PDE-CIM: A 12nm FinFET 9.3TFLOPS/W Post-Aligned FP32-Lossless Digital CIM Macro with Input-Partial-Product LUT and Fused Compression for PDE Acceleration

Ming Li[1,2], Rui Zhang[1,2], Yutianshi Li[1,2], Hao Wu[1,2], Zichen Fan[3], Shang Shi[3], Bingxin Zhang[1,2], Yiyang Yuan[1,2], Yuanwei Song[2], Qianyi Zhang[3], Dengyun Lei[4], Jianfeng Gao[1], Qihao Liu[5], Zhiming Chen[3], Xiaoran Li[3], Yuan He[6], Yang Jiang[7], Xinghua Wang[3*], Feng Zhang[1*]

[1]Institute of Microelectronics of the Chinese Academy of Sciences, Beijing, China;[2]University of Chinese Academy of Sciences, Beijing, China; [3]Beijing Institute of Technology, Beijing, China; [4]Guangdong University of Technology, Guangdong, China; [5]Shandong SinoChip Semiconductors Co., Ltd, Shandong, China; [6]Capital Medical University, Beijing, China; [7]University of Macau, Macao, China

*Corresponding emails: 89811@bit.edu.cn, Zhangfeng_ime@ime.ac.cn

***Abstract*—This paper presents an FP32 Lossless Digital Computing-in-Memory (DCIM) macro for Partial Differential Equation (PDE) acceleration, named PDE-CIM. It has three key features: 1) An Input-Partial-Product-LUT (IPPL) CIM macro achieves 9.3TFLOPS/W with 40.8% power savings and 35.5% area reduction. 2) A Polymorphic Tensor Mapping (PTM) architecture with activation rollback and operand-adaptive adder gating reduces intermediate storage by 10.6x and adder-tree power by 74.3%. 3) A CIM-Fused Segmented Lossless Activation Compression (CFSLC) reaches 1.98x compression ratio while saving 77.5% area and 75.7% power. PDE-CIM achieves 5.5TFLOPS/W system-level energy efficiency, representing >11.2x over SOTA works on PDE acceleration.**

***Keywords*—Digital Computing-in-Memory, SRAM-based Macro, FP32-lossless computing, Lossless activation compression, Partial differential equation acceleration**

I. Introduction

In scientific computation, the high precision and algorithmic diversity of Partial Differential Equations (PDEs) pose significant challenges for hardware acceleration. While deep learning-based solvers effectively mitigate the curse of dimensionality [1-2], scientific applications demand stringent precision. Unlike error-tolerant AI workloads, low-precision formats (e.g., FP16, BF16) risk contradicting physical reality [3]. Although prior PDE ASICs exist [4-5], they lack generality. Digital SRAM-based CIM exhibits significant potential in performance, power, and area (PPA) [7]. However, applying high-precision CIM to general-purpose PDE acceleration faces three challenges (Fig. 1 and Fig. 2): 1) In SRAM-based CIM, the long-mantissa FP32 multiplication is a major PPA bottleneck. Existing paradigms [8-11] fail to achieve satisfactory trade-offs. 2) CIM architectures lack flexibility for diverse PDE computational patterns/dataflows. 3) High-dimensional FP32 data imposes heavy memory/bandwidth burdens, constraining system scalability and efficiency.

To address these challenges, this paper proposes PDE-CIM for general-purpose PDE acceleration, featuring three key innovations: (1) An IPPL-based CIM utilizes Radix-encoded weights and uses a lookup table (LUT) to pre-compute and store high-power-cost input partial products. These partial products are shared across a weight row via input broadcasting and cyclic reuse. (2) A PTM architecture supports configurable dataflows to adapt to diverse models. Through optimized mapping strategies, it significantly reduces storage requirements for intermediate results. Combined with input rollback and Operand-Adaptive Addition Topology, it further enhances CIM utilization and reduces power. (3) A CFSLC leverages the statistical characterization of segmented FP32 data, fusing final decompression with CIM pre-processing, enabling data to be fed directly into the macro, thereby minimizing data-transfer and storage pressure.

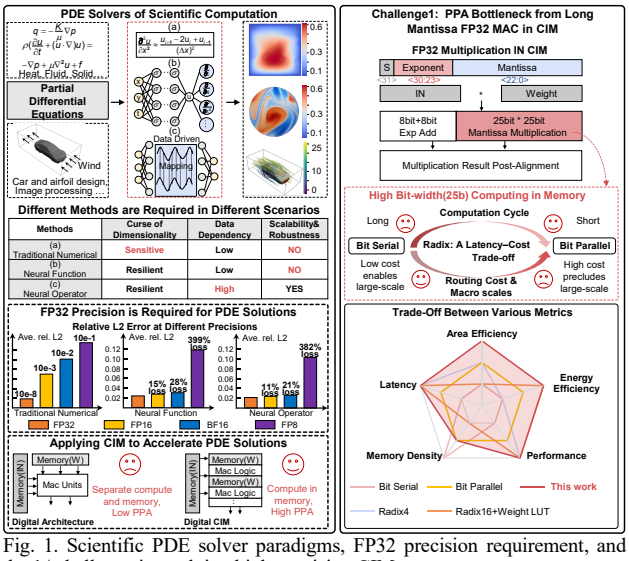


Fig. 1. Scientific PDE solver paradigms, FP32 precision requirement, and the 1st challenge in applying high-precision CIM.

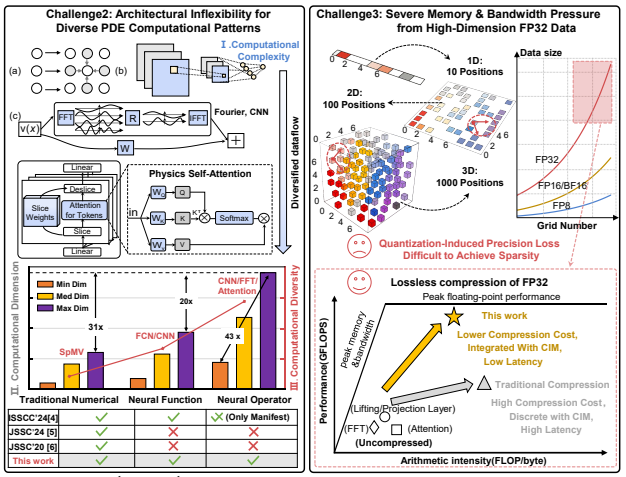


Fig. 2. The 2nd and 3rd challenges of applying high precision CIM to general purpose PDE acceleration.

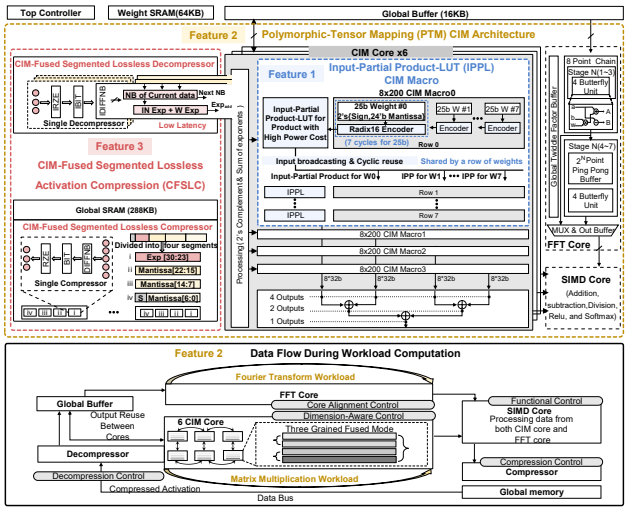


Fig. 3. Overall architecture and workload dataflow of the proposed PDE-CIM system, featuring three key innovations.

II. PDE-CIM OVERALL ARCHITECTURE

Fig. 3 illustrates the overall architecture, comprising six CIM cores, an FFT core, compressor /decompressor, a SIMD core, and memory. Each CIM core integrates four IPPL-based macros, in which input partial products are stored and broadcast across a weight row for reuse over seven cycles. The FFT core is deeply pipelined and optimized for real-valued FFTs to achieve double efficiency, and the system supports diverse dataflows between CIM and FFT cores (and within CIM cores) to minimize intermediate-result movement and storage. SIMD results are compressed via the segmented lossless compressor and stored. Compressed data is read back into CIM-fused decompressor with integrated pre-processing, which directly provides computation-ready data to the CIM macro.

A. Input-Partial Product-LUT CIM Macro

Fig. 4 shows the Radix-16 design exploration. For high-bit-width multiplication, compared with bit-serial and Radix-4 schemes, the Radix-16 approach significantly reduces power and computation cycles with only minor area overhead. Among four Radix-16-based computational paradigms, IPPL exhibits the best PPA trade-off. Power-Area Product analysis further shows that the advantage of IPPL becomes more pronounced as the number of macro columns increases, demonstrating excellent macro scalability.

Fig. 5 presents the Input-Partial-Product-LUT CIM macro. In each cycle, a 5-bit weight {W[i+3], W[i+2], W[i+1], W[i], W[i-1]} is encoded, and the resulting control signals {NEG, ZERO, EIGHT[3:0]} are used to generate the corresponding input partial products. These partial products are divided into low-cost terms, which require only shifts, and high-cost terms, which require both shifts and additions, such as {±3IN, ±5IN, ±6IN, ±7IN}. IPPL uses a LUT to generate and store the high-cost partial products. Its output is shared by a row of weights through input broadcasting and reused over seven cycles, effectively amortizing the hardware cost. The eight rows of 28b partial products in each column then go through post-alignment [12] and compensation, followed by the adder tree, accumulator, and

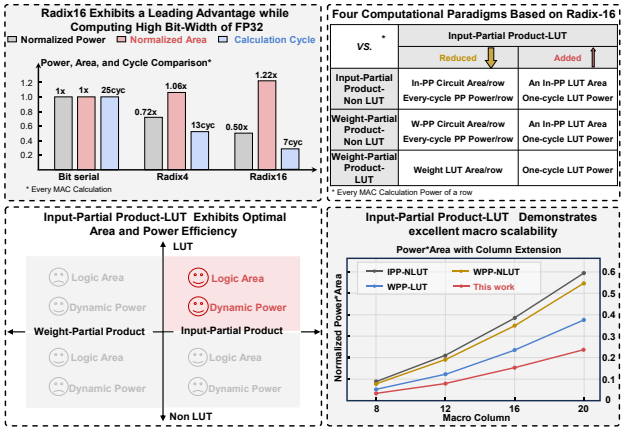


Fig. 4. Radix-16 design-space exploration for high-bit-width FP32: power/area/cycle comparison and four Radix-16 computational paradigms.

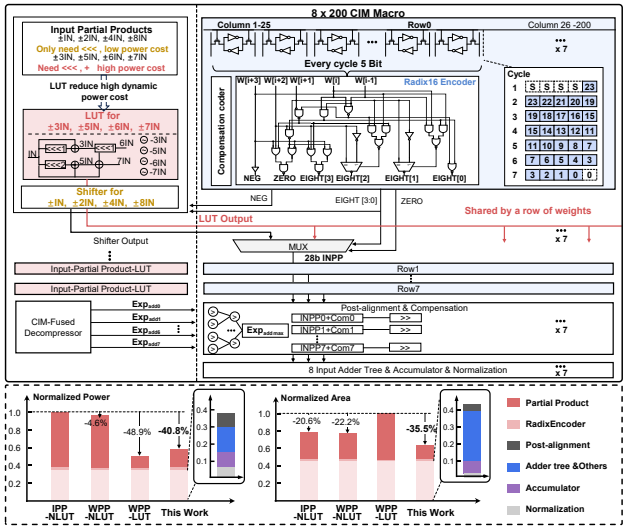


Fig. 5. Input-Partial-Product-LUT macro implementation, architecture and experiments, enabling efficient Radix-16 multiplication by utilizing a lookup table to store and reuse power-intensive partial products.

normalization unit to produce the final FP32 result. Compared with four alternatives, IPPL achieves up to 40.8% power savings and 35.5% area reduction.

B. Polymorphic-Tensor Mapping Architecture

Fig. 6 shows the Polymorphic Tensor Mapping architecture, which supports diverse models while optimizing resource utilization. Many PDE neural solvers lift input data to a high-dimensional space. As the dimensionality increases, the lifting stage becomes more expensive, leading to three challenges: 1) High-dimensional intermediate results cause large memory overhead; 2) CIM array utilization decreases under varying computational patterns; and 3) Adder trees in unused CIM rows still consume power. PTM integrates three solutions. First, an Optimized Dataflow & Mapping Strategy supports flexible dataflow paths for different solvers. In a Fourier layer, our strategy reduces intermediate-result storage by 10.6x while retaining 92% of weight reuse through weight pre-updating that matches the output shapes of Einstein summation and 1D convolution. For QK^T attention, the $K^{\wedge T}$ matrix is pre-stored as weights and each qi is computed on-the-fly and distributed to other cores, avoiding the large storage and access overhead of the Q matrix. Second, input rollback reuses activations through shift registers; when only two input rows are active, this boosts CIM utilization to

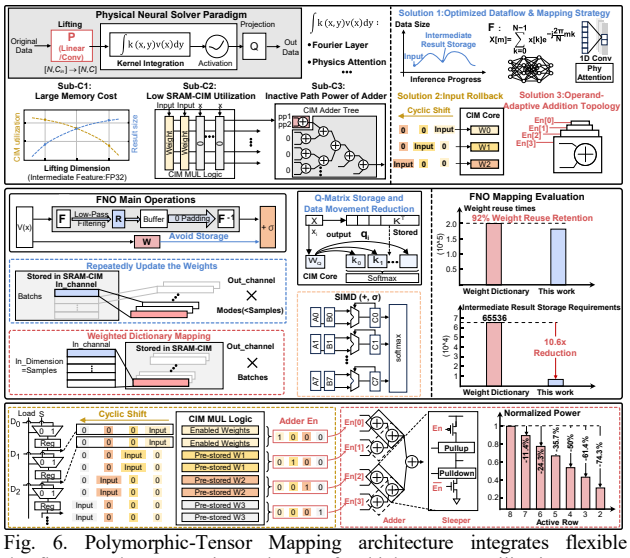


Fig. 6. Polymorphic-Tensor Mapping architecture integrates flexible dataflows and computation schemes for higher CIM utilization across diverse PDE models.

nearly 100%. Third, Operand-Adaptive Addition Topology dynamically gates unused adder-tree branches according to the number of active rows per cycle, saving up to 74.3% power when only two rows are active.

C. CIM-Fused Segmented Lossless Activation Compression

FP32 data in PDE solutions exhibits spatiotemporal redundancy, creating an opportunity for differential coding. Existing schemes, such as Spratio, process FP32 data as a whole and rely on a Magnitude-Sign (MS) operation that is hardware-inefficient [13]. Moreover, as characterized in Fig. 7, the sign, exponent, and mantissa fields have distinct statistical properties and thus different compression potentials. CIM-Fused Segmented Lossless Activation Compression therefore adopts a segmented strategy: FP32 is divided into four 8-bit segments; the first three are compressed using differential coding, hardware-friendly NegaBinary (NB) encoding, bit shuffling, and Run-length Zero Elimination (RZE), while the last is left uncompressed. NB transforms signed deltas into unified leading-zero formats, enabling the inverse transformation to fuse seamlessly with CIM pre-processing.

To further reduce decompression overhead, the decompression flow reverses RZE, bit shuffling, NB transformation, and differential coding, while the final IDIFFNB stage is fused with CIM pre-processing, as illustrated in Fig. 8. A 3-operand Han-Carlson adder [14] uses the reconstructed current value to generate the next original value while simultaneously calculating the exponent sum of the input and weight, thereby hiding the pre-processing latency inside decompression. Compared with a decoupled MS-based scheme, this design saves 77.5% area and 75.7% power. On the Transolver model with Darcy, Burgers, and Navier-Stokes datasets, CFSLC achieves overall compression ratios of 1.978x, 1.578x, and 1.324x, respectively. After amortizing compression/decompression overhead, the system reduces memory read/write power by up to 29.7% and improves effective memory utilization by up to 1.62x.

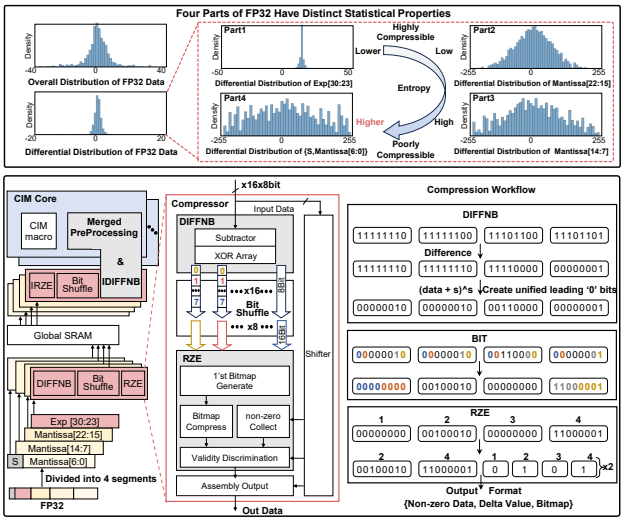


Fig. 7. Statistical characterization of segmented FP32 data and the DIFFNB–Bit-Shuffle–RZE lossless compression flow enabling CFSLC with a compact output format.

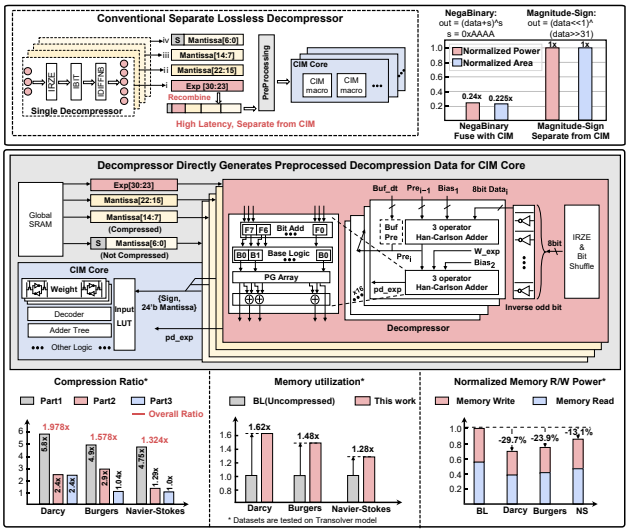


Fig. 8. CIM-fused segmented lossless decompressor architecture and experiments, fusing decompression with CIM computation to reduce power and area overhead.

III. MEASUREMENT RESULTS

Fig. 9 shows the visualization and accuracy /performance results on Traditional Numerical, Neural Function, and Neural Operator solvers. Experiments are conducted on four representative PDE models and datasets, covering both traditional benchmarks and large-scale industrial simulations. Compared with the original FP32 models, PDE-CIM incurs no degradation in relative L2 error under FP32 execution. At 1.0V and 450MHz, the chip achieves speedups of 2.2-55.8x over NVIDIA A100. Compared with the baseline non-LUT CIM macro without PTM and CFSLC, PDE-CIM reduces inference energy by 2.9-4.4x across the four applications.

Fig. 10 presents the measured results, die photo and chip summary. The fabricated 12nm PDE-CIM operates from 0.55V to 1.0V and from 70MHz to 450MHz, reaching a peak performance of 305 GFLOPS. Fig .11 compares PDE-CIM with prior PDE accelerators and related CIM works. Compared to the FP32 CIM Macro in [8], IPPL-CIM

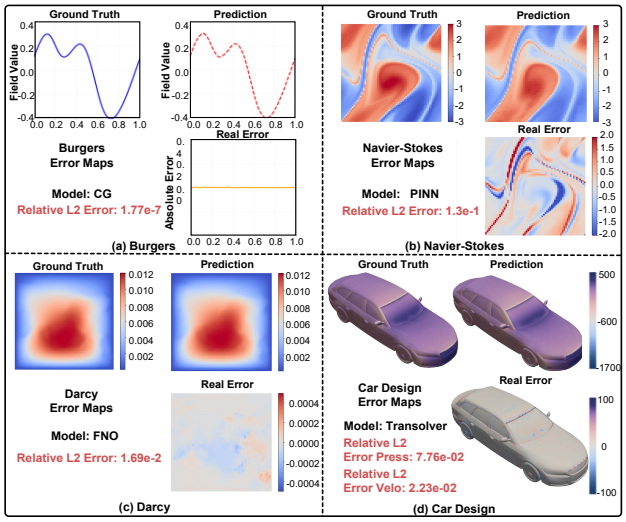
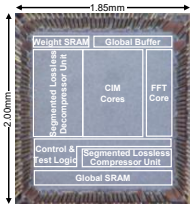


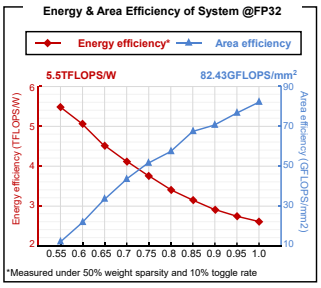
Fig. 9. Visualization results and performance of PDE-CIM on representative PDE models and datasets, validating high FP32 accuracy with the Relative L2 error metric.

Evaluation on Typical PDEs

Model	CG	PINN	FNO	Transolver
Dataset	Burgers	Navier-Stokes	Darcy	Car Design
Precision	FP32			
Compression Ratio	1.59x	1.27x	1.88x	1.29x
Relative L2	1.77e-07	1.3e-01	1.69e-02	2.23e-02(velo)
L2 Loss%*1	-0%	-0%	-0%	-0%
Speed Up*2	26.39x	9.63x	55.83x	2.19x
Energy Saving*3	3.63x	3.16x	4.41x	2.85x

*1: Compared with the original model's Relative L2 under fp32 precision, this work achieves lossless results.
*2: The baseline is NVIDIA A100. Include all on-chip components but exclude Off-chip DRAM access. Measured at 1.0V,450MHz for high performance.
*3:The baseline is input-partial product non-LUT CIM macro without PTM and CFSLC.





CHIP SUMMARY	
Technology	12nm FinFET
Die Area	3.7mm²
CIM Macro Area	0.403mm²
Buffer Size	16KB
Voltage	0.55V-1.0V
Frequency	70MHz-450MHz
Power	8.6-117.3mW
Data Precision	FP32
Peak Performance	305 GFLOPS*1*2
Area Efficiency	0.49 TFLOPS/mm²(macro)*2 0.08 TFLOPS/mm²(system)*2
Energy Efficiency	4.4-9.3 TFLOPS/W(macro)*3 2.6-5.5TFLOPS/W(system)*3

*1: One Operation(OP) represents one multiplication or one addition.
*2: Measured at the highest performance point, 1.0V, 450MHz.
*3: Measured at the highest efficiency point, 0.55V, 70MHz. Macro evaluation includes only CIM macros. System evaluation includes all on-chip components.

Fig. 10. Measured results, die photo, system energy/area-efficiency scaling at FP32, and chip summary table.

achieves 0.49 TFLOPS/mm^2 (3.27x higher) and 9.3 TFLOPS/W (1.58x higher). At the system level, energy efficiency reaches 5.5 TFLOPS/W, which is 11.2x higher than SOTA works on PDE acceleration.

IV. CONCLUSION

This work presents PDE-CIM, a 12nm FP32-lossless digital CIM macro for general-purpose PDE acceleration. By integrating the Input-Partial-Product-LUT macro, the Polymorphic-Tensor Mapping architecture, and the CIM-Fused Segmented Lossless Activation Compression scheme, PDE-CIM improves long-mantissa multiplication efficiency, accommodates diverse PDE dataflows, and reduces FP32 activation storage and transfer overhead. The fabricated chip supports Traditional Numerical, Neural Function, and Neural Operator solvers with no accuracy loss relative to the original FP32 models, achieving 0.49 TFLOPS/mm² and 9.3 TFLOPS/W at the macro level, and 5.5 TFLOPS/W at the system level. The silicon-proven results demonstrate that high-precision digital CIM is a practical and efficient solution for general-purpose PDE acceleration in scientific computing.

	ISSCC'22 [8]	JSSC'20 [6]	JSSC'24 [5]	ISSCC'24 [4]	This work
Architecture	Digital CIM	Analog CIM	Digital NMC	Digital	Digital CIM
Computing Type	Radix4	Bit-Parallel	Bit-Serial	Bit-Parallel	Radix16
Max Bit Precision	FP32 (Lossy)	INT5	Fixed-Point 16	INT32	✓FP32 (Lossless)
Methodology	Non-PDE scenarios	Traditional Numerical	Traditional Numerical	Traditional Numerical Neural Function Neural Operator(Manifest)	Traditional Numerical ✓Neural Function ✓Neural Operator
Memory utilization	1x	1x	1x	1x	✓>1.2x
SRAM	N/A	148KB	92Kb	513KB	352KB
Technology	28nm	180nm	65nm	28nm	12nm FinFET
Supply Voltage(V)	0.6-1.0	/ -1.8	0.6-1.0	0.55-0.9	0.55-1.0
Die Area(mm²)	6.69	1.868	0.811(core)	3.80	3.70
Max Frequency	220MHz	200MHz	300MHz	500MHz	450MHz
Power(mW)	12.5-69.4	66.4*5	0.31*5	147*5	8.6-117.3
Peak Performance (GOPS or GFLOPS)	140@32b	14.22@5b	1.63@16b	336@16b	305@32b*1*2*3
Area Efficiency (TOPS/mm² or TFLOPS/mm²)	0.15@32b(macro) 0.02@32b(system)	0.007@5b	0.002@16b	0.088@16b	0.49@32b (macro)*1*3 0.08@32b (system)*1*3
Energy Efficiency (TOPS/W or TFLOPS/W)	5.9@32b(macro) 3.7@32b(system)	0.857@5b(macro)	N/A	0.49@32b	9.3@32b (macro)*4 5.5@32b (system)*4

*1: One Operation(OP) represents one multiplication or one addition *2: Throughput of FFT Core is included *3: Measured at the highest performance point, 1.0V, 450 MHz *4: Measured at the highest efficiency point, 0.55V, 70MHz *5 Taken from the source text comparison table

Fig. 11. Comparison with state-of-the-art PDE accelerators and related CIM works.

ACKNOWLEDGMENT

This work was supported in part by National Natural Science Foundation of China under Grants 92464201, U2341218, T2293732, 62322412; National Key Research and Development Program of China under Grant 2023YFB4402400.

REFERENCES

[1] S. Huang *et al.*, "Partial Differential Equations Meet Deep Neural Networks: A Survey," *IEEE TNNLS*, vol. 36, pp. 13649–13669, 2025.

[2] P. Wei *et al.*, "PDENNEval: A Comprehensive Evaluation of Neural Network Methods for Solving PDEs," in *IJCAI*, 2024, pp. 5181–5189.

[3] C. Hao , "Exploring and Exploiting Runtime Reconfigurable Floating Point Precision in Scientific Computing: A Case Study for Solving PDEs," in *ASPDAC*, 2025, pp. 1041–1047.

[4] Y. Ju *et al.*, "A 28nm Physics Computing Unit Supporting Emerging Physics-Informed Neural Network and Finite Element Method for Real-Time Scientific Computing on Edge Devices," in *ISSCC*, 2024, pp. 366–368.

[5] J. Mu *et al.*, "A Scalable and Reconfigurable Bit-Serial Compute-Near-Memory Hardware Accelerator for Solving 2-D/3-D Partial Differential Equations," *IEEE JSSC*, vol. 59, pp. 2706–2716, 2024.

[6] T. Chen *et al.*, "A 1.87-mm² 56.9-GOPS Accelerator for Solving Partial Differential Equations," *IEEE JSSC*, vol. 55, pp. 1709–1718, 2020.

[7] Z. Lin *et al.*, "A review on SRAM-based computing in-memory: Circuits, functions, and applications," *J. Semicond.*, vol. 43, p. 031401, 2022.

[8] F. Tu *et al.*, "A 28nm 29.2TFLOPS/W BF16 and 36.5TOPS/W INT8 Reconfigurable Digital CIM Processor with Unified FP/INT Pipeline and Bitwise In-Memory Booth Multiplication for Cloud Deep Learning Acceleration," in *ISSCC*, 2022, pp. 254–256.

[9] Y.-D. Chih *et al.*, "An 89TOPS/W and 16.3TOPS/mm² All-Digital SRAM-Based Full-Precision Compute-In Memory Macro in 22nm for Machine-Learning Edge Applications," in *ISSCC*, 2021, pp. 252–254.

[10] H. Wu *et al.*, "A 28-nm 19.9-to-258.5-TOPS/W 8b Digital Computing-in-Memory Processor With Two-Cycle Macro Featuring Winograd-Domain Convolution and Macro-Level Parallel Dual-Side Sparsity," *IEEE JSSC*, vol. 60, pp. 347–361, 2025.

[11] C.-F. Lee *et al.*, "A 12nm 121-TOPS/W 41.6-TOPS/mm² All Digital Full Precision SRAM-based Compute-in-Memory with Configurable Bit-width For AI Edge Applications," in *Symp. VLSI*, 2022, pp. 24–25.

[12] Z. Yue *et al.*, "A 51.6TFLOPs/W Full-Datapath CIM Macro Approaching Sparsity Bound and <2^-30 Loss for Compound AI," in *ISSCC*, 2025, pp. 1–3.

[13] N. Azami *et al.*, "Efficient Lossless Compression of Scientific Floating-Point Data on CPUs and GPUs," in *ASPLOS*, 2025, pp. 395–409.

[14] A. K. Panda *et al.*, "High-Speed Area-Efficient VLSI Architecture of Three-Operand Binary Adder," *IEEE TCSI*, vol. 67, pp. 3944–3953, 2020.