16.3 A 28nm 16.9-300TOPS/W Computing-in-Memory Processor Supporting Floating-Point NN Inference/Training with Intensive-CIM Sparse-Digital Architecture

Jinshan Yue[1], Chaojie He[1], Zi Wang[1], Zhaori Cong[1], Yifan He[2], Mufeng Zhou[2], Wenyu Sun[2], Xueqing Li[2], Chunmeng Dou[1], Feng Zhang[1], Huazhong Yang[2], Yongpan Liu[2], Ming Liu[1]

[1]Institute of Microelectronics of the Chinese Academy of Sciences, Beijing, China
[2]Tsinghua University, Beijing, China

Computing-in-memory (CIM) has shown high energy efficiency on low-precision integer multiply-accumulate (MAC) [1-3]. However, implementing floating-point (FP) operations using CIM has not been thoroughly explored. Previous FP CIM chips [4-5] require either complex in-memory FP logic or have lengthy alignment-cycle latencies arising from converting FP data having different exponents into integer data. The challenges for an energy-efficient and accurate FP CIM processor are shown in Fig. 16.3.1. Firstly, aligning an FP vector onto a CIM module requires a long bit-serial sequence due to infrequent but long tail values, incurring many CIM cycles. In this work, we observe that most exponents of FP data are clustered in a small range, which motivates dividing FP operations into high-efficiency intensive-CIM and flexible sparse-digital parts. Secondly, to implement the intensive-CIM + sparse-digital FP workflow, a sparse digital core is required for flexible intensive/sparse processing. Thirdly, the FP alignment brings more random sparsity. Though analog CIM can utilize random sparsity with a low-resolution ADC, the corresponding sparse strategy for digital CIM has not been explored.

To overcome the above challenges, this work proposes an accurate and energy-efficient FP CIM processor for neural network (NN) inference/training applications. The main innovations include: 1) An FP-to-INT CIM workflow for the intensive FP operations with reduced execution cycles and high efficiency. 2) A flexible sparse-digital core for the remaining FP operations with encoded sparse activation/weight, which assists the CIM core to achieve both high energy efficiency and high accuracy. 3) An energy-efficient low-MAC-value (MACV) CIM macro that adopts a two-stage adder tree to utilize random sparsity and eliminate high-bit-position circuits with no accuracy loss.

Figure 16.3.2 shows the overall FP CIM processor, illustrating the intensive-CIM sparse-digital workload division. The CIM processor consists of a RISC-V CPU, 512KB global SRAM, a CIM core with four low-MACV CIM macros and a sparse digital core. A convolution or matrix-vector multiplication between FP weight (W) and activation (A) data is divided into intensive and sparse parts, and then re-organized into one intensive and two sparse parts. The intensive part $W_{intensive}•A_{intensive}$ is considered heavy computation and implemented on the CIM core for high efficiency. $W_{all}•A_{sparse}$ and $W_{sparse}•A_{intensive}$ are comparatively sparse computations, so they are implemented on the flexible sparse-digital core, which also performs the final sum. Note that $W_{intensive}$ can be extracted on-chip from W_{all}, while W_{sparse} and A_{sparse} consume extra storage. The CIM core supports both FP and INT operations. Together with the RISC-V CPU and FP digital core, the back-propagation, error calculation and weight update operations can be executed on-chip to support FP/INT inference and training applications.

Figure 16.3.3 presents the overall FP-to-INT CIM architecture for intensive FP operations. It fetches and selects the intensive part ($W_{intensive}$) from the total weight (W_{all}), and then stores it into the CIM macro in INT8/INT16 format. The exponents (Exp) of the intensive FP activations reside in a small range [E_{min}, E_{max}] to reduce the CIM execution cycles. A concise bit-serial FP-to-INT transfer unit converts the FP data as multi-cycle bit vectors into the CIM macro. When Exp is not aligned (1st step), it aligns Exp towards E_{max} with a self-increasing operator, and outputs the sign bit. Once Exp is aligned (2nd step), the MSB of the mantissa is sent to the CIM macro bit-serially with a left-shift operator. An example of the FP-to-INT conversion/transfer is presented, requiring ($E_{max} - E_{min} + 12$) cycles.

Following the FP-to-INT conversion, the CIM macro runs FP MAC operations in integer format. The accumulation circuits support FP/INT activation and INT16/8/4 weight configurations, and convert the INT results back to FP format. The long bit-width INT results for various configurations are translated to an FP21 format with 5 additional mantissa bits (LSB) to avoid intermediate precision loss in the accumulation step, while the accumulation result is stored in a self-defined FP16 format. The self-defined FP16 format omits the Exp shift operation in the standard FP16, since it can be moved to the write-back step and be calculated together with the activation/weight-incurred Exp shift. By omitting the Exp shift in accumulation circuits, the critical path is shortened and power consumption is reduced. The result can also be written back as INT8/INT4 for a low-bit-precision NN. By only computing the intensive FP operations in the CIM module, this work significantly reduces the CIM execution cycles by 2.7× compared with the long-tail full FP16 execution on the CIM module. Compared with direct CIM FP alignment and digital FP circuits, this work shows 2.7× and 1.5× energy efficiency improvement.

Figure 16.3.4 shows the flexible sparse digital core for the remaining sparse FP operations, which implements $W_{all}•A_{sparse}$, $W_{sparse}•A_{intensive}$ and the final sum. Each 32b quantity in the digital core can represent either two 16b dense FP data items, or one 16b sparse FP data item with a 16b index. To simplify the circuit design with the limited index bitwidth, the row/column of the activation/weight are represented as absolute indices, while the channels of activation/weight are represented as an incremental index, which ensures large representation range with simple decoding circuits. Two examples are presented for the convolution of sparse weight/activation. In each cycle, the sparse digital core receives one sparse activation (weight) and eight dense weights (activations) for the MAC operation. Each MAC unit can be configured as BF16/FP16 by tuning the exponent/mantissa mask for bit[9:7]. Accumulating the results of the intensive-CIM and sparse-digital cores, the final accuracy matches with the FP16 baseline. The sparse digital FP execution only takes 4.59% of the total system energy. Layer-wise evaluation on ImageNet, ResNet50 shows that the sparse digital core incurs no performance loss by parallel CIM/digital execution.

Figure 16.3.5 shows the low-MACV CIM macro, consisting of a wordline decoder, activation input, normal IO, 16 128×2b SRAM banks and 16 low-MACV adder trees. The 128×2b SRAM bank adopts a ping-pong structure, with one 6T cell in each ping/pong part. The weight data can be directly selected from Q/QB with a fixed path instead of from the bitline (BL/BLB) with precharge/discharge in each cycle. Observe that with the ping-pong structure and weight update technique, the CIM macro can utilize a large external SRAM as its own storage with slight power overhead. Therefore, there is no need to enlarge CIM capacity with multi-SRAM cells in each CIM unit. The ping-pong structure with the fixed-weight path provides either 1.47× energy reduction by eliminating precharge/discharge similar to [6], or 2× transistor-count reduction at the same power compared with 12T SRAM cells [2].

The low-MACV adder tree in Fig. 16.3.5 contains two stages with the high-bit-position computation circuits discarded. We have observed that most of the CIM MAC results (out of a maximum of N) reside in a small region ($<N/2$ or $<N/4$). The FP-to-INT alignment further contributes to this phenomenon, which motivates random sparsity optimization for the digital CIM. The 1st stage of the 128×1b adder tree accumulates 16×1b to a 4b result, while the 2nd stage accumulates 8×4b to a 6b result. Following the MACV result distribution, the 1st/2nd stage can discard the computation for the high 1b/2b (or high 2b/3b) in the conservative (or aggressive) mode. This work implements the conservative solution to avoid the accuracy loss. Compared with a normal adder tree, the conservative/aggressive low-MACV adder tree offers 9.0%/13.9% power reduction with no accuracy loss.

The measurement results of the fabricated 28nm CIM processor are shown in Fig. 16.3.6. This chip can work at 10-400MHz with 0.469-0.9V digital voltage and 0.397-0.9V CIM voltage, while the best system energy efficiency is at 50MHz with 0.485V (digital), 0.458V (CIM). This chip is verified on several NN models with INT8/4/FP16 bit-precision on the Cifar-10 and ImageNet datasets. From dense models to the average of sparse models, the chip achieves 275-1615TOPS/W@INT4 and 17.2-91.3TOPS/W@FP16 macro energy efficiency for inference tasks. In summary, this work introduces an intensive-CIM sparse-digital architecture, and demonstrates an accurate and energy-efficient FP CIM chip. The INT4/FP16 macro energy efficiency is 6.99×/1.97× compared with the state-of-the-art CIM processor [5]. Fig. 16.3.7 shows the die photo and chip summary.

Acknowledgement:
This work was supported in part by National Key R&D Program 2018YFA0701500; NSFC Grant 62204256, 61888102, 61890944, 61725404, 61934005, 61720106013; Beijing Nova Program Z211100002121125; China Postdoctoral Science Foundation BX20220330; and the Strategic Priority Research Program of Chinese Academy of Sciences Grant XDB44000000.

References:
[1] D. Wang et al., "DIMC: 2219TOPS/W 2569F2/b Digital In-Memory Computing Macro in 28nm Based on Approximate Arithmetic Hardware," *ISSCC*, pp. 266-267, 2022.
[2] H. Fujiwara 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," *ISSCC*, pp. 186-187, 2022.
[3] K. Ueyoshi et al., "DIANA: An End-to-End Energy-Efficient Digital and ANAlog Hybrid Neural Network SoC," *ISSCC,* pp. 256-257, 2022.
[4] J. Lee et al., "A 13.7 TFLOPS/W Floating-Point DNN Processor Using Heterogeneous Computing Architecture with Exponent-Computing-in-Memory," *IEEE Symp. VLSI Circuits*, 2021.
[5] F. Tu et al., "A 28nm 29.2 TFLOPS/W BF16 and 36.5 TOPS/W INT8 Reconfigurable Digital CIM Processor with Unified FP/INT Pipeline and Bitwise In-Memory Booth Multiplication for Cloud Deep Learning Acceleration," *ISSCC*, pp. 254-255, 2022.
[6] J. Su et al., "A 28nm 384kb 6T-SRAM Computation-in-Memory Macro with 8b Precision for AI Edge Chips," *ISSCC*, pp. 250-251, 2021.

2023 IEEE International Solid- State Circuits Conference (ISSCC) | 978-1-6654-9016-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/ISSCC42615.2023.10067779

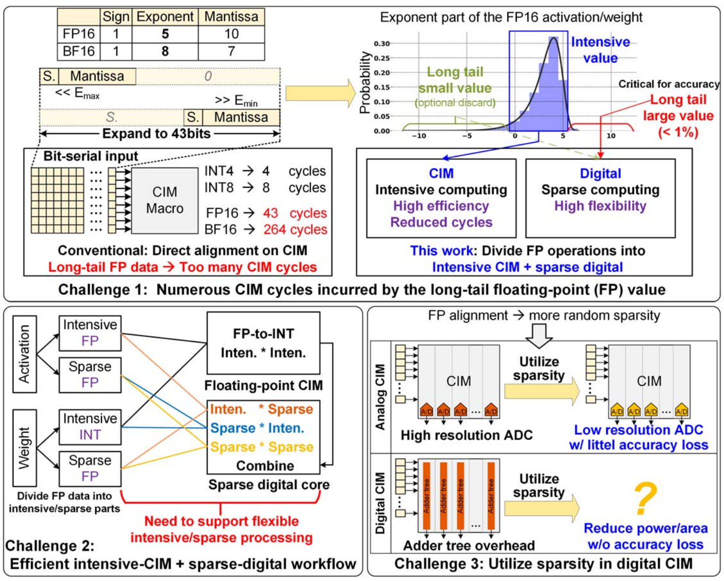


Figure 16.3.1: Challenges for energy-efficient and accurate floating-point (FP) CIM.

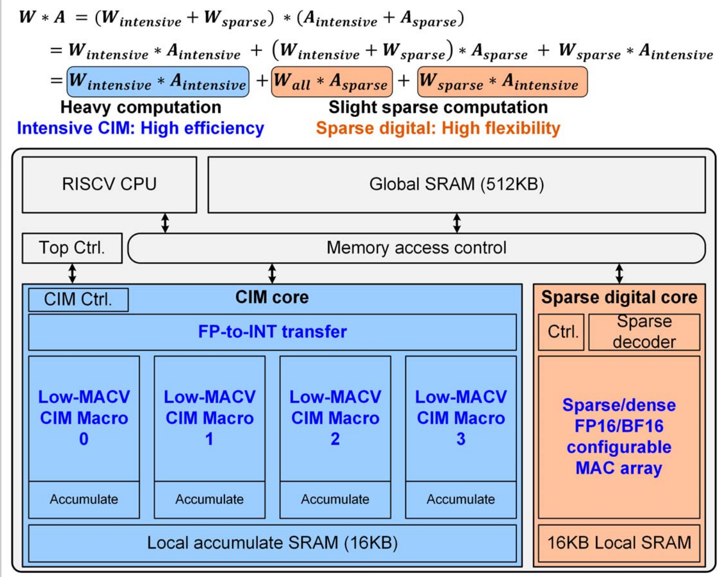


Figure 16.3.2: Overall architecture of the FP CIM processor with intensive-CIM sparse-digital division.

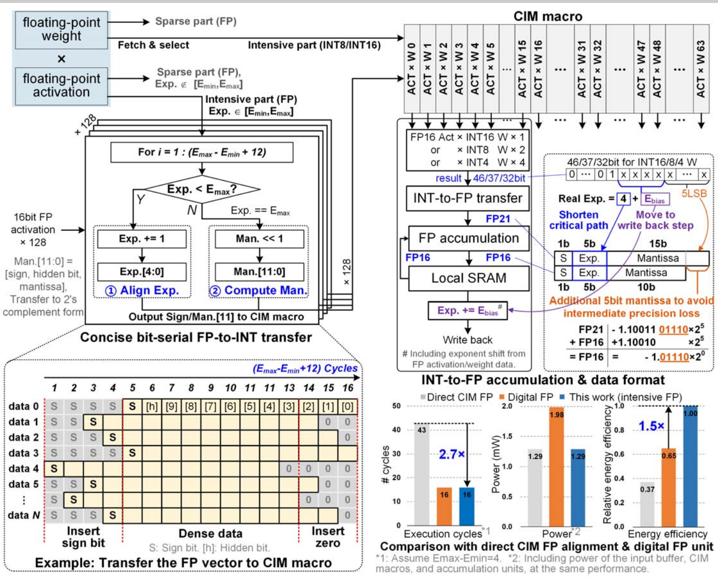


Figure 16.3.3: The FP-to-INT CIM workflow for intensive FP operations with cycle count reduction.

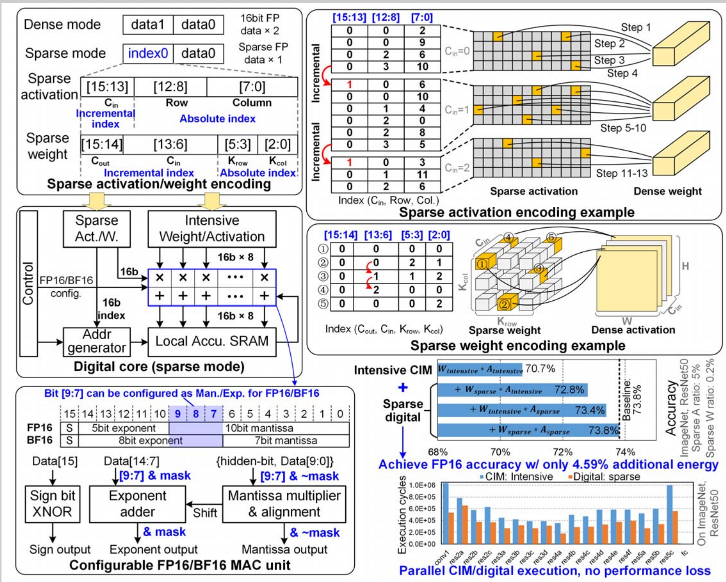


Figure 16.3.4: Flexible sparse-digital core for various sparse FP operations to ensure high accuracy.

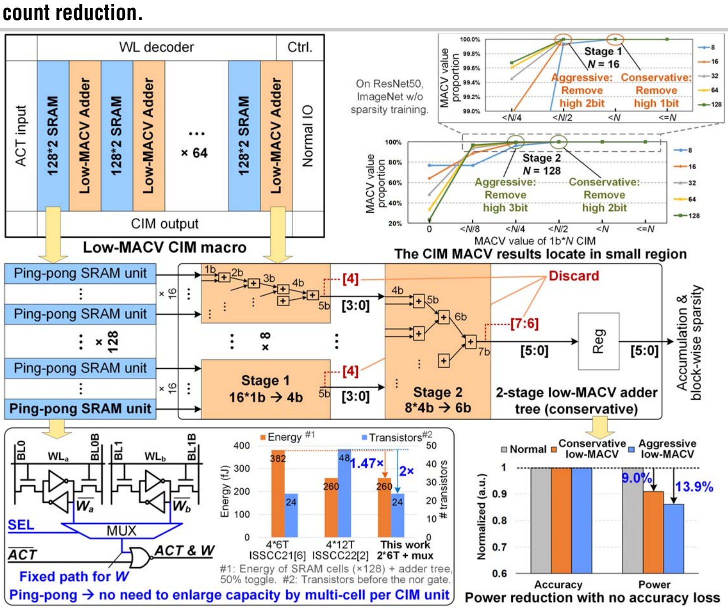
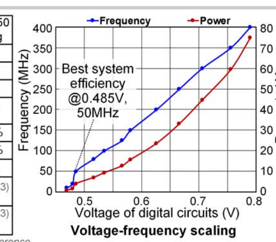


Figure 16.3.5: Energy-efficient low-MACV CIM macro to utilize random sparsity without accuracy loss.

Model	VGG16 inference		ResNet50 inference		ConvNeXt-T inference	ResNet50 training
Dataset	Cifar-10		ImageNet			
Activation	**4**	**FP16**	**8**	**FP16**	**FP16**	**FP16**
Weight	**4**	**FP16**	**8**	**FP16**	**FP16**	**FP16**
Average FP cycles (E_{max}-E_{min}+12)	**--**	**17.93**	**--**	**18.53**	**21.52**	**19.60**
Baseline accuracy	94.20%		80.86%		82.10%	80.86%
Chip accuracy (2)	90.03%	91.57%	76.92%	77.85%	77.98%	80.86%
Execution time	0.523ms	9.35ms	97.7ms	413ms	263ms	2.00s
CIM macro energy effi. (TOPS/W)	**1615**	**91.3**	**106**	**22.7**	**43.9**	**14.1** (3)
System energy effi. (TOPS/W)	**300**	**16.9**	**19.7**	**4.22**	**8.17**	**2.61** (3)

(1) At the best system energy efficiency point. (2) Pre-train with block-wise sparsity for the inference tasks. Trade-off accuracy with energy efficiency. (3) Dense training w/o block-wise sparsity.

Results on test INT/FP NN models (1)



	ISSCC22 [1]	ISSCC22 [2]	ISSCC22 [3]	VLSI21 [4]	ISSCC22 [5]	This work
Technology	28nm	5nm	22nm	28nm	28nm	**28nm**
Area (mm^2)	0.033	0.013	10.2	5.8	6.7	**4.54**
Activation	1b	INT4/8	7b (Analog) 2/4/8b (Digital)	BF16	INT8/16, BF16/FP32	**INT4/8, FP16/BF16**
Weight	1b	INT4/8	Ternary (Analog) 2/4/8b (Digital)	BF16	INT8/16, BF16/FP32	**INT4/8, FP16/BF16**
Sparsity	Approximate adder	N/A	N/A	Activation sparsity	N/A	**Low-MACV adder + block-wise**
FP support	N/A	N/A	N/A	Exponent CIM	Pure CIM (long-tail FP cost unmentioned)	**Intensive CIM + sparse digital**
Macro energy effi. (TOPS/W) (1)	128 (INT4 eq.)	254 (INT4)	600 (7b/Ternary)	(2) 13.7 (BF16)	231 (INT4) (3) 57.8 (INT8) 46.2 (BF16)	**275 - 1615 (INT4)** (4) **68.7 - 403 (INT8)** **17.2 - 91.3 (FP16)**

System energy efficiency is not compared due to the variance in system-level configuration (SRAM sizes, CPUs, etc.).
(1) Maximum reported value in [1-5] for fair comparison. One operation (OP) represents one multiplication or addition. (2) Including external control and SIMD modules. (3) The processing of the long-tail floating-point data is not mentioned. (4) From dense models to average of test sparse NN models.

Comparison with the state-of-the-art CIM processors

Figure 16.3.6: Measurement results and comparison with the state-of-the-art CIM processors.

Technology	28nm
Chip area	2.52mm×1.8mm
CIM macro area	0.56mm×0.12mm ×4
Weight prec.	INT4/8, FP16/BF16
Activation prec.	INT4/8, FP16/BF16
Voltage (CIM)	0.397 - 0.90V
Voltage (digital)	0.469 - 0.90V
Frequency	10 - 400MHz
CIM power *1	0.13 - 13.7mW
System power *1	0.87 - 74.9mW
Performance *2	1.64 - 9.63TOPS (INT4/INT4)
CIM macro energy efficiency *2	275 - 1615TOPS/W (INT4/INT4) 68.7 - 403TOPS/W (INT8/INT8) 17.2 - 91.3TOPS/W (FP16/FP16)
System energy efficiency *2	51 - 300TOPS/W (INT4/INT4) 12.8 - 75.0TOPS/W (INT8/INT8) 3.2 - 16.9TOPS/W (FP16/FP16)

*1: At 10-400M with 0.469-0.79V (digital) and 0.397-0.78V (CIM).
*2: from dense to average block-wise sparsity on test models. Results of BF16/FP16 are similar in this work. Assume average FP cycles = 16 for the dense situation.

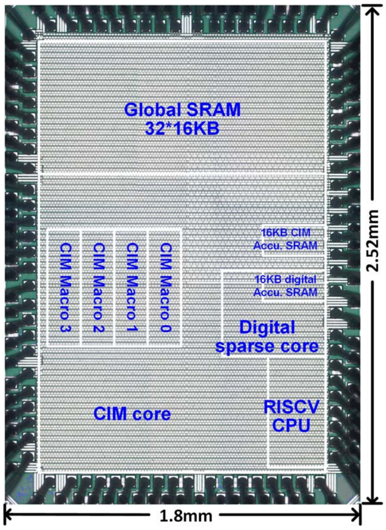


Figure 16.3.7: The chip metrics and die photo.

978-1-6654-9016-0/23/$31.00 ©2023 IEEE