

Precision-Reconfigurable Zero-Skipping SRAM Compute-in-Memory Architecture for Energy-Efficient Edge-AI Acceleration

Dr. T. Satyanarayana¹, Dr. Malothu Amru², Dr. Bitla Prabhakar³, Dr. Somala Rama Kishore⁴

Associate Professor, Dept. of Electronics and Communication Engineering^{1,4}

Professor, Dept. of Electronics and Communication Engineering^{2,3}

CMR Engineering College, Hyderabad, TS, India^{1,2,4}

JNTUH University College of Engineering Jagtial, TS, India³

satyant234@gmail.com, bprabhakar2008@gmail.com.

malothuamru@gmail.com, ramki430@gmail.com

Abstract: The energy cost of data movement between static random-access memory (SRAM) and arithmetic units has become a critical limitation in edge artificial-intelligence accelerators. SRAM-based compute-in-memory (CIM) alleviates this bottleneck by executing multiply-and-accumulate operations near or inside the memory array. This work proposes a Precision-Reconfigurable Zero-Skipping SRAM-CIM (PRZS-CIM) architecture that combines bit-serial exact multiply-accumulate decomposition, 4/6/8-bit precision control, row-level zero gating, and all-zero bit-plane suppression. The method preserves exact arithmetic for the selected quantized representation because only mathematically zero terms are gated. A technology-independent architecture-level behavioral evaluation was performed using 20,000 randomly generated 64-element dot products per operating point, with activation sparsity varied from 0% to 80%. At 50% activation sparsity, the 8-bit mode preserves the full-precision numerical result while reducing the proposed cell-evaluation switching proxy by 79.55% relative to an ungated 8-bit bit-serial baseline. The 6-bit mode achieves 89.70% switching-proxy reduction with 3.36% normalized root-mean-square dot-product error relative to the original 8-bit operands, while the 4-bit mode reaches 95.62% reduction with 12.73% NRMSE. At 80% activation sparsity, switching-proxy reductions rise to 91.82%, 95.89%, and 98.25% for 8-, 6-, and 4-bit modes, respectively. The proposed architecture therefore offers a clear precision-energy trade-off while avoiding analog conversion overhead in the conceptual datapath. The reported proposed results are architecture-level simulation results rather than post-layout silicon measurements.

Keywords: SRAM compute-in-memory, VLSI, bit-serial MAC, zero skipping, edge AI, low-power accelerator, precision scaling

I. INTRODUCTION

Deep neural networks and data-intensive signal-processing workloads require large numbers of multiply-and-accumulate (MAC) operations. In conventional von Neumann systems, model parameters are repeatedly transferred from memory to processing elements, and this movement can dominate energy and latency even when the arithmetic itself is highly optimized. SRAM-based compute-in-memory has therefore emerged as an important VLSI direction because it exploits the physical memory array as part of the computing datapath [1].

Early standard-SRAM demonstrations showed that useful machine-learning functions can be embedded directly in memory arrays. Zhang, Wang, and Verma demonstrated a classifier using a standard 6T SRAM array [2], while Kang et al. used simultaneous multirow access and low-swing peripheral processing for a multifunctional in-memory

inference processor [3]. Subsequent SRAM-CIM designs improved throughput and efficiency through binary product-sum operation [4], analog dot-product computation [5], augmented Boolean-compute bitcells [6], multibit dot-product structures [7], programmable bit-serial arithmetic [8], and mixed-signal XNOR or capacitive-coupling macros [9], [12]. The main trade-off in SRAM-CIM is no longer simply whether computation is placed inside the memory. Practical designs must balance precision, memory density, converter overhead, variation tolerance, signed arithmetic, programmability, and neural-network accuracy. Multi-bit approaches such as IMAC [10], Twin-8T CIM [11], CAP-RAM [13], and large-capacity 8-bit CIM macros [14] demonstrate that precision is essential for modern edge workloads. More recent work has also highlighted the substantial area and energy associated with data converters in mixed-signal CIM and explored bit-serial strategies that reduce this overhead [15].

This paper investigates a complementary architectural approach. Instead of depending on analog multiply accumulation, the proposed PRZS-CIM structure uses an exact bit-serial decomposition and reduces switching through operand-dependent row gating and runtime precision selection. Its novelty is the joint use of: (i) exact zero-term suppression at row level, (ii) all-zero bit-plane cycle suppression, (iii) 4/6/8-bit precision-reconfigurable execution, and (iv) a simple digital accumulation path. These mechanisms are evaluated with a reproducible technology-independent activity model so that numerical accuracy and activity reduction are not conflated with unverified post-layout power claims.

II. RELATED WORK AND DESIGN MOTIVATION

A. Evolution of SRAM Compute-in-Memory

The 6T SRAM classifier in [2] demonstrated that memory arrays can execute mixed-signal inference operations while retaining normal storage functionality. DIMA [3] further exploited simultaneous row access in a 16-kB standard 6T SRAM array. Khwa et al. reported a 65-nm 4-kb CIM-SRAM macro for binary DNNs with a reported 55.8 TOPS/W product-sum energy efficiency [4]. CONV-SRAM used 10T cells and embedded dot-product capability for binary-weight CNNs and demonstrated greater than 98% MNIST test accuracy [5].

Boolean and general-purpose approaches broadened the function set. X-SRAM provided in-memory Boolean operations using augmented SRAM cells [6]. Jaiswal et al. showed an 8T SRAM as a multibit dot-product engine [7]. Wang et al. proposed a 28-nm compute SRAM supporting bit-serial integer and floating-point arithmetic, reporting 0.56 TOPS/W for 8-bit multiplication in their programmable system [8]. These works motivate digital or bit-serial schemes when programmability and deterministic arithmetic are important.

Mixed-signal neural-network CIM macros have achieved very high operation efficiency at low precision. XNOR-SRAM reported 403 TOPS/W for ternary-XNOR-accumulate operations [9], while C3SRAM reported 672 TOPS/W for its binary-network compute mode [12]. Multibit designs such as IMAC [10], Twin-8T [11], CAP-RAM [13], and the 384-kb 8-bit macro in [14] address the accuracy and mapping limitations of binary-only computation. The literature therefore shows that precision scalability and data-conversion overhead remain central design issues.

B. Motivation for Zero-Aware Digital CIM

A quantized ReLU activation contains many exact zeros and, even when nonzero, usually contains several zero bits. In a conventional bit-serial datapath, repeatedly exercising rows associated with a streamed zero input bit cannot change the mathematical result. PRZS-CIM treats these zero-valued products as a switching opportunity: the corresponding row evaluation is disabled, and if an entire bit-plane is zero for the active vector, the bit-plane cycle is bypassed. Unlike approximate multipliers, this mechanism introduces no arithmetic error in the selected precision mode. Precision reduction itself is handled explicitly as quantization, which allows the energy-versus-accuracy trade-off to be measured separately.

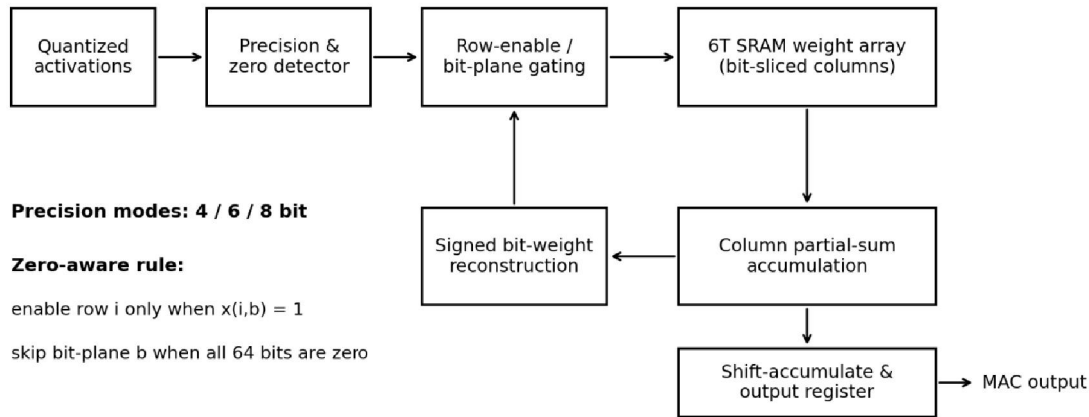


Fig. 1. Proposed PRZS-CIM architecture with precision control, zero-aware row gating, SRAM weight storage, and digital reconstruction.

III. PROPOSED PRZS-CIM ARCHITECTURE

A. Exact Bit-Serial Dot-Product Decomposition

For an N -element dot product, let the post-ReLU activation x_i be an unsigned B_x -bit integer and the weight w_i be a signed B_w -bit integer. The target MAC is

$$S = \sum_{i=0}^{N-1} x_i w_i \quad (1)$$

The activation is decomposed into individual binary planes:

$$x_i = \sum_{\beta=0}^{B_x-1} 2^\beta x_{i,\beta}, \quad x_{i,\beta} \in \{0,1\} \quad (2)$$

Substituting (2) into (1) gives the bit-serial form

$$S = \sum_{\beta=0}^{B_x-1} 2^\beta \left(\sum_{i=0}^{N-1} x_{i,\beta} w_i \right) \quad (3)$$

Equation (3) is exact. When $x_{i,\beta}$ is zero, the corresponding product is identically zero and does not need a row discharge or local accumulation event. The proposed row-enable mask therefore gates only terms that are mathematically inactive. The selected weight precision is represented with ordinary two's-complement bit significance in bit-sliced SRAM columns, and the signed column contributions are combined digitally.

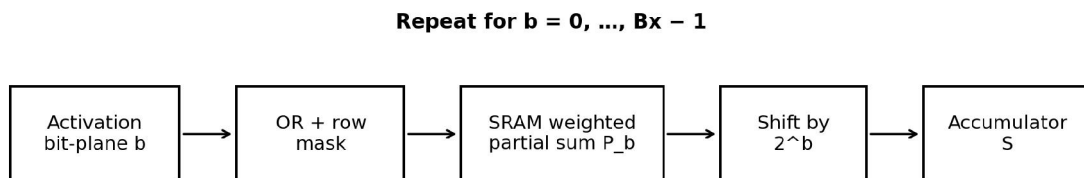


Fig. 2. Exact bit-serial formulation used to justify zero-term gating.

B. Precision-Reconfigurable Operand Representation

Three operating modes are considered: 8-bit, 6-bit, and 4-bit. The 8-bit mode preserves the original quantized operands. Lower-precision modes are produced by uniform rounding and saturation from the 8-bit reference. For a generic unsigned activation, the B-bit quantizer is

$$Q^B(x) = \text{clip}(\text{round}(x / 2^{8-B}), 0, 2^B - 1) \quad (4)$$

Signed weights use the corresponding symmetric two's-complement range. To compare lower-precision results to the original 8-bit arithmetic, the dot product is rescaled by $2^{\{2(8-B)\}}$. Importantly, this quantization error is algorithmic; the CIM computation itself is exact for $Q_B(x)$ and $Q_B(w)$.

C. Row-Level Zero Gating

At activation bit-plane b , a 64-bit row-enable vector is formed from the current activation bits. Only SRAM rows for which $x_{i,b}=1$ participate in the local compute operation. The remaining rows keep their wordline/compute-enable inactive. This eliminates unnecessary dynamic transitions in the row drivers and column evaluation network. The proposed architecture does not require predicting whether a multiplication is small or negligible; gating is performed only for exact zero bit terms.

D. All-Zero Bit-Plane Suppression

A reduction-OR of the row-enable vector determines whether the current bit-plane contains at least one active input bit. If the OR result is zero, the associated memory evaluation, sensing, and accumulation phase is skipped and the controller advances directly to the next plane. This mechanism becomes more useful for sparse activations and for lower precisions, where high-order planes are frequently absent.

E. Activity Model

Because no physical SRAM macro was laid out in this study, the proposed power result is intentionally expressed as a technology-independent switching proxy. An ungated 8-bit bit-serial baseline evaluates N rows across eight activation planes and eight weight bit slices. Its elementary cell-evaluation count is therefore

$$A_{\text{base}} = N \times 8 \times 8 \quad (5)$$

For B-bit PRZS-CIM, the number of row/bit-slice evaluations is proportional to the Hamming weight of the streamed activation words:

$$A_{\text{PRZS}} = B \times \sum_i \text{popcount}(Q^B(x_i)) \quad (6)$$

The reported activity reduction is $1 - E[A_{\text{PRZS}}]/A_{\text{base}}$. This metric captures the savings caused by precision scaling and exact zero-bit gating but does not claim a fabricated technology-node power value.

IV. EXPERIMENTAL METHODOLOGY

A Python behavioral model was used to evaluate arithmetic accuracy, cell-evaluation activity, and active bit-plane cycles. Each test vector contained $N=64$ activation-weight pairs. For every combination of sparsity and precision, 20,000 dot products were simulated. A fixed random seed was used to make the experiment reproducible.

Parameter	Value
Vector length	64 MAC terms
Trials per operating point	20,000
Activation reference precision	Unsigned 8-bit
Weight reference precision	Signed 8-bit two's complement

Activation distribution	Zero-masked positive Gamma distribution, clipped to 0...255
Weight distribution	Zero-mean Gaussian distribution, clipped to -127...127
Activation sparsity	0%, 30%, 50%, 70%, 80%
PRZS precision modes	8 bit, 6 bit, 4 bit
Reference arithmetic	Exact 8-bit integer dot product
Primary metrics	NRMSE, correlation, active bit-plane cycles, switching-proxy reduction

Table I. Behavioral evaluation setup.

NRMSE is computed from the rescaled PRZS output and the original 8-bit dot product. Correlation is the Pearson correlation coefficient between the two output vectors. In the 8-bit mode, the only optimization is zero-aware gating, so the numerical output must match the exact reference. The 6-bit and 4-bit modes intentionally trade quantization accuracy for lower activity.

V. RESULTS AND DISCUSSION

A. Full-Precision Exactness and Activity Reduction

At 50% activation sparsity, the 8-bit PRZS mode produces zero NRMSE and unit correlation with the exact 8-bit reference, confirming that row gating and bit-plane suppression do not alter the arithmetic result. Under the defined activity model, the average cell-evaluation switching proxy is reduced by 79.55% relative to an ungated 8-bit bit-serial baseline. This reduction comes only from suppressing zero-valued activation bits; no multiplication term with value one is discarded.

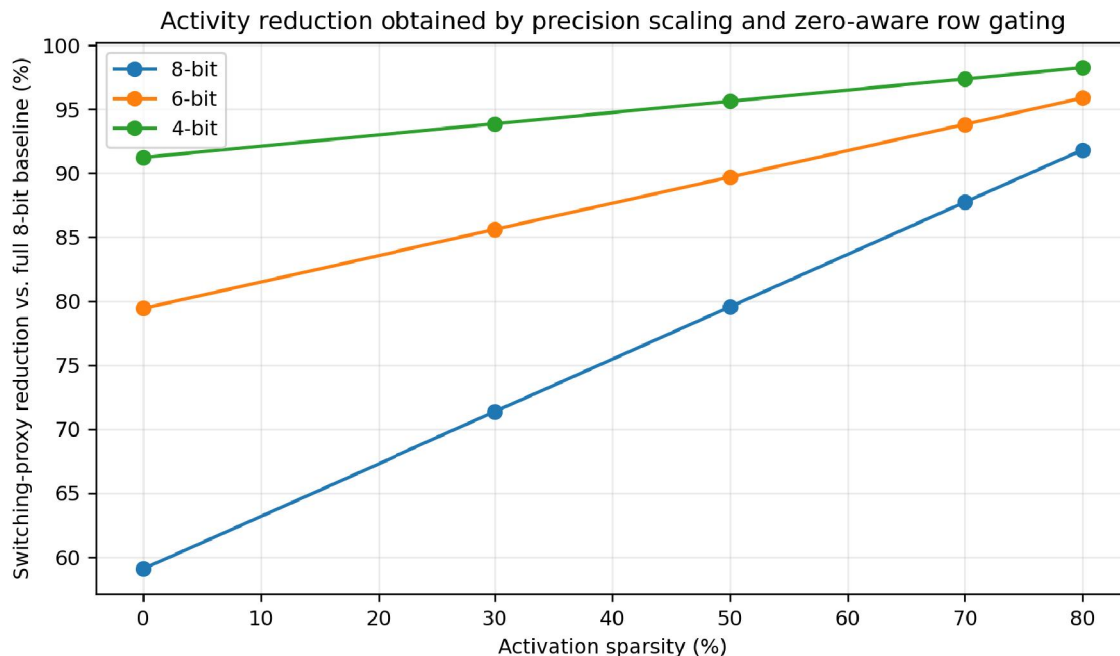


Fig. 3. Switching-proxy reduction versus activation sparsity for the three precision modes.

B. Precision–Accuracy Trade-Off

Precision	NRMSE (%)	Correlation	Switching-proxy reduction (%)	Avg. active cycles
8	0.000	1.00000	79.55	7.954
6	3.361	0.99944	89.70	5.960
4	12.734	0.99192	95.62	3.974

Table II. Proposed PRZS-CIM results at 50% activation sparsity.

The 6-bit mode provides a useful middle point: the switching proxy falls by nearly 90% at 50% sparsity while the dot-product NRMSE remains approximately 3.36% and output correlation remains above 0.999. The 4-bit mode reduces the activity proxy further, but its approximately 12.7% NRMSE shows that aggressive precision reduction should be selected only for layers or workloads that tolerate stronger quantization. The 8-bit mode demonstrates that the zero-skipping mechanism itself is lossless.

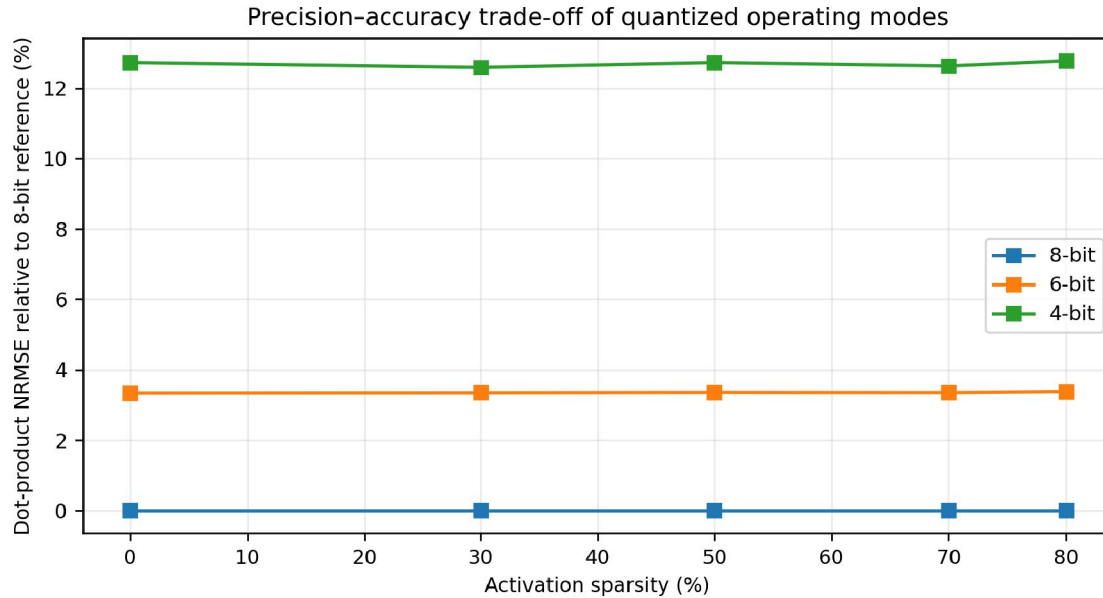


Fig. 4. NRMSE relative to the original 8-bit dot product.

C. Effect of Activation Sparsity

Higher activation sparsity directly increases the opportunity for zero-bit gating. At 80% sparsity, the switching-proxy reductions reach 91.82%, 95.89%, and 98.25% for 8-, 6-, and 4-bit operation, respectively. This result is particularly relevant to ReLU-based networks, where activation sparsity is common. The improvement is monotonic because the number of active row/bit-slice evaluations is proportional to the activation Hamming weight.

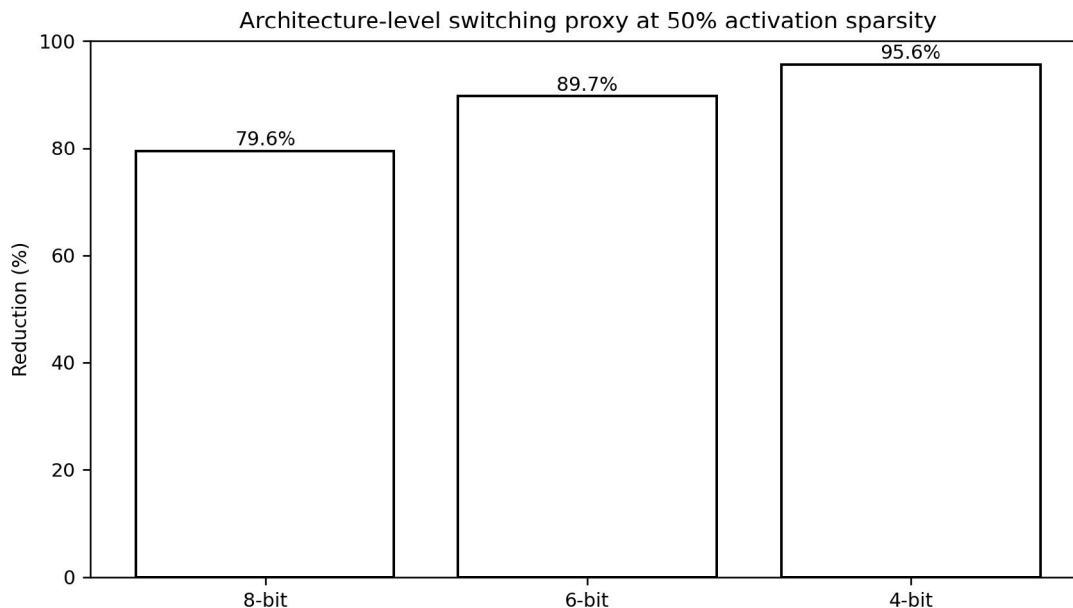


Fig. 5. Switching-proxy reduction at 50% activation sparsity.

D. Bit-Plane Cycle Suppression

Row-level gating primarily reduces switching, whereas all-zero bit-plane suppression can also reduce execution cycles. For a 64-element vector, completely empty bit-planes are uncommon at low sparsity, so the average cycle count remains close to the nominal precision. As sparsity rises, high-order bit-planes become more frequently empty. At 80% sparsity, the average active bit-plane count is 7.67 cycles in 8-bit mode, 5.69 cycles in 6-bit mode, and 3.75 cycles in 4-bit mode.

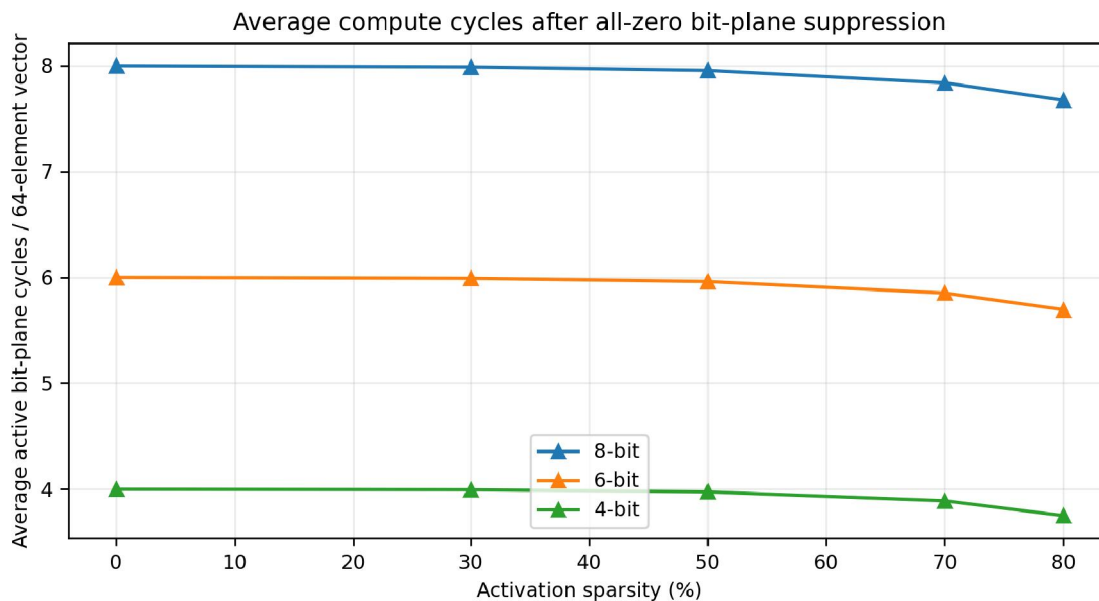


Fig. 6. Average active bit-plane cycles after all-zero-plane suppression.

E. Complete Architecture-Level Result Matrix

Sparsity (%)	Bits	NRMSE (%)	Correlation	Activity reduction (%)	Cycles
0	8	0.000	1.00000	59.11	7.998
0	6	3.342	0.99944	79.43	5.999
0	4	12.734	0.99192	91.24	3.999
30	8	0.000	1.00000	71.38	7.987
30	6	3.349	0.99944	85.61	5.990
30	4	12.595	0.99209	93.87	3.994
50	8	0.000	1.00000	79.55	7.954
50	6	3.361	0.99944	89.70	5.960
50	4	12.734	0.99192	95.62	3.974
70	8	0.000	1.00000	87.74	7.837
70	6	3.354	0.99944	93.82	5.850
70	4	12.637	0.99203	97.37	3.890
80	8	0.000	1.00000	91.82	7.671
80	6	3.384	0.99943	95.89	5.693
80	4	12.783	0.99184	98.25	3.750

Table III. Complete proposed architecture-level results. Each row uses 20,000 64-element dot products.

F. Comparison with Representative Pre-2024 SRAM-CIM Literature

Table IV summarizes representative SRAM-CIM research used in the literature review. The figures for prior work are the reported results of the cited publications and are not directly comparable across technology nodes, precisions, memory capacities, or operation definitions. The proposed row is intentionally marked as an architecture-level model because this paper does not claim fabricated-silicon TOPS/W.

Work	Year	Process	Cell	Precision / function	Reported result	Evidence
Zhang et al. [2]	2017	130 nm	6T	Classifier	630 pJ/decision	Silicon
Khwa et al. [4]	2018	65 nm	6T	Binary PS	55.8 TOPS/W	Silicon
CONV-SRAM [5]	2019	65 nm	10T	Binary W, 6-b I/O	>98% MNIST	Silicon
Wang et al. [8]	2020	28 nm	8T	8-b multiplication	0.56 TOPS/W	Silicon/system

XNOR-SRAM [9]	2020	65 nm	custom	Ternary XAC	403 TOPS/W	Silicon
Twin-8T [11]	2020	55 nm	Twin-8T	1/2/4-b input	37.5–45.36 TOPS/W	Silicon
C3SRAM [12]	2020	65 nm	custom	Binary MAC	672 TOPS/W	Silicon
CAP-RAM [13]	2021	65 nm	6T + charge MAC	programmable	49.4 TOPS/W	Silicon
Su et al. [14]	2021	28 nm	6T	8-b precision	384-kb macro	Silicon
Sehgal et al. [15]	2023	—	SRAM CIM	bit-serial DNN	converter-aware	Silicon/JSSC
Proposed PRZS-CIM	This work	Technology-independent	6T conceptual	4/6/8-b	79.55% activity reduction at 8 b, 50% sparsity	Behavioral model

Table IV. Representative SRAM-CIM literature and positioning of the proposed architecture-level study.

G. Interpretation of the Results

The principal result is that precision scaling and zero gating act on different sources of inefficiency. Zero gating exploits data sparsity without changing the arithmetic, while precision scaling reduces the number of bit slices and bit-serial planes at the cost of quantization error. This separation enables a controller to retain 8-bit precision for sensitive layers and switch to 6-bit or 4-bit execution where model calibration permits.

The activity proxy should not be interpreted as post-layout power. SRAM bitline capacitance, wordline drivers, sense amplifiers, clocking, control logic, and data movement all contribute to real energy. Nevertheless, the cell-evaluation count is a useful architecture-level indicator because the suppressed operations correspond to memory compute events that would otherwise toggle internal nodes. A transistor-level macro implementation would be needed to convert these reductions into measured fJ/op or TOPS/W.

Compared with mixed-signal CIM, a digital bit-serial organization avoids a high-resolution ADC in the core conceptual datapath and naturally preserves deterministic integer arithmetic. The trade-off is additional serial cycles. PRZS-CIM partially compensates for this serial latency by suppressing all-zero planes and allows precision to be reduced dynamically.

VI. IMPLEMENTATION CONSIDERATIONS

A. SRAM Bitcell and Peripheral Compatibility

The conceptual architecture assumes a standard 6T storage core with compute-enable control added at the wordline/peripheral level rather than a mandatory analog modification of every cell. A practical implementation can organize signed weight bit slices across neighboring columns and use local digital population/accumulation circuitry. The exact circuit realization will determine read stability, area overhead, and the achievable frequency.

B. Signed Weights

For two's-complement B-bit weights, the most significant bit has negative significance. The column reconstruction block therefore adds bit slices 0 through B–2 with positive powers of two and subtracts the sign-bit contribution. This keeps the row gating rule unchanged because the streamed activation bit remains binary.

C. Precision Scheduling

Precision mode can be selected statically per layer or dynamically by software. A conservative deployment would profile validation accuracy at 8, 6, and 4 bits and assign the lowest safe precision to each layer. The hardware controller then masks unused high-order bit slices and shortens the activation bit-plane loop.

D. Limitations

The study does not contain a foundry SRAM macro, post-layout parasitics, Monte Carlo variation analysis, or measured TOPS/W. The activation and weight distributions are synthetic rather than taken from a particular trained network. Consequently, the results establish the functional behavior and the architecture-level activity trend, not final silicon efficiency. A publication intended to claim physical power, delay, and area must implement the proposed macro in a defined CMOS PDK and report SPICE/post-layout measurements.

VII. CONCLUSION

This paper presented PRZS-CIM, a precision-reconfigurable zero-skipping SRAM compute-in-memory architecture for edge-AI MAC workloads. The approach uses an exact bit-serial formulation so that zero-valued activation bits can be gated without numerical approximation. A 4/6/8-bit precision controller further scales the number of active memory bit slices and compute cycles. Architecture-level evaluation across 300,000 simulated 64-element dot products shows that the full 8-bit mode retains exact arithmetic while reducing the defined switching proxy by 79.55% at 50% activation sparsity and 91.82% at 80% sparsity. The 6-bit mode provides a balanced operating point with approximately 3.36% NRMSE at 50% sparsity and 89.70% switching-proxy reduction, whereas the 4-bit mode provides stronger activity reduction at a larger quantization penalty. These results motivate a future transistor-level SRAM macro in which row gating, precision control, signed reconstruction, and memory periphery are co-designed and characterized for area, delay, and measured energy efficiency.

REFERENCES

- [1] N. Verma, H. Jia, H. Valavi, Y. Tang, M. Ozatay, L.-Y. Chen, B. Zhang, and P. Deaville, "In-Memory Computing: Advances and Prospects," IEEE Solid-State Circuits Magazine, vol. 11, no. 3, pp. 43–55, 2019, doi: 10.1109/MSSC.2019.2922889.
- [2] J. Zhang, Z. Wang, and N. Verma, "In-Memory Computation of a Machine-Learning Classifier in a Standard 6T SRAM Array," IEEE Journal of Solid-State Circuits, vol. 52, no. 4, pp. 915–924, Apr. 2017, doi: 10.1109/JSSC.2016.2642198.
- [3] M. Kang, S. K. Gonugondla, A. Patil, and N. R. Shanbhag, "A Multi-Functional In-Memory Inference Processor Using a Standard 6T SRAM Array," IEEE Journal of Solid-State Circuits, vol. 53, no. 2, pp. 642–655, Feb. 2018, doi: 10.1109/JSSC.2017.2782087.
- [4] W.-S. Khwa et al., "A 65nm 4Kb Algorithm-Dependent Computing-in-Memory SRAM Unit-Macro with 2.3ns and 55.8TOPS/W Fully Parallel Product-Sum Operation for Binary DNN Edge Processors," in 2018 IEEE International Solid-State Circuits Conference (ISSCC), 2018, pp. 496–498, doi: 10.1109/ISSCC.2018.8310401.
- [5] A. Biswas and A. P. Chandrakasan, "CONV-SRAM: An Energy-Efficient SRAM With In-Memory Dot-Product Computation for Low-Power Convolutional Neural Networks," IEEE Journal of Solid-State Circuits, vol. 54, no. 1, pp. 217–230, Jan. 2019, doi: 10.1109/JSSC.2018.2880918.
- [6] A. Agrawal, A. Jaiswal, C. Lee, and K. Roy, "X-SRAM: Enabling In-Memory Boolean Computations in CMOS Static Random Access Memories," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 65, no. 12, pp. 4219–4232, Dec. 2018, doi: 10.1109/TCSI.2018.2848999.
- [7] A. Jaiswal, I. Chakraborty, A. Agrawal, and K. Roy, "8T SRAM Cell as a Multibit Dot-Product Engine for Beyond Von Neumann Computing," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, no. 11, pp. 2556–2567, Nov. 2019, doi: 10.1109/TVLSI.2019.2929245.

- [8] J. Wang, X. Wang, C. Eckert, A. Subramaniyan, R. Das, D. Blaauw, and D. Sylvester, "A 28-nm Compute SRAM With Bit-Serial Logic/Arithmetic Operations for Programmable In-Memory Vector Computing," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 1, pp. 76–86, Jan. 2020, doi: 10.1109/JSSC.2019.2939682.
- [9] S. Yin, Z. Jiang, J.-S. Seo, and M. Seok, "XNOR-SRAM: In-Memory Computing SRAM Macro for Binary/Ternary Deep Neural Networks," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 6, pp. 1733–1743, Jun. 2020, doi: 10.1109/JSSC.2019.2963616.
- [10] M. F. Ali, A. Jaiswal, S. Kodge, A. Agrawal, I. Chakraborty, and K. Roy, "IMAC: In-Memory Multi-Bit Multiplication and ACcumulation in 6T SRAM Array," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 8, pp. 2521–2531, Aug. 2020, doi: 10.1109/TCSI.2020.2981901.
- [11] X. Si et al., "A Twin-8T SRAM Computation-in-Memory Unit-Macro for Multibit CNN-Based AI Edge Processors," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 1, pp. 189–202, Jan. 2020, doi: 10.1109/JSSC.2019.2952773.
- [12] Z. Jiang, S. Yin, J.-S. Seo, and M. Seok, "C3SRAM: An In-Memory-Computing SRAM Macro Based on Robust Capacitive Coupling Computing Mechanism," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 7, pp. 1888–1897, Jul. 2020, doi: 10.1109/JSSC.2020.2992886.
- [13] Z. Chen et al., "CAP-RAM: A Charge-Domain In-Memory Computing 6T-SRAM for Accurate and Precision-Programmable CNN Inference," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 6, pp. 1924–1935, Jun. 2021, doi: 10.1109/JSSC.2021.3056447.
- [14] J.-W. Su et al., "16.3 A 28nm 384kb 6T-SRAM Computation-in-Memory Macro with 8b Precision for AI Edge Chips," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, 2021, pp. 250–252, doi: 10.1109/ISSCC42613.2021.9365984.
- [15] R. Sehgal, T. Thareja, S. Xie, C. Ni, and J. P. Kulkarni, "A Bit-Serial, Compute-in-SRAM Design Featuring Hybrid-Integrating ADCs and Input Dependent Binary Scaled Precharge Eliminating DACs for Energy-Efficient DNN Inference," *IEEE Journal of Solid-State Circuits*, vol. 58, no. 7, pp. 2109–2124, Jul. 2023, doi: 10.1109/JSSC.2023.3235210.