16.4 An 89TOPS/W and 16.3TOPS/mm² All-Digital SRAM-Based Full-Precision Compute-In Memory Macro in 22nm for Machine-Learning Edge Applications

Yu-Der Chih, Po-Hao Lee, Hidehiro Fujiwara, Yi-Chun Shih, Chia-Fu Lee, Rawan Naous, Yu-Lin Chen, Chieh-Pu Lo, Cheng-Han Lu, Haruki Mori, Wei-Chang Zhao, Dar Sun, Mahmut E. Sinangil, Yen-Huei Chen, Tan-Li Chou, Kerem Akarvardar, Hung-Jen Liao, Yih Wang, Meng-Fan Chang, Tsung-Yung Jonathan Chang

TSMC, Hsinchu, Taiwan

2021 IEEE International Solid- State Circuits Conference (ISSCC) | 978-1-7281-9549-0/20/$31.00 ©2021 IEEE | DOI: 10.1109/ISSCC42613.2021.9365766

From the cloud to edge devices, artificial intelligence (AI) and machine learning (ML) are widely used in many cognitive tasks, such as image classification and speech recognition. In recent years, research on hardware accelerators for AI edge devices has received more attention, mainly due to the advantages of AI at the edge: including privacy, low latency, and more reliable and effective use of network bandwidth. However, traditional computing architectures (such as CPUs, GPUs, FPGAs, and even existing AI accelerator ASICs) cannot meet the future needs of energy-constrained AI edge applications. This is because ML computing is data-centric, most of the energy in these architectures is consumed by memory accesses. In order to improve energy efficiency, both academia and industry are exploring a new computing architecture, namely compute in memory (CIM). CIM research is focused on a more analog approach with high-energy efficiency; however, lack of accuracy, due to a low SNR, is the main disadvantage; therefore, an analog approach may not be suitable for some applications that require high accuracy.

This work proposes a 6T SRAM-based all-digital CIM macro for multiply-accumulate (MAC) operations in a convolutional neural network. It can support input activations with programmable bit-widths (1-8 per macro), signed or unsigned, and weights with 4 different bit widths (4, 8, 12, or 16). By using multiple macros in different configurations, it can support neural networks with different topologies. A new architecture, based on bit-serial multiplication and parallel adder trees, can achieve massively parallel MAC operations, with high energy/area efficiency and throughput. Moreover, by effectively reusing the weights and input activations for the MAC operations of the convolutional layer, the memory access energy can be greatly reduced.

Figure 16.4.1 shows the block diagram and the bitcell layout of the proposed SRAM-based CIM macro. This CIM macro has two operating modes: SRAM and CIM mode. SRAM mode is used to preload the weights into the SRAM of the CIM macro. CIM mode is used to perform MAC operations. The CIM macro contains 256 input activations, 64 partial-sum outputs and 256×64 4b weights. It contains 64 columns of sub-CIM units, each unit contains 256 4b weights, 256×4 bit-wise multipliers, a parallel adder tree, a partial-sum accumulator, and an SRAM read/write circuit. As shown in Fig. 16.4.1 (upper-right), the 6T SRAM cells, (0.379μm²) which do not use push rules, and the 4T NOR gates are used for 1b weight storage and bitwise multiplication.

Figure 16.4.2 shows the detailed schematic view of the CIM macro and the timing waveform for the CIM operation. In CIM operation, 256 input activations are simultaneously fed into the CIM array in an MSB-first bit-serial manner. For the input activations with 4b precision, in the first four cycles, one sub-CIM unit performs 256 multiplications of 1b input activation and 4b weights in each cycle, the results of multiplications are fed into the adder tree to generate the partial sum of 256 multiplications. A partial-sum accumulator circuit is required to accumulate the partial sums of each cycle in a pipelined manner. Therefore, an extra cycle is required to complete the accumulation, and a total of 5 cycles are required to complete the multiply-accumulate for input activations with 4b precision. The operation can be expressed as $Q_j = 2^3\times\Sigma(A_{3i}\times W_{ji}) + 2^2\times\Sigma(A_{2i}\times W_{ji}) + 2^1\times\Sigma(A_{1i}\times W_{ji}) + 2^0\times\Sigma(A_{0i}xW_{ji})$, where i=0 to 255, for unsigned inputs and weights. Figure 16.4.2 (upper right) shows the 5b adder circuit, which is the first stage of adder tree. The circuit consists of a half adder and 4 full adders, where one additional full adder is used for sign extension for signed weights. In the accumulator, additional compliment and increment functions are applied to the partial sum of the input activation's sign bit (MSB) for signed inputs. For CIM macros with a 4b weight and 256 input activations using 4 or 8b, the bit width of each accumulator's outputs are 16 or 20b for no accuracy loss. The bit-width precision of the partial sum output is sufficient for the subsequent vector computations: such as batch normalization, pooling and activation. This is very different from an analog approach, whose accuracy is limited by the precision of the column-based ADC (≤ 8b). In addition to effective data/weight reuse, the data format of inputs and outputs are the same NCHW (batch, channel, height, and width), which can also reduce energy/latency on the data format conversion between two adjacent layers.

For AI edge application, the most crucial metric is energy efficiency, expressed in TOPS/W. Dynamic voltage scaling (DVS) is used to improve the energy efficiency and throughput. As shown in Fig. 16.4.3, a higher supply is used for weight updates, and a lower supply for MAC operations. Simulation results at 25°C show that the latency for updating 256×64 weights is about 0.4μs from a 0.8V supply, while MAC operations perform at 97TOPS/W (4b weight/4b input) from a 0.68V supply. Moreover, simulation results show that TOPS/W can be increased by 30% by interleaving 28T and 14T full adders in the adder tree, rather than using just 28T adders. The energy consumption also depends on the sparsity of the weights and the input activation toggle rate. Simulation shows it mainly come from the energy consumption in the first few stages of adder tree.

Programmability is also an essential feature of AI hardware, as neural networks may have different topologies and bit-width precision. In the proposed CIM, multiple macros, either in parallel, serial, or 2D arrays, can support different neural networks. For example, 3 cascaded CIM arrays can support a convolution operation of a 3×3 filter with 64 channels. Moreover, weight updates can be performed concurrently with each MAC operation, as shown in Fig. 16.4.4. In addition, the weights of each column can be signed or unsigned. In this way, a CIM macro can support 4b, 8b, 12b or 16b weights. When a column is assigned unsigned weights, its additional full adder for sign extension is disabled.

Figure 16.4.5 shows the test chip block diagram of a 22nm CIM macro and the measured shmoo plot for the CIM function. Two SRAM macros are added for storing the input activations and receiving the output partial sums for at-speed tests. Measurements show a 89TOPS/W performance from 0.72V for a sparse pattern (18% input toggle rate, 50% 1s for weights) and 52TOPS/W for a dense pattern (50% input toggle rate, 50% 1s for weights). A 256×64 CIM array archives 3.3TOPS from 0.72V at 25°C. Compared with the listed references (Fig. 16.4.6), the proposed digital CIM can achieve a high FOM (TOPS/W × TOPS/mm²) of 1450 without MAC operation accuracy reduction. Moreover, a 5nm design assessment of this digital CIM approach shows that it has excellent scalability, with TOPS/W and TOPS/mm² improving by 2.8× and 19×, compared to the 22nm design.

Figure 16.4.7 shows the summary of key features and the die photo. A digital 256×64 CIM macro achieving 89TOPS/W and 16.3TOPS/mm² has been presented in a 22nm logic process. The modular approach, with a programmable input activation and weight bit width, either unsigned or signed, can support versatile neural networks, without accuracy loss in the all-digital MAC operation. Moreover, MAC operations with DVS and concurrent weight updates can further improve the energy efficiency and throughput. On the chip level, the energy and latency for memory access and data flow can be greatly reduced by using the proposed CIM macro with effective weight/data reuse and an NCHW data format for both input and output activations.

References:

[1] W.-S. Khwa et al., "A 65nm 4Kb Algorithm-Dependent Computing-in-Memory SRAM Unit-Macro with 2.3ns and 55.8 TOPS/W Fully Parallel Product-Sum Operation for Binary DNN Edge Processors," *ISSCC*, pp. 496-497, 2018.
[2] A. Biswas et al., "Conv-RAM" An Energy-Efficient SRAM with Embedded Convolution Computation for Low-Power CNN-Based Machine Learning Applications," *ISSCC*, pp. 488-489, 2018.
[3] X. Si et al., "A Twin-8T SRAM Computation-In-Memory Macro for Multiple-Bit CNN-Based Machine learning," *ISSCC*, pp. 396-397, 2019.
[4] 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.
[5] Q. Dong et al., "A 351TOPS/W and 372.4GOPS Compute-In-Memory SRAM Macro in 7nm FinFET CMOS for Machine-Learning Applications," *ISSCC*, pp. 242-243, 2020.
[6] 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.

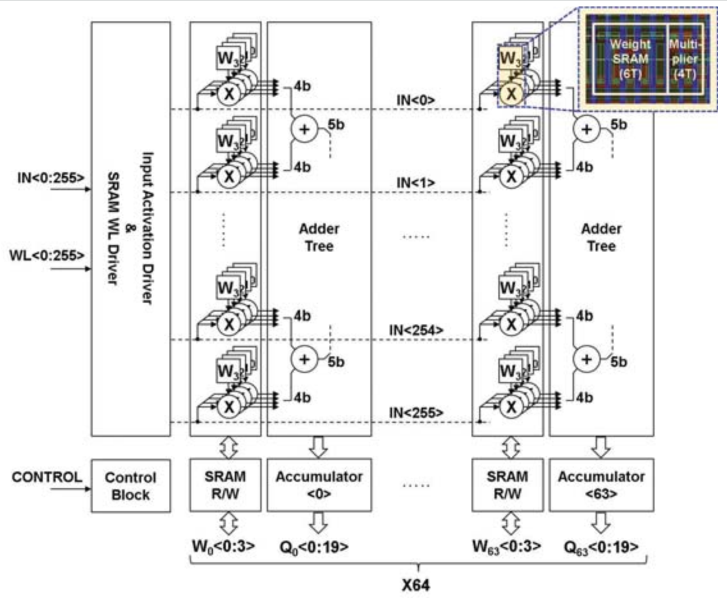


Figure 16.4.1: SRAM-based CIM macro block diagram and bit cell layout.

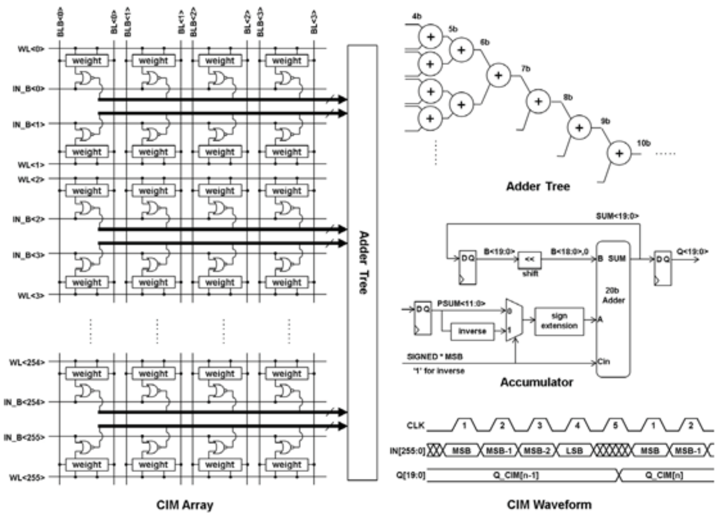


Figure 16.4.2: Schematic view of one sub-CIM unit and CIM operational timing waveforms.

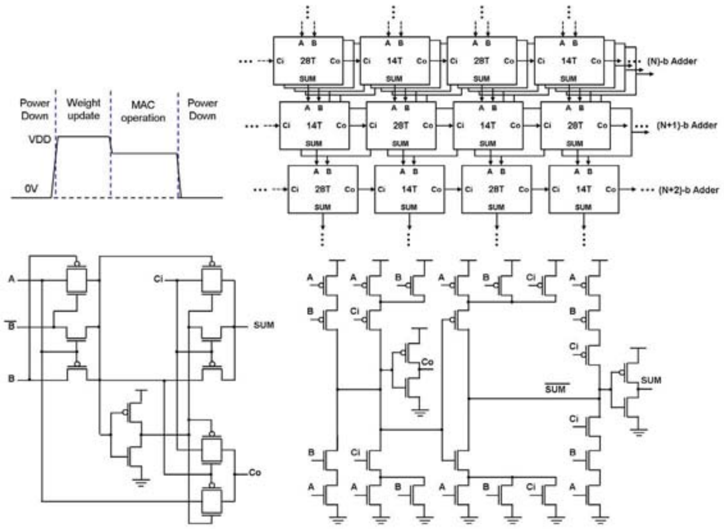
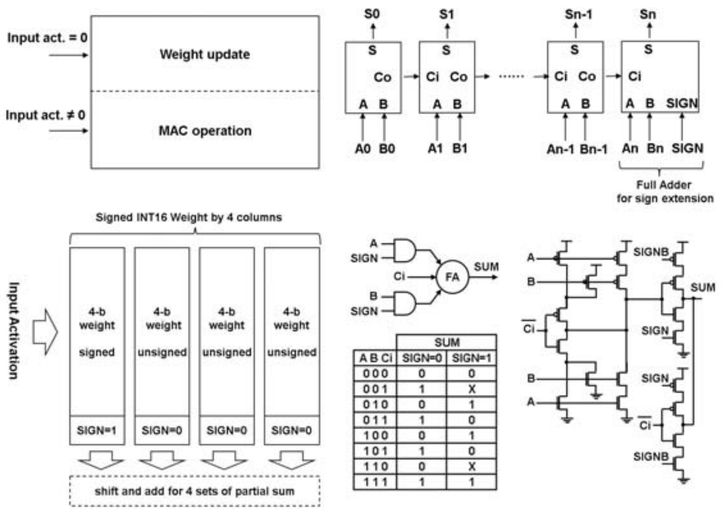


Figure 16.4.3: (Top) DVS for energy saving in MAC operation, and the proposed adder tree structure with two kinds of full adders for dynamic power saving. (Bottom) 14-T and 28-T full-adder.



A B Ci	SUM SIGN=0	SUM SIGN=1
0 0 0	0	0
0 0 1	1	X
0 1 0	0	1
0 1 1	1	0
1 0 0	0	1
1 0 1	1	0
1 1 0	0	X
1 1 1	1	1

Figure 16.4.4: (Left) Concurrent weight update function and MAC operation with programmable signed/unsigned weights. (Right) Schematic and truth table for the extra full-adder used for sign extension.

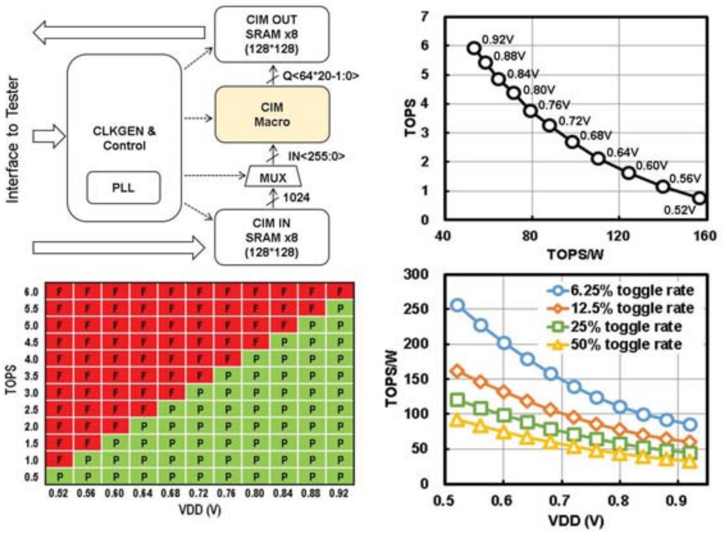
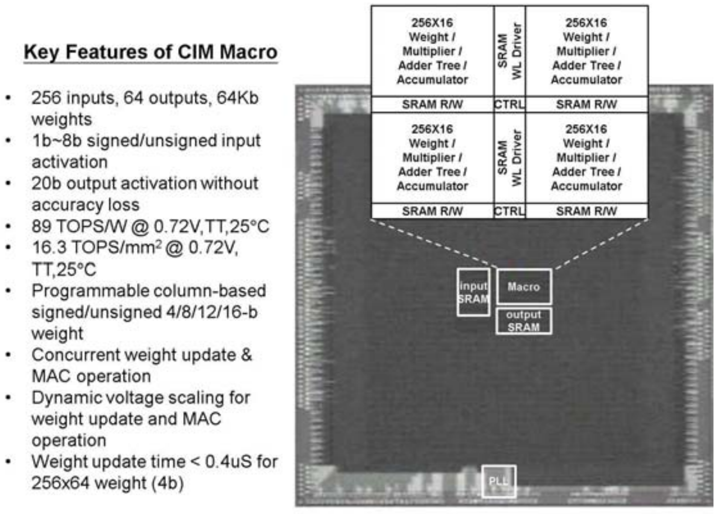


Figure 16.4.5: Test chip block diagram and the measured silicon results.

	ISSCC'18 [1]	ISSCC'18 [2]	ISSCC'19 [3]	ESSCIRC'19 [4]	ISSCC'20 [5]	ISSCC'20 [6]	This work
Technology	65nm	65nm	55nm	65nm	7nm	28nm	22nm
MAC operation	Analog	Analog	Analog	Digital	Analog	Analog	Digital
Array Size	4Kb	16Kb	3.8Kb	16Kb	4Kb	64Kb	64Kb
Cell Type	S6T	10T	T8T	6T	8T	6T	6T
Push rule	Yes	No	Yes	NA	Yes	NA	No
Macro size (mm^2)	NA	0.067	NA	0.2272	0.0032	NA	0.202
Bitcell Area (um^2)	0.525	NA	0.865	NA	0.053	0.25	0.379
Power Supply(V)	1&0.8	1.2&0.9	1	0.6~0.8	0.8	0.7~0.9	0.72
Inputs Bits	1	7	4	1~16	4	4~8	1~8
Weight bits	1	1	5	4/8/12/16	4	4/8	4/8/12/16
Output Bits	1	7	7	8~23	4	12 (4b/4b) 20 (8b/8b)	16 (4b/4b) 24 (8b/8b)
Cycle time (ns)	2.3	150	10.2	NA	5.5	4.1 (4b/4b) 8.4 (8b/8b)	10 (4b/4b) 18* (8b/8b)
Throughput (GOPS)	1780	10.67	17.6	567 (1b/1b)	372.4 (4b/4b)	124.88 (4b/4b) 30.48 (8b/8b)	3300 (4b/4b) 917* (8b/8b)
Energy Efficiency (TOPS/W)	55.6	28.1	18.4	117.3 (1b/1b)	262.3~610.5 (4b/4b)	68.44 (4b/4b) 16.63 (8b/8b)	89 (4b/4b) 24.7* (8b/8b)

*estimation

Figure 16.4.6: Key metric comparison table.



Key Features of CIM Macro

- 256 inputs, 64 outputs, 64Kb weights
- 1b~8b signed/unsigned input activation
- 20b output activation without accuracy loss
- 89 TOPS/W @ 0.72V,TT,25°C
- 16.3 TOPS/mm^2 @ 0.72V, TT,25°C
- Programmable column-based signed/unsigned 4/8/12/16-b weight
- Concurrent weight update & MAC operation
- Dynamic voltage scaling for weight update and MAC operation
- Weight update time < 0.4uS for 256x64 weight (4b)

256X16 Weight / Multiplier / Adder Tree / Accumulator
SRAM WL Driver
256X16 Weight / Multiplier / Adder Tree / Accumulator
SRAM R/W
CTRL
SRAM R/W
256X16 Weight / Multiplier / Adder Tree / Accumulator
SRAM WL Driver
256X16 Weight / Multiplier / Adder Tree / Accumulator
SRAM R/W
CTRL
SRAM R/W
input SRAM
Macro
output SRAM
PLL

Figure 16.4.7: Summary of key features and die photograph.