16.4 TensorCIM: A 28nm 3.7nJ/Gather and 8.3TFLOPS/W FP32 Digital-CIM Tensor Processor for MCM-CIM-Based Beyond-NN Acceleration

Fengbin Tu, Yiqi Wang, Zihan Wu, Weiwei Wu, Leibo Liu, Yang Hu, Shaojun Wei, Shouyi Yin

Tsinghua University, Beijing, China

Applications such as Graph Convolutional Networks (GCNs) and Deep Learning Recommendation Models (DLRMs) have computational and data-movement requirements beyond those seen in typical NN processing. Such beyond-NN applications typically consist of Sparse Gathering (SpG) and Sparse Algebra (SpA). SpG comprises gathering and reducing tensors from sparsely distributed addresses (in GCN's aggregation phase and DLRM's embedding layer). SpA refers to NN-based sparse tensor multiplication for the gathered tensors (in GCN's combination phase and DLRM's fully-connected layer). Due to the large application size, data movement is the main bottleneck for beyond-NN acceleration. Digital Computing-In-Memory (CIM) is an efficient and precise architecture for reducing data movement [1-3]. Large-scale beyond-NN acceleration motivates the demand for scaling out digital CIM processors. However, a large monolithic chip has low-yield issues due to manufacturing defects [4], which are more severe for CIM's memory-intensive logic. A Multi-Chip-Module (MCM) provides a high-yield solution for CIM scaling by integrating multiple smaller chiplets in one package [5]. Fig. 16.4.1 shows a typical MCM-CIM system with 4 CIM chiplets, but it has two challenges for beyond-NN acceleration: 1) SpG involves repeated off-chip DRAM access, inter-chiplet access and redundant reduction operations, which increases inter-chiplet bandwidth requirements and processing latency. 2) SpA suffers from (2a) inter-CIM workload imbalance and (2b) intra-CIM under-utilization, due to irregular tensor sparsity.

This work proposes TensorCIM as the CIM processor chiplet (see Fig. 16.4.2). It consists of 8 Input-LookAhead CIM (ILA-CIM) cores with 8 Redundancy-Eliminated Gathering Managers (REGM), an Equal Operation-based CIM Initializer (EOCI), a 256KB Input Buffer (IB), a 256KB Global Buffer (GB), a SIMD core and a top controller. In beyond-NN acceleration, all chiplets are first configured in SpG mode, fetching feature tensors from their DRAM according to the access tensor in the IB. Each CIM core maintains features via its REGM, performs feature reduction and stores the gathered tensors in the GB. Then, all chiplets are configured in SpA mode and initialize their CIM with weight tensors. The IB loads the gathered tensors and feeds them to the CIM for computation. CIM outputs are written to the GB with activation functions performed by the SIMD core.

TensorCIM has three features: 1) In the SpG mode, the access tensor and SpG workload are repartitioned to maximize feature locality and minimize inter-chiplet access. The REGM is designed to dynamically maintain frequently accessed features and reduction results in the CIM. This eliminates redundant accesses and reductions, lowering inter-chip bandwidth requirements. 2a) In SpA mode, each chiplet relies on an EOCI to calculate effective MAC operations based on the non-zero distribution of gathered and weight tensors, initializing CIM macros with a balanced inter-CIM workload at the subarray level. 2b) The TensorCIM exploits an ILA-CIM architecture to look ahead at future inputs to fully utilize CIM logic, keeping all CIM subarrays busy to lower computation latency.

Figure 16.4.3 shows TensorCIM's SpG workflow and the REGM. In initialization, the access tensor is reordered and repartitioned for the chiplets, with features distributed in different DRAMs based on the chiplet's access pattern. Then, each chiplet stores its access tensor in the IB, and performs SpG mainly using its own DRAM with minimum inter-chiplet access. During SpG, off-chip DRAM or inter-chiplet access occurs and triggers the REGM only when the required feature is not found in CIM macros. Due to the large tensor (e.g. GCN's huge graph), the CIM stores frequently accessed features for reuse, as determined by the current status and requiring dynamic management for gathering. The gathered features' access tensor (e.g. f0A, f3A) is loaded into the REGM and sent to frequently accessed feature filters. The feature with less frequent access is pushed into the eviction candidate FIFO. If the CIM macros are full, the head feature is popped out and the REGM generates a configuration to evict the feature out of the CIM. In addition, the REGM implements reduction redundancy speculators to store high-reuse reduction results into the CIM (see f0+f3 reuse on the top right). Compared to SpG without repartition and the REGM, our solution achieves 1.74× fewer DRAM accesses, 3.55× fewer inter-chiplet accesses and 1.35× fewer reductions for a typical GCN layer on the 4-chiplet system.

Figure 16.4.4 shows the EOCI that balances inter-CIM workload for SpA. EOCI performs CIM initialization for weight tensor (WT) and gathered tensor (GT) assignment before SpA computation. The EOCI reads the GT's non-zero column IDs from the GB and the WT's non-zero row IDs from DRAM. The effective operation recognizer finds the intersections between the two ID sequences. An intersection implies effective MACs exist between the corresponding GT column and WT row (e.g. ID0,4,7 marked on top left). For each new intersected ID, the EOCI triggers a WT/GT assignment. The IB loads the GT column and the on-demand weight fetcher loads the WT row. The EOCI utilizes the balanced inter-CIM distributor to manage GT and WT assignment in the CIM subarray level for inter-CIM workload balance. The distributor maintains the current MAC count assigned to each subarray. After calculating the intersected ID's MAC count, the distributor first finds the least-busy core and then assigns the ID to its least-operation subarray. In the top-right example, we fill up the CIM with the WT's non-zero values in Row0,10,4,13 for Core0, and Row7,15,9,16 for Core1. With corresponding GT columns assigned to the IB, the MAC operation count is the same for all subarrays. On a typical GCN layer, the EOCI reduces the inter-CIM imbalance ratio from 5.51 to 1.24, and avoids unnecessary WT access by 1.28×.

Figure 16.4.5 shows the ILA-CIM architecture that improves intra-CIM utilization in SpA. Each CIM core consists of 4 ILA-CIM macros that share the same input lookahead unit and Booth controller. The core reads 2 non-zero inputs per cycle and pushes them to the subarray FIFO. The macro includes 16 subarrays and processes 16 inputs together when all FIFOs are ready. Thus, the macro is able to look ahead for future inputs and activate all subarrays' CIM logic together. Each subarray has 16×128 6T-SRAM cells and 4 Booth FP32 subarray multipliers. We use a full-digital CIM architecture with 16 rows time-sharing the subarray multipliers to balance computing accuracy and memory density [2]. Based on the input GT's Col.ID, one row of the subarray is read out, and multiplied by the input mantissa's bitwise Booth signals with a ~50% cycle reduction [3]. After shift-accumulation for input bits, the products are normalized according to input and weight exponents. The output merger collects all the results from subarrays, and converts the output indexes to GB addresses. The CIM outputs are added up with the corresponding partial sums and written to the GB. The ILA-CIM is also applicable for SpG's reduction operations. Our SpA optimizations achieve intra-CIM utilization of 95.4% and 5.28× total speedup, with area overhead of only 1.96% for the EOCI and 2.64% for the ILA logic.

Figure 16.4.6 shows measurement results for a 28nm TensorCIM processor and 4-chiplet MCM-CIM system. Fig. 16.4.7 presents the MCM package, single-chip package, die photo, voltage-frequency scaling curves and summary table. The chip operates at 0.6-to-1.0V supply, 115-290MHz with 320Mbps inter-chiplet bandwidth. Evaluation is conducted on two typical beyond-NN applications, GCN on the Pubmed Dataset and DLRM on the MovieLens Dataset. Since they require high-precision computation, we use FP32 to maintain accuracy. The proposed techniques achieve total energy savings of 4.58× for GCN and 3.52× for DLRM on the MCM-CIM system. TensorCIM can also support INT8/INT16 through bypassing exponent-related processing in the ILA-CIM. Owing to better sparsity exploitation, TensorCIM's peak algebra energy efficiency reaches 85.0TOPS/W at INT8 (2.17× over [2]) and 8.3TFLOPS/W at FP32 (2.24× over [3]) at 0.65V, 175MHz. The total energy consumption for GCN on Pubmed is 0.31mJ, which is 5.64× lower than a simulated MCM-CIM system based on the digital CIM that supports FP32 [3]. The state-of-the-art MCM-CIM [5] is based on analog CIM whose INT3/4 precision is not suitable for beyond-NN applications. Its direct MCM access may cause high communication during SpG. TensorCIM features optimized SpG/SpA support with a high-precision (up to FP32) digital CIM and redundancy-eliminated MCM access.

Acknowledgement:
This work was supported in part by NSFC Grant 62125403, Grant U19B2041, and Grant 92164301; in part by the National Key Research and Development Program under Grant 2021ZD0114400; in part by Beijing National Research Center for Information Science and Technology; and in part by the Beijing Advanced Innovation Center for Integrated Circuits. The corresponding author of this paper is Shouyi Yin (yinsy@tsinghua.edu.cn).

References:

[1] Y. 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. 252-253, 2021.
[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] F. Tu et al., "A 28nm 29.2TFLOPS/W BF16 and 36.5TOPS/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.
[4] A. Arunkumar et al., "MCM-GPU: Multi-Chip-Module GPUs for Continued Performance Scalability," *ISCA*, pp. 320–332, 2017.
[5] H. Zhu et al., "COMB-MCM: Computing-on-Memory-Boundary NN Processor with Bipolar Bitwise Sparsity Optimization for Scalable Multi-Chiplet-Module Edge Machine Learning," *ISSCC*, pp. 250-251, 2022.

2023 IEEE International Solid-State Circuits Conference (ISSCC) | 978-1-6654-9016-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/ISSCC42615.2023.10067285

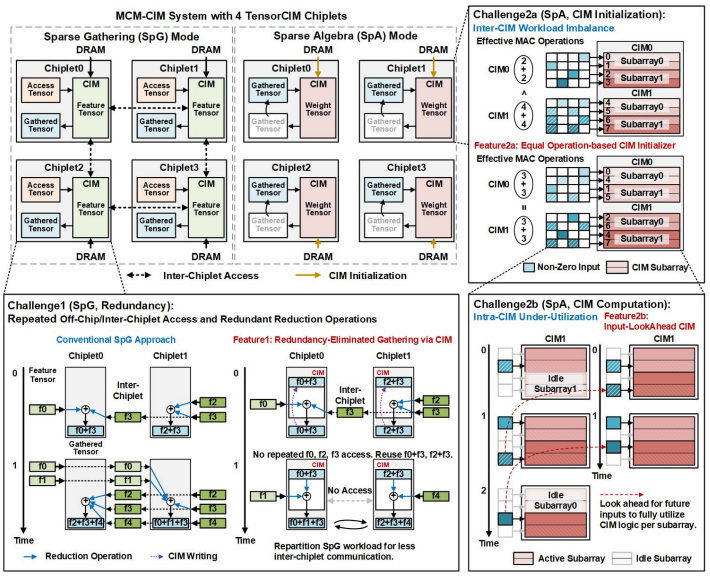


Figure 16.4.1: Challenges of designing an MCM-CIM system for beyond-NN acceleration and TensorCIM's three features.

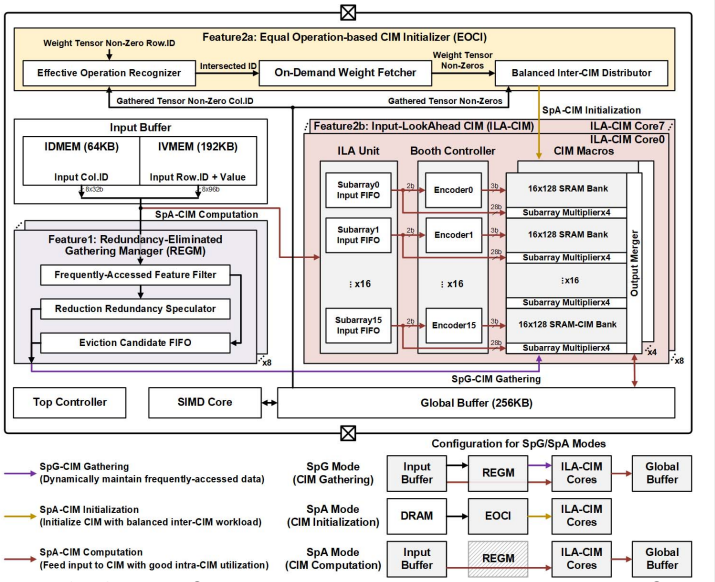


Figure 16.4.2: TensorCIM's overall architecture and configuration for Sparse Gathering (SpG)/Sparse Algebra (SpA) modes.

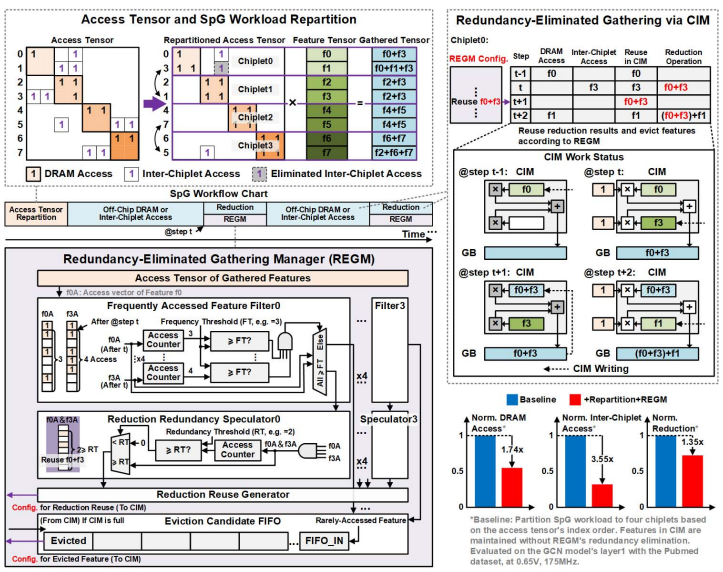


Figure 16.4.3: TensorCIM's SpG workflow and redundancy-eliminated gathering manager (REGM).

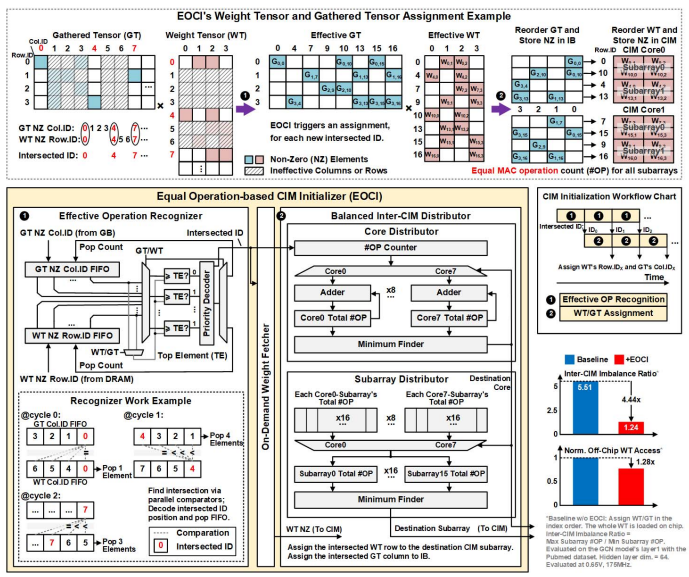


Figure 16.4.4: Equal operation-based CIM initializer (EOCI) that balances inter-CIM workload for SpA.

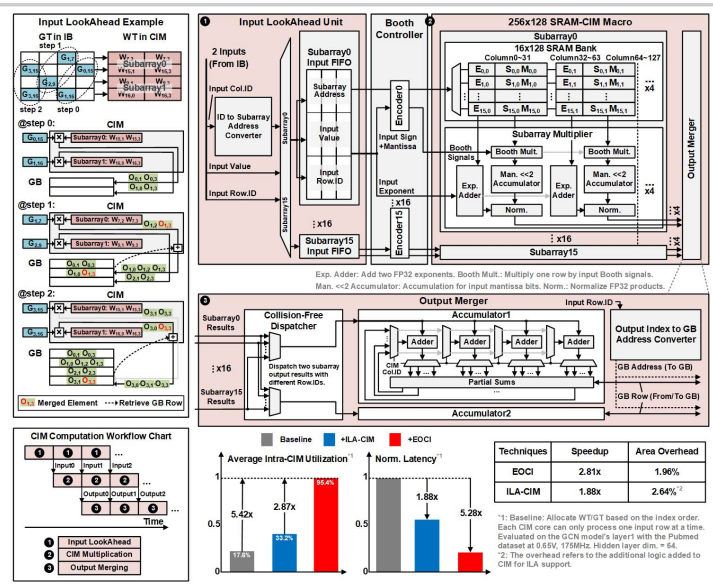
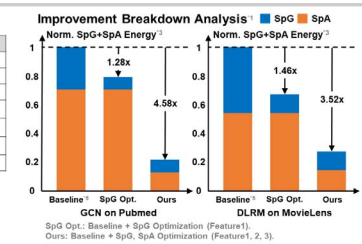


Figure 16.4.5: Input-LookAhead CIM (ILA-CIM) architecture that improves intra-CIM utilization for SpA.

Evaluation on Typical Beyond-NN Applications[1]

Task	Graph-based Learning	Recommendation
Dataset	Pubmed	MovieLens
Model	GCN	DLRM
Data Precision[2]	FP32	FP32
Execution Time	2.11ms	2.07ms
Gathering Efficiency[3]	3.7nJ/Gather	2.3nJ/Gather
Algebra Efficiency[3,4]	7.6TFLOPS/W	6.4TFLOPS/W
Total Energy Saving[3,5]	4.58x	3.52x

*1: Tested on the MCM-CIM system with 4 TensorCIM chiplets. Measured at 0.65V, 175MHz for high-efficiency evaluation.
*2: Work at FP32 to keep no accuracy loss for beyond-NN applications.
*3: Energy excludes off-chip memory to focus on the MCM evaluation.
*4: One operation (OP) represents one multiplication or one addition.
*5: The baseline is the same MCM but without the proposed techniques.

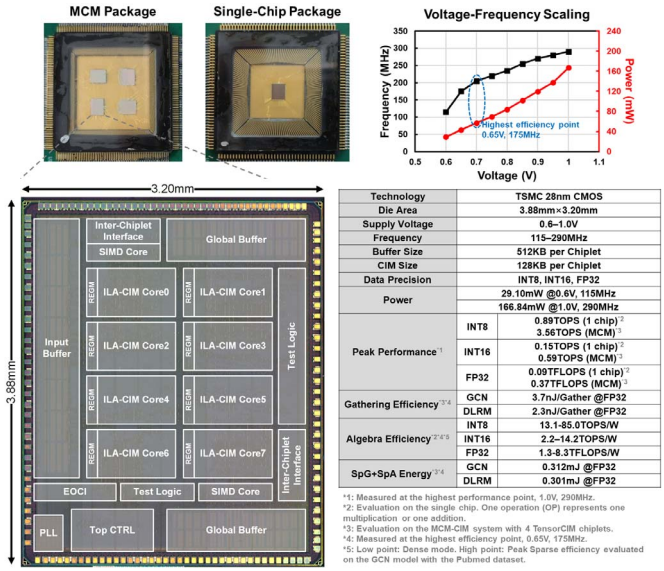


Comparison with State-of-the-art CIM Processors

	ISSCC'22-11.6 [2]	ISSCC'22-15.3 [5]	ISSCC'22-15.5 [3]	This Work
CIM Implementation	Digital CIM	Analog CIM	Digital CIM	**Digital CIM**
Data Precision	INT4/8	INT3/4	INT8/16, BF16, √ FP32	**INT8/16, √ FP32**
Work Mode	Dense Algebra[1]	Dense Algebra[1]	Dense Algebra[1]	**√ SpG/SpA**
MCM Access Optimization	×	× Direct Access	×	**√ Reduce Access by Locality**
Technology	5nm	28nm	28nm	**28nm**
Supply Voltage (V)	0.5-0.9	0.63-0.95	0.6-1.0	**0.6-1.0**
Frequency (MHz)	360-1440	30-105	50-220	**115-290**
Die Area (mm²)[2]	0.0133 (macro)	N/A	6.69	**12.42**
Power (mW)[2]	18.7[3]	29.5	12.5-69.4	**29.1-166.8**
Peak Performance[2] (TOPS or TFLOPS)	0.74	0.97	1.35 (INT8)	**0.89 (1 chip), 3.56 (MCM) INT8**[4]
	N/A	N/A	0.14 (FP32)	**0.09 (1 chip), 0.37 (MCM) FP32**[4]
Algebra Efficiency[2] (TOPS/W or TFLOPS/W)	39.1 (INT8)[3]	32.9 (INT4/3)	36.5 (INT8)	**13.1-85.0 (INT8)**[5,6]
	No FP precision	No FP precision	3.7 (FP32)	**1.3-8.3 (FP32)**[5,6]
SpG+SpA Energy[7] on GCN, Pubmed Dataset	N/A (w/o FP32 support)	N/A (w/o FP32 support)	SpG: 35.1nJ/Gather Total: 1.76mJ (5.64x)	**SpG: 3.7nJ/Gather**[5] **Total: 0.31mJ (1x)**[5]

*1: Prior CIM processors can exploit sparsity for power reduction, but not for cycle saving. They don't have optimization for SpG or MCM access.
*2: Evaluation on the single chip. One operation (OP) represents one multiplication or one addition.
*3: On-chip buffers are included for fair comparison. Buffer capacity is 512KB, same as TensorCIM.
*4: Measured at the highest performance point, 1.0V, 290MHz.
*5: Measured at the highest efficiency point, 0.65V, 175MHz.
*6: Low point: Dense mode. High point: Peak Sparse efficiency evaluated on the GCN model with the Pubmed dataset.
*7: Evaluation on the MCM-CIM system with 4 chiplets. Prior work's results are based on simulation.

Figure 16.4.6: Measurement results and comparison with state-of-the-art CIM processors.



Technology		TSMC 28nm CMOS
Die Area		3.88mm×3.20mm
Supply Voltage		0.6–1.0V
Frequency		115–290MHz
Buffer Size		512KB per Chiplet
CIM Size		128KB per Chiplet
Data Precision		INT8, INT16, FP32
Power		29.10mW @0.6V, 115MHz
		166.84mW @1.0V, 290MHz
Peak Performance*1	INT8	0.89TOPS (1 chip)*2 3.56TOPS (MCM)*3
	INT16	0.15TOPS (1 chip)*2 0.59TOPS (MCM)*3
	FP32	0.09TFLOPS (1 chip)*2 0.37TFLOPS (MCM)*3
Gathering Efficiency*3*4	GCN	3.7nJ/Gather @FP32
	DLRM	2.3nJ/Gather @FP32
Algebra Efficiency*2*4*5	INT8	13.1-85.0TOPS/W
	INT16	2.2–14.2TOPS/W
	FP32	1.3-8.3TFLOPS/W
SpG+SpA Energy*3*4	GCN	0.312mJ @FP32
	DLRM	0.301mJ @FP32

*1: Measured at the highest performance point, 1.0V, 290MHz.
*2: Evaluation on the single chip. One operation (OP) represents one multiplication or one addition.
*3: Evaluation on the MCM-CIM system with 4 TensorCIM chiplets.
*4: Measured at the highest efficiency point, 0.65V, 175MHz.
*5: Low point: Dense mode. High point: Peak Sparse efficiency evaluated on the GCN model with the Pubmed dataset.

Figure 16.4.7: MCM package, single-chip package, die photo, voltage-frequency scaling curves and summary table.

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