D²CIM: A 28nm 53.3 TFLOPS/W Decoding Digital CIM Macro for Efficient FlashMLA-based LLM Inference

Jia Chen[1,2], Zihao Xuan[1,2], Yewen Li[1,2], Zhuoya Yan[1], Zimu Li[1], Yat-Fong Yung[2], Yi Deng[2], Xiao Huo[2], Chi-Ying Tsui[1,2], Kwang-Ting Cheng[1,2], Fengbin Tu[1,2*]
[1]The Hong Kong University of Science and Technology; [2]AI Chip Center for Emerging Smart Systems, Hong Kong, China
*Corresponding Author (fengbintu@ust.hk)

***Abstract*—This paper presents D²CIM, a Digital Compute-in-Memory (DCIM) macro optimized for FlashMLA-based LLM decoding. To overcome low utilization, FP alignment conflicts, and throughput mismatch in conventional DCIMs, D²CIM introduces a hybrid AT-ACC MAC, a transpose-free FP8 alignment mechanism, and CIM-friendly flash-attention dataflow optimizations. Fabricated in 28 nm CMOS, it achieves up to 53.3 TFLOPS/W, improving LLM decoding energy efficiency by 1.93×-2.79× over SOTA designs. Evaluated on DeepSeek V3 model, D²CIM improves the runtime latency by 2.57× and improves computing accuracy by 1.17×.**

***Keywords*—edge LLM, decoding, FlashMLA, digital computing-in-memory**

I. Introduction

In power- and bandwidth-constrained edge scenarios, achieving efficient inference of large language models (LLMs) has become a critical challenge. As shown in Fig. 1, LLM inference consists of two main stages: the prefilling stage and the decoding stage. The prefilling stage processes a long input sequence (X_0-X_{N-1}) in parallel, which is a compute-intensive workload; the decoding stage generates tokens one by one (X_N), while memory access latency becomes the dominant bottleneck. Digital Compute-in-Memory (DCIM) technology, by keeping weight matrices resident within the memory arrays and significantly reducing data movement, is inherently well-suited to the high energy-efficiency requirements of edge LLM deployment. State-of-the-Art (SOTA) DCIM accelerators [1-6] have been increasingly optimized for the compute-intensive prefilling workloads, which feature parallel processing of multiple tokens and extensive reuse of both weight and KV data. Under such conditions, a weight-stationary CIM architecture can effectively deliver high throughput and energy efficiency. However, in the decoding stage, weights and KV cannot remain stationary on-chip for long periods due to sequential token generation. Frequent data movement and memory updates diminish CIM's architectural advantages. To mitigate this, Flash-MLA [7], as an advanced algorithm against basic Multi-Head Attention (MHA) method, attempts to improve decoding efficiency by introducing low-rank decomposition (LoRA), KV compression, FP8 quantization, and flash attention. These techniques help reduce off-chip data access and improve KV reuse at the algorithm level. Nevertheless, when implemented on previous DCIM architectures, even Flash-MLA cannot fully overcome the fundamental limitations of DCIM during token-by-token decoding, which remains the primary performance and energy bottleneck in DCIM-based LLM acceleration.

This limitation in applying DCIM techniques to Flash-MLA can be attributed to three key design challenges, as shown in Fig. 2: **1) Low utilization caused by irregular matrix multiplications .** LoRA and dynamic KV growth produce matrix multiplications (MMs) with varying aspect ratios, which causes poor utilization of fixed-size DCIM

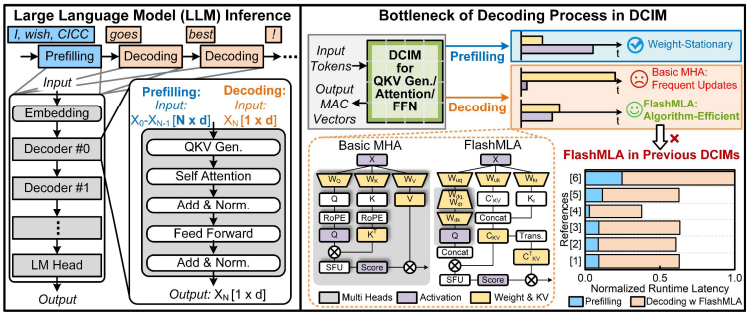


Fig. 1 Illustration of LLM inference and the bottleneck of LLM decoding in previous DCIM macros.

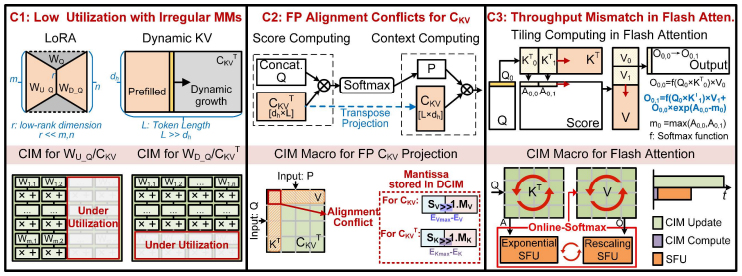


Fig. 2 Proposed three design challenges when applying DCIM techniques to Flash-MLA.

arrays and thus degrades throughput. For LoRA in Q/K/V generation, the low-rank adaptation introduces smaller projection matrices ($r \ll m,n$), resulting in high-aspect-ratio up-projection matrix (W_U) and low-aspect-ratio down-projection matrix (W_D). For dynamic KV workload, the compressed K/V (CKV) matrix size further expands dynamically with the increasing token length ($L \gg dh$) during decoding. CKVT represents a low-aspect-ratio matrix, while CKV represents a high-aspect-ratio matrix. **2) FP Alignment conflicts and clipping loss in transposable C_{KV} projection.** C_{KV} is reused in score ($Q \times C_{KV}^T$) and context ($P \times C_{KV}$) computing, which requires transposed and non-transposed MAC computing in DCIM array, respectively. However, the FP8-based pre-alignment operation causes alignment conflicts in DCIM along orthogonal access directions, which needs re-upload the exponent-aligned mantissa to the memory array. Besides, pre-alignment method in DCIM is recognized as precision loss due to the truncated effective mantissa residual. **3) Throughput mismatch in tiling-based flash attention.** Performing tile-wise flash attention on DCIM arrays requires frequent full-size KV loading. Such intensive access pattern places heavy requirement on weight update in DCIM, impacting overall performance. Moreover, for SoftMax operation with tiling-based manner, the row-wise maximums of score matrix must be updated and attention output require extra exponential scaling and normalization, leading to throughput mismatch between VMM and non-linear operation.

II. D²CIM Overall Architecture

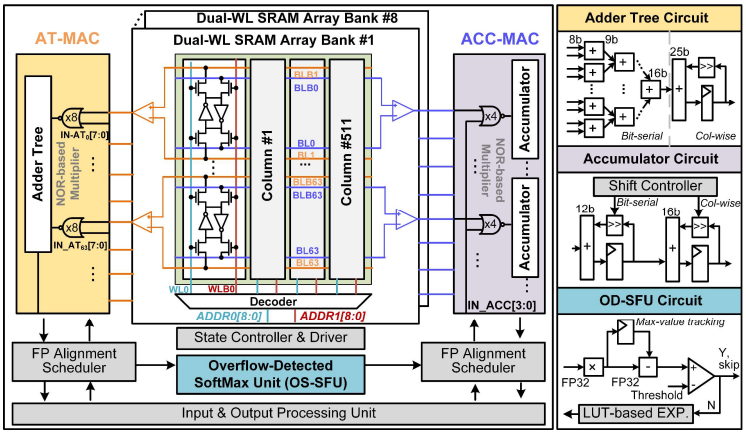


Fig. 3 Architecture overview of proposed D²CIM.

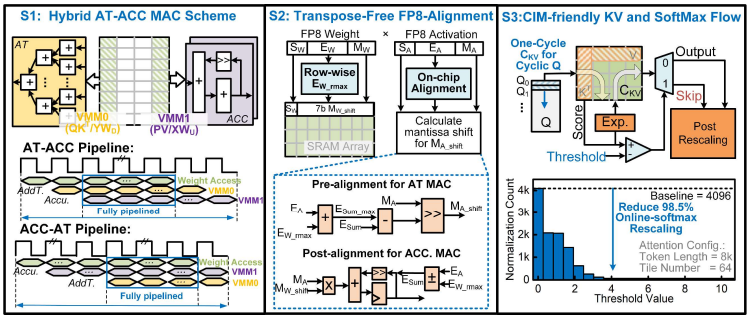


Fig. 4 Three design solutions for implementing FlashMLA-based LLM decoding.

This work proposes a D²CIM macro, aiming at accelerating FlashMLA-based LLM decoding in Digital CIM (Fig. 3). The macro includes 8 64×512 SRAM-based memory banks, NOR-gate-based multipliers, an adder tree (AT), 512 accumulators (ACC), an overflow-detected SoftMax unit (OD-SFU), and peripheral controller and drivers, such as input/output processing unit and FP alignment scheduler. Specifically, the dual-WL SRAM cells are used to enable parallel readout of two weight vectors from either the same or different two columns. Multi-bit weight vectors are sequentially fed into NOR-based multipliers on both sides of the array to perform multiplication between 1-bit W and multi-bit IN. Bit-serial manner is utilized for 8-bit weight representation. In AT MAC path, the input of the multipliers is a 8-bit 512-element activation vector (IN_AT). The adder tree is used for the summation of 512-row 8-bit multiplications in parallel, while the circuit includes full adders and a shift accumulator for total 25-bit output aggregation. In ACC MAC path, the input is a 4-bit data (IN_ACC) which is broadcasted to all multipliers. 512 accumulators is designed for the summations of 512 row-wise partial sums. Total 21-bit MAC results are obtained with cycle-by-cycle accumulation. Although the 21-bit value should be the maximum bit-width, later INT-to-FP operation will truncate the INT value to obtain the FP representation. By only preserving high 16bit value in accumulator, the hardware cost will be reduced without severe computing error. Then, for the whole array, the width-oriented computation is implemented sequentially in both AT and ACC MAC path. The FP alignment scheduler is designed for the two MAC paths to complete INT-to-FP conversion. Furthermore, an OD-SFU is designed to implement SoftMax operation IN D²CIM macro to enable a complete attention flow.

Targeting mentioned design challenges, D²CIM macro mainly features following three solutions (Fig. 4), including a hybrid AT-ACC MAC scheme, a transpose-free FP8 alignment mechanism, and CIM-friendly flash-attention dataflow optimizations:

1) To address the utilization bottleneck, a hybrid Adder-Tree and Accumulator (AT–ACC) MAC scheme is proposed. In the AT path, wide-but-shallow MMs are processed across multiple input channels in parallel but time-multiplexed along the width dimension, providing low latency with modest hardware cost. Conversely, the ACC path handles narrow-but-deep MMs, where deep partial sums are accumulated cycle-by-cycle with high output channel parallelism. By co-scheduling the two paths with complementary parallelism, the fully-pipelined AT-ACC and ACC-AT flow for MMs with varying aspect ratios is established without idle bubbles. The hybrid AT–ACC MAC circuit sustains high throughput across LoRA and dynamic KV workloads, while maximizing CIM utilization under various MM aspect ratios.

2) To support transposable C_{KV} projection and enhance FP computing precision, a transpose-free FP8 alignment mechanism is proposed. The FP8 weights are first pre-aligned according to their row-wise maximum and then stored as 8-bit mantissas in the SRAM array. The row-wise weight alignment scheme is proposed to minimize the weight mantissa shift. Then, the FP8 activations are dynamically re-scaled on-chip based on the locally accumulated sum maximum in both AT and ACC MAC path. Hence, the static weight matrix is reused for both orthogonal directions in DCIM. The overall representable shift range of original FP8 weights is expanded, thereby enhancing floating-point precision with minimal hardware overhead.

3) To alleviate the performance loss caused by the throughput mismatch from CIM updates and non-linear operations, CIM-friendly flash-attention dataflow is proposed. On the one hand, by re-ordering tile-wise computing with Q-loop-first manner, the C_{KV} tiles are retained in the DCIM array, while multiple Q tiles are streamed sequentially to maximizes in-array data reuse. On the other hand, in OD-SFU, a pre-set threshold is introduced to perform post-rescaling SoftMax with reduced exponential and normalization operations. One-time global normalization is performed to all SoftMax outputs as the last step. Statistically, about 98.5% online-SoftMax rescaling can be eliminated with introduced threshold. By eliminating repetitive max-update and scaling steps during per-tile processing, smooth pipeline with throughput matching is completed between the VMM and the non-linear SoftMax.

III. Circuit Design and Implementation

A. A hybrid Adder-Tree and Accumulator (AT–ACC) MAC circuit

Fig. 5 illustrates the fully pipelined MMs computation with the hybrid AT-ACC MAC circuit for LoRA and dynamic KV workloads, respectively. For LoRA-MM mode, an ACC-to-AT flow is utilized to execute $X \times W_U$ and $Y \times W_D$ computation. The W_U matrix with a high aspect ratio is first multiplied with X in the ACC MAC path. The resulting Y vectors are then fed into the AT MAC path to process W_D, which has a low aspect ratio. Since both matrices share the same low-rank dimension, the two steps can be iteratively overlapped to improve hardware utilization. Conversely, for KV-MM mode, an AT-to-ACC flow is adopted to perform $Q \times C_{KV}^T$ and $P \times C_{KV}$ computation. C_{KV}^T matrix with low aspect ratio is stored and calculated with Q vector,

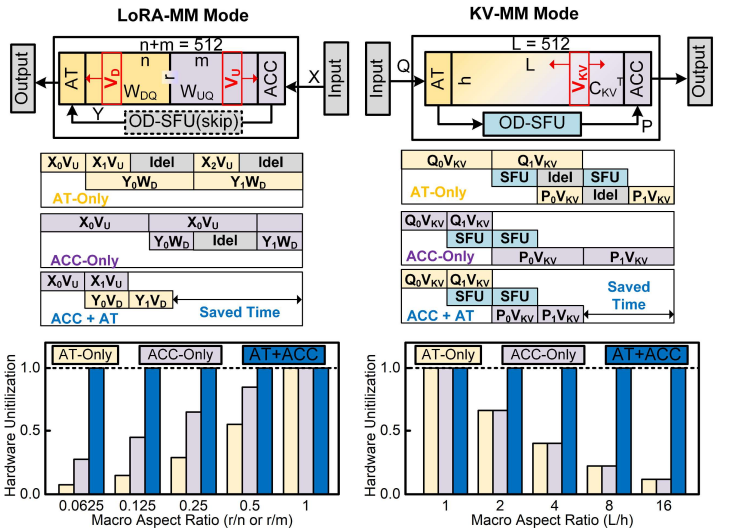


Fig. 5 Fully pipelined MMs with the hybrid AT-ACC MAC circuit for LoRA and dynamic KV workloads.

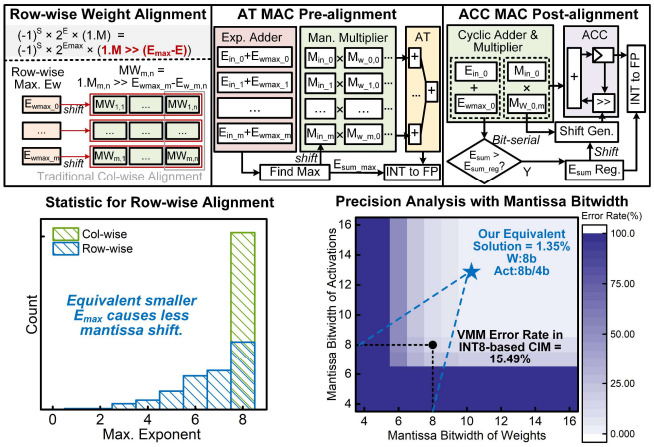


Fig. 6 The transpose-free FP8 alignment implementation for C_{KV} projection on DCIM array.

and the result is sent into OD-SFU for SoftMax operation. Then, $P \times C_{KV}$ computation is implemented by cycle-by-cycle accumulation in ACC MAC path. This temporal accumulation enables direct reuse of the same weight values previously accessed, eliminating weight transposition. The continuous AT-to-ACC flow thus maintains pipeline integrity and maximizes CIM utilization across variable KV workloads. Compared with AT-only CIMs, which suffer from low utilization in $Y \times W_D$ or $Q \times C_{KV}^T$ operations, and ACC-only CIMs, which exhibit underutilization in $X \times W_U$ or $P \times C_{KV}$ operations, the hybrid design improves the CIM utilization by 1.17×–13× for LoRA and 1.3×–8× for dynamic KV growth, respectively.

B. A transpose-free FP8 alignment mechanism

To enable transpose-free FP8 C_{KV} projection in DCIM array, a hybrid alignment mechanism is proposed as shown in Fig. 6. At first, the FP8 C_{KV} matrix are row-wise pre-aligned and statically stored in the CIM array. Compared with traditional col-wise alignment in a weight mantissa matrix, the row-wise method statistically reduces large exponent occurrences and thus minimizes mantissa shifts, preserving higher effective precision in the stored weights. To support FP computing, as designed, for $Q \times C_{KV}^T$ which is implemented in AT MAC path, activations here are dynamically pre-aligned based on the locally

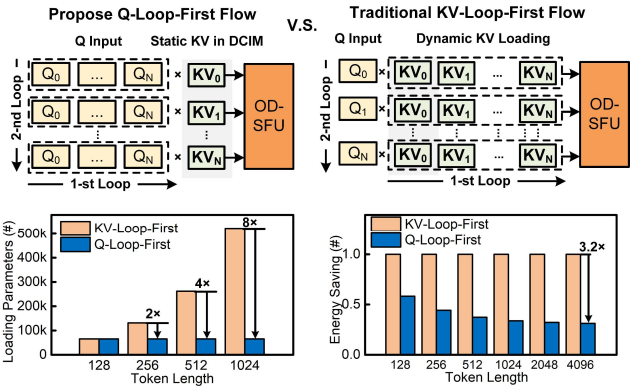


Fig. 7 Dataflow optimization for tiling KV loading in flash attention.

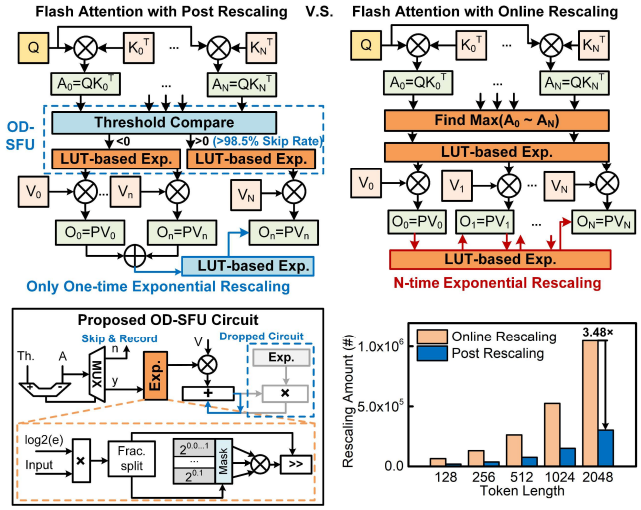


Fig. 8 Dataflow optimization for SoftMax post-rescaling normalization with tiling-based manner in flash attention.

accumulated exponent ($E_{in} + E_{wmax}$). Then, for ACC MAC based $P \times C_{KV}$, the cycle-by-cycle accumulation allows E_{sum}-based post-alignment to be performed within the iterative shift-and-add cycles. This hybrid approach ensures that the same weight matrix can be reused along orthogonal FP-MAC direction, achieving transpose-free operation with <1.35% accuracy loss in DCIM MAC computing, while providing 11.47× numerical precision improvement with similar hardware cost.

C. CIM-friendly flash attention dataflow optimizations

A CIM-friendly flash attention dataflow is optimized to eliminate frequent CIM updates and heavy non-linear operation overhead. By re-ordering tiling-loop sequence in flash attention, frequent CIM loading is eliminated; By introducing a post-rescaling method, an overflow-detected SoftMax unit (OD-SFU) is implemented for fully pipelined operation with parallel MM operations in CIM array.

In vanilla flash attention, the standard dataflow processes Q in 1-st loop and C_{KV} tiles in 2-nd loop. Frequent KV loading in CIM macros incurs significantly high energy consumption and latency, which is magnified during decoding with expanding KV cache size. Here, A Q-loop-first flow is proposed, while the tiled C_{KV} data is stored stationary in CIM with looping Q, to enable single-cycle KV loading. The loading parameters is reduced by 8× and the energy consumption of attention computation is reduced by 3.2×.

For SoftMax operation, tiling computing manner requires iterative LUT-based exponential scaling and normalization, incurring

Comparison with State-of-the-art DCIM Macros

		[1]	[2]	[3]	[4]	[5]	[6]	This Work
Technology (nm)		28	28	28	28	28	28	28
Memory Size (kb)		64	192	64	32	224	96	257
CIM Type		Digital	Hybrid	Digital	Digital	Digital	Digital	Digital
MAC Type		AddT	Accu	AddT	AddT	Accu	AddT	**AddT & Accu**
Function Support		MM	MM	MM	MM/ Transpose	MM/Route	MM/NL	**MM//NL/ Transpose/**
Data Precision		INT8/BF16	INT8/BF16	INT8/BF16	INT4/8/FP8/16	INT8/FP8/16	INT4/8	INT8/FP8
Voltage (V)		0.6~0.9	0.6~0.95	0.55~0.9	0.55~0.9	0.62~0.9	0.6~1.0	0.6~0.9
Peak Performance (TFLOPS) *		0.375	5.31	0.13	0.12	10.17	0.63	0.25
Energy Efficiency (TOPS/W) **		19.5~44	51.12~111.17	25.47~90.15	17.5~50 (51.1~145.8'')	43.2~115	-- ~46.6	**18.2 ~53.3''**
Area Efficiency (TOPS/mm²) ***		-- ~2.57	-- ~2.74	0.34~0.99	-- ~2.89	0.72~3.81	-- ~2.47	2.29
Energy Consumption on DeepSeek V3 (uJ/token)	Prefill	9.13	8.78	9.18	7.91	11.57	10.82	12.38
	Decode	53.84	51.91	54.15	35.02	51.20	32.92	**10.33**
	Overall	**2.77×↑**	**2.67×↑**	**2.79×↑**	**1.89×↑**	**2.76×↑**	**1.93×↑**	**1×**

* Measured at 0.9V, 500MHz under FP8-IN,FP8-OUT and FP16 OUT. ** Measured from 0.6V to 0.9V,100MHz; INT4-based FP8 MAC in [4]; INT8-based FP8 MAC in this work. *** Measured at 0.9V, 500MHz, without SFU.

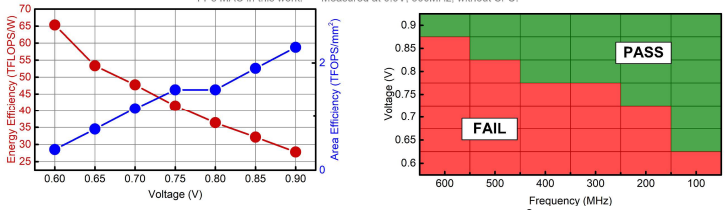
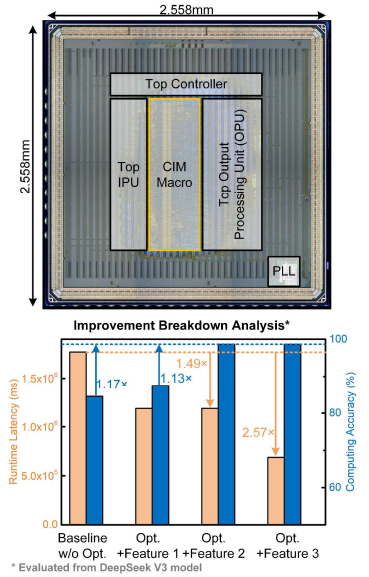


Fig. 9 Measurement results in fabricated D²CIM macro and comparison with state-of-the-art DCIM macros.



Technology	TSMC 28nm CMOS
Die Area	2.558mm × 2.558mm (CIM Macro: 0.511 mm²)
Memory Size	256Kb + 1Kb (Calibre)
Supply Voltage	0.6-0.9 V
Data Precision	INT8/FP8
Macro Performance for VMM (FP8)	
Energy Efficiency (AT. MAC)	18.2~53.35 TFLOPS/W
Energy Efficiency (ACC. MAC)	16.98~44.74 TFLOPS/W
Macro Performance for LoRA MM (FP8)	
Energy Efficiency	11.5~30.42 TFLOPS/W
Peak Area Efficiency	0.96 TFLOPS/mm²
Macro Performance for K^TV MM (FP8)	
Energy Efficiency	10.33~29.9TFLOPS/W
Peak Area Efficiency	0.93TFLOPS/mm²
Total Performance for DeepSeek one-layer (FP8)	
Prefill	12.38 uJ/token
Decode	10.33 uJ/token

Fig. 10 Die micrograph, improvement analysis, and summary table.

significant computation cost. Post-rescaling dataflow in this work handles score ($Q \times C_{KV}^T$) values dynamically. If the values is below a threshold (Thre.), it is propagated for subsequent $P \times C_{KV}$ calculation. Otherwise, the values which surpass the Thre. are temporarily skipped and recorded, awaiting one-time batched exponential rescaling at the last step. Under post-rescaling dataflow, a lightweight OD-SFU Circuit is designed, integrating a comparator to skip computations to eliminate extra cost of an exponential rescaling unit. Results show that the post-rescaling method reduces computation by over 70% across token lengths from 128 to 2048, demonstrating its critical advantage for deploying flash attention mechanism on CIM accelerators.

IV. Meseasurement Results

Fig. 9 presents the measured voltage-frequency scaling curves, Shmoo results, and technical comparison with SOTA DCIM macros of the fabricated D²CIM macro in 28nm CMOS technology. The chip operates over a 0.6–0.9V supply range and a 100–600MHz frequency range with MM/NL/Transpose function support. For FP8 VMM operations, peak energy-efficiency is achieved by 53.3 TFLOPS/W at 0.65V, while peak area efficiency reaches 2.29 TFLOPS/mm² at 0.9V. Estimated under one-layer DeepSeek V3 model, the overall performance improvement can be obtained with 1.93× to 2.79× improvement. Fig. 10. presents the chip's die photo, evaluation breakdown and summary table. Macro performance under LoRA and dynamic KV workloads are also measured. In LoRA-MM scenarios, the macro reaches a peak energy efficiency of 30.42 TFLOPS/W at FP8. For KV-MM mode, the 257Kb macro integrates in-situ transpose, $Q \times C_{KV}^T$, SoftMax, and $P \times C_{KV}$ operations, achieving a peak energy efficiency of 30.42 TFLOPS/W at FP8 precision. Evaluation on the DeepSeek V3 model demonstrates energy consumption of 12.38 µJ/token during the prefill stage and 10.33 µJ/token during decoding, which reduce 1.89× to 2.79× energy consumption than SOTA CIM macros. To evaluate three proposed features, the D²CIM improves the runtime latency of DeepSeek V3 by 2.57× and improves computing accuracy by 1.17×, compared with the baseline design without optimizations. These results shows that the proposed D²CIM macro can achieve potential performance gain under LLM decoding workloads.

V. Conclusion

In conclusion, the proposed D²CIM macro effectively addresses the fundamental limitations of existing DCIM architectures in LLM decoding by hybrid AT-ACC MAC, precision-enhanced FP8 processing, and optimized dataflow scheduling. Through its hybrid AT-ACC MAC, transpose-free FP alignment, and CIM-friendly flash-attention dataflow optimizations, D²CIM achieves high utilization, reduced energy consumption, and improved reliability across the complete attention computing. Experimental results confirm significant performance and efficiency improvements, demonstrating D²CIM's strong potential for low-latency, energy-efficient edge LLM inference.

Acknowledgment

This research was supported in part by NSFC Grant 62422407, RGC Grant 26204424, and partially conducted by ACCESS – AI Chip Center for Emerging Smart Systems, supported by the InnoHK initiative of the Innovation and Technology Commission of the Hong Kong Special Administrative Region Government.

References

[1] An Guo et al., "A 28nm 64-kb 31.6-TFLOPS/W Digital-Domain Floating-Point-Computing-Unit and Double-Bit 6T-SRAM Computing-in-Memory Macro for Floating-Point CNNs," ISSCC, pp. 128-130, 2023.

[2] Y. Yuan et al., "34.6 A 28nm 72.12TFLOPS/W Hybrid-Domain Outer-Product Based Floating-Point SRAM Computing-in-Memory Macro with Logarithm Bit-Width Residual ADC," ISSCC, pp. 576-578, 2024.

[3] X. Wang et al., "14.3 A 28nm 17.83-to-62.84TFLOPS/W Broadcast-Alignment Floating-Point CIM Macro with Non-Two's-Complement MAC for CNNs and Transformers," ISSCC, pp. 254-256, 2025.

[4] Y. Yuan et al., "14.5 A 28nm 192.3TFLOPS/W Accurate/Approximate Dual-Mode-Transpose Digital 6T-SRAM CIM Macro for Floating-Point Edge Training and Inference," ISSCC, pp. 258-260, 2025.

[5] Z. Yue et al., "14.4 A 51.6TFLOPs/W Full-Datapath CIM Macro Approaching Sparsity Bound and <2-30 Loss for Compound AI," ISSCC, pp. 1-3, 2025.

[6] Y. Wang et al., "LLM-CIM: A 28nm 126.7TOPS/W Input-LUT-Based Digital CIM Macro with Reconfigurable Matrix Multiplication and Nonlinear Operation Modes for LLMs," VLSI, pp. 1-3, 2025.

[7] Chenggang Zhao, et al., "Insights into DeepSeek-V3: Scaling Challenges and Reflections on Hardware for AI Architectures," ISCA, pp 1731–1745, 2025.