SRAM Compute-in-Memory Macro with Dual-Dataflow Architecture for Efficient Support of Multi-Modal Transformers and CNNs

Tianqi Wang[*]
ECE Department
National University of Singapore
Singapore
tianqi@u.nus.edu

Zhongheng Xie[*]
ECE Department
National University of Singapore
Singapore
e0973821@u.nus.edu

Massimo Alioto
ECE Department
National University of Singapore
Singapore
massimo.alioto@nus.edu.sg

***Abstract*—This work presents a SRAM compute-in-memory macro with a dual-dataflow architecture that enables both uninterrupted static and dynamic-input matrix multiplication and accumulation (MAC) for the first time. This uniquely eliminates the traditionally heavy inter-bank load-store overhead of dynamic MACs in all prior SRAM art, where one of the two operands is required to statically reside into the array. The dual-dataflow architecture efficiently supports both CNNs and (multi-modal) transformers.**

Unified and efficient static and dynamic MACs are enabled by the proposed 1) reset-less in-bitcell Boolean computations, 2) hybrid-domain accumulator, 3) concurrent write/compute. A 28-nm 160-kb SRAM testchip shows a competitive energy efficiency of up to 138.2 (30.5) TOPS/W for 8-bit integer S-MAC (D-MAC) at full output precision.

***Keywords*—AI, compute-in-memory, transformer models.**

I. Introduction

SRAM compute-in-memory (CIM) enables sizeable energy efficiency improvements in AI systems, thanks to superior data locality enforcement [1]. However, even in SRAM macros the latter is not preserved under workloads requiring frequent data updates and inter-bank load/store. Indeed, workloads in CNNs and attention vector/matrix computation in transformers are naturally supported by static MACs (S-MAC, Fig. 1) with weights being statically stored. Conversely, attention score computation in transformers involves dynamic MACs (D-MAC) where both inputs need to change at every new computation.

Prior SRAM compute-in-memory macros perform only S-MACs, negating true data locality in D-MACs and full bank utilization for computations due to the need for preliminary load (i.e., static storage) of 1) one input into the array performing the computation, 2) the other input on another bank [2]. Though transposable buffer/CIM can partially mitigate such overhead [2], recent SRAM designs still execute D-MACs by repurposing S-MAC schemes, inevitably introducing extra interruptions and energy overheads (generally not included in prior energy efficiency evaluations) due to reloads [3], [4]. In key models such as multi-modal transformers, both S- and D-MACs substantially contribute to system energy and throughput, in view of their comparable number of operations (e.g., 30% D-MAC in [5]) and dominant memory accesses (e.g., 83% memory accesses in attention score [5]). This motivates the introduction of a dual-dataflow architecture that efficiently supports both S- and D-MACs, posing three challenges: 1) mitigate energy in data update/read/transfer, 2) reduce area overhead of S- and D-MAC unification, 3) reduce system interruption cycles.

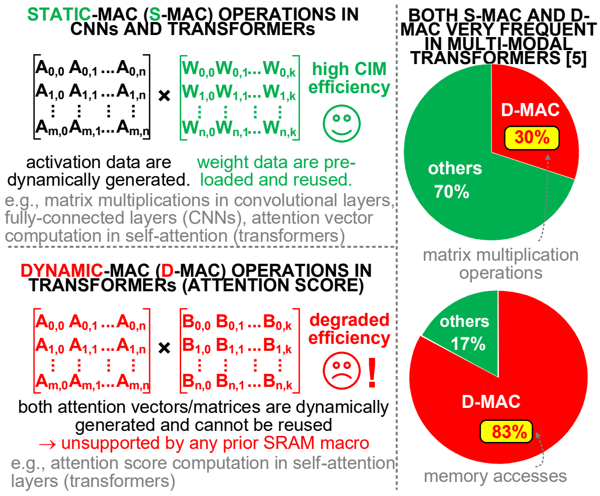

STATIC-MAC (S-MAC) OPERATIONS IN CNNs AND TRANSFORMERs
A0,0 A0,1 ... A0,n
A1,0 A1,1 ... A1,n
Am,0 Am,1 ... Am,n
×
W0,0 W0,1 ... W0,k
W1,0 W1,1 ... W1,k
Wn,0 Wn,1 ... Wn,k
high CIM efficiency
activation data are dynamically generated.
weight data are pre-loaded and reused.
e.g., matrix multiplications in convolutional layers, fully-connected layers (CNNs), attention vector computation in self-attention (transformers)
BOTH S-MAC AND D-MAC VERY FREQUENT IN MULTI-MODAL TRANSFORMERS [5]
D-MAC 30%
others 70%
matrix multiplication operations
DYNAMIC-MAC (D-MAC) OPERATIONS IN TRANSFORMERs (ATTENTION SCORE)
A0,0 A0,1 ... A0,n
A1,0 A1,1 ... A1,n
Am,0 Am,1 ... Am,n
×
B0,0 B0,1 ... B0,k
B1,0 B1,1 ... B1,k
Bn,0 Bn,1 ... Bn,k
degraded efficiency
!
both attention vectors/matrices are dynamically generated and cannot be reused
→ unsupported by any prior SRAM macro
e.g., attention score computation in self-attention layers (transformers)
others 17%
D-MAC 83%
memory accesses

Fig. 1. Motivation for compute-in-memory designs with dual-dataflow architecture to efficiently support both static MAC (S-MAC) and dynamic MAC (D-MAC) operations in CNNs and transformers.

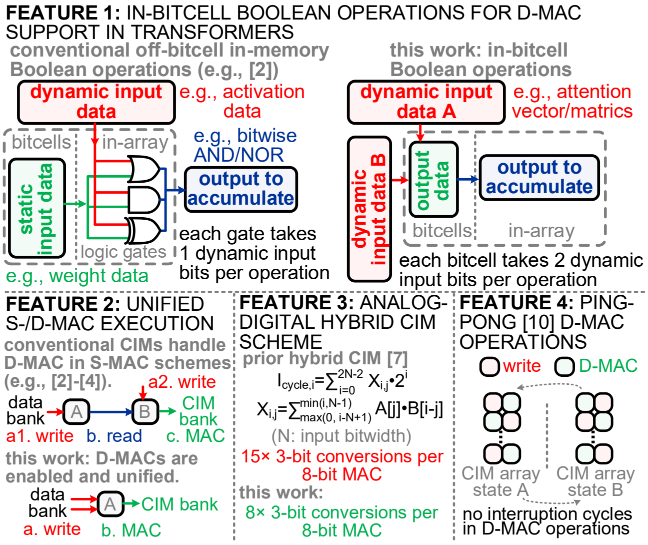

FEATURE 1: IN-BITCELL BOOLEAN OPERATIONS FOR D-MAC SUPPORT IN TRANSFORMERS
conventional off-bitcell in-memory Boolean operations (e.g., [2])
dynamic input data
e.g., activation data
bitcells
in-array
e.g., bitwise AND/NOR
static input data
output to accumulate
logic gates
each gate takes 1 dynamic input bits per operation
e.g., weight data
this work: in-bitcell Boolean operations
dynamic input data A
e.g., attention vector/matrics
dynamic input data B
output data
output to accumulate
bitcells
in-array
each bitcell takes 2 dynamic input bits per operation
FEATURE 2: UNIFIED S-/D-MAC EXECUTION
conventional CIMs handle D-MAC in S-MAC schemes (e.g., [2]-[4]).
data bank
A
B
CIM bank
a1. write
a2. write
b. read
c. MAC
this work: D-MACs are enabled and unified.
data bank
A
CIM bank
a. write
b. MAC
FEATURE 3: ANALOG-DIGITAL HYBRID CIM SCHEME
prior hybrid CIM [7]
$I_{cycle,i}=\sum_{i=0}^{2N-2} X_{i,j}\cdot 2^i$
$X_{i,j}=\sum_{max(0,\,i-N+1)}^{min(i,N-1)} A[j]\cdot B[i-j]$
(N: input bitwidth)
15× 3-bit conversions per 8-bit MAC
this work:
8× 3-bit conversions per 8-bit MAC
FEATURE 4: PING-PONG [10] D-MAC OPERATIONS
write
D-MAC
CIM array state A
CIM array state B
no interruption cycles in D-MAC operations

Fig. 2. Limitations in prior SRAM compute-in-memory architectures, innovation and advances enabled by the proposed SRAM CIM architecture.

This work introduces an input-versatile CIM with dual-dataflow architecture (Fig. 2), as enabled by: 1) in-bitcell 1-bit Boolean computations to support D-MAC, while suppressing the energy cost and the interruption cycles of conventional reset, 2) unified S- and D-MAC support for efficient and flexible workload execution, 3) analog-digital hybrid computation with competitive energy, 4) time-interleaved D-MAC for concurrent write/compute.

II. Proposed Architecture and Circuit Techniques

In the proposed architecture (Fig. 3), each CIM bank

[*] Equally credited authors.

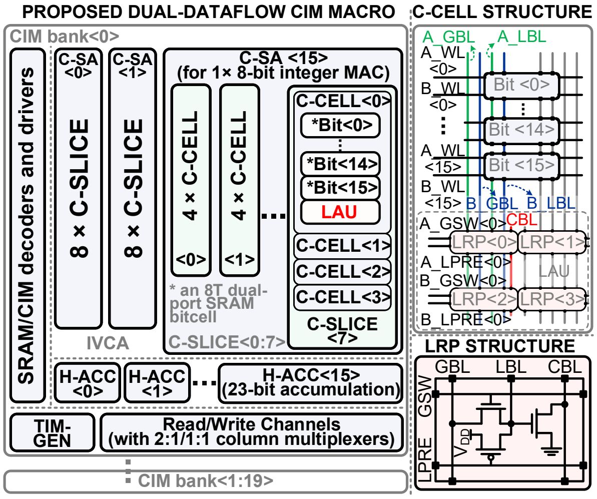


Fig. 3. Proposed dual-dataflow CIM architecture with input-versatile computing arrays (left), bitcell/circuit arrangement in a compute cell C-CELL (top-right), and local read port connections (bottom-right).

comprises an input-versatile computing array (IVCA), read and write channels with 2:1 and 1:1 column multiplexers, 16 hybrid accumulators (H-ACC), and a timing generator (TIM-GEN). The IVCA consists of 16 integer MAC computing sub-arrays (C-SA), each with 8 computing slices (C-SLICE). Each C-SLICE consists of 4 compute cells (C-CELL), each housing 16 bitcells and a local accumulation unit (LAU) containing 4 local read ports (LRP) for current accumulation.

In D-MACs (Fig. 4), each bank performs 8-bit unsigned integer multiplication/accumulation for two dynamic input streams. In *phase 1*, currently active C-CELLs in an IVCA compute in bitcell and store 16×8×8 1-bit multiplications (equivalent to 16 8-bit multiplications, see below) of a row- (wordline) and a column-wise (bitline) input. The above 1-bit in-bitcell multiplications are based on 8T dual-port bitcells (e.g., [6]), which uniquely process two dynamic 1-bit inputs without pre-load/reset (Fig. 5). Reset-less Boolean operation is evidenced in Fig. 5, as the output depends only on the current inputs. The Boolean function (e.g., AND) is straightforwardly configured via simple bitline digital setting, eliminating extra routing and in-array e.g., gates [2]) or off-array extra resources (e.g., multipliers [7]). Since computation takes inputs directly from bitlines and wordlines, it suppresses inter-bank data transfers and array reloads, in contrast to prior CIMs (e.g., [2]-[4]). As opposed to prior in-bitcell Boolean operations [8], [9], the proposed reset-less operation avoids any interruption cycle, and drastically reduces bitcell bit-flips (i.e., energy), inducing high sparsity (e.g., 98% at 90% input sparsity for adopted AND). In *phase 2*, the 8 partial results from all 4 C-CELLs within 128 C-SLICEs are accumulated by the LAUs in the form of cumulative compute-bitline current (CBL) for 128×8 simultaneous accumulations/bank of partial results. In *phase 3*, the CIM currents are digitized in H-ACCs and then systolically aggregated into 23-bit MACs at full output precisions (Fig. 6).

In S-MACs (Fig. 4), each bank performs 8-bit unsigned integer MAC for one dynamic input stream with stationary weights. In *phase 1*, each C-CELL is conventionally pre-loaded with partial 1-bit stationary inputs. In *phase 2*, all C-CELLs perform two multiplications between two row-wise 1-bit partial dynamic inputs and two stationary 1-bit partial static inputs. Accumulated output currents are converted and aggregated in *Phase 3*.

In the array periphery, the efficient hybrid scheme in Fig. 6 suppresses ADC activity per MAC by 1.8× over [7] by

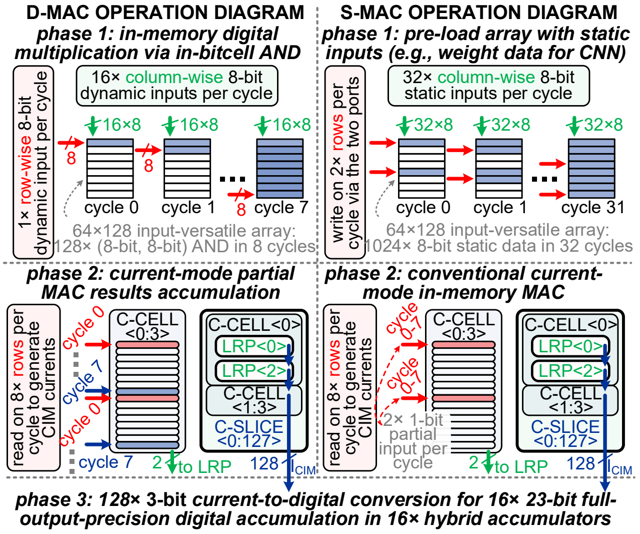
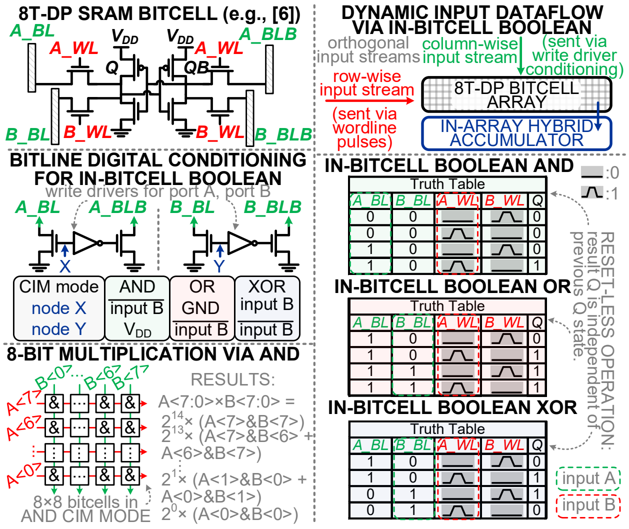

D-MAC OPERATION DIAGRAM
phase 1: in-memory digital multiplication via in-bitcell AND
1× row-wise 8-bit dynamic input per cycle
16× column-wise 8-bit dynamic inputs per cycle
16×8
16×8
16×8
8
8
8
cycle 0
cycle 1
cycle 7
64×128 input-versatile array: 128× (8-bit, 8-bit) AND in 8 cycles
S-MAC OPERATION DIAGRAM
phase 1: pre-load array with static inputs (e.g., weight data for CNN)
write on 2× rows per cycle via the two ports
32× column-wise 8-bit static inputs per cycle
32×8
32×8
32×8
cycle 0
cycle 1
cycle 31
64×128 input-versatile array: 1024× 8-bit static data in 32 cycles
phase 2: current-mode partial MAC results accumulation
read on 8× rows per cycle to generate CIM currents
cycle 0
cycle 7
cycle 0
C-CELL <0:3>
C-CELL<0>
LRP<0>
LRP<2>
C-CELL <1:3>
C-SLICE <0:127>
cycle 7
2
to LRP
128
I_CIM
phase 2: conventional current-mode in-memory MAC
read on 8× rows per cycle to generate CIM currents
cycle 0-1
cycle 0-1
2× 1-bit partial input per cycle
C-CELL <0:3>
C-CELL<0>
LRP<0>
LRP<2>
C-CELL <1:3>
C-SLICE <0:127>
2
to LRP
128
I_CIM
phase 3: 128× 3-bit current-to-digital conversion for 16× 23-bit full-output-precision digital accumulation in 16× hybrid accumulators

Fig. 4. Dual-dataflow operation: D-MACs (left) and S-MACs (right) are unified into the same macro. Reloads in conventional S-MACs repurposed for D-MACs are here eliminated (left).


8T-DP SRAM BITCELL (e.g., [6])
A_BL
A_WL
VDD
Q
VDD
QB
A_WL
A_BLB
B_BL
B_WL
B_WL
B_BLB
DYNAMIC INPUT DATAFLOW VIA IN-BITCELL BOOLEAN
orthogonal input streams
column-wise input stream
(sent via write driver conditioning)
row-wise input stream
(sent via wordline pulses)
8T-DP BITCELL ARRAY
IN-ARRAY HYBRID ACCUMULATOR
BITLINE DIGITAL CONDITIONING FOR IN-BITCELL BOOLEAN
write drivers for port A, port B
A_BL
A_BLB
B_BL
B_BLB
X
Y
CIM mode
node X
node Y
AND
input B
VDD
OR
GND
input B
XOR
input B
input B
8-BIT MULTIPLICATION VIA AND
B<0>
B<6>
B<7>
A<7>
A<6>
A<0>
&
8×8 bitcells in AND CIM MODE
RESULTS:
A<7:0>×B<7:0> =
2^{14}× (A<7>&B<7>) +
2^{13}× (A<7>&B<6> + A<6>&B<7>)
2^{1}× (A<1>&B<0> + A<0>&B<1>)
2^{0}× (A<0>&B<0>)
IN-BITCELL BOOLEAN AND
▁ :0
⎍ :1
RESET-LESS OPERATION: result Q is independent of previous Q state.
IN-BITCELL BOOLEAN OR
IN-BITCELL BOOLEAN XOR
input A
input B

IN-BITCELL BOOLEAN AND

A_BL	B_BL	A_WL	B_WL	Q
0	0	▁	⎍	0
0	0	⎍	▁	0
1	0	▁	⎍	0
1	0	⎍	▁	1

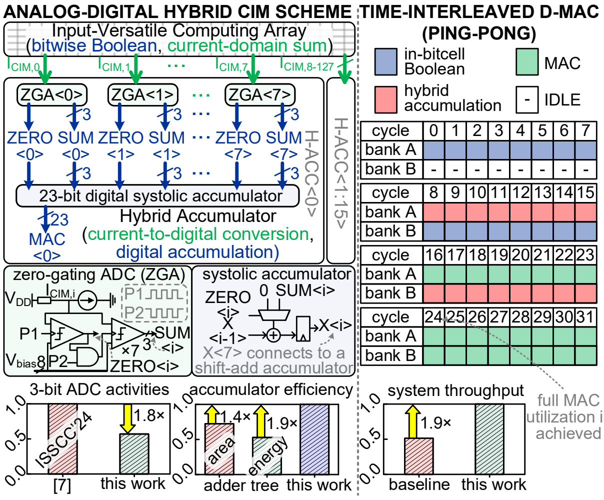
IN-BITCELL BOOLEAN OR

A_BL	B_BL	A_WL	B_WL	Q
1	0	▁	⎍	0
1	0	⎍	▁	1
1	1	▁	⎍	1
1	1	⎍	▁	1

IN-BITCELL BOOLEAN XOR

A_BL	B_BL	A_WL	B_WL	Q
1	0	▁	⎍	0
1	0	⎍	▁	1
0	1	▁	⎍	1
0	1	⎍	▁	0

Fig. 5. Proposed in-bitcell Boolean computations, bitline digital conditioning in CIM mode (center-left), multiplication via in-bitcell AND (bottom-left), truth tables for in-bitcell computations (right).



cycle	0	1	2	3	4	5	6	7
bank A	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
bank B	-	-	-	-	-	-	-	-

cycle	8	9	10	11	12	13	14	15
bank A	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥
bank B	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦

cycle	16	17	18	19	20	21	22	23
bank A	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
bank B	🟥	🟥	🟥	🟥	🟥	🟥	🟥	🟥

cycle	24	25	26	27	28	29	30	31
bank A	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
bank B	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩

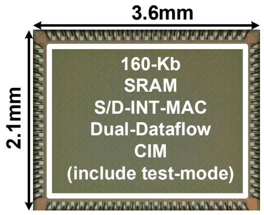
Fig. 6. Hybrid analog-digital accumulation (top-left), time-interleaved D-MACs via ping-pong scheduling (top-right), and comparison with prior baseline designs (conventional adder tree, no time interleaving, bottom).

accumulating 8 (instead of 1) partial multiplication results residing in same bit position. Periphery is pitch-matched to the minimum poly pitch of the bitcell. The 23-bit digital systolic accumulator (Fig. 6) improves area and energy efficiency by 1.9× and 1.4× compared with adder trees. Concurrent write/compute in D-MACs is enabled via ping-pong scheduling [10], improving throughput by 1.9×.

III. Measurement Results

A testchip containing a 28-nm 20-bank SRAM macro with 160-Kb total capacity (Figs. 7-8) exhibits a compute (read) access time of 5 ns (2 ns) at V_{DD}=1.05 V for 8-bit integer S- and D-MACs with 128 accumulations into 23-bit full output precision, as shown in Fig. 9.

In S-MACs, state-of-the-art peak energy efficiency of 138.2 TOPS/W is achieved, improving prior art by 1.2-2.0× [4], [7], [11], [13], [14]. In D-MACs, peak energy efficiency is 30.5 TOPS/W at typical 90% input sparsity. Fair comparison with prior CIM executing transformers [4] shows an energy efficiency improvement by 1.4×. At the same time, the FoM used in [7] is improved by 1.3-2.0× compared to [2], [4], [12], [14], which simply repurpose S-MACs due to D-MAC unavailability, while not reporting the additional load-store energy required. When accounting for the latter one (not needed in proposed dual-dataflow architecture), the actual improvement of the proposed SRAM macro is even more pronounced.



CHIP SUMMARY		
technology	28nm CMOS	
macro memory capacity	160Kb	
supply voltage (V)	0.6 – 1.05	
read (write) energy (pJ) [1]	9.23 (9.55)	
read (compute) access time (ns)	2 [2] – 13.5 [3] (5 [2] – 33 [3])	
input precision (bit)	8	
weight precision (bit)	8	
number of accumulations	128	
output precision (bit)	23	
output ratio	1	
unified MAC operations	S-MAC	D-MAC
average energy efficiency (TOPS/W) [4]	11.6 [6] – 35.5 [7]	8.3 [6] – 25.9 [7]
peak energy efficiency (TOPS/W) [5]	47.2 [6] – 138.2 [7]	9.7 [6] – 30.5 [7]
area efficiency (GOPS/mm^2)	22.1 – 305.8	
inference accuracy loss [8] (ResNet50@ImageNet1K)	0.6%	
inference accuracy loss [9] (ViT-Base@ImageNet1K)	1.1%	

[1] Measured at 500MHz, 1.05V, 25°C. [2] Measured at 1.05V, 25°C. [3] Measured at 0.60V, 25°C. [4] Measured with average case under 50% input/weight sparsity. [5] Measured with average case under 90% input/weight sparsity. [6] Measured under 0.6V at 30MHz, 25°C. [7] Measured under 1.05V at 200MHz, 25°C. [8] Using ResNet50 model and the software baseline (FP32) was 76.1%. [9] Using ViT-Base model and the software baseline (FP32) was 81.0%.

Fig. 7. Chip micrograph in 28nm (top-left), and chip summary table (right).

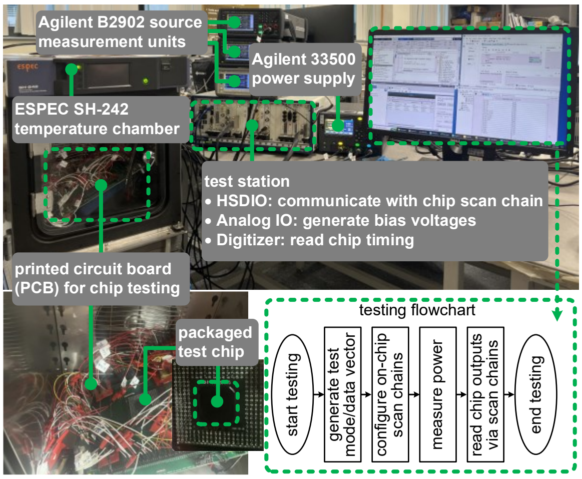

Agilent B2902 source measurement units
Agilent 33500 power supply
ESPEC SH-242 temperature chamber
test station
• HSDIO: communicate with chip scan chain
• Analog IO: generate bias voltages
• Digitizer: read chip timing
printed circuit board (PCB) for chip testing
packaged test chip
testing flowchart
start testing
generate test mode/data vector
configure on-chip scan chains
measure power
read chip outputs via scan chains
end testing

Fig. 8. Testing setup and testing flowchart for testchip functionality validation, and performance measurement.

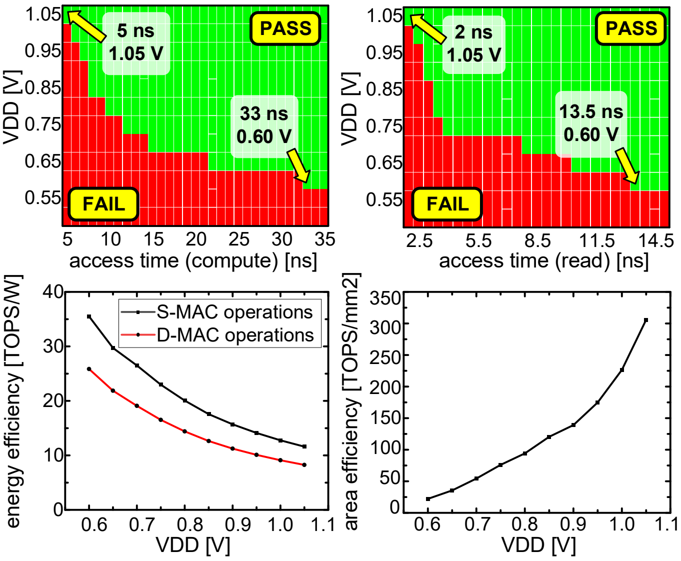


Fig. 9. Measured compute and read access time in SRAM and CIM mode (top), average energy and area efficiency at different V_{DD} (bottom).

The characterization of the periphery and the ADCs is shown in Fig. 10, where digital output value readouts are shown to be correctly disoverlapped and hence distinguished from each other at 6σ, down to lowest 0.6-V supply voltage (σ found from Monte Carlo simulations).

From layer-level analysis (Fig. 11), the S-MAC (D-MAC) average energy efficiency in ViT transformer is 35.5 TOPS/W (25.9 TOPS/W), and 138.2 TOPS/W in ResNet-50. Compared with prior SRAM CIM designs (Table I), the proposed architecture uniquely enables S- and D-MAC unification with no interruption cycle, no load-store in other banks, and no re-training as opposed to [2]-[4], [12], [14].

IV. Conclusion

A SRAM CIM macro uniquely featuring a dual-dataflow architecture is presented, addressing key limitations of prior

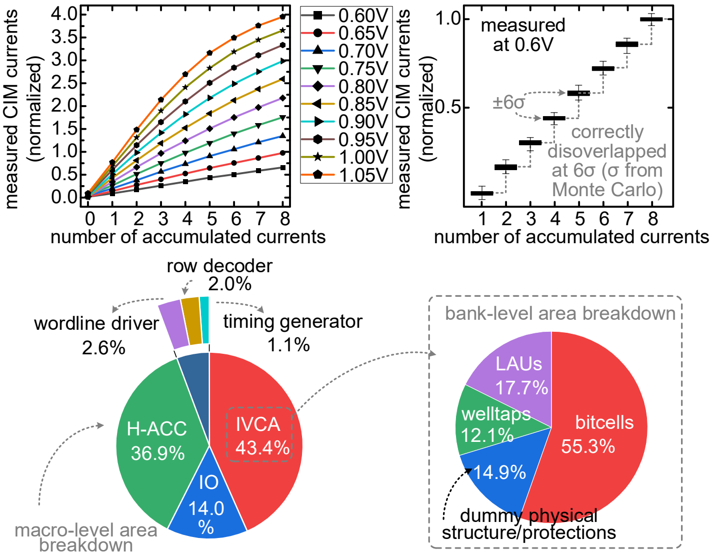


Fig. 10. Measured compute bitline accumulation currents at different supply voltages (top-left). The ADC transfer function shows correctly disoverlapped at $\pm 6\sigma$ bands even at the lowest V_{DD} (top-right). Macro- and array-level area breakdown in the proposed architecture (bottom).

TABLE I. COMPARISON WITH STATE-OF-THE-ART SRAM CIMS

		ISSCC'24 [3]	ISSCC'24 [4]	ISSCC'22 [2]	ISSCC'24 [7]	ISSCC'24 [11]	ISSCC'25 [12]	ISSCC'25 [13]	ISSCC'25 [14]
	this work								
technology	28 nm	22 nm	65 nm	28 nm	28 nm	16 nm	28 nm	28 nm	28 nm
supply voltage (V)	0.6 – 1.05	0.6 – 0.9	0.6 – 1.1	0.6 – 1.0	0.7 – 0.95	0.4 – 0.8	0.55 – 0.9	0.55 – 0.9	0.6 – 0.9
bitcell reuse (no modification)	**YES**	**YES**	NO	NO	YES	NO	**YES**	**YES**	**YES**
CNNs and transformers	**YES**	**YES**	**YES**	NO	NO	NO	**YES**	NO	**YES**
unified S-MACs and D-MACs [a]	**YES**	NO	NO	NO	NO	NO	NO	NO	NO
compute domain	hybrid	hybrid	analog	digital	hybrid	digital	digital	digital	hybrid
full output precision	**YES**	**YES**	NO	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
average energy efficiency (TOPS/W) [b]	S-MAC	S-MAC							
	11.59 – 35.51	15 [c] – 46 [c]	–	12.5 – 20.5	22.7 – 50.5	43.2 – **58.1**	25.4	17.5 – 50	–
	D-MAC	D-MAC							
	8.25 – **25.85**	–	–	–	–	–	–	–	–
peak energy efficiency (TOPS/W) [b]	S-MAC	S-MAC							
	47.2 – **138.2**		108		50.1 – 111.1	74.1 – 96.3	90.1	36.5 – 104	21 – 67.8
	D-MAC	D-MAC							
	9.7 – **30.5**	–	21 [d]	–	–	–	–	–	–
concurrent write/compute	**YES**	NO	NO	NO	NO	**YES**	NO	NO	NO

[a] Support of both S-MAC and D-MAC operations without extra data loading/interruptions [b] Chip performance normalized to 28nm (scaling factors from public foundry announcements), 8-bit integer MAC [c] Approximate [d] Transformer operations executed by inefficient repurposing of S-MAC (necessary load-store energy also omitted)

CIMs via unification of static/dynamic MAC operations in CNNs and multi-modal transformers. Reset-less in-bitcell Boolean, analog-digital hybrid accumulation, and concurrent write/compute demonstrate state-of-the-art peak energy efficiency in both S-MACs and D-MACs for highly efficient and uninterrupted resource-constrained edge AI systems.

ACKNOWLEDGMENT

Singapore Ministry of Education (T2EP50223-0040), TSMC for fabrication, and Filippo Martinini for discussions.

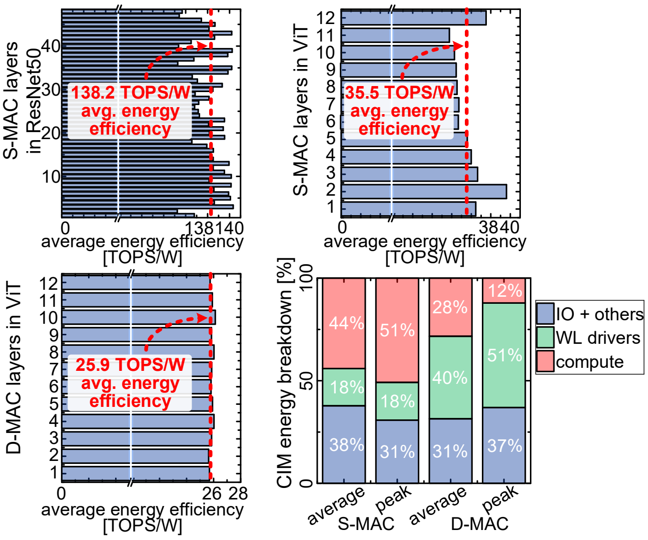


Fig. 11. Measured layer-level energy efficiency in ResNet-50 and ViT (top and bottom-right), average (peak) energy breakdown in S-MAC and D-MAC operations under 50% (90%) input/weight sparsity (bottom-right).

REFERENCES

[1] T. Wen *et al.*, “Fusion of memristor and digital compute-in-memory processing for energy-efficient edge computing,” *Science*, vol. 384, no. 6693, pp. 325–332, Apr. 2024.

[2] F. Tu *et al.*, "A 28nm 15.59µJ/Token Full-Digital Bitline-Transpose CIM-Based Sparse Transformer Accelerator with Pipeline/Parallel Reconfigurable Modes," in *IEEE Int. Solid-State Circuits Conf.*, San Francisco, CA, USA, 2022, pp. 466-468.

[3] A. Guo *et al.*, "34.3 A 22nm 64kb Lightning-Like Hybrid Computing-in-Memory Macro with a Compressed Adder Tree and Analog-Storage Quantizers for Transformer and CNNs," in *IEEE Int. Solid-State Circuits Conf.*, San Francisco, CA, USA, 2024, pp. 570-572.

[4] K. Yoshioka, "34.5 A 818-4094TOPS/W Capacitor-Reconfigured CIM Macro for Unified Acceleration of CNNs and Transformers," in *IEEE Int. Solid-State Circuits Conf.*, San Francisco, CA, USA, 2024, pp. 574-576.

[5] J. Lin *et al.*, "Av-Sepformer: Cross-Attention Sepformer for Audio-Visual Target Speaker Extraction," in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, Rhodes Island, Greece, 2023.

[6] Y. Yokoyama, K. Nii, Y. Ishii, S. Tanaka and K. Kobayashi, "Disturbance Aware Dynamic Power Reduction in Synchronous 2RW Dual-Port 8T SRAM by Self Adjusting Wordline Pulse Timing," *IEEE J. Solid-State Circuits*, vol. 58, no. 7, pp. 2098-2108, July 2023.

[7] Y. Yuan *et al.*, "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 *IEEE Int. Solid-State Circuits Conf.*, San Francisco, CA, USA, 2024, pp. 576-578.

[8] Y. Tan *et al.*, “Implementing Boolean Logic in Ferroelectric Field-Effect Transistors,” *Adv. Mater.*, vol. 9, no. 4, 2023.

[9] W. -X. You, C. -Y. Wang, Y. Wang, T. -Y. Jonathan Chang and S. S. Liao, "Write-enhanced Single-ended 11T SRAM Enabling Single Bitcell Reconfigurable Compute-in-Memory Employing Complementary FETs," in *IEEE Symp. VLSI Technology and Circuits*, Kyoto, Japan, 2023.

[10] J. Yue *et al.*, "An Energy-Efficient Computing-in-Memory NN Processor With Set-Associate Blockwise Sparsity and Ping-Pong Weight Update," *IEEE J. Solid-State Circuits*, vol. 59, no. 5, pp. 1612-1627, May 2024.

[11] W. -S. Khwa *et al.*, "34.2 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," in *IEEE Int. Solid-State Circuits Conf.*, San Francisco, CA, USA, 2024, pp. 568-570.

[12] X. Wang *et al.*, "14.3 A 28nm 17.83-to-62.84TFLOPS/W Broadcast-Alignment Floating-Point CIM Macro with Non-Two's-Complement MAC for CNNs and Transformers," in *IEEE Int. Solid-State Circuits Conf.*, San Francisco, CA, USA, 2025, pp. 254-256.

[13] Y. Yuan *et al.*, "14.5 A 28nm 192.3TFLOPS/W Accurate/Approximate Dual-Mode-Transpose Digital 6T-SRAM CIM Macro for Floating-Point Edge Training and Inference," in *IEEE Int. Solid-State Circuits Conf.*, San Francisco, CA, USA, 2025, pp. 258-260.

[14] X. Chen *et al.*, "14.6 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 *IEEE Int. Solid-State Circuits Conf.*, San Francisco, CA, USA, 2025, pp. 260-262.