ELectronics EXpress

LETTER

An Energy-Efficient Digital Computing-In-Memory STT-MRAM Macro for AdderNets with Optimized Addition Operations

Yuan Xue[1, 2, 3, a)], **Sinan Zou**[1, 2, 3, b)], **Jianfeng Gao**[1, 2, c)], **Yilu Li**[1, 2, d)], **Yan Cui***[1, 2, e)], **and Jun Luo***[1, 2, 3, f)]

Abstract Adder neural networks (AdderNets), a promising lightweight alternative to traditional CNNs, face the "memory wall" challenge in von-Neumann architectures. Computing-in-memory (CIM) has emerged as a promising solution to address this memory bottleneck. This work proposes a novel spin-transfer torque magnetic random-access memory based digital-CIM macro tailored for AdderNets mapping, leveraging Boolean logic-optimized architecture to seamlessly integrate storage and computation. The architecture is validated through simulations under 40 nm CMOS technology. Results show that the architecture achieves an energy efficiency of 31.48 TOPS/W in 8-bit network inference, representing a 1.5× to 4.1× improvement over state-of-the-art digital CIM designs.

Keywords: Computing-in-memory (CIM), digital computing, magnetic random-access memory (MRAM), adder neural network
Classification: Integrated circuits (logic)

1. Introduction

Convolutional neural networks (CNNs) represent a classic paradigm of generative artificial networks and are widely applied across various domains, including artificial intelligence and the Internet of Things (IoT). Learning algorithms based on multiply-accumulate (MAC) operations have dominated and driven the rapid development of high-performance computing [1–3]. Currently, most mainstream neural networks are deployed on computational systems based on the von-Neumann architecture. These platforms process and store intermediate data and weight information generated during network calculation. The intensive data transfer between processors and storage units results in significant latency and energy costs [4], leading to a performance bottleneck commonly known as the "memory wall" [5]. This severely limits the integration of neural networks in resource-constrained devices, which is critical for embedded and edge applications [6, 7].

Computing-in-memory (CIM) has emerged as a promising approach to mitigate this challenge. It directly deploys partial computational tasks to memory cells, physically eliminating the boundary between processors and memory units [8, 9]. As the architectural cornerstone of CIM systems, the intrinsic properties of memory cells dictate system performance [10]. Neuromorphic implementations exploit their programmable conductance to emulate synaptic functions and enable logic-in-memory operations [11]. Spin-transfer torque (STT) and spin-orbit torque (SOT) based magnetic random-access memory (MRAM) utilizes crossbar arrays for parallel computation, demonstrating superior energy efficiency and performance over conventional CMOS-based in-memory computing (IMC) designs [12–16].

Adder neural networks (AdderNets), a novel type of CNNs, replaces multiplication operations with addition in convolution operations to reduce computational complexity [17]. AdderNets utilize the ℓ_1-distance instead of the L2 norm to measure feature-filter discrepancies, demonstrating competitive performance in tasks such as computer vision [18], object detection [19], and super-resolution [20]. A hardware-algorithm co-design methodology has facilitated successful AdderNet implementations on multiple digital CIM platforms, including SRAM [21], FPGA [22], and MRAM [23]. However, these digital IMC schemes inevitably rely on dedicated auxiliary processing units (e.g., adder trees or multi-stage latches) for intermediate data storage and ℓ_1 -distance computation. Such architectures inherently limit advancements in computational parallelism and efficiency.

In this paper, we present a novel STT-MRAM-based digital CIM macro designed for high-performance and energy-efficient inference in AdderNets. Our macro integrates comprehensive basic Boolean logic operations tailored for the accelerator architecture of AdderNets, along with an efficient ℓ_1-distance computation method. The main contributions of this paper can be summarized as follows: (1) We propose an efficient IMC paradigm tailored for low bit-width AdderNet training, which replaces the convolution process with the ℓ_1-distance of vectors to reduce computational cost. (2) We introduce an innovative method for co-optimizing architecture and circuits based on Boolean logic, achieving seamless integration of storage functions and logical computation operations within the MRAM array. (3) We validate the proposed CIM architecture through simulations under SMIC 40 nm CMOS technology. Results demonstrate that our additive logic implementation achieves over 3.6× energy consumption optimization. Additionally, in 8-bit net-

[1] Key Laboratory of Fabrication Technologies for Integrated Circuits, Chinese Academy of Sciences, Beijing 100029
[2] Institute of Microelectronics, Chinese Academy of Sciences (IMECAS), Beijing 100029, China
[3] University of Chinese Academy of Sciences (UCAS), Beijing 100049, China
[a)] xueyuan@ime.ac.cn
[b)] zousinan@ime.ac.cn
[c)] gaojianfegn@ime.ac.cn
[d)] liyilu@ime.ac.cn
[e)] cuiyan@ime.ac.cn
[f)] luojun@ime.ac.cn

work inference, the architecture delivers a maximum energy efficiency of 31.48 TOPS/W.

2. Computing in STT-MRAM fundamentals

2.1 2T1MTJ Bit-cell

A typical STT-MRAM cell consists of magnetic tunnel junctions (MTJs) and one or more access transistors [24]. The core of the MTJ is a sandwich structure formed by two ferromagnetic metal layers separated by a barrier layer, commonly made of MgO. The two ferromagnetic layers are distinguished as the reference layer (RL) and the free layer (FL) based on the difficulty of magnetization flipping. Depending on the magnetization orientations of the RL and FL, the MTJ can exhibit high-resistance (AP) or low-resistance (P) states [25]. Accordingly, we define the AP and P states of the MTJ as representing logical ‘1’ and ‘0’, respectively.

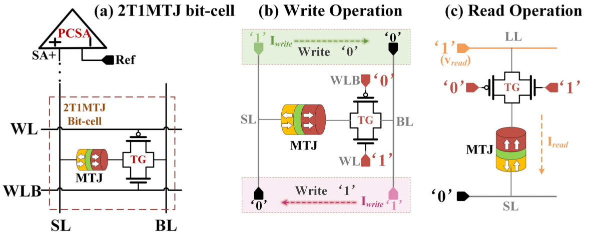

Fig. 1 (a) Structure of the Proposed 2T1MTJ Bitcell with (b) Write and (c) Read operations.

We use a 2T-1MTJ bit-cell structure that enhances storage density while also handling logic computation tasks. Fig. 1 illustrates the actual configuration: a single MTJ is accessed and interconnected through one transmission gate (TG). During write operations, the TG is activated by the word lines (WLs) to generate the write current Iwrite, with the direction and magnitude of the current determined by the voltage difference applied between the source line (SL) and the bit line (BL). For read operations, the target cell will connecting the cell to the readout circuit, which is mainly composed of a pre-charge current sense amplifier (PCSA).

Table I Key parameters of the perpendicular-magnetic-anisotropy MTJ.

Parameter	Description	Value
Area	MTJ surface (W x L)	32 nm x 32 nm
tox	Thickness of the Oxide Barrier	0.85 nm
tsl	Thickness of the Free Layer	1.3 nm
TMR	TMR (0) with Zero Volt Bias Voltage	200%
T_0	Ambient Temperature	300 K
I_{c0}	Critical Switching Current	34.1 µA
R_{MTJ}	Nominal Resistance at R_L (R_H) of MTJ	6.3 $K\Omega$ (18.9 $K\Omega$)

A Verilog-A compact model [26] describes the perpendicular-magnetic-anisotropy MTJs (pMTJs) device. Its critical parameters are shown in Table I, which are based on theoretical calculation and experimental measurements [27]. For the access transistors, a commercial 40 nm CMOS technology was considered.

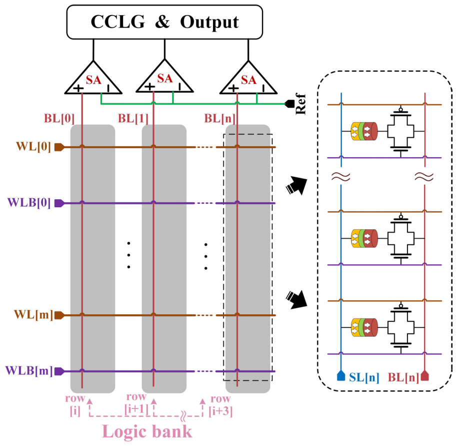

Fig. 2 Proposed memory array architecture for enhanced logic computations.

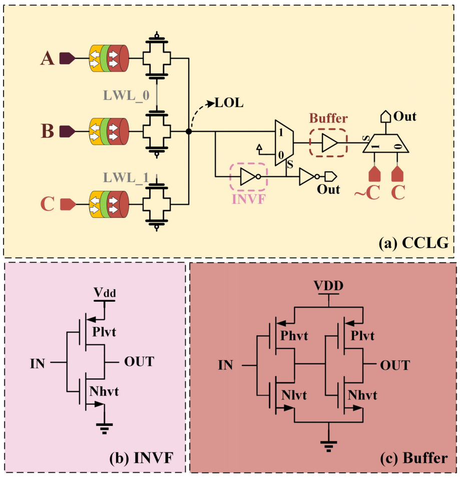

Fig. 3 (a) Overview of the proposed configurable composite logic gate (CCLG) and its components: (b) INVF and (c) Buffer.

2.2 Motivation for in-memory logic based on readout circuit

Fig. 2 illustrates the proposed 2T1M bit-cell, which implements inter-row Boolean logic through a sense amplifier (SA) and a configurable composite logic gate (CCLG). The NVM array is divided into logic banks, each containing three columns connected to the CCLG via the SA. The first three columns form a majority gate for basic Boolean logic, while the fourth column serves as a redundancy line for data storage or composite logic.

As shown in Fig. 3(a), external control signals in the CCLG enable switching between different logic operations. Table II summarizes the signal configurations and corresponding logic functions. For basic Boolean and MAJ operations, ports A, B, and C are connected via TG, while port D is isolated. Port C is set to ‘0’ for AND/NAND, ‘1’ for OR/NOR, or MAJ/MNJ(A, B, C). For XOR/XNOR, ports C and D are isolated, and the SA reads operands A and B, triggering a voltage change at the logic output line (LOL). This change is captured by a feedback structure (INV and MUXs) and amplified for output. Fig. 3(b) and Fig. 3(c) detail the feedback circuit, which utilizes multi-threshold voltage transistors to fine-tune the logic output margin.

Table II Configuring logic gates for diverse logical operations

Signals	Input	Description	Operations

LWL_1=1
LWL_2=1
	A	Input	AND/NAND

	B	Input	
	C	’0’	
	A	Input	OR/NOR
	B	Input	
	C	’1’	
	A	Input	MAJ/MNJ
	B	Input	
	C	Input	

LWL_1=1
LWL_2=0
	A	Input	XOR/XNOR

	B	Input	
	C	X	

2.3 Full Adder for CIM

In the realm of CIM, the Full Adder (FA) serves as a crucial hardware unit for performing addition and multiplication operations. It is typically realized using specific CMOS combinational logic circuits or digital/analog NVM units combined with sequential logic scheduling [13, 16, 28]. To balance computational efficiency and throughput, we propose an innovative high-parallel full addition solution based on CCLG, as illustrated in Fig. 4.

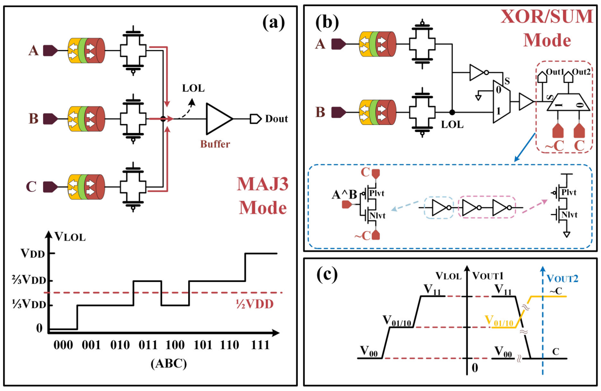

Fig. 4 (a) Resistive-based MAJ3 gate. (b) Proposed feedback-selective XOR/SUM logic gate circuit structure, and (c) Its node voltage distribution.

Generally, the input of the FA consists of two computational input bits, A and B, along with a carry-in bit (C_i). The FA generates and outputs the sum (S) and the carry-out (C_{i+1}) based on the logical relationships described in Eq. 1:

$$\begin{aligned} C_{i+1} &= AB + (A \oplus B)\, C_i = AB + AC_i + BC_i \\ S &= A \oplus B \oplus C_i \end{aligned} \tag{1}$$

It is important to note that Eq. 1 can be equivalently expressed as Eq. 2:

$$C_{out} = M_3\,(A, B, C_{in}) \tag{2}$$

Therefore, we configure the CCLG to operate in MAJ3 logic mode, as shown in Fig. 3(a). Using Eq. 2, the carry value C_i for each bit of input data A and B is iteratively generated. Additionally, we introduce a decision mechanism as follows:

$$\begin{aligned} S &= (A \oplus B) \oplus C \\ &= SEL(\bar{C}, C)|_{A \oplus B} \end{aligned} \tag{3}$$

We configure the CCLG to operate in XOR logic mode, and its topology is shown in Fig. 4(b). At this stage, the voltages at the LOL node and Out1 vary according to the level signals at ports A and B. Based on the logic of Eq. 3, the final sum value is computed at Out2. The voltage waveforms at each node are shown in Fig. 4(c).

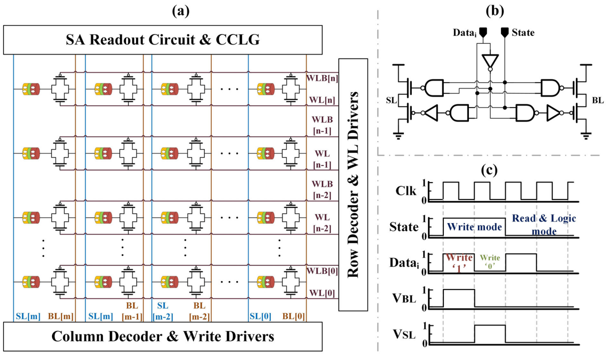

Fig. 5 Overall architecture of the proposed CIM macro. (a) Circuit-level interconnection design of the NVM array. (b) Write driver used for multi-functional switching and (c) Its waveforms in different function mode.

3. Hardware implementation strategies for CIM framework

AdderNets introduce a novel approach by replacing traditional multiplicative convolutions with additive convolutions. This innovation significantly reduces computational energy while maintaining high inference accuracy. Such advancements enhance the feasibility of deploying neural networks on edge devices and hardware platforms [29].

$$Y = -\sum_{i=0}^{d}\sum_{j=0}^{d}\sum_{k=0}^{C_{in}} |X\,(m+i, n+j, k) - W\,(i, j, k, t)| \tag{4}$$

The core concept of AdderNets is to use the ℓ_1-distance metric to measure the similarity between the input feature X and the filters W, as shown in Eq. 4. This approach eliminates the need for complex multiplications typically required for traditional Euclidean distance calculations, significantly reducing computational complexity and energy consumption.

3.1 CIM macro description

To demonstrate the hardware implementation of the proposed additive convolution, we employ an $m \times n$ MRAM array. As shown in Fig.5(a), the array comprises $m \times n$ cells, each with a 2T1MTJ structure. Fig. 5(b) and (c) illustrates the write driver design and its waveforms in different operational modes. This architecture enables independent and parallel writing to each column’s computing unit via dedicated write drivers, while read operations and logical processing are performed in parallel across rows.

3.2 Full adder design

To execute the additive convolution mapping, it is essential to establish a FA circuit. Fig. 6 illustrates the construction process of a single-bit adder. *Step*1: We initialize all MTJ units to the state ’0’. Load operands A and B, and the carry-in signal C_i, from external registers into their corresponding MTJ cells. *Step*2: Adjust the control signals to place the

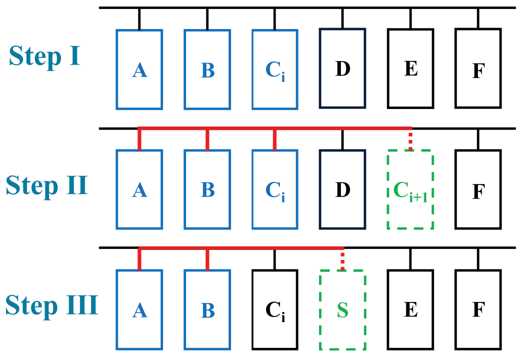

Fig. 6 Illustration of the array-level implementation scheme for a single-bit full adder.

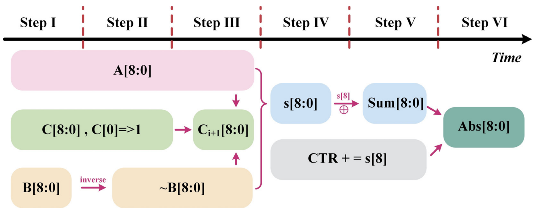

Fig. 7 Illustration of the step-by-step operations for implementing an 8-bit metric function S based on the proposed architecture.

array in read mode and configure the CCLG to enter MAJ logic state. Perform MAJ logic on the first three cols of MTJ cells storing operands A, B, and C_i. The carry-out signal Ci+1 is calculated according to Eq. 2. $Step3$: Reconfigure the RCLG to enter XOR logic state. Execute XOR logic on the top two cols of MTJ cells storing operands A and B, yielding an intermediate result $(A \oplus B)$. The final sum, S, is then determined using Eq. 3 and stored.

Fig. 7 illustrates the construction process of the multi-bit FA. To FA operations for multi-bit data, the ripple carry adder (RCA) principle is employed, which propagates carry signals to enable computation [30]. In hardware implementations, the ℓ_1-distance calculation, defined as the sum of absolute differences between two vectors, is optimized using two's complement arithmetic to convert subtractions into additions [31]. To balance computational precision and energy efficiency, an 8-bit quantization scheme is adopted for the design of digital computing architectures [32].

Furthermore, to implement the absolute value operation, we need to evaluate the sign bit s[8]: when s[8]==1, the sum value s[8:0] needs to be bit-wise inverted and incremented by 1; conversely, when s[8]==0, the sum value can be output directly without any processing. This evaluation process can be expressed by the Eq. 5:

$$\begin{aligned} Sum[i] &= s[8] \oplus s[i] \\ Abs &= Sum + s[8] \end{aligned} \tag{5}$$

4. Design methodology & performance evaluation

To evaluate the performance of the proposed computing architecture, we utilized the SMIC 40 nm commercial PDK library for simulating hybrid CMOS/MTJ circuits. Using the Cadence Virtuoso IC6.1.8 environment, we employed Virtuoso ADEL and Spectre tools to construct and evaluate the circuit structures. Additionally, we performed circuit-level performance simulations using the NeuroSim Framework [33].

Table III Bit-cell level evaluation in memory operations.

Oper.	Iwrite (uA)	Vwrite (mV)	t (ns)	PDP (fJ)	Iw/Ic0
Write 0	119.8	967.7	1.76	207.5	2.48
Write 1	130.2	821.8	1.46	156.2	2.96
Oper.	**Iread (uA)**	**Vread (mV)**	**Delay (ns)**	**Power (fJ)**	**Error (%)**
Read 0	9.310	58.78	1.0	0.547	0.3
Read 1	6.101	81.25	1.0	0.496	2

Table IV Evaluation of the proposed CCLG in different logic mode.

Evaluation	OR/NOR	AND/NAND	XOR/XNOR	MAJ/MNJ
Power (uW)	5.867	4.983	3.088	4.581
Delay (ps)	65.02	39.95	63.98	48.79
PDP (aJ)	381.5	199.1	197.6	223.5

4.1 MRAM bit-cell design and simulation

To enhance the accuracy of computational results, ensuring the high reliability of the MTJ's read and write processes is crucial. While increasing the write pulse current to enhance write reliability, we also introduce a PCSA structure to balance power consumption and accuracy during the readout process [34]. Under the framework of the Verilog-A pMTJs model used, our read and write pulse times are set to 1 ns and 2 ns, respectively. Operating at 1.1 V, the details during the read and write processes are presented in Table III.

4.2 Circuit-level simulation of the proposed CCLG

The proposed CCLG structure's basic logic functionality was tested using 1,000 times Monte Carlo (MC) simulations, evaluating key parameters like delay and power consumption. Simulation results are shown in Fig. 8, with average values from the MC simulations detailed in Table IV.

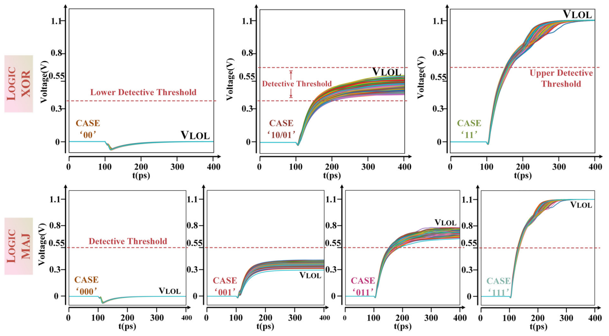

Fig. 8 MC simulation of proposed CCLG in different logic mode.

4.3 Function simulation of addition operation

Fig. 9 illustrates the transient simulation waveforms of the ℓ_1-distance calculation process. The activation value X = 47 and the weight W = 100 were used in the computation. During the initial clock cycle, the two's complement data corresponding to the activation X (X[8:0] = 0,0010,1111) and the weight W (W[8:0] = 0,0110,0100) were initialized in the same row of the array. Additionally, the carry data C[8:0] = 0,0000,0001 was also initialized. Subsequently, the carry data C[8:1] was computed bitwise according to

Eq. 3, and the updated carry data C[8:0] = 0,0011,1111 was written into the array within a single write cycle. Next, the sum value S[8:0] = 1,1100,1011 was computed in parallel according to Eq. 4, and both the sum value and the sign bit S[8] = 1 were written into a new column. Finally, the absolute value A[8:0] = 0,00110100 was computed according to Eq. 5, and the accumulation signal CTR = 1 was recorded. The entire 8-bit ℓ_1-distance computation process requires 31 ns, with the first 28 ns dedicated to the 8-bit two's complement addition of (A - W), consuming approximately 2.71 pJ. The final 3 ns are used to compute and output the absolute value result A[8:0] and the counter signal CTR, consuming approximately 1.72 pJ.

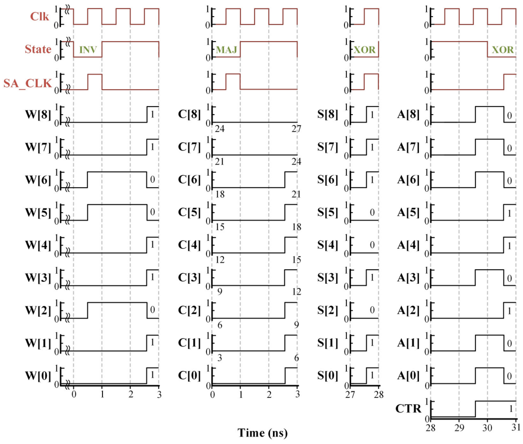

Fig. 9 Simulation waveforms during the absolute value calculation.

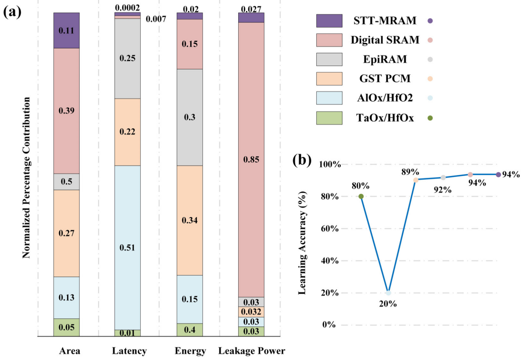

Fig. 10 (a) Normalized performance evaluation of different benchmark CIM frameworks and the proposed design. (b) Simulation learning accuracy comparison between our structure and others.

4.4 Performance analysis

We evaluated the computational performance of the proposed CIM framework in terms of metric functions and the full addition process, and compared it with related designs. The detailed results are presented in Table. V. Compared to a typical CPU [29], and MTJ-based CIM frameworks [35–37], our FA design achieves the lowest 8-bit addition power consumption (2.92 pJ). Additionally, its highly parallel array scheduling enables a computation time (28 ns) lower than that of typical MTJ-based CIM solutions. This provides a hardware foundation for high-efficiency AdderNet mapping.

Table V Performance summary and comparison with related works

	[29]	[35]	[36]	[37]	This Work
Device	CPU	STT MTJ	SOT MTJ	SHE MTJ	STT MTJ
Technology	45 nm	28 nm	28 nm	40 nm	40 nm
Cell Structure	46xT	8NVFF-8NVFA*	256x32 2T1M	7x8 2T1M	3x8 2T1M
Computing Type	8bit RCA	8bit RCA	8bit PPA*	8bit RCA	8bit RCA
Latency (ns)	12	140	33.5	85	28
Energy (pJ)	70	50.93	51.6	10.65	2.92
Voltage (V)	1.1	0.7	1.2	N/A	1.1

* Non-volatile flip-flops (NVFFs) and non-volatile (NVFAs) full adders are included in [49].

** PPA: parallel prefix adder.

Fig. 10 illustrates the benchmark performance comparison between the proposed CIM architecture and other CIM designs under the NeuroSim framework. Under the same computational task (1 million MNIST images recognition), proposed architecture demonstrates significant advantages in terms of latency and energy consumption. Furthermore, our proposed CIM architecture achieves a 6.28× speedup and a more than 7.5× reduction in leakage power compared to standard digital SRAM. Additionally, the proposed CIM framework also maintains the highest accuracy (94%) during the networks inference.

To further evaluate the performance of our proposed CIM architecture, we employed classical neural network models, including ResNet and VGG network. Using NeuroSim framework, we modified the standard VGG-8 and ResNet-18 networks. The macro size was set to 16 KB (128×128), with a maximum operating frequency of 181 MHz (in pipeline mode). For the VGG-8 network model inference, the proposed CIM macro achieves an energy efficiency of 31.48 TOPS/W for 8b/8b/8b precision and an accuracy of 86.98% in the CIFAR-10 classification task. For ResNet-18 network, our architecture also achieves an energy efficiency of 25.8 TOPS/W and a Top-5 accuracy of 73.97% (database: ImageNet). Compared to traditional DRAM- and MRAM-based digital IMC designs, our architecture achieves a 1.5× to 4.1× improvement in energy efficiency [38, 39].

5. Conclusion

This work proposes a novel STT-MRAM-based digital CIM macro designed to support low-bitwidth CNN training processes employing additive convolution. We propose a reliable in-memory logic implementation based on the CCLG structure, combined with efficient parallel computation processes, effectively reduces the hardware resource overhead associated with convolution calculations. We extracted key circuit parameters on the 40 nm CMOS technology platform and conducted system-level simulations using the NeuroSim framework. Simulation results demonstrate that the macro achieves a peak energy efficiency of 31.48 TOPS/W at 8b/8b precision while maintaining a top prediction accuracy of 86.98 %. This makes it suitable for hardware-constrained platforms, such as edge devices.

Acknowledgments

This work is financially supported by Guangzhou City Research and Development Program in Key Fields (Grant No. 202103020001). National Key R&D Program of China (Grant No.2022YFB4400200). Beijing Superstring Academy of Memory Technology (Grant SAMT-2022-PM02-15). Semiconductor Technology Innovation Center Corporation (Grant QYJS-2021-1000-B).

References

[1] Y. Chen, *et al.*: "Dadiannao: A machine-learning supercomputer," in Proc. Int. Symp. Microarchitecture (2014) 609. (DOI: 10.1109/MICRO.2014.58)

[2] N. D. Lane, *et al.*: "An early resource characterization of deep learning on wearables, smartphones and internet-of-things devices," in Proceedings of the 2015 international workshop on internet of things towards applications, ACM, (2015) 7. (DOI: 10.1145/2820975.2820980)

[3] A. Dundar, *et al.*: "Embedded streaming deep neural networks accelerator with applications," IEEE Trans. Neural Netw. Learn. Syst. **28** (2017) 1572. (DOI: 10.1109/TNNLS.2016.2545298)

[4] A. Pedram, *et al.*: "Dark memory and accelerator-rich system optimization in the dark silicon era," IEEE Design & Test **34** (2016) 39. (DOI: 10.1109/MDAT.2016.2573586)

[5] O. Mutlu, *et al.*: "Processing data where it makes sense: Enabling in-memory computation," Microprocessors and Microsystems **67** (2019) 28. (DOI: 10.1016/j.micpro.2019.01.009)

[6] M. Rastegari, *et al.*: "XNOR-Net: ImageNet classification using binary convolutional neural networks," in Proc. Eur. Conf. Comput. Vis. (2016) 525. (DOI: 10.1007/978-3-319-46493-032)

[7] J. Shen, *et al.*: "An anchor-free lightweight deep convolutional network for vehicle detection in aerial images," IEEE Trans. Intell. Transp. Syst. **23** (2022) 24330. (DOI: 10.1109/TITS.2022.3203715)

[8] S. M. PD, *et al.*: "A scalable network-on-chip microprocessor with 2.5D integrated memory and accelerator," IEEE Transactions on Circuits and Systems I: Regular Papers **64** (2017) 1432. (DOI: 10.1109/TCSI.2016.2647322)

[9] A. Sebastian, *et al.*: "Memory devices and applications for in-memory computing," Nature Nanotechnol. **15** (2020) 529. (DOI: 10.1038/s41565-020-0655-z)

[10] S. Khoram, *et al.*: "Challenges and opportunities: From near-memory computing to in-memory computing," in Proc. ACM Int. Symp. Phys. Des. (2017) 43. (DOI: 10.1145/3036669.3038242)

[11] H. A. D. Nguyen, *et al.*: "Memristive devices for computing: Beyond CMOS and beyond vonneumann," in Proc. IFIP/IEEE Int. Conf. Very Large Scale Integration (2017) 1. (DOI: 10.1109/VLSI-SoC.2017.8203479)

[12] M. Zabihi, *et al.*: "In - Memory Processing on the Spintronic CRAM: From Hardware Design to Application Mapping," IEEE Trans. Comput. **68** (2018) 1159. (DOI: 10.1109/TC.2018.2858251)

[13] S. Jain, *et al.*: "Computing in memory with spin-transfer torque magnetic RAM," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **26** (2018) 470. (DOI: 10.1109/TVLSI.2017.2776954)

[14] S. Angizi, *et al.*: "MRIMA: An MRAM-based in-memory accelerator," IEEE Trans. Comput. -Aided Design Integr. Circuits Syst. **39** (2019) 1123. (DOI: 10.1109/TCAD.2019.2907886)

[15] Y. Zhang, *et al.*: "Time-domain computing in memory using spintronics for energy-efficient convolutional neural network," IEEE Trans. Circuits Syst. I: Regul. Pap. **68** (2021) 1193. (DOI: 10.1109/TCSI.2021.3055830)

[16] S. Jung, *et al.*: "A crossbar array of magnetoresistive memory devices for in-memory computing," Nature **601** (2022) 211. (DOI: 10.1038/s41586-021-04196-6)

[17] H. Chen, *et al.*: "AdderNet: Do we really need multiplications in deep learning?" in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR) (2020) 1465.

[18] H. Shu, *et al.*: "Adder attention for vision transformer," Adv. Neural Inf. Process. Syst. **34** (2021) 19899.

[19] X. Chen, *et al.*: "An empirical study of adder neural networks for object detection," Adv. Neural Inf. Process. Syst. **34** (2021) 6894.

[20] D. Song, *et al.*: "Addersr: Towards energy efficient image super-resolution," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (2021) 15648.

[21] H. Diao, *et al.*: "A Multiply-Less Approximate SRAM Compute-In-Memory Macro for Neural - Network Inference," IEEE J. Solid - State Circuits (2024). (DOI: 10.1109/JSSC.2024.3433417)

[22] G. Seo and S. Ryu: "Area-efficient AdderNet hardware accelerator with merged adder tree structure," IEICE Electron. Express **20** (2023) 20230427. (DOI: 10.1587/elex.20.20230427)

[23] S. Zhu, *et al.*: "imad: An in - memory accelerator for addernet with efficient 8 - bit addition and subtraction operations," in Proc. Great Lakes Symp. VLSI (2022) 65. (DOI: 10.1145/3526241.3530313)

[24] P. Barla, *et al.*: "Spintronic devices: A promising alternative to CMOS devices," J. Comput. Electron. **20** (2021) 805. (DOI: 10.1007/s10825-020-01648-6)

[25] F. Ren and D. Markovic: "True energy-performance analysis of the MTJ-based logic-in-memory architecture (1-bit full adder)," IEEE Trans. Electron Devices **57** (2010) 1023. (DOI: 10.1109/TED.2010.2043389)

[26] Y. Wang, *et al.*: "Compact model of dielectric breakdown in spin - transfer torque magnetic tunnel junction," IEEE Trans. Electron Devices **63** (2016) 1762. (DOI: 10.1109/TED.2016.2533438)

[27] A. A. Khan, *et al.*: "Dielectric breakdown in Co-Fe-B/MgO/Co-Fe-B magnetic tunnel junction," J. Appl. Phys. **103** (2008) 123705. (DOI: 10.1063/1.2939571)

[28] S. Zhu, *et al.*: "FAT: An in-memory accelerator with fast addition for ternary weight neural networks," IEEE Trans. Comput. -Aided Design Integr. Circuits Syst. **42** (2022) 781. (DOI: 10.1109/TCAD.2022.3184276)

[29] M. Horowitz: "Computing's energy problem (and what we can do about it)," in Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC) (2014) 10. (DOI: 10.1109/ISSCC.2014.6757323)

[30] M. Shafique, *et al.*: "Cross-layer approximate computing: From logic to architectures," in Proc. 53rd Annu. Design Autom. Conf. (DAC) (2016) 1. (DOI: 10.1145/2897937.2906199)

[31] T. H. Cormen, *et al.*: *Introduction to Algorithms*, 3rd ed. Cambridge, MA (MIT Press, USA, 2009).

[32] B. Murmann: "Mixed-signal computing for deep neural network inference," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **29** (2021) 3. (DOI: 10.1109/TVLSI.2020.3020286)

[33] P.-Y. Chen, *et al.*: "NeuroSim+: An integrated device-to-algorithm framework for benchmarking synaptic devices and array architectures," in Proc. IEEE Int. Electron Devices Meeting (IEDM) (2017). (DOI: 10.1109/IEDM.2017.8268337)

[34] L. Zhang, *et al.*: "A novel sense amplifier to mitigate the impact of NBTI and PVT variations for STT - MRAM," IEICE Electron. Exp. **12** (2019) 20190238. (DOI: 10.1587/elex.16.20190238)

[35] E. Deng, *et al.*: "Synchronous 8-bit non-volatile full-adder based on spin transfer torque magnetic tunnel junction," IEEE Trans. Circuits Syst. I: Regul. Pap. **62** (2015) 1757. (DOI: 10.1109/TCSI.2015.2423751)

[36] X. Li, *et al.*: "Parallel-prefix adder in spin-orbit torque magnetic RAM for high bit-width non-volatile computation," IEEE Trans. Circuits Syst. II: Express Briefs **70** (2023) 761.

[37] M. Zabihi, *et al.*: "Using spin-hall MTJs to build an energy-efficient in-memory computation platform," in Proc. 20th Int. Symp. Qual. Electron. Design (ISQED) (2019) 52. (DOI: 10.1109/ISQED.2019.8697377)

[38] Y. He, *et al.*: "A 28nm 2.4Mb/mm2 6.9-16.3TOPS/mm2 eDRAM LUT-Based Digital-Computing-in-Memory Macro with In-Memory Encoding and Refreshing," in Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC) **67** (2024) 578. (DOI: 10.1109/ACCESS.2024.3511492)

[39] H. Cai *et al.*: "A 28 nm 2 Mb STT-MRAM computing-in-memory macro with a refined bit-cell and 22.4-41.5 TOPS/W for AI inference," in IEEE Int. Solid - State Circuits Conf. (ISSCC) Dig. Tech. Papers (2023) 500. (DOI: 10.1109/ISSCC42615.2023.10067339)