Proceedings of CONF-APMM 2025 Symposium: Controlling Robotic Manipulator Using PWM Signals with Microcontrollers
DOI: 10.54254/2753-8818/2025.AD27977

In-Memory Computing for AI Acceleration and System Design

Haoming Lou

Hajim School of Engineering & Applied Sciences, University of Rochester, Rochester, USA
hlou@u.rochester.edu

Abstract. The energy and latency bottlenecks of traditional von Neumann architectures, caused by frequent data movement, are exposed by the massive parallel computation and high memory bandwidth demands of modern artificial intelligence (AI), especially deep learning. In-memory computing (IMC) integrates memory and computation to overcome the "memory wall," thus enabling significant performance gains in AI workloads. This paper reviews recent advances in IMC architectures, devices, and design methodologies, with a focus on their applications in neural network acceleration. In particular, representative IMC implementations are analyzed based on memory technologies such as SRAM, RRAM, and PCM, with key performance trade-offs involving speed, energy, area, and precision examined. Through an in-depth analysis of literature from the past decade, this study highlights the technical principles, circuit-level optimizations, system integration strategies, and fault-tolerance mechanisms used in existing IMC designs. The results demonstrate that IMC architectures can deliver one to two orders of magnitude improvements in energy efficiency compared to traditional digital accelerators, particularly for deep neural network inference. However, challenges such as limited training support, device variability, analog precision constraints, and system-level bottlenecks remain key obstacles.

Keywords: In-memory computing, Neural network acceleration, Memory technologies, Energy efficiency, Reliability and precision trade-offs

1. Introduction

Modern artificial intelligence (AI), particularly deep learning, has created an unprecedented demand for large-scale parallel matrix operations. However, traditional von Neumann architectures, which separate processing and memory, face the "memory wall," marked by significant energy and latency overheads due to frequent data movement between compute and storage units [1]. To alleviate the performance bottlenecks caused by frequent off-chip memory access in neural networks with billions of parameters, in-memory computing (IMC), also referred to as compute-in-memory (CIM), integrates computation directly into memory arrays [1]. IMC architectures typically exploit the electrical properties of memory devices, such as Ohm's and Kirchhoff's laws, to perform operations like matrix-vector multiplication (MVM) efficiently and in parallel. Prototype systems have shown orders-of-magnitude gains in energy efficiency over conventional accelerators [2]. Despite these advancements, several technical challenges persist in practical deployment, including the lack of

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2753-8818/2025.AD27977

standardized system-level design methodologies, device-level non-idealities (e.g., variability, drift, limited endurance), and the difficulty of supporting on-chip training with sufficient precision and stability. Besides, the reliable large-scale deployment of IMC systems still faces significant challenges in integration and calibration [3,4]. This paper reviews a decade of IMC research, covering core principles, key memory technologies (SRAM, RRAM, PCM), optimization strategies, and AI applications. By analyzing current designs and performance metrics to reveal key strengths, limitations, and design trade-offs, it offers practical guidance for future research and development of energy-efficient AI hardware.

2. Basic principles and technology classification

2.1. Fundamental concept of in-memory computing

The core idea of IMC is to repurpose memory arrays as computational units, thereby eliminating the need to move data to separate arithmetic logic. Instead, operations like multiply-and-accumulate (MAC) are performed directly through the intrinsic electrical behavior of memory cells and their interconnects. As illustrated in Figure 1, the memristive crossbar array serves as a canonical example of this approach, where each crosspoint device is programmed with a conductance value representing a stored weight [4]. Input voltages applied to the rows induce currents along the columns, which accumulate according to Kirchhoff's current law and reflect the weighted sum of the inputs. This process effectively performs matrix–vector multiplication (MVM) in a single, parallel computational step. Importantly, the crossbar architecture is not limited to memristive devices. For instance, standard CMOS SRAM arrays have also been adapted for IMC purposes by designing specialized bit cells capable of performing local logic or multiplication operations on stored data [5].

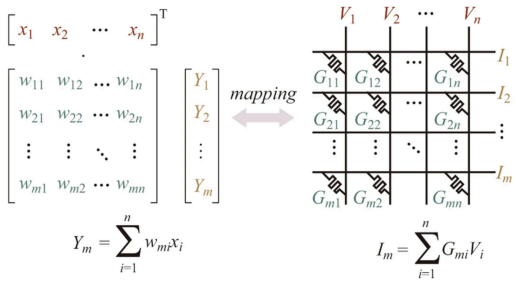


Figure 1. Crossbar-based architecture for in-memory matrix-vector multiplication

2.2. Major technologies for in-memory computing

IMC leverages a range of memory technologies, with static random-access memory (SRAM), resistive RAM (RRAM), and phase-change memory (PCM) being the most commonly adopted. Each presents distinct trade-offs in precision, integration complexity, energy efficiency, and scalability, as shown in Table 1. SRAM-based IMC operates in the digital domain by adapting standard cells for bitwise or analog-like functions. It features fast access and CMOS compatibility, making it suitable for digital inference, but its large cell size limits density and precision [5]. In crossbar-based analog computing, RRAM-based IMC offers key advantages by storing multilevel weights via resistive switching. High integration density is achieved due to its compact footprint, and efficient, highly parallel matrix-vector multiplication is supported at low power. Nonetheless, challenges such as device variability, limited endurance, and conductance drift remain significant

DOI: 10.54254/2753-8818/2025.AD27977

barriers to practical deployment [2,4]. PCM-based IMC stores data as resistance states in chalcogenide materials. It supports multibit storage and exhibits good endurance, thus making it promising for analog computing and vertical 3D integration. Compared to RRAM, PCM generally provides better stability, but it suffers from higher write energy and slower switching speeds [3,6].

Ideally, SRAM is chosen for small inference models that require low latency, while RRAM and PCM are more fitted for bigger models with high-density, with their precision-efficiency balance carefully evaluated for each specific task.

Table 1. SRAM, RRAM and PCM characteristics

Technology	Cell Type	Volatility / States	Analog Support	Pros	Cons
SRAM	CMOS 6T/8T	Volatile, 1-bit	Limited (binary)	Fast, mature, no new process	Large area, low precision, needs ADC/DAC
RRAM	MIM structure	Nonvolatile, multibit	Yes (multilevel)	High density, low leakage, analog MAC	Variability, limited endurance, needs calibration
PCM	Chalcogenide-based	Nonvolatile, multibit	Yes (multilevel)	Multilevel storage, 3D stacking	Drift, higher write energy, needs compensation

3. The design and optimization of in-memory computing circuit

3.1. Overall design process and key implementation steps

The design of an IMC accelerator requires tight integration between algorithm and hardware. A typical design workflow involves memory technology selection, weight mapping and quantization, crossbar and peripheral circuit design, control logic development, algorithm-hardware co-optimization, as well as system-level validation [2,4]. The process begins with selecting the memory device and determining array size, where characteristics like RRAM conductance nonlinearity, PCM drift behavior, or SRAM bitline capacitance are modeled and incorporated into circuit-level simulations [3,6]. The weight matrix is then quantized according to device precision and mapped onto the physical array, often partitioned across tiles or cores [2,4]. Peripheral circuits, such as digital-to-analog converters (DACs) for input and analog-to-digital converters (ADCs) for output, are designed to interface with the memory arrays. The control logic handles signal timing, pulse width modulation, and read/write scheduling, and ensures synchronized operation across compute tiles, ensuring low-latency and reliable execution [3,6]. Due to the analog imperfections of IMC devices, retraining or fine-tuning neural networks with device-aware models is a crucial part of algorithm-hardware co-optimization. Techniques such as noise injection or differential encoding are commonly used to improve robustness. For example, in the NeuRRAM chip, inference accuracy across non-ideal cores was maintained through per-array calibration combined with chip-in-the-loop progressive fine-tuning [2,3]. To maintain accuracy under high analog throughput, modern IMC designs integrate calibration and retraining, and are evaluated under realistic workloads for performance, efficiency, and reliability.

3.2. Performance optimization targets and techniques

IMC design involves balancing four key metrics: speed, energy, area, and precision, and these metrics often constrain one another, and optimization strategies must be chosen based on specific application demands. The following outlines how each target is addressed in current IMC

DOI: 10.54254/2753-8818/2025.AD27977

architectures. Speed is mainly influenced by throughput and latency. Throughput is improved by leveraging the parallelism of crossbar arrays and implementing pipelined data flows. Latency can be reduced by shrinking array sizes, though this increases tiling overhead. Additional acceleration techniques include time encoding like time-to-digital conversion, fast analog sensing, parallel row activation, and input reuse [2]. Energy efficiency is critical, particularly in edge computing. Strategies include voltage scaling and reducing DAC/ADC resolution to lower switching energy. Low-leakage memory technologies such as RRAM and PCM further reduce power consumption. Analog computation helps eliminate clock-related energy loss but increases sensitivity to noise. Dynamic power gating and event-driven control are also used to limit idle power [4]. Area is largely dictated by peripheral components like ADCs and control logic. Reduction techniques include shared data converters, analog multiplexing, and array tiling. Nonvolatile memories such as RRAM and PCM enable higher integration density than SRAM due to their smaller bitcells. However, excessive area optimization may limit system flexibility and degrade performance [3]. Precision is challenged by non-ideal device behavior, limited resolution, and signal degradation in analog domains. To address these challenges, designers use calibration methods like reference tuning and offset compensation, along with mixed-precision computing and noise-aware training [3]. Higher DAC and ADC resolution boosts accuracy but increases area and power consumption.

3.3. Reliability issues and fault-tolerant design

IMC circuits face significant reliability challenges due to the physical limitations of emerging memory technologies and the tight coupling of storage and computation. Key issues include device variability, such as cell-to-cell mismatch and cycle-to-cycle fluctuations, resistance drift in nonvolatile memories like PCM and RRAM, and parasitic effects such as line resistance and IR drop in large crossbar arrays [2,3,6]. These factors impair signal integrity, increase errors, and reduce system stability over time. To mitigate these challenges, multiple fault-tolerance techniques have been developed. One common method involves refresh and calibration, especially in analog IMC systems based on RRAM or PCM. Periodic weight reprogramming and readout calibration are used to restore accuracy as conductance drifts over time or with use. Reference tuning and offset cancellation further enhance sensing reliability [3,6]. Redundancy and reconfiguration provide another layer of protection. Faulty or unstable cells are bypassed through critical-path duplication or spare row/column rerouting. Programmable crossbar structures support selective deactivation of unreliable units, hence maintaining system functionality with minimal performance overhead [4]. In digital and mixed-signal IMC systems, error detection and correction (ECC) schemes mitigate bit-level faults arising from noise and device variability. Traditional ECC approaches, including parity and Hamming codes, are adapted to IMC contexts, particularly in analog-to-digital conversion and bitplane decoding. In addition, approximate schemes have been proposed to recover partial precision in compressed representations [4]. At the algorithmic level, robust training methods improve system resilience to hardware-level variability and noise. Methods like noise injection, weight clipping, and quantization-aware retraining enhance model resilience, while some designs employ “chip-in-the-loop” training using hardware feedback to address analog imperfections [2,3]. By integrating circuit-level, architectural, and algorithmic approaches, practical IMC systems achieve reliable performance under real-world conditions.

DOI: 10.54254/2753-8818/2025.AD27977

4. The applications and performance of in-memory computing in AI acceleration

IMC accelerates Deep Neural Network (DNN) inference by performing matrix-vector multiplications in memory, reducing data movement and power consumption. And it suits energy-limited devices and supports Convolutional Neural Network (CNN) and Transformers, while delivering significantly higher energy efficiency and maintaining accuracy comparable to digital accelerators.

4.1. Performance gains and system-level challenges

The compute-intensive nature of DNN inference, dominated by MAC operations, makes it well-suited to IMC acceleration. By integrating storage and computation, IMC significantly increases throughput and reduces energy consumption, especially when implemented using parallel analog crossbar arrays. A key performance metric for IMC is energy efficiency, typically measured in MACs per joule or tera-operations per second per watt (TOPS/W). For instance, the NeuRRAM chip delivers up to twice the energy efficiency of earlier RRAM-based designs, reaching over 1 TOPS/W in inference workloads [2]. Similarly, IBM's HERMES, a hybrid analog-digital architecture, achieves more than 10 TOPS/W while maintaining high accuracy on CNN-based vision tasks [7]. These figures significantly exceed those of conventional digital accelerators like GPUs, which typically achieve 0.1-1 TOPS/W depending on workload and precision [7,8]. Simulations on large-scale transformer models further suggest that 3D analog IMC could provide 10×-100× energy savings compared to high-end datacenter hardware [9]. Despite reduced numerical precision, analog IMC maintains competitive accuracy in many real-world tasks. NeuRRAM achieves inference accuracy within 1% of floating-point software baselines on MNIST and CIFAR-10 [2]. Hybrid approaches like HERMES combine analog MAC units with digital correction logic to achieve 8-bit effective accuracy in classification tasks [7]. Nevertheless, limited precision still confines IMC to inference tasks, thus making it unsuitable for training or applications requiring full floating-point range and numerical stability. Despite its efficiency advantages for edge and mobile deployment, IMC also faces system-level challenges. Among these, I/O bottlenecks caused by data movement into and out of compute arrays can significantly impact energy consumption and latency in large-scale systems. Additional constraints include limited on-chip memory for intermediate data, lack of support for non-MAC operations like activation functions, and restricted programmability. These factors shift bottlenecks to memory access and interconnects, prompting research into dataflow and hybrid architectures [6,8].

4.2. Training challenges and emerging solutions

Though IMC performs well in DNN inference, it remains ill-suited for on-chip training. Existing commercial and prototype IMC chips primarily target inference, where weights are programmed once and reused across input batches. In contrast, training requires frequent weight updates with higher numerical precision, broader dynamic range, and stronger tolerance to device noise and drift. One major barrier is the mismatch between learning algorithm requirements and IMC device characteristics. Algorithms like stochastic gradient descent are sensitive to quantization noise and hardware variability. However, many IMC-compatible memories like RRAM and PCM often exhibit nonlinearity, drift, and limited write endurance, and these limitations become more pronounced during frequent updates, making stable training convergence challenging [2,3]. Training poses an additional challenge due to its higher precision requirements. In contrast to inference, which often

DOI: 10.54254/2753-8818/2025.AD27977

tolerates low-resolution weights and activations, training requires more fine-grained updates, particularly in the early layers of neural networks. Analog IMC circuits struggle to deliver multi-bit precision consistently, especially under low-voltage or noisy conditions, which further complicates gradient-based learning. In response, some architectures support simplified or approximate training by introducing algorithm-device co-design. For example, the NeuRRAM chip features a programmable interface that allows in-situ weight updates for small models [2]. Other implementations combine analog accumulation with periodic digital correction, enabling coarse-grained gradient descent to mitigate analog inaccuracies [3]. In addition, chip-in-the-loop training has been explored, using real-time hardware feedback to adjust model parameters in response to non-ideal circuit behavior [2,6]. Besides utilizing algorithm-device co-design, recent methods adopt hybrid architectures that delegate precision-critical tasks, such as gradient computation and weight updates, to digital units or off-chip processors. This strategy is enabled by advances in device modeling, adaptive learning, and low-power write circuits [3,9].

4.3. Architectural evolution of IMC hardware

IMC hardware has taken various architectural forms, such as analog, mixed-signal, digital-centric, and 3D-stacked implementations, each tailored to balance performance, energy efficiency, and scalability for different AI workload demands. Each direction reflects distinct trade-offs between computational precision, efficiency, and system complexity. Analog IMC architectures represent the earliest and most energy-efficient implementations. For example, the NeuRRAM chip combines 48 RRAM-based CIM cores with 12-bit ADCs to directly support CNN and RNN inference in hardware. NeuRRAM achieves near-software accuracy on benchmarks like MNIST and CIFAR-10, while achieving twice the energy efficiency of earlier RRAM-based compute-in-memory designs [2]. Mixed-signal approaches seek to balance the compute density of analog hardware with the precision and reliability of digital correction. For example, IBM's HERMES, built on 14 nm CMOS, integrates phase-change memory, time-domain ADCs, and digital post-processing. It achieves over 10 TOPS/W and competitive inference accuracy for vision tasks, exhibiting how charge-domain MAC units and calibration circuits help address analog variability [7]. Digital-centric IMC designs integrate compute and memory at the architectural level. For example, IBM's NorthPole, built with 12 nm technology, includes 224 MB of SRAM and over 22 billion transistors, enabling inference without external memory access. It achieves 25 times higher energy efficiency and 22 times lower latency on ResNet 50 than conventional GPUs [10]. Emerging 3D-stacked IMC systems offer new scalability. In a recent IBM design for transformer-scale models, eight RRAM layers are stacked vertically, with each layer mapped to an expert in a Mixture-of-Experts architecture. This setup enhances vertical bandwidth and parallelism, enabling IMC adaptation for large language models [9].

5. Conclusion

This study reviewed leading IMC architectures using SRAM, RRAM, and PCM, emphasizing their strengths in energy-efficient and highly parallel AI inference. In response to device-level challenges, hardware-aware training and mixed-precision design effectively enhance computational accuracy. In addition, prototype chips validate the potential for high-performance inference at ultra-low energy cost. However, current IMC research remains largely focused on inference, as on-chip training is still limited by device variability and precision constraints. At the system level, large-scale deployment is hindered by challenges such as I/O bottlenecks, high ADC power consumption, and

DOI: 10.54254/2753-8818/2025.AD27977

the limited scalability of crossbar arrays. It should be noted that this review does not address emerging technologies like FeRAM or optical IMC, which provides new possible IMC pathways with their unique features such as high read/write speed and low energy loss, but still in early lab phase with their adaptabilities yet to be discovered, nor does it explore broader application domains such as neuromorphic computing or commercial deployment scenarios. Future research should focus on algorithm-hardware co-design to enhance training robustness and precision, advance 3D integration and near-memory computing for greater scalability, and develop standardized toolchains to reduce deployment barriers and facilitate the transition of IMC toward general-purpose AI acceleration.

References

[1] IBM Research. (2024). In-memory computing. https: //research.ibm.com/projects/in-memory-computing
[2] Wan, W., et al. (2022) A compute-in-memory chip based on resistive random-access memory. Nature, 608(7923), 504-512.
[3] Gallo, M.L. et al. (2018) Mixed-precision in-memory computing. Nature Electronics, 1(4), 246-253.
[4] Bao, H., et al. (2022) Toward memristive in-memory computing: principles and applications. Frontiers of Optoelectronics, 15(1).
[5] Yoshioka, K., Ando, S., Miyagi, S., Chen, Y.C. and Zhang, W. (2024) A review of SRAM-based compute-in-memory circuits. Japanese Journal of Applied Physics, 63, 120802.
[6] Gallo, M.L. and Sebastian, A. (2020) An overview of phase-change memory device physics. Journal of Physics D Applied Physics, 53(21), 213002.
[7] Pool, R. (2023) Could IBM's AI chip reinvent deep learning inference? EE Times Europe. https: //www.eetimes.eu/could-ibms-ai-chip-reinvent-deep-learning-inference/
[8] Modha, D.S., et al. (2023) Neural inference at the frontier of energy, space, and time. Science, 382(6668), 329-335.
[9] Büchel, J., et al. (2025) Efficient scaling of large language models with mixture of experts and 3D analog in-memory computing, Nature Computational Science.
[10] IBM Research. (2025) How analog in-memory computing could power the AI models of tomorrow. https: //research.ibm.com/blog/how-can-analog-in-memory-computing-power-transformer-models