A 13.7 TFLOPS/W Floating-point DNN Processor using Heterogeneous Computing Architecture with Exponent-Computing-in-Memory

Juhyoung Lee, Jihoon Kim, Wooyoung Jo, Sangyeob Kim, Sangjin Kim, Jinsu Lee and Hoi-Jun Yoo
Dept. of EE, KAIST, 291 Daehak-ro, Yuseong-gu, Daejeon, 305-701, Republic of Korea
Email: juhyoung@kaist.ac.kr

Abstract

An energy-efficient floating-point DNN training processor is proposed with heterogenous bfloat16 computing architecture using exponent computing-in-memory (CIM) and mantissa processing engine. Mantissa free exponent calculation enables pipelining of exponent and mantissa operation for heterogenous bfloat16 computing while reducing MAC power by 14.4 %. 6T SRAM exponent computing-in-memory with bitline charge reusing reduces memory access power by 46.4 %. The processor fabricated in 28 nm CMOS technology and occupies 1.62×3.6 mm^2 die area. It achieves 13.7 TFLOPS/W energy efficiency which is 274× higher than the previous floating-point CIM processor.

Introduction

Computing-in-memory (CIM) architecture can reduce the large on-chip memory access power consumption for the high energy-efficiency. Although previous CIM processors [1]-[2] showed remarkable energy efficiency, most of them supported only fixed-point precision which cannot support DNN training. Commercial DNN accelerators including Google's TPUv2, and ARM's Armv8-A usually use bfloat16 (BFP16) precision to maintain high accuracy in DNN training [3]. Therefore, a CIM processor that supports BFP16 is essential for accurate and energy-efficient DNN training in edge devices.

However, the existing CIMs cannot realize energy-efficient DNN training because of the heterogeneous characteristic of floating-point (FP) computing with a mantissa and an exponent, as shown in Fig. 1(a). While the exponent operation requires only simple addition and subtraction, the mantissa operation requires complex operations such as multiplication, shifting, and leading-one detecting. Therefore, CIM cannot satisfy both exponent and mantissa operations at once due to complicated complexity and area overhead. As a result, the training with CIM suffers from excessive operation cycles for FP MAC due to repetitive read/write for partial sum results. Indeed, the previous FP-CIM processor [4], which implemented both exponent and mantissa as CIM, required >5000 cycles for a BFP16 multiplication and an FP32 accumulation.

In this paper, we propose a floating-point DNN processor that can realize the FP MAC in 2 cycles with the following 3 key features; 1) A heterogenous computing architecture with an exponent-computing-in-memory (ECIM) and a mantissa-processing-engine (MPE) to support energy-efficient bfloat16 operation, 2) mantissa-free-exponent-computation (MFEC) to minimize communication cost between exponent and mantissa while reducing MAC power by 14.4 %, 3) 6T SRAM ECIM with bitline charge reusing to reduce overall memory access power by 46.4 %.

Proposed Processor w/ Heterogenous FP Computation

Fig. 2 shows the overall architecture of the proposed processor. It consists of 16 heterogenous-exponent-mantissa-training-core (HEMTC), an aggregation and activation core for accumulate partial sums from HEMTC, a top RISC controller, and a 1-D SIMD Core. Each HEMTC integrates a total of 24.25 KB memories that can store exponent and mantissa separately, including 8 KB of weight-ECIM and 2 KB of output-ECIM. HEMTC also integrates an exponent peripheral circuit shared by two ECIMs, a mantissa processing engine (MPE) line consisting of 16 MPEs, a feature map zero skip controller, and a 16-way activation unit. HEMTC can skip the computation for zero feature map value by feeding only non-zero feature map values and their indexes to the ECIMs and MPE line.

Fig. 3(a) shows a comparison of conventional 2-stage FP MAC and the proposed MFEC. The conventional FP MAC performs a normalization process for every accumulation cycle, which leads to a long critical path and communication cost between exponent and mantissa. The proposed MFEC does not normalize mantissa every cycle and replace it with an overflow counter and an accumulation register. The normalization is performed once after partial sum is generated. By selecting the appropriate accumulation register width (>21 bit), MFEC can perform FP MAC without calculation error. Fig. 3(b) shows the accuracy of MFEC according to the accumulation register width. The MFEC not only enables pipelining of exponent and mantissa computation for heterogenous FP computing but also reduces the total MAC power by 14.4 % and total MAC area by 11.7 % through shortening the critical path.

Fig. 4(a) shows the architecture of the proposed ECIM. ECIM is composed of the weight-ECIM cell array, output-ECIM cell array, shared exponent peripheral, WL drivers, CIM Local Array (CLA) decoders, and normal I/O interfaces. Each ECIM adopts a hierarchical BL structure, and the number of CLAs connected to the global BL (GBL) is 16 for weight-ECIM and 4 for output-ECIM. Each CLA consists of 32 6T bitcells connected to a local BL (LBL), a precharger, 2 write switches, and 2 GBL drivers. In the case of weight-ECIM, the weight is stored in each bitcell and LBL/LBLB are precharged with the exponent of the input feature map or error. The near-memory adder/comparator loads the result of in-memory computing from CLA and finalize exponent calculation.

Fig. 4(b) shows the operation flow of the ECIM. The in-memory 'and/nor' operation is performed by applying a lower WL voltage of 0.57V~0.7V after precharging the feature map value. The result of the 'and/nor' operation is transferred to GBL/GBLB through the GBL driver. Since GBL can maintain the previous charge with a hierarchical BL structure, the GBL charge can be reused if the calculated exponent results in the previous cycle and current cycle are the same. By utilizing spatial-wise and channel-wise scale similarity in DNN training, we can reuse >84 % of the GBL charge in ResNet-18 training scenario. Precharging, CIM, and GBL driving are pipelined between CLAs for high throughput, and each CLA creates partial sums that can be accumulated with each other. Fig. 4(c) shows the operational waveform of ECIM. By reducing precharge power with 6T CIM and GBL charge reusing, the proposed ECIM reduces memory access power by 46.4 %.

Implementation Results and Conclusions

Fig. 5 shows the proposed processor, which is fabricated in 28nm CMOS process. It occupies 5.832 mm^2 with a total of 396 KB on-chip SRAM including 160 KB ECIM. It can operate at 0.76V-1.1V supply with a maximum frequency of 250 MHz. Its peak performance is 119.4 GFLOPS for 0% sparsity and 662 GFLOPS for 90% sparsity. Fig. 6 shows voltage-frequency scaling and energy efficiency measurement results. We observed that the processor operates at 40 MHz and 0.76V, but the processor does not operate at supply voltage which is lower than 0.76V even in lower operating frequency. This is because the WL voltage is bounded at 0.58V to guarantee charge moving between bitcell and LBL. The energy efficiency, as measured on a convolutional layer (512 channels)

with consideration of PE utilization is 1.43 TFLOPS/W for 0% sparsity and 13.7 TFLOPS/W for 90% sparsity operating at 40 MHz and 0.76 V. Table. 1 shows the comparison table. The proposed chip support BFP16 computing for DNN training while achieving ×274 higher energy efficiency than previous FP CIM processor [4]. In conclusion, an energy-efficient floating-point DNN training processor is fabricated using heterogenous bfloat16 computing architecture with exponent-computing-in-memory and bitline charge reusing.

References

[1] J. Yue, et al., "A 65nm Computing-in-Memory-Based CNN Processor with 2.9-to-35.8TOPS/W System Energy Efficiency Using Dynamic-Sparsity Performance-Scaling Architecture and Energy-Efficient Inter/Intra-Macro Data Reuse," ISSCC 2020.

[2] J. Kim, et al., "Z-PIM: An Energy-Efficient Sparsity Aware Processing-In-Memory Architecture with Fully-Variable Weight Precision," VLSI 2020.

[3] D. Kalamkar et al., "A study of BFLOAT16 for deep learning training," 2019, arXiv: 1905.12322.

[4] J. Wang et al., "A Compute SRAM with Bit-Serial Integer/Floating-Point Operations for Programmable In-Memory Vector Acceleration," ISSCC 2019.

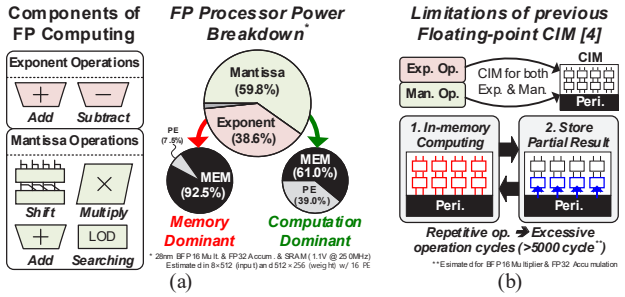


Fig. 1 (a) Heterogenous characteristic of floating-point computing (b) Limitation of previous floating point CIM [4].

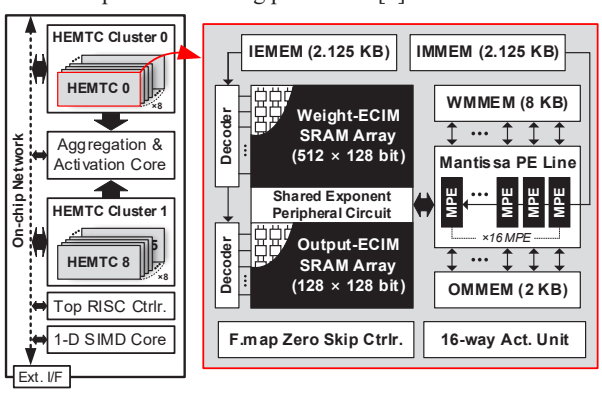


Fig. 2 Overall architecture of the proposed processor.

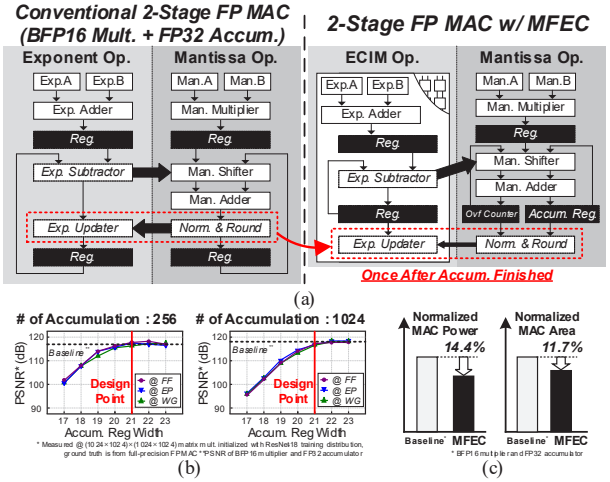
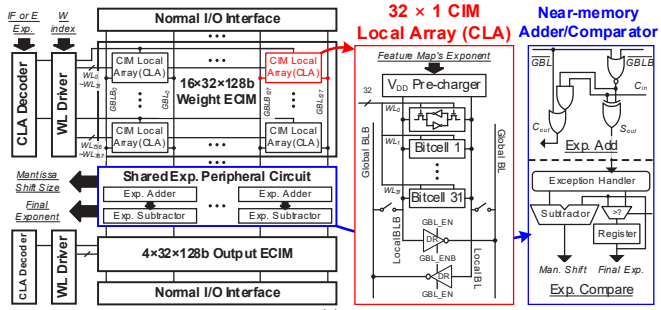
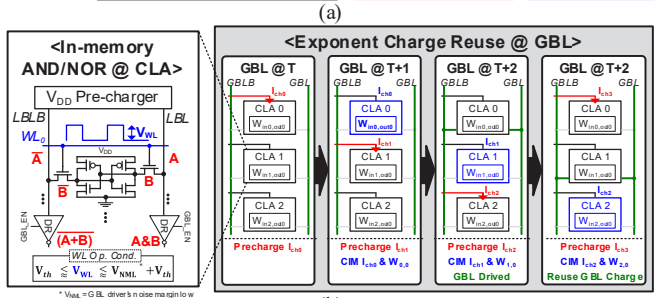


Fig. 3 (a) Conventional & proposed 2-stage FP MAC. (b) MFEC error according to accumulation register's width. (c) Comparison result.





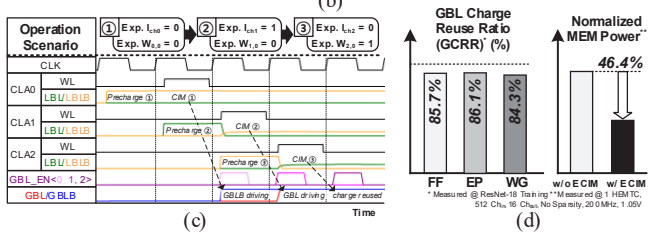
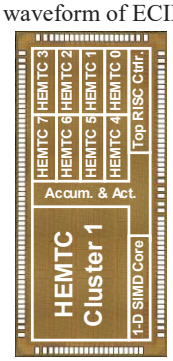


Fig. 4 (a) Overall architecture of ECIM (b) Operation flow of the exponent charge reusing and in-memory AND/NOR (c) Operation waveform of ECIM. (d) GCRR results & power reduction results.



	Specifications
Technology	28nm Logic CMOS
Die Area	1.62 mm × 3.6 mm
Precision	BFloat16
Supply Voltage	0.76 V – 1.1 V
Frequency	~ 250 MHz
Peak Performance [GFLOPS]	119.4* – 662.0**
Power Consumption [mW]	1.2** – 2.1* (5 MHz, 0.76 V)
	91.0** – 156.1* (250 MHz, 1.1 V)
Energy Efficiency [TFLOPS/W]	1.43* – 13.7** (40 MHz, 0.76 V)
	0.76** – 7.3* (250 MHz, 1.1 V)

* = Activation Sparsity 0%, ** = Activation Sparsity 90% (1 MAC = 2 OP)

Fig. 5 Chip photograph and performance summary.

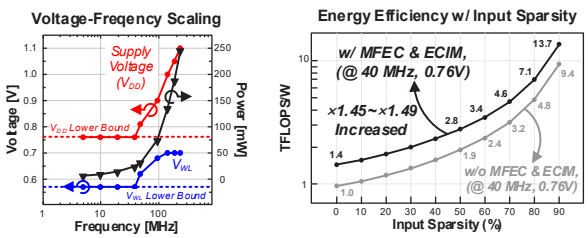


Fig. 6 Measurement results.

		ISSCC 19'[4]	VLSI 20'[2]	This Work
Process [nm]		28	65	28
Die Area [mm²]		2.55	7.568	5.832
Training		X	X	O
Sparsity Handling		X	O (Weight)	O (Feature Map)
Precision	Weight	FXP 1-16, FP 32	FXP 1-16	BFloat16
	Activation	FXP 1-16, FP 32	FXP 16	BFloat16
Supply Voltage [V]		0.6-1.1V	1.0	0.68-1.1
Memory Size		128 KB	4.75 KB	396 KB (ECIM : 160 KB)
Peak Performance		1.13 GFLOPS[5]	0.37[3] – 4.0[4] GOPS	119.4[1] – 662.0[2] GFLOPS
Peak Energy Efficiency		0.05 TFLOPS/W (1.14MHz, 0.6V)	0.31[3] – 3.07[4] TOPS/W (200MHz, 1.0V)	1.43[1] – 13.7[2] TFLOPS/W (40MHz, 0.76V)

1) BFloat16, Activation sparsity 0%, 2) BFloat16, Activation sparsity 90%, 3) 16b Weight, 0% Weight Sparsity, 4) 16b Weight, 90% Weight Sparsity, 5) Normalized to BFP16

Table. 1 Comparison table.