

A Survey on FPGA-Based Precision-Scalable Systolic Array Accelerators for DSP and Machine Learning Applications

¹Monisha M Reddy, ²Likitha G S, ³Sharanya Varsha Nayak, ⁴Yashupradha B S,
⁵Dr. Shalini J

^{1,2,3,4}Student, ⁵Asst. Professor
^{1,2,3,4,5}Electronics and Communications,
^{1,2,3,4,5}RV Institute of Technology and Management, Bangalore, Karnataka

Abstract: Systolic array architectures have emerged as a prominent design paradigm for accelerating computationally intensive tasks due to their inherent parallelism and regular dataflow structure. While recent research has heavily favored optimizations for machine learning (ML) workloads—specifically convolutional and transformer-based neural networks—their integration into classical digital signal processing (DSP) domains such as spectral analysis and filtering remains relatively sparse.

This survey provides a comprehensive analysis of the current landscape of FPGA-based systolic arrays, evaluating advancements in arithmetic unit design, dataflow strategies, and precision scaling. A critical review of existing literature reveals a prevailing reliance on fixed-precision arithmetic, with most frameworks being monolithic and specialized for neural network execution. Despite the maturity of approximate computing and energy-efficient methodologies, their adoption within scalable, general-purpose systolic fabrics is still in its infancy.

Furthermore, this paper identifies fundamental research gaps, notably the absence of unified, precision-scalable architectures capable of adapting to the deterministic requirements of DSP workloads. We propose a structured taxonomy to categories for these advancements and discuss future research trajectories aimed at developing versatile, next-generation accelerators. This work serves as a technical reference to bridge the gap between specialized AI hardware and adaptive signal processing engines.

Index Terms – Systolic Array, FPGA, Digital Signal Processing (DSP), Precision-Scalable Architecture, Approximate Computing, Dataflow Optimization

I.INTRODUCTION

The exponential growth of computationally intensive workloads in signal processing, machine learning, and embedded systems has necessitated the development of high performance, energy-efficient hardware accelerators. Field Programmable Gate Arrays (FPGAs) have become a leading platform for these designs, offering a unique combination of flexibility, reconfigurability, and the capacity for massive parallelism. Within this landscape, systolic array architectures have distinguished themselves through their efficiency in executing repetitive arithmetic operations, specifically multiply accumulated (MAC) functions. Architecturally, a systolic array is defined by a spatial grid of processing elements (PEs) that operate synchronously, transferring data via localized interconnections. This highly regular structure facilitates superior throughput and hardware utilization, rendering systolic arrays exceptionally effective for large-scale matrix computations.

While significant research has focused on optimizing these arrays for deep learning models—such as convolutional neural networks (CNNs) and transformers—their application in traditional digital signal processing (DSP) remains an area with substantial potential for further integration and advancement [3], [9], [18]. In parallel, several studies have explored FPGA-based implementations of systolic array architectures with optimizations in dataflow, arithmetic units, and memory access patterns. Techniques such as Winograd convolution [3], activation stationary dataflow [30], and multi-mode reconfigurable architectures [29] have been proposed to improve computational efficiency and reduce memory bandwidth requirements. Additionally, approximate computing and quantization-based approaches have been investigated to achieve energy-efficient designs [1], [12].

Despite these advancements, the application of systolic arrays to classical digital signal processing (DSP) tasks has received comparatively less attention. While implementations of FIR filters [25], FFT accelerators [26]–[28] and matrix computation engines [19], [23] exist, these designs are often developed independently of modern architectural optimizations and lack integration with advanced features such as precision scalability and adaptive control mechanisms.

Another important observation is that most existing systolic array designs rely on fixed-precision arithmetic, limiting their flexibility across diverse application requirements. Although several works have proposed approximate arithmetic units and multi-precision computation techniques [2], [22], their integration within scalable systolic array frameworks remains limited. Furthermore, dataflow optimization strategies are typically static and do not incorporate runtime monitoring or adaptive control mechanisms.

Unlike conventional surveys that treat systolic array optimizations in isolation, this work adopts a system-level perspective by examining the interplay between architectural design, arithmetic precision and application domains. Existing literature largely

focuses on optimizing systolic arrays for deep learning workloads, often overlooking their applicability to classical DSP systems. This paper argues that the lack of unified design methodologies—particularly in precision scalability and adaptive dataflow control—represents a fundamental bottleneck in extending systolic architectures across heterogeneous workloads. To address this gap, we introduce a structured taxonomy and provide a comparative analysis that highlights the trade-offs between performance, energy efficiency, and computational flexibility.

The main contributions of this survey are as follows:

- A structured classification of systolic array architectures based on application domain and design techniques
- A comparative analysis of recent works focusing on arithmetic optimization, dataflow strategies, and FPGA implementations.
- Identification of key research gaps in precision scalability, DSP integration, and adaptive architectures.
- Discussion of future research directions for next generation systolic array accelerators.

II. MATHEMATICAL MODEL OF SYSTOLIC ARRAYS

Systolic array architectures are primarily designed to accelerate matrix-based computations through distributed processing. The fundamental operation performed within each processing element (PE) is the multiply-accumulate (MAC) operation:

$$P_{out} = P_{in} + (A \times B)$$

Where A and B represent input operands, and P_{in} and P_{out} denote the accumulated partial sums. Matrix multiplication, a key operation in both DSP and machine learning applications, is defined as:

$$C = A \times B$$

where each element is computed as:

$$C_{i,j} = \sum^N A_{i,k} \cdot B_{k,j}$$

In a systolic array, this computation is mapped spatially across an $N \times N$ grid of processing elements, where data flows rhythmically between neighboring PEs. Input matrix elements are streamed into the array, and partial results are accumulated as data propagates through the structure over time. This spatial-temporal mapping enables high parallelism and efficient reuse of data, significantly improving throughput compared to conventional sequential architectures.

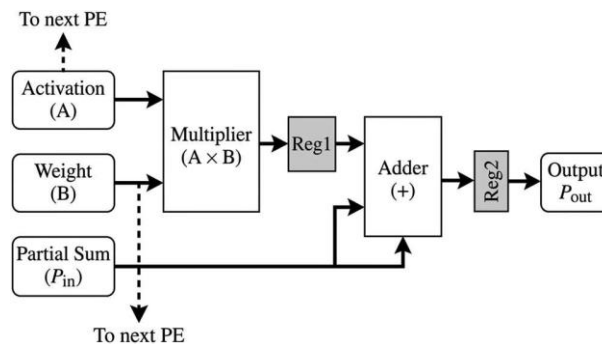


Fig. 1. Detailed architecture of the Processing Element (PE) for systolic array acceleration.

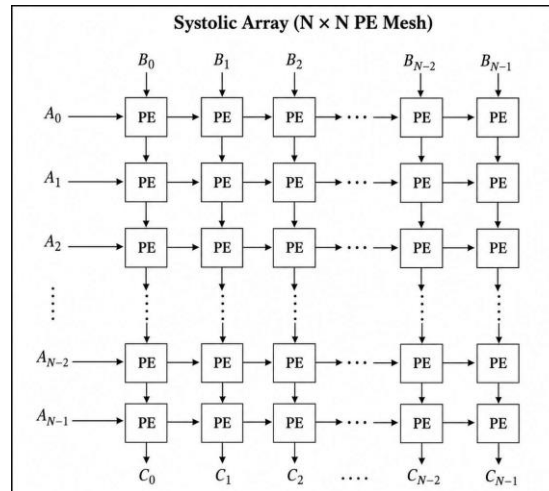


Fig. 2. Complete $N \times N$ systolic array architecture showing dataflow of input matrix A (horizontal), weight matrix B (vertical) and accumulation of partial sums across processing elements.

III. LITERATURE REVIEW

A. Systolic Arrays for DSP Applications

Systolic array architectures have been widely utilized for accelerating computationally intensive operations due to their inherent parallelism and regular dataflow structure. In the context of digital signal processing (DSP), these architectures are particularly effective for operations such as filtering, matrix multiplication, and spectral analysis. Several works have explored FPGA-based implementations of DSP functions using systolic arrays. For instance, systolic FIR filter architectures have demonstrated improved throughput and efficient hardware utilization for real-time signal processing applications [25]. Similarly, energy-efficient FIR filter designs have been proposed to reduce power consumption in domain-specific applications such as biomedical signal processing [24]. In addition to time-domain processing, frequency-domain operations such as Fast Fourier Transform (FFT) have also been accelerated using FPGA-based architectures. Various FFT accelerators have been proposed to achieve high-speed computation and support real-time processing in edge computing environments [26]–[28]. Furthermore, systolic array-based matrix multiplication engines have been investigated to optimize DSP workloads, emphasizing efficient data movement and parallel computation [19], [23]. Despite these advancements, many DSP-oriented implementations rely on fixed precision arithmetic and lack integration with modern architectural optimizations.

B. Core Systolic Array Architectures

The fundamental design of systolic arrays consists of a network of processing elements (PEs) that perform synchronized computations and pass data locally between neighboring units. Numerous studies have focused on optimizing systolic architectures for high-performance computing applications. For example, pipelined processing element designs have been proposed to improve throughput and reduce latency in matrix multiplication tasks [21]. Similarly, radix-based and hybrid systolic architectures have been explored to enhance computational efficiency in machine learning workloads [20]. Other works have introduced hybrid computation strategies that combine serial and parallel processing to improve resource utilization and efficiency in edge devices [6]. Accumulator optimization techniques have also been proposed to enhance the performance of systolic arrays in deep learning applications [10]. While these architectures demonstrate significant improvements in performance, they are often tailored for specific applications and lack flexibility for supporting diverse workloads such as DSP operations.

C. Arithmetic Unit Design for Systolic Arrays

The performance of systolic array architectures is highly dependent on the efficiency of the underlying arithmetic units, particularly multipliers and adders. Approximate computing techniques have been widely investigated to reduce hardware complexity and power consumption. Approximate array multipliers have been proposed to achieve significant reductions in area and energy consumption at the cost of controlled computational error [1]. Similarly, FPGA-based implementations of approximate multipliers have demonstrated improved energy efficiency and performance [4]. In contrast, accuracy-oriented designs focus on enhancing computational speed and precision through advanced multiplier architectures. Hybrid compressor-based multipliers and optimized binary multiplier designs have been proposed to improve speed and hardware efficiency [2]. Additionally, modified parallel multipliers have been integrated into systolic array architectures to improve performance in machine learning applications [11]. Despite these advancements, most works focus on arithmetic unit optimization independently, with limited integration into scalable systolic array frameworks supporting precision adaptability. A key trade-off exists between approximate and exact arithmetic units. While approximate multipliers reduce area and power consumption, they introduce computational errors that may affect accuracy. In contrast, exact multipliers ensure high precision but incur higher hardware costs. Therefore, selecting the appropriate arithmetic design depends on application requirements, particularly in DSP systems where accuracy constraints vary across different operations.

D. Dataflow and Memory Optimization Techniques Efficient

Dataflow management is critical for maximizing the performance of systolic array architectures. Various dataflow strategies, including weight stationary, output stationary, and activation stationary approaches, have been proposed to optimize memory access and data reuse. Activation stationary dataflow, for example, has been shown to improve data reuse and reduce memory bandwidth requirements in deep learning accelerators [30]. Recent works have introduced advanced dataflow techniques such as channel-wise stationary approaches and multi-dataflow architectures to enhance flexibility and performance [14], [29]. These techniques aim to reduce data movement overhead and improve overall system efficiency. However, most existing dataflow optimization strategies are designed for machine learning workloads and lack adaptability for general-purpose DSP applications. Furthermore, runtime dataflow monitoring and adaptive control mechanisms remain largely unexplored. Different dataflow strategies offer distinct trade-offs. Weight stationary approaches minimize weight movement but increase activation data traffic, whereas activation stationary methods improve data reuse at the cost of increased storage requirements within processing elements. Selecting an optimal dataflow strategy is therefore dependent on memory constraints and application characteristics.

E. Precision Scaling and Energy Optimization

Energy efficiency has become a critical consideration in modern hardware accelerator design. Several studies have explored precision scaling techniques, including quantization and reduced bit-width computation, to achieve lower power consumption while maintaining acceptable accuracy levels. Quantization-based systolic array accelerators have demonstrated significant improvements in energy efficiency, particularly in edge computing scenarios [12], [13]. In addition, multi-precision arithmetic approaches, such as posit-based systolic array accelerators, provide enhanced flexibility by supporting multiple numerical formats [22]. Low-power design techniques have also been applied to DSP architectures, including FIR filters, to optimize energy consumption and hardware utilization [24]. Despite these advancements, most implementations rely on static precision configurations and do not support dynamic precision adaptation during runtime.

F. Precision-Scalable Systolic Array Architectures

Precision scalability has emerged as a critical requirement in modern hardware accelerators, particularly for balancing performance, energy efficiency, and computational accuracy. Existing systolic array designs predominantly employ fixed-precision arithmetic, limiting their adaptability across diverse workloads. Precision scaling techniques can be broadly classified into static and dynamic approaches. In static precision scaling, the bit-width of computation is fixed during design time, typically using reduced precision formats such as 8-bit or 16-bit representations to improve performance and reduce power consumption [12], [13]. While effective, this approach lacks flexibility for applications requiring varying levels of accuracy. Dynamic precision scaling enables runtime adaptation of computational precision based on workload requirements. This approach allows the system to switch between high-precision and low-power modes, improving overall efficiency. Multi-precision architectures, such as posit-based systolic arrays, have been proposed to support multiple numerical formats within a single hardware framework [22]. In addition, approximate computing techniques provide an alternative approach to precision scaling by intentionally introducing controlled inaccuracies to reduce hardware complexity and energy consumption. Approximate multipliers, for example, can significantly reduce area and power usage while maintaining acceptable accuracy for error-tolerant applications [1], [4]. Despite these advancements, the integration of precision-scalable arithmetic units within systolic array architectures remains limited. Most existing designs do not support dynamic switching between approximate and exact computation modes, highlighting a key research gap in the development of flexible and efficient systolic array accelerators.

G. FPGA-Based AI Accelerators

Systolic array architectures have been extensively utilized in FPGA-based accelerators for machine learning applications. Techniques such as Winograd-based convolution have been proposed to reduce computational complexity and improve efficiency in convolutional neural networks (CNNs) [3]. Similarly, high-frequency systolic array designs have been developed to accelerate transformer-based models [9]. Other works have focused on implementing deep learning models, including CNN and LSTM architectures, on FPGA platforms for real-time applications such as signal recognition [8]. Power-efficient and high-performance FPGA-based accelerators have also been proposed to support advanced deep learning algorithms [7]. While these designs demonstrate significant performance improvements, they are predominantly optimized for neural network workloads and do not directly address the requirements of classical DSP applications.

H. Reconfigurable and Advanced Architectures

To improve flexibility and adaptability, recent research has focused on reconfigurable systolic array architectures. Multi-mode architectures enable dynamic adaptation to different computational workloads, improving resource utilization, and performance [29]. Reconfigurable systolic array-based accelerators have also been proposed to achieve power efficiency and scalability in AI applications [15]. Advanced architectures incorporating novel data movement strategies, such as triangular input movement and sparse computation techniques, further enhance performance and scalability [16], [17]. Application-specific accelerators, such as those designed for object detection using YOLO models, demonstrate the effectiveness of systolic arrays in high-performance computing scenarios [18]. However, these architectures are primarily optimized for machine learning applications and provide limited support for precision-scalable DSP processing.

IV. PROPOSED TAXONOMY OF SYSTOLIC ARRAY ARCHITECTURES

This taxonomy is proposed to unify fragmented research directions and provide a structured framework for analyzing systolic array design trade-offs. Systolic array architectures can be classified along three orthogonal dimensions: computation domain, arithmetic precision, and dataflow strategy.

- 1) Application Domain: Architectures are categorized into DSP-oriented, machine learning-oriented and hybrid systems. DSP-oriented designs prioritize deterministic latency and precision, whereas ML-oriented systems emphasize throughput and energy efficiency.
- 2) Arithmetic Precision: Architectures are classified as fixed precision, quantized, approximate, or dynamically scalable. Fixed-precision designs offer simplicity but lack adaptability. In contrast, dynamic precision architectures enable runtime trade-offs between accuracy and power consumption.
- 3) Dataflow Strategy: Dataflow mechanisms are categorized into weight stationary, output stationary, activation stationary, and adaptive dataflow. Adaptive dataflow represents an emerging paradigm where runtime conditions influence data movement.
- 4) Reconfigurability: Architectures are further distinguished by their ability to adapt to workload variations, including static, partially reconfigurable and fully dynamic systems. This taxonomy highlights that current research predominantly focuses on isolated optimization axes, with limited exploration of integrated multi-dimensional design approaches.

V. COMPARISON OF EXISTING WORKS

Table I provides a cross-dimensional comparison of 30 reported systolic-array architectures considering precision, arithmetic structure, dataflow, reconfigurability, power optimization, target platform, and application domain. Rather than treating these parameters independently, the comparison reveals several architectural trends that are important for the development of flexible FPGA-based systolic accelerators.

The first observation is the strong dominance of fixed-precision computation. Twenty of the 30 reviewed architectures employ fixed precision, while the remaining designs mainly use quantization, dynamic precision, or multi-precision approaches. This indicates that precision is still predominantly selected during the design stage rather than according to the characteristics of the incoming workload. Although fixed precision simplifies hardware control and timing, it can result in unnecessary computational cost when the input data does not require the maximum available numerical range.

The comparison also shows a strong concentration on machine-learning-oriented applications. CNN, transformer, deep-learning, YOLO, and other AI accelerators account for a substantial portion of the surveyed architectures. These designs generally prioritize throughput, data reuse, and energy efficiency through optimized MAC structures and stationary or streaming dataflows. In contrast, DSP-oriented implementations such as FIR filters, FFT accelerators, and general matrix engines appear less frequently and are generally developed as application-specific architectures. This imbalance suggests that the architectural techniques developed for AI acceleration have not yet been fully generalized to conventional DSP workloads.

A second important trend concerns reconfigurability. Only a small subset of the reviewed architectures provides full dynamic reconfiguration, including SAPER, multi-precision SIMD systolic designs, and MRSA. Several other architectures provide partial reconfigurability, but most remain statically configured. Consequently, there is a distinction between an architecture that supports multiple configurations and one that can autonomously select an appropriate configuration during operation. The latter capability is particularly relevant for workloads whose numerical requirements vary over time.

The arithmetic structures reported in the literature also demonstrate a clear specialization trend. Conventional MAC-based structures remain dominant because of their regularity and suitability for FPGA implementation. Approximate multipliers provide opportunities for reducing hardware cost and energy, whereas hybrid, radix-based, and modified multiplier structures primarily target arithmetic efficiency. However, these arithmetic improvements are generally evaluated as individual optimizations rather than as components of a runtime-adaptive systolic architecture.

Power optimization is another major design objective, particularly in edge-oriented AI accelerators. Quantization, approximate arithmetic, power-aware multiplier design, and reconfigurable architectures have demonstrated different approaches to reducing energy consumption. However, the comparison indicates that power optimization is often coupled to a predetermined precision or application rather than controlled dynamically according to input characteristics.

Overall, Table I indicates that existing architectures tend to optimize one or two dimensions—such as arithmetic efficiency, dataflow, precision, or power—rather than jointly optimizing all of them. The most significant opportunity therefore lies in combining these dimensions within a common systolic framework. Such a framework could retain the regularity and throughput advantages of conventional arrays while introducing runtime precision selection and workload-aware control for both DSP and ML applications.

Table 1. Comparison of Existing Works

Ref.	Precision	Arithmetic	Dataflow	Reconfig-uration	Power Opt.	Platform	Application
1	Fixed	Approx. Mult.	Conventional	No	Yes	ASIC/FPGA	Approx. Computing
2	Fixed	Hybrid Compressor	Conventional	No	Moderate	FPGA	DSP/ML
3	Fixed	Optimized MAC	Winograd	Partial	Yes	FPGA	CNN
4	Fixed	Approx. Mult.	Conventional	No	Yes	Spartan-6	Approx. Computing
5	Fixed	MAC	Streaming	No	Moderate	FPGA	Deconvolution
6	Fixed	Serial/Parallel MAC	Hybrid	Partial	Yes	FPGA	Edge AI
7	Fixed	MAC	Stationary	No	High	FPGA	Deep Learning
8	Fixed	MAC	Streaming	No	Moderate	FPGA	CNN/LSTM
9	Fixed	MAC Array	Systolic Streaming	No	Moderate	FPGA	Transformer
10	Fixed	Hybrid Accumulator	Stationary	No	Moderate	FPGA/ASIC	ML
11	Fixed	Parallel Mult.	Stationary	No	Yes	FPGA	ML
12	Quantized	Quantized MAC	Stationary	No	High	FPGA	Edge AI
13	Quantized	Quantized MAC	Stationary	No	High	FPGA	Edge Computing
14	Fixed	MAC	Multi-Dataflow	Yes	Moderate	FPGA	AI Accelerator
15	Dynamic	MAC	Adaptive	Yes	High	FPGA	Reconfigurable AI
16	Fixed	MAC	Triangular	Partial	Moderate	FPGA	CNN
17	Quantized	Sparse MAC	Sparse	Partial	High	FPGA	Sparse CNN
18	Fixed	MAC Array	Stationary	Partial	Moderate	FPGA	YOLO
19	Fixed	MAC	Streaming	No	Moderate	FPGA	Matrix Multiplication
20	Fixed	Radix-8 Mult.	Systolic	No	Moderate	FPGA/ASIC	ML
21	Fixed	Pipelined PE	Streaming	No	Moderate	FPGA	DNN
22	Multi-Precision	SIMD Posit	Stationary	Yes	High	FPGA	Deep Learning
23	Fixed	DSP-Optimized MAC	Streaming	Partial	Moderate	FPGA	Matrix Engine
24	Fixed	MAC	Streaming	No	High	FPGA	FIR Filter
25	Fixed	MAC	Streaming	No	Moderate	FPGA	FIR Filter
26	Fixed	Butterfly	Pipeline	No	Moderate	FPGA	FFT
27	Fixed	Butterfly	Pipeline	No	Moderate	FPGA	FFT
28	Fixed	FFT Core	Pipeline	Partial	Moderate	PYNQ FPGA	FFT
29	Dynamic	MAC	Channel-Wise Stationary	Yes	High	FPGA	CNN
30	Fixed	MAC Array	Activation Stationary	No	Moderate	FPGA	Fully Connected NN

VI. DISCUSSION

The comparative analysis indicates that current systolic array research is largely driven by machine learning applications, with emphasis on throughput, data reuse, and energy-efficient computation. Most reviewed architectures employ fixed or quantized precision, while only a limited number of support dynamic or multi-precision operation. This restricts their ability to adapt computational resources according to changing input characteristics.

The analysis also reveals that arithmetic optimization, dataflow organization, and reconfigurability are commonly investigated as separate design objectives. Approximate multipliers can reduce hardware and energy requirements, whereas optimized dataflows improve data reuse and memory efficiency; however, their combined use with runtime precision control remains limited.

For DSP applications, this limitation is particularly relevant because different signal-processing operations may require different numerical accuracy levels. A fixed high-precision architecture can therefore consume unnecessary computational resources for inputs with a limited dynamic range. A workload-aware systolic architecture that dynamically selects precision based on input characteristics could provide a better balance between accuracy, performance, and energy consumption.

Consequently, future systolic-array designs should focus on integrating lightweight runtime monitoring, precision adaptation, and efficient arithmetic within a common FPGA-based architecture. Such integration could improve flexibility while retaining the regularity and parallelism that make systolic arrays attractive for both DSP and ML workloads.

VII. FUTURE DIRECTIONS

Based on these observations, future research directions should focus on the following key areas:

- **Dynamic Precision Scaling:** Developing systolic arrays capable of run-time bit-width adaptation to balance computation accuracy and power consumption.
- **Hybrid Arithmetic Frameworks:** Integrating approximate and exact computing elements within a single, unified systolic fabric.
- **Adaptive Dataflow Control:** Moving beyond static, design-time schedules toward runtime reconfigurable dataflows to accommodate varied workloads.
- **Convergent Architectures:** Designing systolic architectures that bridge the gap between classical DSP algorithms and AI-orientated computing paradigms.
- **Edge-Optimized Design:** Implementing energy-efficient, robust accelerators specifically tailored for real-time FPGA-based edge computing scenarios.

VIII. CONCLUSION

This paper presented a comprehensive survey of systolic array architectures, tracking their evolution and optimization across FPGA-based hardware acceleration domains. By systematically analyzing arithmetic optimizations, dataflow strategies, and precision scaling techniques, this work highlights a distinct evolutionary imbalance: while neural network acceleration has reached high architectural maturity, classical DSP applications remain structurally rigid and underserved. Ultimately, this survey establishes a foundational roadmap for the hardware community. By identifying the critical gaps in runtime adaptability, precision scaling, and algorithm convergence, this work serves as a catalyst for developing next-generation, versatile systolic architectures that bridge the divide between specialized AI accelerators and general-purpose signal processing engines.

REFERENCES

- [1] P. Balasubramanian, R. Nayar and D. Maskell, "Approximate array multipliers," *Electronics*, vol. 10, no. 3, pp. 1–18, 2021.
- [2] V. Thamizharasan and V. Parthipan, "Design of efficient binary multiplier architecture using hybrid compressor with FPGA implementation", *Journal of Electronics*, vol. 38, no. 4, pp. 623–632, 2021.
- [3] X. Liu, Y. Chen, C. Hao, A. Dhar and D. Chen, "WinoCNN: Kernel sharing Winograd systolic array for efficient convolutional neural network acceleration on FPGAs," *Proc. IEEE ASAP*, 2021, pp. 258–265.
- [4] G. Shobana, R. Chithiraimuthu and A. Adhithyavel, "Performance analysis and implementation of approximate multipliers on Spartan-6 FPGA," *International Journal of Health Sciences*, vol. 6, no. S3, pp. 10986–10995, 2022.
- [5] A. N. J. Raj, R. J. Arockiam and M. S. Kumar, "FPGA-based systolic deconvolution architecture for up sampling," *PeerJ Computer Science*, vol. 8, e1123, 2022.
- [6] S. Choi, H. Kim and J. Lee, "Enhancing computation efficiency of deep neural network processing on edge devices through serial/parallel systolic computing," *IEEE Access*, vol. 10, pp. 45213–45225, 2022.
- [7] J. Wang, L. Chen and Y. Zhang, "A high-performance and ultra-low power accelerator design for advanced deep learning algorithms on an FPGA," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 11, pp. 6201–6212, 2022.
- [8] H. Zhang, Y. Liu and Q. Sun, "Efficient FPGA implementation of CNN and LSTM for radar emitter signal recognition," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 3, pp. 2345–2356, 2023.
- [9] Y. Chen, X. Zhao and L. Wang, "High-frequency systolic array-based transformer accelerator on FPGA," *Electronics*, vol. 12, no. 15, pp. 1–15, 2023.
- [10] J. Park, S. Lee and K. Choi, "Hybrid accumulator factored systolic array for machine learning acceleration," *IEEE Transactions on VLSI Systems*, vol. 31, no. 7, pp. 1010–1021, 2023.
- [11] M. Kumar, S. Singh and R. Patel, "Power-intent systolic array using modified parallel multiplier for machine learning acceleration," *International Journal of VLSI Design*, vol. 2023, pp. 1–10, 2023.
- [12] A. Sanchez-Flores, J. Ramirez and L. Torres, "Energy and precision evaluation of a systolic array accelerator using quantization approach," *Electronics*, vol. 13, no. 2, pp. 1–16, 2024.
- [13] Y. Li, X. Zhou and T. Wang, "Energy and precision evaluation of a systolic array accelerator using a quantization approach for edge computing," *IEEE Edge AI Computing Journal*, vol. 1, no. 1, pp. 45–55, 2024.
- [14] M. Han, L. Wang and L. Xiao, "ReDas: A lightweight architecture for supporting fine-grained reshaping and multiple dataflows on systolic array," *IEEE Transactions on Computers*, vol. 73, no. 8, pp. 1997–2011, 2024.

- [15] F. B. Muslim, A. Rahman and M. A. Hossain, "SAPER: A systolic array-based power-efficient reconfigurable AI accelerator," *Frontiers of Information Technology & Electronic Engineering*, vol. 26, no. 1, pp. 45–58, 2025.
- [16] C. Sestito, S. Agwa and T. Prodromakis, "TrIM: Triangular input movement systolic array for convolutional neural networks: Architecture and hardware implementation," *IEEE Transactions on Circuits and Systems I*, vol. 72, no. 5, pp. 2263–2273, 2025.
- [17] J. Xu, H. Pu and D. Wang, "Sparse convolution FPGA accelerator based on multi-bank hash selection," *Micromachines*, vol. 16, no. 1, pp. 1–15, 2025.
- [18] Z. Zhang, Y. Li and K. Huang, "SARPAR: Systolic array pallet integrated accelerator for YOLO models on FPGA," *Integration, the VLSI Journal*, vol. 92, pp. 101–112, 2026.
- [19] A. Puscasu, C. B. Ciobanu and O. Buiu, "Systolic array matrix multiplication accelerator," in *Proc. IEEE CAS*, 2024.
- [20] S. Inayat, "Factored systolic arrays based on radix-8 multiplication for machine learning acceleration," *IEEE Transactions on VLSI Systems*, 2024.
- [21] E. S. Elango, S. Sagayaraj, D. Surendhar and M. Nivethitha, "VLSI implementation of pipelined PE systolic array-based 3×3 matrix multiplication for deep neural network accelerator," *Proc. IEEE STCR*, 2025.
- [22] G. Raut, D. Selvakumar, P. J. Edavoor and R. Thakur, "Multi-precision SIMD posit systolic array accelerator for deep learning," in *Proc. IEEE VLSI SAT*, 2025.
- [23] J. Li, T. Li, G. Shen, D. Zhao, Q. Zhang and Y. Zeng, "Revealing untapped DSP optimization potentials for FPGA-based systolic matrix engines," *Proc. IEEE FPL*, 2024.
- [24] T. Jayachandran and M. Priyanka, "Energy and area-efficient FIR filter architecture for low-power EEG signal processing," in *Proc. IEEE DevIC*, 2025.
- [25] A. R. Kumar, K. H. L. P. Prasad, B. Sriraj and P. Rajasri, "FPGA implementation of systolic FIR filter using single-channel method," in *Proc. IEEE Confluence*, 2023.
- [26] P. Schulz and G. Sleahitichi, "FPGA-based accelerator for FFT processing in edge computing," in *Proc. IEEE IDAACS*, 2023.
- [27] V. K. Challagundla, J. U. Kidav, S. K. Singh, A. J. Hati and S. K. Mandiga, "An FFT accelerator IP core implemented on FPGA," in *Proc. IEEE SCEECs*, 2025.
- [28] K. Agrawal and A. Asati, "Improved implementation of PYNQ-based FFT hardware accelerator," in *Proc. IEEE DICCT*, 2024.
- [29] X. Zhang, X. Song and J. Gao, "MRSA: A multi-mode reconfigurable systolic array architecture based on channel-wise stationary dataflow for CNNs," *Microelectronics Journal*, 2026.
- [30] H. Wan, C. Rao, Y. Zheng, P. Zhou and X. Lou, "A systolic array with activation stationary dataflow for deep fully connected networks," in *Proc. IEEE AICAS*, 2023.

Copyright & License:

© Authors retain the copyright of this article. This work is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.