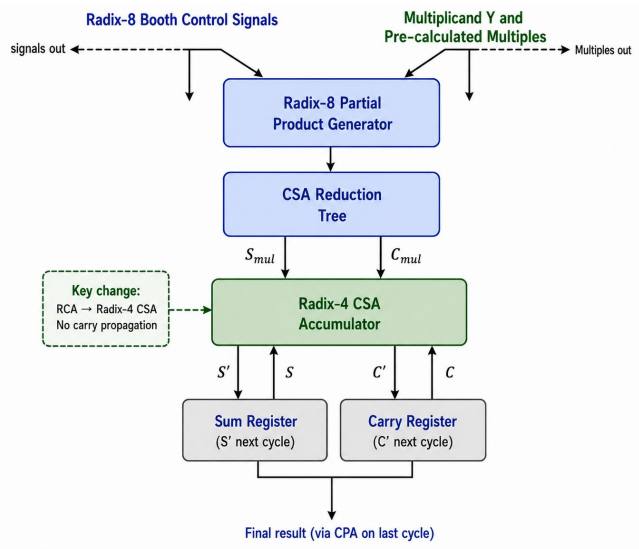

Fig. 4. Datapath of the proposed FSA PE

Figure 5 represents the core building block of the Radix-4 Adder Unit (RD4A), designed to process two bits of the operands simultaneously. The RD4A unit accepts five inputs: two bits from the first operand (A_i, A_{i+1}), two bits from the second operand (B_i, B_{i+1}), and a carry-in (C_{in}). It generates three outputs: two sum bits and a carry-out whose logical functions are as given in Eqns. (6) to (10).

$$Sum_i = A_i \oplus B_i \oplus C_{in} \quad (6)$$

$$C_i = (A_i \wedge B_i) \vee (A_i \wedge C_{in}) \vee (B_i \wedge C_{in}) \quad (7)$$

$$Sum_{i+1} = A_{i+1} \oplus B_i + 1 \oplus C_i \quad (8)$$

$$C_{out} = (A_{i+1} \wedge B_{i+1}) \vee (A_{i+1} \wedge C_i) \vee (B_{i+1} \wedge C_i) \quad (9)$$

where C_i is the intermediate carry generated and used internally within the RD4A unit.

An N-bit accumulator is built using N/2 such RD4A units. The accumulator computes $Acc_{new} = S_{mult} + C_{mult} + Acc_{old}$. In the proposed design, Acc_{old} is also stored in carry-save format (S′ and C′ vectors). The N/2 RD4A units perform the combined addition ($S_{mult} + C_{mult} + S' + C'$) each cycle. Crucially, the carry-out signals (Eqn. 9) between RD4A blocks are registered alongside the sum bits, rather than being immediately propagated within the same clock cycle. The registered sum bits form the new sum vector S, and the appropriately shifted registered inter-block carries form the new carry vector C. These become the S′ and C′ inputs for the subsequent accumulation cycle.

By maintaining the accumulated value in this redundant carry-save format and registering carries between RD4A blocks, the critical path associated with full carry propagation across the N-bit accumulator width is entirely eliminated.

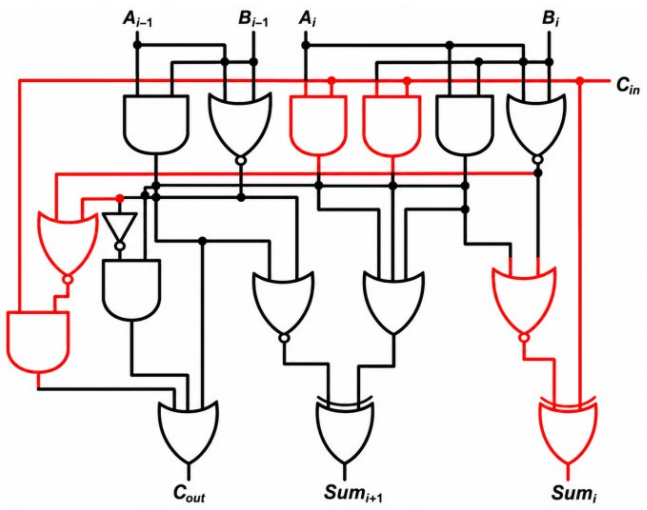

Fig. 5. The radix-4 adder structure

The proposed Radix-4 CSA-based MAC exhibits inherent scalability owing to its modular architecture, enabling efficient expansion to larger SA configurations. The current implementation employs a 16-bit datapath, which can be seamlessly extended to 32-bit and 64-bit MAC units commonly used in modern ML accelerators. The Radix-4 CSA carry-save accumulation mechanism eliminates intermediate carry propagation during accumulation cycles. Consequently, as the array size increases, the reduction in accumulation latency becomes more pronounced by eliminating the carry-propagation overhead.

In addition, the low power consumption of each PE enhances overall energy efficiency in larger arrays by preventing excessive power growth during scaling. The factored computation approach also decreases redundant operations and facilitates efficient data propagation between neighbouring PEs, which is beneficial for large-scale implementations. Furthermore, the modular structure simplifies routing and integration on FPGA or ASIC platforms, enabling enhanced scalability in throughput and parallel processing capabilities.

5 Results and discussion

Synthesis, placement and routing were performed using Xilinx ISE Design Suite Version 14.7, targeting the Spartan-3E FPGA platform. The implementation results provided post-synthesis estimates of hardware utilization, timing and power for comparative analysis.

Five hardware configurations were evaluated at the Processing Element (PE) level, as the primary objective was to isolate and evaluate accumulator-level optimization within a single PE datapath: the baseline MAC with RCA accumulator, standalone 8×8 Array and Wallace Tree multipliers for comparison, the proposed MAC with Radix-4 CSA accumulator, and a proposed PE variant using the Lower-part OR Adder. All designs were implemented using a 16-bit datapath. Table 2 presents the hardware resource utilization, while Tab. 3 reports the design metrics for further performance analysis.

5.1 Area efficiency (resource utilization)

Table 2 presents the hardware resource utilization. Compared to the baseline RCA-based MAC, the proposed Radix-4 CSA MAC achieves a 21.3% reduction in LUT (282 to 222), a 24.5% reduction in Slice (152 to 116) and an 11.1% reduction in estimated gate count (1896 to 1698). The LOA-based PE variant achieves further reductions (LUTs: 216, Slices: 102, Gates: 1574), consistent with the elimination of carry logic in its lower bits. Fig. 6 provides a graphical representation of the performance comparison. The reduced hardware overhead enables improved scalability, allowing a greater number of PEs to be integrated within the same silicon area.

Table 2. Hardware resource utilization

Design	LUTs	Slices	Gates
Array multiplier	285	124	1838
Wallace tree multiplier	290	120	1928
Existing MAC (RCA)	282	152	1896
Proposed MAC (CSA)	222	116	1698
Proposed PE (LOA)	216	102	1574

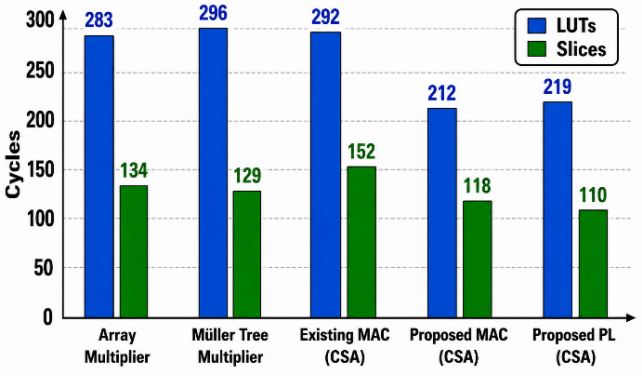

Fig. 6. Hardware resource utilization comparison

5.2 Timing performance

Table 3 presents the design metric analysis evaluated in terms of power consumption, delay and Power-Delay product (PDP). The most significant benefit of the proposed design is the reduction in critical path delay. The proposed MAC with Radix-4 CSA accumulator achieves a minimum path delay of 6.815 ns, compared to 8.791 ns for the baseline RCA-based MAC, for a 22.5% reduction. This empirically confirms that the accumulation stage was the primary delay component and that the carry-save technique effectively eliminates it. The LOA-based PE variant achieves the lowest delay at 5.176 ns. The standalone 8×8 Wallace Tree multiplier exhibits a delay of 8.462 ns and the Array multiplier 9.231 ns. It confirms that optimizing the complete MAC feedback loop yields greater speed benefits than minimizing the multiplication stage alone. The reduction in MAC delay translates directly into a proportionally higher achievable F_{max} for an FSA built from these augmented MACs.

Since the MAC unit serves as the primary computational kernel within each PE, optimizing its architecture directly impacts overall ML accelerator performance. The proposed design primarily targets ML workloads dominated by matrix multiplication and convolution operations, such as CNNs, transformer-based inference engines, and edge-AI accelerators. In these applications, a large number of MAC operations are executed concurrently within SAs, making accumulator latency a critical factor influencing performance and power consumption. Consequently, the proposed accumulator optimization not only improves PE-level timing performance but also enhances the computational efficiency of large-scale systolic-array-based ML accelerators.

5.3 Power consumption and power-delay product

As shown in Tab. 3, compared with the baseline existing MAC (RCA), the proposed MAC with the Radix-4 CSA achieves a 4.21% reduction in estimated power consumption (438.94 mW vs. 458.21 mW). The proposed LOA-based PE achieves a 7.06% reduction (425.87 mW vs. 458.21 mW), confirming the suitability of the proposed LOA-based architecture for multimedia applications.

Table 3. Design metric analysis

Design	Power (mW)	Delay (ns)	PDP (pJ)
Array multiplier	557.75	9.231	5148.6
Wallace tree multiplier	528.84	8.462	4475.0
Existing MAC (RCA)	458.21	8.791	4028.1
Proposed MAC (CSA)	438.94	6.815	2991.4
Proposed PE (LOA)	425.87	5.176	2204.3

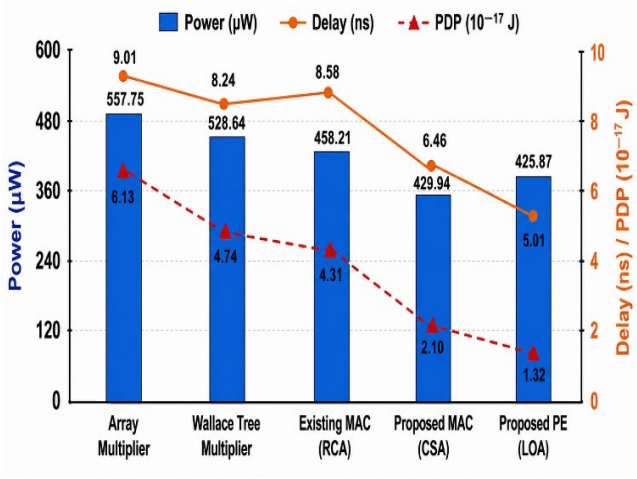

Fig. 7. Design metric comparison: Power, Delay and PDP

Figure 7 shows that power consumption, propagation delay and PDP decrease progressively from the standalone multiplier to the proposed MAC architectures. The Power-Delay Product (PDP) provides a composite efficiency metric: the proposed MAC achieves 2991 pJ vs. 4028 pJ for the baseline, a 25.7% improvement and the LOA variant achieves the best PDP at 2204 pJ. The Radix-4 CSA-based MAC provides a balanced trade-off between power and delay, while the LOA-based MAC achieves the highest energy efficiency

through the lowest PDP. These findings highlight the suitability of the proposed MAC architectures for deployment in ML and AI inference accelerators, where minimizing energy consumption while maintaining high processing performance is essential.

5.4 Error metric evaluation

Error metrics such as Mean Absolute Error (MAE) and Error Rate (ER) are analyzed to evaluate the LOA-based accumulator. The MAE measures the average magnitude of errors between exact and approximate outputs, as given by Eqn. (10)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y_i}| \tag{10}$$

where n represents the total number of samples, y_i denotes the exact output, $\hat{y_i}$ represents the approximate output, and $|y_i - \hat{y_i}|$ corresponds to the absolute error for each sample.

The Error Rate (ER) quantifies the percentage of erroneous outputs among all evaluated samples and is calculated as shown in Eqn. (11)

$$ER = \frac{N_{error}}{N_{total}} \tag{11}$$

where N_{error} is the number of erroneous outputs and N_{total} is the total number of evaluated outputs.

Table 4 presents the error metric analysis of the proposed architecture. The MAE evaluation was performed by analyzing all possible input combinations for the 8-bit approximate accumulator. Since both operands are 8-bit values, a total of 65,536 input combinations were evaluated to compute the average approximation error. The evaluated LOA-based design achieved a Mean Absolute Error (MAE) of 620.17 with a corresponding error rate of 77.3%.

Table 4. Error metric analysis

Error metrics	Values
Mean absolute error	620.17
Error rate (%)	77.3

The approximation errors are primarily confined to the lower significant bits and therefore have reduced impact on overall ML inference, demonstrating the expected trade-off between computational accuracy and hardware efficiency in approximate accumulation architectures

6 Conclusion

This paper addressed the critical issue of accumulation latency in Radix-8 Factored Systolic Array MACs used for accelerating ML workloads. The conventional RCA in the accumulation stage is identified as the primary performance setback. To overcome this issue, a Radix-4 Carry-Save Adder accumulator was proposed that eliminated the critical path dependency on carry propagation. The synthesis results demonstrate that the proposed MAC with the Radix-4 CSA achieves significant reductions in critical-path delay, LUT and Slice utilization compared with the baseline design and other existing multiplier architectures. The power-delay product improves by 25.7%, confirming that the performance and area benefits are not obtained at the expense of increased power. The LOA-based PE offers a favourable trade-off between accuracy and efficiency, thereby emerging as a promising solution for efficient ML accelerator implementations. The proposed framework offers a scalable and efficient solution for next-generation ML accelerators. Future work will focus on extending the analysis to larger array sizes and modern FPGA families, followed by the integration of highly optimized accumulation techniques.

Acknowledgements

The authors thank the management, CSI College of Engineering, Ooty, The Nilgiris, for providing laboratory facilities.

References

[1] K. Inayat, I. Ullah, and J. Chung, “Factored Systolic Arrays Based on Radix-8 Multiplication for Machine Learning Acceleration,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 32, no. 7, pp. 1205–1215, 2024.

[2] S. N. Kumar, S. V. Kumar, J. Abishek, and R. Sakthivel, “Design of High-Speed Machine Learning Accelerators using Hybrid Accumulator in Systolic Arrays,” in Proc. 6th Int. Conf. Electronics Sustainable Commun. Syst. (ICESC), pp. 500–507, 2025.

[3] F. B. Muslim, K. Inayat, M. Z. Siddiqi, S. Khan, T. Mahmood, and I. ul Islam, “SAPER-AI accelerator: asystolic array-based power-efficient reconfigurable AI accelerator,” Front. Inf. Technol. Electron. Eng., vol. 26, no. 9, pp. 1624–1636, 2025.

[4] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, “Efficient processing of deep neural networks: A tutorial and survey,” Proc. IEEE, vol. 105, no. 12, pp. 2295–2329, 2017.

[5] H. T. Kung, “Why systolic architectures?” Computer, vol. 15, no. 1, pp. 37–46, 1982.

[6] N. P. Jouppi et al., “In-datacenter performance analysis of a tensor processing unit,” in Proc. ACM/IEEE 44th Annu. Int. Symp. Comput. Archit. (ISCA), pp. 1–12, 2017.

[7] J. Abdelmaksoud, C. Sestito, S. Wang and T. Prodromakis, "ADiP: Adaptive-Precision Systolic Array for Matrix Multiplication Acceleration," in IEEE Open Journal of the Solid-State Circuits Society, pp. 1–1, 2026.

[8] D. K. J. Rajanendiran, C.G. Babu and K. Priyadharsini, "A certain examination on heterogeneous systolic array (HSA) design for deep learning accelerations with low power computations", Sustainable Computing: Informatics and Systems, 44, 101042, 2017.

[9] T. Suguna, B. K. Suresh, and A. M. Malim, "Design and performance analysis of multiply and accumulate (MAC) unit for machine learning acceleration," J. Comput. Sci., vol. 19, no. 3, pp. 19–33, 2026.

[10] D. N. Devi, G. Ajay Kumar, B. G. Gowda, and M. Rao, "Integrated MAC-based systolic arrays: Design and performance evaluation," in Proc. Great Lakes Symp. VLSI, pp. 292–295, 2024.

[11] Del Barrio and R. Hermida, "A slack-based approach to efficiently deploy radix-8 Booth multipliers," in Proc. Design, Autom. Test Eur. Conf. (DATE), pp. 1153–1158, 2017.

[12] N. H. E. Weste and D. M. Harris, CMOS VLSI Design: A Circuits and Systems Perspective, 4th ed. Boston, MA, USA: Addison-Wesley, 2010.

[13] Q. Song, Y. Dai, H. Lu and G. Jin, "High-throughput systolic array-based accelerator for hybrid transformer-CNN networks," J. King Saud Univ. Comput. Inf. Sci., vol. 36, no. 8, p. 102194, 2024.

[14] M. S. Kumar, D. A. Kumar and P. Samundiswary, "Design and performance analysis of Multiply-Accumulate (MAC) unit," in Proc. Int. Conf. Circuits, Power Comput. Technol. (ICCPCT), pp. 1084–1089,2014.

[15] S. Zhang, J. Gu, S. Yin, L. Liu, and S. Wei, "A multiple-precision multiply and accumulation design with multiply-add merged strategy for AI accelerating," in Proc. 26th Asia South Pacific Design Autom. Conf. (ASPDAC), pp. 229–234, 2021.

[16] S. Agasthiya, S. Balambigai, and R. Sarankumar, "An efficient Radix-8 factored systolic array for machine learning acceleration," in Proc. 5th Int. Conf. Trends Mater. Sci. Inventive Mater. (ICTMIM), pp. 1437–1444, 2025

[17] S. Zhang, J. Gu, S. Yin, L. Liu, and S. Wei, "A multiple-precision multiply and accumulation design with multiply-add merged strategy for AI accelerating," in Proc. 26th Asia South Pacific Design Autom. Conf. (ASPDAC), pp. 229–234,2021.

[18] P. Gurjar, R. Solanki, P. Kansliwal, and M. Vucha, "VLSI implementation of adders for high speed ALU," in Proc. Annu. IEEE India Conf., pp. 1–6, 2011.

[19] R. A. Javali, R. J. Nayak, A. M. Mhetar, and M. C. Lakkannavar, "Design of high speed carry save adder using carry lookahead adder," in Proc. Int. Conf. Circuits, Commun., Control Comput., pp. 33–36, 2014.

[20] M. D. Ercegovac and T. Lang, Digital Arithmetic. San Francisco, CA, USA: Morgan Kaufmann, 2003.

[21] M. Hassan, M. S. Ansari, and W. A. Khan, "Performance analysis of carry select adder based MAC unit for DSP applications," Microelectron. J., vol. 90, pp. 1–8, 2019.

[22] D. N. Devi, G. A. Kumar, B. G. Gowda, and M. Rao, "Performance-aware design of approximate integrated MAC factored systolic array accelerators," in Proc. 25th Int. Symp. Qual. Electron. Des. (ISQED), pp. 1–8, 2024.

[23] P. Jaswal, L. H. Krishna, and B. Srinivasu, "Energy efficient exact and approximate systolic array architecture for matrix multiplication," in Proc. 39th Int. Conf. VLSI Des. & 25th Int. Conf. Embedded Syst. (VLSID), pp. 524–529, 2026.

[24] P. M. Mohan and S. S. Kumar, "Design and analysis of high speed low power radix-4 carry save adder," in Proc. Int. Conf. Commun. Signal Process. (ICCSP), pp. 1016–1020, 2016.

Received 13 April 2026