DM-CIM: A 28nm SRAM Compute-in-Memory Macro with Dynamic Microscaling and Flexible Mixed-Precision MAC

Xiaofeng Hu[1], HanGyeol Mun[1], Yuan Liao[1], Yifan He[1], Arnab Raha[2], Amit Agarwal[2], Mark Anders[2], Jae-sun Seo[1]
Email: {xh387, js3528}@cornell.edu
[1]School of ECE, Cornell Tech, New York, NY, USA [2]Intel Corporation, Hillsboro, OR, USA

***Abstract*—This paper presents DM-CIM, a 28nm digital SRAM compute-in-memory macro supporting dynamic microscaling formats, featuring dynamic pre-alignment, group-wise mixed precision, and a mixed AS-OS dataflow. Fabricated in 28nm, the DM-CIM chip advances accuracy-efficiency Pareto frontier, achieves 45–48% higher energy and area efficiency than prior CIM works with comparable or better accuracy, and narrows the macro-to-system energy gap by 1.61–2.02×.**

***Index Terms*—AI, compute-in-memory (CIM), microscaling, dynamic pre-alignment, mixed precision**

I. Introduction

SRAM compute-in-memory (CIM) offers notable energy efficiency and standard CMOS compatibility [1–6]. However, large language models (LLMs) pose two major challenges for SRAM CIM (Fig. 1, top): (1) **Accuracy vs. efficiency:** Floating-point CIMs (FP-CIM) preserve high accuracy but rely on costly post-alignment [1–4], which increases MAC bitwidth and degrades efficiency. Conversely, pre-alignment [5] with microscaling (MX) formats reduce alignment and MAC costs but suffer truncation-induced accuracy loss. This issue is especially pronounced in transformer models with outliers, which are difficult to quantize yet remain important for accuracy. In general, conventional static pre-alignment methods fail to handle outliers effectively, causing large accuracy loss, particularly at low precisions (e.g., 4b). (2) **Macro-system energy gap:** Conventional weight-stationary (WS) or activation-stationary (AS) dataflows yield limited system-level gains, as weight activation memory consumes <2% of system energy while partial sum (Psum) traffic dominates. Output-stationary (OS) designs with local Psum memory [6] mitigate large Psum access overhead, but introduce expensive local accumulators that degrade MAC energy and offset system-level savings.

To address these challenges, we propose DM-CIM (Fig. 1, bottom) that features: (1) a dynamic microscaling (DM-INT) format implemented through 3-stage dynamic pre-alignment (DPA) that adaptively adjusts preserved bits for group-wise pre-alignment; (2) a reconfigurable DM-CIM macro efficiently supporting group-wise mixed-precision MACs; and (3) a mixed AS-OS dataflow with both local Psum and activation reuse. Fabricated in 28nm, DM-CIM chip achieves 45-48% Figure-of-Merit (FoM: TOPS/W × TOPS/mm^2) improvement over prior art with comparable or better accuracy result.

This work was supported in part by NSF under grant 2403723, and SRC/DARPA JUMP 2.0 CoCoSys and ACE centers.

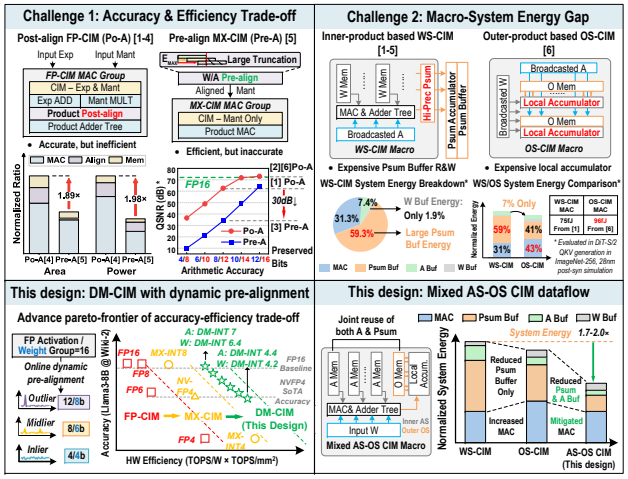

Fig. 1: Challenges of conventional CIM designs (top) and overview of the proposed DM-CIM design (bottom).

II. DM-CIM Architecture and Proposed Methods

A. Dynamic Microscaling with Dynamic Pre-alignment

Conventional FP-CIM and MX-CIM statically assign identical bit-widths during group-wise alignment. However, transformer models exhibit significant variance in group-wise inner products. Profiling the Llama3-3B model on Wikitext-2 dataset (Fig. 2(a), left) reveals that the top 5% of inner-product groups contribute >90% of the final product magnitude. Thus, uniform precision is suboptimal, motivating DM-INT to dynamically allocate precision based on group importance.

To predict group-wise product importance, we propose using the input-activation group-wise scaling factor—obtained during pre-alignment—as a lightweight proxy. This is motivated by the strong correlation between the activation scaling factor A_{Scale}, and the group product magnitude. Fig. 2(a) shows that the top 10% of the group product and A_{Scale} exhibit strong spatial overlap. Quantitatively, A_{Scale} achieves a Pearson/Spearman correlation of 0.86/0.64 with the group product, while the weight scaling factor W_{Scale} shows little

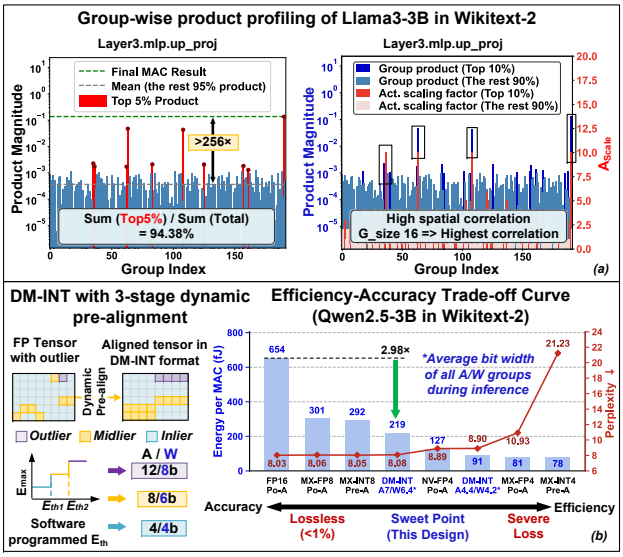

Fig. 2: DM-INT format with efficiency-accuracy trade-off.

correlation. This observation is also consistent with the rationale of prior LLM quantization works [7, 8]. We further observe that a smaller group size strengthens this correlation, which led to our choice of a relatively small group size of 16.

With these observations and design choices, we propose DM-INT, as an enhanced MX-INT scheme with group-wise mixed precision (Fig. 2(b)). We conduct DM-INT quantization online through the DPA module in a 3-stage scaling method. DPA quantizes FP activation and weight vectors to 12b/8b/4b and 8b/6b/4b, respectively, by comparing the 5-bit maximum exponent (E_{max}) against preset thresholds (E_{th}). Activations are pre-aligned and stored in DM-CIM memory first, followed by weight alignment according to its corresponding activation group's precision. Activations receive higher bit-widths due to their higher quantization sensitivity. For Llama3-3B and Qwen2.5-3B models on Wikitext-2 dataset, DM-INT (A7/W6.4) maintains $<1\%$ perplexity loss versus FP16 with $2.98\times$ energy saving. While MX-INT4/FP4 exhibits significant accuracy loss, DM-INT (A4.4/W4.2) preserves $<10\%$ of critical groups in higher precision, matching NV-FP4 accuracy with $1.40\times$ lower MAC energy. More extensive accuracy results on LLM and ViT models are reported in Sec. III-A.

B. DM-CIM Macro with Group-wise Mixed-Precision Support

To support the proposed DM-INT format, we designed the DM-CIM macro tailored for this 3-stage group-wise mixed precision. As shown in Fig. 3, instead of weight-stationary (WS) dataflow, we employ activation-stationary (AS) dataflow for two primary reasons. First, as we discussed in Sec. II-A, we choose MAC precision according to activation group scaling factor, and weight precision is selected accordingly. This means that the activation precision is determined once and kept fixed, but the same weight group can possibly be scaled to different precision. AS mapping leads to a more regulated and tight memory floorplan. Second, in matrix-vector multiplication workload (GEMV) with small input batch (e.g.,

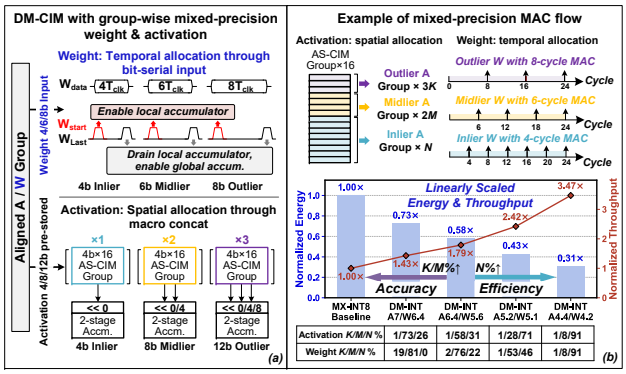

Fig. 3: DM-CIM macro's mixed-precision MAC flow.

edge LLM reasoning), weight reuse is limited. WS dataflow suffers from frequent weight update and struggles to achieve high hardware utilization. Instead, AS dataflow can benefit from activation reuse and reduce the memory update overhead. Therefore, we adopt AS to efficiently support GEMV on top of the GEMM workloads that are inherently efficient with CIMs.

Upon receiving a MAC workload, DM-CIM dynamically pre-aligns the activation groups to 4b/8b/12b, splitting them into 4b×16 chunks stored across 1/2/3 AS-CIM memory groups. Weights are subsequently fetched, pre-aligned, and fed bit-serially (Fig. 3(a)). Since the weight precision varies per inner-product group, standard input broadcasting is sub-optimal. Broadcasting input aligns compute latency to a few longest-cycle outlier groups, forcing the vast majority of lower-precision groups to be idle, resulting in limited throughput improvement [3].

To fully boost DM-CIM's system throughput with fine-grained group-wise mixed precision, we propose to use the unicast input method. Each group receives an independent input bit-serial vector with a pair of control signals: W_{start} and W_{last}. W_{start} initiates a new MAC by resetting the local shifting accumulator, while W_{last} marks the weight vector's end, draining the local result to the global 2-stage accumulator for Psum merging. W_{start} and W_{last} orchestrate each group's bit-serial MAC in an asynchronous way. This asynchronous control eliminates inter-group idle time, linearly scaling system throughput under a given precision budget (Fig. 3(b)). Unicasting input can cause large routing pressure in conventional bit-parallel designs such as systolic arrays, but such overhead is mitigated through bit-serial MAC, making it an ideal fit for bit-serial CIM design. Owing to spatial activation allocation and temporal weight allocation, DM-CIM flexibly supports arbitrary precision budgets, delivering near-ideal linear scaling in both energy efficiency and throughput across different A/W precision settings (Fig. 3(b), bottom).

C. AS-OS Mixed Dataflow with Activation/Psum Reuse

Conventional WS-CIM/AS-CIM designs gain limited benefit from CIM's efficient local memory [1–5], as the stored weights/activations are read out only once and reused for all input channels/tokens (Fig. 4(a)). This read-once-only MAC flow maximizes the macro-wise peak energy efficiency, but large

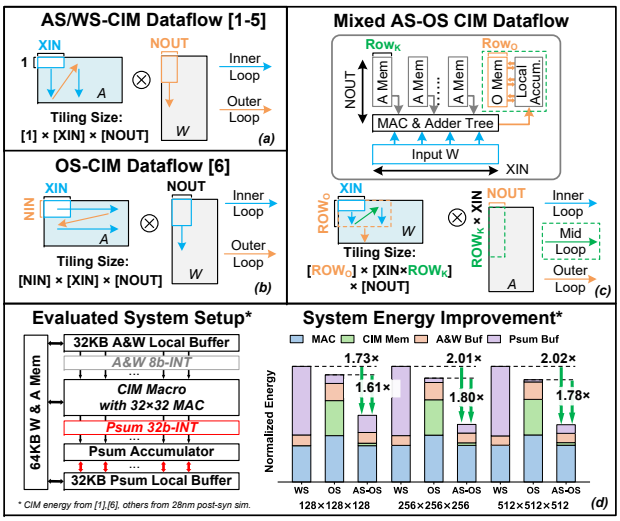

Fig. 4: Mixed AS-OS dataflow and system energy evaluation.

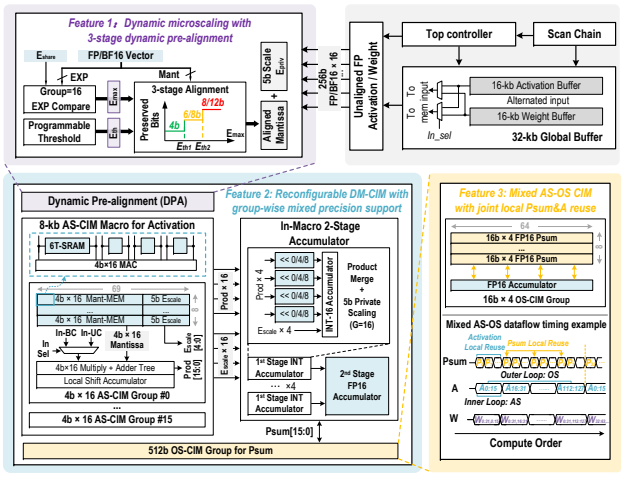

Fig. 5: Overall architecture of proposed DM-CIM system.

Psum buffer energy heavily degrades the system-wise energy efficiency. Recent outer-product-based OS work [6] attempts to address this by storing and accumulating Psum locally within CIM memory (Fig. 4(b)). While effectively reducing the Psum buffer energy, the required expensive local accumulators inflate MAC energy, offsetting system-level improvements.

To bridge this macro-to-system energy gap and fully unlock the potential benefit of CIM's efficient local memory, we propose a mixed AS-OS dataflow (Fig. 4(c)) that keeps both activations and outputs in local CIM memory and jointly reuses them. In the inner loop, a fetched activation remains stationary while O-Mem Psums are read row-by-row for accumulation. Before draining O-Mem, we introduce a "Mid Loop" (marked by green arrow in Fig. 4(c)) that iterates through the activation memory and repeats the inner loop, ensuring the joint reuse of both local activations and Psum. This effectively expands the tiling size by a factor of Row_K and Row_O, directly replacing the expensive global buffer accesses with highly efficient local CIM memory read/write operations.

To evaluate the proposed dataflow, we conducted a system-wise iso-throughput energy evaluation by integrating a generic CIM macro with 32KB input/output local buffer and 64KB of shared memory, maintaining a memory hierarchy aligned with a single Nvidia H100 Tensor core, targeting GEMM workloads. The detailed configurations are illustrated in Fig. 4(d). Results across three GEMM workloads show WS-CIM suffers a large macro-to-system energy gap due to high Psum buffer access energy. OS-CIM reduces this via local Psum reuse, but its increased MAC energy offsets the savings. By jointly reusing Psums and activations locally without inflating MAC energy, our AS-OS CIM effectively reduces global buffer accesses and achieves a 1.61–2.02× system-wise energy improvement over conventional WS and OS baselines.

D. DM-CIM System Overview

Fig. 5 presents the overall DM-CIM system, consisting of a dynamic pre-alignment (DPA) module, eight 9-kb SRAM DM-CIM macros that support dynamic microscaling MAC, 32-kb global buffer, and a top controller. The DPA module receives activation and weight vectors in FP format, finds the maximum value for each group, and performs the 3-stage DPA. Based on the calculated group-wise scaling factor E_{scale} and software-programmed thresholds E_{th}, FP vectors are aligned to 4/6/8/12b and dispatched to the macros alongside their scaling factors for computation.

Each macro contains 16 AS-CIM groups for activations and one OS-CIM group for local partial sum (Psum) accumulation. One AS-CIM group has 8×69b 6T SRAM cells: 64b for the 4b×16 aligned mantissa bit trunk and 5b for the scaling factor (E_{scale}). A shared 16-input bit-serial adder tree computes the inner product, locally aggregated via a shifting accumulator. Internal 2-stage FP accumulation minimizes Psum traffic, while the OS-CIM group allocates an extra 8×64b SRAM for local Psum reuse. The 2-stage accumulator consists of four first-stage INT16 accumulators and one second-stage FP16 accumulator, operating in a coarse-to-fine manner. The FP16 stage is activated only when a first-stage INT16 accumulator overflows to reduce dynamic power, after which the corresponding INT16 result is reset for continued accumulation.

TABLE I: Accuracy results on Llama3, Qwen2.5 and ViT models with DM-INT scheme.

		Wikitext-2 Perplexity(↓)		ImageNet-1K Accuracy(↑)
W/A Precision		Llama-3 3B/8B	Qwen-2.5 3B/7B	ViT-Small/Base
(1) Accuracy Loss <1%	FP16 Baseline	7.81 / 6.24	8.03 / 6.84	81.38 / 85.10
	MX-FP A8/W8	7.85 / 6.27	8.06 / 6.87	81.31 / 85.05
	MX-INT A8/W8	7.83 / 6.25	8.05 / 6.86	81.35 / 85.11
	DM-INT A7/W6.4	**7.86 / 6.29**	**8.08 / 6.88**	**81.33 / 85.07**
(2) A6-W6	MX-INT A6/W6	7.96 / 6.37	8.29 / 6.95	81.19 / 85.01
	MX-FP A6/W6	8.48 / 6.39	8.24 / 6.96	79.69 / 84.52
	DM-INT A5.2/W5.1	**7.99 / 6.39**	**8.36 / 6.96**	**80.70 / 84.75**
(3) A4-W4	MX-INT A4/W4	10.57 / 7.89	21.23 / 8.02	74.84 / 82.79
	MX-FP A4/W4	9.98 / 7.98	10.93 / 8.55	63.45 / 78.16
	NV-FP A4/W4	8.63 / 6.88	8.89 / 7.29	79.78 / 84.50
	DM-INT A4.4/W4.2	**8.61 / 6.73**	**8.90 / 7.21**	**79.26 / 84.37**

TABLE II: Comparison to prior CIM works.

	ISSCC24 34.2 [2]	ISSCC25 14.4 [1]	ISSCC25 14.3 [5]	CICC25 30.1 [6]	ISSCC26 30.1 [3]	DM-CIM (This Work)		
Technology	16nm	28nm	28nm	28nm	28nm	28nm		
CIM Type	WS	WS	WS	OS	WS	**Mixed AS-OS**		
Voltage (V)	0.4~0.8	0.62~0.9	0.55~0.9	0.45~0.9	0.55~0.9	0.53~0.9		
Accumulator	No	In-Macro FP	No	In-Macro INT	No	**In-Macro INT-FP 2-stage**		
MAC Type	Post-Align FP	Post-Align FP	Pre-Align FP	INT	Post-Align FP	**Dynamic Pre-Align FP**		
A/W Precision	BF16/BF16	FP16/FP16	BF16 aligned to MX-INT8/INT8	INT8/INT8	MXFP8/MXFP8	**Programmable DM-INT**		
						A7/W6.4	**A5.2/W5.1**	**A4.4/W4.2**
Macro TOPS/W [1]	33.2~45.4	15.1~51.6	17.8~62.8	20.9~137.2	49.53	**21.4**	**37.5**	**52.8**
Macro TOPS/mm²	2.63 (0.8V)	0.46~2.44 (0.9V)	0.70 (0.9V)	0.30~1.60 (0.9V)	1.24 (0.9V)	**1.43** (0.9V)	**2.42** (0.9V)	**3.48** (0.9V)
FoM [2]	119.4	125.9	44.0	219.5	61.4	**30.6**	**90.8**	**183.7**
System TOPS/W [3]	-	4.8~16.2	-	6.7~40.2	-	**12.06**	**20.84**	**29.48**
ImageNet-1K Accuracy (↑)	68.33% (-0.02) @ ResNet-18	79.05%(-1.52) @ ViT-B	79.82% (-0.01) @ DeiT-S	78.77% (-1.22) @ ViT-S	-	**85.07% (-0.03)** @ ViT-B	**84.75% (-0.35)** @ ViT-B	**84.37% (-0.73)** @ ViT-B
Wikitext-2 Perplexity (↓)	-	-	-	9.54 (+0.25) @Llama3-1B	8.87 (+0.15) @ Llama3-8B	**6.29 (+0.05)** @Llama3-8B	**6.39 (+0.15)** @Llama3-8B	**6.88 (+0.64)** @Llama3-8B

1. [1] uses 1:2 structured weight sparsity, [2] uses 80% input sparsity and 10% input toggle rate, [6] uses 1:2 to 3:16 structured weight sparsity. DM-CIM uses real, dense vector from llama3-8B in Wikitext-2 inference. TOPS/W measured @ 0.53V, 127MHz, TOPS/mm² measured @ 0.9V, 693MHz. 2. FoM: Peak TOPS/W × Peak TOPS/mm².
3. Full-chip energy, including accumulator, on-chip A/W/Psum buffer & controller.

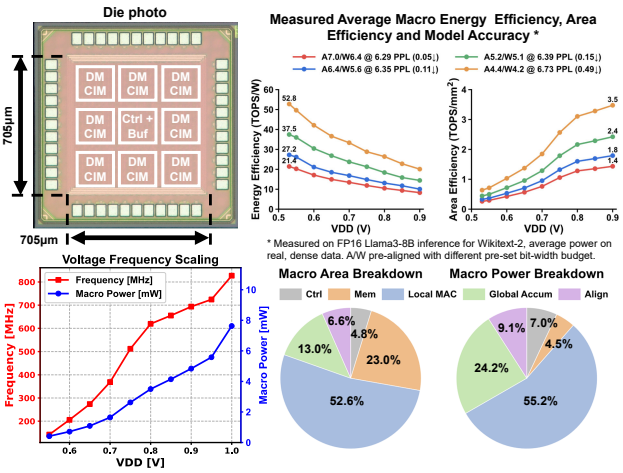

Fig. 6: Die photo and chip measurement results.

III. Results

A. DM-INT Accuracy Results

Table I demonstrates DM-INT's effectiveness and flexibility across LLMs and ViTs through three precision configurations with different priority: **(1) Accuracy**: DM-INT A7/W6.4 maintains $<1\%$ accuracy loss versus FP16, while reducing bit-wise computation by $1.33-2.98\times$ compared to MX-FP/INT8 and FP16. **(2) Balance**: DM-INT A5.2/W5.1 optimally trades off accuracy and efficiency, consistently matching or exceeding MX-FP6 and MX-INT6 accuracy. **(3) Efficiency**: DM-INT A4.4/W4.2 maximizes throughput and efficiency with near-4b MACs. It effectively recovers the severe accuracy degradation of MX-INT4/FP4 via group-wise mixed precision, matching the strong NV-FP4 baseline accuracy while using compact INT-only MACs (resulting in $1.40\times$ lower MAC energy).

B. Chip Measurement Results

Fabricated in 28nm CMOS (Fig. 6), the DM-CIM system comprises eight 9-kb DM-CIM macros and a 32-kb SRAM buffer. Operating at 0.53–1.0V and 127–867MHz, it achieves 21.4–52.8 TOPS/W at 0.53V and 1.4–3.5 TOPS/mm^2 at 0.9V across A4.4/W4.2 to A7.0/W6.4 DM-INT precisions. The architecture flexibly adapts to varying precision configurations, demonstrating near-linear area and energy efficiency scaling.

Table II shows the comparison to prior art. Compared to [1] (1.52% ViT-Base accuracy loss with 1:2 sparsity), our dense A4.4/W4.2 configuration limits the accuracy drop to 0.73% and improves FoM by 45%. While [5] achieves high macro-level efficiency via 80% input sparsity, it neglects system-level energy overheads and lacks precision flexibility. Compared to [3], DM-CIM (A5.2/W5.1) achieves $1.95\times$ higher TOPS/mm^2 and 47.8% FoM improvement at identical 0.15 perplexity drop for Llama3-8B. Overall, DM-CIM not only achieves state-of-the-art FoM, but also delivers superior system-to-macro efficiency and highly flexible precision scaling.

IV. Conclusion

This paper proposes DM-CIM that features dynamic microscaling, flexible mixed-precision MAC, and mixed AS-OS dataflow. Prototyped in 28nm CMOS, DM-CIM effectively minimizes alignment overhead via DPA, advances the accuracy-efficiency Pareto frontier, and substantially bridges the macro-to-system energy gap of SRAM CIM designs.

References

[1] Z. Yue et al., "A 51.6TFLOPs/W Full-Datapath CIM Macro Approaching Sparsity Bound and $< 2^{-30}$ Loss ...," in *ISSCC*, 2025.

[2] W.-S. Khwa et al., "A 16nm 96Kb Integer/Floating-Point Dual Mode Gain Cell Computing in Memory ...," in *ISSCC*, 2024.

[3] X. Wang et al., "A 28nm 127.54TFLOPS/W MXFP6 and 117.42TFLOPS/W MXFP8 ...," in *ISSCC*, 2026.

[4] H. Mori et al., "A 3nm 125 TOPS/W-29 TFLOPS/W, 90 TOPS/mm^2-17 TFLOPS/mm^2 SRAM-Based ...," in *Symp. VLSI*, 2025.

[5] X. Wang et al., "A 28nm 17.83-to-62.84TFLOPS/W Broadcast-Alignment Floating-Point CIM Macro ...," in *ISSCC*, 2025.

[6] X. Hu et al., "A 28nm 20.9-137.2 TOPS/W Output-Stationary SRAM Compute-in-Memory Macro Featuring ...," in *CICC*, 2025.

[7] J. Lin et al., "AWQ: Activation-aware Weight Quantization for On-Device LLM Compression and Acceleration," in *MLSys*, 2024.

[8] C. Lee et al., "OWQ: Outlier-Aware Weight Quantization for Efficient Fine-Tuning and Inference of Large Language Models," in *AAAI*, 2023.