A Highly Energy-Efficient Binary BERT Model on Group Vector Systolic CIM Accelerator

***Abstract*—Transformer-based large language models (LLMs) impose significant bandwidth and compute challenges when deployed on edge devices. SRAM-based compute-in-memory (CIM) accelerators offer a promising solution to reduce data movement but are still limited by model size. This work develops a ternary weight splitting (TWS) binarization to obtain Brain-Floating-Point-16xINT1 (BF16×1-b) based transformers that exhibit competitive accuracy while significantly reducing model size compared to full precision counterparts. Then, a fully digital SRAM-based CIM accelerator is designed incorporating a bit-parallel SRAM macro within a highly efficient group vector systolic architecture, which can store one column of BERT-Tiny model with stationary systolic data reuse. The design in a 28nm technology only requires 2KB SRAM with an area of 2mm^2. It achieves a throughput of 6.55TOPS and consumes a total power of 419.74mW, resulting in a state-of-the-art area efficiency of 3.3TOPS/mm^2 and normalized energy efficiency of 20.98TOPS/W for BERT-Tiny model, demonstrating a 10.25× improvement in area efficiency and a 2.23× improvement in energy efficiency compared to other state-of-the-art counterparts. Additionally, our proposed configuration compresses the model size by 32% with only a 0.5% accuracy loss on SST-2.**

***Index Terms*—Compute-in-Memory, Binary, BERT, Systolic Array.**

I. Introduction

Edge computing necessitates on-device network inference, which underscores the need to optimize transformer models for resource-constrained environments without compromising accuracy. The demands impose additional constraints on both algorithm optimization and hardware implementation in terms of area, power, and storage. To address the challenges of deploying transformer models on edge devices, various optimization methodologies have been explored. One common approach involves reducing network size and minimizing hardware usage. Popular techniques [1], [2] include quantization, pruning and knowledge distillation.

Accelerating neural networks on edge devices often necessitates tailored hardware for specific dataflows. CIM architectures have emerged as promising candidates for MAC operations due to their potential for higher throughput and energy efficiency [3], [4]. While analog-domain CIM approaches have been explored, they often suffer from device nonlinearity and parasitic effects, limiting their performance [4]–[6]. Digital CIM architectures address these limitations [7]–[9] but typically introduce additional overhead due to memory cell connectivity and adder tree complexity [7]. Furthermore, transformer models often exceed on-chip memory capacity, necessitating frequent off-chip access.

This work significantly enhances energy efficiency and area efficiency without sacrificing accuracy through a two-pronged approach. First, it employs ternary weight splitting (TWS) binarization on BERT models, achieving a 32× compression rate compared to FP32 models while minimizing quantization error. Second, we introduce a 2KB digital-domain CIM accelerator capable of performing batch-wise Brain-Floating-Point-16xINT1 (BF16×1-b) for activation×weight matrix multiplications for better accuracy. The CIM accelerator features BF16×1-b processing elements (PEs) with 2-to-1 multiplexed adder trees, group-vector systolic PE groups, standardized data frames, as illustrated in Fig. 1. Our contributions can be summarized as:

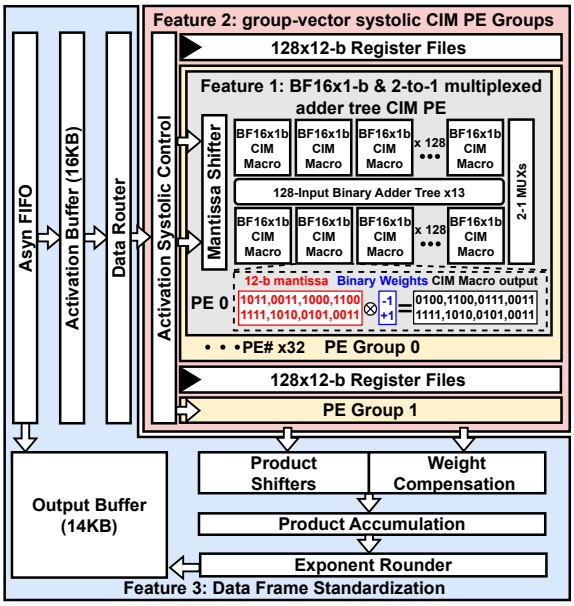

Fig. 1. Proposed CIM accelerator with BF16x1-b CIM PE, group vector systolic CIM PE groups and data frame standardization for efficient inference of binarized BF16x1-b BERTs.

- We employ TWS to binarize BERT models into a BF16×1-b data format, achieving higher accuracy compared to INT4/INT8×1-b binarization at the same compression rate.
- We propose a BF16×1-b SRAM-based CIM macro to support floating-point operations with high parallelism. The proposed batch-wise mantissa shifter in CIM PEs and exponent rounder in CIM PE groups reduce computational error by two orders of magnitude while incurring only around 5% area and power overhead.

In this work, we further integrate a group vector systolic array with the CIM architecture, striking a balance between

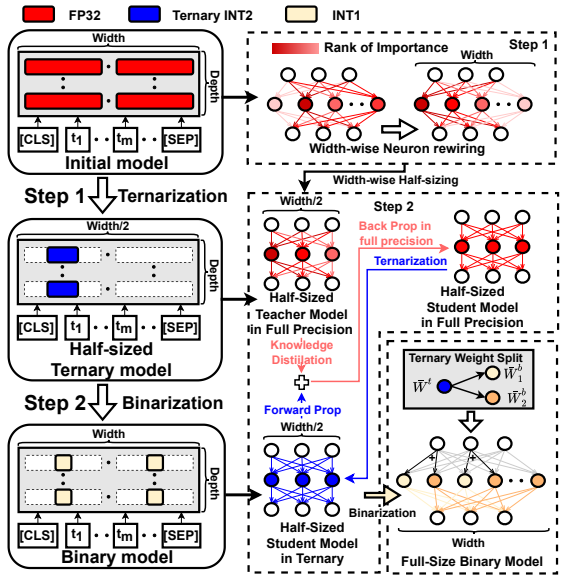

Fig. 2. Overview of the two-step binarization flow of BF16×1-b BERT model.

throughput and register dynamic power. A data frame standardization strategy is employed to eliminate the overhead associated with tensor reshaping and facilitate systolic data flow. The proposed design is verified with post-layout results in 28nm technology. The results demonstrate a 10.25× improvement in area efficiency and a 2.23× improvement in energy efficiency compared to state-of-the-art counterparts.

II. BERT Binarization

While transformer models like BERT achieve exceptional performance [10], their high computational and memory requirements pose challenges for edge devices. Binarization reduces model size and computational complexity by representing weights and activations with fewer bits. However, directly binarizing transformer models often leads to significant accuracy degradation due to the sensitivity of attention mechanisms to quantization.

To address this, we propose a two-step binarization process illustrated in Fig. 2 to decompose ternary weights into binary components while preserving performance. To further mitigate accuracy loss, we employ Knowledge Distillation with a full-precision teacher model. The overall process is summarized in Algorithm 1.

A. Ternary Quantization

We first ternarize the weights to reduce computational load while retaining performance. For each layer l, we compute a threshold $\Delta^{(l)}$ scaled by a factor t as:

$$\Delta^{(l)} = t \cdot \mathbb{E}[|W^{(l)}|] \tag{1}$$

We then quantize the weights to $-1, 0, +1$ using:

$$\hat{W}_i^{(l)} = \begin{cases} +1, & \text{if } W_i^{(l)} > \Delta^{(l)} \\ 0, & \text{if } |W_i^{(l)}| \le \Delta^{(l)} \\ -1, & \text{if } W_i^{(l)} < -\Delta^{(l)} \end{cases} \tag{2}$$

Algorithm 1 Pseudo-code of the two-step binarization process for BERT models.

Input: Pre-trained transformer model weights W, input activation X, learning rate η.

Initialize scaling factors $\{\alpha^{(l)}\}$ for each layer l.
Step 1: Ternarization and Knowledge Distillation
for each layer l **do**
  Compute threshold $\Delta^{(l)}$ using Eq. (1).
  Quantize weights to ternary values $\hat{W}^{(l)}$ using Eq. (2) and quantize activation X to BF16.
  Compute layer output using $\hat{W}^{(l)}$ and $\hat{X}$.
end for
Compute loss $\mathcal{L}$ with knowledge distillation using Eq. (6).
Compute gradients $\nabla_W \mathcal{L}$.
for each layer l **do**
  Update weights: $W^{(l)} \leftarrow W^{(l)} - \eta \nabla_{W^{(l)}} \mathcal{L}$.
end for
Step 2: Binarization and Fine-tuning
for each layer l **do**
  Convert ternary weights $\hat{W}^{(l)}$ to binary weights using TWS.
  Generate $W_i^{(b1)}$ and $W_i^{(b2)}$ using Eq. (4) and Eq. (5)
end for
Fine-tune the binary model with knowledge distillation.
Output: Binarized transformer model with weights $\{\tilde{W}^{(l)}\}$.

This ternary model serves as the foundation for weight splitting and knowledge distillation.

B. Ternary Weight Splitting

The core idea of TWS is to approximate full-precision weights using a combination of binary bases and scaling factors. Given a pre-trained ternary weight matrix W^t and its quantized counterpart $\hat{W}^t$, each ternary weight matrix W^t is decomposed into two binary matrices $W^{(b1)}$ and $W^{(b2)}$:

$$W^t = W^{(b1)} + W^{(b2)}, \quad \hat{W}^t = \hat{W}^{(b1)} + \hat{W}^{(b2)} \tag{3}$$

To satisfy this condition, we define the binary weight components as:

$$W_i^{(b1)} = \begin{cases} a \cdot W_i^t, & \text{if } \hat{W}_i^t \neq 0 \\ b + W_i^t, & \text{if } \hat{W}_i^t = 0,\ W_i^t > 0 \\ b, & \text{otherwise} \end{cases} \tag{4}$$

$$W_i^{(b2)} = \begin{cases} (1-a) \cdot W_i^t, & \text{if } \hat{W}_i^t \neq 0 \\ b, & \text{if } \hat{W}_i^t = 0,\ W_i^t > 0 \\ b + W_i^t, & \text{otherwise} \end{cases} \tag{5}$$

Here, a and b are variables chosen to minimize reconstruction error.

C. Knowledge Distillation and Fine-tuning

To mitigate accuracy degradation, we use knowledge distillation with a full-precision teacher model. The distillation process involves:

- *Intermediate-layer Distillation*: Minimizing the mean squared error between the student's and teacher's embeddings (E vs. $\hat{E}$), multi-head attention outputs ($M^{(l)}$ vs. $\hat{M}^{(l)}$), and feed-forward outputs ($F^{(l)}$ vs. $\hat{F}^{(l)}$)

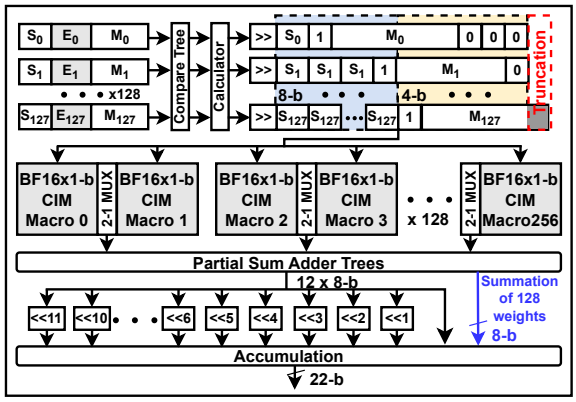

Fig. 3. Proposed BF16x1-b CIM PE with 2-to-1 multiplexed adders.

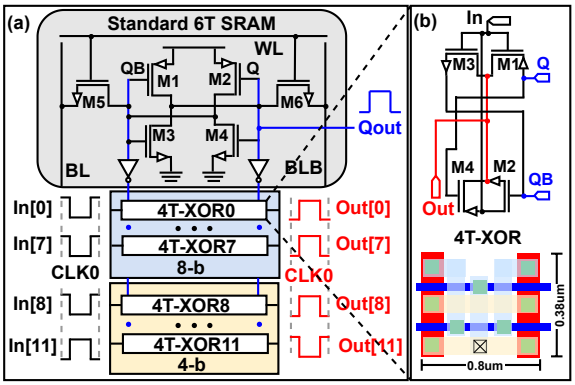

Fig. 4. (a) Proposed BF16x1-b CIM macro. (b) Proposed 4T-XOR module.

- *Prediction-layer Distillation*: Minimizing the soft cross-entropy between the student's logits $\hat{y}$ and the teacher's logits y

The total loss function combines the intermediate and prediction-layer distillation losses:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{intermediate}} + \mathcal{L}_{\text{prediction}} \tag{6}$$

D. Activation Quantization

To further reduce computational complexity without significant accuracy loss, we quantize activations using BF16, which retains the 8-bit exponent of standard 32-bit floats but reduces the mantissa to 7 bits, preserving dynamic range while decreasing memory usage and computational overhead. BF16 is supported by modern accelerators, allowing efficient computation and storage, especially when combined with knowledge distillation.

E. Binarization Algorithm Flow

The complete binarization process is summarized in Algorithm 1, which consists of two main steps:

- **Step 1**: Perform ternarization of weights using threshold $\Delta^{(l)}$ and quantize activations to BF16. Train the model with knowledge distillation from a full-precision teacher.
- **Step 2**: Convert ternary weights to binary weights using TWS, generating binary components $W^{(b1)}$ and $W^{(b2)}$. Fine-tune the binary model with knowledge distillation to recover any performance loss.

The result is a binarized transformer model $\tilde{W}^{(l)}$ optimized for efficient inference on edge devices with limited computational resources.

III. BF16x1-b CIM PE

As shown in Fig. 3, the proposed CIM PE contains 256 BF16×1-b CIM macros, organized in two rows for enhanced throughput. To reduce summation and multiplexing cost, 128 2-to-1 MUXs and 13 128-b adder trees are included. A batch-wise mantissa shifter with an extra 4-b shifting space is integrated to enable batch-wise exponent pre-alignment and improve computational accuracy.

The key innovation in our PE design lies in the synergistic combination of bit-parallel computation and efficient data movement. The 256 BF16×1-b CIM macros are carefully organized to maximize throughput while minimizing routing complexity. The 2-to-1 multiplexed adder tree structure represents an optimal trade-off between computational capability and hardware overhead, enabling efficient processing of both dense and sparse operations common in transformer architectures.

A. BF16x1-b CIM Macro

The proposed CIM macro incorporates a standard 6T SRAM for unsigned 1-bit weight storage and 12 4T-XORs for shifted mantissa with signed, as shown in Fig. 4(a), forming a 58T SRAM-based cell. By sharing 1-bit weights and duplication of XOR units, bit-parallel computation paradigm reduces the cycle count from 12 to 1 compared to bit-serial methods. As shown in Fig. 4(b), the proposed 4T-XOR requires fewer peripherals than traditional 4T-XORs that use two transmission gates [7], [8]. Compared to [7], this macro achieves a 12x throughput improvement with only a 8.38x area overhead. The first eight 4T-XORs compose the original computing space, while the final four 4T-XORs implement the mantissa shifting space for BF16 activations, reducing the computing error by 2-order of magnitude.

B. Batch-wise Mantissa Shifter

Computation of floating point operations imposes additional challenges to CIM architecture. First, floating point operation on CIM necessitates exponent alignment and corresponding mantissa shifting [11], requiring additional comparison tree and shifters. Moreover, a coarse-grain layer-wise shifting of mantissa [8] would truncate excessive bit information, as shown in Fig. 5(a) left, thus impairing system accuracy.

To address these challenges, this work employs exponent pre-alignment and batch-wise mantissa shifter to conduct MAC operation for FP-based data format in CIM architecture. As shown in Fig. 5(b), proposed CIM PE finds the largest exponent E_{max0} among the first batch of 128 activations. It then calculates exponent difference $E_{max0} - E_i$ and shifts mantissa M_i accordingly. The proposed CIM PE provides fine-grain batch-wise shifting and saves more bits compared

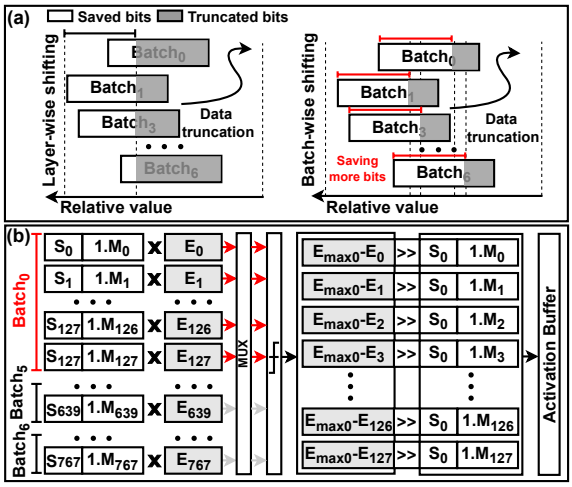

Fig. 5. (a) Comparison between layer-wise shifting and batch-wise shifting. (b) Proposed batch-wise mantissa shifter design for floating-point operation.

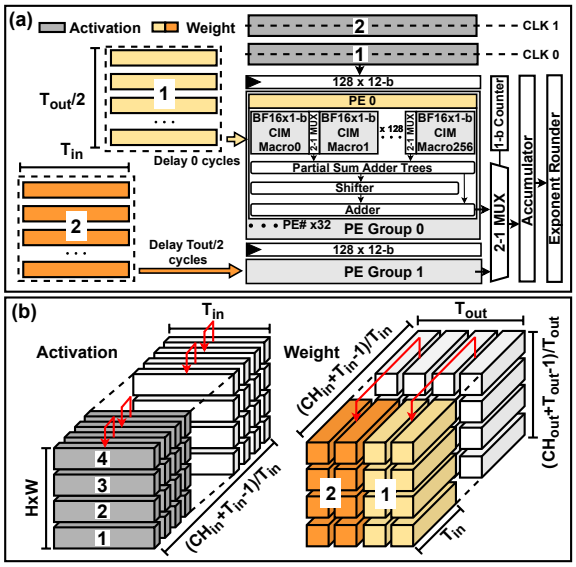

Fig. 6. (a) Neural network mapping of proposed group vector systolic CIM array. (b) Proposed weight and activation data frame standardization.

to conventional layer-wise shifting. The obtained E_{max} of different batches is stored in activation buffer along with the shifted mantissa for subsequent batch-wise exponent alignment in exponent rounder to generate final results.

IV. CIM Accelerator Architecture

As shown in Fig. 6, the proposed digital CIM accelerator has a 2 KB (128x128) storage capacity, matching the attention layer size of BERT-Tiny. It comprises two PE groups, each containing 32 PEs. Additional peripherals, such as exponent rounders, are included to enhance BF16 accuracy. A systolic controller manages the group vector systolic data flow, optimizing power consumption.

A. Group Vector Systolic CIM Array

Previous CIM works [3], [7] often employ broadcast computing diagrams to handle convolution or matrix multiplication, suffering from large fan-out and poor timing. To boost throughput, CIM-based implementation [12] adopted fine-grained, atomic systolic strategies to increase system frequency. However, this approach necessitates substantial register files for intermediate data storage, leading to significant dynamic power and area overhead.

To mitigate this overhead, vector systolic arrays were proposed in [13], elevating systolic operations from the macro to the PE level, significantly reducing register files. Building upon this, our work deploys a group vector systolic design [14], elevating the data systolic level to the PE-group level in the CIM architecture, as shown in Fig. 6, further reducing register files.

Fig. 6(a) illustrates the implementation of the proposed group vector systolic CIM array. The PE groups store 128x128 binary weights and process 128 12-b activations per cycle. A 1-bit counter multiplexes outputs from two PE groups in a systolic manner. To reduce latency and area overhead, activations are broadcasted within PE groups while weights remain static. An exponent rounder aligns batch-wise exponent and truncates the accumulator results to BF16 format. The proposed design requires only 2 cycles for weight updating, significantly outperforming traditional atomic and vector systolic CIM arrays, which require $n+m-1$ and $n-1$ cycles, respectively, for an $m \times n$ matrix multiplication.

B. Data Frame Standardization

Matrix operations in transformers have various tensor shapes, e.g., Matrix-multiplication, LayerNorm, Multi-head Self-Attention. General processing of these tensors requires extra temporary buffer for reshaping. This work proposes a standardized data frame for all kinds of tensors in transformer and facilitates group vector systolic. The fundamental principle combines adjacent output channels ($T_{in} = 128, T_{out} = 128$), as shown in Fig. 6(b). Specifically, the standardized data frame is a four-dimensional tensor, e.g., $((CH_{in} + T_{in} - 1)/T_{in}, H, W, T_{in})$. For low dimensional tensor like LayerNorm in Bert-Tiny with shape $(CH_{in}/T_{in}, Emb, T_{in}) = (1, 128, 128)$, the Emb is divided into H and W. For high dimensional tensor like multi-head self-attention (MHSA), the tensor shape is reorganized as $((CH_{in}+T_{in}-1)/T_{in}, Head \times H, W, T_{in})$.

The benefits of data frame standardization are evident: as the outputs of all operations are equivalent to their inputs, the subsequent tensor alignment procedures commonly seen in other CIM implementations [8] can be omitted.

The proposed architecture demonstrates strong scalability characteristics. The group vector systolic design can be readily extended to accommodate larger model sizes by increasing the number of PE groups, while the standardized data frame structure ensures efficient handling of varied tensor operations regardless of scale. This flexibility makes our approach particularly suitable for future transformer variants with different attention mechanisms and model architectures.

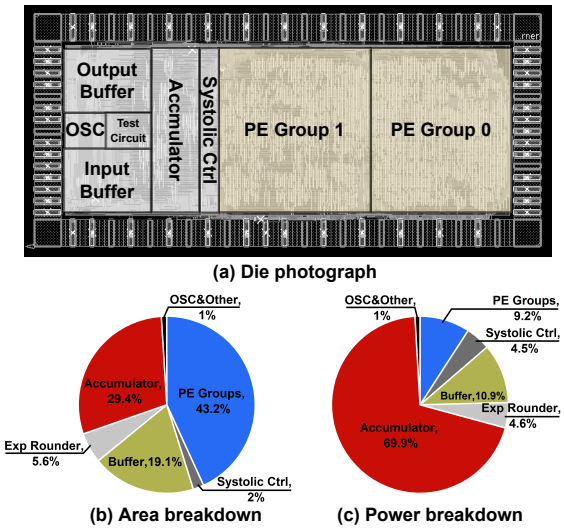

Fig. 7. (a) Die photograph of proposed CIM accelerator. (b) Area breakdown of proposed CIM accelerator. (c) Power breakdown of proposed CIM accelerator.

C. Experiment Setup

The proposed CIM macro was designed and characterized using Cadence Virtuoso and Liberate MX. A latch-like SRAM RTL behavioral model was employed as a software baseline benchmark. The remaining accelerator components were implemented in RTL Verilog HDL, leveraging Memory Compiler-generated SRAM. Accuracy performance was evaluated using Synopsys PrimeSim AMS, incorporating SDF/DSPF RC file back-annotation. Area and power consumption were assessed through place-and-route in Cadence Innovus and timing analysis in Synopsys PrimeTime, respectively, targeting a 28nm technology node. The clock frequency was set to 400 MHz, resulting in a total post-layout area of 2 mm², as illustrated in Fig. 7(a).

To avoid unfair comparison, the counterparts' energy efficiency is normalized to 28nm technology under 0.65V supply voltage. The scaling methodology is deployed from [15]. CIM implementation [8] with 8-b computing space and layer-wise mantissa shifter is designed in accuracy comparison. Two weight binarized BERT models, e.g., BERT-Base and BERT-Tiny with four different datasets, e.g., SQuAD 1.1/2.0, STT-2 and Natural Questions (NQ) are used as the benchmarks for evaluation.

D. System Performance Evaluation

We evaluate the performance focusing on three aspects: BERT Binarization Evaluation, Binarized BERT Performance on CIM Accelerator, and System Comparison with existing designs.

The comprehensive evaluation demonstrates our design's advantages across multiple dimensions. In terms of model compression, our TWS binarization achieves 32× reduction in model size while maintaining competitive accuracy. This is particularly significant for edge deployment, where both memory footprint and computational efficiency are critical

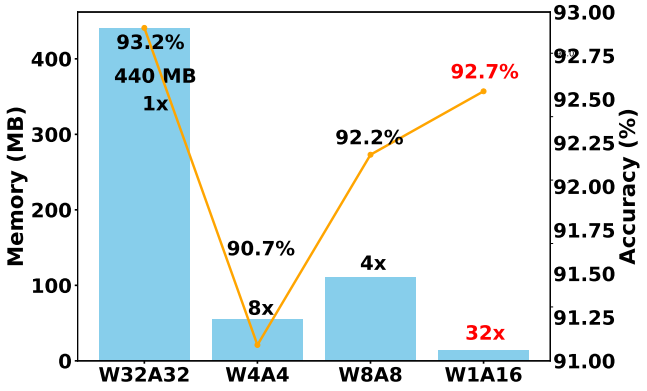

Fig. 8. Comparison of compression ratio and accuracy for various quantization configurations on the SST-2 Dataset with [16].

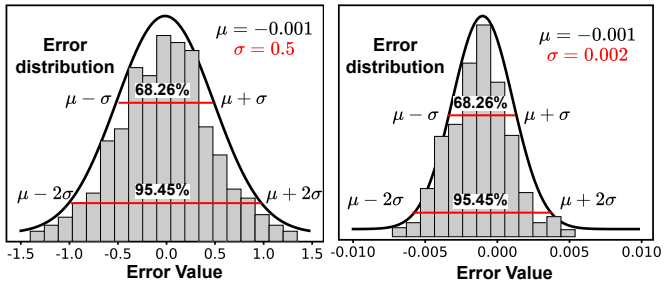

Fig. 9. Floating point computational accuracy comparison between conventional CIM implementation and proposed CIM accelerator. (a) Computational result of conventional CIM with layer-wise shifting. (b) Computational result of proposed CIM accelerator with batch-wise shifting.

constraints. The batch-wise processing strategy, combined with our group vector systolic architecture, enables efficient handling of different transformer operations while minimizing data movement overhead.

1) BERT Binarization Evaluation: From Table I, our model outperforms prior designs across all datasets. On SQuAD 1.1, our BF16x1 binarized BERT achieves 87.5% accuracy, surpassing VLSI'22 with 8-bit precision (87%). On SQuAD 2.0, our design achieves 88.3%, while JSSC'24's 16-bit BERT only reaches 45.73%. Our proposed configuration achieves a 32x compression ratio while maintaining 92.7% accuracy in SST-2 as shown in Fig. 8, outperforming the AxW=FP4/8 configuration in [16].

V. Experiment Results

1) Binarized BERT Performance on CIM Accelerator: The area and power breakdown are depicted in Fig. 7(b) and (c). The systolic PE groups and Accumulator constitute 43.24% and 29.4% of area, respectively. The Accumulator and Buffer occupy 69.9% and 10.9% of power consumption. The systolic control module takes only 2% in both area and power, minimizing data transaction power overhead. The exponent rounder reduces error rate by 2-order of magnitude compared to conventional CIM implementation, as shown in Fig. 9, while requiring only 5.6% area and 4.6% power overhead.

TABLE I
MEASUREMENT RESULT AND COMPARISON TABLE

Design	VLSI'22 [17]	ISSCC'23 [16]	ISSCC'22 [18]	JSSC'24 [19]	This work			
Technology	5nm	12nm	28nm	28nm	28nm			
Implementation	N.A.	N.A.	Digital CIM	Digital CIM	Digital CIM			
Precision (Bits)	AxW=4/8	AxW=FP4/8	AxW=8/16-b	AxW=8/16-b	AxW=BF16x1-b			
CIM Size (KB)	N.A.	N.A.	24	24	2			
On-chip SRAM (KB)	141	647	192	192	36			
Die Area (mm2)	0.153	4.6	6.83	6.98	1.995			
Supply Voltage (V)	0.46∼1.05	0.62∼1.0	0.6-1.0	0.6-1.0	0.65-1.0			
Frequency (Mhz)	152∼1760	77∼717	80∼240	80∼275	60∼400			
Power (mW)	N.A.	9∼122	27.04∼118.21	20.4∼119.7	62.96∼419.74			
Throughput(TOPS)	3.6(4-b) 1.8(8-b)	0.734(FP-4) 0.367(FP-8)	1.48(8-b) 0.37(16-b)	2.24(8-b) 0.56(16-b)	6.55			
Area Efficiency (TOPS/mm2)	23.3(4-b) 11.7(8-b)	0.16(FP-4) 0.08(FP-8)	0.22(8-b) 0.05(16-b)	0.32(8-b) 0.08(16-b)	3.3			
Model structure	BERT-Base	BERT-Base	BERT-Base	BERT-Tiny	BinaryBERT (BERT-Base)			BinaryBERT (BERT-Tiny)
Dataset	SQuAD 1.1	SST-2	NQ	SQuAD 2.0	SST-2	SQuAD 1.1/2.0	NQ	SQuAD 2.0
Accuracy Score(%)	87(4-b) 87.5(8-b)	90.7(FP-4) 92.2(FP-8)	54.4 (16-b)	48.94 (16-b)	92.7	87.9/76.9	56.4	45.73
Reported Energy Efficiency (TOPS/W)	95.6(4-b) 39.1(8-b)	18.1(FP-4) 8.24(FP-8)	20.5(8-b) 5.1(16-b)	15.71(16-b)	17.84	19.31/18.01	16.7	20.98
Normalized Energy Efficiency[1] (TOPS/W)	8.55 (4-b) 3.5 (8-b)	7.76 (FP-4) 3.53 (FP-8)	15.28	8.61	17.84	19.31/18.01	16.7	20.98

1: Normalized to 28nm at 0.65V operational power (Reported EE×(Node/28nm)×(Voltage^2/0.65V^2))

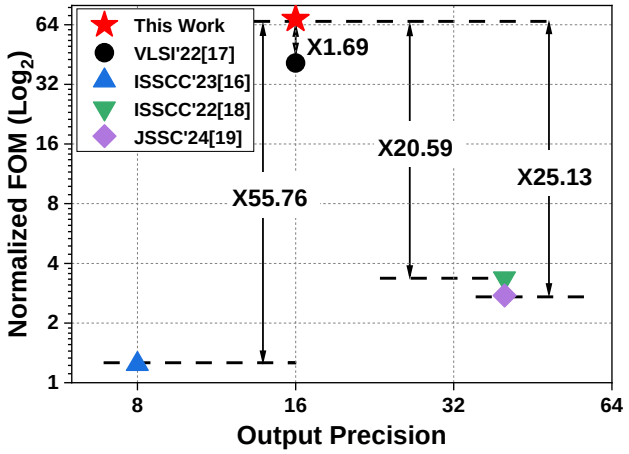

Fig. 10. FoM comparison between this work and digital CIM implementations for BERT [16]–[19] (FoM=Normalized Energy efficiency × Area efficiency)

2) System Comparison: The significant improvements in both area and energy efficiency stem from three key architectural decisions: (1) the bit-parallel computation paradigm that reduces cycle count and simplifies control logic, (2) the group vector systolic architecture that optimizes data movement and reduces register file requirements, and (3) the batch-wise mantissa shifting strategy that enables efficient floating-point operations with minimal overhead. These innovations work together to achieve superior performance metrics while maintaining high accuracy across different BERT variants and datasets. Our design's superior power efficiency is particularly evident in the normalized metrics, where we achieve 20.98 TOPS/W despite operating at a higher frequency than comparable designs. The power overhead from the batch-wise mantissa shifter and exponent rounder (4.6%) is more than compensated by the reduction in data movement and register file requirements, resulting in a net improvement in energy efficiency.

As shown in Fig. 10, our proposed accelerator surpasses state-of-the-art designs [16]–[19] in Figure of Merit (FoM), combining normalized energy efficiency and area efficiency. This improvement stems from minimized transaction power enabled by our group vector systolic CIM array and enhanced throughput through BF16×1-b macros. Table I provides a detailed comparison with state-of-the-art implementations.

VI. CONCLUSION

This work introduces a weight-binarized BERT model that can be highly efficiently mapped on a group vector systolic CIM accelerator. We employ knowledge distillation and TWS to obtain BF16x1-b BERT models, achieving a 0.5% to 2% accuracy improvement and a 4x to 8x compression ratio compared to INT4/8 quantization. To enhance efficiency, we propose high parallelism BF16x1-b CIM macro and group vector systolic CIM PE groups. Additionally, a batch-wise mantissa shifter and exponent rounder are introduced to boost computational accuracy. Post-layout results demonstrate that the proposed CIM architecture achieves an area efficiency of 3.3 TOPS/mm² and an energy efficiency of 20.98 TOPS/W for BERT-Tiny. Compared to state-of-the-art designs, our design achieves a 10.25x to 20.62x improvement in area efficiency and a 1.1x to 2.23x improvement in energy efficiency.

REFERENCES

[1] Z. Guan, H. Huang, Y. Su, H. Huang, N. Wong, and H. Yu, "APTQ: Attention-aware post-training mixed-precision quantization for large language models," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*, 2024, pp. 1–6.

[2] X. Dong, S. Chen, and S. Pan, "Learning to prune deep neural networks via layer-wise optimal brain surgeon," *Advances in neural information processing systems*, vol. 30, 2017.

[3] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 14–26, 2016.

[4] D. Liu, H. Zhou, W. Mao, J. Liu, Y. Han, C. Man, Q. Wu, Z. Guo, M. Huang, S. Luo, M. Lv, Q. Chen, and H. Yu, "An energy-efficient mixed-bit cnn accelerator with column parallel readout for reram-based in-memory computing," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 12, no. 4, pp. 821–834, 2022.

[5] Q. Zheng, Z. Li, J. Ku, Y. Wang, B. Taylor, D. Fan, and Y. Chen, "Improving the efficiency of in-memory-computing macro with a hybrid analog-digital computing mode for lossless neural network inference," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*, 2024, pp. 1–6.

[6] Y.-C. Lo and R.-S. Liu, "Morphable cim: Improving operation intensity and depthwise capability for sram-cim architecture," in *2023 60th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2023, pp. 1–6.

[7] Y.-D. Chih, P.-H. Lee, H. Fujiwara, Y.-C. Shih, C.-F. Lee, R. Naous, Y.-L. Chen, C.-P. Lo, C.-H. Lu, H. Mori, W.-C. Zhao, D. Sun, M. E. Sinangil, Y.-H. Chen, T.-L. Chou, K. Akarvardar, H.-J. Liao, Y. Wang, M.-F. Chang, and T.-Y. J. Chang, "An 89tops/w and 16.3 tops/mm2 all-digital sram-based full-precision compute-in memory macro in 22nm for machine-learning edge applications," in *Proc. IEEE Int. Solid-State Circuits Conf.(ISSCC)*, vol. 64, 2021, pp. 252–254.

[8] F. Tu, Y. Wang, Z. Wu, L. Liang, Y. Ding, B. Kim, L. Liu, S. Wei, Y. Xie, and S. Yin, "Redcim: Reconfigurable digital computing-in-memory processor with unified fp/int pipeline for cloud ai acceleration," *IEEE Journal of Solid-State Circuits*, vol. 58, no. 1, pp. 243–255, 2022.

[9] Z. Chen, Y. Ma, K. Li, Y. Jia, G. Li, M. Wu, T. Jia, L. Ye, and R. Huang, "An in-memory computing accelerator with reconfigurable dataflow for multi-scale vision transformer with hybrid topology," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*, 2024, pp. 1–6.

[10] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[11] T.-H. Wen, H.-H. Hsu, W.-S. Khwa, W.-H. Huang, Z.-E. Ke, Y.-H. Chin, H.-J. Wen, Y.-C. Chang, W.-T. Hsu, C.-C. Lo, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, S.-H. Teng, C.-C. Chou, Y.-D. Chih, T.-Y. J. Chang, and M.-F. Chang, "A 22nm 16mb floating-point reram compute-in-memory macro with 31.2 tflops/w for ai edge devices," in *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 67. IEEE, 2024, pp. 580–582.

[12] Z. Dai, S. Yan, Z. Cong, Z. Guo, Y. He, W. Sun, C. Dou, F. Zhang, J. Yue, Y. Liu, and M. Liu, "A 41.7 tops/w@ int8 computing-in-memory processor with zig-zag backbone-systolic cim and block/self-gating cam for nn/recommendation applications," in *2024 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*. IEEE, 2024, pp. 1–2.

[13] M. Huang, Y. Liu, C. Man, K. Li, Q. Cheng, W. Mao, and H. Yu, "A high performance multi-bit-width booth vector systolic accelerator for nas optimized deep learning neural networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 9, pp. 3619–3631, 2022.

[14] M. Huang, J. Luo, C. Ding, Z. Wei, S. Huang, and H. Yu, "An integer-only and group-vector systolic accelerator for efficiently mapping vision transformer on edge," *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2023.

[15] A. Stillmaker and B. Baas, "Scaling equations for the accurate prediction of cmos device performance from 180 nm to 7 nm," *Integration*, vol. 58, pp. 74–81, 2017.

[16] B. Keller, R. Venkatesan, S. Dai, S. G. Tell, B. Zimmer, W. J. Dally, C. T. Gray, and B. Khailany, "A 17–95.6 tops/w deep learning inference accelerator with per-vector scaled 4-bit quantization for transformers in 5nm," in *2022 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*. IEEE, 2022, pp. 16–17.

[17] B. Keller, R. Venkatesan, S. Dai, S. G. Tell, B. Zimmer, W. J. Dally, C. Thomas Gray, and B. Khailany, "A 17–95.6 tops/w deep learning inference accelerator with per-vector scaled 4-bit quantization for transformers in 5nm," in *2022 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, 2022, pp. 16–17.

[18] F. Tu, Z. Wu, Y. Wang, L. Liang, L. Liu, Y. Ding, L. Liu, S. Wei, Y. Xie, and S. Yin, "A 28nm 15.59 μj/token full-digital bitline-transpose cim-based sparse transformer accelerator with pipeline/parallel reconfigurable modes," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65. IEEE, 2022, pp. 466–468.

[19] R. Guo, X. Chen, L. Wang, Y. Wang, H. Sun, J. Wei, H. Han, L. Liu, S. Wei, Y. Hu, and S. Yin, "Cimformer: A systolic cim-array-based transformer accelerator with token-pruning-aware attention reformulating and principal possibility gathering," *IEEE Journal of Solid-State Circuits*, pp. 1–3, 2024.