ADVANCED INTELLIGENT SYSTEMS Open Access

www.advintellsyst.com

Optimal Weight-Splitting in Resistive Random Access Memory-Based Computing-in-Memory Macros

*Choongseok Song, Jeeson Kim, and Doo Seok Jeong**

Computing-in-memory (CIM) is considered a feasible solution to the acceleration of multiply-accumulate (MAC) operations at low power. The key to CIM is parallel MAC operations in the memory domain, and thus reductions in power consumption and memory-access latency. Resistive random access memory (RRAM) can be a good candidate for the memory for CIM given its data nonvolatility, high data density, low-latency read-out, multilevel representation, and inherent current accumulation capability. Particularly, the last two attributes offer analog MAC operations in parallel in the memory domain. However, the fully analog MAC operation scheme causes significant power and area overheads for its peripheral circuits, particularly, analog-to-digital converters. To compensate for these downsides using digital processing, a method for sub-array-wise partial MAC operations over weight-resistors that are optimally split to minimize power and area overheads for the peripheral circuits is proposed. The simulations performed highlight the optimal sub-array of $4 \times w/2$ in size. That is, weight-splitting such that a single w-bit weight is represented by $w/2$ RRAM cells, i.e., 2-bit for each cell. For 8-bit weights, the figure of merit (FOM) for this optimal case reaches $\approx 28.3\times$ FOM for the case of no weight-splitting.

1. Introduction

Deep learning (DL) prevalently applies to many tasks that are used to be done using a set of instructions. The improvement of DL capability is desired, which requires the deep neural network (DNN) to be deeper and larger. The wide use of DL with deep and large DNNs causes an immense workload for hardware, which is expected to continue onward. To support this trend, new hardware that accelerates major DL operations with reduced power consumption is strongly demanded. The current mainstream hardware for DL is general-purpose graphics processing units (GPGPUs), which enable parallel multiply-accumulate (MAC) operations—a major operational workload—but consume enormous power. DL accelerators aim to achieve high performance and power efficiency beyond GPGPUs.

Several DL accelerators (popularly, referred to as neural processing units) have already been commercialized.[1,2] A next-generation (but not far) DL acceleration is considered to be based on the memory-centric architecture that minimizes data movement by computing data near or in memory domains, which is known to consume an immense amount of power.[3] Near-data processing (NDP) realizes parallel floating-point MAC operations in the vicinity of the memory domain, minimizing data movement.[4,5] However, this architecture also suffers from the notorious memory wall issue[6] due to large latency in access to random access memory (RAM). Additionally, NDP leverages its advantages only for memory-bound models, e.g., fully connected networks and recurrent neural networks, where one weight is used for one operation, unlike convolutional neural networks in which one weight is used for many operations.

A workaround solution is to merge processing and memory domains into a single domain in which parallel MAC operations are executed in an analog manner.[7–18] This strategy realizes in-place MAC operations that allow memory access and operations to occur simultaneously in the same domain. We refer to this strategy as analog computing-in-memory (aCIM). Its feasibility has been demonstrated with various types of memory such as volatile memory, e.g., dynamic RAM,[7] static RAM,[8–10] and nonvolatile memory, e.g., magnetic RAM,[11] resistive RAM (RRAM).[12,13,15–17] Among various embedded memories for aCIM, resistance-based nonvolatile RAMs have been attracting large attention because of their feasible analog MAC operations in parallel based on Kirchhoff's current law (KCL). Particularly, RRAM offers multilevel data representations, so that a single RRAM cell can represent a multi-bit weight, which allows parallel fixed-point MAC operations at reduced space and computational complexities. However, the larger the parallelism and the number of levels of a single cell, the more likely a higher bit-resolution is required for the analog-to-digital converters (ADCs) to avoid data loss, which causes prohibitive power consumption and area overhead.

In this regard, it may be necessary to limit the number of levels of a single cell and the parallelism such that: i) multiple

C. Song, J. Kim, D. S. Jeong
Division of Materials Science and Engineering
Hanyang University
222 Wangsimni-ro, Seongdong-gu, Seoul 04763, Republic of Korea
E-mail: dooseokj@hanyang.ac.kr

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/aisy.202200289.

DOI: 10.1002/aisy.202200289

cells (rather than a single cell) represent a single weight and ii) the partial results from parallel operations are processed in a digital processing domain. However, this costs operational latency. Considering the power- and area-efficiency of operations (PAE) as a figure of merit, there may be optimal parallelism and number of levels of a single cell for the largest PAE given the trade-off between the operational latency and power/area-efficiency. In this work, we attempt to find the optimal operational strategy to maximize PAE. The primary contributions of our work are as follows: 1) We provide a strategy for optimal weight-splitting and mapping onto an RRAM array to maximize PAE for mixed-signal-based CIM (mCIM). 2) We validate the strategy for various weight and activation resolutions (w and a, respectively) and compare it with naive weight-mapping methods. 3) We provide the power and area overheads for a successive-approximation-register ADC (SAR-ADC) and shift-and-add unit (S&A) with various data resolutions, which were designed using the Cadence GPDK 45 nm technology.

Note that the GPDK 45 nm technology is comparable with the technologies for: 1) the state-of-the-art RRAM-based CIM macros, e.g., TSMC 65 nm for ref. [19] 55 nm technology for ref. [20], Winbond 90 nm for ref. [15], and TSMC 22 nm for ref. [21], and 2) RRAM-based CIM simulators, e.g., CACTI 32 nm for refs. [12,13], FreePDK 45 nm for ref. [22], TSMC 65 nm PDK for ref. [23], and TSMC 40 nm PDK (scaled to 32 nm) for ref. [24].

The mCIM macro (and thus operational scheme) considered in this work is similar to previous designs.[12,13] Additionally, there exist some efforts to reduce the ADC resolution in mCIM macros to alleviate the area and power overheads. For instance, Anirban Nag et al. proposed a method to reduce the ADC resolution by sacrificing the operational precision to a marginal degree.[13] Compared with these works to the same end, the novelties of our method include the following: 1) Our method considers the number of splits of a single weight as a parameter subject to optimization to maximize the figure of merit, unlike the previous methods. 2) Our method also considers the operational parallelism (number of word (row) lines addressed at a cycle) as an optimizable parameter, unlike the previous methods in which all word lines are addressed at a cycle to maximize the parallelism. 3) Our method can maximize the figure of merit given circuit parameters without losing any operational precision unlike the previous methods.

The rest of this article is organized as follows. Section 2 briefly addresses the basics and challenges of RRAM-based CIM. Section 3 is dedicated to: 1) the basics of RRAM-based mCIM, 2) power and area overheads for mCIM peripheral circuits, which strongly rely on the data resolution, and 3) operational latency for mCIM. Section 4 addresses the proposed strategy to optimally partition the RRAM array to maximize PAE and the simulation results validating the strategy for various weight and activation resolutions. Finally, this work is concluded in Section 5.

2. Preliminaries

KCL applied to RRAM arrays realizes parallel analog MAC operations with a-bit activations and w-bit weights in fixed-point data formats. A common type of unit RRAM cell is 1T1R, i.e., one-transistor and one-resistor, where the transistor serves as an active selector for random access.[25,26] Passive unit RRAM cells are also considered, e.g., 1S1R (one-passive selector and one-resistor)[27,28] and 1 R (one self-rectifying resistor).[18,29,30] For simplicity, we illustrate RRAM arrays as arrays of passive resistors hereafter. Additionally, we refer to the lines to which activations are applied as row lines and the lines that output the current sums as column lines.

Consider an $M \times N$ RRAM array, i.e., M row and N column lines. For parallel analog MAC operations, one column line is addressed at one cycle to add up the currents through the RRAM cells that share the same column line. The situation is illustrated in **Figure 1**a. The a-bit activations are converted to analog voltage signals by digital-to-analog converters (DACs) and subsequently applied to the row lines simultaneously. The

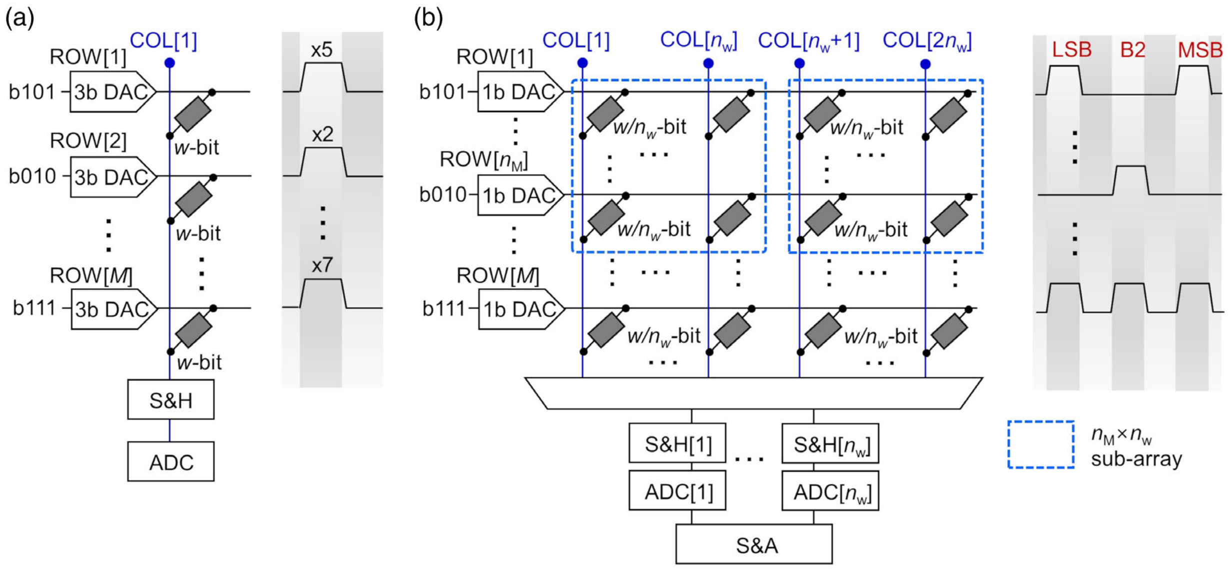

Figure 1. Schematics of: a) the fully analog multiply-accumulate (MAC) operation scheme and b) analog/digital MAC operation scheme.

aggregate current through the parallel RRAM cells is converted to an analog voltage signal, which is subsequently latched by a sample-and-hold unit (S&H) and eventually converted to a digital signal by an ADC. To this end, each RRAM cell needs to represent: 1) clearly distinguishable 2^w conductance levels, 2) perfect linearity in current–voltage (I–V) characteristics for each conductance level, 3) excelent memory retention, 4) negligible read-disturbance, and so forth.

A daunting challenge is the desired precision (bit resolution) of ADCs to avoid any data loss. For an $M \times N$ RRAM array, M parallel MAC operations of a-bit activations and w-bit weights need minimal output bit width (b) to avoid data loss.

$$\begin{aligned} b \ &= \log_2 M + w + a \quad \text{if } w \text{ and } a > 1 \\ &= \log_2 M + w + a - 1 \quad \text{otherwise} \end{aligned} \tag{1}$$

For instance, when $M = 128$ and $w = a = 8$, the desired bit width b already reaches 23-bit, which leads to prohibitive power and area overheads, and latency of the ADC.

To date, available ADC designs are diverse ;[31] they largely differ in power and area overheads, and latency. SAR-ADCs highlight high power efficiency at a mediocre resolution,[31–34] so they are frequently deployed in RRAM-based CIM macros.[12,13,24] The SAR-ADC consists of four components: capacitive DAC, reference buffer, comparator, and SAR logic.[34] The large power and area overheads of the SAR-ADC at high resolutions arise mainly from the capacitive DAC, which exponentially scales with data resolution.[33] The overheads for the SAR-ADC will be elaborated in Section 3.2.

3. Mixed-Signal-Based CIM

3.1. Overview

The mCIM considered in this work significantly improves the power- and area-efficiency, and reliability by executing a considerable amount of sub-operations in the digital processing domain. To this end, it differs from the aforementioned aCIM such that: 1) Bit-serial activations are applied to the row lines to reduce the area and power overheads of the DACs. 2) Multiple (n_w) RRAM cells collectively represent a single w-bit weight, which are addressed simultaneously. 3) M row lines are partitioned into M/n_M groups (each group is of n_M lines), and the RRAM cells in the same group are simultaneously addressed.

A schematic of the mCIM is illustrated in Figure 1b. The bit-serial representation allows each a-bit activation $A[k]$ (for the kth row line) to be encoded as a bit stream over a cycles

$$A[k] = \sum_{t=1}^{a} A_t[k] 2^{t-1}, \text{ where } A_t[k] \in \{0, 1\} \tag{2}$$

and thus fully supported by a 1-bit DAC. This largely alleviates the power and area overheads for the DACs compared with the aCIM but at the cost of latency (a cycles). Additionally, the linearity in the I–V relationship of the RRAM cell is no longer required given that only constant read-out voltage pulses are used.

Given the use of n_w RRAM cells to represent a single w-bit weight $W[k]$ at the kth row line, each RRAM cell represents a w/n_w-bit partial weight $W_i[k]$ such that

$$W[k] = \sum_{i=1}^{n_w} W_i[k] 2^{w/n_w(i-1)} \tag{3}$$

This weight-partitioning improves the operational reliability because it can enlarge the multilevel conductance read-out margins. We partition the M row lines into M/n_M groups (each of which is of size n_M) and address each group (rather than the whole M row lines) at one cycle. Owing to the bit-serial activation representation and weight- and row line-partitioning, the desired resolution b of the ADCs is significantly reduced to

$$b = \log_2 n_M + w/n_\mathrm{w} \tag{4}$$

We consider n_w parallel SAR-ADCs to simultaneously convert the current sum from the n_w column lines.

Using Equation (2) and (3), the product $W[k]A[k]$ is given by

$$W[k]A[k] = \sum_{t=1}^{a} \left(\sum_{i=1}^{n_w} W_i[k] A_t[k] 2^{w/n_w(i-1)} \right) 2^{t-1} \tag{5}$$

Addressing each partition including $n_M \times n_w$, RRAM cells performs n_M MAC operations in parallel, yielding a partial sum S_p (for $k = 1, \ \cdots, n_M$)

$$\begin{aligned} S_p = \ & \textstyle\sum_{k=1}^{n_M} W[k]A[k] \\ = \ & \underbrace{\sum_{t=1}^{a}}_{\text{digital}} \overbrace{\left[\underbrace{\sum_{i=1}^{n_w}}_{\text{digital}} \left(\underbrace{\sum_{k=1}^{n_M}}_{\text{analog}} W_i[k] A_t[k] \right) 2^{w/n_w(i-1)} \right]}^{\text{parallel ops}} 2^{t-1} \end{aligned} \tag{6}$$

The two inner summations in Equation (6) are performed in parallel at one cycle as follows. The innermost summation $\sum_k W_i[k]A_t[k]$ is performed by adding up the currents through the RRAM cells for the row lines ($k = 1, \ \cdots, n_M$) and a given column i (analog sum). The summation over columns i is also performed in parallel by shifting the summation for each column by $w/n_w(i-1)$ bits and subsequently adding them (digital sum). To this end, we deploy n_w ADCs and one S&A.

The sum S_p in Equation (6) is eventually obtained by repeating the aforementioned summations over each bit t in the activations, and shifting the summation results by $t-1$ bits using the same S&A and accumulating them (digital sum). Similar operational schemes were proposed in refs. [12,13] for the same purpose as our scheme. Yet, ours is distinguishable from these schemes given that the operational parallelism n_M is subject to optimization in terms of area- and power efficiency unlike the previous methods, where n_M is always set to M.

3.2. Area and Power Overheads of Peripheral Circuits

As shown in Figure 1b, the peripheral circuits include M 1-bit DACs, n_w sets of an S&H and SAR-ADC, and one S&A. Given that a SAR-ADC includes an S&H, we do not consider the area and power overheads of an S&H separately. Because we aim to find the optimal memory partition size ($n_M \times n_w$)

for a given weight-resolution w, we only address the peripheral circuits whose power and area scale with n_M and n_w, e.g., SAR-ADC and S&A. Note that the 1-bit DACs are independent of the partition size. We designed the peripheral circuits using the Cadence GPDK 45 nm technology with a 1 V supply voltage. The clock frequency was set to 100 MHz.

3.2.1. *SAR-ADC*

A SAR-ADC converts an analog voltage signal to its binary number by iteratively comparing the analog signal with the reference signals that are generated by a capacitive DAC and SAR logic. A SAR-ADC consists of a binary-weighted capacitive DAC, SAR logic, and comparator (**Figure 2**a). The capacitive DAC utilizes capacitive voltage division using a set of capacitors. A unit capacitive DAC with b-bit resolution uses b capacitors, $C_1, \ldots, C_b$, each of which represents a digit of a given b-bit binary number such that $C_i = 2^{i-1}C_0$. A unit capacitor is denoted by C_0 (30 fF), which is also included in the capacitive DAC, so the capacitive DAC uses $(b+1)$ capacitors in aggregate. The $(b+1)$ capacitors mostly dictate the circuit area and power consumption of the capacitive DAC. The capacitance in total C_{tot} is given by

$$C_{\text{tot}} = C_0 + \sum_{i=1}^{b} 2^{i-1}C_0 = 2^b C_0 \tag{7}$$

A timing diagram for a SAR-ADC ($b = 4$) is illustrated in Figure 2b. Because each digit of the b-bit output is iteratively output, the conversion latency T_{ADC} is proportional to the resolution b such that

$$T_{\text{ADC}} = (b+1)f_{\text{clk}}^{-1} \tag{8}$$

where f_{clk} denotes clock frequency. The latency includes one sampling cycle as shown in Figure 2b.

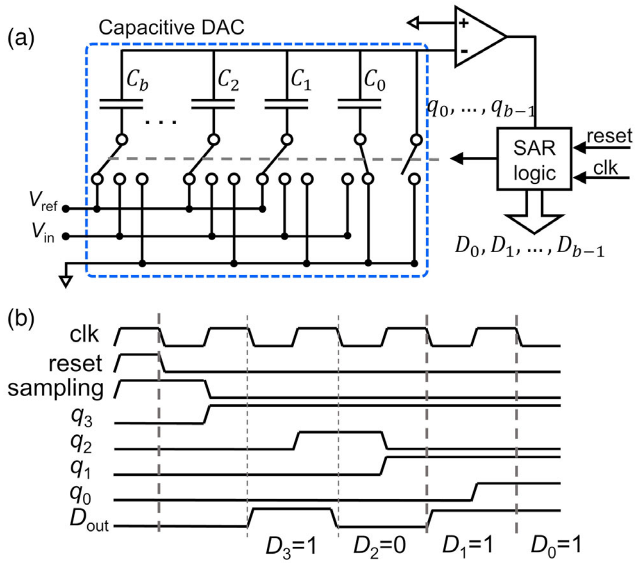

Figure 2. a) successive-approximation-register analog to digital converter (SAR-ADC) circuit diagram. The reference voltage V_{ref} was set to 1 V. b) Timing diagram for 4-bit resolution.

The power consumed by the capacitive DAC including the capacitor array and switches (P_{cDAC}) is given by a function of the resolution b as follows.[33]

$$P_{\text{cDAC}} = \underbrace{P_{A0}\frac{2^b}{b+1}}_{\text{cap array}} + \underbrace{P_1 b}_{\text{switch}} + P_2 \tag{9}$$

where P_{A0}, P_1, and P_2 are constant with respect to the resolution b. Regarding the area A_{cDAC}, the areas of the capacitor array and switches linearly scale with C_{tot} in Equation (7) and resolution b, respectively, and thus we have

$$A_{\text{cDAC}} = \underbrace{A_{A0}2^b}_{\text{cap array}} + \underbrace{A_1 b}_{\text{switch}} + A_2 \tag{10}$$

where A_{A0}, A_1, and A_2 are constant with respect to the resolution b.

The SAR logic in Figure 1 consists of b flip-flops, so that its power consumption P_{logic} and area A_{logic} linearly scales with the resolution b

$$\begin{cases} P_{\text{logic}} = P_3 b + P_4 \\ A_{\text{logic}} = A_3 b + A_4 \end{cases} \tag{11}$$

The comparator power P_{comp} and area A_{comp} are independent of the resolution b. Therefore, the power P_{ADC} and area A_{ADC} of the unit SAR-ADC are modeled using Equation (9)–(11) as

$$\begin{cases} P_{\text{ADC}} &= P_{\text{cDAC}} + P_{\text{logic}} + P_{\text{comp}} \\ &= P_{\text{A0}}\frac{2^b}{b+1} + P_{\text{A1}}b + P_{\text{A2}} \\ A_{\text{ADC}} &= A_{\text{cDAC}} + A_{\text{logic}} + A_{\text{comp}} \\ &= A_{\text{A0}}2^b + A_{\text{A1}}b + A_{\text{A2}} \end{cases} \tag{12}$$

where $P_{\text{A1}} = P_1 + P_3$, $P_{\text{A2}} = P_2 + P_4$, $A_{\text{A1}} = A_1 + A_3$, and $A_{\text{A2}} = A_2 + A_4$. The constants $P_{\text{A0}} - P_{\text{A2}}$ and $A_{\text{A0}} - A_{\text{A2}}$ are extracted from power and area breakdown, respectively, for our design of a SAR-ADC using the GPDK 45 nm technology as shown in **Figure 3**. The results are listed in **Table 1**.

3.2.2. *S&A Units*

The S&A unit is responsible for the digital summations in Equation (6) such that: i) the n_w b-bit signals from the n_w parallel SAR-ADCs for a given bit t are shifted and subsequently summed, and ii) the result for the cycle t is accumulated for the outermost summation in Equation (6). The S&A consists of: 1) n_w b-bit registers with $n_w - 1$ shift registers, 2) an adder tree, and 3) an accumulator (see **Figure 4**). As shown in Figure 4, the operation takes two clock cycles, so the latency T_{SA} is given by

$$T_{\text{SA}} = 2f_{\text{clk}}^{-1} \tag{13}$$

To address the power consumption of the S&A unit P_{SA}, we consider the power consumption of each part in Figure 4, which is proportional to the resolution of data output. The shift-register

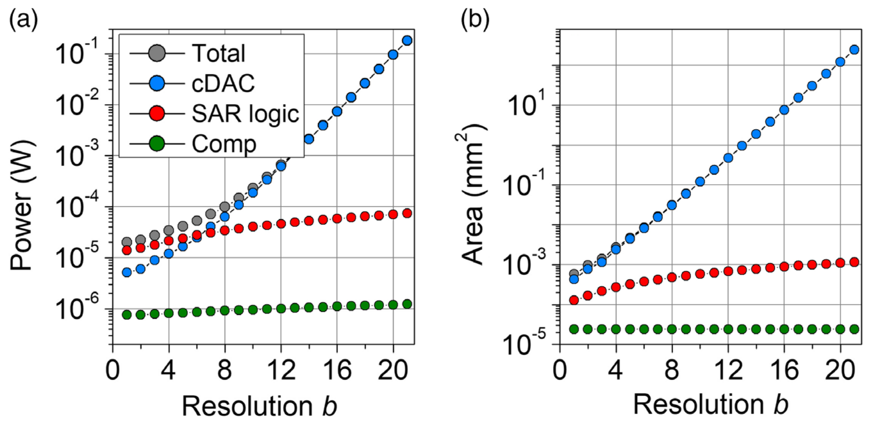

Figure 3. a) Power- and b) area-breakdowns for a unit SAR-ADC with respect to its resolution b.

Table 1. Area and power overheads of major components in an RRAM-based CIM macro.

SAR-ADC	(GPDK 45 [nm]); $V_{ref} = 1V$
Power P_{ADC}	$P_{A0}\frac{2^b}{b+1} + P_{A1}b + P_{A2}$ (W)
	$P_{A0} = 1.9 \times 10^{-6}$
	$P_{A1} = 4.3 \times 10^{-6}$
	$P_{A2} = 1.12 \times 10^{-5}$
Area A_{ADC}	$A_{A0}2^b + A_{A1}b + A_{A2}$ [mm^2]
	$A_{A0} = 1.16 \times 10^{-4}$
	$A_{A1} = 1.64 \times 10^{-4}$
	$A_{A2} = 1.72 \times 10^{-4}$
Latency T_{ADC}	$(b+1)f_{clk}^{-1}$
S&A	(GPDK 45 nm)
Power P_{SA}	$P_{S0}b''n_w + P_{S1}b''(n_w - 1) + P_{S2}b'$ [W]
	$P_{S0} = 3.35 \times 10^{-7}$
	$P_{S1} = 1.73 \times 10^{-7}$
	$P_{S2} = 5.58 \times 10^{-7}$
Area A_{SA}	$(A_{S0}b''n_w)^{\varepsilon} + A_{S1}b''(n_w - 1) + A_{S2}b'$ [mm^2]
	$A_{S0} = 7.09 \times 10^{-6}$
	$A_{S1} = 5.93 \times 10^{-6}$
	$A_{S2} = 1.59 \times 10^{-5}$
	$\varepsilon = 0.78$
Latency T_{SA}	$2f_{clk}^{-1}$
1-bit DAC	(GPDK 45 nm)
Power P_{DAC}	1 μW
Area A_{DAC}	6.25×10^{-6} [mm^2]
RRAM array	$M \times N$ array
Power P_M	$n_M n_w P_{cell}$; single-cell power $P_{cell} = 10$ nW
Area A_M	MNA_{cell}; single cell size $A_{cell} = 50 \times 50$ nm^2
Latency T_M	50 ns

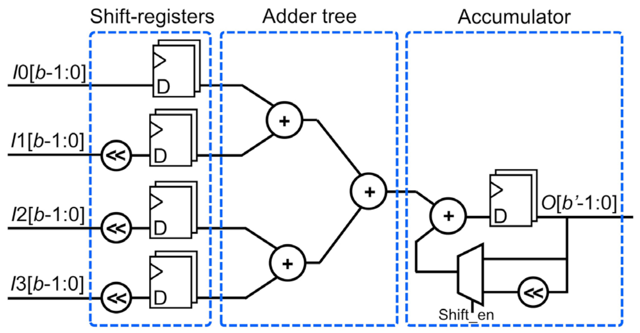

Figure 4. Shift-and-add unit (S&A) circuit diagram for $n_w = 4$.

and adder tree parts output b''-bit data, whereas the accumulator outputs b'-bit data, where

$$b'' = \log_2 n_M + w$$
$$b' = \log_2 M + w + a \tag{14}$$

Thus, the power consumption of the S&A unit P_{SA} can be modeled as

$$P_{SA} = \underbrace{P_{S0}b''n_w}_{\text{shift-registers}} + \underbrace{P_{S1}b''(n_w - 1)}_{\text{adder tree}} + \underbrace{P_{S2}b'}_{\text{accumulator}} \tag{15}$$

where P_{S0}, P_{S1}, and P_{S2} are constant. **Figure 5** shows the power consumption with weight splits n_w for an S&A unit for 8-bit weights ($w = 8$) and 64 rows in aggregate in an array ($M = 64$). The constants P_{S0}, P_{S1}, and P_{S2} were acquired by fitting Equation (15) to the data in Figure 4, which is listed in Table 1.

The area of the S&A unit is modeled using a heuristic equation as

$$A_{SA} = (A_{S0}b''n_w)^{\varepsilon} + A_{S1}b''(n_w - 1) + A_{S2}b' \tag{16}$$

where A_{S0}, A_{S1}, A_{S2}, and ε are constant. Unlike the power in Equation (15), (16) involves an additional parameter ε that indicates the optimal arrangement of the CMOS components in the

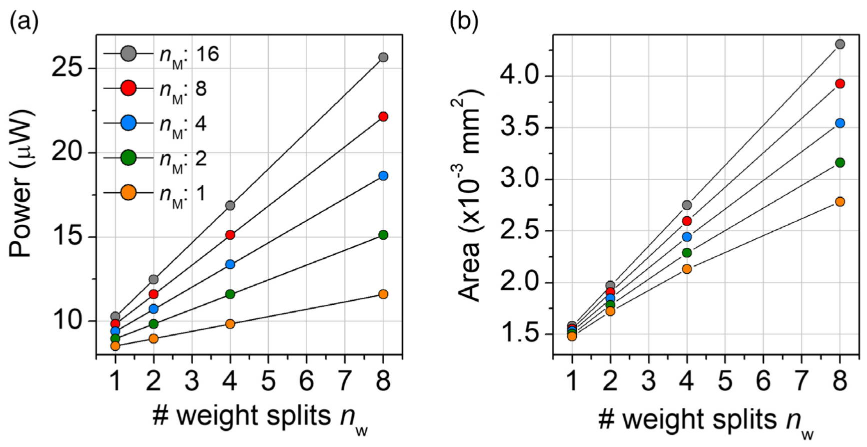

Figure 5. a) Power- and b) area-breakdowns for an S&A with respect to the number of weight splits n_w for $w = 8$ and $M = 64$.

S&A unit. These parameters for GPDK45nm are also listed in Table 1.

3.3. Power and Area Estimation Per Core

The major components of the RRAM-based mCIM include an RRAM array, DACs, ADCs, and S & A (Figure 1b). Using the power and area overhead of these components in Table 1, we can estimate the power and area of the mCIM core, P_{core} and A_{core}, as follows.

$$\begin{aligned} P_{\text{core}} &= n_M n_w P_{\text{cell}} + M P_{\text{DAC}} + n_w P_{\text{ADC}} + P_{\text{SA}} + P_{\text{r}} \\ A_{\text{core}} &= MNA_{\text{cell}} + MA_{\text{DAC}} + n_w A_{\text{ADC}} + A_{\text{SA}} + A_{\text{r}} \end{aligned} \tag{17}$$

where P_{r} and A_{r} denote the power and area of the other components in the mCIM, which are not subject to optimization.

3.4. Operational Latency

The MAC operations in the mCIM are fully pipelined as shown in **Figure 6**. The $M \times N$ RRAM array is partitioned into $M/n_M \times N/n_w$ sub-arrays, each of which is of $n_M \times n_w$ in size. We index a given sub-array using the notation Memory$[i,j]$, where $i \in [1, M/n_M]$, and $j \in [1, N/n_w]$. Each sub-array Memory$[i,j]$ is addressed at one cycle to perform the parallel MAC operations indicated in Equation (6). A set of n_w SAR-ADCs are shared among the sub-arrays using time-division multiplexing, and thus, for an addressed sub-array, an independent SAR-ADC is dedicated to each column. Given the encoding of a-bit activation, the MAC operations need a cycles and two additional cycles in Figure 6,i.e., $a + 2$ cycles in total. Note that the pipeline includes one pre-cycle to read activations from the buffer, but omitted in Figure 5 because this pre-cycle is much faster than the others.

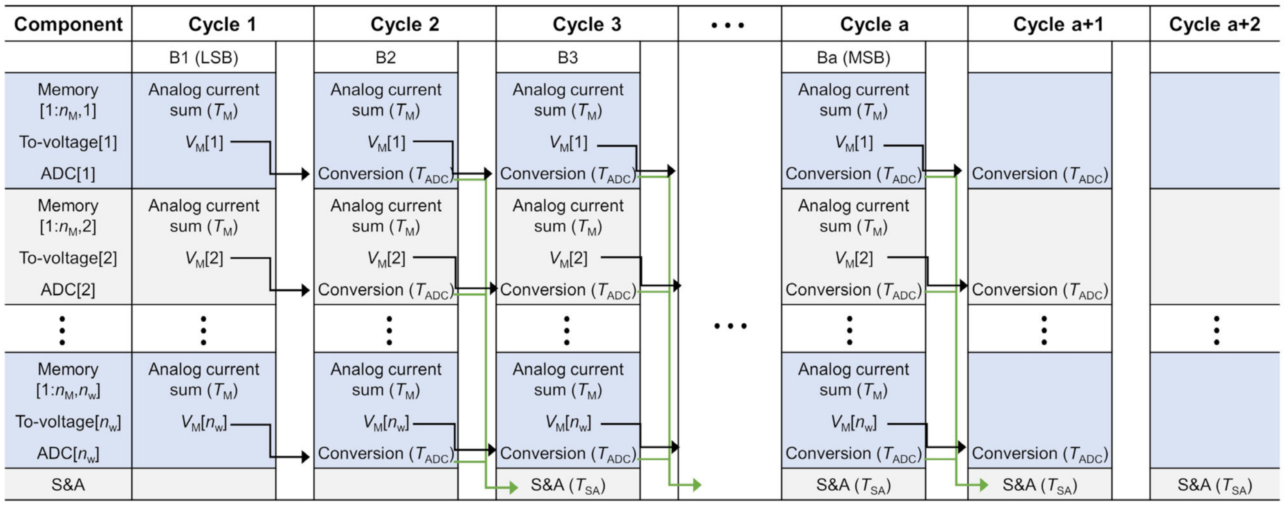

Figure 6. Mixed-signal-based computing-in-memory (mCIM) pipeline for a set of n_M parallel MAC operations for partial sum S_p. The RRAM array is partitioned into $n_M \times n_w$ sub-arrays. Memory$[1:n_M, i]$ denotes the ith column of a given sub-array. To-voltage$[i]$ denotes the conversion of the current sum for the ith column to its corresponding voltage $V_M[i]$.

According to Figure 6, the duration of each cycle T_{cyc} is given by

$$T_{\text{cyc}} = \max(T_{\text{M}}, T_{\text{ADC}}, T_{\text{SA}}) \tag{18}$$

where T_{M} denotes RRAM read-out time, which is set to 50 ns. The ADC latency (T_{ADC}) and S&A latency (T_{SA}) are given by Equation (8) and (13), respectively. Thus, the cycle duration is determined by the resolution of ADCs such that, for given f_{clk} (= 100 MHz) and T_{M} (= 50 ns), $T_{\text{cyc}} = T_{\text{M}}$ if $b \leq 4$, and $T_{\text{cyc}} = (b+1)f_{\text{clk}}^{-1}$ otherwise. The total latency T for the MAC operations S_{p} in Equation (6) is therefore given by

$$T = (a+2)T_{\text{cyc}} \tag{19}$$

4. Optimal Weight-Split Strategy

We address the optimal arrangement of the $M/n_{\text{M}} \times N/n_{\text{w}}$ sub-arrays (each of which is of $n_{\text{M}} \times n_{\text{w}}$ in size) in the $M \times N$ array considering the trade-off between the operational latency and area/power overhead. Here, we consider the PAE of the mCIM macro as figures of merits.

4.1. Power- and Area-Efficiency of mCIM Macro

One MAC operation is considered as two operations (multiplication and accumulation), i.e., 1 MAC = 2 OPs. The calculation of the partial sum S_p in Equation (6), which involves $2n_M$ OPs, is the unit task to evaluate the PAE.

$$\text{PAE} = \frac{2n_M}{P_{\text{core}}A_{\text{core}}T} \tag{20}$$

where P_{core} and A_{core} are given by Equation (17), and T is given by Equation (19).

The optimal n_w value regarding the maximum PAE can be calculated by differentiating Equation (20) with respect to n_w.

$$\left.\frac{\partial \text{PAE}}{\partial n_w}\right|_{n_w^*} = -\frac{2n_M}{(P_{\text{core}}A_{\text{core}}T)^2}\frac{\partial(P_{\text{core}}A_{\text{core}}T)}{\partial n_w} = 0 \tag{21}$$

and thus the optimal n_w^* value leads to

$$\frac{\partial(P_{\text{core}}A_{\text{core}}T)}{\partial n_w}[n_w^*] = 0 \tag{22}$$

The optimal n_M (n_M^*) (if exists) can be acquired by differentiating Equation (20) with n_M.

$$\frac{\partial \text{PAE}}{\partial n_M} = \frac{2}{(P_{\text{core}}A_{\text{core}}T)^2}\left[P_{\text{core}}A_{\text{core}}T - n_M\frac{(\partial P_{\text{core}}A_{\text{core}}T)}{\partial n_M}\right] \tag{23}$$

Thus, the optimal n_M^* satisfies the following equation.

$$(P_{\text{core}}A_{\text{core}}T)[n_M^*] = n_M^*\frac{\partial(P_{\text{core}}A_{\text{core}}T)}{\partial n_M}[n_M^*] \tag{24}$$

In search of the optimal n_w, we calculated the $P_{\text{core}}A_{\text{core}}T$ product for a mCIM macro with a 128×128 array (i.e., $M = N = 128$) with respect to the number of sub-weights n_w. The activation resolution a was set to 8, and the parameters listed in Table 1 were used. **Figure 7**a shows the calculation results for $w = 8$ and various n_M values ($1-64$). For all n_M except for $n_M = 1$, the $P_{\text{core}}A_{\text{core}}T$ product attains its minimum at $n_w = 4$, i.e., 2-bit sub-weight per RRAM cell, whereas, for $n_M = 1$, the minimum is placed at $n_w = 2$. The corresponding PAE values are shown in Figure 7b. Given n_w^*, the PAE peaks at $n_w = 2$ and $n_w = 4$ for $n_M = 1$ and $n_M > 1$, respectively.

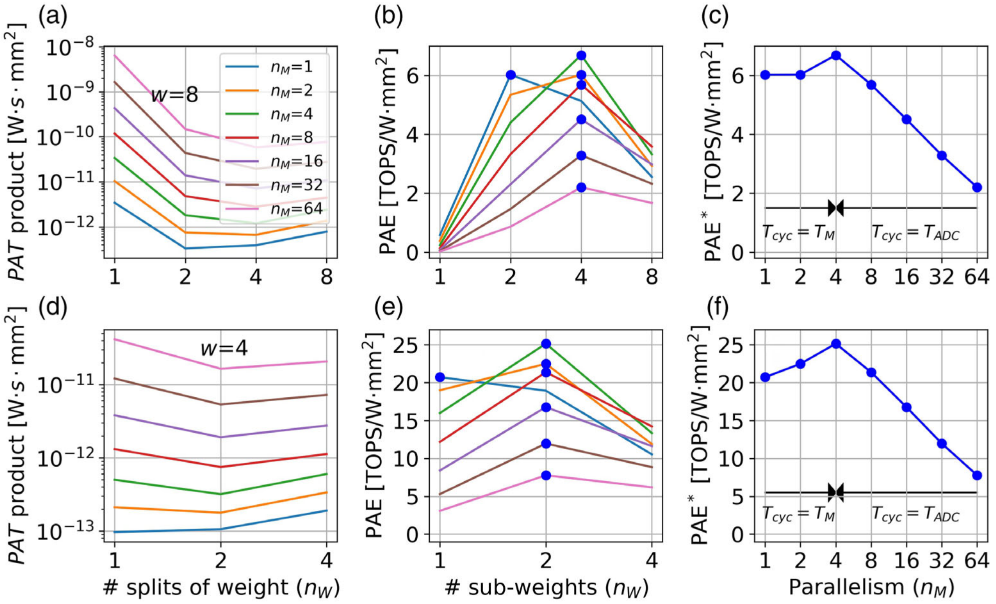

Figure 7. a) $P_{\text{core}}A_{\text{core}}T$ product, b) corresponding power- and area-efficiency of operations (PAE) with the number of sub-weights (n_{w}), and c) PAE at the optimal n_{w} with the operational parallelism n_{M} for $w = 8$. The same plots for $w = 4$ are shown in d), e), and f). For both cases, the array size and activation resolution a were set to 128×128 and 8, respectively.

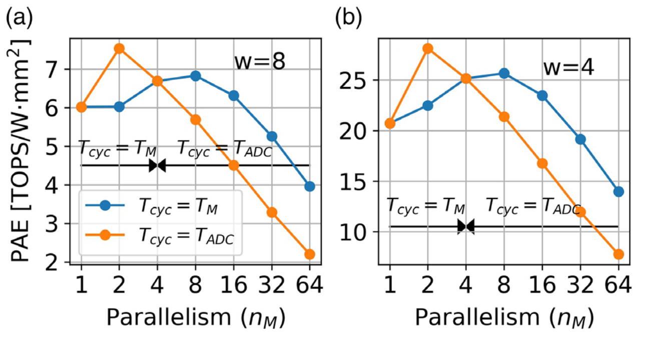

Figure 8. PAE* with $T_{\text{cyc}} = T_{\text{M}}$ and $T_{\text{cyc}} = T_{\text{ADC}}$ for: a) $w = 8$ and b) $w = 4$.

The presence of the optimal n_w for $w = 4$ was also identified as seen in Figure 6d, yielding $n_w^* = 2$ for $n_M > 1$, i.e., 2-bit sub-weight per RRAM cell while $n_w^* = 1$ for $n_M = 1$. Accordingly, PAE peaks at $n_w^* = 2$ for $n_M > 1$ and $n_w^* = 1$ for $n_M = 1$ (Figure 6e).

We subsequently identified if there exists the optimal n_M for the optimal n_w by plotting PAE at n_w^* (PAE*) with respect to n_M (Figure 7c,f for $w = 8$ and $w = 4$, respectively). Both figures indicate the maximum PAE* at $n_M = 4$, i.e., $n_M^* = 4$. This is because of the cycle duration T_{cyc} in Equation (18) determined by T_M for $n_M \leq 4$ while determined by T_{ADC} for $n_M > 4$. As explained in Section 3.4, the RRAM read-out time T_{M} (= 50 ns) dictates T_{cyc} when $b \leq 4$ which corresponds to $n_M \leq 4$ for $w/n_w = 2$ and $n_M = 1$ for $w/n_w = 4$ according to Equation (4). Therefore, PAE* in Figure 6c,f is dictated by T_{M} and T_{ADC} for $n_M \leq 4$ and $n_M > 4$, respectively. **Figure 8** identifies this by separately showing PAE* with $T_{\text{cyc}} = T_{\text{M}}$ and $T_{\text{cyc}} = T_{\text{ADC}}$ for $w = 8$ and $w = 4$. The optimal n_M is placed at $n_M = 4$ for which $T_{\text{M}} = T_{\text{ADC}}$, i.e., $n_M^* = 4$ for both $w = 8$ and $w = 4$. Note that the abrupt drops in PAE*($T_{\text{cyc}} = T_{\text{ADC}}$) for $n_M = 1$ (compared with $n_M = 2$) in Figure 8 are due to the larger bit-width b for $n_M = 1$ than $n_M = 2$. For $w = 8$, $n_w^* = 2$ for $n_M = 1$, yielding a bit-width b of 4 using Equation (4). However, $n_w^* = 4$ for $n_M = 2$, yielding a bit-width b of 3, so that the product $P_{\text{core}}A_{\text{core}}T$ for $n_M = 1$ outweighs the product for $n_M = 2$, leading to the drop in PAE* at $n_M = 1$. The same holds for $w = 4$.

We highlight the significance of performance improvement due to the optimal n_w for a given n_M by comparing the maximum PAE at n_w^* (PAE*) with two naive cases: 1) $n_w = 1$ and 2) $n_w = w$. The former indicates the case in which each RRAM cell is forced to represent as many conductance levels as possible to increase the memory density. To this end, the number of conductance levels is frequently taken as an important measure of RRAM performance.[35–37] For Case (i), the relative PAE (PAE/PAE[$n_w = 1$]) for $w = 8$ is shown in **Figure 9**a. As such, the relative PAE peaks at $n_w = n_w^* = 4$ insomuch as PAE*/PAE[$n_w = 1$] for $n_M^*(= 4)$ attains ≈28.3× PAE for the naive case. This is because the naive case ($n_w = 1$) causes the prohibitive power and area overheads of the peripheral circuits, and thus considerably degrading PAE. The other naive case (Case (ii)) uses 1-bit RRAM cells, so that $n_w = w$. For $n_M^*(= 4)$, the maximum PAE at $n_w^* = 4$ achieves a 2× improvement in PAE compared with Case (ii) as seen in Figure 8b. For Case (ii), the power and area overheads of a unit ADC are much lower than Case (i) and even the optimal case. Nevertheless, more parallel ADCs are needed than the optimal case, so that the overall power and area overheads outweigh the optimal case. Note that the relative PAEs in Figure 8 can be compared among different n_w (but the same

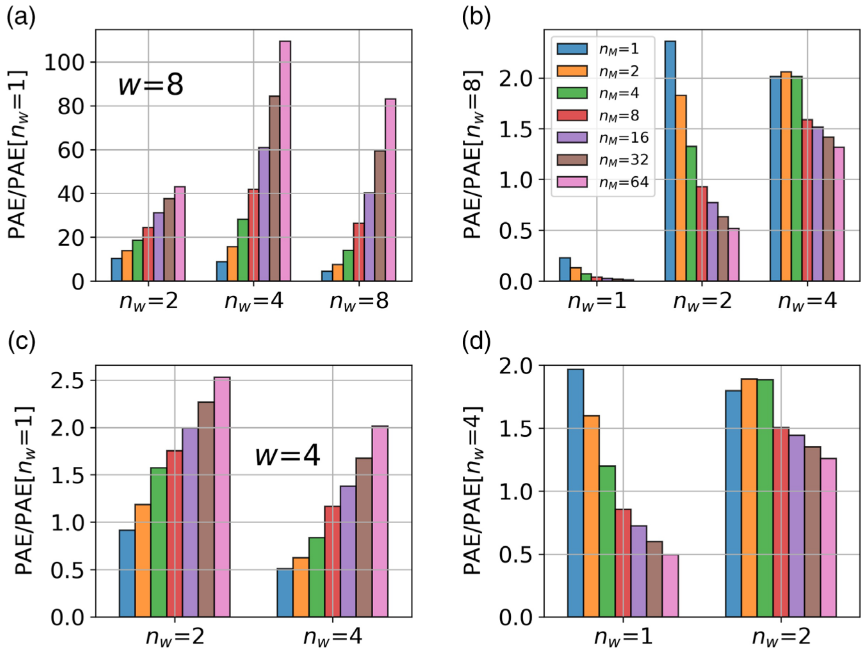

Figure 9. PAE relative to PAE at: a) $n_w = 1$ and b) $n_w = 8$ for $w = 8$. For $w = 4$, PAE relative to PAE at $n_w = 1$ and $n_w = 4$ is shown in c) and d), respectively.

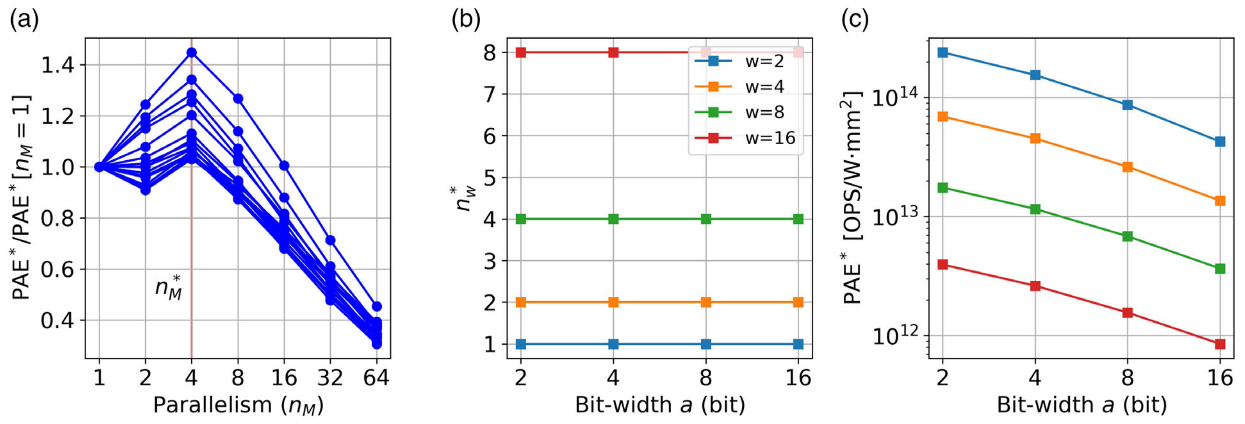

Figure 10. a) Relative PAE* (PAE*/PAE*[$n_M = 1$]) with n_M, b) n_w^*, and c) PAE* for $w, a \in \{2, 4, 8, 16\}$.

n_M) cases only. This is because PAE[n_w] is normalized by PAE[$n_w = 1$] of the same n_M.

Similar features are seen for the case of $w = 4$ as shown in Figure 8c,d. Albeit similar, a large difference from the case $w = 8$ lies in the relative PAE (PAE*/PAE[$n_w = 1$]) in that the maximum relative PAE is ≈1.6× PAE for $n_w = 1$ unlike 28.3× for the case $w = 8$. This is because the use of low-weight resolution ($w = 4$) causes much lower power and area overheads than the higher weight resolution ($w = 8$).

For generalization, we address the optimal sub-array size ($n_M^* \times n_w^*$) for various weight and activation formats (w, a); $w \in \{2, 4, 8, 16\}$ and $a \in \{2, 4, 8, 16\}$, i.e., 16 distinct cases in aggregate. **Figure 10**a identifies $n_M^* = 4$ for the 16 cases given the peak of relative PAE* (PAE*/PAE*[$n_M = 1$]) at $n_M = 4$ for all cases. The optimal n_w values for all cases were calculated at $n_M^* = 4$ and plotted in Figure 10b. The results highlight the relationship $n_w^* = w/2$ irrespective of weight and activation formats. Thus, we have the optimal sub-array size $n_M^* \times n_w^* = 4 \times w/2$ for $w, a \in \{2, 4, 8, 16\}$. PAE* for all 16 cases is plotted in Figure 10c.

5. Conclusion

The increase of memory density in the RRAM array domain by multilevel memory operations comes at the cost of considerable power and area overheads for the peripheral circuits of an RRAM-based mCIMs macro, mainly, ADCs and S&As. The same holds for operational parallelism, i.e., the number of parallel MAC operations. We have clarified the prohibitive power and area overheads for the peripheral circuits, which are often ignored. Particular emphasis was placed on the ADCs and S&As whose power and area overheads strongly rely on the resolution of data processed; the resolution is determined by the memory bit of each RRAM cell and the operational parallelism. In this regard, we have proposed a strategy to determine the optimal memory bit of each RRAM cell and parallelism, which corresponds to the optimal size of a sub-array that is addressed at one cycle, to maximize the PAE as a figure of merit. Various weight and activation data formats ($w, a \in \{2, 4, 8, 16\}$) with SAR-ADCs and S&As (designed using the Cadence GPDK 45 nm) commonly indicate $n_M^* \times n_w^* = 4 \times w/2$ as the optimal sub-array size.

Acknowledgements

This work was also supported by National R&D Program through the National Research Foundation of Korea (NRF) funded by Ministry of Science and ICT (NRF-2021M3F3A2A01037632 and NRF-2019R1C1C1009810). The EDA tool was supported by the IC Design Education Center (IDEC), Korea.

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Keywords

computing-in-memory, mixed signal-based CIM, optimal weight-spliting, RRAM-based CIM macro

Received: August 28, 2022
Revised: October 13, 2022
Published online:

[1] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers, et al., in *Proc. of the 44th Annual Int. Symp. on Computer Architecture*, **2017** pp. 1–12.
[2] J. Jung, J. Park, A. Kumar, in *2019 20th Int. Workshop on Microprocessor/SoC Test, Security and Verification (MTV)*, IEEE, Piscataway, NJ **2019**, pp. 13–17.
[3] A. Boroumand, S. Ghose, Y. Kim, R. Ausavarungnirun, E. Shiu, R. Thakur, D. Kim, A. Kuusela, A. Knies, P. Ranganathan, O. Mutlu, in *Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks*, Association for Computing Machinery, New York, NY **2018**, pp. 316–331.

[4] Y.-C. Kwon, S. H. Lee, J. Lee, S.-H. Kwon, J. M. Ryu, J.-P. Son, O. Seongil, H.-S. Yu, H. Lee, S. Y. Kim, Y. Cho, J. G. Kim, J. Choi, H.-S. Shin, J. Kim, B. Phuah, H. Kim, M. J. Song, A. Choi, D. Kim, S. Kim, E.-B. Kim, D. Wang, S. Kang, Y. Ro, S. Seo, J. Song, J. Youn, K. Sohn, N. S. Kim, in *IEEE Int. Solid- State Circuits Conf. (ISSCC)*, Vol *64*, IEEE, Piscataway, NJ **2021**, pp. 350–352.
[5] S. Lee, K. Kim, S. Oh, J. Park, G. Hong, D. Ka, K. Hwang, J. Park, K. Kang, J. Kim, J. Jeon, N. Kim, Y. Kwon, K. Vladimir, W. Shin, J. Won, M. Lee, H. Joo, H. Choi, J. Lee, D. Ko, Y. Jun, K. Cho, I. Kim, C. Song, C. Jeong, D. Kwon, J. Jang, I. Park, J. Chun, et al.in *IEEE Int. Solid- State Circuits Conf. (ISSCC)*, Vol *65*, IEEE, Piscataway, NJ **2022**, pp. 1–3.
[6] J. L. Hennessy, D. A. Patterson, in *Computer Architecture – A Quantitative Approach (4. ed.)*, Morgan Kaufmann, Burlington, MA **2007**.
[7] S. Xie, C. Ni, A. Sayal, P. Jain, F. Hamzaoglu, J. P. Kulkarni, in *IEEE Int. Solid- State Circuits Conf. (ISSCC)*, Vol *64*, IEEE, Piscataway, NJ **2021** pp. 248–250.
[8] J.-W. Su, Y.-C. Chou, R. Liu, T.-W. Liu, P.-J. Lu, P.-C. Wu, Y.-L. Chung, L.-Y. Hung, J.-S. Ren, T. Pan, S.-H. Li, S.-C. Chang, S.-S. Sheu, W.-C. Lo, C.-I. Wu, X. Si, C.-C. Lo, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, M.-F. Chang, in *IEEE Int. Solid- State Circuits Conf. (ISSCC)*, Vol *64*, IEEE, Piscataway, NJ **2021**, pp. 250–252.
[9] K. Ueyoshi, I. A. Papistas, P. Houshmand, G. M. Sarda, V. Jain, M. Shi, Q. Zheng, S. Giraldo, P. Vrancx, J. Doevenspeck, D. Bhattacharjee, S. Cosemans, A. Mallik, P. Debacker, D. Verkest, M. Verhelst, in *IEEE Int. Solid- State Circuits Conf. (ISSCC)*, Vol *65*, IEEE, Piscataway, NJ **2022** pp. 1–3.
[10] I. A. Papistas, S. Cosemans, B. Rooseleer, J. Doevenspeck, M.-H. Na, A. Mallik, P. Debacker, D. Verkest, in *IEEE Custom Integrated Circuits Conf. (CICC)*, IEEE, Piscataway, NJ **2021**, pp. 1–2.
[11] S. Jung, H. Lee, S. Myung, H. Kim, S. K. Yoon, S.-W. Kwon, Y. Ju, M. Kim, W. Yi, S. Han, B. Kwon, B. Seo, K. Lee, G.-H. Koh, K. Lee, Y. Song, C. Choi, D. Ham, S. J. Kim, *Nature* **2022**, *601*, 211.
[12] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, V. Srikumar, *ACM SIGARCH Comput. Arch. News* **2016**, *44*, 14.
[13] A. Nag, R. Balasubramonian, V. Srikumar, R. Walker, A. Shafiee, J. P. Strachan, N. Muralimanohar, *IEEE Micro* **2018**, *38*, 41.
[14] F. Cai, J. M. Correll, S. H. Lee, Y. Lim, V. Bothra, Z. Zhang, M. P. Flynn, W. D. Lu, *Nat. Electron.* **2019**, *2*, 290.
[15] S. Yin, X. Sun, S. Yu, J. S. Seo, *IEEE Transactions on Electron Devices* **2020**, *67*, 4185.
[16] P. Yao, H. Wu, B. Gao, J. Tang, Q. Zhang, W. Zhang, J. J. Yang, H. Qian, *Nature* **2020**, *577*, 641.
[17] J.-H. Yoon, M. Chang, W.-S. Khwa, Y.-D. Chih, M.-F. Chang, A. Raychowdhury, in *IEEE Int. Solid- State Circuits Conf. (ISSCC)*, Vol *64*, Piscataway, NJ **2021**, pp. 404–406.
[18] K. Jeong, J. Kim, J. J. Ryu, S.-J. Yoo, C. Song, M. K. Yang, D. S. Jeong, G. H. Kim, *Nat. Commun.* **2021**, *12*, 2968.
[19] W.-H. Chen, C. Dou, K.-X. Li, W.-Y. Lin, P.-Y. Li, J.-H. Huang, J.-H. Wang, W.-C. Wei, C.-X. Xue, Y.-C. Chiu, Y.-C. King, C.-J. Lin, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, J. J. Yang, M.-S. Ho, M.-F. Chang, *Nat. Electron.* **2019**, *2*, 420.
[20] C.-X. Xue, W.-H. Chen, J.-S. Liu, J.-F. Li, W.-Y. Lin, W.-E. Lin, J.-H. Wang, W.-C. Wei, T.-W. Chang, T.-C. Chang, T.-Y. Huang, H.-Y. Kao, S.-Y. Wei, Y.-C. Chiu, C.-Y. Lee, C.-C. Lo, Y.-C. King, C.-J. Lin, R.-S. Liu, C.-C. Hsieh, K.-T. Tang, M.-F. Chang, in *IEEE Int. Solid-State Circuits Conf. (ISSCC)*, IEEE, Piscataway, NJ **2019**, pp. 388–390.
[21] Z. Li, Z. Wang, L. Xu, Q. Dong, B. Liu, C. I. Su, W. T. Chu, G. Tsou, Y. D. Chih, T. Y. J. Chang, D. Sylvester, H. S. Kim, D. Blaauw, *IEEE J. Solid-State Circuits* **2021**, *56*, 1105.
[22] P.-Y. Chen, X. Peng, S. Yu, *IEEE Trans. Comput-Aid. Des. Integr. Circuit. Syst.* **2018**, *37*, 3067.
[23] X. Sun, S. Yin, X. Peng, R. Liu, J.-S. Seo, S. Yu, in *Design, Automation & Test in Europe Conf. & Exhibition (DATE)*, IEEE, Piscataway, NJ **2018**, pp. 1423–1428.
[24] Y. Park, S. Y. Lee, H. Shin, J. Heo, T. J. Ham, J. W. Lee, in *IEEE/ACM Int. Conf. on Computer Aided Design (ICCAD)*, IEEE, Piscataway, NJ **2020** pp. 1–9.
[25] R. Waser, R. Dittmann, G. Staikov, K. Szot, *Adv. Mater.* **2009**, *21*, 2632.
[26] D. S. Jeong, R. Thomas, R. Katiyar, J. Scott, H. Kohlstedt, A. Petraru, C. S. Hwang, *Rep. Progr. Phys.* **2012**, *75*, 076502.
[27] N. K. Upadhyay, W. Sun, P. Lin, S. Joshi, R. Midya, X. Zhang, Z. Wang, H. Jiang, J. H. Yoon, M. Rao, M. Chi, Q. Xia, J. J. Yang, *Adv. Electron. Mater.* **2020**, *6*, 1901411.
[28] N. K. Upadhyay, T. Blum, P. Maksymovych, N. V. Lavrik, N. Davila, J. A. Katine, A. V. Ievlev, M. Chi, Q. Xia, J. J. Yang, *Front. Nanotechnol.* **2021**, *3*, 656026.
[29] C.-W. Hsu, T.-H. Hou, M.-C. Chen, I.-T. Wang, C.-L. Lo, *IEEE Electron. Device Lett.* **2013**, *34*, 885.
[30] K. M. Kim, J. Zhang, C. Graves, J. J. Yang, B. J. Choi, C. S. Hwang, Z. Li, R. S. Williams, *Nano let.* **2016**, *16*, 6724.
[31] B. Murmann, ADC Performance Survey 1997-2021, http://web.stanford.edu/murmann/adcsurvey.html, (accessed: June 2021).
[32] C.-C. Liu, S.-J. Chang, G.-Y. Huang, Y.-Z. Lin, *IEEE J. Solid-State Circuits* **2010**, *45*, 731.
[33] M. Saberi, R. Lotfi, K. Mafinezhad, W. A. Serdijn, *IEEE Trans. Circuits Syst. I: Regul. Papers* **2011**, *58*, 1736.
[34] L. Kull, T. Toifl, M. Schmatz, P. A. Francese, C. Menolfi, M. Brändli, M. Kossel, T. Morf, T. M. Andersen, Y. Leblebici, *IEEE J. Solid-State Circuits* **2013**, *48*, 3049.
[35] G. H. Kim, H. Ju, M. K. Yang, D. K. Lee, J. W. Choi, J. H. Jang, S. G. Lee, I. S. Cha, B. K. Park, J. H. Han, T.-M. Chung, K. M. Kim, C. S. Hwang, Y. K. Lee, *Small* **2017**, *13*, 1701781.
[36] L. Wu, H. Liu, J. Li, S. Wang, X. Wang, *Nanoscale Res. Lett.* **2019**, *14*.
[37] V. Milo, C. Zambelli, P. Olivo, E. Pérez, M. K. Mahadevaiah, O. G. Ossorio, C. Wenger, D. Ielmini, *APL Mater.* **2019**, *7*, 081120.