14.3 A 28-nm 17.83-62.84 TFLOPS/W Broadcast-Alignment Floating-Point CIM Macro with Non-2's-Complement MAC for CNNs and Transformers

Xing Wang*[1,2], Tianhui Jiao*[1], Yi Yang[1], Shaochen Li[1], Dongqi Li[1], An Guo[1], Yuhui Shi[1], Yuchen Tang[1], Jinwu Chen[1], Zhican Zhang[1], Zhichao Liu[1], Bo Liu[1], Weiwei Shan[1], Xin Wang[3], Hao Cai[1], Wenwu Zhu[3], Jun Yang[1,2], Xin Si[1]

[1]Southeast University, Nanjing, China, [2]National Center of Technology Innovation for EDA, Nanjing, China, [3]Tsinghua University, Beijing, China

The rapid advancement of artificial intelligence (AI) models has increased the demand for high-precision, energy-efficient AI edge chips. The support of floating-point (FP) processing is essential for high-precision neural network (NN) training and inference. Nevertheless, it incurs higher energy and area overhead due to complex FP multiplication and accumulation (MAC) operations. Digital compute-in-memory (DCIM) and floating-point CIM (FP-CIM) [1-10] have emerged as a promising technique to enhance energy efficiency with higher accuracy. Previous FP-CIM implementations [1-7] have achieved good performance through various alignment schemes and computing processes. However, as illustrated in Figure 14.3.1, implementing digital-domain FP-CIM faces several challenges: (1) the difficulty of balancing FP-computation precision and input reusability, as alignment operations are unfriendly to CIM structure; (2) large performance loss or area overhead due to peripheral parallel-alignment schemes; and (3) huge dynamic energy consumption of digital MAC circuits induced by low sparsity of 2's complement (2C) negative weights, coupled with additional sign bit computation overhead in digital CIM. This work presents a hierarchical Broadcast-Alignment-Non-2's Complement MAC (B-A-N2CMAC) true FP-CIM macro featuring (1) a broadcast input and embedded light-convertor structure to enable BF16/INT8 MAC operations with improved input reusability; (2) an embedded area-efficient adaptive-alignment scheme with dual-bit-serial MAC; and (3) a format-mixed N2CMAC flow to reduce circuit dynamic activity and signed computation overhead. A 28nm 64-kb B-A-N2CMAC true FP-CIM macro is fabricated to support FP-MAC operations with BF16 and INT8. This CIM macro achieved an energy efficiency of 62.84TFLOPS/W@BF16 and 90.15TOPS/W@INT8.

Figure 14.3.2 illustrates the overall structure of the proposed B-A-N2CMAC true FP-CIM macro, which includes 16 triple-stacked arrays (TS-As), a hierarchical input buffer, a calibrated accumulator & quantizer (CA&Q), and peripheral circuits. Each TS-A computes one output channel and comprises (1) an embedded serial input convert unit (ESICU), (2) a 16-row × 256-column 6T SRAM array, and (3) a format-mixed non-2's complement MAC unit (N2CMACU). Unlike previous true FP-CIM designs [2,3], global floating-point/integer inputs (Gin) from hierarchical input buffer can be reused by 16 TS-As to maximize the array utilization rate. Each Gin consists of an 8-bit parallel exponent input (Ein) and a 2-bit serial mantissa input (Min). ESICU, which contains an align-signal generator (ASG) and 16 light-converters (LCs), is designed to achieve the local 2b-serial alignment of Gin according to the readouts of exponent data from the SRAM array. This approach eliminates the need for area-hungry subtractors and barrel shifters, compared to traditional digital alignment methods [1,2,5]. To fully utilize the sparsity of negative weights with small absolute values, an N2CMAC computation flow is proposed to enable weight-mapping with sign-magnitude data format. In this flow, a signed-floating-point/integer MAC is divided into an unsigned MAC and a compensation performed in TS-A and CA&Q, respectively. The CIM macro supports four modes: 1 mode for conventional read/write, and 3 modes for computation (BF16A, BF16B, and INT8). BF16A and BF16B modes differ from the preserved bit-width of the aligned mantissa, 10bits for BF16A and 8bits for BF16B, which are sufficient to handle the target CNN/Transformer networks.

Figure 14.3.3 illustrates the structure of ESICU and the embedded adaptive 2b-serial alignment scheme. Inside ESICU, the ASG consists of 16 exponent adders (EAs), a maximum value finder (MVF), a shift destination tracer (SDT), and 16 equality comparators (ECs). The LC consists of an adaptive-reset register chain (ARRC), a self-alignment controller (SAC), and an output select & inverse unit (OSIU). After the generation of exponent sum and maximum sum value through EAs and MVF, each EC compares the SDT value with the corresponding exponent sum (SUM[8:1]) and then generates shift control signals (Shift0-15) for the LCs. The SDT value is first determined by the maximum sum, and then it decreases by 1 each cycle. When signal Shift = 0, the ARRC is in the “Store” state, sequentially storing the serial Min values; when Shift = 1, the ARRC switches to the “Shift” state, sequentially outputting the previously stored Min values (shifting two bits per cycle). The OSIU then generates OUT[1:0] and registers it as local input Lin[1:0] for N2CMAC operations. The deployment of the proposed adaptive 2b-serial alignment scheme can reduce area overhead by 36.23%, compared to traditional subtractor and barrel shifter circuits-based alignment schemes [1-6].

Figure 14.3.4 illustrates the detailed implementation of format-mixed N2CMAC computation flow, along with the memory bank configuration and waveform for 3 computation modes. Unlike the previous 2's complement-based MAC operations [11-13], weights are stored in the TS-As with sign-magnitude format. Format-mixed N2CMAC computation flow computes MAC results of 2's complement inputs and sign-magnitude weights, with the output in 2's complement form. The computation flow consists of 4 steps. Firstly, the hierarchical input buffer provides Gin, including 8b parallel Exp (BF16A/B mode) and 2b serial Man/IN (BF16A/B and INT8 mode). Secondly, the LC serially converts Gin, performing serial alignment (BF16 mode) and input sign inversion (BF16/INT8 mode). Thirdly, the digital multiply unit (DMU) multiplies Lin with the sign-magnitude weight, and the results are accumulated in the channel-wise adder tree (CAT). Finally, the CA&Q then sums the output of CAT over multiple cycles to generate PMACV, which is calibrated by adding the pre-set compensation value in the accumulator to obtain MACV (20b for INT8, 23b for BF16A, 21b for BF16B). Because the sign bit of the 2's complement Lin[m-1:0] is inversed, converting it into an m-bit unsigned number ({~Lin[m-1], Lin[m-2:0]} = Lin[m-1:0] + {1,(m-1)'b0}), which is then used in the MAC operation with weight magnitude, the overhead of sign-bit computation is therefore reduced. Furthermore, since the compensation value is only determined by weights, a 23b compensation value can be computed by two additional cycles or pre-set offline and shared across multiple cycles. In this work, memory bank configuration is achieved through the MUX shared by adjacent columns. In BF16 mode, a staggered mapping scheme is used for exponents and mantissa, while in INT8 mode, the column select (CS) functions as a column decoder, increasing the storage-compute ratio (SCR).

Figure 14.3.5 presents the performance of the proposed schemes. The serial alignment and serial MAC scheme proposed in this work achieves 3.83×, 1.56× reductions in area overhead and 1.71×, 1.20× reductions in power consumption compared to previous parallel alignment with parallel MAC [6] and parallel alignment with serial MAC schemes [1,2]. The proposed N2CMAC computation flow saves computing cycles in different networks with varying input precisions, a 19.80%/16.50% reduction for 8b/10b input on the ViT (DeiT-S) @ImageNet. Due to the utilization of sign-magnitude format weights, the sparsity of negative INT8 weights is significantly improved, showing a 1.83× increase in ResNet50 compared to the 2's complement format. The format-mixed N2CMAC circuit designed for this computation flow achieves 1.96×, 1.25×, and 1.18× lower power consumption, latency, and area overhead compared to traditional 2’s complement MAC circuits.

Figure 14.3.6 presents the measured test chip results, fabricated using a 28nm CMOS technology via a 64kb B-A-N2CMAC true FP-CIM macro. For MAC operation in BF16B mode, the measured MAC access time was 5.4ns @0.9V for BF16 inputs and weights, with FP32 outputs. The maximum energy efficiency and area efficiency are measured to be 62.84TFLOPS/W and 697.17GFLOPS/mm^2 for BF16 MAC operations from a 0.55-0.9V supply running ResNet50 @ImageNet. Compared to prior SRAM CIM macros [1-5], this work improves the IN × W × memory density × Norm. energy efficiency × output ratio FoM by 1.93 – 116.90×. This work can achieve 79.818% inference accuracy when employing ViT @ImageNet. Figure 14.3.7 shows the die photo and chip summary table. All input/weight sparsity and toggle rates are bit-level and all simulation and measurements are obtained at room temperature (300K).

Acknowledgement:

This work was supported in part by the National Science and Technology Major Project under Grant 2022ZD0118902, in part by the National Natural Science Foundation of China under Grant 92264203 and Grant 62204036, in part by the Key Research and Development Program of Jiangsu Province under Grant BE2023020-1, in part by the Postgraduate Research and Practice Innovation Program of Jiangsu Province under Grant KYCX23_0344, and in part by the Fundamental Research Funds for the Central Universities under Grant 2242022k60009. The corresponding author is Xin Si (xinsi@seu.edu.cn).

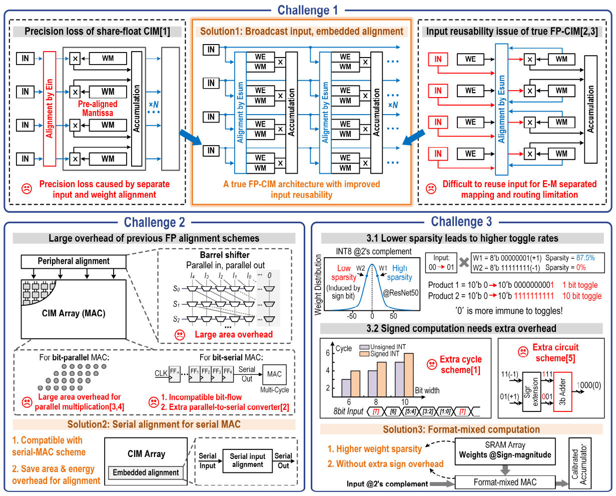

Challenge 1
Precision loss of share-float CIM[1]
Solution1: Broadcast input, embedded alignment
Input reusability issue of true FP-CIM[2,3]
Precision loss caused by separate input and weight alignment
A true FP-CIM architecture with improved input reusability
Difficult to reuse input for E-M separated mapping and routing limitation
Challenge 2
Large overhead of previous FP alignment schemes
Peripheral alignment
Barrel shifter
Parallel in, parallel out
CIM Array (MAC)
Large area overhead
For bit-parallel MAC:
For bit-serial MAC:
Large area overhead for parallel multiplication[3,4]
1. Incompatible bit-flow
2. Extra parallel-to-serial converter[2]
Solution2: Serial alignment for serial MAC
1. Compatible with serial-MAC scheme
2. Save area & energy overhead for alignment
CIM Array
Embedded alignment
Serial input alignment
Challenge 3
3.1 Lower sparsity leads to higher toggle rates
INT8 @2's complement
Low sparsity
High sparsity
'0' is more immune to toggles!
3.2 Signed computation needs extra overhead
Extra cycle scheme[1]
Extra circuit scheme[5]
Solution3: Format-mixed computation
1. Higher weight sparsity
2. Without extra sign overhead
SRAM Array Weights @Sign-magnitude
Input @2's complement
Format-mixed MAC

Figure 14.3.1: Design challenges of FP-CIM, and solutions of this proposed work.

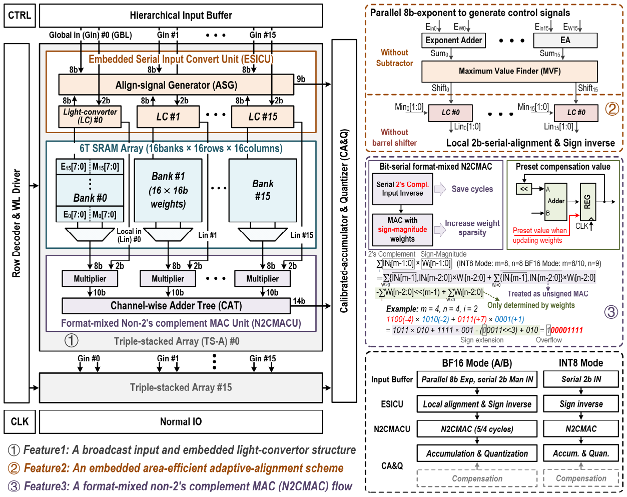

CTRL
Hierarchical Input Buffer
Row Decoder & WL Driver
Embedded Serial Input Convert Unit (ESICU)
Align-signal Generator (ASG)
Light-convertor (LC) #0
LC #1
LC #15
6T SRAM Array (16banks × 16rows × 16columns)
Bank #0
Bank #1 (16 × 16b weights)
Bank #15
Multiplier
Channel-wise Adder Tree (CAT)
Format-mixed Non-2's complement MAC Unit (N2CMACU)
Triple-stacked Array (TS-A) #0
Triple-stacked Array #15
Calibrated-accumulator & Quantizer (CA&Q)
CLK
Normal IO
Parallel 8b-exponent to generate control signals
Exponent Adder
Maximum Value Finder (MVF)
Local 2b-serial-alignment & Sign inverse
Bit-serial format-mixed N2CMAC
Preset compensation value
BF16 Mode (A/B)
INT8 Mode
Input Buffer
Parallel 8b Exp, serial 2b Man IN
Serial 2b IN
ESICU
Local alignment & Sign inverse
Sign inverse
N2CMACU
N2CMAC (5/4 cycles)
N2CMAC
CA&Q
Accumulation & Quantization
Accum. & Quan.
Compensation
① Feature1: A broadcast input and embedded light-convertor structure
② Feature2: An embedded area-efficient adaptive-alignment scheme
③ Feature3: A format-mixed non-2's complement MAC (N2CMAC) flow

Figure 14.3.2: Overall structure and key features of proposed B-A-N2CMAC true FP-CIM macro.

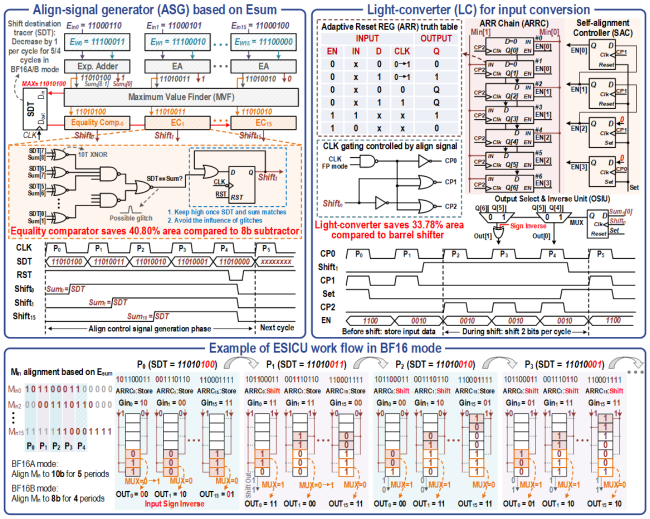

Align-signal generator (ASG) based on Esum
Maximum Value Finder (MVF)
Equality comparator saves 40.80% area compared to 8b subtractor
Light-converter (LC) for input conversion
Adaptive Reset REG (ARR) truth table
ARR Chain (ARRC)
Self-alignment Controller (SAC)
CLK gating controlled by align signal
Output Select & Inverse Unit (OSIU)
Light-converter saves 33.78% area compared to barrel shifter
Align control signal generation phase
Next cycle
Before shift: store input data
During shift: shift 2 bits per cycle
Example of ESICU work flow in BF16 mode
BF16A mode: Align Mₙ to 10b for 5 periods
BF16B mode: Align Mₙ to 8b for 4 periods
Input Sign Inverse

Figure 14.3.3: Detailed schematic and work flow of ESICU.

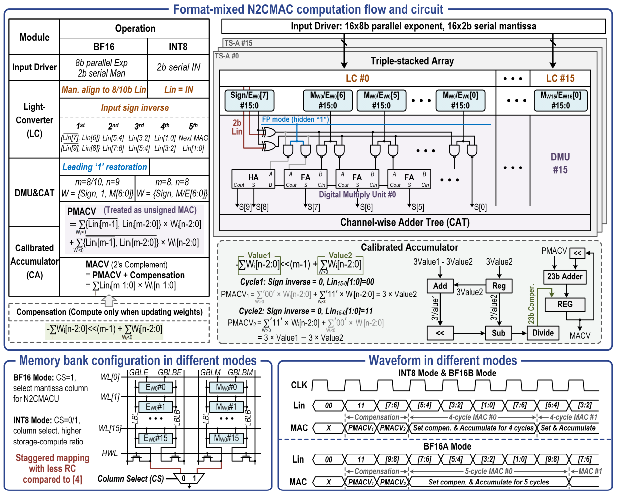

Format-mixed N2CMAC computation flow and circuit
Module
Operation
BF16
INT8
Input Driver
8b parallel Exp 2b serial Man
2b serial IN
Light-Converter (LC)
Man. align to 8/10b Lin
Lin = IN
Input sign inverse
Leading '1' restoration
DMU&CAT
PMACV (Treated as unsigned MAC)
Calibrated Accumulator (CA)
MACV (2's Complement)
= PMACV + Compensation
Compensation (Compute only when updating weights)
Input Driver: 16x8b parallel exponent, 16x2b serial mantissa
Triple-stacked Array
Channel-wise Adder Tree (CAT)
Digital Multiply Unit #0
DMU #15
Calibrated Accumulator
Memory bank configuration in different modes
Staggered mapping with less RC compared to [4]
Column Select (CS)
Waveform in different modes
INT8 Mode & BF16B Mode
BF16A Mode

Figure 14.3.4: Implementation of format-mixed N2CMAC computation flow, and memory bank configuration & waveform for different modes.

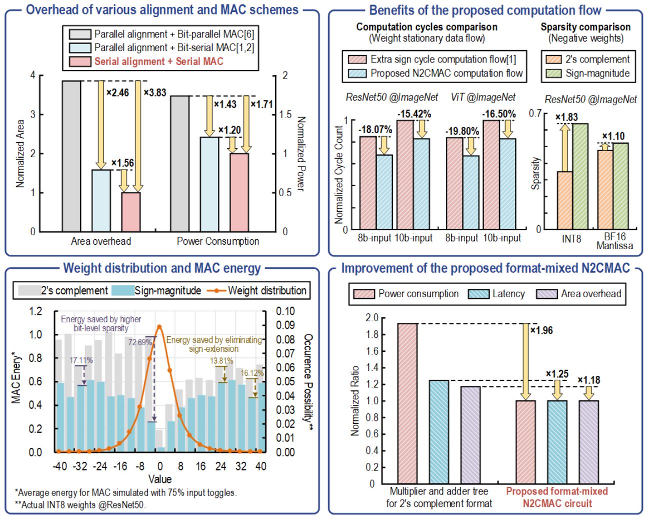


Figure 14.3.5: Simulated performance of the proposed CIM macro.

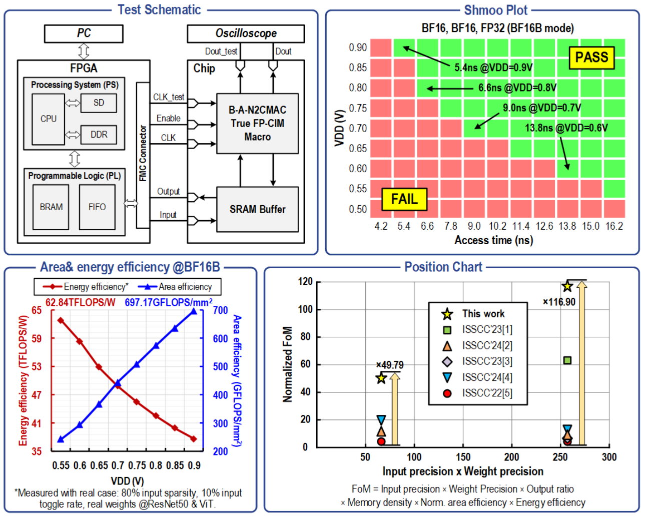
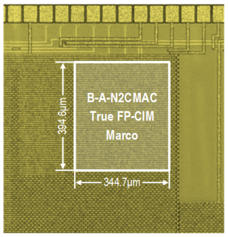


Figure 14.3.6: Measurement results and FoM comparison to prior works.



CHIP SUMMARY		
Technology	28nm CMOS	
Cell structure	6T+ESICU&N2CMAC	
Macro size	64Kb	
Macro area (mm^2)	0.136	
Input precision (bit)	8	BF16
Weight precision (bit)	8	BF16
Output precision (bit)	20	FP32
Supply voltage (V)	0.55-0.9	
Output ratio	1	
Access time (ns)	4.8	5.4/6.8
Memory Density (Kb/mm^2)	470.59	
Energy efficiency (TOPS/W) (TFLOPS/W)	[1]25.47-[2]90.15	[1]17.83-[2]62.84
Inference accuracy (%) ([3]ResNet50 @ImageNet)	76.030	76.130/76.130
Inference accuracy (%) ([4]ViT @ImageNet)	-	79.806/79.818
Inference mAP50 (%) ([5]YOLOv8 @COCO)	56.968	57.970/57.973

[1] Measured with worst case under 0.9V. Worst case: 25%input sparsity, 50%input toggle rate, 15%weight sparsity.
[2] Measured with real case under 0.55V. Real case: 80%input sparsity, 10%input toggle rate, real weights @ResNet50 & ViT.
[3] Software baseline is 76.140%
[4] Data-efficient Image Transformer-Small (DeiT-S), software baseline is 79.834%
[5] Software baseline is 57.974%

Figure 14.3.7: Die micrograph and chip summary table.

Additional References:

[1] A. 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] W. -S. Khwa et al., "A 16nm 96Kb Integer/Floating-Point Dual-Mode-Gain-Cell-Computing-in-Memory Macro Achieving 73.3-163.3TOPS/W and 33.2-91.2TFLOPS/W for AI-Edge Devices," *ISSCC*, pp. 568-570, 2024.

[3] P. -C. Wu et al., "A 22nm 832Kb Hybrid-Domain Floating-Point SRAM In-Memory-Compute Macro with 16.2-70.2TFLOPS/W for High-Accuracy AI-Edge Devices," *ISSCC*, pp. 126-128, 2023.

[4] Y. Yuan et al., "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.

[5] 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," *ISSCC*, pp. 254-256, 2022.

[6] J. Saikia et al., "FP-IMC: A 28nm All-Digital Configurable Floating-Point In-Memory Computing Macro," *ESSCIRC*, pp. 405-408, 2023.

[7] Y. Wang et al., "A 28nm 83.23TFLOPS/W POSIT-Based Compute-in-Memory Macro for High-Accuracy AI Applications," *ISSCC*, pp. 566-568, 2024.

[8] H. Fujiwara et al., "A 3nm, 32.5TOPS/W, 55.0TOPS/mm2 and 3.78Mb/mm2 Fully-Digital Compute-in-Memory Macro Supporting INT12 × INT12 with a Parallel-MAC Architecture and Foundry 6T-SRAM Bit Cell," *ISSCC*, pp. 572-574, 2024.

[9] 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," *ISSCC*, pp. 578-580, 2024.

[10] H. Mori et al., "A 4nm 6163-TOPS/W/b 4790–TOPS/mm2/b SRAM Based Digital-Computing-in-Memory Macro Supporting Bit-Width Flexibility and Simultaneous MAC and Weight Update," *ISSCC*, pp. 132-134, 2023.

[11] B. Wang et al., "A 28nm Horizontal-Weight-Shift and Vertical-feature-Shift-Based Separate-WL 6T-SRAM Computation-in-Memory Unit-Macro for Edge Depthwise Neural-Networks," *ISSCC*, pp. 134-136, 2023.

[12] A. Guo et al., "A 22nm 64kb Lightning-Like Hybrid Computing-in-Memory Macro with a Compressed Adder Tree and Analog-Storage Quantizers for Transformer and CNNs," *ISSCC*, pp. 570-572, 2024.

[13] Tai-Hao Wen et al., "Fusion of memristor and digital compute-in-memory processing for energy-efficient edge computing," *Science* 384, 325-332, 2024.