SCIENCE CHINA
Information Sciences

October 2023, Vol. 66 200403:1–200403:19
https://doi.org/10.1007/s11432-023-3800-9

• REVIEW •
Special Topic: Recent Progress of Fundamental Research on Post-Moore Novel Devices

From macro to microarchitecture: reviews and trends of SRAM-based compute-in-memory circuits

Zhaoyang ZHANG, Jinwu CHEN, Xi CHEN, An GUO, Bo WANG, Tianzhu XIONG, Yuyao KONG, Xingyu PU, Shengnan HE, Xin SI* & Jun YANG*

National Application Specific Integrated Circuit Center, Southeast University, Nanjing 210096*, China*

Received 23 April 2023/Revised 6 June 2023/Accepted 9 June 2023/Published online 21 September 2023

Abstract The rapid growth of CMOS logic circuits has surpassed the advancements in memory access, leading to significant "memory wall" bottlenecks, particularly in artificial intelligence applications. To address this challenge, compute-in-memory (CIM) has emerged as a promising approach to enhance the performance, area efficiency, and energy efficiency of computing systems. By enabling memory cells to perform parallel computations, CIM improves data reuse and minimizes data movement between the memory and the processor. This study conducts a comprehensive review of various domains of SRAM-based CIM macros and their associated computing paradigms. Additionally, it presents a survey of recent SRAM-CIM macros, with a specific focus on the key challenges and design tradeoffs involved. Furthermore, this research identifies potential future trends in SRAM-CIM macro-level design, including hybrid computing, precision enhancement, and operator reconfiguration. These trends aim to resolve the tradeoff between computational accuracy, energy efficiency, and support for diverse operators within the SRAM-CIM framework. At the microarchitecture level, two possible solutions for tradeoffs are proposed: chiplet integration and sparsity optimization. Finally, research perspectives are proposed for future development.

Keywords artificial intelligence (AI), compute-in-memory (CIM), static random access memory (SRAM)

Citation Zhang Z Y, Chen J W, Chen X, et al. From macro to microarchitecture: reviews and trends of SRAM-based compute-in-memory circuits. Sci China Inf Sci, 2023, 66(10): 200403, https://doi.org/10.1007/s11432-023-3800-9

1 Introduction

In recent years we have witnessed a growing number of artificial intelligence (AI) applications in the fields of medicine, finance, transportation, and entertainment. The throughput required by AI models is experiencing exponential growth, while the energy efficiency gains from scaling down are gradually declining [1–4]. To bridge this "strong and weak molar" gap, compute-in-memory (CIM) is expected to be a new principle of computing. On the one hand, by calculating inside the memory, it can significantly reduce the number of data accesses. On the other hand, an order of magnitude multiplication of computational energy efficiency can be achieved through different computing paradigms.

CIM refers to the latest computing paradigm that combines memory and computation within the same hardware unit [5]. CIM promises to be the approach to improving the energy efficiency of AI edge and AIoT devices [6]. Different from traditional computing systems, in which memory and processing units are separate entities, leading to data transfer overheads and energy inefficiencies, CIM enables the memory unit to perform computation, thus reducing data movement and improving energy efficiency [7]. As shown in Figure 1, the energy efficiency of traditional digital computing AI chips based on advanced processes has been improved slowly. Meanwhile, CIM AI chips, while still immature in terms of multiple data types and multi-operator support, have demonstrated their great potential in energy efficiency [8,9].

In recent years, memory cells employed in CIM circuits can be divided into volatile and non-volatile memory devices [10–24]. Static random-access memory (SRAM) is currently one of the most mature and

* Corresponding author (email: xinsi@seu.edu.cn, dragon@seu.edu.cn)

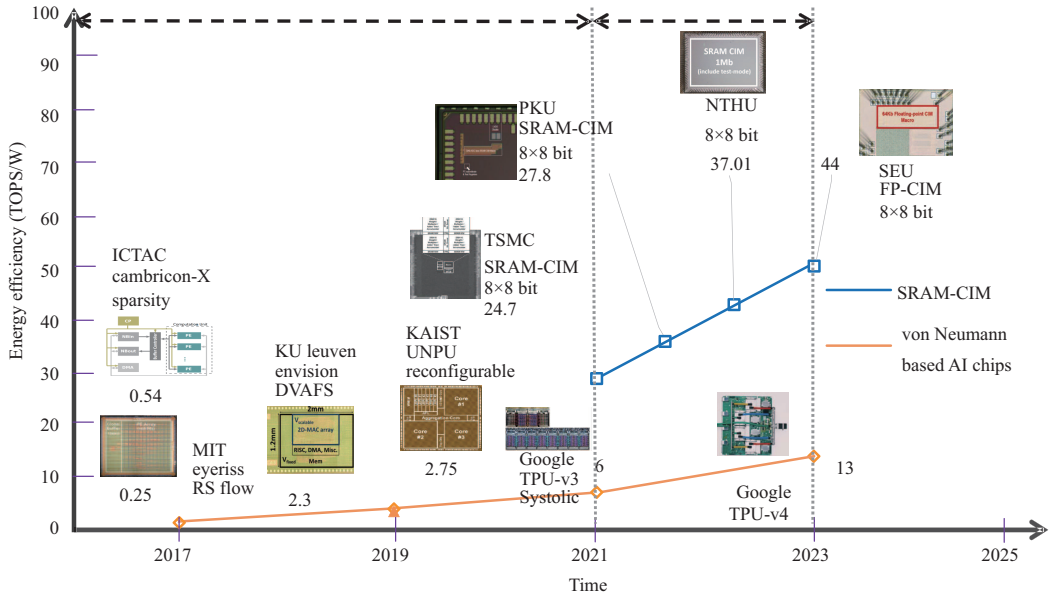


Figure 1 (Color online) The energy efficiency of traditional digital computing AI chips and SRAM-CIM.

stable technologies that exhibits high speed and durability. The newly developed embedded non-volatile memories (eNVM) such as phase-change random memory (phase-change RAM, PcRAM), spin-transfer torque magnetic RAM (STT-MRAM), ferroelectric memory (ferroelectric RAM, FeRAM), and resistive RAM (RRAM) are compatible with the CMOS BEOL process, and these emerging memories have high array density, high energy efficiency but limited endurance. This paper mainly focuses on the development of SRAM-CIM.

Figure 2 enumerates the representative studies at the macro and microarchitecture level CIM technology in recent years. After the rapid development of CIM technology over the span of years, both academia and industry carried out a series of exploration and research in the fields of device types, computing paradigms, and overall architectures. They applied CIM technology to machine learning, edge computing, parallel computing, and reconfigurable computing initially, further verifying the feasibility of CIM design and its great potential in AI applications.

The remainder of the paper is organized as follows. Section 2 introduces the computing paradigm in SRAM-CIM macro design and challenges in its further development. Section 3 analyzes the design tradeoffs of recent SRAM-CIM macros. Section 4 presents several potential research trends for SRAM-based CIM at the macro and microarchitecture level. The conclusion is drawn in Section 5.

2 Computing paradigm in SRAM-CIM macro design

There are a large number of computational operations in AI-oriented applications. CIM was initially designed to solve the multiply-accumulate (MAC) computation and memory access problem in convolutional neural networks (CNNs). CNN and fully convolutional networks (FCNs) require a lot of pixel-by-pixel and channel-by-channel MAC computation, and their large number of MAC and memory access operations consume a lot of energy. The CIM computing paradigm achieves fast MAC operations by involving multiple weight values stored in SRAM arrays in bitwise multiplication with input feature values and completing accumulation through accumulation paths, which greatly reduces computational energy consumption. The core lies in how to efficiently perform "in-memory multiplication" and "in-memory accumulation".

At the early stage of the development of CIM macro, traditional digital computing was superseded by analog computing, which reduced the computational energy consumption by replacing the digital quantity with analog quantity to complete MAC, but at the same time introduced computational errors and reduced the computational accuracy due to the non-ideal characteristics of the device and the fluctuation of the PVT (process, voltage and temperature) environment. To address these problems, some researchers started to focus on the use of partial digital computing instead of analog computing to

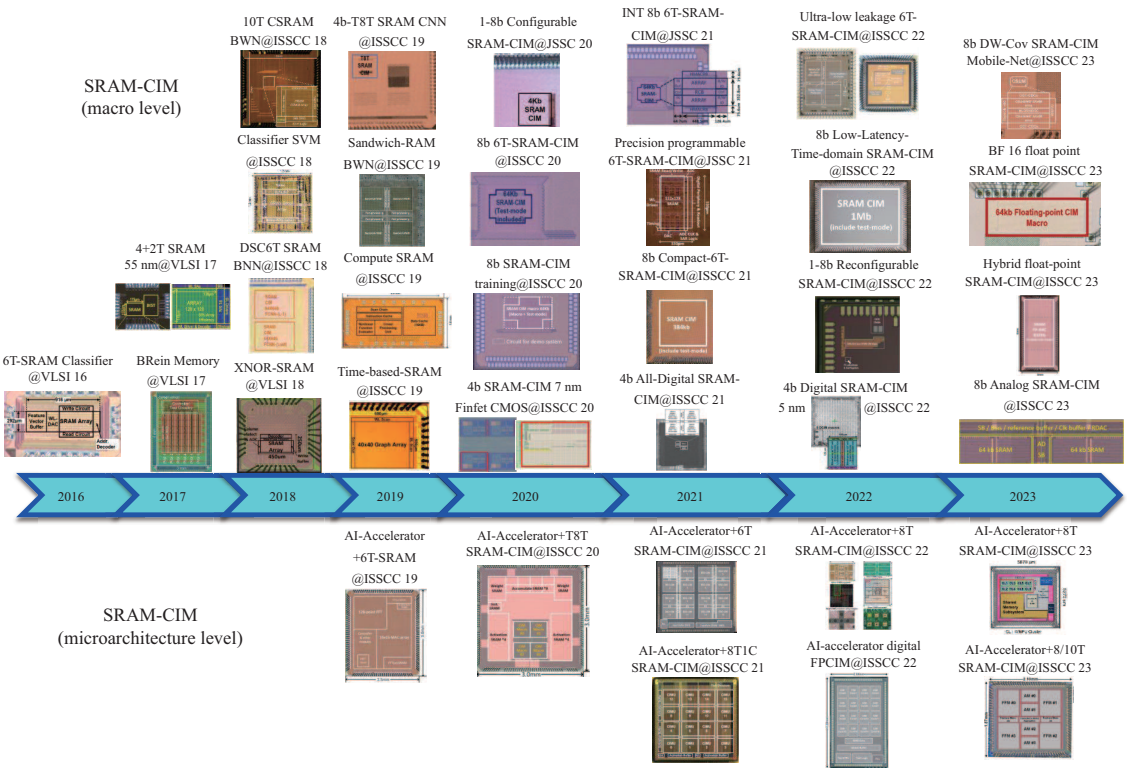


Figure 2 (Color online) The representative studies on SRAM-CIM.

achieve the guarantee of computational accuracy, while others centered on the architecture of sufficiently flexible and reliable analog computing circuits. This section reviews the silicon-verified SRAM-CIM macros in academia and industry in previous years and classifies them as time domain [25–41], current domain [42–54], charge domain [55–70], and digital domain [71, 72] according to different computation paradigms. In these schemes, the logic operation of analog domain computation (time domain, electric domain, voltage domain) is implemented by modifying the standard 6T SRAM cell or adding transistors to the logic operation. Digital domain computation is achieved by adding logic gates next to the SRAM bit-cells or next to the SRAM array to complete more logic operations. Each computing paradigm has its own characteristics, and this section concludes with a comparative analysis of the advantages and disadvantages of each one.

2.1 Time domain CIM

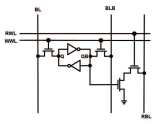
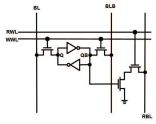
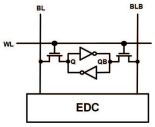
In the time domain, data is expressed in terms of pulse widths or path delays, where MAC is achieved by varying the pulse width proportionally [31] or by comparing the time difference between the rising edges of the pulses [38, 40]. A time-to-digital converter (TDC) is required for readout. As pulses in time domain computing have rare switching activity, energy consumption is low and is more in line with the trend towards data-intensive applications. Time-domain-CIM (TDCIM) is therefore the ideal CIM architecture for implementing energy-efficient edge AI processors. The time-domain computing moves beyond the limits of the dynamic range of analog signals, and larger data features wider pulses, which in theory can be extended in width indefinitely. However, due to time-domain computing unfolds computation and quantization in the time domain, the computation is relatively slow and the larger the data is, the longer the computation takes, thus reducing the arithmetic power of the memory computation unit.

This subsection examines a selection of silicon-verified SRAM-CIM studies adopting time domain computing paradigm (see Table 1) [31, 38, 73].

2.1.1 *Sandwich-RAM*

Yang et al. [31] proposed a "sandwich" structure-based SRAM-CIM circuit design. The CIM macro stacks weights, eigenvalues, and MAC operations on top of each other, the design utilized a tightly coupled

Table 1 Comparison table of time-domain SRAM-CIM studies

Reference	ISSCC19 [31]	JSSC21 [73]	ISSCC22 [38]
Technology (nm)	28	28	28
Supply voltage (V)	0.6–0.9	0.55–1.05	0.65–0.9
Cell-structure			
Bit-cell	8T	8T	6T+EDC
Input precision	8	8	8
Weight precision	1	1–8	8
Output precision	8	8	22
Energy efficiency (TOPS/W)	46.6	60.18	37.01

memory-computation-storage structure shaped like a "sandwich" and uses a pulse width modulation (PWM) unit (PWMU) based time domain computation circuit to perform MAC operations. Pulse quantizer by replica delay units (PQRDC) converted the calculated pulse into a digital output. In addition, a delay-sensitive control-voltage generator (DSCVG) was put forward to dynamically track the PVT variation in order to improve the accuracy and power efficiency of the analog calculation. Two energy efficiency enhancement schemes that reduce the amount of computation in the memory were also proposed to improve power efficiency. The design was flow-tested and validated for use in a binary weight network (BWN), resulting in a peak power efficiency of 119.7 TOPS/W.

2.1.2 *TIMAQ: a time-domain computing-in-memory based processor*

Yang et al. [73] proposed a TD-CIM-based processor TIMAQ. Supporting both convolutional (CONV) and fully connected (FC) layers in 1-8b NUQ (nonuniform quantization)- and UQ (uniform quantization)-DNNs (deep neural networks), this design employed a time-domain CIM macro cell to implement convolution for bitmap convolutions. TD-CIM macros used PWM to perform MACs. The bit-cell used a read/write separated 8T structure to read out the 8b data into the pulse delay cell. The PDC (pulse delay cell) adopted a delay chain structure with separate MSB (most significant bit) and LSB (least significant bit). The delay chain internally used a 2–4 decoder to convert the 2-b data into four voltage control pulse widths from V0–V3. This macro can compute 1152 MACs each time. TIMAQ achieved 60.18 and 29.78 TOPS/W system-level energy efficiency for 1-b and 2-b uniform quantized DNNs.

2.1.3 *Time-domain computing-in-memory 6T-SRAM macro*

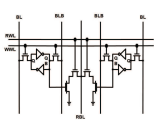
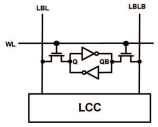
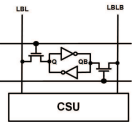
Wu et al. [38] developed a 28 nm 1 Mb SRAM macro. This time domain CIM macro employed standard 6T bit-cell. The 2-column SRAM array shared a single delay computing unit (DCU). The input 2-bit data was converted into an analog input voltage (VSS, V01, V10, or V11) to the MUL node. The modulation of the gate delay by different inputs and weights resulted in a delay of kt between the input and the output, which characterizes the multiplication result and transmits the delay to the next level. The EDC (edge-delay cell) introduces an intrinsic delay (t_0) when the result is 00. When the result is 01, the EDC generates a delay of $t_0 + \Delta t$, where Δt represents the delay-step of the EDC. In this work, we have set Δt to be 30 ps, resulting in remarkably low latency. This work also proposed dynamic a differential-reference TDC (D2REF-TDC). Faster TDC quantization is achieved by different REF values adopted bias reference differencing. This work registered a 6.6 ns access time (tAC) and a 37.01 TOPS/W energy efficiency for a nearly full output-ratio (22b-OUT).

These designs have demonstrated progressive moving towards higher accuracy and shorter computation times in recent years, as well as increased energy efficiency in time-domain CIM. Since the most distinctive feature of time-domain computing is the accumulation of results by delay, it is still difficult to break through the logical operations outside the MAC for time-domain CIM.

2.2 Current domain CIM

In the current domain, data is expressed in terms of the magnitude of the current and the MAC is implemented by the accumulation of current on the current accumulation line. The readout requires an

Table 2 Comparison table of current-domain SRAM-CIM studies

Reference	ISSCC19 [44]	ISSCC20 [46]	ISSCC23 [74]
Technology (nm)	55	28	28
Supply voltage (V)	0.7–1.0	0.7–0.9	0.6–0.9
Cell-structure			
Bit-cell	T8T	6T+LCC	SW6T+CSU
Input precision	4	8	8
Weight precision	5	8	8
Output precision	7	20	20
Energy efficiency (TOPS/W)	18.37	16.63	33.44

analog-to-digital converter (ADC) and the SRAM reads the data based on the bit line voltage value. Some of the current domain CIM designs utilize the BL (bit line) accumulation current to achieve MAC operation by changing the specification of the bit-cell while turning on multiple lines of words [43, 44]. Some of the current domain CIM designs implement the current representation of multiplication results through local computation cells, in conjunction with the use of external current accumulation lines to accumulate current for MAC operation [46, 74]. The use of current domain computation allows for highly area-efficient and energy-efficient computation in a neural network accelerator. However, since current domain computation converts digital signals into analog signals for computation and quantization into digital signals. The reliance on high performance digital-to-analog converters (DACs) and ADCs increases the overhead of the circuit and reduces the computational power of the stored computational units.

The compute current variation in this process is influenced by the operating state of the transistors. Transistors are subject to variations due to process variability, temperature effects, and power supply noise. To address these variations, dummy computing modules are commonly incorporated to provide a reference current for calibration purposes. By comparing the current outputs of the actual computation modules with the reference current, adjustments can be made to compensate for the compute current variation. Furthermore, the compute current variation of the quantization circuit is dependent on its specific quantization methodology. In the quantization process, the conversion of current into voltage is typically achieved using resistors or capacitors. However, the introduction of resistors and capacitors in the quantization circuit introduces variations in capacitance and resistance. These variations contribute to the overall compute current variation, resulting in a more significant cumulative effect.

This subsection examines some of the silicon-verified SRAM-CIM studies using the current domain computing paradigm (see Table 2) [44, 46, 74].

2.2.1 *A twin-8T SRAM computation-in-memory macro*

Si et al. [44] designed a CIM macro based on twin 8T SRAM cells. The 2 bit weights were stored in two immediately adjacent 8T SRAM cells, the right LSB cell used a standard 8T bit-cell with a discharge tube control of 1× for the readout bit-line (RBL), and the left MSB cell increased the output discharge tube so that its discharge current to the RBL was LSB twice as much, the left and right bit-cell shared a common current accumulation line, realizing a tightly coupled multi-bit weight calculation at the same time, achieving 4 discharge stages through the RWL (read word line) voltage, and reflecting 16 current differences on the RBL. The solution ultimately reached an energy efficiency of 72.1 TOPS/W.

2.2.2 *A 28 nm 64 Kb 6T SRAM-CIM macro with LCC*

Si et al. [46] proposed a CIM macro for CIM based on a local computation cell, featuring up to 8b-IN, 8b-W, and 20b output accuracy. At the level of data mapping and computational texture, this work uses weight bit MAC operation to extend sensing margins and improve IN/W/OUT accuracy. At the architectural level, this work uses 6T local computing cells (LCC) for compact area and robust reading of process variations, this research achieved an energy efficiency of 16.63 TOPS/W.

2.2.3 *A 28-nm separate-WL 6T-SRAM-CIM unit-macro*

Wang et al. [74] proposed a CIM macro at both the architectural and circuit levels. At the architectural level, a flexible and configurable memory-computing array was proposed for the first time, which supports two data mapping methods, improving array utilization and greatly reducing data update costs. At the circuit level, (a) a shift-multiply-add local computation unit embedded in a separate word line SRAM array was proposed for the first time to support CIM and low data update cost convolutional longitudinal shift operations. (b) A weighted transverse shift circuit and alternate load-computation scheme were developed to efficiently perform convolutional transverse shift operations while significantly reducing data storage overhead. (c) A current-digital quantization readout circuit with computational current tracking was established. The proposed current-digital quantization readout circuit (CCT-CDC), with a shared reference current generation circuit and a "three-step quantization" readout acceleration scheme, significantly improved the area efficiency of the storage array while reducing the computation-readout access time compared to conventional SAR (successive-approximation register) ADCs.

Previous studies have reported innovations in the design of the bit-cell. Unique designs paired with custom-designed data mapping methods and calculations have achieved greater energy and area efficiencies. However, the issue of analog-to-digital conversion overhead in current domain computing remains challenging.

2.3 Charge domain CIM

In the charge domain, the data is expressed in terms of voltage magnitude. The individual multiplication results are fed into a parallel array of capacitors, and then the redistribution of charge on the capacitors completes the MAC operation. An ADC is required for readout. Since charge domain calculations usually require charge sharing on the columns, MAC calculations are often implemented at the architectural level using a read/write separated bit-cell structure with word-lines turned on at the same time. This achieves a very high memory-to-computation ratio. Moreover, capacitive coupling and charge sharing schemes usually achieve a high degree of linearity and improve the accuracy of the calculation. As with the current domain, the charge domain also requires the introduction of a large number of DACs and ADCs, adding a lot of additional circuit overhead.

This subsection outlines some of the silicon-verified SRAM-CIM studies that adopt charge domain computing paradigm (see Table 3) [69, 75, 76].

2.3.1 *A 28 nm 384 kb 6T-SRAM computation-in-memory macro*

Su et al. [69] proposed a charge domain SRAM-CIM macro. At the overall architecture level, a segmented-BL charge-sharing (SBCS) scheme was used to share the results of local operations with the rest of the operations for overall charge sharing. A conventional 6T structure was applied to the bit-cell, paired with a new LCC cell called the source injection local multiplication cell (SILMC), which separated storage from computation. Reducing area overhead with a prioritized-hybrid-ADC (Ph-ADC) at the analog-to-digital conversion level, this work supported the accumulation of 8b inputs and 8b weights on 16 channels, with near-full precision outputs. This macro achieved a 7.2 ns tAC and a 22.75 TOPS/W energy efficiency.

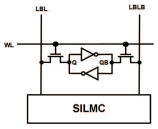
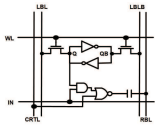
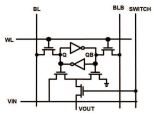
2.3.2 *A fully bit-flexible computation in memory macro*

Yao et al. [76] proposed a fully bit-flexible CIM macro. A CIM computing bit cell (CIMC) was formed using a conventional 6T cell on a bit-cell with gates, non-gates, and capacitors. The four operating modes were standard read/write access, 1-b×1-b MAC, reference voltage generation, and memory A/D conversion. This design improved area efficiency and reduced ADC overhead. This work also adopted an embedded input sparsity sensing and an automatic on-chip reference voltage generation scheme on the ADC circuit. Adaptive dynamic range extension based on real-time input sparsity characteristics was achieved. In addition, the use of interleaved layouts and the CIMC structure enabled simultaneous CIM and writing ping-pong operations. The energy efficiency of the proposed CIM design reached 383 TOPS/W(1×1).

2.3.3 *PVT-insensitive 8b word-wise ACIM*

Hsieh et al. [75] proposed a 12 nm FinFET CIM macro. The charge-sharing MAC achieved PVT-insensitive linear 1bIN 1bW accumulation. The multiplication of 8 bit IN with 1 bit W was implemented

Table 3 Comparison table of charge-domain SRAM-CIM studies

Reference	ISSCC21 [69]	JSSC23 [76]	ISSCC23 [75]
Technology (nm)	28	28	12 Finfet
Supply voltage (V)	0.7–0.9	0.7–0.8	0.7–0.8
Cell-structure			
Bit-cell	6T+SILMC	6T+AND+NOR+1C	9T
Input precision	8	1/4/8/16	8
Weight precision	8	1/4/8/16	8
Output precision	20	8	8
Energy efficiency (TOPS/W)	22.75	5.98	86.27

at the memory structure level using a 6T bit-cell + 3T transfer tube control logic structure. The 8-bit W stored in the SRAM was divided into a 4-bit LSB and a 4-bit MSB, which corresponded to a capacitive array of 1, 2, 4, and 8 C respectively for capacitive coupling. The coupling charges of the LSB and MSB were shared on the 64 channels and fed into the 8 bit SAR ADC via a buffer at the output. This achieved a 70.85–86.27 TOPS/W energy efficiency and more than 10b linearity.

The studies have provided evidence that work in the charge domain in recent years was designated to solve different paradigms of capacitive coupling and charge distribution and the corresponding energy efficiency improvements in ADC quantization circuits. So far, current operations are mainly confined to traditional MAC operations, and there have been limited attempts at new operators.

2.4 Digital domain CIM

In the digital domain, the data is represented as the output of a logic gate and an adder tree, where the MAC is implemented by multiplying the logic gate to obtain a partial product and using the adder tree to achieve accumulation. The data is always stored and calculated in digital form. There is no need for additional DACs and ADCs. Since the addition and multiplication in digital domain calculations are implemented using digital circuitry, full precision outputs can be achieved with an output ratio of 1. However, the accumulation of digital domain calculations relies on the adder tree, leading to a significant area overhead within the array. Digital domain computation addresses the problem of low accuracy in analog domain computation, allowing greater data throughput and better adaptation to a wider range of data types (binary, INT8, BF16). In spite of that, compared with analog domain computation, the less sensitivity of the digital domain to data sparsity causes an unnecessary loss of energy efficiency.

This subsection reviews some of the silicon-verified SRAM-CIM studies that use digital domain computing paradigm (see Table 4) [71, 72, 77].

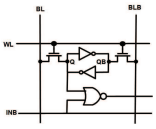
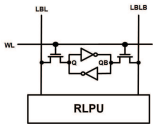
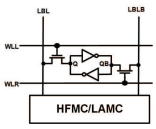
2.4.1 *An all-digital SRAM-based full-precision CIM macro in 22 nm*

Chih et al. [71] proposed the first all-digital SRAM-CIM macro. The bit-cell used a conventional 6T structure with a non-gate to implement multiplication. Due to the additive flexibility of numeric domain calculations, arrays could support input activations with programmable bit widths (1–8 per macro), signed or unsigned, and weights with four different bit widths (4, 8, 12, or 16). The parallelism of the MAC was enhanced at the architectural level using a new architecture based on bit-serial multiplication and parallel adder trees. A 30 percent improvement in energy efficiency was achieved by interleaving 14T and 28T full adders in the adder tree. This work accomplished 89TOPS/W energy efficiency in a 22 nm logic process.

2.4.2 *A signed-INT8 dynamic-logic-based ADC-less SRAM-CIM macro*

Yan et al. [72] proposed an ADC-less SRAM-CIM macro. Reconfigurable local processing units (RLPUs) within bit cell arrays supporting reconfigurable bit operations including AND, XOR, and OR. Dynamic logic makes more operator support possible. The summation circuit was implemented using an adder tree, and a bypass design was added to the adder tree so that it can support the depth-wise convolution of the mobile-net. This work also extended the MAC operation to vector matrix multiplication (VMM)

Table 4 Comparison table of digital-domain SRAM-CIM studies

Reference	ISSCC21 [71]	ISSCC22 [72]	ISSCC23 [77]
Technology (nm)	22	28	28
Supply voltage (V)	0.72	0.8	0.6–0.9
Cell-structure			
Bit-cell	6T+NOR	6T+RLPU	DB6T+HFMC/LAMC
Input precision	1–8	1–8	8
Weight precision	4/8/12/16	1/4/8	8
Output precision	16/24	8/21	23
Energy efficiency (TOPS/W)	24.7	27.38	44

and vector Hadamard product (VHP) calculations, expanding the application scenario of CIM. This work achieved 19.21–35.55 TOPS/W energy efficiency in signed 8b integer (INT8) inputs and weights.

2.4.3 *A 28 nm 64-kb digital-domain macro for floating-point CNNs*

Guo et al. [77] proposed a floating-point convolutional neural network oriented digital CIM macro, which adopted double 6T cell and high bit full precision low bit approximate computing unit structure, reaching the highest energy efficiency compared to previous studies while taking into account the tradeoff among energy efficiency, area efficiency, and memory density. The double-6T structure using split word lines on the bit-cell allows 2 bit weights to be read out in the same cycle to participate in the calculation. The partial approximation, partial full precision calculation approach accomplishes an optimal trade-off between energy efficiency, area efficiency, and calculation accuracy. This work achieved 31.6-TFLOPS/W energy efficiency in BF16.

As suggested by the research above, the work in the digital domain in recent time has mainly been exploring different approaches to data mapping and logical computer reasoning, as well as area efficiency improvements in adder trees. These studies focus on the trade-offs between output ratio, inference accuracy, and circuit overhead. However, the overhead of the adder tree is difficult to avoid, and the overhead ratio of the whole circuit is still large. This overhead limits further improvements in the energy efficiency of the digital domain SRAM-CIM.

The charge domain computing fits better with the characteristics of SRAM cell read operations, but there are some drawbacks. The charge domain is graded by voltage, with each voltage value corresponding to one piece of data, and since there is an upper limit to the operating voltage, a sufficient voltage margin is required for each voltage level. Furthermore, as high energy efficiency forces the operating voltage towards the near threshold, the dynamic range of the voltage signal is further reduced, and its disadvantage of insufficient voltage margin is gradually amplified.

The current domain computing is similarly limited by the dynamic range of the analog signal. At the same time, the high computational currents in electrical basin calculations result in relatively high computational energy consumption. In order to improve throughput in electric basin computing, it is necessary to increase the number of parallel computations, which in turn leads to higher computing power consumption. Currently, there is no optimal solution to address the trade-off between throughput and computing power consumption.

In theory, time domain computing can have infinitely scalable pulse widths, with less flip-flopping of the computational signal and lower energy efficiency. However, by unfolding the computation and quantization in the time domain, it is susceptible to PVT fluctuations and the computation speed is relatively slow.

Although digital domain computation has seen a significant improvement over the analog domain computation described above in terms of dynamic reconfigurability, type of computational logic, and computational accuracy modulation, the area overhead of digital logic remains inefficient.

In summary, each of the four computational approaches has its own advantages and disadvantages, and there are trade-offs among them that will collectively navigate in-deposit computing into the future.

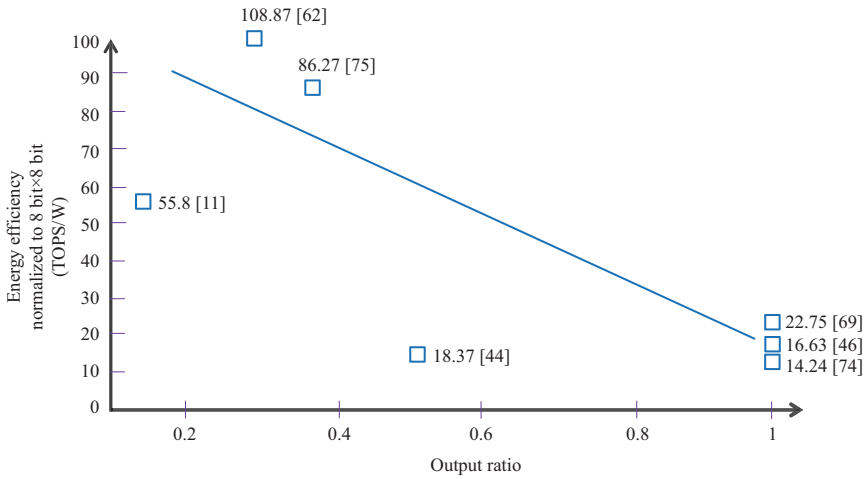


Figure 3 (Color online) Tradeoff between output ratio and energy efficiency.

3 SRAM-CIM design tradeoffs

In the initial SRAM-CIM work, various institutions and companies have made some explorations on the design of CIM macros. However, from an architecture perspective, completing the design of macros alone does not enable a complete AI computing deployment. In addition to CIM macro, supporting memory layers, data flow, and instruction design, are needed to form a complete AI acceleration core or even a system-on-chip that enables AI computing. Therefore, in recent years, there have been a number of microarchitecture designs based on CIM macro.

This section analyzes the tradeoffs in SRAM-CIM at both the macro and microarchitecture levels. The design tradeoffs are found through the analysis of the silicon-verified SRAM-CIM macro. The design tradeoffs of microarchitecture are sought through an abstract analysis based on the existing macro work.

3.1 Tradeoffs at the macro level

There are many tradeoffs and considerations in the design of the SRAM-CIM macro. Energy efficiency, as one of the most significant metrics, has always been the focus of research by various institutions and companies. High energy efficiency enables higher computing throughput with lower energy, making it possible to use larger amounts of data and more complex network types. By analyzing the parameters of the recently silicon-verified SRAM-CIM operation, this work identifies three important metrics that constrain the energy efficiency of SRAM-CIM and points out the potential for further breakthroughs.

3.1.1 *High energy efficiency and high precision*

In both analog and digital domain CIM, higher precision requirements will bring additional overhead to the circuit. In analog domain computing, higher accuracy is accompanied by more digital-to-analog conversion and analog-to-digital conversion overhead. In digital domain computing, higher precision is associated with more digital logic circuit overhead.

As shown in Figure 3 [11, 44, 46, 62, 69, 74, 75], the output ratio (actual output accuracy/theoretical output accuracy) of existing CIM is inversely proportional to energy efficiency under different data accuracy requirements. Therefore, in many efforts to pursue energy efficiency, higher energy efficiency is often achieved by reducing the output ratio. For neural networks with good robustness, such as VGG (visual geometry group) networks, a low output ratio will not affect its reasoning accuracy [5]. However, for other networks with high precision requirements, a higher output ratio is essential for general purposes.

3.1.2 *High energy efficiency and high throughput*

There is often a trade-off between throughput and energy efficiency in SRAM-CIM. On the one hand, increasing the throughput typically involves increasing the clock frequency or parallelism of the CIM macro. This can result in higher power consumption due to increased activity and more frequent data movement. Additionally, more power may be required to drive the larger number of transistors in the

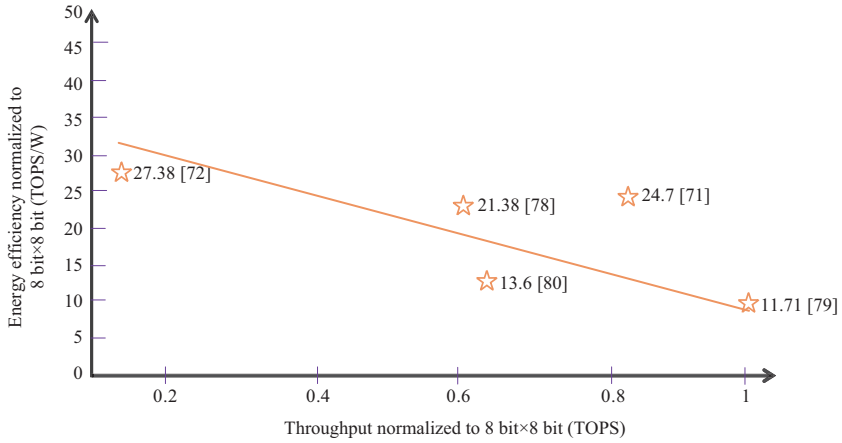


Figure 4 (Color online) Tradeoff between throughput and energy efficiency.

macro. On the other hand, improving energy efficiency often involves reducing power consumption through techniques such as reducing supply voltage or minimizing data movement. However, these techniques may lead to lower throughput due to slower operation or reduced parallelism.

As shown in Figure 4 [71, 72, 78–80], throughput and energy efficiency showed an inverse relationship. Throughput measures the area efficiency of the CIM design, energy efficiency measures the energy consumption performance, one corresponds to the area overhead of the design, the other corresponds to the power overhead. When the design focuses on area optimization, it is difficult to optimize energy consumption at the same time, which is a tradeoff in design. With the development of technology and design progress, the FOM (figure of merit) of computing power density and energy efficiency is gradually increasing.

3.1.3 *High energy efficiency and reconfigurability*

So far, CIM can only support the calculation of certain layers of some networks since it gained attention. The remaining layers of the network still rely on conventional digital circuits for acceleration, which may limit the further improvement of system-level energy efficiency. As a result, CIM gradually moves towards universality, which requires CIM to support more operators and cover most or even all network layers, further reduce the data handling of CIM macro, external storage and external logical operation circuits.

As Figure 5 [72, 74, 77, 81] shows, more operator support contradicts greater energy efficiency. Current energy-efficient CIMs support only specific operators, such as MACs. However, the need for operator support grows as networks and network layer operations increase, which have to be realized through various reusable and reconfigurable logical operation units combined with unique data mapping methods. This poses new challenges to the architecture design of CIM.

3.2 Tradeoffs at the microarchitecture level

Compared to CIM macro, there is more design freedom existing in the design of the SRAM-CIM microarchitecture. There are two key aspects of CIM microarchitecture design. One is the memory hierarchy design and the other is the dataflow design. As shown in Figure 6, this work evaluated a more basic typical CIM-based AI microarchitecture, with a three-level structure of two levels of SRAM buffer and one level of CIM computation. The calculation is processed by six INT8-CIM macros. The test performance of silicon verification is used as model parameters of the CIM macro.

A typical Conv2 layer in ResNet18 is developed on the proposed AI-core architecture using the ImageNet dataset. The area overhead and energy consumption of each part are evaluated by pairing existing macro-level work with abstract analysis. The evaluation result identifies two important metrics that constrain the SRAM-CIM microarchitecture, and points to the potential for further breakthroughs.

3.2.1 *Memory size and system performance*

Under the initial conditions of a single conv2 layer with a 256 kB shared memory, a 32 kB local memory, a 128 B feed register and a 64 B accumulation register. The analysis has estimated the area and energy

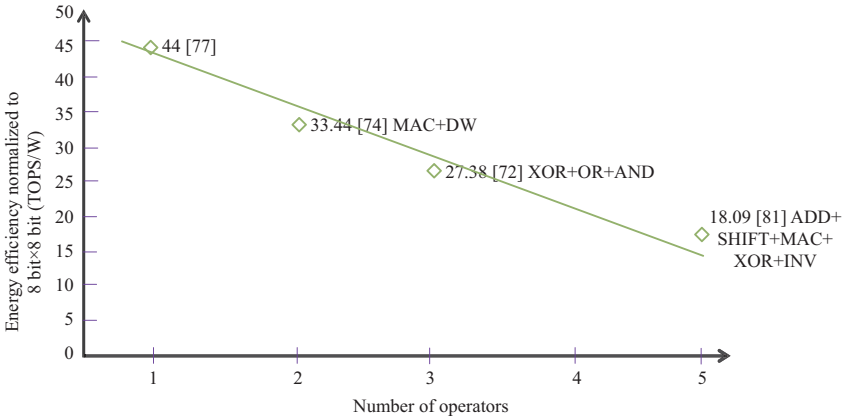


Figure 5 (Color online) Tradeoff between operators and energy efficiency.

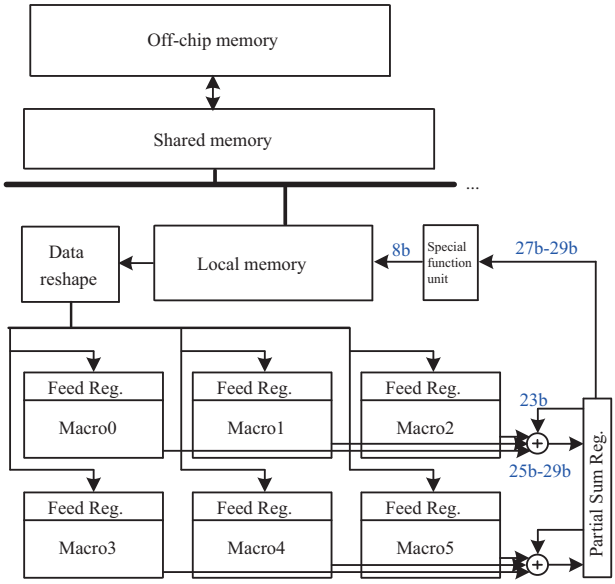


Figure 6 (Color online) A CIM-based AI microarchitecture.

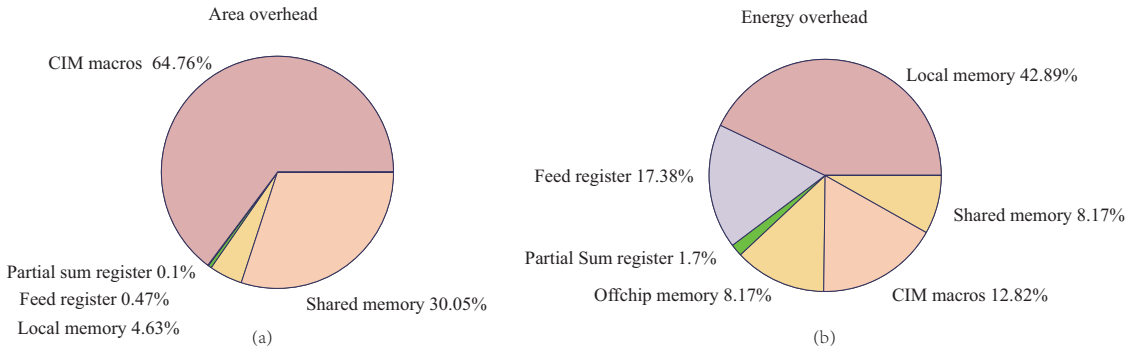


Figure 7 (Color online) (a) Area and (b) energy overhead of CIM microarchitecture.

efficiency of each part of the circuit.

As shown in Figure 7, CIM macros have the largest area overhead, accounting for 64 percent of the total area. Meanwhile, local memory has the largest energy consumption. As the feature is serially input, there is data multiplexing at the input register level for different bit feature values. 128 2bits-feature is fed to each CIM-macro in four cycles by feed register.

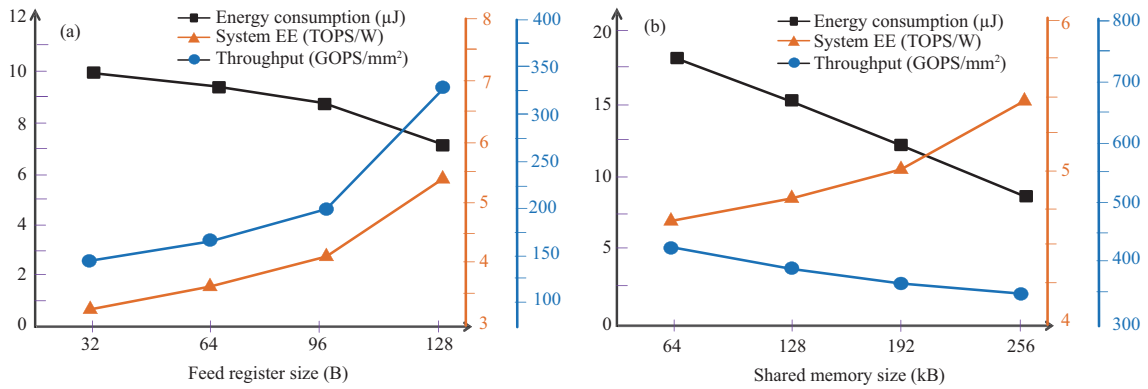


Figure 8 (Color online) (a) Relationship between feed reg size and system performance; (b) relationship between shared memory size and system performance.

As shown in Figure 8(a), when a larger feed register is used, the energy used to update data is reduced and the system's throughput and overall energy efficiency are significantly improved. As shown in Figure 8(b), when larger shared memory is used, the power consumption of shared and off-chip memory is significantly reduced. The throughput of the microarchitecture decreases slightly due to the larger memory area. The reduction in computing time associated with larger storage results in improved overall efficiency.

In summary, the larger the memory size used for caching, the easier data reuse in the microarchitecture. Large memory size reduces some data moving overhead and makes the microarchitecture more energy efficient. However, a large memory size will result in a large on-chip area, which will reduce area efficiency and reduce throughput.

3.2.2 *Dataflow and system performance*

Data flow design is another focus of microarchitecture design. Network operators in different networks have different data characteristics, and the characteristics of operations vary between different layers of the same network. A shallow network has a smaller number of channels and a larger feature map, and the feature is much larger than the weight. Therefore, shallow networks have a greater need for shift operations and input of features. The deeper networks have a larger number of channels and smaller features, and the weight is much larger than the feature, which often requires more weight update operations and fewer feature shifts. This poses a major challenge to microarchitecture design in terms of data flow scheduling and data mapping.

In the microarchitecture there are multiple macros of the same computation in parallel. As shown in Figure 9, two types of parallelism exist when different macros are working simultaneously. Column-parallel macros can process the same input feature at the same time, thus enabling the same feature to be input to a column-parallel macro, but the full multiplicative sum of the results in a row is not available, incurring some partial sum data storage overhead. The row-parallel approach is faster in obtaining the full multiplication result, but requires different features to be entered, resulting in a larger input bandwidth.

In summary, the column-parallel mapping approach is more friendly for shallow networks with a small number of channels, and the row-parallel approach is more friendly for deeper networks with smaller feature maps. Column-parallel solutions require more shared memory while row-parallel solutions require more feed register. The upper limit of data reuse for both mappings depends on the network.

4 From macro to microarchitecture: future trends of SRAM-CIM

Sections 1–3 describe recent research on SRAM-CIM. The advantages and disadvantages of different computing methods are analyzed from the perspective of computational mechanisms. At the macro level, tradeoffs between several key metrics are emphasized. At the microarchitecture level, the link between data reuse, memory size, and system performance is analyzed. This section addresses the bottlenecks in the development of SRAM-CIM and discusses potential directions for future SRAM-CIM work.

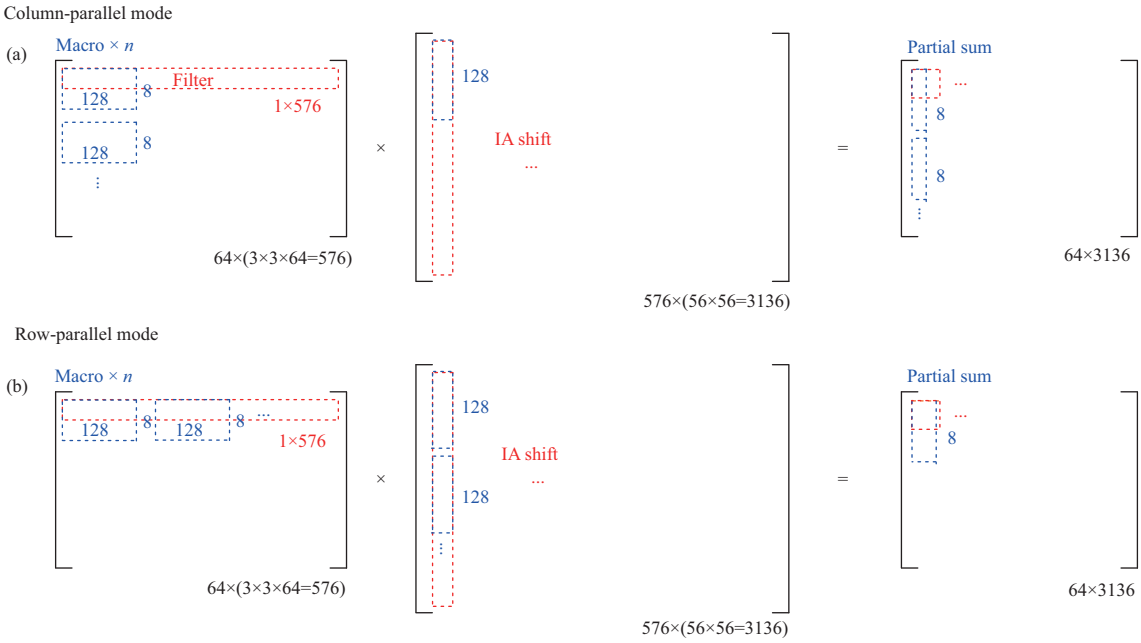


Figure 9 (Color online) (a) Column-parallel data mapping and (b) row-parallel data mapping.

4.1 Future trends of CIM macro

The work of the macro level has received a great deal of attention in recent years. The research has also encountered many contradictions as described above. Higher computational accuracy, higher energy and area efficiency, and diverse network requirements are driving further work at the macro level. Three potential trends are described in this subsection.

4.1.1 *Hybrid computing*

As mentioned above, energy efficiency has always been one of the most important indicators of SRAM-CIM work. Much work has been done to mitigate energy overheads through simulations and approximations. However, the progressively larger data set models and increasingly complex neural network layers have made it challenging to reduce the computational accuracy in exchange for high energy efficiency. How to achieve a good balance between energy efficiency and computational accuracy is the focus of future research in CIM. The hybrid domain computing paradigm can be a possible solution to outperform the existing CIM architecture.

Hybrid domain SRAM computes in memory is a new computing paradigm that combines the benefits of the digital domain and analog domain. Digital domain computing ensures full accuracy, but area efficiency is low, and the energy efficiency improvements have come across a bottleneck. In contrast, analog domain calculations have a limited accuracy and non-accurate calculation errors but are more area efficient and energy efficiency improvements are easier to achieve. An implementable hybrid domain calculation approach is proposed, where the low partial product of a multi-bit multiplication operation is calculated using analog, and the high partial product is calculated using digital. As shown in Figure 10, in a recent study, Tsinghua University and Southeast University jointly proposed a computational paradigm for high level full precision low level approximation calculations. The work at Taiwan Tsing Hua University implemented the low-bit approximation using the current domain paradigm and utilized time domain computation for the exponential addition and subtraction operations of floating-point computation, taking full advantage of the various computational paradigms.

Current hybrid domain computing is still confined to adding other computing paradigms to some scenarios to improve some of the metrics. There is a lack of a systematic analysis of network characteristics to match different network layers, network sparsity, network operator types, network data types, and different computational paradigms. This will be the focus of future research.

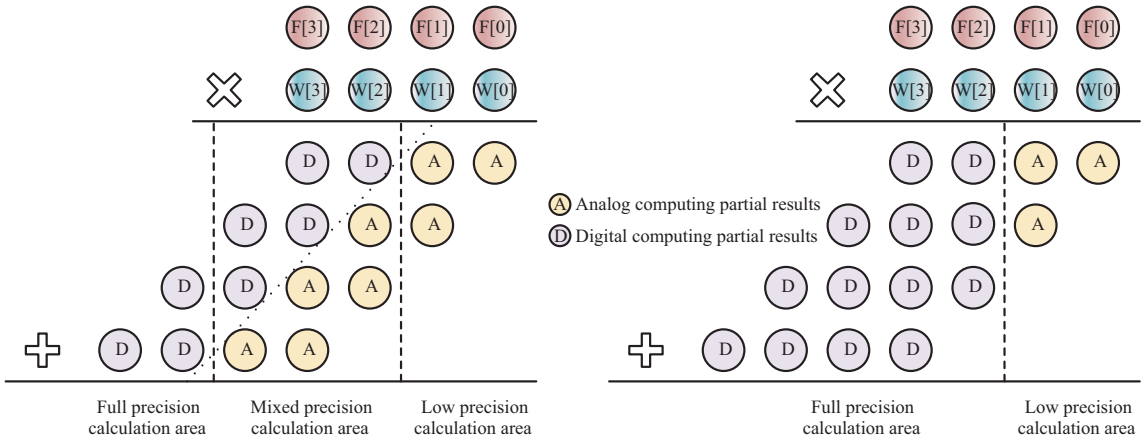


Figure 10 (Color online) Two different types of hybrid domain CIM.

4.1.2 *Precision reconfiguration*

In the first stage, all work surrounding CIM work faced the need to quantify feature and weight data into 8-bit specific point data before mapping it into the SRAM array. As data in the same feature map may have uneven data distribution, the quantization process introduces uncontrollable computational errors, and relies on optimization of the mapping algorithm and a series of redistribution and retraining of the weights, resulting in a lot of off-chip energy overhead. It is imperative to introduce more precise data types.

As AI tasks become more and more complex, floating-point MAC is needed to ensure ideal performances. In computer science, floating-point refers to a numerical representation format that allows a computer to store and manipulate real numbers (numbers with fractional values). A floating-point number is represented in binary form as a sign, a mantissa (also known as a significand), and an exponent. The sign indicates whether the number is positive or negative, the mantissa represents the significant digits of the number, and the exponent indicates the position of the decimal point. The floating-point format is used in most programming languages and is essential for many scientific and engineering calculations that require high precision.

The current floating-point CIM solution is "global floating-point, local fixed-point". Compared with pure floating-point in the past, this solution employs a common index over a range of time, taking advantage of the high dynamic and high accuracy characteristics of floating-point, while making the best of the similarity of the network in the local data. At present, a common exponentiation scheme is widely applied in floating-point operations, which converts common points to fixed points and combines common points with the mantissa which is computed by CIM macro into floating-point result. However, this process creates additional overhead on the circuit. Given this, further research needs to be carried out to reduce the overhead of over floating-point extraction or find alternative floating-point solutions.

4.1.3 *Operator reconfiguration*

Along with the rapid development of AI, more complex application scenarios have come into being, with emerging scenarios such as face recognition and binocular ranging bringing in new operators. The original MAC-based operators for CIM have difficulty supporting the new tasks. Multi-operator support in SRAM-CIM refers to the ability of the memory array to perform multiple arithmetic or logical operations in parallel on the data stored within it. This feature can significantly improve the performance of many computational tasks, such as matrix multiplication, convolutional neural networks, and encryption/decryption.

To implement multi-operator support in SRAM-CIM, the memory array must be equipped with multiple compute cells that can perform arithmetic or logical operations on the data stored in their local memory cells. These computing cells are usually connected via control signals to an external network at the top level, enabling them to be configured and selected for different operations. Current work implements different operators' support through the design of unique local computational cells, which often need to be paired with different data representations, such as binary complement. However, because often only basic logical operations are supported, it is difficult to implement functions like MAC,

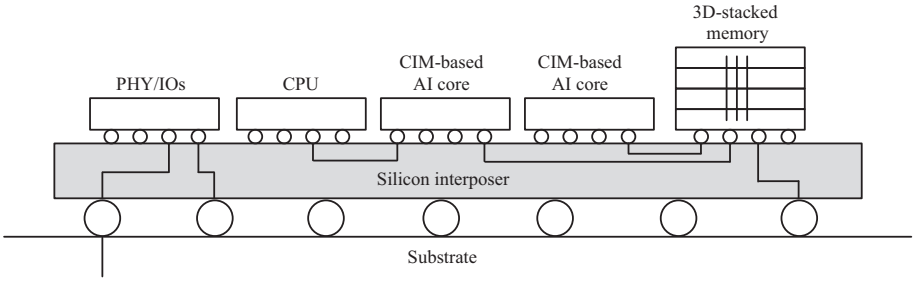


Figure 11 Profile map of Chiplet based AI chip.

making SRAM-CIM not flexible enough compared with the digital logic of the traditional von Neumann architecture. Based on that, configurable multi-operator support would be a potential direction for future research.

Overall, multi-operator support in SRAM-CIM has the potential to revolutionize computing by providing highly efficient and scalable processing-in-memory solutions for a wide range of applications.

4.2 Future trends of CIM microarchitechture

The work at the micro-architectural level is an important step in the evolution of CIM into applications. The contradictions in the current microarchitecture work are also described above. The area overhead of various memories is a key design point for microarchitectures. So how to break the z-constraint of memory area versus system performance has become the focus of research.

4.2.1 *Chiplet*

There are considerable challenges to deploying large-scale networks onto CIM microarchitecture. Typically, high throughput chip designs require larger on-chip areas and advanced technology nodes to provide more computing resources and higher area efficiency. Nevertheless, due to manufacturing limitations such as reticle size, the area of a single die cannot increase indefinitely. Furthermore, a larger chip area will lead to higher chip production costs. On the one hand, the design difficulty of advanced processes is higher, and the yield of chips will decrease as the area increases. On the other hand, the design and verification of large-area chips are more complex, resulting in longer time-to-market (TTM) and higher manufacturing thresholds.

For CIM applications, to meet the system's high-throughput computing requirements, multiple CIM macros are organized to form small AI cores, and many AI cores are integrated on a single die to form a whole AI accelerator. Therefore, the deployment of large-scale networks often requires the design of large-area CIM-based chips, which poses significant challenges for chip designers.

To address this problem, chiplet solutions based on 2.5D and 3D packages have been widely discussed in both academia [82] and industry [83] and are expected to be the development direction of future chips, including CIM-based AI accelerators. Figure 11 shows a CIM-based AI accelerator structure using chiplets, where small chiplets with different functions are integrated together on a 2D silicon interposer. Unlike traditional organic substrates, silicon interposer is manufactured under an advanced process and can provide multiple layers of high-speed and high-density interconnects between chiplets. Thus, the design and manufacturing of chiplets do not affect each other, and only the interconnect matching with the interposer needs to be considered. As the area of each chiplet is reduced, the corresponding production and testing costs are greatly reduced, and the yield can also be well guaranteed. In addition, different chiplets can employ different process nodes. For example, advanced nodes can be used for CPU and AI cores, while some low-speed units and analog circuits can use lower processes. Chip production costs are reduced, and processes that are more suitable for different circuit functions could be used in an SoC.

Chiplets bring great scalability and configurability to SoC design, which is particularly important for CIM-based AI accelerators. During the design of fundamental AI cores, operators' implementation and data flow design are highly correlated with the deployed network tasks. For example, convolution operators for CNNs are implemented in [46], and matrix transposition and multiplication operations for Transformer are implemented in [84]. The efficient deployment of different operators is difficult to balance simultaneously. However, chiplet-based CIM accelerators can simply use AI core chiplets with different functions. Moreover, by increasing or decreasing the number of chiplets used, different computational

requirements for networks of different scales can be met, thereby achieving configurability and scalability of AI accelerators at the system level.

Currently, there are still some obstacles to building chiplet-based CIM accelerators. In response, a uniform chiplet interconnect standard needs to be formulated, and CIM-based AI cores suitable for chiplet operating modes need to be developed.

4.2.2 *Sparsity optimization*

Different layers of a neural network have different degrees of fitness for CIM. Those layers that can be mapped to MVM (matrix-vector multiplication) operations are structurally friendly to CIM. However, for some of the layers in the neural network, the sparsity of their inputs and weights imposes an additional overhead on CIM. For the sparsity of the weights, zero-weight data does not generate currents but non-zero values in the same block still need to be computed, which imposes an additional overhead on the quantization circuit. For the sparsity of the input activation (feature-map), the additional control circuitry is required if the configuration is required, as the regular CIM can only activate continuous multiple rows rather than random-distributed non-zero rows.

At the same time the advantages offered by sparsity optimization are very attractive. Energy efficiency improvements based on sparsity have been demonstrated in previous digital ASIC (application specific integrated circuit) architectures. In the CIM microarchitecture weight data can be compressed by a factor of 13–71 by means of weight pruning techniques. Skipping zero operations saves energy and execution time. So sparsity optimization significantly improves the macro usage efficiency of the CIM micro-architecture, taking the system energy efficiency to new heights.

There are two main approaches to optimizing sparsity at the microarchitectural level. The first is to perform sparsity detection on the input. Microarchitecture uses a dynamic sparsity monitoring system to sense the sparsity of the input and configures different CIM execution modes when the input is sparse. For example, microarchitecture activates more rows to maximize resource utilization. This approach has no specific requirements for the network, but the dynamic detection introduces additional circuit overhead. The second approach is the sparsity optimization of weights, where the microarchitecture can also sense whether the weight data is all zeroes downwards by means of a sparsity index for the stored weights in the macro, and change the input policy by feeding back an index to the input, and the corresponding ADC of the sparse block can also be powered down to save power. This approach has high adaptation requirements for the network, but the additional overhead of the circuit is less than in the first approach.

Sparsity optimization is an important research direction in the design of CIM microarchitectures. Better adaptation of sparsity optimization for neural networks and better control of additional circuit overheads will drive more efficient microarchitectural systems.

5 Conclusion

SRAM-CIM is a future-oriented solution that can break memory and power walls. This study reviews the computational principles of SRAM-CIM macros from the perspective of different computational paradigms and analyzes the problems faced by the current technical architecture of the time, current, charge, and digital computing paradigms. This paper also studied the SRAM-CIM macro recently validated by silicon and analyzed its trade-offs with other indicators, focusing on energy efficiency and bottlenecks in the further development of SRAM-CIM. In addition, this paper has analyzed the link between memory size, data reuse, and system performance in the SRAM-CIM microarchitecture. To address the contradiction between energy efficiency, computational accuracy, area efficiency, and operator needs in CIM macro, three future trends are introduced. Chiplet and sparsity optimization will be used to improve the efficiency of microarchitecture.

Acknowledgements This work was supported by National Key R&D Program of China (Grant No. 2022ZD0118902) and National Natural Science Foundation of China (Grant Nos. 92264203, 62204036).

References

1 Chang L, Li C, Zhang Z, et al. Energy-efficient computing-in-memory architecture for AI processor: device, circuit, architecture perspective. Sci China Inf Sci, 2021, 64: 160403
2 Cheng C, Tiw P J, Cai Y, et al. In-memory computing with emerging nonvolatile memory devices. Sci China Inf Sci, 2021, 64: 221402

3 Zhang W, Gao B, Yao P, et al. Array-level boosting method with spatial extended allocation to improve the accuracy of memristor based computing-in-memory chips. Sci China Inf Sci, 2021, 64: 160406
4 Zou X, Xu S, Chen X, et al. Breaking the von Neumann bottleneck: architecture-level processing-in-memory technology. Sci China Inf Sci, 2021, 64: 160404
5 Jhang C J, Xue C X, Hung J M, et al. Challenges and trends of SRAM-based computing-in-memory for AI edge devices. IEEE Trans Circ Syst I, 2021, 68: 1773–1786
6 Si X, Zhou Y L, Yang J, et al. Challenge and trend of SRAM based computation-in-memory circuits for AI edge devices. In: Proceedings of the 14th International Conference on ASIC (ASICON), 2021
7 Wang Y F, Zhou Y L, Wang B, et al. Design challenges and methodology of high-performance SRAM-based compute-in-memory for AI edge devices. In: Proceedings of International Conference on UK-China Emerging Technologies (UCET), 2021
8 Xiong T Z, Zhou Y L, Kong Y Y, et al. Design methodology towards high-precision SRAM based computation-in-memory for AI edge devices. In: Proceedings of the 18th International SoC Design Conference (ISOCC), 2021
9 Dong F Y, Si X, Chang M F. Design methodology and trends of SRAM-based compute-in-memory circuits. In: Proceedings of the 16th International Conference on Solid-State & Integrated Circuit Technology (ICSICT), 2022
10 Chang M F, Lin C C, Lee A, et al. 17.5 A 3T1R nonvolatile TCAM using MLC ReRAM with Sub-1ns search time. In: Proceedings of IEEE International Solid-State Circuits Conference, 2015
11 Khwa W S, Chang M F, Wu J Y, et al. 7.3 A resistance-drift compensation scheme to reduce MLC PCM raw BER by over 100× for storage-class memory applications. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2016
12 Lin C C, Hung J Y, Lin W Z, et al. 7.4 A 256b-wordlength ReRAM-based TCAM with 1ns search-time and 14× improvement in wordlength-energyefficiency-density product using 2.5T1R cell. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2016
13 Xue C X, Chen W H, Liu J S, et al. 24.1 A 1Mb multibit ReRAM computing-in-memory macro with 14.6ns parallel MAC computing time for CNN based AI edge processors. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2019
14 Chang T C, Chiu Y C, Lee C Y, et al. 13.4 A 22nm 1Mb 1024b-read and near-memory-computing dual-mode STT-MRAM macro with 42.6GB/s read bandwidth for security-aware mobile devices. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2020
15 Liu Q, Gao B, Yao P, et al. 33.2 A fully integrated analog ReRAM based 78.4TOPS/W compute-in-memory chip with fully parallel MAC computing. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2020
16 Xue C X, Huang T Y, Liu J S, et al. 15.4 A 22nm 2Mb ReRAM compute-in-memory macro with 121-28TOPS/W for multibit MAC computing for tiny AI edge devices. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2020
17 Xue C X, Hung J M, Kao H Y, et al. 16.1 A 22nm 4Mb 8b-precision ReRAM computing-in-memory macro with 11.91 to 195.7TOPS/W for tiny AI edge devices. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2021
18 Yoon J H, Chang M, Khwa W S, et al. 29.1 A 40nm 64Kb 56.67TOPS/W read-disturb-tolerant compute-in-memory/digital RRAM macro with active-feedback-based read and in-situ write verification. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2021
19 Chang M, Spetalnick S D, Crafton B, et al. A 40nm 60.64TOPS/W ECC-capable compute-in-memory/digital 2.25MB/768KB RRAM/SRAM system with embedded cortex M3 microprocessor for edge recommendation systems. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2022
20 Chiu Y C, Yang C S, Teng S H, et al. A 22nm 4Mb STT-MRAM data-encrypted near-memory computation macro with a 192GB/s read-and-decryption bandwidth and 25.1-55.1TOPS/W 8b MAC for AI operations. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2022
21 Hu H W, Wang W C, Chen C K, et al. A 512Gb in-memory-computing 3D-NAND flash supporting similar-vector-matching operations on edge-AI devices. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2022
22 Hung J M, Huang Y H, Huang S P, et al. An 8-Mb DC-current-free binary-to-8b precision ReRAM nonvolatile computing-in-memory macro using time-space-readout with 1286.4-21.6TOPS/W for edge-AI devices. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2022
23 Khwa W S, Chiu Y C, Jhang C J, et al. A 40-nm, 2M-cell, 8b-precision, hybrid SLC-MLC PCM computing-in-memory macro with 20.5-65.0TOPS/W for tiny-Al edge devices. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2022
24 Spetalnick S D, Chang M, Crafton B, et al. A 40nm 64kb 26.56TOPS/W 2.37Mb/mm^2 RRAM binary/compute-in-memory macro with 4.23× improvement in density and >75 use of sensing dynamic range. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2022
25 Everson L R, Liu M, Pande N, et al. A 104.8TOPS/W one-shot time-based neuromorphic chip employing dynamic threshold error correction in 65nm. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2022
26 Mohammed M U, Chowdhury M H. Reliability and energy efficiency of the tunneling transistor-based 6T SRAM cell in sub-10 nm domain. IEEE Trans Circ Syst II, 2018, 65: 1829–1833
27 Everson L R, Liu M, Pande N, et al. An energy-efficient one-shot time-based neural network accelerator employing dynamic threshold error correction in 65 nm. IEEE J Solid-State Circ, 2019, 54: 2777–2785
28 Huynh K, Saltin J, Han J W, et al. Study of layout dependent radiation hardness of FinFET SRAM using full domain 3D TCAD simulation. In: Proceedings of IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference, 2019
29 Sayal A, Fathima S, Nibhanupudi S S T, et al. 14.4 all-digital time-domain CNN engine using bidirectional memory delay lines for energy-efficient edge computing. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2019
30 Wang T, Shan W W. An energy-efficient in-memory BNN architecture with time-domain analog and digital mixed-signal processing. In: Proceedings of IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH), 2019
31 Yang J, Kong Y Y, Wang Z, et al. 24.4 sandwich-RAM: an energy-efficient in-memory BWN architecture with pulse-width modulation. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2019
32 Agrawal A, Kosta A, Kodge S, et al. CASH-RAM: enabling in-memory computations for edge inference using charge accumulation and sharing in standard 8T-SRAM arrays. IEEE J Emerg Sel Top Circ Syst, 2020, 10: 295–305
33 He Y X, Choi M, Kim K K, et al. A time-domain computing-in-memory micro using ring oscillator. In: Proceedings of the

18th International SoC Design Conference (ISOCC), 2021
34 Lin C S, Tsai F C, Su J W, et al. A 48 TOPS and 20943 TOPS/W 512kb computation-in-SRAM macro for highly reconfigurable ternary CNN acceleration. In: Proceedings of IEEE Asian Solid-State Circuits Conference (A-SSCC), 2021
35 Song J, Wang Y, Guo M, et al. TD-SRAM: time-domain-based in-memory computing macro for binary neural networks. IEEE Trans Circ Syst I, 2021, 68: 3377–3387
36 Kong Y, Chen X, Si X, et al. Evaluation platform of time-domain computing-in-memory circuits. IEEE Trans Circ Syst, 2023, 70: 1174–1178
37 Park H, Lee K, Park J. A 10T SRAM compute-in-memory macro with analog MAC operation and time domain conversion. In: Proceedings of the 4th International Conference on Artificial Intelligence Circuits and Systems (AICAS), 2022
38 Wu P C, Su J W, Chung Y L, et al. A 28nm 1Mb time-domain computing-in-memory 6T-SRAM macro with a 6.6ns latency, 1241GOPS and 37.01TOPS/W for 8b-MAC operations for edge-AI devices. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2022
39 Wang Y S, Liu L B, Yin S Y, et al. Hierarchical representation of on-chip context to reduce reconfiguration time and implementation area for coarse-grained reconfigurable architecture. Sci China Inf Sci, 2013, 56: 112401
40 Miyashita D, Kousai S, Suzuki T, et al. A neuromorphic chip optimized for deep learning and CMOS technology with time-domain analog and digital mixed-signal processing. IEEE J Solid-State Circ, 2017, 52: 2679–2689
41 Biswas A, Chandrakasan A P. Conv-RAM: an energy-efficient SRAM with embedded convolution computation for low-power CNN-based machine learning applications. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2018
42 Zhang J, Wang Z, Verma N. In-memory computation of a machine-learning classifier in a standard 6T SRAM array. IEEE J Solid-State Cir, 2017, 52: 915–924
43 Khwa W S, Chen J J, Li J F, et al. A 65nm 4Kb algorithm-dependent computing-in-memory SRAM unit-macro with 2.3ns and 55.8TOPS/W fully parallel product-sum operation for binary DNN edge processors. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2018
44 Si X, Chen J J, Tu Y N, et al. 24.5 A twin-8T SRAM computation-in-memory macro for multiple-bit CNN-based machine learning. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2019
45 Choi I, Choi E J, Yi D, et al. An SRAM-based hybrid computation-in-memory macro using current-reused differential CCO. IEEE J Emerg Sel Top Circ Syst, 2022, 12: 536–546
46 Si X, Tu Y N, Huang W H, et al. 15.5 A 28nm 64Kb 6T SRAM computing-in-memory macro with 8b MAC operation for AI edge chips. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2020
47 Su J W, Si X, Chou Y C, et al. 15.2 A 28nm 64Kb inference-training two-way transpose multibit 6T SRAM compute-in-memory macro for AI edge chips. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2020
48 Xue C X, Huang T Y, Liu J S, et al. 15.4 A 22nm 2Mb ReRAM Compute-in-Memory Macro with 121-28TOPS/W for Multibit MAC Computing for Tiny AI Edge Devices. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2020
49 Yin S, Jiang Z, Seo J, et al. XNOR-SRAM: in-memory computing SRAM macro for binary/ternary deep neural networks. IEEE J Solid-State Circ, 2020, 55: 1733–1743
50 Wang Y, Zou Z, Zheng L. Design framework for SRAM-based computing-in-memory edge CNN accelerators. In: Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS), 2021
51 Xu T, Li S, Su F, et al. A current domain computing-in-memory SRAM macro with hybrid IAF-SAR ADC for signal margin enhancement. In: Proceedings of IEEE International Conference on Integrated Circuits, Technologies and Applications (ICTA), 2022. 119–120
52 Yue J S, Feng X Y, He Y F, et al. 15.2 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. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2021
53 Song J, Tang X, Luo H, et al. A calibration-free 15-level/cell eDRAM computing-in-memory macro with 3T1C current-programmed dynamic-cascoded MLC achieving 233-to-304-TOPS/W 4b MAC. In: Proceedings of IEEE Custom Integrated Circuits Conference (CICC), 2023
54 Peng S Y, Liu I C, Wu Y H, et al. An SRAM-based reconfigurable cognitive computation matrix for sensor edge applications. IEEE J Solid State Circ, 2023. doi: 10.1109/JSSC.2023.3303910
55 Yin G, Cai Y, Wu J, et al. Enabling lower-power charge-domain nonvolatile in-memory computing with ferroelectric FETs. IEEE Trans Circ Syst, 2021, 68: 2262–2266
56 Song J, Tang X, Luo H, et al. Spike-CIM: a 290TOPS/W spike-encoding sparsity-adaptive computing-in-memory macro with differential charge-domain integrate-and-fire. In: Proceedings of IEEE Asian Solid-State Circuits Conference (A-SSCC), 2022
57 Gonugondla S K, Kang M, Shanbhag N. A 42pJ/decision 3.12TOPS/W robust in-memory machine learning classifier with on-chip training. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2018
58 Valavi H, Ramadge P J, Nestler E, et al. A mixed-signal binarized convolutional-neural-network accelerator integrating dense weight storage and multiplication for reduced data movement. In: Proceedings of IEEE Symposium on VLSI Circuits, 2018
59 Jiang Z W, Yin S H, Seo J S, et al. C3SRAM: in-memory-computing SRAM macro based on capacitive-coupling computing. IEEE Solid-State Circ Lett, 2019, 2: 131–134
60 Kim H, Chen Q, Kim B. A 16K SRAM-based mixed-signal in-memory computing macro featuring voltage-mode accumulator and row-by-row ADC. In: Proceedings of IEEE Asian Solid-State Circuits Conference (A-SSCC), 2019
61 Valavi H, Ramadge P J, Nestler E, et al. A 64-tile 2.4-Mb in-memory-computing CNN accelerator employing charge-domain compute. IEEE J Solid-State Circ, 2019, 54: 1789–1799
62 Dong Q, Sinangil M E, Erbagci B, et al. 15.3 A 351TOPS/W and 372.4GOPS compute-in-memory SRAM macro in 7nm FinFET CMOS for machine-learning applications. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2020
63 Chen Z Y, Chen X, Gu J. 15.3 A 65nm 3T dynamic analog RAM-based computing-in-memory macro and CNN accelerator with retention enhancement, adaptive analog sparsity and 44TOPS/W system energy efficiency. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2021
64 Chen Z Y, Yu Z H, Jin Q, et al. CAP-RAM: a charge-domain in-memory computing 6T-SRAM for accurate and precision-programmable CNN inference. IEEE J Solid-State Circ, 2021, 56: 1924–1935
65 Jia H Y, Ozatay M, Tang Y Q, et al. 15.1 A programmable neural-network inference accelerator based on scalable in-memory computing. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2021

66 Lee E, Han T, Seo D, et al. A charge-domain scalable-weight in-memory computing macro with dual-SRAM architecture for precision-scalable DNN accelerators. IEEE Trans Circ Syst I, 2021, 68: 3305–3316
67 Lee J, Valavi H, Tang Y, et al. Fully row/column-parallel in-memory computing SRAM macro employing capacitor-based mixed-signal computation with 5-b inputs. In: Proceedings of Symposium on VLSI Technology, 2021
68 Song J H, Wang Y, Tang X Y, et al. A 16Kb transpose 6T SRAM in-memory-computing macro based on robust charge-domain computing. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2021
69 Su J W, Chou Y C, Liu R, et al. 16.3 A 28nm 384kb 6T-SRAM computation-in-memory macro with 8b precision for AI edge chips. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2021
70 Bharti P K, Jain S, Pillai K R, et al. Compute-in-memory using 6T SRAM for a wide variety of workloads. In: Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS), 2022
71 Chih Y D, Lee P H, Fujiwara H, et al. 16.4 An 89TOPS/W and 16.3TOPS/mm2 all-digital SRAM-based full-precision compute-in memory macro in 22nm for machine-learning edge applications. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2021
72 Yan B, Hsu J L, Yu P C, et al. A 1.041-Mb/mm^2 27.38-TOPS/W signed-INT8 dynamic-logic-based ADC-less SRAM compute-in-memory macro in 28nm with reconfigurable bitwise operation for AI and embedded applications. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2022
73 Yang J X, Kong Y Y, Zhang Z, et al. TIMAQ: a time-domain computing-in-memory-based processor using predictable decomposed convolution for arbitrary quantized DNNs. IEEE J Solid-State Circ, 2021, 56: 3021–3038
74 Wang B, Xue C, Feng Z Y, et al. A 28nm horizontal-weight-shift and vertical-feature-shift-based separate-WL 6T-SRAM computation-in-memory unit-macro for edge depthwise neural-networks. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2023
75 Hsieh S, Wei C, Xue C, et al. 7.6 A 70.85-86.27TOPS/W PVT-insensitive 8b word-wise ACIM with post-processing relaxation. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2023
76 Yao C Y, Wu T Y, Liang H C, et al. A fully bit-flexible computation in memory macro using multi-functional computing bit cell and embedded input sparsity sensing. IEEE J Solid-State Circ, 2023, 58: 1487–1495
77 Guo A, Si X, Chen X, et al. A 28nm 64-kb 31.6-TFLOPS/W digital-domain floating-point- computing-unit and double-bit 6T-SRAM computing-in-memory macro for floating-point CNNs. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2023
78 Chen P Y, Wu M, Zhao W T, et al. 7.8 A 22nm delta-sigma computing-in-memory (delta sigma CIM) SRAM macro with near-zero-mean outputs and LSB-first ADCs achieving 21.38TOPS/W for 8b-MAC edge AI processing. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2023
79 Wang H C, Liu R Z, Dorrance R, et al. A 32.2 TOPS/W SRAM compute-in-memory macro employing a linear 8-bit C-2C ladder for charge domain computation in 22nm for edge inference. In: Proceedings of IEEE Symposium on VLSI Technology and Circuits, 2022
80 Park J S, Jang J W, Lee H, et al. 9.5 A 6K-MAC feature-map-sparsity-aware neural processing unit in 5nm flagship mobile SoC. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2021
81 Yue Z H, Wang Y, Wang H Z, et al. 7.7 CV-CIM: a 28nm XOR-derived similarity-aware computation-in-memory for cost-volume construction. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2023
82 Vivet P, Guthmuller E, Thonnart Y, et al. IntAct: a 96-core processor with six chiplets 3D-stacked on an active interposer with distributed interconnects and integrated power management. IEEE J Solid-State Circ, 2021, 56: 79–97
83 Gomes W, Koker A, Stover P, et al. Ponte Vecchio: a multi-tile 3D stacked processor for exascale computing. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2022
84 Tu F B, Wang Y Q, Wu Z H, 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. In: Proceedings of IEEE International Solid-State Circuits Conference (ISSCC), 2022