

FSDB: A Folded-Store Dynamic-Broaden Hybrid Compute-in-ROM/SRAM Architecture for Deploying Large-Scale DNNs On-Chip

Tianyi Yu, Teng Yi, Huazhong Yang, Xueqing Li

Department of Electronic Engineering, LFET/BNRist/SKLSNC, Tsinghua University

Corresponding Email: xueqingli@tsinghua.edu.cn, yiteng1005@163.com

Abstract—Compute-in-Memory (CiM) has emerged as a promising paradigm to overcome the memory bottleneck of von Neumann architectures in data-intensive applications. While SRAM-based CiM benefits from mature fabrication support and high design flexibility, it suffers from significant access energy due to limited memory density. Recent advances in ROM-based CiM provide a high-density, energy-efficient alternative for deploying entire deep neural network (DNN) models on-chip, often assisted by small SRAM CiM modules to enhance task-level flexibility. However, existing ROM CiM architectures still face critical challenges in further scaling memory density and achieving finer-grained flexibility improvement.

This paper presents FSDB, a digital hybrid ROM/SRAM CiM architecture to address these limitations. FSDB incorporates a folded-store compressed ROM CiM macro implemented using a sparsity-aware quantization methodology, achieving a record-high memory density of 40.2 Mb/mm² in a 28nm CMOS technology. Furthermore, the proposed dynamic-broaden computing architecture enables updates to parameters stored in ROM, providing kernel-level reconfigurability and cross-model scalability. Experimental results on an extended ResNet-50 demonstrate that FSDB improves inference accuracy by >5% on ImageNet compared to prior state-of-the-art (SOTA) flexible ROM CiM architectures.

Index Terms—compute-in-memory (CiM), read-only memory (ROM), multiply-and-accumulate (MAC), deep neural network (DNN), sparse quantization, transfer learning.

I. INTRODUCTION

Deep neural networks (DNNs) have driven remarkable progress across various artificial intelligence (AI) domains, such as computer vision, natural language processing, and AI generated content [1]. However, the growing computational and memory requirements have intensified the “memory-wall” bottleneck inherent in the von Neumann architecture [2]. Frequent data transfers between memory and processing units severely degrade both performance and energy efficiency, particularly in resource-constrained edge computing platforms. To mitigate this challenge, compute-in-memory (CiM) has emerged as a promising solution. By performing multiply-and-accumulate (MAC) operations directly within memory, CiM significantly reduces data movement and improves system efficiency. Among various CiM technologies, SRAM-based implementations have received considerable attention owing to their high flexibility with mature semiconductor fabrication processes [3]–[6]. Nevertheless, conventional SRAM cells suffer from low memory density, which constrains on-chip capacity and necessitates frequent off-chip weight accesses during large-scale DNN infer-

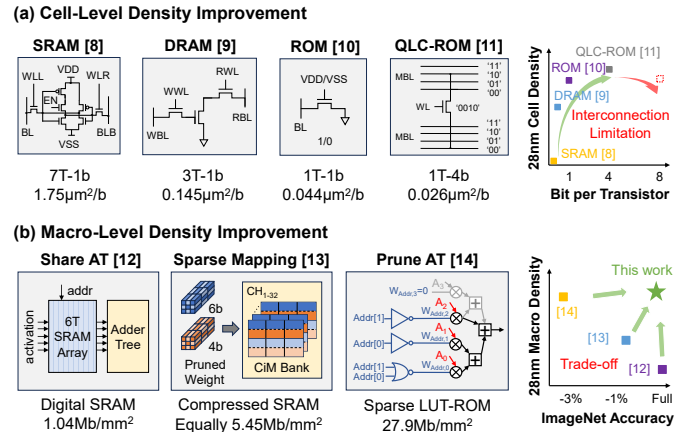


Fig. 1. Approaches to on-chip memory density enhancement: (a) Memory cell customization with reduced transistor count, (b) Streamlining of computing resources and model sparsity compression.

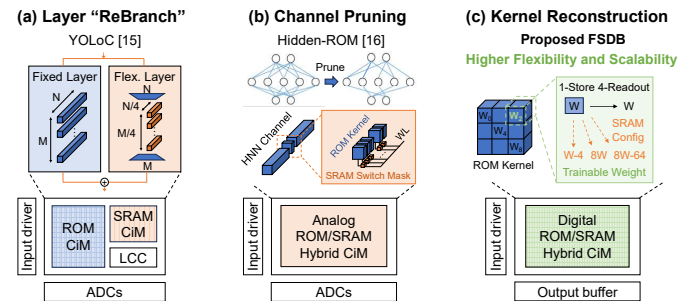


Fig. 2. Comparison of hybrid ROM/SRAM CiM architectures for improving flexibility at different granularities: (a) YOLOc [8] at layer-level, (b) Hidden-ROM [9] at channel-level, (c) Proposed kernel-level FSDB.

ence. These external memory transactions introduce substantial energy consumption and latency overheads [7].

To enhance on-chip storage capacity, various customized memory cells [10]–[13] have been proposed. As illustrated in Fig. 1(a), read-only designs increase the storage bits per transistor, significantly improving memory density. However, further reduction in area per bit remains challenging due to limitations in layout interconnects, particularly with scaling technology nodes. On the other hand, software-hardware co-optimization provides a promising alternative. As shown in Fig. 1(b), techniques such as sparse mapping and streamlined com-

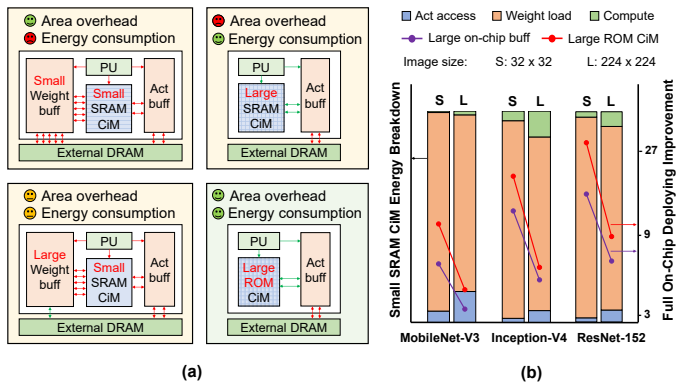


Fig. 3. SRAM CiM challenge and ROM CiM opportunity: (a) Overcoming area and energy overhead limitation by enhancing memory density, (b) Task-level energy breakdown and improvement through full on-chip deployment.

puting resources can alleviate memory array constraints [14]–[16]. Nevertheless, pruning the adder tree (AT) and applying model compression often introduces precision loss, particularly in complex applications. Consequently, existing solutions face a trade-off between memory density and inference accuracy.

Although high-density ROM-based CiM effectively improves memory density, its flexibility is limited by the fixed weights. Fortunately, the integration of small SRAM modules with transfer learning techniques offers a promising pathway to extend the applicability of ROM CiM to diverse scenarios. As illustrated in Fig. 2, the task-flexible YOLOc [8] and the model-scalable Hidden-ROM [9] architectures introduce parallel trainable branch and channel pruning mechanisms, respectively. However, the convolutional kernels mapped onto ROM in current implementations remain non-reconfigurable, constraining each macro to a single DNN architecture. As a result, ROM CiM still faces significant challenges in achieving cross-model compatibility.

To address the aforementioned challenges, this paper proposes a Folded-Store Dynamic-Broaden (FSDB) hybrid compute-in-ROM/SRAM architecture. The key contributions of this work are as follows:

- **Folded-Store hybrid ROM/SRAM CiM architecture:** This work introduces a reconfigurable LUT-based ROM compression design approach enabled by sparsity-aware DNN quantization. The proposed hybrid CiM macro achieves a record-high memory density of 40.2 Mb/mm² in a 28nm technology.
- **Dynamic-Broaden transfer learning framework:** A dynamically reconfigurable ROM kernel implementation is proposed, supported by runtime SRAM reconfiguration during MAC operations. The proposed design enhances inference accuracy by >5% on ImageNet while enabling cross-model migration.
- **Comprehensive experimental validation:** This work conducts extensive hardware evaluations at both macro and system levels, along with software assessments at algorithmic level across multiple datasets and model architectures, demonstrating the effectiveness of the proposed design.

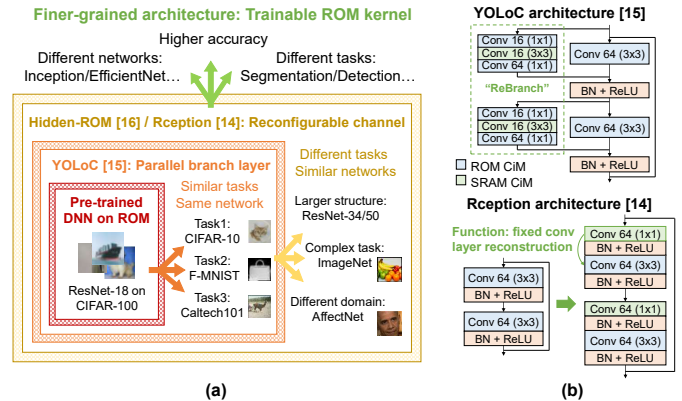


Fig. 4. Prior hybrid ROM/SRAM architectures for improving flexibility: (a) Task-specific limitations of ROM CiM across different architectural levels, (b) Detailed structures of YOLOc [8] and Reception [19].

II. BACKGROUND

On-chip capacity challenge. In low-density implementation scenarios, large-scale computing platforms typically face a fundamental trade-off between area and power, as illustrated in Fig. 3(a). SRAM CiM architectures struggle with high inter-chip communication overhead and substantial area consumption. Model sparse compression techniques [15], [17], [18] effectively reduce data movement but introduce accuracy loss and area/latency overhead. Enhancing memory density provides a viable pathway to achieve higher energy efficiency at a lower cost. As shown in Fig. 3(b), the energy consumption of off-chip versus fully on-chip deployment is compared across multiple model architectures and input sizes. Thanks to its inherently lower read energy, ROM CiM achieves superior energy efficiency compared to large-scale SRAM implementations.

High-density CiM. While advanced process scaling [20], [21] provides high memory density, it incurs prohibitively high fabrication costs. Alternative non-volatile memory technologies such as RRAM [22], [23], FeFET [24], and memristors [25] face challenges related to immature fabrication processes, limited endurance, and high write energy. Embedded DRAM [26] offers a smaller cell area, but introduces refresh energy and latency consumption. Recent ROM CiM designs leveraging multi-level cell (MLC) techniques [27] have achieved notable density improvements. However, due to constraints in available metal layers and inherent non-writability, ROM CiM remains limited by further scaling and flexibility.

ROM CiM Flexibility. To address the task-specific limitations of ROM CiM, several transfer learning architectures have been proposed, as illustrated in Fig. 4(a). For instance, the YOLOc architecture [8] incorporates a trainable parallel SRAM CiM branch to improve flexibility within a single task domain. In contrast, Hidden-ROM [9] and Reception [19] enhance cross-task adaptability through channel pruning and Inception-based [28] reconstruction within scalable models. Nevertheless, existing array-level hybrid architectures, as depicted in Fig. 4(b), remain constrained in supporting structural network modifications due to the fixed convolutional kernels. In response, this work proposes a finer-grained hybrid ROM/SRAM architecture that enables kernel-level reconfigurability.

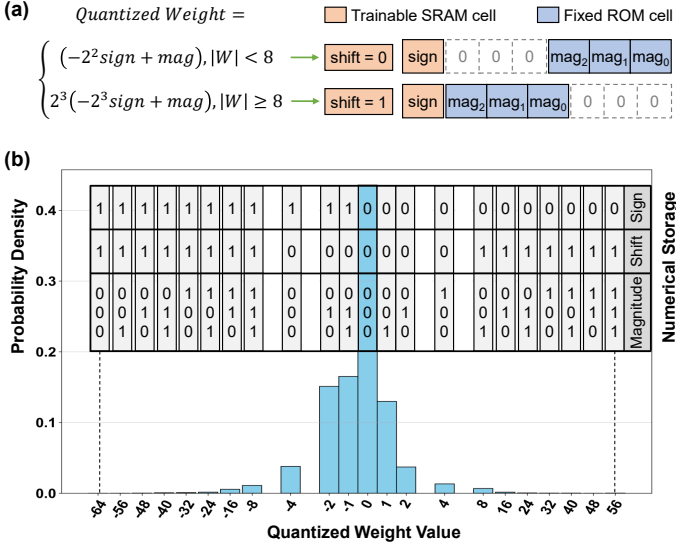


Fig. 5. Proposed folded-store sparse quantization method: (a) Schematic of the numerical storage strategy, (b) Probability density distribution of quantized weights in ResNet-18.

III. PROPOSED FSDB ARCHITECTURE

This section introduces the proposed FSDB architecture, detailing its sparsity-aware quantization method, weight mapping strategy, circuit-level implementation, and multi-mode computing workflow.

A. Folded-Store Quantization Method

Motivation. The sparsity of quantized DNN weights plays a critical role in determining the memory density and energy efficiency of ROM CiM macros. Conventional 8-bit quantized weights in two’s complement format exhibit inherently limited sparsity, whereas sign-magnitude representation introduces additional circuit overhead. Although low-bit quantization (e.g., W4A8, W4A4) can significantly improve weight sparsity, it often leads to degraded inference accuracy.

Folded weight storing. Fig. 5(a) illustrates the representation and storage strategy for quantized weights. Weights are stored in two’s complement format, with the “sign” bit retained in SRAM and the folded 3-bit “magnitude” (mag) stored in ROM. The 1-bit “shift” signal, also stored in SRAM, controls whether the weight is shifted left by three bits. Compared to the fully ROM implementation, the hybrid compressed storage scheme reduces computational overhead while maintaining parametric flexibility.

Non-uniform step size quantization. Building upon weight compression, the Folded-Store Quantization (FSQ) scheme is proposed employing non-uniform step sizes [29]. Fig. 5(b) illustrates the probability distribution of quantized weight values and their corresponding stored representations. The quantization step size is set to 1 in the dense region (0~2) to preserve accuracy, increased to 2 and 4 in the intermediate region (2~8) to promote sparsity, and raised to 8 in the outlier region (>8) to maintain a high dynamic range. Additionally, weight values within the -4~-1 interval are encoded using a 2-bit magnitude representation to further enhance sparsity.

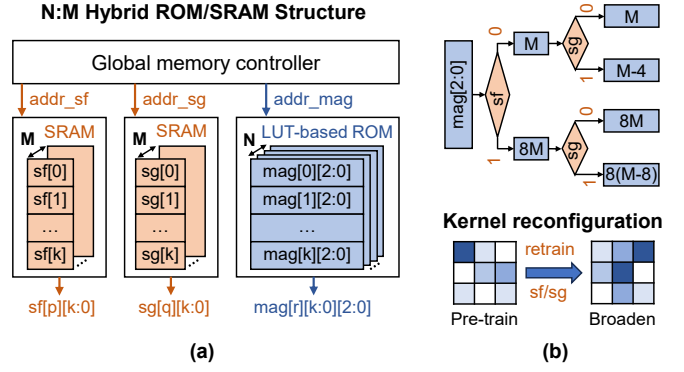


Fig. 6. Proposed read-only parameter dynamic-broaden structure: (a) N:M hybrid ROM/SRAM mapping strategy, (b) Convolutional kernel reconfiguration mechanism.

B. Dynamic-Broaden Flexibility Mechanism

N:M hybrid ROM/SRAM structure. The mag of quantized weights from different layers of ResNet-18 is proportionally mapped onto an N-depth ROM array based on each layer’s size. Owing to the more concentrated weight distribution and larger parameter counts [30], the deeper convolutional layers of ResNet attain higher compression rates. Consequently, a one-to-one correspondence between the shift (sf) and sign (sg) in SRAM and the mag in ROM is generally unnecessary. As illustrated in Fig. 6(a), a hybrid storage scheme employing compression ratios such as 4:1 or 16:1 substantially reduces circuit overhead while preserving performance.

Convolutional kernel reconfiguration. Fig. 6(b) illustrates the principle of convolution kernel reconstruction. Each mag value can generate up to four distinct readout values, depending on the configurations of the sf and sg. By retraining sf and sg, the expanded convolution kernels are capable of extracting diverse features from the input feature maps. This approach enables the adaptation of pre-trained DNNs to various application scenarios and different network architectures.

Dynamic-broaden workflow. Equation 1 illustrates the principle of the two-stage MAC operation. The input activations are first processed by a shift masker before entering the macro for computation, generating a partial sum (PSUM). The mask value applied in the second stage is the bit-wise inverse of that used in the first stage. The two PSUM results are then shifted and accumulated to produce the final MAC output. As shown in Equation 2, the underlying PSUM computation involves separate processing of the sign and mag components, which are subsequently combined during the shift-and-sum phase.

$$\begin{aligned}
 MAC &= PSUM_1 + 2^3 \cdot PSUM_2 \\
 PSUM_1 &= \sum_i (\overline{shift}_i \cdot A_i) W_i \\
 PSUM_2 &= \sum_i (shift_i \cdot A_i) W_i
 \end{aligned} \tag{1}$$

$$\begin{aligned}
 PSUM_k &= \sum_i A_{i,k} W_i \\
 &= \sum_i A_{i,k} mag_i - 2^{k+1} \sum_i A_{i,k} sign_i
 \end{aligned} \tag{2}$$

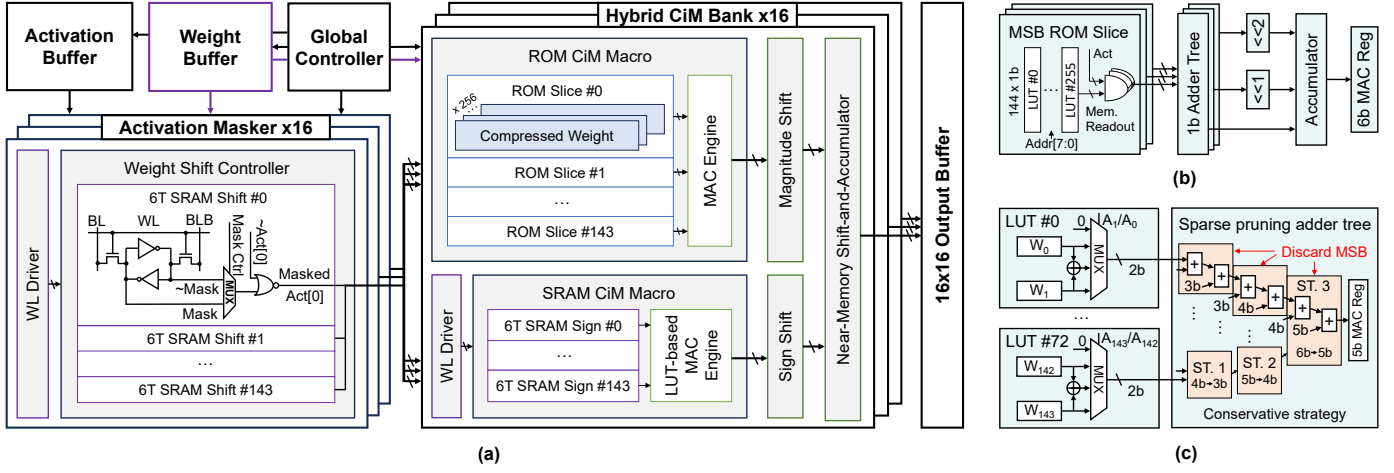


Fig. 7. Overall architecture of the proposed FSDB core: (a) Digital hybrid CiM structure and activation masking mechanism, (b) ROM macro employing LUT-compressed memory, (c) SRAM macro integrated with LUT-based MAC engines and pruned adder tree.

C. Hybrid ROM/SRAM CiM Architecture

Overall architecture. Fig. 7(a) illustrates the overall architecture of the digital hybrid ROM/SRAM CiM core in the proposed FSDB design. The architecture consists of 16 CiM banks accompanied by peripheral circuits, including 16 activation maskers, wordline (WL) drivers, normal I/O interfaces, distributed SRAM buffers, and a global controller. Each hybrid CiM bank integrates two separate computing modules, a ROM macro and an SRAM macro, along with two independent barrel shifters, and a near-memory shift-and-accumulator block. Each activation masker is composed of a $16 \times 3 \times 3$ array of 6T read-free SRAM cells, supplemented by a multiplexer (MUX) and a NOR gate, to enable dynamic shift control for the weights within the CiM bank.

ROM CiM macro. The ROM macro consists of 144 LUT-based slices, each storing 256 compressed magnitude (mag) values. As shown in Fig. 7(b), the 3-bit mag is partitioned into three 1-bit segments, each processed by a dedicated 144-operand adder tree. The intermediate results are then merged through shift-and-accumulate operations to generate the final MAC output. The ROM array is synthesized using Electronic Design Automation (EDA) tools and replaced with a highly compressed look-up table (LUT) implementation. The ROM compression technique, adapted from [31], ensures all transistors within the LUT maintain a static state throughout the bit-serial computation cycles.

SRAM CiM macro. The SRAM CiM macro comprises 144 foundry-compliant 6T SRAM cells and a LUT-based MAC engine, adopting a design methodology consistent with [32]. As depicted in Fig. 7(c), two adjacent weights are pre-added, and the four possible summation results are selected by a MUX controlled by two corresponding activation bits. This approach reduces dynamic power consumption and shortens the critical path latency. The adder tree incorporates sparse pruning optimization similar to [33], leveraging both activation and weight sparsity to reduce the bit-width of the shift-accumulator by 3 bits without sacrificing accuracy.

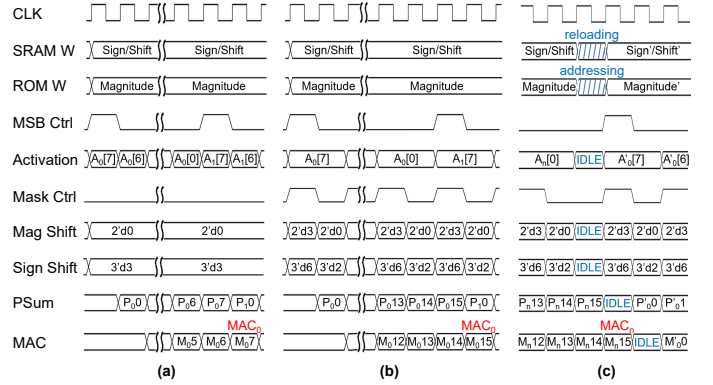


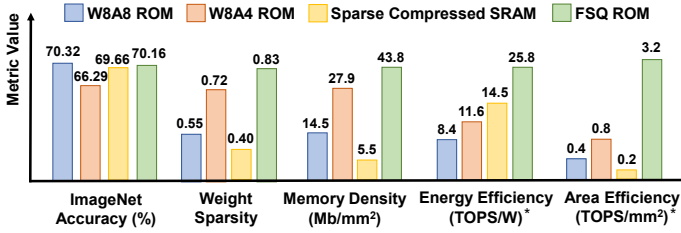
Fig. 8. Timing waveforms of (a) W4A8, (b) W8A8, and (c) SRAM reloading.

D. Reconfigurable Pipelined Computing Flow

Two-stage W4A8 MAC. As illustrated in Fig. 8(a), the 8-bit activation is transmitted to the FSDB macro in a bit-serial manner, with the “MSB Ctrl” signal indicating the start bit of each activation. The “Mask Ctrl”, “Mag Shift”, and “Sign Shift” signals remain constant during the process. The “PSUM” and “MAC” consist of a two-stage pipeline, reducing the critical path latency by 40%.

Four-stage W8A8 MAC. As illustrated in Fig. 8(b), each bit of the serially input activation is processed over two consecutive cycles. In the first cycle, Mask Ctrl is asserted high, while Mag/Sign Shift are set to 3/6, respectively. Activations corresponding to non-shifted weights are masked to zero to compute the shifted PSUM. During the second cycle, Mask Ctrl is driven low, and Mag/Sign Shift are configured to 0/2, respectively, to compute the non-shifted PSUM.

ROM addressing and SRAM reloading. The global memory controller requires one cycle to perform SRAM reloading and ROM addressing, as depicted in Fig. 8(c). Typically, since the feature map size ($7 \times 7 \sim 56 \times 56$) is considerably larger than the convolution kernel size ($1 \times 1 \sim 5 \times 5$), the memory configuration overhead is substantially amortized.



*: Normalized to W8A8, evaluated at 0.9V.

Fig. 9. Joint comparison of accuracy, sparsity, and PPA characteristics across CiM macros using corresponding quantization methods.

IV. EXPERIMENTAL RESULT

A. Experiment Setup

Software configuration. The folded-store quantization method is validated on ImageNet using a pre-trained ResNet-18 from Huggingface. PyTorch serves as the DNN framework for quantization-aware training (QAT) and fine-tuning under the proposed FSDB architecture. During transfer learning, the ROM-mapped parameters (mag) in all convolutional layers (except the first “conv1” layer) remain frozen, while the SRAM-mapped parameters (sign, shift, and fully connected layers) are updatable. The “conv1” and non-linear layers are trained using the Brain Floating Point 16 (BF16) format.

Hardware configuration. The macro-level evaluation is performed through post-layout simulations in 16nm, 28nm, and 65nm CMOS technology nodes. Energy estimation and static timing analysis are conducted using Synopsys PrimeTime. Dynamic power consumption is measured under an average input toggle rate of 25%. System-level energy assessment incorporates both on-chip and off-chip data transport, with energy costs estimated at 5 pJ/B for SRAM read operations and 160 pJ/B for DRAM read operations [34], respectively.

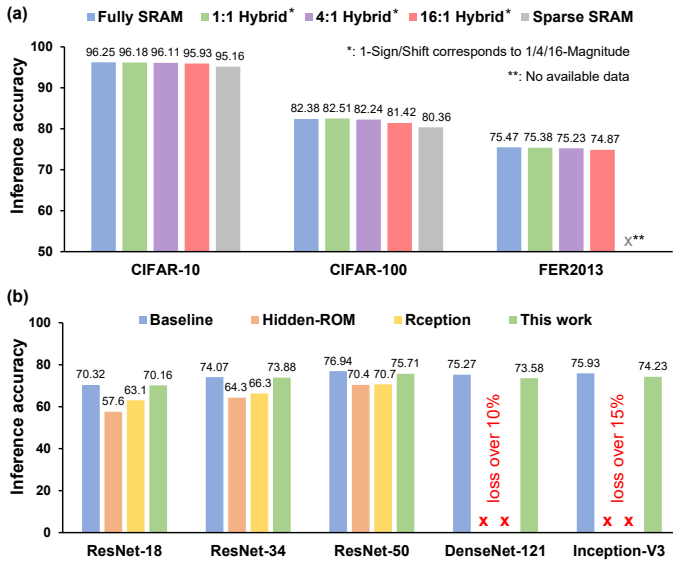


Fig. 10. Flexibility evaluation of the proposed FSDB architecture: (a) Inference accuracy of ResNet-18 across multiple datasets, (b) ImageNet accuracy comparison under model expansion with Hidden-ROM and Rception.

TABLE I
PERFORMANCE COMPARISON

Specifications	Fully SRAM [12]	Sparse SRAM [13]	¹ YOLOc [15]	¹ DSC-ROM [14]	¹ This work
Process	28nm CMOS				
CiM operation	Digital SRAM	Analog SRAM	Analog SRAM+ROM	Digital SRAM+ROM	Digital Hybrid
Capacity (Mb)	0.032	0.5	1.26	18.01	37.75
Macro area (mm ²)	0.03	1.44	0.48	0.65	0.94
Precision (Input x Weight)	1~8-bit x 1/4/8-bit	4/8-bit x 2~8-bit	1~8-bit x 8-bit	1~8-bit x 4-bit	1~8-bit x 4/8-bit
² Throughput (GOPS)	5.4	264.1	57.3	1251.6	2801.0
Memory density (Mb/mm ²)	1.07	³ 5.45	2.62	27.9	40.2 1.4x
² Energy efficiency (TOPS/W)	21.80	14.51	11.50	11.63	22.20
² Area efficiency (GOPS/mm ²)	180.0	183.0	119.4	792.2	2979.8 3.8x
Flexibility	Full	Model-pruning High	Layer-level Low	Channel-level Medium	Kernel-level High

¹: Evaluated by post-layout simulation at 0.9V.

²: Normalized to 8-bit input x 8-bit weight.

³: Equally memory density = compression rate x capacity / macro area.

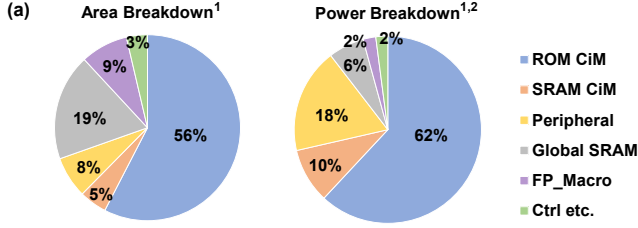
B. Macro-level Performance

Folded-store quantization. Fig. 9 compares the performance among W8A8, W4A8, model sparse compression [15], and the proposed FSQ method. With only a 0.16% accuracy loss relative to W8A8, FSQ achieves over 3% higher accuracy than W4A8 on ResNet-18. Additionally, FSQ attains a weight sparsity of 0.83, enabling the highest memory density of 43.8 Mb/mm². Owing to reduced mapping overhead and a pipelined architecture, the FSQ-based ROM CiM macro achieves the highest energy efficiency of 25.8 TOPS/W and an outstanding area efficiency of 3.2 TOPS/mm² at 0.9V.

PPA comparison. Table I presents a comparison with prior SRAM and ROM CiM architectures. As a hybrid ROM/SRAM CiM design, the proposed FSDB achieves a record-high memory density of 40.2 Mb/mm², which is 1.4x higher than the SOTA DSC-ROM. Additionally, by leveraging sparsity-aware optimization and pipelined computation, FSDB attains 3.8x higher area efficiency. Furthermore, this work is the first to demonstrate weight precision reconfigurability within a ROM CiM architecture, thereby providing a broader spectrum of configurable inference options.

C. Flexibility and Scalability

Flexibility. Fig. 10(a) presents the Top-1 accuracy of FSDB under various N:M ROM/SRAM structures. Using a ResNet-18 model pre-trained on ImageNet, transfer learning is performed on CIFAR-10, CIFAR-100 (item classification), and FER2013 (facial expression recognition). Compared to a fully SRAM implementation, the 1:1 hybrid achieves a 0.13% accuracy improvement on CIFAR-100 with a 25% SRAM access energy overhead. The 4:1 hybrid strikes a balance between accuracy and energy efficiency, exhibiting only 0.14~0.24% inference accuracy degradation while reducing SRAM access energy to 6.25%. The 16:1 hybrid further reduces SRAM access energy to 1.56%, albeit with an accuracy loss ranging from 0.32~0.96%, yet still outperforming the model sparse compression [15].



1: Simulated at 0.9V, 1GHz clock, 28nm CMOS layout level.
2: Off-chip and I/O power is not included. Off-chip operation time is not included.

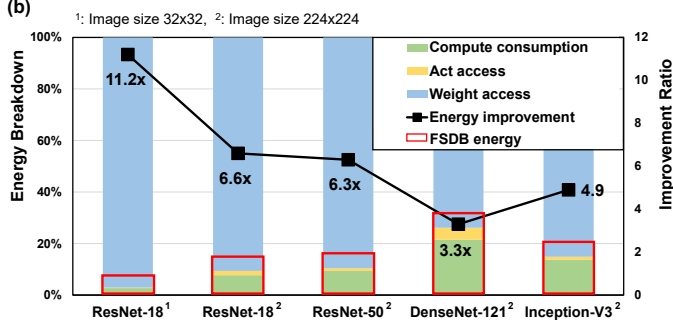


Fig. 11. System-level evaluations: (a) Area and energy breakdown. (b) Comparison of task-level inference energy consumption between partially deployed SRAM CiM and fully on-chip FSDB CiM.

Scalability. Fig. 10(b) presents the results of model expansion from ResNet-18 to ResNet-34/50, as well as cross-model adaptation to DenseNet-121 and Inception-V3 on ImageNet. In comparison to Hidden-ROM and Reception architectures, the proposed FSDB achieves over 5% accuracy improvement on ResNet-50 with only a 6.25% energy overhead for SRAM reloading. Moreover, by leveraging fine-grained pipelining and a compute-reload ping-pong strategy, FSDB maintains negligible latency overhead during expansion. It is noteworthy that Hidden-ROM and Reception are unable to support post-fabrication cross-model expansion due to the fixed convolutional kernels. In contrast, the proposed FSDB maintains strong adaptability, exhibiting only a 1.7% accuracy degradation on DenseNet-121 and Inception-V3.

D. System-level Evaluations

System breakdown. Fig. 11(a) illustrates the system-level power and area breakdown of the FSDB architecture. The results indicate that the majority of power consumption and area utilization is attributed to the MAC logic, which includes the ROM CiM, SRAM CiM, and floating-point (FP) macro blocks. In contrast, the peripheral circuits and global SRAM buffer account for only 24% of the total power and 27% of the total area, demonstrating the low hardware implementation overhead of the proposed design.

Task-level energy consumption. Fig. 11(b) compares the task-level energy consumption of FSDB with a partially deployed SRAM CiM. By storing all convolutional layer parameters in ROM, the proposed FSDB macro requires weight updates only for a small subset of SRAM cells. This design reduces off-chip weight reloading by over 90%, resulting in an energy efficiency improvement of 3.3-11.2x across various DNN models and input image sizes.

TABLE II
SCALING COMPARISON

Benchmark	¹ MLC-ROM [26]		² This work		
	65nm	28nm	65nm	28nm	⁴ 16nm
Process	65nm	28nm	65nm	28nm	0.55-1.0
Voltage (V)	0.9	0.9	0.9	0.9	
Capacity (Mb)	37.75				
Macro area (mm ²)	9.5	3.2	4.9	0.94	0.40
Frequency (GHz)	0.13	0.20	0.75	1.25	0.3-1.43
³ Throughput (TOPS)	0.61	0.94	1.67	2.74	0.66-3.15
Memory density (Mb/mm ²)	3.98	11.9	7.7	40.2	94.4
³ Energy efficiency (TOPS/W)	4.3	8.5	4.7	22.2	21.8-75.1
³ Area efficiency (TOPS/mm ²)	0.064	0.29	0.34	2.91	1.66-7.87

¹: Normalized to 37.75 Mb capacity. ²: Evaluated by post-layout simulation.
³: 1 Op = 8-bit input x 8-bit weight. ⁴: FinFET technology.

E. Scaling and Discussion

scaling capability. Table II compares the scaling performance of the analog MLC-ROM and the digital FSDB macro. When scaling from 65nm to 28nm CMOS technology, MLC-ROM only improves 3.0x in memory density, 2.0x in energy efficiency, and 4.5x in area efficiency due to the layout design rule limitation. In contrast, FSDB delivers higher gains of 5.2x, 4.7x, and 8.6x, respectively, demonstrating superior utilization of advanced process nodes. Further scaling to 16nm FinFET technology enables the proposed design to reach a memory density of 94.4 Mb/mm², an energy efficiency of 75.1 TOPS/W at 0.55V, and an area efficiency of 7.87 TOPS/mm² at 1.0V.

Future works. When extended to large language models (LLMs), ROM CiM demonstrates strong suitability for cloud inference acceleration, owing to its high throughput and low deployment cost. The flexibility limitations of ROM CiM are alleviated by the relatively infrequent parameter updates characteristic of LLMs. Continued progress in storage density and computational performance remains a highly promising direction for future research.

V. CONCLUSIONS

This paper presents FSDB, a digital folded-store dynamic-broaden hybrid compute-in-ROM/SRAM architecture to deal with the bottlenecks of prior ROM CiM. The proposed folded-store LUT-based compression macro achieves exceptional computing efficiency and a record-high memory density. Furthermore, the proposed dynamic-broaden architecture supports transfer learning with minimal accuracy loss and enables cross-model expansion through kernel-level reconfiguration, facilitating seamless adaptation to more complex scenarios and model upgrades.

ACKNOWLEDGMENT

This work is supported in part by NSFC (Grant # U24B6015, # 92264204, and # U21B2030), and in part by National Key R&D Program of China (Grant # 2024YFB3214500).

REFERENCES

- [1] S. Pouyanfar, S. Sadiq, Y. Yan, H. Tian, Y. Tao, M. P. Reyes, M.-L. Shyu, S.-C. Chen, and S. S. Iyengar, "A survey on deep learning: Algorithms, techniques, and applications," *ACM computing surveys (CSUR)*, vol. 51, no. 5, pp. 1–36, 2018.
- [2] W. A. Wulf and S. A. McKee, "Hitting the memory wall: Implications of the obvious," *ACM SIGARCH computer architecture news*, vol. 23, no. 1, pp. 20–24, 1995.
- [3] K. Yoshioka, "34.5 A 818-4094TOPS/W Capacitor-Reconfigured CIM Macro for Unified Acceleration of CNNs and Transformers," in *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 67, IEEE, 2024, pp. 574–576.
- [4] M. Wu, W. Ren, P. Chen, W. Zhao, Y. Jing, J. Ru, Z. Wang, Y. Ma, R. Huang, T. Jia, and L. Ye, "S2D-CIM: A 22nm 128Kb Systolic Digital Compute-in-Memory Macro with Domino Data Path for Flexible Vector Operation and 2-D Weight Update in Edge AI Applications," in *2024 IEEE Custom Integrated Circuits Conference (CICC)*, 2024, pp. 1–2.
- [5] B. Zhang, J. Saikia, S. Kwon, S. Myung, H. Kim, S. J. Kim, J.-S. Seo, M. Seok *et al.*, "MACC-SRAM: A Multistep Accumulation Capacitor-Coupling In-Memory Computing SRAM Macro for Deep Convolutional Neural Networks," *IEEE Journal of Solid-State Circuits*, 2023.
- [6] Y. Yuan, Y. Yang, X. Wang, X. Li, C. Ma, Q. Chen, M. Tang, X. Wei, Z. Hou, J. Zhu, H. Wu, Q. Ren, G. Xing, P.-I. Mak, and F. Zhang, "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," in *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 67, 2024, pp. 576–578.
- [7] H. Jiang, S. Huang, X. Peng, J.-W. Su, Y.-C. Chou, W.-H. Huang, T.-W. Liu, R. Liu, M.-F. Chang, and S. Yu, "A two-way SRAM array based accelerator for deep neural network on-chip training," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2020, pp. 1–6.
- [8] Y. Chen, G. Yin, Z. Tan, Y. Liu, H. Yang, K. Ma, X. Li *et al.*, "YOLoC: deploy large-scale neural network by ROM-based computing-in-memory using residual branch on a chip," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, 2022, pp. 1093–1098.
- [9] Y. Chen, G. Yin, M. Lee, W. Tang, Z. Yang, Y. Liu, H. Yang, and X. Li, "Hidden-ROM: A compute-in-ROM architecture to deploy large-scale neural networks on chip with flexible and scalable post-fabrication task transfer capability," in *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, 2022, pp. 1–9.
- [10] C. Dai, J. Zhang, R. Wang, J. Chen, W. Hu, L. Hao, W. Lu, Z. Lin, C. Peng, and X. Wu, "A PVT-insensitive 7T SRAM CIM macro for multibit multiplication with dynamic matching quantization circuits," *Microelectronics Journal*, vol. 161, p. 106703, 2025. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1879239125001523>
- [11] Y. He, S. Fan, X. Li, L. Lei, W. Jia, C. Tang, Y. Li, Z. Huang, Z. Du, J. Yue *et al.*, "An Area-Efficient Lookup-Table-Based eDRAM Digital CIM Macro for Neural Network Inference," *IEEE Journal of Solid-State Circuits*, 2025.
- [12] G. Yin, Y. Chen, M. Lee, X. Du, Y. Ke, W. Tang, Z. Chen, M. Zhou, J. Yue, H. Yang, H. Jia, Y. Liu, and X. Li, "A 28nm 8928Kb/mm²-Weight-Density Hybrid SRAM/ROM Compute-in-Memory Architecture Reducing >95% Weight Loading from DRAM," in *2024 IEEE Custom Integrated Circuits Conference (CICC)*, 2024, pp. 1–2.
- [13] L.-A. Cheong, C. Wang, M. Zhou, W. Tang, Y. Chen, X. Du, Y. Liu, H. Yang, X. Li *et al.*, "A 28nm 166.9 TOPS/W x Mb/mm² DRAM-Free QLC Compute-in-ROM Macro Supporting High Task-Level Inference Energy Efficiency for Tiny AI Edge Devices," in *2024 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, 2024, pp. 1–3.
- [14] B. Yan, J.-L. Hsu, P.-C. Yu, C.-C. Lee, Y. Zhang, W. Yue, G. Mei, Y. Yang, Y. Yang, H. Li, Y. Chen, and R. Huang, "A 1.041-Mb/mm² 27.38-TOPS/W Signed-INT8 Dynamic-Logic-Based ADC-less SRAM Compute-in-Memory Macro in 28nm with Reconfigurable Bitwise Operation for AI and Embedded Applications," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 188–190.
- [15] Y.-W. Chen, R.-H. Wang, Y.-H. Cheng, C.-C. Lu, M.-F. Chang, and K.-T. Tang, "SUN: Dynamic Hybrid-Precision SRAM-Based CIM Accelerator With High Macro Utilization Using Structured Pruning Mixed-Precision Networks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 43, no. 7, pp. 2163–2176, 2024.
- [16] T. Yu, T. Liao, M. Zhou, X. Chu, G. Yin, M. Lee, Y. Liu, H. Yang, and X. Li, "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*, pp. 1–11, 2025.
- [17] Z. Yue, X. Xiang, Y. Wang, R. Guo, H. Han, S. Wei, Y. Hu, and S. Yin, "14.4 A 51.6TFLOPs/W Full-Datapath CIM Macro Approaching Sparsity Bound and ≥ 20 Loss for Compound AI," in *2025 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 68, 2025, pp. 1–3.
- [18] Y. Ma, Y. Meng, Z. Xuan, S. Chen, and Y. Kang, "HNM-CIM: An Algorithm-Hardware Co-designed SRAM-based CIM for Transformer Acceleration Exploiting Hybrid N:M Sparsity," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 30, no. 3, Apr. 2025. [Online]. Available: <https://doi.org/10.1145/3724394>
- [19] T. Yu, Z. Chen, Y. Chen, S. Wang, Y. Liu, H. Yang, and X. Li, "DSC-ROM: A Fully Digital Sparsity-Compressed Compute-in-ROM Architecture for on-Chip Deployment of Large-Scale DNNs," in *2025 Design, Automation Test in Europe Conference (DATE)*, 2025, pp. 1–6.
- [20] H. Fujiwara, H. Mori, Y.-H. Chen, H.-J. Liao, T.-Y. J. Chang *et al.*, "A 5-nm 254-TOPS/W 221-TOPS/mm² Fully-Digital Computing-in-Memory Macro Supporting Wide-Range Dynamic-Voltage-Frequency Scaling and Simultaneous MAC and Write Operations," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 1–3.
- [21] H. Mori, W.-C. Zhao, C.-E. Lee, C.-F. Lee, Y.-H. Hsu, H. Fujiwara, Y. Wang, Y.-D. Chih, Y.-H. Chen, H.-J. Liao, T.-Y. J. Chang *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," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*, 2023, pp. 132–134.
- [22] M. Ezzadeen, A. Majumdar, S. Thomas, J.-P. Noël, B. Giraud, M. Bocquet, F. Andrieu, D. Querlioz, and J.-M. Portal, "Binary ReRAM-based BNN first-layer implementation," in *2023 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2023, pp. 1–6.
- [23] M. Chen, L. Zheng, J.-Y. Lin, P. D. Ye, and H. Li, "Analog Multilevel eDRAM-RRAM CIM for Zeroth-Order Fine-tuning of LLMs," in *2025 IEEE International Memory Workshop (IMW)*, 2025, pp. 1–4.
- [24] Z. Xu, C.-K. Liu, T. Kämpfe, M. Imani, C. Li, C. Zhuo, X. Yin *et al.*, "FeReX: A Reconfigurable Design of Multi-Bit Ferroelectric Compute-in-Memory for Nearest Neighbor Search," in *2024 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2024, pp. 1–6.
- [25] Y. Biyani, R. Bishnoi, T. Spyrou, and S. Hamdioui, "C3CIM: Constant Column Current Memristor-Based Computation-in-Memory Micro-Architecture," in *2025 Design, Automation Test in Europe Conference (DATE)*, 2025, pp. 1–7.
- [26] A. Mamdouh, H. Geng, M. Niemier, X. S. Hu, and D. Reis, "Shared-PIM: Enabling Concurrent Computation and Data Flow for Faster Processing-in-DRAM," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2025.
- [27] G. Yin, Y. Chen, M. Zhou, W. Tang, M. Lee, Z. Yang, T. Liao, X. Du, V. Narayanan, H. Yang, H. Jia, Y. Liu, and X. Li, "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, pp. 1912–1925, 2024.
- [28] C. Szegedy, V. Vanhoucke, Z. Wojna *et al.*, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.
- [29] S. Gongyo, J. Liang *et al.*, "Learning Non-Uniform Step Sizes for Neural Network Quantization," in *Proceedings of the Asian Conference on Computer Vision (ACCV)*, December 2024, pp. 4385–4402.
- [30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [31] T. Yu, T. Liao, M. Zhou, X. Chu, G. Yin, M. Lee, Y. Liu, H. Yang, and X. Li, "DCiROM: A Fully Digital Compute-in-ROM Design Approach to High Energy Efficiency of DNN Inference at Task Level," in *Proceedings of the 30th Asia and South Pacific Design Automation Conference*, ser. ASPDAC'25. Association for Computing Machinery, 2025, p. 100–105.
- [32] C.-F. Lee, C.-H. Lu, C.-E. Lee, H. Mori, Y.-D. Chih, T.-Y. J. Chang *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 *2022 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, 2022, pp. 24–25.
- [33] J. Yue *et al.*, "A 28nm 16.9-300TOPS/W Computing-in-Memory Processor Supporting Floating-Point NN Inference/Training with Intensive-CIM Sparse-Digital Architecture," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*, 2023, pp. 1–3.
- [34] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC)*. IEEE, 2014, pp. 10–14.