MNEMOSENE++: Scalable Multi-Tile Design with Enhanced Buffering and VGSOT-MRAM based Compute-in-Memory Crossbar Array

Carlos Escuin*†, Fernando García-Redondo‡, Mahdi Zahedi§, Pablo Ibáñez†, Teresa Monreal¶, Víctor Viñals†, José María Llabería¶, James Myers‡, Julien Ryckaert*, Dwaipayan Biswas*, and Francky Catthoor*
*IMEC, Leuven, Belgium, ‡IMEC, Cambridge, UK
†Dept. Informática e Ingeniería de Sistemas - I3A, Universidad de Zaragoza, Zaragoza, Spain
§Department of Quantum and Computer Engineering, Delft University of Technology, Delft, The Netherlands
¶Universitat Politècnica de Catalunya, Barcelona, Spain

***Abstract*—This paper optimizes the MNEMOSENE architecture, a compute-in-memory (CiM) tile design integrating computation and storage for increased efficiency. We identify and address bottlenecks in the Row Data (RD) buffer that cause losses in performance. Our proposed approach includes mitigating these buffering bottlenecks and extending MNEMOSENE's single-tile design to a multi-tile configuration for improved parallel processing. The proposal is validated through comprehensive analyses exploring the mapping of diverse neural networks evaluated on CiM crossbar arrays based on NVM technologies. These proposed enhancements lead up to 55% reduction in execution time compared to the original single-tile architecture for any general matrix multiplication (GEMM) operation. Our evaluation shows that while ReRAM and PCM offer notable energy advantages, their integration with scaled CMOS is limited, which leads to VGSOT-MRAM emerging as a promising alternative due to its good balance between energy efficiency and superior integration capabilities. The VGSOT-MRAM crossbar arrays provide 12×, 49×, and 346× more energy efficiency than PCM, ReRAM, and STT-MRAM ones, respectively. It translates, on average for the considered workload, in 1.5×, 3×, and 14.5× better energy efficiency of the entire system.**

***Index Terms*—Compute in Memory, NVM, Memristor, MRAM, Convolutional Neural Networks, Machine Learning.**

I. Introduction

In the pursuit of improving the energy efficiency and computational prowess of future generations of computers, the focus has steadily shifted towards compute-in-memory paradigms. Contrasting the traditional von Neumann architecture that segregates computing and memory units, in-memory computing seamlessly integrates these units to yield significant energy savings. A vast body of research has already been undertaken on the design of memory arrays and their peripheral circuitry, with various emerging NVM technologies, such as resistive RAM (ReRAM), phase-change memory (PCM), spin-transfer torque magnetic RAM (STT-MRAM) or voltage gate assisted spin-orbit torque magnetic RAM (VGSOT-MRAM) under exploration [1], [5], [8], [11].

One of the distinctive architectures in this domain is the compute-in-memory-periphery (CiM-P) tile [9], [10], where storage and computation are performed in the analog memory array, with the resultant data delivered in the digital periphery. This approach not only promotes energy efficiency but also presents a more streamlined operational model.

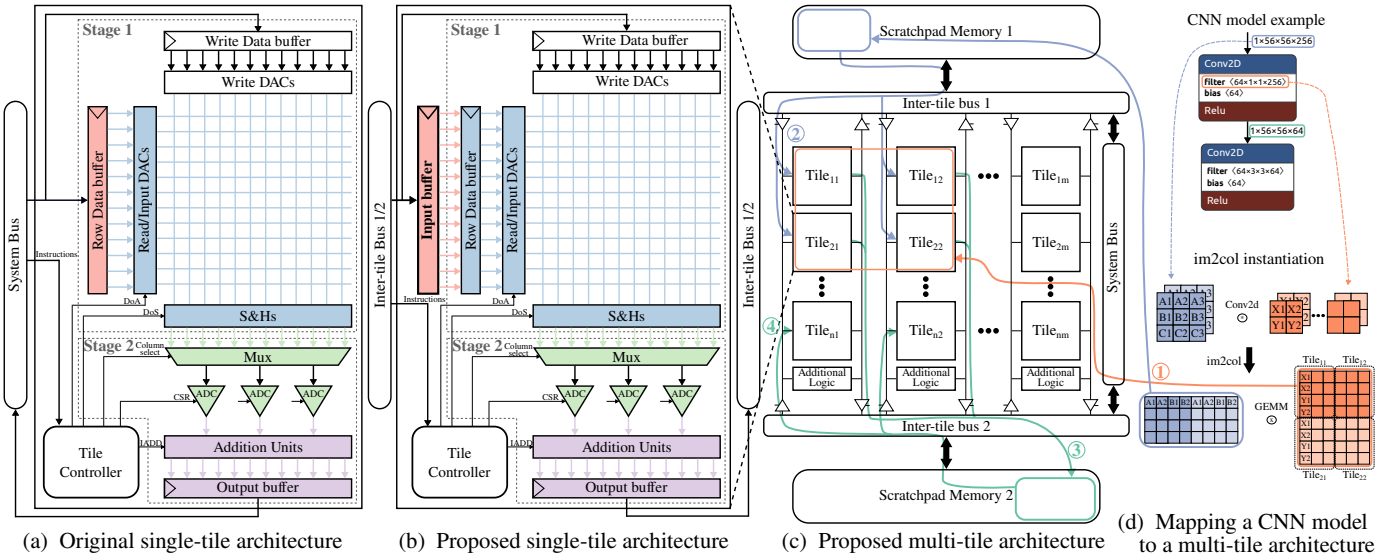
The MNEMOSENE CiM architecture is an epitome of this approach, featuring a NVM array and peripheral circuitry with a defined instruction-set architecture (ISA) for bridging higher-level programming languages to the underlying circuit designs [9], [10]. Nevertheless, as cutting-edge as the MNEMOSENE architecture is, there remains room for improvement. This paper focuses on the shortcomings of the current design, particularly concerning the memory transactions to internal buffers constrains and single-tile architecture limitations.

This paper's key contribution is threefold: We first propose to revisit the current internal buffering in the MNEMOSENE tile architecture to address the bottleneck issues caused by expensive system memory requests. To this end, we suggest implementing a double input buffer to decouple the data load to the tile from the analog computation, alleviating such bottleneck issues. This adjustment aims to reduce performance overheads and enhance energy efficiency.

Secondly, we introduce enhancements to the MNEMOSENE design enabling a multi-tile architecture. These enhancements involve designing a shared scratchpad memory and an efficient interconnection framework. By optimally sizing both the scratchpad and the interconnection we can further mitigate unnecessary delays, ensure efficient data retrieval, and enable seamless synchronization across multiple tiles; thus augmenting its computational capabilities.

Finally, to validate the proposal, we perform a comprehensive analysis of both small and large convolutional neural networks (CNNs), ranging from TinyML and AnalogNet to more extensive networks like Nasnet and Resnet. The performance of these CNNs under the enhanced MNEMOSENE architecture provide valuable insights into the potential of our proposed modifications. Finally, we perform an extensive evaluation using in-house MRAM technologies, such as STT-MRAM and VGSOT-MRAM, and state-of-the-art PCM and ReRAM. This evaluation offers a broader perspective on the implications of our proposed enhancements. The comprehensive

 https://dx.doi.org/10.1109/ICECS58634.2023.10382874



(a) Original single-tile architecture (b) Proposed single-tile architecture (c) Proposed multi-tile architecture (d) Mapping a CNN model to a multi-tile architecture

Fig. 1: CiM intra-tile and multi-tile organization. (a) MNEMOSENE original architecture, (b) double-buffer proposal, (c) proposed multi-tile organization, and (d) mapping a CNN model to the multi-tile architecture. We highlight how after a CNN is transformed to be computed as a GEMM using *im2col*, the different filters in a layer are distributed into a group of tiles. Independent layers access separated areas on the two scratchpad memories, pipelining the read/writes from/to the buffered data.

analysis underscores the effectiveness of faster, low-power NVM technologies such as VGSOT-MRAM for energy-efficient CiM computations. The suitability of VGSOT-MRAM stems from its ability to utilize reduced currents during inference time, a characteristic attributed to its higher resistances.

The subsequent sections of this paper delve into the specifics of the proposed extensions to the MNEMOSENE architecture, followed by the analysis of neural network performance and the comprehensive evaluation of the improved architecture under different NVM technologies.

II. Enhancing the MNEMOSENE Tile Architecture

Many studies in CiM have focused on developing standalone accelerators tailored to address specific computational tasks. General matrix multiplications (GEMMs) are the dominant operations in today's Machine Learning workloads. From standard CNN to large transformers networks, the most common layers are accelerated by its conversion to GEMM operations [11], [12].

However, the number of CiM architectures designed to seamlessly integrate with general-purpose multiprocessor systems remains relatively limited. One notable project that exemplifies this integration is MNEMOSENE. In their work, Zahedi et al. present a programmable single-tile architecture alongside an ISA and a compiler that aims to establish a standardized simulation framework for CiM designs [9], [10]. This framework facilitates the creation of a flexible interface between the CiM tile and the broader system, enabling effortless interaction and integration within a general-purpose multiprocessor environment. The defined ISA lies at the core of this integration and aims to enhance the flexibility and generality of the hardware by shifting the complexity towards the compiler.

A. Original single-tile architecture

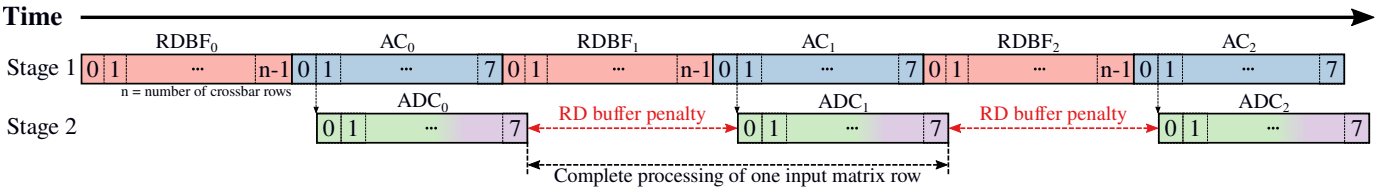
We now show the interactions between the main components of a single CiM-tile architecture. More details on how the different instructions work and what their circuit specifications are can be found in [9], [10].

Figure 1a illustrates the original MNEMOSENE single-tile architecture able to run a GEMM. One matrix is stored in the memristive crossbar array through the *write data (WD) buffer* while the other one is transferred row by row to the *row data (RD) buffer* before the analog computation can take place.

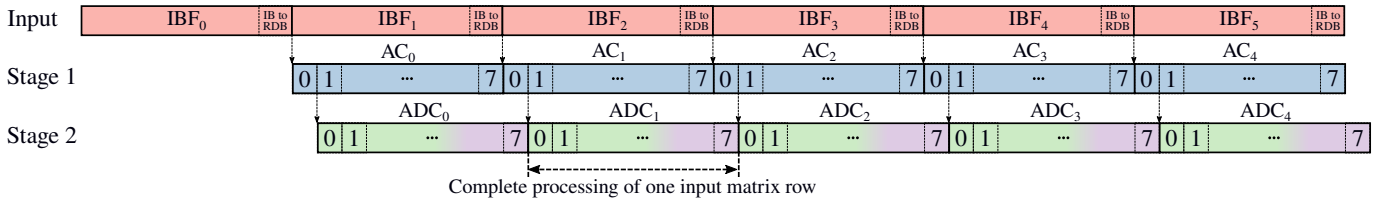
A row computation is pipelined in two stages. Beyond these stages, the *tile controller* plays a vital role in receiving nano-instructions, decoding them into various control signals, and efficiently scheduling their execution across each stage. In Stage 1, the RD buffer serves as the recipient for the data elements, which act as inputs for each crossbar row. Then, the incoming elements are shifted one bit at a time into the DACs to perform the analog computation in the memristive crossbar array. The crossbar outputs are then collected from each column by the *Sample-and-Hold (S&H)* unit. In Stage 2, multiple columns share a single ADC. To handle this scenario, the partial results from each column are sampled by the ADC in a time-multiplexed manner. Following this, the *addition units* combine these partial results to generate the final outcomes, which are then stored in the *output buffer*.

B. Overcoming RD buffer limitations

Unfortunately, bringing the data to the RD buffer imposes significant overhead in the original single-tile architecture. The tile is required to wait for the system to refill the RD buffer for each subsequent matrix row since the RD buffer is written byte to byte. Consequently, the RD buffer may inadvertently cause delays in standard workloads, thereby hindering the overall efficacy of the compute-in-memory operations.



(a) Original MNEMOSENE single-tile architecture.



(b) Decoupling the data load to the tile from the analog computation.

Fig. 2: Timing diagram of multiplying the first matrix row in both (a) the original single-tile and (b) the proposed optimization. Colors match the components of Figure 1. RDBF stands for RD buffer fill, AC for analog computation, ADC for ADC conversion and addition units, and IBF for IB fill. The penalty imposed by the RD buffer is highlighted in red in (a).

Figure 2a illustrates the timing diagram of computing the first row of a GEMM. The timing depicts the breakdown of the execution time among the different components. In the original configuration, the RD buffer has one entry per crossbar row (n rows in the figure) and every entry is 8 bits wide.

Filling the RD buffer consumes one cycle per buffer entry, while the full execution of the 8-bit element requires a delay that is dominated by the ADC conversion. On a permanent basis, transferring the following row to the RD buffer imposes a significant latency overhead that could be overlapped with the analog computation (AC) of the previous row.

Therefore, the goal is to hide the RD buffer latency penalty by overlapping it with the analog computation. To this end, we propose to add an *input buffer (IB)* of the same size of the RD buffer, see Figure 1b, to decouple the data load to the tile from Stage 1. Thus, while the analog computation of the first row is taking place, the second row is being fetched into the IB in the background, see AC_0 and IBF_1 in Figure 2b. While the analog computation of the current row, AC_i, is finishing, the next row, already loaded in the IB, is copied to the RD buffer. Assuming the IB can be completely filled during the analog computation of the previous element, the tile pipeline execution is now dominated by Stage 2. In Section III-B we will further investigate how the rest of the multi-tile components are sized and parameterized so that the IB is seamlessly integrated and can be completely filled in the background to avoid any additional overheads.

Our experiments showed that the double input buffer approach leads up to 55% reduction in execution time for a single GEMM compared to the original MNEMOSENE tile architecture.

III. Enabling A Multi-Tile Architecture

A. Multi-tile μArchitecture

A multi-tile architecture is indispensable in CiM operations due to inherent scalability and performance constraints associated with single-tile configurations. As computational tasks grow in complexity, the capacity of a single tile to effectively address these requirements diminishes. Transitioning to a multi-tile structure provides the system with the capability to distribute computational tasks across multiple tiles, facilitating parallel processing and augmenting overall performance.

Figure 1c depicts the proposed multi-tile architecture. It consists of numerous tiles, logically placed in a two-dimensional fashion, two inter-tile buses and two scratchpad memories. Each bus is connected to a scratchpad memory, allowing decoupling the read and write flows of the tiles. The scratchpad memory serves both as intermediary with the outer system and to enable tile-to-tile synchronization.

Figure 1d shows how a CNN model is mapped into such a multi-tile architecture. The convolutions, which are the pivotal computational task in CNNs, are turned into GEMM operations by means of the image to column (im2col) transformation [12]. The ever-growing dimensions of these matrices hinder them to fit into a single tile. Weight matrices are thereby broken down into chunks and distributed across multiple rows and/or columns of tiles ①. Input matrices are also split up and forwarded to the tiles matching the corresponding weight chunks. While the analog computation takes place, a consistent data stream flows from one scratchpad memory to the IB of the tiles ②, and reciprocally, from the tile output buffer back to the other scratchpad memory ③. In this way, the partial results from multiple tiles are aggregated in the *additional logic* conforming the complete result of a GEMM. It is essential to note that the output matrix of one layer may become the input matrix of the subsequent layer ④. Hence, certain layers read from the top-positioned scratchpad and write to the one below, while the adjacent layers operate vice versa.

B. Properly sizing scratchpads and interconnections

To overcome the limitations outlined in Section II-B, it is crucial to optimally size the scratchpad memory and the buses, preventing additional overheads during the population of the tiles' IBs. The architectural components delineated in the

TABLE I: NVM technologies specification.

	ReRAM	PCM	STT-RAM	VGSOT
Low Resistance State	5 kΩ	20 kΩ	6.2 kΩ	824.1 kΩ
High Resistance State	1 MΩ	10 MΩ	15 kΩ	2.1 MΩ
Memory Read Voltage	0.2 V	0.2 V	0.5 V	0.55 V
Memory Read Time	10 ns	10 ns	10 ns	3 ns

previous sections are predominantly dependent on the workload. However, the flexibility of the MNEMOSENE architecture enables comprehensive parameterization, allowing for optimal customization for the task at hand. Experiments were conducted to size the proposed components appropriately, validating the design's feasibility.

To accurately size the scratchpad memory, focus is placed on identifying the CNN layer with the largest memory footprint. This evaluation encompasses not only the dimensions of the input and output matrices, but also takes into account the input data of concurrent residual connections within the CNN. On the other hand, to avoid additional communication overheads between the scratchpad and the tiles, it is essential to ensure that the inter-tile buses can transmit any matrix row from the scratchpad to the tiles within the time frame in which the analog computation (AC_i) of the previous row takes place, as depicted in Figure 2b. Therefore, the bus width must be properly sized to accommodate the transmission of the matrix row of maximum size within this time frame; note that excessive IBF_i times in Figure 2b would lead to undesired delays. Section IV-A shows the optimal sizing values we obtained for both scratchpad memory and buses and for the evaluated workloads.

IV. Evaluation

A. Experimental Setup

Experiments utilized the MNEMOSENE simulator [9], [10] with a 256x256 analog crossbar array per tile, and 8-bit datatype size. All NVM technologies, ReRAM [3], PCM [6], STT-MRAM [2], and VGSOT-MRAM, are configured in dual-state memory cells; their most important figures are provided in Table I. In the present experiments, the deployment of a weight involves one column per weight bit. Our in-house STT-MRAM devices are accompanied by a projected VGSOT-MRAM device simulation that, making use of a higher voltage-controlled magnetic anisotropy (VCMA) coefficient, relaxes the critical write current and enables 4-pillar bitcells. DACs, S&Hs, and ADCs specifications, as well as a 1GHz operating frequency, matched the original MNEMOSENE papers [9], [10]. Input and RD buffers were modeled with Nandgate 15nm technology [7].

The workload comprised two small AnalogNets CNN models (keyword spotting -KWS-, and visual wake words -VWW-) [11], and two Tensorflow CNN models (Nasnet [13], Resnet50 [4]). Notably, Nasnet presented the layer with the largest memory footprint, reaching 7.3 MB. To accommodate memory requirements, 2 scratchpad memories of 4 MB each were designed, consisting of eight 512 KB banks each, modeled using an in-house STT-MRAM data memory featuring 22nm technology node, considering latency and energy for comprehensive CNN inference evaluation. For efficient data transfer between tiles and the scratchpad, a 48-byte bus width sufficed.

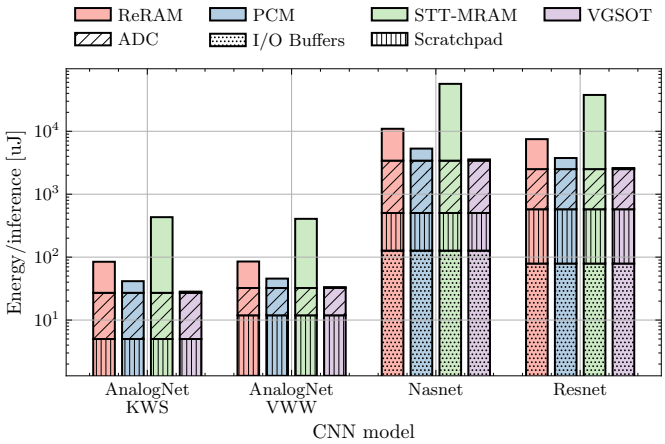


Fig. 3: Energy/inference (in log scale) breakdown of the different architectural components for each of the CNN models, evaluating distinct NVM technologies.

B. Experimental Results

Table II summarizes the results after mapping one inference of the different CNN models to our multi-tile architecture. Looking from a workload perspective, and regarding the memory footprint of the models, the small AnalogNet models use at most 405 KB of the scratchpad memory, 5% of its total size. Larger models thereby have larger memory requirements, with 2.8 and 7.3 MB for Resnet50 and Nasnet, respectively. One complete inference is performed and the latency and energy results are reported. The tile execution is dominated by Stage 2, see Figure 2b, so all implementations of the same CNN, regardless of the NVM technology, report the same expected latency. While small models complete one inference within the range of 1 ms, the larger ones take up to 39.5 ms to finish it.

Regarding energy, and due to the different conductances of each NVM technology, the crossbar arrays report varying energy consumption figures for the same CNN model. In particular, for our in-house NVM technologies, STT-MRAM provides ∼7× and ∼28× higher energy consumption than ReRAM and PCM, respectively. Despite these higher power consumption, the great advantage of STT-MRAM lies in its high integration capabilities with scaled nodes [2]. As a successful tradeoff, VGSOT-MRAM combining reduced area and lower energy requirements, becomes the most promising technology compared to other NVMs. Based on our in-house VGSOT-MRAM devices projection, the VGSOT-MRAM crossbar arrays are 12×, 49×, and 346× more energy efficient than PCM, ReRAM, and STT-MRAM ones, respectively. It translates, on average for the considered CNN models, in 1.5×, 3×, and 14.5× better energy efficiency of the entire system.

Figure 3 reports the energy per inference breakdown including the independent architecture components for the different CNN models and technologies. As can be seen, the energy numbers differ significantly between CNN models. For instance, for PCM and ReRAM, smaller analog CiM specific CNN models operate in the range of 40 to 400 μJ, while standard larger models (NasNet and Resnet50) consume 2 to 60 mJ per

TABLE II: Analyzed CNN models, required number of tiles, computational utilization, memory footprint, latency and energy.

CNN model	Model size (#Parameters)	Required tiles	Computational utilization (%)	Maximum footprint	One inference latency	One inference total energy			
						ReRAM	PCM	STT-RAM	VGSOT
KWS	Small	46 (7×7)	79.8	136.5 KB	185.8 μs	84.3 μJ	41.4 μJ	431.5 μJ	28.2 μJ
VWW	Small	75 (9×9)	57.6	405.0 KB	1.3 ms	85.3 μJ	45.6 μJ	406.1 μJ	33.4 μJ
Nasnet	Medium (5.3M)	6364 (80×80)	**8.3**	7.3 MB	39.5 ms	11.0 mJ	5.3 mJ	56.7 mJ	3.6 mJ
Resnet50	Large (25.6M)	2966 (55×55)	**97.6**	2.8 MB	10.9 ms	7.5 mJ	3.8 mJ	37.8 mJ	2.6 mJ

inference. Our research highlights that AnalogNet workloads, optimized for analog CiM accelerators, demonstrated a better utilization of tile resources versus peripheral areas and buffers. Therefore we can highlight how optimizing its deployment on analog hardware via Network-Architecture-Search (NAS) not only improved NN accuracy [11], but also significantly boosted energy efficiency, underscoring the value of NAS for these specific accelerators.

The varying energy efficiency across different technologies reveals a significant contrast in the energy consumption of periphery and scratchpad components. On the one hand, the energy consumed by these assisting components is minimal compared to the overall energy for some technologies, see ADC + I/O Buffers + Scratchpad energy in Figure 3. For instance, for STT-MRAM, the energy of the assisting components accounts for at most 7.9% of the overall one for VWW. On the other hand, for VGSOT-MRAM, these components emerge as the primary energy factor, up to 96.8% of the overall one for VWW. Further improving the energy efficiency of such low power technologies will involve special efforts to improve the peripheral components as well.

Depthwise-Convolution Analog vs Digital: During this research and in line with findings from previous studies [11], limitations with depthwise convolutions were encountered. Such bottlenecks are critically notable when NasNet was deployed on the multi-tile architecture. Compared to Resnet50, which has $5\times$ as many parameters (see Table II), Nasnet requires more than twice as many tiles to be deployed, but reduces significantly the computation utilization of the arrays (8.3% in NasNet versus 97.6% in Resnet50). Out of the scope of this work, but highlighted by it, our results underscore a crucial need to adapt the NAS algorithm to the specific requirements and constraints of analog tile-based systems [11].

V. Discussion and Conclusions

In this study, we have effectively addressed the primary bottlenecks in the MNEMOSENE architecture. By enhancing the original framework, we developed a robust multi-tile architecture capable of improved parallel processing and optimized buffering. Moreover, we successfully refined the communication between the tiles and the scratchpads, providing efficient data handling and significant performance enhancement. The architectural enhancements served to ensure efficient computation and storage integration, thereby propelling the MNEMOSENE design forward.

In addition, a thorough evaluation involving a wide range of neural networks and four different NVM technologies was conducted. These evaluations revealed a marked discrepancy in energy consumption across the different technologies. Amidst these, VGSOT-MRAM emerged as a promising compromise. It not only offers reduced area and lower energy requirements, but also presents a high degree of integration and speed, thus standing superior to STT-MRAM, and to both ReRAM and PCM. This finding accentuates VGSOT-MRAM's potential to effectively bridge the gap between computational efficiency and energy conservation in future technological advancements.

Acknowledgments

The authors would like to thank Dawit Abdi, Yang Xiang, Kaiming Cai, Mohit Gupta, and Marie Garcia Bardon for their helpful discussions. The authors would like to thank Chuteng Zhou for his help and examination of AnalogNet workloads. This project has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876925, from MCIN/AEI/10.13039/501100011033 (grants PID2019-105660RB-C21 and PID2019-107255GB-C22), and from Aragón Government (T58_23R research group).

References

[1] T. Andrulis *et al.*, "Raella: Reforming the arithmetic for efficient, low-resolution, and low-loss analog pim: No retraining required!" in *The 50th International Symposium on Computer Architecture (ISCA)*, 2023.

[2] F. García-Redondo *et al.*, "Stt-mram stochastic and defects-aware dtco for last level cache at advanced process nodes," in *IEEE 53st European Solid-State Device Research Conference (ESSDERC)*, 2023.

[3] A. Hardtdegen *et al.*, "Improved switching stability and the effect of an internal series resistor in hfo 2/tio x bilayer reram cells," *IEEE Transactions on Electron Devices*, 2018.

[4] K. He *et al.*, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[5] R. Khaddam-Aljameh *et al.*, "Hermes core – a 14nm cmos and pcm-based in-memory compute core using an array of 300ps/lsb linearized cco-based adcs and local digital processing," in *Symposium on VLSI Circuits*, 2021.

[6] M. Le Gallo *et al.*, "Compressed sensing with approximate message passing using in-memory computing," *IEEE Transactions on Electron Devices*, 2018.

[7] M. Martins *et al.*, "Open cell library in 15nm freepdk technology," in *the International Symposium on Physical Design (ISPD)*, 2015.

[8] A. Singh *et al.*, "Cim-based robust logic accelerator using 28 nm stt-mram characterization chip tape-out," in *IEEE 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2022.

[9] M. Zahedi *et al.*, "Tile architecture and hardware implementation for computation-in-memory," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2021.

[10] M. Zahedi, M. A. Lebdeh *et al.*, "MNEMOSENE: Tile Architecture and Simulator for Memristor-based Computation-in-memory," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 2022.

[11] C. Zhou *et al.*, "Ml-hw co-design of noise-robust tinyml models and always-on analog compute-in-memory edge accelerator," *IEEE Micro*, 2022.

[12] Y. Zhou *et al.*, "Characterizing and demystifying the implicit convolution algorithm on commercial matrix-multiplication accelerators," in *IEEE International Symposium on Workload Characterization (IISWC)*, 2021.

[13] B. Zoph *et al.*, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.