11.7 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

Bonan Yan[1], Jeng-Long Hsu[2], Pang-Cheng Yu[2], Chia-Chi Lee[2], Yaojun Zhang[3], Wenshuo Yue[1], Guoqiang Mei[3], Yuchao Yang[1], Yue Yang[2], Hai Li[4], Yiran Chen[4], Ru Huang[1]

[1]Peking University, Beijing, China
[2]NeoNexus, Singapore, Singapore
[3]Pimchip Technology, Beijing, China
[4]Duke University, Durham, NC

Advanced intelligent embedded systems perform cognitive tasks with highly-efficient vector-processing units for deep neural network (DNN) inference and other vector-based signal processing using limited power. SRAM-based compute-in-memory (CIM) achieves high energy efficiency for vector-matrix multiplications, offers <1ns read/write speed, and saves vastly repeating memory accesses. However, prior SRAM CIM macros require a large area for compute circuits (either using ADC for analog CIM [1-4] or CMOS static logic for all-digital CIM [5-6]), have limited CIM functions, and use fixed vector-processing dimensions that cause a low-spatial-utilization rate when deploying DNN (Fig. 11.7.1).

Aiming to boost density and flexibility, this work presents a 32Kb ADC-less SRAM CIM macro in a 28nm technology. This work has three main features: (1) dynamic logic compute circuits (DCC) for compact area, that replace the conventional ADC [1-4] or CMOS static logic [5-6]; (2) reconfigurable local processing units (RLPUs) inside the bitcell array to support reconfigurable bitwise operations, including AND, XOR and OR; and (3) novel post-sum circuits for a >98% DNN utilization rate. The proposed macro also extends CIM multiply-accumulate (MAC) operations to both vector-matrix multiplication (VMM) and vector-Hadamard-product (VHP) computation. This work has the highest weight density (1.041Mb/mm²) normalized to a 28nm node, and the best energy efficiency, 19.21 - 35.55TOPS/W using signed 8b integer (INT8) inputs and weights, among prior SRAM-CIM macros. The classic efficiency metric of space, wattage, and performance (SWaP = TOPS/W × Mb/mm²) is used as the figure of merit; this work presents >6× better SWaP than other state-of-the-art SRAM CIM macros.

Figure 11.7.2 shows the overall architecture of the proposed ADC-less SRAM CIM macro: including memory read/write circuits, the memory subarray that is segmented to 32 compartments (scalable for larger designs), and the computation circuits (shift & add, post-sum, and CIM controller). Each compartment comprises of WL drivers & CIM input control (WLDCIC), input combinatory logic (ICL), foundry compact-6T bitcells with RLPUs and DCC. Each 4T RLPU is associated with 16 bitcells. In a compartment, every 4 RLPUs share 1 output wire connected to a DCC transistor and are multiplexed during computation. The routing channels for the output wires are above RLPUs and each compartment is facilitated with 16 output wires. The macro works in either memory mode or CIM mode. Memory mode enables the memory read/write circuits; weights are preloaded into the CIM macro in this mode. This work supports signed 2's-compliment 4 or 8b data. In CIM mode, compute inputs are provided in a bit-serial form, using an MSB-first-in principle. The input. x_i for the i^{th} compartment, multiplies the stored weights with the default RLPU configuration: bitwise AND. It takes 8 cycles to complete an 8b shift & add element-wise multiplication. The 32 shift & add 16b results are the elements forming the VHP resultant vector. The post-sum circuit can add them up, along columns, to produce the VMM results. This in-memory computation is lossless with bit-by-bit operations.

Figure 11.7.3 illustrates the DCC and RLPU mechanism. During each cycle, the dynamic logic switch Φ is first set to ground to pre-charge the DLO output to V_{DD}. Then, Φ is asserted for evaluation. A following D-FF samples and holds the computational results. The DCC and D-FFs occupy 17.5× less area and perform the same function as the latest area-optimized ADC. Compared to CMOS static logic, DCC saves transistors in the pull-up network inside the subarray, and its output drops abruptly with RLPU as a pull-down network. RLPU comprises of 4 transistors with their gates connecting to the SRAM complementary bitlines, BLP and BLN, and the intermediate input wires INP and INN. The datum (*w*) stored in the selected bitcell is equal to the logic value of BLP and the inverted version of BLN. The bitwise operation between IN and *w* can be reconfigured to AND (INP=IN, INN=0), OR (INP=0, INN=inverted IN) and XOR (INP=IN, INN=inverted IN) via ICL. For the ease of routing, the entire array shares the same ICL configuration. By incorporating different bitwise operations and the post-sum circuit, the macro can realize various types of computation including VHP/VMM for AI applications, hamming distance computation in image signal processing and in-memory data masking in the embedded systems.

Figure 11.7.4 shows the post-sum circuits revised from the adder tree in [6]. The proposed macro supports VHP, but the direction of MAC dataflow in the array is different from [1-3]; thus, the sum of the output elements can be arbitrarily combined for flexibility. The post-sum circuit is designed to give the sum of 4, 9 and 32 VHP resultant vectors. The sum of *k* elements is referred as Σ_k. Σ_4 and Σ_{32} can be directly obtained from the adder tree. Σ_9 is implemented by the addition of Σ_8 in the adder tree and 1 element of the VHP result. Σ_9 mode gives threes Σ_9 results per operation and outputs an additional Σ_4 result to increase the utilization rate. Σ_9 mode is convenient for mapping the most common 3×3 convolutional kernels. CIM operates with fixed input-to-output vector dimensions. For each 1×32 weight matrix column, a 3×3 convolution kernel is unrolled and deployed either to 3 memory columns with a $^9/_{32}$ spatial utilization rate or to one column by taking 3 times the length of a single operation to complete a single 3×3 convolution. The introduction of a Σ_9 mode allows the proposed CIM macro to realize 3×3 convolution with a $^{31}/_{32}$ utilization rate, a 3.44× improvement, within 1 operation period: 8 clock cycles for 8b inputs. The overall utilization rate, including both fully connected and convolution layers in ResNet, is >98.4% with the available Σ_9 mode. Figure 11.7.4 shows an example of an image-comparing task in near-sensor embedded systems. By caching a picture into the SRAM CIM macro and configuring RLPU to XOR mode, the Σ_{32} post-sum directly gives the hamming distances between the cached picture and the input. This method combines the computation with the image obtaining process and reduces memory access by ~50%.

Figure 11.7.5 shows the block diagram for testing and the measured silicon results, where an off-chip LVDS clock, up to 2GHz, is used. The measurement results show an average 27.38TOPS/W with 8b input and weight precisions, using a 3ns clock period, and a 0.8V supply. The chip can function with a lower supply (<0.8V) using lower clock frequencies. The average energy efficiency is measured using quantized ResNet-34 and MobileNet for CIFAR-100 and CIFAR-10 datasets. The power breakdown shows that the single-transistor-per-output-wire structure of DCC only occupies 0.5% of the macro power consumption. To minimize its area, DCC does not employ high keepers. Hence, the voltage of DLO can be dragged down by leakage current, aka. the long-compute-time hazard; however, measurement results show a safe working region for compute time below 600ns, which successfully avoids the long-compute-time hazard of DCC designs.

Figure 11.7.6 compares this work with state-of-the-art SRAM CIM macros. With the compact DCC replacing the ADC and CMOS static logic compute circuits, this work features the highest weight storage density of 1067Kb/mm², normalized to 28nm, and the highest energy efficiency, 27.38TOPS/W on average using 8b input and weight precision. The SWaP figure-of-merit emphasizes the importance of density and energy efficiency, show that this work achieves the highest SWaP of 1826, which is 6× more than prior SRAM-CIM macros. Figure 11.7.7 shows the die photo and summarizes the chip.

Acknowledgement:
This work is supported by National Key R&D Program of China (2017YFA0207600), National Natural Science Foundation of China (61925401, 92064004, 61927901), PKU-Baidu Fund Project 2019BD002 & 2020BD010, Peking University Startup Package 7100603362 for Prof. B. Yan, and the 111 Project (B18001). Prof. Y. Yang acknowledges the support from the Fok Ying-Tong Education Foundation, Beijing Academy of Artificial Intelligence (BAAI), and the Tencent Foundation through the XPLORER PRIZE. The authors thank for the support from Pimchip Technology Co., Ltd.

References:
[1] S. K. Gonugondla et al., "A 42pJ/Decision 3.12TOPS/W Robust In-Memory Machine Learning Classifier with On-Chip Training," *ISSCC*, pp. 490-491, 2018.
[2] X. Si et al., "A 28nm 64Kb 6T SRAM Computing-in-Memory Macro with 8b MAC Operation for AI Edge Chips," *ISSCC*, pp. 246-247, 2020.
[3] M. E. Sinangil et al., "A 7-nm Compute-in-Memory SRAM Macro Supporting Multi-Bit Input, Weight and Output and Achieving 351 TOPS/W and 372.4 GOPS," *IEEE JSSC*, vol. 56, no. 1, pp. 188- 198, Jan., 2021.
[4] J.-W. Su et al., "A 28nm 384kb 6T-SRAM Computation-in-Memory Macro with 8b Precision for AI Edge Chips," *ISSCC*, pp. 250-252, 2021.
[5] H. Kim et al., "A 1-16b Precision Reconfigurable Digital In-Memory Computing Macro Featuring Column-MAC Architecture and Bit-Serial Computation," *ESSCIRC*, pp. 345-348, 2019.
[6] Y.-D. Chih et al., "An 89TOPS/W and 16.3TOPS/mm² All-Digital SRAM-Based Full-Precision Compute-In Memory Macro in 22nm for Machine-Learning Edge Applications," *ISSCC*, pp. 252-254, 2021.

2022 IEEE International Solid- State Circuits Conference (ISSCC) | 978-1-6654-2800-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ISSCC42614.2022.9731545

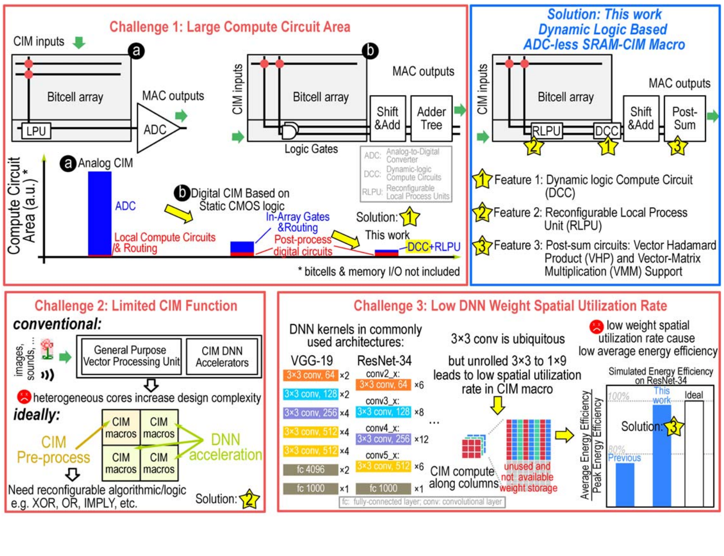


Figure 11.7.1: Motivation and the key features of this work.

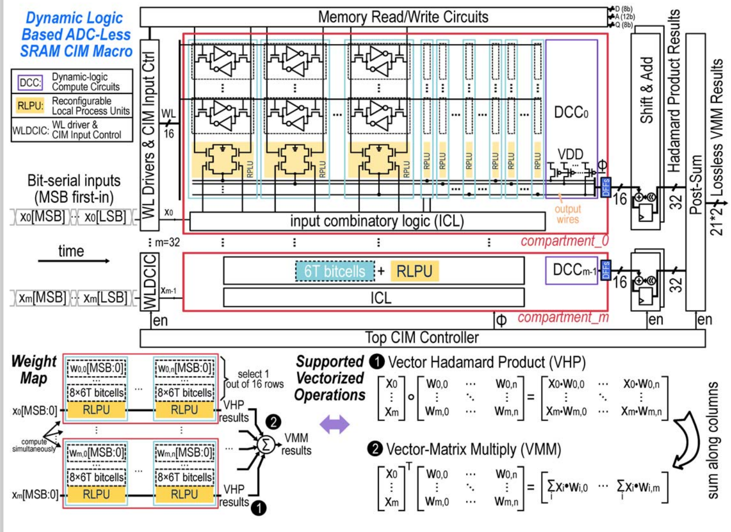


Figure 11.7.2: Macro architecture and the weight mapping scheme. The proposed ADC-less SRAM CIM macro is based on dynamic-logic compute circuits, supporting Hadamard-product and vector-matrix multiplication.

11

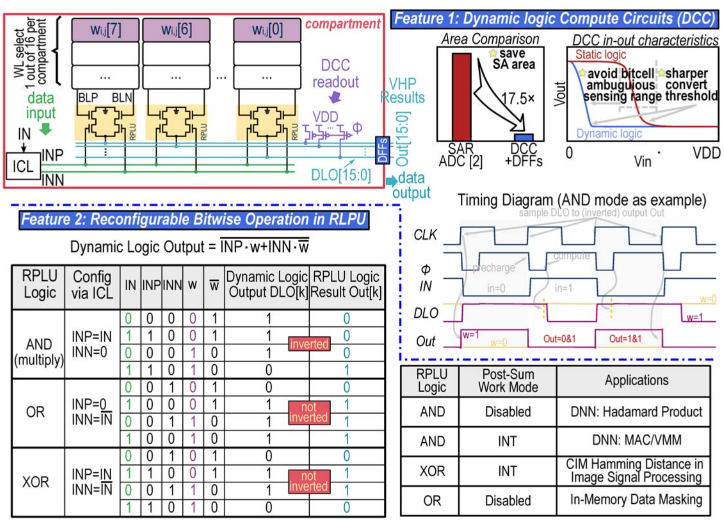


Figure 11.7.3: The mechanism of DCC and RPLU. DCC significantly reduces CIM macro area and RPLU extends CIM MAC computation to reconfigurable bitwise AND/OR/XOR operations.

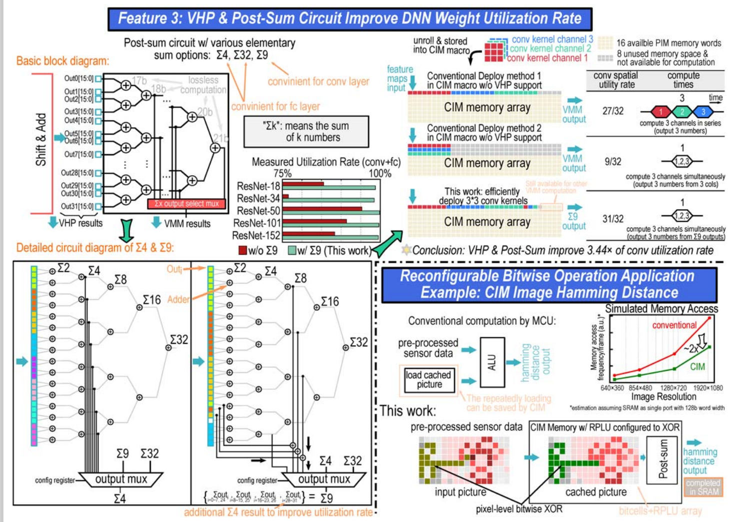


Figure 11.7.4: Post-sum circuit can arbitrarily combine and add 4, 9 and 32 VHP resultant elements to improve the spatial utilization rate for the most widely used 3×3 convolutional kernels.

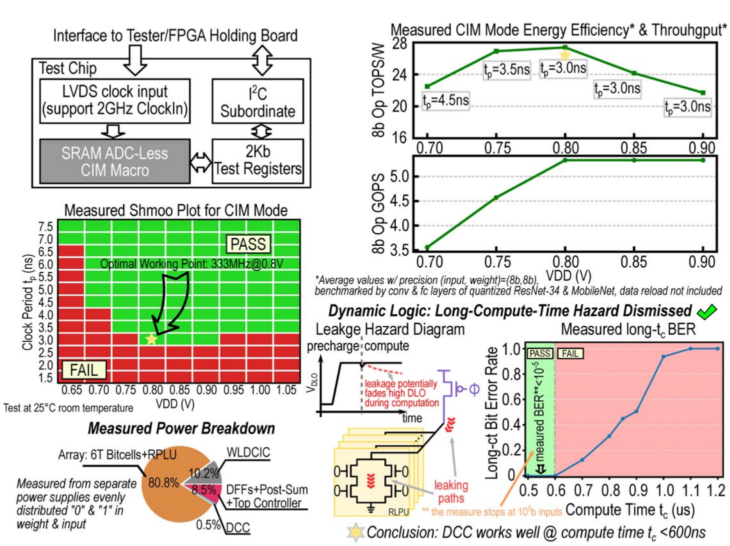
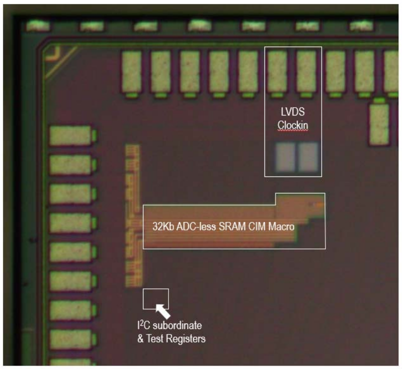


Figure 11.7.5: Test chip block diagram and measured silicon results.

	CIM Macro Type	Analog CIM				Digital CIM		
		ISSCC'18 [1]	ISSCC'20 [2]	JSSC'21 [3]	ISSCC'21 [4]	ESSCIRC'19 [5]	ISSCC'21 [6]	**This work**
Form Factor	Technology	65nm	28nm	7nm	28nm	65nm	22nm	**28nm**
	Array Size	4Kb	64Kb	4Kb	384Kb	16Kb	64Kb	**32Kb**
	Cell Type	S6T	6T	8T	6T	6T	6T	**6T**
	Macro Area	N/A	0.362mm^2	0.0032mm^2	1.4mm^{2†}	0.2272mm^2	0.202mm^2	**0.030mm^2**
	CIM Weight Density	N/A	177Kb/mm^2 @28nm	1250Kb/mm^2 @7nm	234Kb/mm^2 @28nm	71Kb/mm^2 @65nm	317Kb/mm^2 @22nm	**1067Kb/mm^2 @28nm**
	CIM Weight Density (normalized to 28nm)	N/A	177Kb/mm^2	78Kb/mm^2	234Kb/mm^2	383Kb/mm^2	196Kb/mm^2	**1067Kb/mm^2**
	Power Supply	1V, 0.8V	0.7V-0.9V	1V, 0.8V	0.7-0.9V	0.6V-0.8V	0.72V	**0.8V**
Function Versatility	Compute Circuits	CMI-VSA	LMAR-SAR-ADC	Flash ADC	Ph-ADC	CMOS Static Logic	CMOS Static Logic	**Dynamic Logic Compute Circuit**
	Support Bitwise Operation	AND	AND	AND	AND	AND	AND	**AND, XOR, OR**
	Fundamental Vector Operation	VMM	VMM	VMM	VMM	VMM	VMM	**Hadamard Product, VMM**
Computation & Efficiency	Input Bits	1	4b/8b	4b	4b/8b	1b-16b	1b-8b	**1b-8b**
	Weight Bits	1	4b/8b	4b	4b/8b	4b/8b/12b/16b	4b/8b/12b/16b	**1b/4b/8b**
	Output Bits	1	12b* (4b/4b) 20b* (8b/8b)	4b	12b* (4b/4b) 20b* (8b/8b)	8b-23b*	16b* (4b/4b) 24b* (8b/8b)	**VMM*: 21b VHP*: 8b (8b/8b)**
	Cycle Time	2.3ns	4.1ns (4b/4b) 8.4ns (8b/8b)	4.5ns (1.0V) 5.5ns (0.8V)	4ns (4b/4b) 7.2ns (8b/8b)	N/A	10ns (4b/4b) 18ns (8b/8b)	**3ns/input bit**
	Energy Efficiency** (TOPS/W)	55.8 (1b/1b)	58.1(4b/4b) 14.1(8b/8b)	351 (1b/1b)	77.3(4b/4b) 18.9(8b/8b)	117.3 (1b/1b)	24.7 (8b/8b)	**27.38 (8b/8b)**
	SWaP (FoM)*** (TOPS/W Mb/mm^2)	N/A	161	26.7	283	8.13	303	**1826**

† Estimated from [4] * Full-precision CIM computation in terms of bitwise length ** Average energy efficiency, no sparsity technique employed
***The figure of merit (FoM) SWaP (space, wattage and performance)=(Density*Performance)/Power=Density*Energy Efficiency
Here we use density normalized to 28nm, energy efficiency scaled with input and weight bitwise length

Figure 11.7.6: Comparison to previous work.



Chip Summary	
Technology	28nm
Single Array Size	32Kb
Area of Bitcell (16cells+1RLPU)	5.795um*0.71um (1.34x of compact 6T cell)
Concurrent VMM input channel	32
Number of VMM elements accumulated simultaneously	Σ4/Σ9/Σ32
Supply voltage	0.8V
Operand Type	Signed INT8
Clock Frequency	333MHz
Computation time	3ns/input bit
Input Precision	1-8b
Weight Precision	1b/4b/8b
Energy Efficiency	19.21~35.55 (8b/8b)
Reconfigurable Bitwise Operation	AND, XOR, OR
Fundamental Vector Operation	Hadamard product, Vector-matrix multiplication
Compute Circuit	Dynamic logic Compute Circuit (DCC)

Figure 11.7.7: Die micrograph and key metric comparison table.