From Macro to System: Quantifying SoC Overheads in a 22nm 165 TOPS/W Innovative Digital CIM Accelerator

A. El Arrassi*, L. Huijbregts*, G. Gaydadjiev*, M. Taouil*, R. Joshi†, and S. Hamdioui*
*Delft University of Technology, Netherlands †IBM T.J. Watson Research Center, USA

***Abstract*—While many digital Compute-in-Memory (CIM) accelerators report exceptional energy efficiency, these figures are often overly optimistic as they frequently omit the non-trivial overhead of system-level and SoC integration. This paper presents a 128x128-bit digital CIM core fabricated in 22nm, integrated into a complete SoC via a custom programmable wrapper and a dedicated memory interface. The proposed CIM architecture features three primary innovations: 1) the elimination of area-intensive adder trees, 2) the implementation of a pipelined accumulation periphery, and 3) an inherent in-read multiplication mechanism. Prototyped at 4-bit precision and a 0.6V supply, the standalone CIM core achieves a peak energy efficiency of 165 TOPS/W and a compute density of 25 TOPS/mm^2. In contrast, our system-level measurements reveal an SoC efficiency of 120 TOPS/W, a 27% reduction compared to the macro-level performance. These results quantify the impact of system overhead and underscore the necessity of end-to-end SoC evaluation for a transparent assessment of CIM as a viable alternative computing paradigm.**

***Index Terms*—Computation-in-Memory, SRAM, digital, MAC.**

I. Introduction

While Compute-in-Memory (CIM) accelerators offer a promising alternative for energy-constrained edge AI applications, existing solutions often prioritize CIM core performance while neglecting critical system-level bottlenecks [1]–[3]. Fig. 1(a) illustrates examples of such challenges including data movement cost (e.g., streaming the input data to SoC/AI cores), shared memory (on chip memory) cost, control and wrapper cost, on-chip interconnect cost, etc. At the core level, digital CIM typically relies on adder-tree-based accumulation; the disruption of the dense memory structure imposes high energy and area consumption [1], as shown in Fig. 1(b).

This paper presents an innovative 165 TOPS/W digital Compute-in-Memory (CIM) accelerator in 22nm CMOS and evaluates its performance at both the macro and SoC levels. At the macro level, the architecture implements a novel Multiply-Accumulate (MAC) scheme designed to replace area-intensive adder trees with a hierarchical, pipelined accumulation periphery, maximizing both energy and area efficiency [4]. The design utilizes 8T-SRAM cells to enable inherent bitwise multiplication during memory read operations. The core is organized into eight 16×128 sub-arrays featuring flying Read Bit-Lines (RBLs) to enhance throughput. At the system level, the core is integrated via a custom programmable wrapper and an adaptive-bandwidth on-chip shared memory (global buffer).

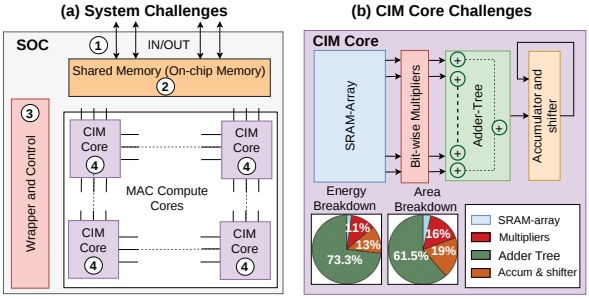

Fig. 1: System and Core-level challenges of CIM.

The wrapper provides flexible control over CIM operations, while the global buffer interface optimizes operand retrieval (inputs/weights) and the management of MAC results (outputs/partial sums).

Prototyped in 22nm GlobalFoundries technology, the core-level measurements report a peak energy efficiency of 165 TOPS/W, and a compute density of 25 TOPS/mm² at 4-bit precision and 0.6V. At the SoC level, the system achieves a peak energy efficiency of 120 TOPS/W. This 27% efficiency delta highlights the critical importance of accounting for SoC-level overhead in providing a transparent assessment of digital CIM as a viable paradigm for edge AI.

II. System Architecture and Implementation

The top-level architecture of the proposed accelerator is illustrated in Fig.2. To facilitate high-throughput edge processing while minimizing data movement energy, the design is partitioned into three specialized subsystems: the MAC CIM core, a programmable wrapper for control, and a shared memory module; these subsystems are generated using the SNAX interface [5] and will be explained next.

A. CIM Core

Fig. 3(a) shows an overview of the CIM core. It utilizes a split-array topology consisting of two mirrored SRAM arrays; each is further divided into four 16 $rows \times$ 128 $columns$ sub-arrays to increase parallelism. One of the key innovations in this design is the implementation of a "flying bitline" architecture shown in Fig. 3(b), where Read Bit Lines (RBLs) are routed vertically across memory instances. This routing strategy allows the sub-arrays to connect directly to a centralized accumulation block, effectively bypassing the need for

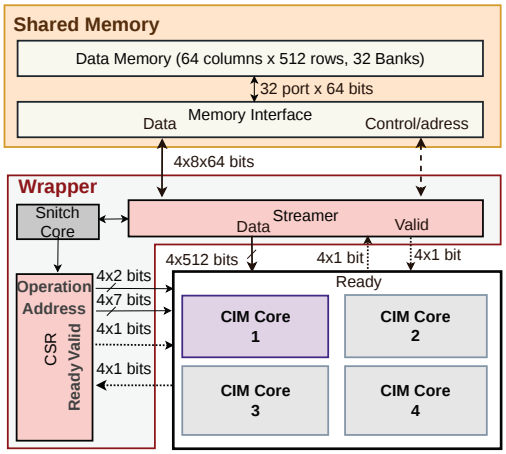

Fig. 2: Block diagram of the SoC/accelerator

traditional, energy-heavy adder trees. As detailed in Fig. 3(c), the accumulation block follows a specific hierarchical flow. First, BWM-Registers store the bit-wise multiplication (BWM) partial results between (bit-serial) inputs and stored weights. Second, these partial results are processed by the Column Accumulator (CA), which performs parallel additions of eight BWM results per column. Third, in order to support 4-bit precision, four CAs are aligned and summed by a Word Accumulator (WA) to generate partial sums. Finally, the Bit-serial and Row Accumulator performs sequential accumulation using adders and registers to produce the final MAC results.

The proposed digital CIM design increases memory read parallelism by a factor of eight. The flying bitline approach ensures the design's scalability to advanced technology nodes. Since a standard 8T-SRAM cell layout is designed to accommodate the three baseline bitlines (BL, BLB, and RBL), it inherently provides sufficient routing tracks for the three additional "flying" bitlines on the upper metal layers without increasing the cell area. The strategic integration of pipeline stages within the accumulation block enhances the operating frequency, leading to improved latency and energy efficiency for MAC. By utilizing flying RBLs and a centralized accumulation block, the design avoids the structural disruptions typically present in adder-tree implementations, resulting in higher area efficiency and simplified routing and placement.

B. Wrapper

Fig.2 shows the Wrapper for 4 CIM cores. Wrapper is a central control block designed to manage the coordination between the CIM cores and the Shared Memory, and it incorporates three key components to enable high-performance processing:

Snitch Core [6]: An energy-efficient, lightweight RISC-V processor that acts as the system orchestrator. It manages CIM Cores by reading from and writing to the Control and Status Register (CSR) and coordinates the Streamer by issuing data-transfer instructions, ensuring synchronized operation between memory access and computation.

Programmable CSR: This interface facilitates communication with CIM Cores, allowing the Snitch core to issue specific instructions, such as initiating MAC operations, while receiving *ready* and *valid* status signals from CIM cores. This provides the real-time programmability required to handle varying workload sizes and requirements.

Streamer: A dedicated Direct Memory Access (DMA)-like engine responsible for controlling data access to the CIM core. By autonomously managing 512-bit input/output data streams per core, the streamer decouples memory accesses from active computation, effectively hiding data movement latency.

C. Shared Memory (Global Buffer)

The third key subsystem is Shared Memory, shown in Fig. 2, which manages data storage and accessibility for the entire accelerator. This subsystem includes a shared Data Memory SRAM block and a flexible Memory Interface network. The former is organized into 32 banks, each 64 bits wide, serving as the primary data repository for inputs, weights (needed to program the core before inference), and temporary MAC results; while the latter manages data traffic between Data memory and CIM Cores through the wrapper, supporting both standard 64-bit traffic control and high-bandwidth 512-bit vector transfer per core.

III. Measurements

A. Setup and Experiments Performed

We prototyped the SoC shown in Fig. 2, albeit with a single core only (to maintain design simplicity and ensure an accurate, hardware-verified breakdown). The core was designed for 4-bit MAC inputs and weights. The prototype was fabricated in 22nm GlobalFoundries technology, and a custom-designed PCB was developed to facilitate measurements. Fig. 4 shows a picture of the bare die, and that of the packaged die mounted on a PCB with the measurement setup. Notably, the SoC is designed to enable standalone execution to isolate and characterize the energy consumption of individual sub-blocks; this is achieved by incorporating independent power-gating and control signals.

To perform the evaluation under realistic conditions, we performed the following three experiments. The experimental parameters are summarized in Tab. I & II.

1) CIM core measurements: this is done by using synthetic patterns of 50% sparsity (motivated by YOLOv6-S workload average sparsity), sweep supply voltage (0.5V to 0.9V), and determine peak energy-efficiency TOPS/W. While the CIM core is capable of running a clock frequency of 1 GHz, these characterization measurements were performed at a nominal operating fre-

TABLE I: Setup Parameters

Parameter	Value
Technology	22 nm
Supply voltage	0.55-0.9 V
SRAM cell	8-T
CIM array size	16 kb

TABLE II: App. Overview

Application	Object Det.
Topology	YOLOv6-S
Datasets	COCO
Num of param	41K

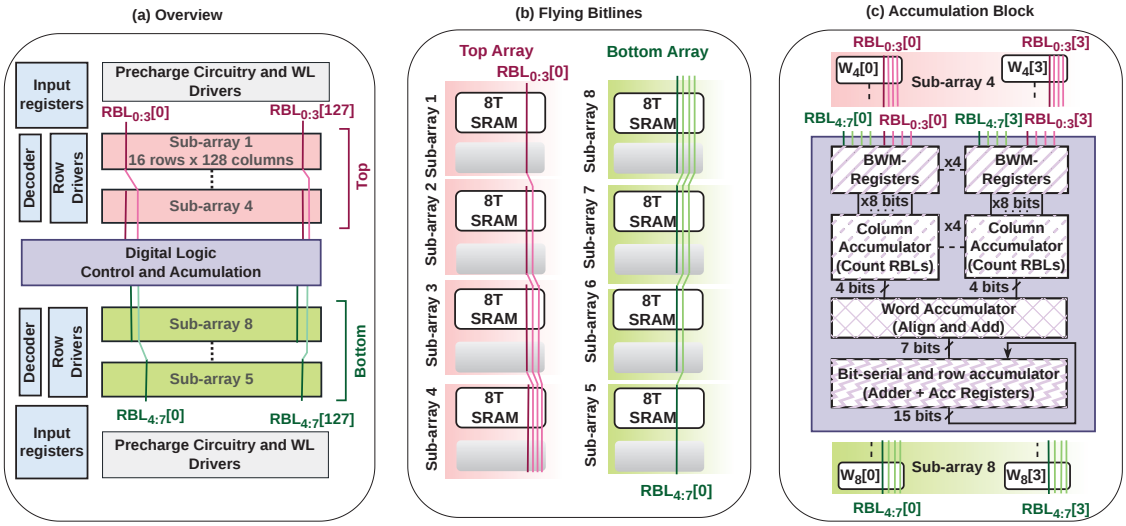

Fig. 3: Architecture of the Proposed CIM Core.

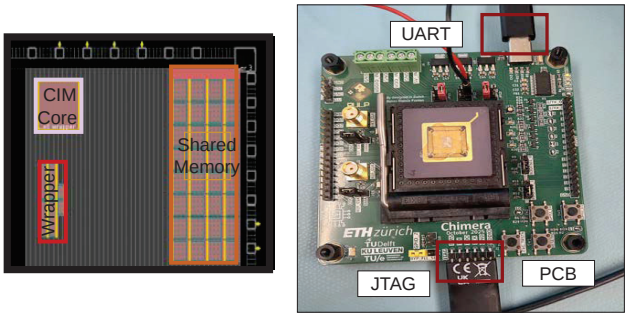

Fig. 4: Chip Picture and Measurement Set Up.

quency of 100 MHz to ensure functional stability and reliable data acquisition.

2) SoC Measurements: Same as for CIM core measurements, but now by considering the entire SoC (i.e., CIM Core, programmable Wrapper, and Shared Memory).
3) System evaluation using YOLOv6-S: The system was validated using a YOLOv6-S object detection workload, focusing exclusively on MAC operations executed sequentially on SoC. This experiment incorporates the full overhead of data communication and I/O offloading for each operation. By assuming sequential processing rather than parallel execution or pipelining, this evaluation provides a conservative, pessimistic baseline for real-world system-level performance.

B. Macro-Level Measurements (CIM Core)

Fig. 5(a) shows the breakdown of the CIM core area and energy consumption. The physical implementation of the CIM core requires a total area of 0.02mm^2, and the breakdown reveals a balanced distribution: the SRAM arrays occupy 46.6%, while the accumulation block accounts for 53.4% (within which the combinational arithmetic units are dominant at 71%). This core design clearly solves the challenge highlighted in Fig. 1(b), where the area of a digital CIM core is typically dominated by adder-tree-based accumulation.

The average measured energy is 132pJ, calculated by monitoring the current across numerous CIM MAC operations. The energy breakdown (see Fig. 5(a)) reveals that the SRAM array consumes 40%, while the accumulation block accounts for 60%; within the latter, the arithmetic logic consumes more energy than the registers. These results demonstrate that the proposed CIM architecture successfully addresses the challenge illustrated in Fig. 1(b), where traditional adder-tree-based designs typically consume over 90% of total energy.

The derived peak energy efficiency (based on the measured power, frequency and number of MAC operations) is 165 TOPS/W and the derived computing density is 25 TOPS/mm^2

C. SoC Measurements

Fig. 5(b) and (c) illustrate the area and energy breakdown for the other two subsystems of the SoC: Shared Memory and Programmable Wrapper (see Fig. 2). Physically, these occupy 0.1 mm^2 and 0.01 mm^2, respectively. Within Shared Memory, data storage arrays consume the vast majority of the area. Conversely, the Snitch control core dominates the Wrapper's footprint. The measured total SoC energy is 355 pJ, and an energy efficiency of 120 TOPS/W, implying a system-level overhead (Shared Memory and Wrapper) of 223 pJ (355 pJ − 132 pJ for the CIM core). The specific breakdown shown in Fig. 5(b) and (c) is derived from the above measurements and the relative percentages extracted from the post-layout simulations.

In summary, the CIM core accounts for only 37.1% of the total SoC energy, with the rest is consumed by Shared Memory and Wrapper. This discrepancy demonstrates that reporting energy efficiency based solely on a standalone core is insufficient. An end-to-end co-design approach is essential for realistic evaluation and deployment of practical edge AI hardware.

TABLE III: Comparison with State-of-the-Art CIM Accelerators (* denotes projected values)

Metric	Macro-Level Comparison					SoC-Level Comparison	
	ISSCC'22 [1]	ISSCC'23 [2]	SSC-L'25 [3]	This Work		ISSCC'23 [7]	This Work
Tech. (nm)	5	4	5	22	7*	18	22
Inp. Sparsity (%)	12.5	12.5	12.5	50	50	50	50
Voltage (V)	0.5	0.32	0.48	0.6	0.5	0.52	0.6
Precision (In/W)	4/4	8/8	4/4	4/4	4/4	4/4	4/4
Eng. Eff. (TOPS/W)	254	87.4	100	165	396*	77	120
Array size (Kb)	64	54	128	16	16	2K	16
Area (mm^2)	0.0133	0.0172	0.022	0.02	0.004*	4.2	0.13
Area Eff. (TOPS/mm^2)	55.3	13.7	56 (0.9V)	25	160*	13.6	3.9

D. Workload Evaluation: YOLOv6-S

To evaluate the proposed design's AI potential, we profiled a YOLOv6-S workload on the COCO dataset. Fig. 6(a) shows the stage-wise breakdown, identifying the Backbone as the most intensive phase (18.7 mJ, 3.36 s). Fig. 6(b) identifies specific bottlenecks via layer-level analysis. For instance, Layer 8 (Stage 5) exhibits high computational density, consuming 3.3 mJ in just 0.59 s, whereas Layer 6 (Stage 4) shows higher latency (1.77 s) for less energy.

Based on these results, we derive a total energy of 39 mJ per inference. In a production scenario where the SoC integrates multiple CIM cores, this latency and energy-per-inference can be significantly reduced, potentially reaching the sub-mJ range. This represents a significant advancement over the current state of the art in edge AI acceleration.

E. Comparison with State-of-the-Art

Table III benchmarks the proposed design against recent reported digital CIM accelerators, categorized by macro and SoC-level efficiencies. At the macro level, our 22nm core achieves 165 TOPS/W and 25 TOPS/mm². To ensure a fair comparison with advanced-node prototypes, we projected these results to 7nm using scaling laws from [8]. The resulting 396 TOPS/W and 160 TOPS/mm² demonstrate that our adder-tree-free architecture outperforms prior work solutions.

At the SoC level, integrated system metrics are rarely reported. Compared to the most relevant literature [7], our design achieves 120 TOPS/W, approximately a 2× efficiency improvement. While our energy efficiency is superior, our compute density is currently lower. This is primarily attributed to a non-optimized floorplan, as shown in Fig. 4, leaving significant room for area optimization in future iterations.

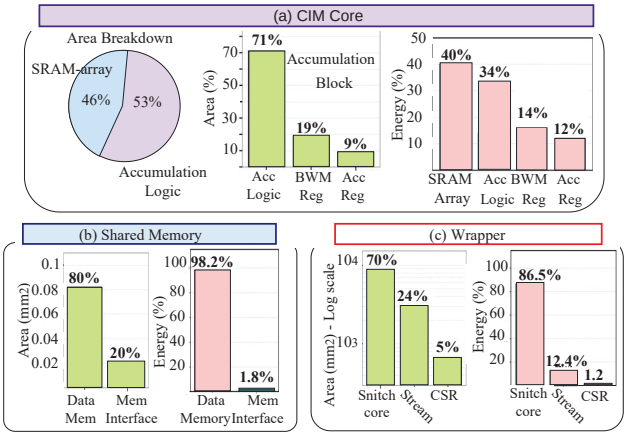

Fig. 5: Area and Energy Breakdowns for the SoC blocks.

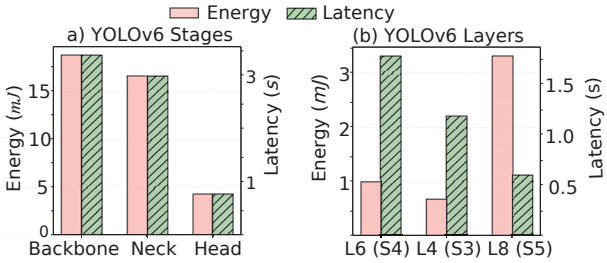

Fig. 6: System Energy Breakdown for YOLOv6-S.

IV. Conclusion

While our CIM core achieves exceptional standalone efficiency, the reduction to 120 TOPS/W at the SoC level underscores the critical impact of peripheral and memory overhead. These findings suggest that future research must shift from isolated macro optimization toward co-designing the integration wrappers and data-movement hierarchies.

Acknowledgment

This work is funded by the Convolve project evaluated by the EU Horizon Europe research and innovation program under grant agreement No. 101070374.

References

[1] H. Fujiwara *et al.*, "A 5-nm 254-tops/w 221-tops/mm 2 fully-digital computing-in-memory macro supporting wide-range dynamic-voltage-frequency scaling and simultaneous mac and write operations," in *ISSCC*, 2022.

[2] H. Mori *et al.*, "A 4nm 6163-tops/w/b 4790-tops/mm2/b sram based digital-computing-in-memory macro supporting bit-width flexibility and simultaneous mac and weight update," in *ISSCC*, 2023.

[3] S.-J. Yun *et al.*, "5-nm high-efficiency and high-density digital sram in-memory-computing macros for ai accelerators," *IEEE Solid-State Circuits Letters*, vol. 8, pp. 269–272, 2025.

[4] A. E. Arrassi *et al.*, "Sram-based computation-in-memory device," PCT Application Patent PCT/EP2026/061 235, Apr. 17, 2026, pending.

[5] X. Yi *et al.*, "Opengemm: A highly-efficient gemm accelerator generator with lightweight risc-v control and tight memory coupling," in *Proceedings of ASP-DAC*, 2025.

[6] F. Zaruba *et al.*, "Snitch: A tiny pseudo dual-issue processor for area and energy efficient execution of floating-point intensive workloads," *IEEE Transactions on Computers*, vol. 70, no. 11, pp. 1845–1860, 2020.

[7] G. Desoli *et al.*, "16.7 a 40-310tops/w sram-based all-digital up to 4b in-memory computing multi-tiled nn accelerator in fd-soi 18nm for deep-learning edge applications," in *2023 ISSCC*. IEEE, 2023, pp. 260–262.

[8] S.-Y. Wu *et al.*, "A 7nm cmos platform technology featuring 4 th generation finfet transistors with a 0.027 um 2 high density 6-t sram cell for mobile soc applications," in *2016 IEEE International Electron Devices Meeting (IEDM)*. IEEE, 2016, pp. 2–6.