國立清華大學 電機工程學系

實作專題研究成果摘要

A 65nm 6T-SRAM Ping-Pong Weight Updating Computing-in-Memory Macro with Adaptive Precision ADC for AI edge chips

65 奈米製程同步權重更新及計算與自適性精度類比數位轉換器之靜態隨機存取記憶體內運算架構

專題領域：系統領域

組別：A399

指導教授：張孟凡 教授

組員姓名：李昀達

研究期間：112 年 7 月 1 日至 113 年 4 月 30 日止，共計 10 個月

Abstract

This research presents the development and implementation of a 65nm 6T-SRAM Ping-Pong Weight Updating Computing-in-Memory (CIM) Macro with an Adaptive Precision ADC (Analog-to-Digital Converter), designed for AI edge chips. This architecture overcomes the significant challenges of conventional von Neumann systems, particularly the intensive power and time required for data transfer between memory and processing units by minimizing data movement and enabling efficient parallel multiply-and-accumulate (MAC) operations within the memory macro. The SRAM-based CIM is chosen for its low latency, high throughput, and durability, optimal for AI edge applications.

However, deploying large neural network models on CIM processors presents issues such as insufficient storage and frequent weight update requirements, which degrades performance. Thus, the implemented ping-pong weight updating mechanism, referencing paper [1] and [2], allows for simultaneous CIM operations and dynamic weight adjustments, improving system utilization and reducing loss from continuous memory updates. An adaptive bitline header from paper [3], paired with an adaptive-precision ADC using sense amplifiers and priority encoders balances between accuracy and efficiency during operations.

Experimental results demonstrate the efficacy of the ping-pong CIM architecture. The system's ability to adapt between accumulation-8 (acc8) and accumulation-16 (acc16) modes optimizes for different operational demands, as shown through precise digital conversion of MAC values into 4-bit outputs signals with zero-bit errors. Power consumption analysis reveals that switching to acc8 mode significantly reduces energy use by 48.71%, illustrating the system's capability to efficiently manage power depending on input distribution. Moreover, the design achieves a 20~50% reduction in total operation cycles through the ping-pong CIM architecture, enhancing operation time savings compared to conventional CIM structures.

The implementation of this innovative 65nm 6T-SRAM Ping-Pong Weight Updating CIM Macro with Adaptive Precision ADC significantly advances the performance and functionality of AI edge devices, addressing key limitations of traditional architectures while setting new benchmarks for the deployment of complex neural network models in edge computation. This system combines robust performance with high throughput and reduced power consumption, showcasing potential for further advancements in CIM technology.

摘要

隨著深度學習的依賴增加，由於神經網絡（NN）應用如圖像分類和語音識別的擴展，傳統電路架構下數據在記憶體與 CPU 來回傳輸所需的時間和能源成為關鍵挑戰。記憶體內運算（Computing-In-Memory）架構通過最小化數據移動並直接在記憶體內啟用平行乘加（MAC）操作來解決這一瓶頸，從而提高處理速度和能源效率，適合於記憶體要求高和運算量大的操作。

然而，在 CIM 處理器上部署大型神經網絡模型存在如存儲空間不足與需要頻繁更新權重等問題，導致性能降低。為解決這些問題，我參考文獻[1]和[2]的同步權重更新及計算機制，結合參考文獻[3]的適應性乘加累計值（MAC value）架構，提出了一種針對 AI 邊緣晶片設計的 65 奈米 6T-SRAM 同步權重更新及計算記憶體內運算（CIM）架構與自適性精度類比數位轉換器（ADC）。通過同步權重更新及計算，此記憶體支持 CIM 操作和動態權重調整同時進行，提高系統利用率，減少持續記憶體更新導致的性能損失。結合自適性位元線（bitline header）用於 CIM 累加，與適應性精度類比數位轉換器（ADC）配對，平衡精度性能之間的折衷。此研究專注於基於 SRAM 的 CIM，因其低延遲、低能耗及耐用性，適合應用於人工智慧邊緣運算裝置上。

實驗結果證明了同步權重更新及計算 CIM 架構的有效性。系統能夠在 accumulation-8（acc8）模式和 accumulation-16（acc16）模式之間適應不同的操作需求，通過精確的 MAC 值無誤的轉換到信號線輸出端的 4 位數表示。基於實驗結果功耗分析顯示，在新架構下切換到 acc8 模式可顯著降低能耗達 48.71％，並保持 3 位數的輸出，說明系統能根據輸入分佈有效減少能耗。此外，通過同步權重更新及計算 CIM 架構，設計在總運行周期上實現了 20~50％的降低，較傳統 CIM 結構提高了運行時間節省。

此專題的 65 奈米 6T-SRAM 同步權重更新及計算 CIM 架構與自適性精度 ADC，結合了強大時間效率、精準度和低功耗，提高了 AI 邊緣設備的性能和功能，解決了傳統架構的關鍵限制，為複雜神經網絡模型在邊緣計算中的部署設定了新的標準。

1. Motivation and Purpose

Reliance on deep learning has grown significantly due to expanding neural network (NN) applications like image classification and speech recognition. As AI edge devices depend on on-device training and large datasets, the time and power required for data transfer under the conventional von Neumann architecture present critical challenges. The Computing-in-Memory (CIM) architecture addresses the memory bottleneck by minimizing data movement and enabling parallel multiply-and-accumulate (MAC) operations directly within a memory macro, thus enhancing processing speed and energy efficiency for handling memory-demanding and computation-heavy operations. Compared to other CIM schemes based on nonvolatile memory, this research specifically focuses on SRAM-based CIM for its low latency, high throughput, and durability, suitable for AI edge applications.

However, challenges in deploying large NN models on CIM processors include insufficient storage in standard CIM macros (10kb – 10Mb) for larger NN models (10Mb – 10Gb), require frequent weight updates that degrade performance. Additionally, voltage dividing CIM accumulation architectures forces a tradeoff between signal linearity and margin, with variations in bitline-header resistance affecting MAC values' linearity and margin. Resolving this problem is essential as NN datasets require strict precision for reducing accuracy loss.

By implementing the ping-pong CIM scheme in [1], the memory macro supports simultaneous CIM and write operations to improve system utilization rate, reducing performance loss from continuous memory updates. An adaptive bitline header for CIM accumulation is paired with and adaptive-precision analog-to-digital converter (ADC) to balance tradeoff between accuracy and performance. These optimizations lead to a reduction in power and an increase in precision, achieving higher throughput compared to conventional CIM structure.

2. Research Methodology

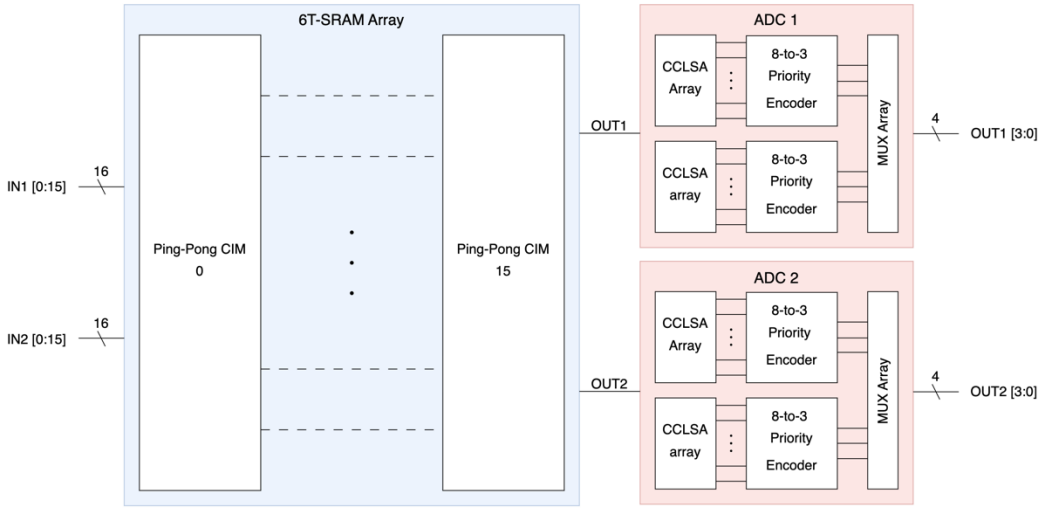

Fig. 1. Overall architecture of ping-pong CIM with adaptive precision ADC.

Fig. 1 presents the overall architecture of this design. It contains a 6T-SRAM array, containing 16 ping-pong CIM cells. Two 4-bit ADCs, each containing seventeen current-controlled latch sense amplifier, two 8-to-3 priority encoders and a MUX array, for supporting two different accumulation and ADC modes: acc 8 mode and acc16 mode under an adaptive control signal.

The overall workflow is described as follows. The weight data is written and stored into the CIM macros and MAC operations are executed in weight-stationary mode. In each CIM macro, the ping-pong bitwise memory units (BMUs) alternate between executing CIM operations and updating weights, achieving continuous utilization of the local computing cells (LCCs) and analog-to-digital converters (ADCs).

To accomplish adaptive precision, when the number of 1's at inputs exceeds 8, an adaptive control signal is sent to the CIM processor to indicate a switch to acc16 mode and shifts to acc8 mode when the number of 1's sits below 8. In acc8 mode, at most eight inputs equaling to 1 are fed into the CIM macro, resulting in an accumulation value no greater than 8. The adaptive control signal switches both the CIM macro and ADC into acc8 mode, increasing the signal margin between neighboring MAC values, leading to higher computing accuracy, while only using half of the ADC for power saving. In acc16 mode, since the MAC value ranges from 0 to 16, the adaptive control switches the CIM macro and ADC into acc16 mode, enabling full range sensing for ADC and an evenly distributed MAC value from the CIM macro.

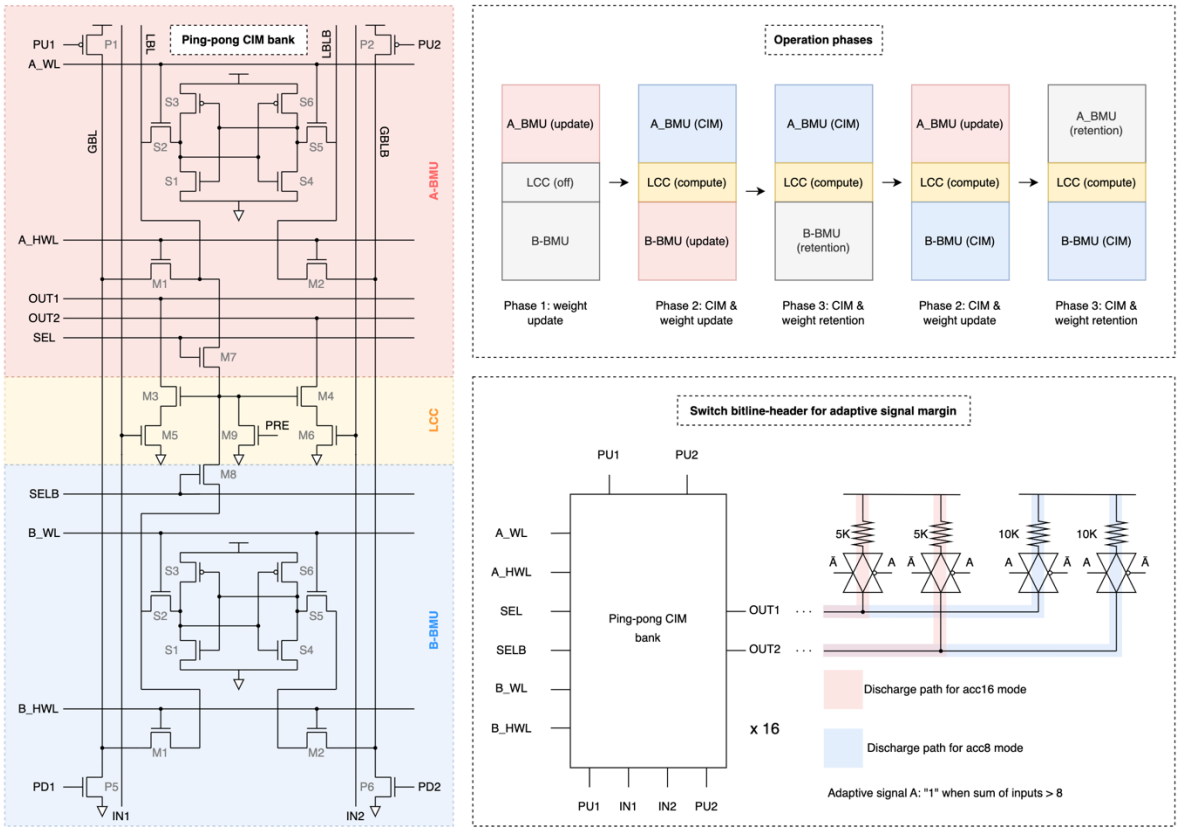

Fig. 2. Ping-pong CIM circuit and operation phases

Fig. 2 presents the ping-pong mechanism that supports simultaneous weight update and CIM operations for full-time utilization for LCCs and ADCs as proposed in [1]. The CIM memory is split into two homogeneous regions (A-BMU and B-BMU). These two regions are comprised of the same CIM cell architecture. Signals SEL and SELB control the connection of BMUs to the LCC. As one BMU is running CIM operation, the other BMU unit updates its weight data for the next cycle of CIM operations. The typical operations could be split into three phases with reference to [2]. In phase 1, weight data are written into A-BMU while B-BMU stays in retention mode. In phase 2, A-BMU executes CIM operations with the weight data from phase 1, B-BMU writes and stores the next section of weight data. In phase 3, if the CIM operation is still running in A-BMU, B-BMU goes into weight retention and waits for the next cycle for CIM operation. Phase 2 and phase 3 iterates through the entire computing cycle, alternating between A-BMU and B-BMU for CIM operations to complete all MAC computation.

Implementation of the voltage dividing method is used for a 16-cell CIM row accumulation in this project, thus the output for all CIM cells is connected (all cells share the same OUT1 and OUT2) with accumulation ranging from 0 to 16. When multiplication results are 1 in each CIM cell, the pull-down path from opening transistors (M3 and M5 or M4 and M6 in Fig. 2) contains an internal resistance R_0, which is used for voltage dividing with the bitline-header resistance R connected to VDD. The total resistance from all ping-pong CIM cells is inversely proportional to the number of multiplication results equaling to 1. Thus, the voltage on output node OUT1 and OUT2 would follow the formula:

$$V_{out} = \frac{R_0 / num\ of\ 1's}{R_0 / num\ of\ 1's \times R} \times V_{DD} = \frac{R_0}{R_0 + (num\ of\ 1's) \times R} \times V_{DD}$$

Following the formula above, we can see that when the accumulation result is 0, we can observe an output voltage of VDD. For output value 16, the voltage reaches a low value.

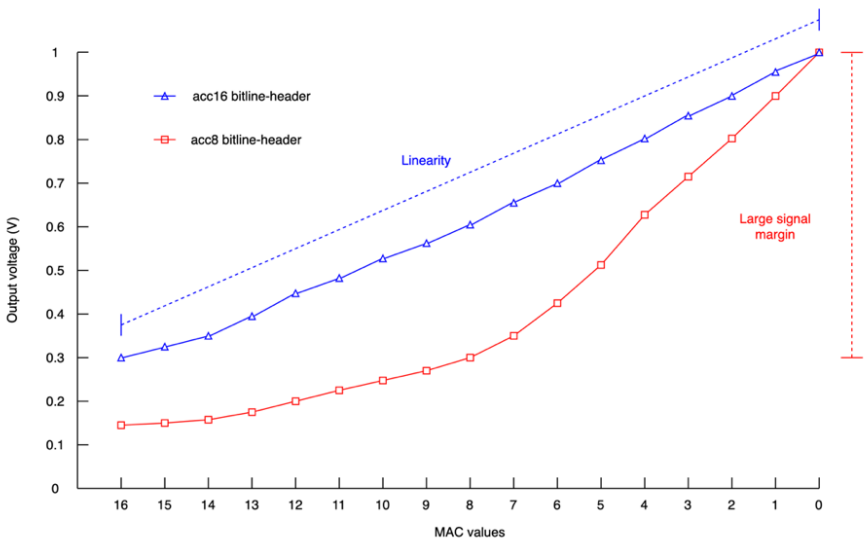

Fig. 3. MAC value comparison between acc8 and acc16 mode.

To preserve the advantages under different bitline-header resistances, I proposed an adaptive control scheme as seen in Fig. 2, which switches between acc8 and acc16 modes according to input values. In acc8 mode, the number of 1's at 16 cells is no larger than 8, the resulting accumulation result is at most 8, thus we use 10k ohms for the bitline-header resistance to increase the signal margin from MAC values 0~8. In acc16 mode, the accumulation results could range from 0~16, thus we use 5k ohms for the bitline-header resistance to split the full range voltage equally to preserve the correct MAC values. Fig. 3 shows the voltage distribution under both modes, showing the tradeoff between linearity and signal margin.

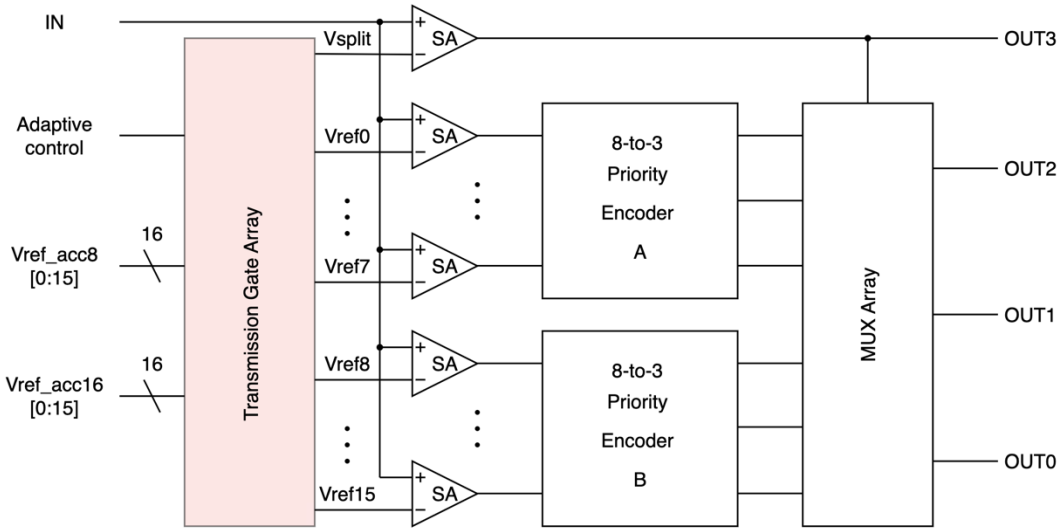

Fig. 4. Adaptive precision ADC structure.

Fig. 4 present the overall structure of the ADC circuit. Compared to the successive approximation register (SAR) ADC presented in [2], this structure achieves the functionality of an ADC in a simpler term, shortening the cycles required for digital conversion, with a tradeoff of area and power. It consists of sixteen n-type and p-type current-controlled latch sense amplifiers from [4] for directly comparing accumulation voltage output with reference voltage, and two 8-to-3 priority encoders for 3-bit digital output logic. A MUX is used to select between the output of the two 8-to-3 priority encoders according to the OUT3 signal from an additional sense amplifier. The sixteen sense amplifiers and priority encoders could be modified by the adaptive control signal to support acc8 and acc16 modes from the CIM macro for 3-bit or 4-bit sensing.

Priority encoder allocates a priority level to each input, the output corresponds to the currently active input which has the highest priority. When an input with a higher priority is present, all other inputs with a lower priority will be ignored. The sense amplifier with the highest reference voltage corresponds to the highest priority bit in the priority encoder since MAC value = 0 is of voltage VDD and all sense amplifiers would output 1, while the largest MAC value is of the lowest voltage with only a 1 on the last sense amplifier. An additional inverter is need at every output node to convert the correct output sequence, as MAC value = 0 feeds 1's into all input nodes of the encoder, it should produce a digital output of 000.

For the ADC to support adaptive precision between 3-bit and 4-bit output for acc8 and acc16 modes, additional transmission gates are used to achieve this functionality as shown in Fig. 4. Under acc8 and acc16 modes, although both supporting reading of MAC values 0~8, due to voltage dividing from different bitline-header resistance, same MAC values have different voltage levels. Transmission gates are utilized to switch the sense amplifier's reference voltage between acc8 and acc16 values for accurate sensing.

Under acc8 mode, priority encoder B dedicated for MAC values 9-16 is turned off for power saving, and the ADC output depends solely on priority encoder A. Under acc16 mode, both priority encoders are turned on generating two 3-bit digital output. A MUX is used to select the desired 3-bit value according to the OUT3 signal from an additional sense amplifier as presented in Fig. 4. This sense amplifier connects to the reference voltage that separates MAC values 0~8 and 9~16. Its output not only acts as the MSB (output bit 3) of the digital CIM operation output, but also the control signal for the MUX for selection of output bits 0~2. Output bit from the single sense amplifier accompanied with the 3-bit output from MUX forms the 4-bit digital 16 cell row MAC operation output.

3. Experimental Results

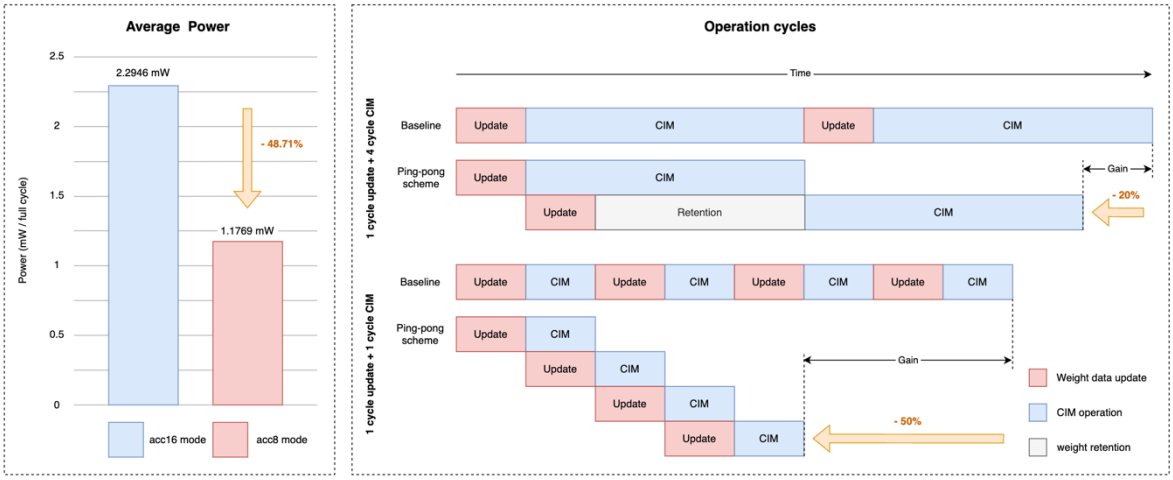

Fig. 5. Power and operation cycle comparison.

In this work, innovative designs contribute to the reduction of computational power, including switching between acc8 and acc16 mode and adaptive precision ADC. By switching to acc8 mode for the ADC, one of the 8-to-3 priority encoders and 8 sense amplifiers could be turned off, reducing power consumption by 48.71% as shown in Fig. 5. Alternative switching between acc8 and acc16 modes would yield an approximate 25% power saving, supposing an even distribution for input data, centering around the midpoint. As cases for insufficient 1's in the input would be detected, the CIM would compute in acc16 mode for half of the operation time, and in acc8 mode for the other. This work shows potential for higher power savings in datasets with input distributions skewed toward the lower end of the range, clustering closer to 0, fully utilizing the power saving characteristics of operating in acc8 mode.

Through implementation of ping-pong CIM design, for each full cycle of CIM operations with 4 cycles of MAC operation per CIM phase and 1 cycle weight updating (A-BMU and B-BMU both completing 2 cycles CIM operations), this structure was able to achieve a reduction in operation time by 20% (total of 8 cycles compared to 10 cycles). For single cycle of MAC per CIM phase, a reduction of 50% could be achieved as presented in Fig. 5. This work shows high potential in datasets with higher weight value variations for efficient updating and operation time savings compared to conventional CIM structure.

The innovative adaptive ADC architecture plays a crucial role in enhancing operational efficiency by significantly reducing the time required for data conversion, as compared to the traditional Successive Approximation Register (SAR) ADCs, albeit with a notable tradeoff in terms of increased silicon area usage. By comparing MAC output signals directly to voltage references simultaneously, digital conversion to 4-bit output could be accomplished by a single priority encoder in a cycle, rather than 4 cycles of single-bit sensing from the sequential approximation process from SAR ADC architectures, streamlining the overall data processing cycle.

4. Conclusion

In this work, the ping-pong SRAM-CIM macro proposed in [1] is implemented with the new design of an adaptive sensing margin and precision scheme, aiming to minimize computation time through full-time utilization while maintaining precision, effectively increasing the throughput. The ping-pong weight update mechanism enables the CIM processor to execute large NN models far beyond the on-chip storage capacity, while the energy efficiency and accuracy can be trade off by the flexible precision ADC utilizing input sparsity to save power.

This work achieves an adaptive sensing margin and precision through switching between 2 operating modes: acc8 mode for supporting MAC accumulation up to 8 with a 3-bit ADC output; acc16 mode which supports the full range of accumulation for a 4-bit ADC output. Operations in acc8 mode saves 48.71% of power through switching off one of the 8-to-3 priority encoders compared to acc16 mode. The flexible precision ADC using priority encoders saves 2~3 sensing cycles (for 3 ~ 4-bit precision) compared to common SAR ADCs traded off using chip area, while the ping-pong weight update increases performance gain in time seen in Fig. 4. These results validate the advantages detailed in [1] and underline the significance of innovation while highlighting the necessity of trade-offs in circuit design.

In summary, the implementation of ping-pong CIM with adaptive precision ADC showcases a robust solution to the challenges faced by conventional CIM architectures, particularly in terms of scalability and efficiency. The proposed system is highly suitable for efficient computation of large NN models, presenting a shortened total operation time while maintaining MAC precision for achieving high throughput, low power computation. These enhancements confirm the significant potential of advanced CIM structures over traditional setups, improving the practicality of deploying complex neural network models in real-world applications.

5. Reference

[1] J. Yue, X. Feng, et al., “A 2.75-to-75.9TOPS/W Computing-in-Memory NN Processor Supporting Set-Associate Block-Wise Zero Skipping and Ping-Pong CIM with Simultaneous Computation and Weight Updating” ISSCC 2021

[2] J. Yue, Y. Liu, et al., “An Energy-Efficient Computing-in-Memory NN Processor With Set-Associate Blockwise Sparsity and Ping-Pong Weight Update” JSSC 2023

[3] J. Su, X. Si, et al., “Two-Way Transpose Multibit 6T SRAM Computing-in-Memory Macro for Inference-Training AI Edge Chips” JSSC 2022

[4] T. Kobayashi, K. Nogami, et al., “A current-controlled latch sense amplifier and a static power-saving input buffer for low-power architecture” JSSC 1993

6. Review and Reflection

During the early stages of this special topic implementation research, under the guidance of mentors in Professor Meng-Fan, Chang’s intelligent Memory Design Lab (iMDL), I started by surveying fundamental principles and topics, practicing presentations skills surrounding computing-in-memory and fundamental memory structures, while also familiarizing myself with the 65nm TSMC process. Specific topics covered included SRAM structure, neural network architecture, and basic physically unclonable function (PUF) design. Initially focusing on a broad understanding of CIM operations, I then narrowed my focus to the structure of SRAM CIMs and their peripheral circuits.

After reviewing publications, I selected [1] as the foundation for this research project, aiming for a successful implementation and potential innovative performance enhancements. Tools such as HSPICE, Virtuoso, and Waveview were used for simulation and testing. Weekly meetings with peers and mentors in iMDL helped monitor progress and provided immediate feedback. During implementation, I encountered challenges with complex circuit architecture, including voltage spikes and signal coupling. These tested my analog design skills and required deeper research. Through troubleshooting, I combined the proposed circuit in [1] with an optimized adaptive sensing scheme for successful implementation.

Throughout this special topic implementation, I often faced problems beyond my grasp. Reporting progress and discussing issues with mentors, Professor Chang, and peers in iMDL was invaluable. Their advanced insight in CIM computing offered guidance through obstacles and provided key skills for the problem solving. Despite working independently on this project, collaboration with iMDL highlighted the importance of teamwork and creative thinking. This experience underscored the value of determination and innovation, while also broadening my understanding, deepening my knowledge, and increasing my confidence in tackling engineering designs and complex problems. I am deeply grateful for the guidance and support received throughout this journey.