A 28nm 105.4TOPS/W Via-Programmed Digital Compute-in-ROM Accelerator With In-Cell Multiplication and Trimmed Adder Tree

Minghan Jiang*, Xinran Li*, Jianan Pan, Shuaiting Li, Haibin Shen, Kejie Huang
College of Information Science & Electronic Engineering, Zhejiang University
Hangzhou 310027, China
email: {jiangmh, lixinran, panjian_an, list, shen_hb, huangkejie}@zju.edu.cn

***Abstract*—ROM-based compute-in-memory (CIM) can keep more fixed weights on chip than SRAM-based CIM, enabling reduced DRAM accesses for edge inference. However, conventional analog compute-in-ROM relies on charge-domain accumulation and ADC-intensive peripherals, which limit robustness, parallelism, and frequency scalability. Recent digital compute-in-ROM designs alleviate the analog bottleneck, yet their MAC operations are still mainly realized by ROM-external logic. This paper presents a 28-nm via-programmed digital compute-in-ROM accelerator that embeds multiplication directly inside the cell through a two-stage isolated precharge-evaluate scheme. At the architecture level, a kernel-array organization and a selection-aware mapping scheme enable flexible layer deployment, while a statistics-guided adder-tree bit-width trimming scheme reduces the digital reduction overhead with preserved redundancy. Updating to a new model only requires via-layer mask re-instantiation together with remapping. Measurement results show a peak energy efficiency of 105.4 TOPS/W and a peak area efficiency of 5.86 TOPS/mm^2 in 28-nm CMOS.**

***Index Terms*—digital compute-in-ROM, via-programmed, in-cell multiplication, adder-tree trimming**

I. Introduction

Compute-in-memory (CIM) is an attractive approach for compact CNN inference because it reduces the dominant cost of repeated weight movement and MAC operations. Among existing CIM techniques, SRAM-based CIM benefits from mature process support and runtime flexibility, but its relatively low density limits the total on-chip weight capacity [1], [2]. As a result, SRAM-CIM accelerators often require frequent weight reloads or allocate substantial silicon area to memory arrays in practical deployments.

ROM-based CIM addresses this problem by exploiting the much higher density of ROM to keep substantially more weights on chip [3]–[7]. This is particularly beneficial for fixed-weight edge inference, where reducing off-chip parameter traffic directly improves task-level energy efficiency.

However, conventional analog compute-in-ROM still relies on charge-domain accumulation and ADC-heavy peripherals [3]–[5]. As the accumulation depth increases, ADC precision, readout energy, and reliable parallelism become strongly coupled, which limits frequency scalability and increases area/power overhead. Therefore, although analog ROM-CIM improves weight density over SRAM-CIM, its overall compute path remains constrained by analog readout complexity.

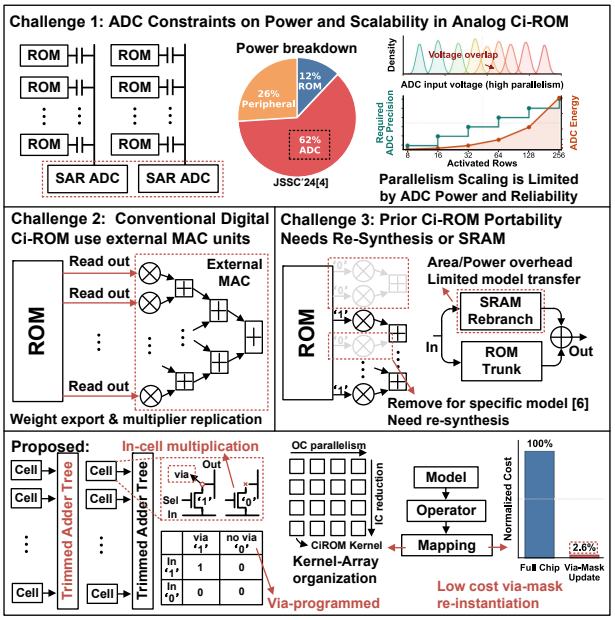

Fig. 1. Comparison of representative ROM-based CIM challenges and the proposed via-programmed digital compute-in-ROM direction.

Recent digital ROM-CIM works avoid the ADC readout bottleneck by replacing the analog readout path with digital MAC units [6], [7]. However, the ROM array and MAC logic remain largely separated, introducing additional data readout overhead and dedicated computation circuitry. Moreover, prior designs often tailor the MAC units to a fixed target model. Consequently, migrating to a new model either requires costly logic re-synthesis or relies on auxiliary SRAM to provide only limited post-fabrication flexibility.

Fig. 1 summarizes these representative challenges and motivates the proposed direction. Analog ROM-CIM is mainly constrained by ADC-heavy readout, while prior digital ROM-

This work was supported by National Natural Science Foundation of China (Grant No. 62274142), Sino-German Mobility Programme (Grant No. M-0499), and Zhejiang Province's Leading Talent Project in Science and Technology Innovation (2023R5204).

*Minghan Jiang and Xinran Li contributed equally to this work.

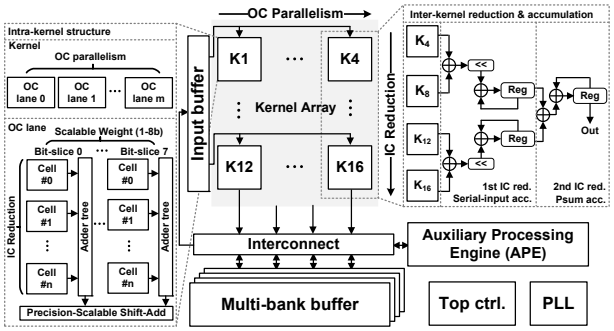

Fig. 2. Overall architecture of the proposed accelerator, including the kernel array, intra-kernel hierarchy, and inter-kernel reduction path.

CIM still mainly relies on ROM-external MAC units and auxiliary support logic for efficient deployment.

To address these limitations, this work presents a 28-nm digital via-programmed compute-in-ROM accelerator with three key features: 1) a via-programmed compute cell realizes high-frequency in-cell multiplication through two-stage output isolation and precharge-and-evaluate operation; 2) a selection-aware mapping scheme and a statistics-guided trimmed adder tree are jointly adopted to improve utilization and reduce reduction overhead; and 3) low-cost model migration is enabled through via-layer-only re-instantiation under a kernel-array organization, without costly logic re-synthesis or large auxiliary SRAM.

II. Proposed Architecture

A. Overall Architecture

Fig. 2 shows the overall architecture of the proposed accelerator, which is organized as a 4×4 kernel array. The horizontal dimension exploits output-channel (OC) parallelism, whereas the vertical dimension supports hierarchical input-channel (IC) reduction across kernels. Each kernel comprises 16 parallel OC lanes, and each OC lane is further divided into 8 bit-slices. Within a kernel, cell outputs are locally reduced by adder trees and then merged by a precision-scalable shift-add unit to complete intra-kernel IC reduction. At the array level, a two-stage inter-kernel reduction network is employed. The first stage performs cross-kernel IC reduction and serial-input accumulation, while the second stage accumulates both the intermediate sums from inter-kernel reduction and the partial sums generated across multiple computation rounds of a single convolution. The input buffer, interconnect, and multi-bank buffer support data movement and storage, while the Auxiliary Processing Engine (APE) executes auxiliary operations such as activation and the top controller coordinates the overall control flow.

B. Via-Programmed Compute Cell Design

Fig. 3 details the proposed via-programmed compute cell. As shown in Fig. 3(a), the cell adopts a two-stage structure, where the first stage embeds the fixed weights through the presence or absence of vias, and the second stage is dedicated to output isolation. The via-programmed first stage

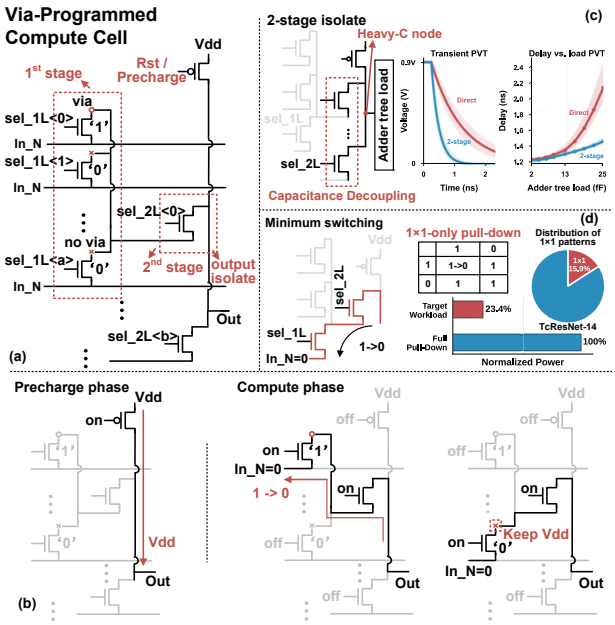

Fig. 3. Via-programmed compute cell: (a) two-stage cell structure, (b) precharge-and-evaluate operation, (c) transient/load behavior under PVT with second-stage output isolation, and (d) workload-aware pull-down activity and power reduction.

directly determines whether the selected input activates a pull-down path, thereby realizing weighted computation without an explicit multiplier. The second-stage MOS does not store weight information; instead, it isolates the internal weighted-compute node from the heavy capacitive load of the following adder tree. As illustrated in Fig. 3(b), the cell operates in a precharge-and-evaluate manner: the output is first precharged to V_{dd}, and is then conditionally discharged during the compute phase according to the input and the programmed weight. This two-stage isolation effectively decouples the internal compute node from the downstream load, leading to faster transients and smaller delay degradation under process-voltage-temperature (PVT) variations, as shown in Fig. 3(c). In addition, the precharge-and-evaluate scheme suppresses unnecessary switching activity under the target workload. As shown in Fig. 3(d), the 1×1-only pull-down pattern accounts for only 15.9% of all activation cases, reducing the normalized power under the target workload to 23.4% of that of the full pull-down design.

C. Selection-Aware Mapping and Reduction Optimization

Fig. 4 illustrates how convolution layers are mapped onto the proposed architecture. For a layer with dimensions OC $\times$ $W \times L \times$ IC, the output-channel (OC) dimension is mapped across the horizontal direction of the kernel array to provide output parallelism, whereas the input-channel (IC) dimension is mapped along the vertical direction to support hierarchical accumulation across kernels. Meanwhile, the filter width and height dimensions (W, L) are embedded into the intra-cell selection space, where different spatial positions are assigned

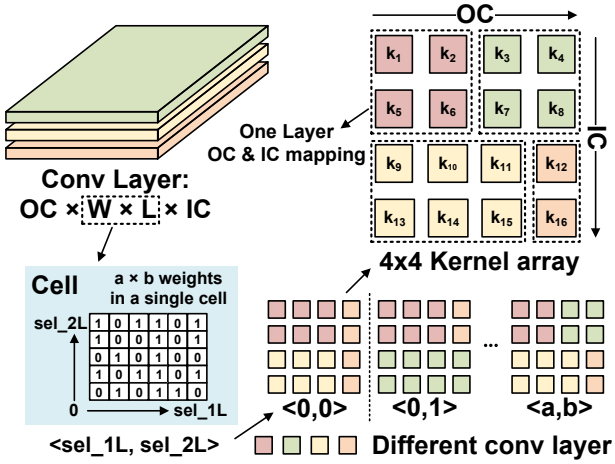

Fig. 4. Layer mapping onto the 4×4 kernel array and hierarchical fixed-weight selection inside each active cell.

to different selection pairs $\langle \text{sel\_1L}, \text{sel\_2L} \rangle$. In this way, a single cell can store multiple weights corresponding to different (W, L) locations. Therefore, the proposed mapping spans three levels: OC is spatially expanded for parallelism, IC is reduced through kernel-array accumulation, and (W, L) is encoded through cell-level selections. Moreover, since different convolution layers may require different kernel-level OC and IC tiling factors, a given 4×4 kernel array under one selection pair does not have to be exclusively assigned to a single layer. Instead, it can be partitioned and shared by multiple layers, as illustrated in the upper-right example, to improve hardware utilization under layer-dependent mapping demands.

Fig. 5 illustrates the proposed statistics-guided adder-tree bit-width trimming and remapping scheme. As shown in Fig. 5(a), the high sparsity of deep neural networks leads to frequent low-range patterns in the digital reduction path. For a conventional first-stage binary adder, the output range is $\{0, 1, 2\}$, requiring 2 bits. With selection-aware mapping, however, the two inputs of each first-stage adder can be organized to avoid the $(1, 1)$ case, reducing the output range to $\{0, 1\}$ and thus enabling 1-bit implementation. Owing to the same sparsity property, higher-stage adders also exhibit tightened output ranges under frequent low-activity joint patterns, which creates further opportunities for stage-wise width reduction.

Accordingly, a generic trimmed adder tree with preserved redundancy is constructed, as shown in Fig. 5(c). Rather than structurally pruning the tree for a specific fixed network, the proposed design trims the stage-wise adder width according to statistically supported range constraints, thereby retaining generality across different models. As shown in Fig. 5(d), new models can then be remapped onto the existing trimmed tree by reassigning weight entries through the $\langle \text{sel\_1L}, \text{sel\_2L} \rangle$ selection space, allowing the same trimmed reduction structure to be reused across model updates. The resulting benefit is summarized in Fig. 5(b), where the normalized energy and area are reduced to 0.59× and 0.68×, respectively, compared with those of a conventional adder tree.

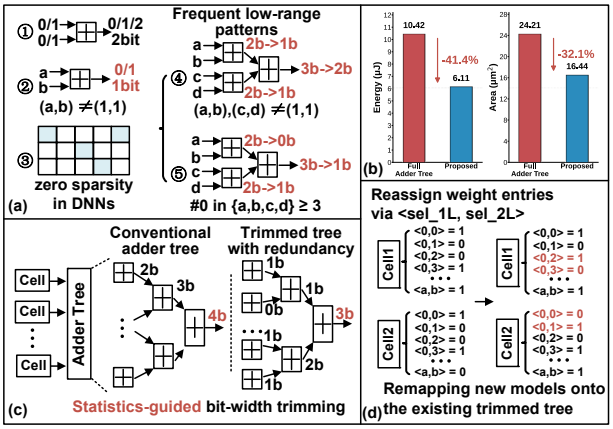

Fig. 5. Statistics-guided adder-tree bit-width trimming and remapping.

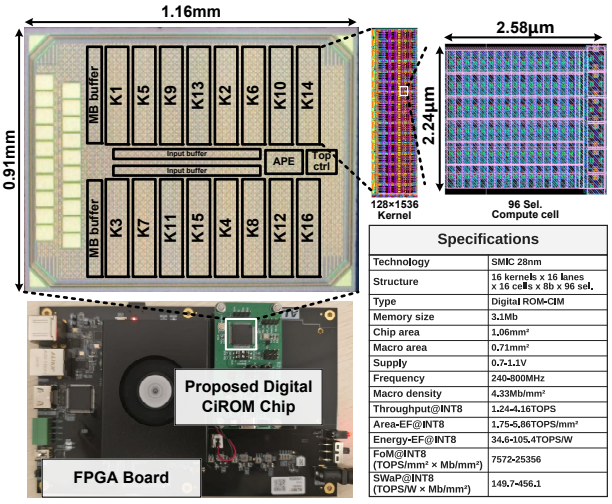

Fig. 6. Chip die photo, FPGA-based test platform, and key specifications.

III. Measurement Results

Fig. 6 shows the fabricated prototype and the measurement platform. Implemented in a 28-nm process, the chip occupies 1.06 mm^2, including a 0.71 mm^2 Compute-in-ROM macro. Measured on an FPGA-based setup, the prototype operates from 0.7–1.1 V at 240–800 MHz. The measured macro density reaches 4.33 Mb/mm^2. At INT8, the chip delivers 1.24–4.16 TOPS throughput, with an area efficiency of 1.75–5.86 TOPS/mm^2 and an energy efficiency of 34.6–105.4 TOPS/W.

Fig. 7 shows the measured VDD-dependent performance of the prototype over a 0.7–1.1 V range. The task-level results are obtained using the TCResNet-14 network. As VDD increases, the area efficiency improves whereas the energy efficiency decreases. Across the measured range, the peak/task-level area efficiency varies from 1.75/0.49 to 5.86/1.76 TOPS/mm^2, while the peak/task-level energy efficiency changes from 105.4/44.9 to 34.6/8.5 TOPS/W. The measured shmoo plot further confirms stable operation over the evaluated voltage-frequency range.

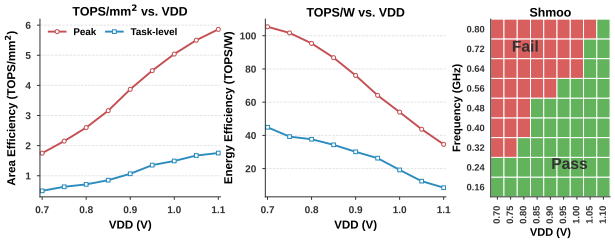

Fig. 7. Measured area efficiency, energy efficiency, and operating shmoo versus supply voltage.

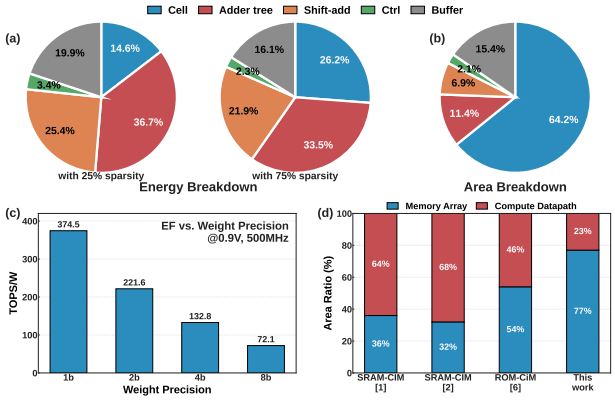

Fig. 8. (a) Post-layout-simulated energy breakdown with 25%/75% sparsity, (b) area breakdown, (c) measured energy efficiency versus weight precision, and (d) memory/compute-datapath area ratio comparison.

Fig. 8 summarizes the macro-level area and energy characteristics of the proposed design. Fig. 8(a) shows the post-layout-simulated energy breakdown at 0.9 V and 500 MHz under 25% and 75% sparsity. Fig. 8(b) shows the macro area breakdown, where the cell array occupies 64.2% of the total area, indicating a memory-centric implementation with low peripheral overhead. Fig. 8(c) shows measured energy efficiencies of 374.5–72.1 TOPS/W from 1b to 8b weight. Fig. 8(d) compares the area ratio between the memory array and compute datapath. This work achieves a 77% memory-array ratio, substantially higher than prior SRAM-CIM and ROM-CIM designs, confirming the high area efficiency of the proposed macro.

Table I compares this work with recent SRAM- and ROM-based CIM macros. Compared with prior analog ROM-CIM works, the proposed fully digital design avoids ADC-based readout bottlenecks and therefore achieves a peak throughput of 4.16 TOPS and a peak computing density of 5.86 TOPS/mm^2, corresponding to 2.6× and 9.2× improvements over analog ROM-CIM results, respectively. Compared with prior digital ROM-CIM works, where ROM readout is followed by external MAC operations, this work performs multiplication directly inside the cell, leading to a peak energy efficiency of 105.4 TOPS/W, a peak computing density of 5.86 TOPS/mm^2, and a memory density of 4327 Kb/mm^2, corresponding to 5.6×, 5.7×, and 8.9× improvements, respectively. In addition, low-cost model update is supported through via-layer-only re-instantiation together with flexible mapping, with a re-instantiation cost of 2.6%.

TABLE I
COMPARISON WITH REPRESENTATIVE SRAM-CIM AND ROM-BASED CIM ACCELERATORS.

Metric	ISSCC'25 [1]	ISSCC'25 [2]	JSSC'24 [4]	CICC'24 [3]	TCAS-I'25 [6]	**This work**
Technology	28nm	28nm	65nm	28nm	65nm	28nm
CIM operation	Hybrid SRAM	Digital SRAM	Analog ROM	Analog ROM	Digital ROM	Digital ROM
Flexibility + Cost	Full none	Full none	Limited + extra SRAM	Limited + extra SRAM	Limited + extra SRAM	Full + 2.6% re-inst.
Capacity (Kb)	64	32	2000	22528	3024	3072
Macro area (mm^2)	0.11	0.04	0.51	2.52	6.21	0.71
Frequency (MHz)	83.3	400	50∼210	120∼220	130∼800	240∼800
Input precision	8	4	1∼8	1∼8	1∼8	1∼8
Weight precision	8	8	2∼8	2∼8	4	1∼8
Output precision	Full	FP8/BF16	5	Full	Full	Full
Voltage (V)	0.6∼0.9	0.55∼0.9	0.7∼1.2	0.7∼1.1	0.6/1.2	0.7∼1.1
Throughput[a,b] (TOPS)	0.16∼0.68	0.25	0.013∼0.055	0.88∼1.6	1.71/6.41	1.24∼4.16[c]
Energy efficiency[a,b] (TOPS/W)	84∼271	35∼100	1.24∼4.33	80∼168	19.0/4.5	34.6∼105.4[c]
Computing density[a,b] (TOPS/mm^2)	1.48∼6.28	5.78	0.026∼0.109	0.32∼0.64	0.28/1.03	1.75∼5.86[c]
Memory density (Kb/mm^2)	594	732	3984	8928	487	4327
SWa[a,b] (TOPS/W · Mb/mm^2)	49.9∼161.0	25.6∼73.2	4.94∼17.25	714.2∼1499.9	9.3/2.2	149.7∼456.1[c]
FoM[a,b] (TOPS/mm^2 · Kb/mm^2)	879∼3730	4231	104∼434	2856∼5713	134/502	7572∼25356[c]

[a] 1 MAC = 2 OP. [b] Normalized to 8b weight and 8b activation. [c] With 50% weight and 50% activation sparsity.

IV. CONCLUSION

This work presents a 28-nm via-programmed digital compute-in-ROM accelerator for CNN inference. By enabling in-cell multiplication and adopting selection-aware mapping with a trimmed adder tree, the proposed design improves compute density and energy efficiency while supporting low-cost model migration through via-layer-only re-instantiation. The fabricated prototype achieves 4.16 TOPS peak throughput, 5.86 TOPS/mm² peak area efficiency, and 105.4 TOPS/W peak energy efficiency, demonstrating the potential of digital compute-in-ROM for compact and efficient edge AI acceleration.

REFERENCES

[1] X. Chen *et al.*, "A 28nm 64kb bit-rotated hybrid-CIM macro with an embedded sign-bit-processing array and a multi-bit-fusion dual-granularity cooperative quantizer," in *Proc. IEEE International Solid-State Circuits Conference (ISSCC)*, 2025.

[2] Y. Yuan *et al.*, "A 28nm 192.3TFLOPS/W accurate/approximate dual-mode-transpose digital 6T-SRAM CIM macro for floating-point edge training and inference," in *Proc. IEEE International Solid-State Circuits Conference (ISSCC)*, 2025.

[3] G. Yin *et al.*, "A 28nm 8928Kb/mm²-weight-density hybrid SRAM/ROM compute-in-memory architecture reducing >95% weight loading from DRAM," in *Proc. IEEE Custom Integrated Circuits Conference (CICC)*, 2024.

[4] G. Yin *et al.*, "Cramming More Weight Data Onto Compute-in-Memory Macros for High Task-Level Energy Efficiency Using Custom ROM With 3984-kb/mm² Density in 65-nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 59, no. 6, Jun. 2024.

[5] L.-A. Cheong *et al.*, "A 28nm 166.9 TOPS/W × Mb/mm² DRAM-Free QLC Compute-in-ROM Macro Supporting High Task-Level Inference Energy Efficiency for Tiny AI Edge Devices," in *Proc. IEEE Asian Solid-State Circuits Conference (A-SSCC)*, 2024.

[6] T. Yu *et al.*, "DCiROM: A High-Density Fully-Digital Compute-in-Read-Only-Memory Macro for Energy-Efficient Task-Level DNN Inference," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 72, no. 12, Dec. 2025.

[7] T. Yu *et al.*, "DSC-ROM: A Fully Digital Sparsity-Compressed Compute-in-ROM Architecture for On-Chip Deployment of Large-Scale DNNs," in *Proc. Design, Automation and Test in Europe Conference (DATE)*, 2025.