# ADRA: Extending Digital Computing-In-Memory With Asymmetric Dual-Row-Activation

Akul Malhotra, Atanu K. Saha, Chunguang Wang, and Sumeet K. Gupta

***Abstract*—In this letter, we propose an asymmetric dual row activation scheme called ADRA for enhancing digital computing-in-memory (CiM) based on in-memory operands. ADRA solves the many-to-one mapping problem of input vectors to output voltages faced by previous CiM techniques, enabling (a) simultaneous single-cycle memory read and CiM of primitive Boolean functions (b) computation of any Boolean function and (c) CiM of non-commutative functions such as subtraction and comparison. While the proposed technique is technology-agnostic, we show its utility for ferroelectric transistor (FeFET)-based non-volatile memory. Compared to the standard near-memory computation, we show that our method can achieve a full scale two-operand digital CiM using just one memory access, leading to a 23.2% - 72.6% decrease in energy-delay product (EDP).**

***Index Terms*—Computing-in-memory, FeFET, in-memory subtraction, in-memory comparison.**

## I. Introduction

COMPUTING in-memory (CiM) is a promising alternative to standard von-Neumann architectures [1]. CiM performs certain computations within the memory array and has been extensively explored for several operations such as vector matrix multiplications, Boolean logic and arithmetic functions [2], [3]. In this brief, we will focus on digital CiM for Boolean and arithmetic operations for in-memory operands.

CiM of Boolean and arithmetic functions typically utilize multi-wordline assertion to compute primitive Boolean functions (AND/OR) within the memory array. A compute unit added to the peripheral circuitry enables the computation of standard arithmetic functions such as addition. Although CiM for Boolean and arithmetic functions has been thoroughly examined, such CiM techniques are capable of computing *only commutative functions*. This is because these techniques have a *many-to-one* mapping of input vectors to senseline outputs (details discussed later). This leads to the hardware treating two different input vectors as identical, limiting the computation to only a subset of all the possibilities. For example, an important function that is not feasible using such CiM approaches is in-memory subtraction. This is because the (0, 1) and (1, 0) input vectors are mapped to the same senseline current/voltage. Hence, while CiM of commutative functions such as addition is possible, CiM of noncommutative functions such as subtraction and comparison in a single cycle are infeasible.

Certain other CiM techniques utilize the voltage divider action between the two words to compute their bitwise XOR and can be used for CiM of some noncommutative functions [4]. However, these techniques also suffer from the many-to-one mapping problem and are unsuitable for CiM of common functions like addition. Works like [5] demonstrate a single cycle two-bit read functionality, which could further be used to implement both commutative and noncommutative functions, but their applicability is limited to differential memories with two access transistors (e.g., SRAMs).

In this brief, we propose a CiM technique which can compute *any* two-operand Boolean or arithmetic function in-memory with a single memory access. Our approach is based on asymmetric dual row activation (ADRA), which achieves *one-to-one* mapping of the input vectors to the senseline current or voltage, enabling full-scale digital CiM with in-memory operands. Although ADRA is applicable for any memory technology, we illustrate its applications employing Ferroelectric Field Effect Transistor (FeFET)-based memory. The contributions of this brief are as follows:

- We propose a CiM technique, referred to as ADRA, which employs asymmetric multi-wordline assertion to avert the many-to-one mapping problem, enabling simultaneous read of two bits in a single-cycle along with CiM of any two-input Boolean function (in conjunction with a peripheral compute unit).
- We show the utility of ADRA, supported by a compute module, to perform CiM of non-commutative functions such as subtraction and comparison (in *addition* to the commutative functions explored in previous techniques).
- We evaluate ADRA considering FeFET based memory array and show that our method can achieve a full scale digital CiM with a 23.2% - 72.6% energy-delay product (EDP) decrease compared to standard CiM/near-memory methods with negligible hardware overhead.

## II. Background

### A. Related Work

Boolean logic and arithmetic CiM using multi-wordline assertion has been presented in prior works with CMOS and emerging memories [4], [5], [6], [7] (Fig. 1(a)). Each bitcell produces a low resistance state (LRS) current ($I_L$) and a high resistance state (HRS) current ($I_H$) during read when it stores '1' and '0' respectively. To perform the compute operation, the bitline (RBL) and the senseline (SL) are driven to a read voltage $V_{READ}$ and ~0V, respectively. Then, wordlines ($WL_1$ and $WL_2$) corresponding to the two in-memory operands (A and B) are asserted to $V_{GREAD}$. The SL current ($I_{SL}$) is, therefore, the sum of the currents from both the bitcells. $I_{SL}$ can have three possible values for four different input vectors (see Fig. 1(b)).

Manuscript received 25 January 2023; accepted 4 March 2023. Date of publication 7 March 2023; date of current version 31 July 2023. This work was supported by the Center for Brain-Inspired Computing (C-BRIC), one of six centers in JUMP, funded by Semiconductor Research Corporation (SRC) and DARPA. This brief was recommended by Associate Editor J. Yang. *(Corresponding author: Akul Malhotra.)*

The authors are with the School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: malhot23@purdue.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSII.2023.3253659.

Digital Object Identifier 10.1109/TCSII.2023.3253659

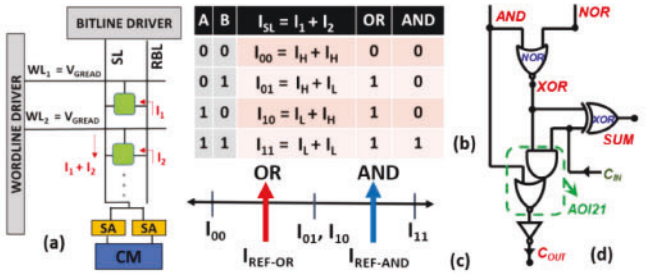



Fig. 1. (a) CiM in one column of a generic memory array, (b) $I_{SL}$ values for different input vectors, (c) reference currents for the sense amplifiers (SAs) and (d) Compute module for addition.

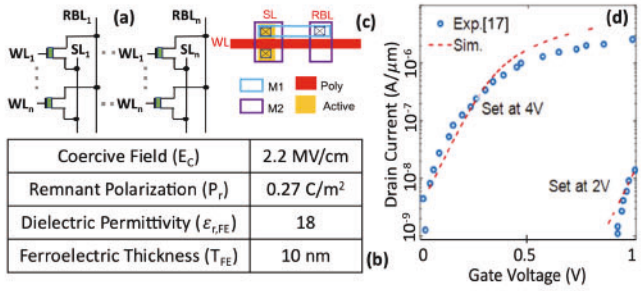



Fig. 2. (a) 1T FeFET array, (b) simulation parameters, (c) layout of a 1T FeFET bitcell (HZO compatible with process technology) and (d) I-V characteristics of the modelled device calibrated using [17].

Using two sense amplifiers (SAs) with appropriate reference currents (as shown in Fig. 1(c)), bitwise OR (A + B), bitwise AND (AB) and their complements can be computed. By adding a compute module (CM) to the peripherals (Fig. 1(d)), addition can be performed from the SA outputs.

It can be observed that $I_{SL} = (I_H + I_L)$ for both the input vectors (0, 1) and (1, 0). Therefore, when $I_{SL}$ is equal to $(I_H + I_L)$, it is impossible to distinguish between (0, 1) and (1, 0). This is not a problem for enabling the computation of commutative Boolean and arithmetic functions such as AND, OR, XOR and addition but prohibits the computation of *noncommutative functions* such as subtraction and comparison. The only way to distinguish between the input vectors is to perform a *second read operation* on one of them, which nearly doubles the latency and energy consumption.

Stateful Boolean logic in-memory is a commonly used approach to implement any Boolean function. However, it requires multiple memory accesses and several writes, leading to energy and latency overheads [8], [9]. Works like [10] perform in-memory comparison but do not have both operands stored in-memory, which is the target application of our method. Reference [11] uses partial memory access to perform energy efficient in-memory comparison but requires two cycles and can be only applied to non-volatile memories that allow partial memory access. Reference [12] proposes a SRAM-based CAM circuit for in-memory comparison, but it relies on monolithic 3-D integration process. Reference [4] is able to distinguish between (0, 1) and (1, 0) input vectors for XOR computation but is unable to distinguish between (0, 0) and (1, 1) input vectors, limiting the possible computations. Techniques like Memristor-Aided-Logic enable computation in RRAM based crossbar arrays, but need many intermediate cycles leading to performance and energy inefficiencies [13]. The work in [5] uses a split-wordline 6T SRAM bitcell to enable single-cycle two-bit read. Although this technique overcomes the previously discussed mapping issue, its use is limited to differential bitcells with two read access transistors.

ADRA overcomes the input mapping problem by mapping the 4 input vectors to 4 different $I_{SL}$ values. This allows us to implement any Boolean function, including subtraction and comparison along with those implemented in prior works. Moreover, ADRA is technology-agnostic, compatible with all known CMOS and non-volatile memories and can be used in conjunction with CiM-enhancing techniques such as write-back to implement multi-step computations [4].

### B. FeFET Based Non-Volatile Memories (NVMs)

FeFET-based memories are amongst the most promising emerging NVMs due to their low power electric-field driven write, high distinguishability and read-write path separation [14]. Although our technique is technology agnostic, we choose 1T-FeFET NVMs to illustrate ADRA because of their compatibility with current- and voltage-based sensing [15], which allows us to analyze ADRA considering both the sensing schemes. Also, its array footprint is amongst the lowest amongst various memories [16], resulting in a near-worst-case analysis of the overheads of the peripherals required by ADRA. It should be noted that FeFET NVMs have their own challenges such as high write voltages, endurance issues and variability; however, these limitations are not that of our technique but of the technology we have chosen as an example. Also, many solutions are being explored to counter the aforementioned drawbacks [21].

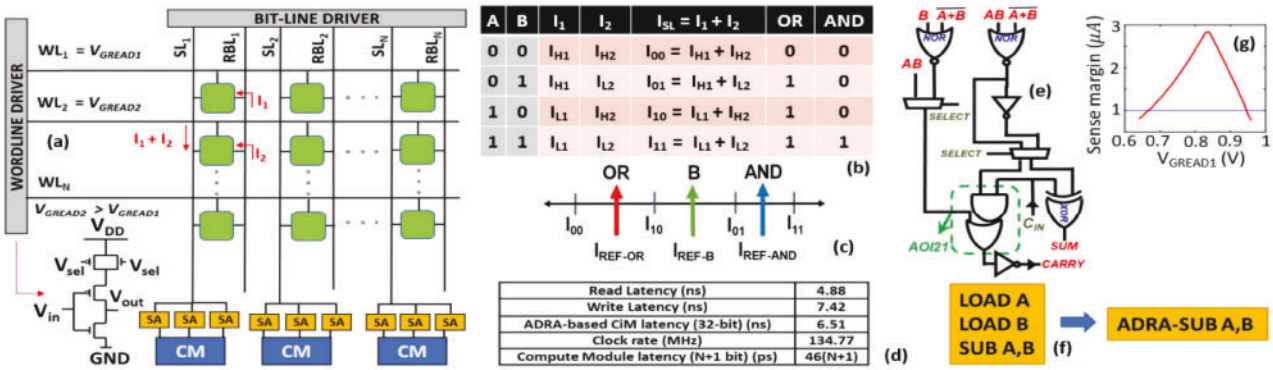
The FeFET NVM array is composed of 1T-FeFET bitcells (Fig. 2(a), layout in Fig. 2(c)). Each bitcell is connected to a senseline (SL) and a read bitline (RBL) along the column and a wordline (WL) along the row. The bit value is stored in the polarization of the FeFET [18]. To write a value into the bitcell, an appropriate gate-to-source voltage ($V_{GS}$) is applied on the FeFET ($V_{GS} > V_C$ to write positive polarization (+P: LRS) and $V_{GS} < -V_C$ for negative polarization (−P: HRS): where $V_C$ is the coercive voltage of the FeFET). In an array, this can be accomplished in different ways such as using a two phase write or FLASH-like global reset followed by selective set [19]. For read, both voltage and current-based sensing can be used. For the former, RBL is precharged to a read voltage ($V_{READ}$) with SL biased at 0V. The wordline is asserted to $V_{GREAD}$ ($<V_C$). Hence, RBL discharges for $+P$ and remains at $V_{READ}$ for $-P$, which is sensed by a SA. For the current-based sensing, RBL is biased to $V_{READ}$ and on the assertion of WL, the value of $I_{SL}$ ($I_L$ for $+P$ and $I_H$ for $-P$) is sensed. Some previous works have explored CiM with FeFETs, albeit with the limitations described before [15].

To evaluate the FeFET arrays, we use Preisach/Miller's equation for FE coupled with 45nm FET models [20], [22]. We calibrate the FeFET model with experiments on CMOS-compatible $Hf_{0.5}Zr_{0.5}O_2$-based (HZO) FeFET (Fig. 2(d), simulation parameters in Fig. 2(b)) [17].

## III. Proposed CiM Using Asymmetric Dual Row Activation (ADRA)

### A. Asymmetric Wordline Assertion Scheme

As discussed in the previous section, the limitations of the previous CiM techniques are due to the many-to-one mapping of input vectors to senseline currents. In this section, we describe an asymmetric wordline assertion scheme called ADRA, which maps each input vector to a different $I_{SL}$ value.




| A | B | $I_1$ | $I_2$ | $I_{SL} = I_1 + I_2$ | OR | AND |
|---|---|---|---|---|---|---|
| 0 | 0 | $I_{H1}$ | $I_{H2}$ | $I_{00} = I_{H1} + I_{H2}$ | 0 | 0 |
| 0 | 1 | $I_{H1}$ | $I_{L2}$ | $I_{01} = I_{H1} + I_{L2}$ | 1 | 0 |
| 1 | 0 | $I_{L1}$ | $I_{H2}$ | $I_{10} = I_{L1} + I_{H2}$ | 1 | 0 |
| 1 | 1 | $I_{L1}$ | $I_{L2}$ | $I_{11} = I_{L1} + I_{L2}$ | 1 | 1 |

| | |
|---|---|
| Read Latency (ns) | 4.88 |
| Write Latency (ns) | 7.42 |
| ADRA-based CiM latency (32-bit) (ns) | 6.51 |
| Clock rate (MHz) | 134.77 |
| Compute Module latency (N+1 bit) (ps) | 46(N+1) |

Fig. 3. (a) ADRA-based CiM in a generic memory array, (b) $I_{SL}$ values for different input vectors, (c) reference currents for the three sense amplifiers (SAs), (d) table with ADRA-based CiM measurements, (e) add/subtract compute module, (f) Illustration of the target application of ADRA, where a custom instruction can replace three traditional instructions required for a noncommutative function like subtraction and (g) Sense margin vs. $V_{GREAD1}$ with $V_{GREAD2} = 1$V, showing that sufficient sense margin can be obtained for a range of $V_{GREAD1}$ values.

We exploit the fact that the current from each bitcell is dependent on both the bitcell value and the $V_{GREAD}$ its wordline is driven to. Therefore, if we have two different $V_{GREAD}$ values for the two wordlines, unique $I_{SL}$ values can be achieved for each input vector. Fig. 3(a) describes the proposed technique using a generic memory array. During the compute operation, the RBL is driven to $V_{READ}$. The wordline of the first word ($WL_1$) is asserted to $V_{GREAD1}$, and the wordline of the second word ($WL_2$) is asserted to $V_{GREAD2}$, where $V_{GREAD2} > V_{GREAD1}$. $V_{GREAD1}$ and $V_{GREAD2}$ are generated by modifying the inverters in the final stage of the wordline driver, as shown in Fig. 3(a), allowing for the generation of $V_{DD}(V_{GREAD2})$ or $V_{DD} - V_{TH}(V_{GREAD1})$ based on the value of $V_{sel}$. Here, $V_{TH}$ and $V_{DD}$ are the threshold voltage and supply voltage of the FeFET. Fig. 3(b) illustrates how distinct $I_{SL}$ values are obtained for each input vector. Like the prior CiM works, SL is connected to the positive input of the SAs with appropriate references (shown in Fig. 3(c)) to compute different functions. Since there are 4 different $I_{SL}$ values, an additional SA is needed (i.e., in addition to the two SAs used in standard CiM). The reference current $I_{REF-B}$ is placed between $(I_{L1}+I_{H2})$ and $(I_{H1}+I_{L2})$. Thus, this additional SA outputs the value of the bit 'B', which is the word in the row with the wordline voltage $= V_{GREAD2}$. The other two SAs are used to compute the bitwise OR/NOR and AND/NAND of the two words, like the prior CiM works. Likewise, ADRA can be used with voltage-based sensing by comparing the voltage discharge on the senseline to appropriate references. The additional SA required by ADRA has a minute contribution to the already large bitline capacitance, leading to negligible impact on the read/write performance.

By using the outputs of the three SAs and an additional OR-AND-INVERT (OAI) gate, A can be computed:

$$A = \overline{\overline{AB}(B + \overline{A + B})} = OAI21(B, \overline{A + B}, \overline{AB}) \quad (1)$$

Since both A and B are obtained, a single cycle 2-bit read operation is possible with the proposed technique, along with simultaneous computation of AND and OR. Once both A and B are available, a compute module (CM) can be integrated near the memory to compute any Boolean and arithmetic functions. However, it is important to mention that the CM's complexity must be kept in mind while deciding the functions to be computed. Nevertheless, ADRA achieves full-scale digital CiM of functions with two in-memory-operands, and thus offers extensive flexibility in designing the CiM engine as per the system's needs. For example, our approach enables the CiM of important class of functions, such as subtraction and comparison (details in the next sub-section), which are crucial for many mainstream and emerging workloads.

### B. In-Memory Subtraction and Comparison

In this section, we utilize the one-on-one mapping between input vectors and senseline currents to enable single cycle CiM of subtraction and comparison. A compact CM which can perform both addition and subtraction is shown in Fig. 3(e). The inputs to the CM are the outputs of the three SAs: OR (A+B), AND (AB) and B, and their compliments. Having B as an input allows for the computation of $A\overline{B}$, which is necessary for computing the subtraction (A-B). An additional 'SELECT' enables selection between the computation of addition ('SELECT' = '0') and subtraction ('SELECT' = '1'). The proposed module has additional two 2:1 multiplexers, one NOT and one NOR gate compared to the CMs used in prior CiM works (Fig. 1(d)). Our area estimates based on scalable CMOS rules show that ADRA incurs an area increase from 6.21% (1024x1024 array) to 12.12% (256x256 array) over previous CiM, considering full parallelism. However, if the peripherals can be shared amongst the columns (as in some architectures), the area overhead ranges from 4.07% to 6.32% (considering 4:1 column multiplexing). As an alternate design of the CM, the two additional multiplexers can be replaced by an XOR and an AND-OR-INVERT (AOI) gate to obtain *both* subtraction and addition outputs. This design has a 4 transistor overhead compared to the former but enables single-cycle CiM of both addition and subtraction.

The *SUM* values of all the CMs constitute the addition/subtraction outputs. The input carry bit $C_{IN}$ of the first stage is '0/1' for addition/subtraction. The *CARRY* of the previous stage is propagated as the $C_{IN}$ of the next stage. $n+1$ CMs are used for every $n$ bit subtraction. The additional $(n+1)^{th}$ CM is used for overflows and has $C_{IN} = CARRY$ of the $n^{th}$ CM. Since the inputs being subtracted are in two's complement form and can be sign-extended, the other two inputs to the $(n+1)^{th}$ CM are the same as that to the $n^{th}$ CM.

With in-memory subtraction, CiM of comparison can be achieved seamlessly. Since both input and output are in two's complement form, the value of the most significant bit of the subtraction output (SUM of the $(n+1)^{th}$ CM) can reveal the larger number. If both the inputs are equal, all output bits will be zero, which can be detected by a near-memory AND gate

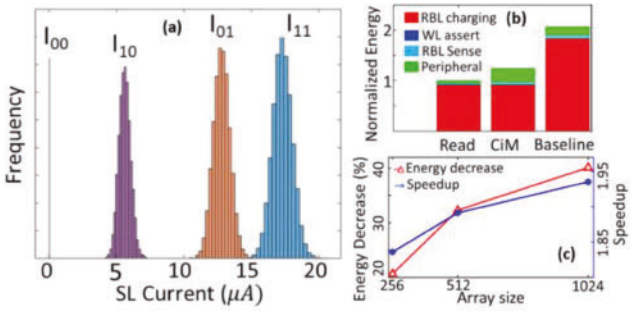



Fig. 4. (a) Frequency of $I_{SL}$ values during ADRA-based CiM, (b) energy components of the read, ADRA-based CiM and the baseline and (c) energy decrease and speedup of ADRA-based CiM as a function of the array size for current-based sensing.

tree. For an $n$ bit comparison, $n-1$ two-input AND gates are needed for the AND tree, adding an overhead of just 1 gate per column. Thus, the CiM of comparison is enabled through the CiM of subtraction with minimal overheads.

## IV. Results

In this section, we analyze in-memory subtraction/comparison using ADRA in the context of both current and voltage-based sensing (since previous CiM works have considered both sensing schemes [3], [4]) We evaluate the technique on a 1T FeFET NVM array using the following bias conditions: $V_{READ}$ (RBL) = 1V; $V_{DD} = V_{GREAD2} =$ 1V; $V_{GREAD1} = 0.79$V; $V_{SET} = 3.7$V and $V_{RESET} = -5$V. Our simulations are performed using PTM 45nm high performance models, hence $V_{GREAD1}$ is $V_{DD} - V_{TH} = 0.79$V. We obtain sense margin >50mV and >1μA (Fig. 3(g)) and thus sufficient distinguishability for both voltage and current-based sensing for these values of $V_{GREAD1}$ and $V_{GREAD2}$, further verified by the $V_{TH}$ variation analysis done (Fig. 6(a) and 4(a) respectively) with $\sigma = \frac{20\ mV}{\sqrt{W/W_{\min}}}$ [23]. We also evaluate speedup/energy benefits and trade-offs compared to the baseline, which needs two memory accesses and near-array subtraction (recall previous CiM approaches can't perform a single cycle subtraction [11], [15]).

### A. Current-Based Sensing

Fig. 4(c) shows the energy decrease and speedup of ADRA-based CiM over the near-memory baseline as a function of the array size. For array size of 1024 × 1024, ADRA-based CiM is 1.94x faster and uses 41.18% lesser energy than the baseline, resulting in an energy-delay product (EDP) decrease of 69.04%. The near 2× speedup is because ADRA-based CiM and the baseline require 1 and 2 cycles for subtraction respectively. The energy benefits can be understood by examining the various energy components of the standard read and proposed CiM operation. Fig. 4(b) shows the energy components per column in a 1024 × 1024 1T FeFET array corresponding to RBL charging/discharging, flow and sensing of read current, word line charging/discharging, and peripheral circuitry (SAs and compute module). The RBL charging is the primary energy component for both read and CiM, comprising 91% and 74% of the total energy, respectively. Since the peripheral circuit energy is a minor component, the energy overheads of the additional hardware in the compute module leads to only a small increase in the overall energy. Our analysis shows that the CiM operation expends 1.24× the energy of the standard read operation. However, subtraction in the baseline designs would require two reads and a compute. Thus, ADRA-based CiM leads to a 41.18% decrease in energy. Also, both the speedup and energy benefits of ADRA increase with increase in array size, because RBL charging energy becomes more dominant with increased array size, amortizing the energy/latency overheads of the ADRA peripherals.

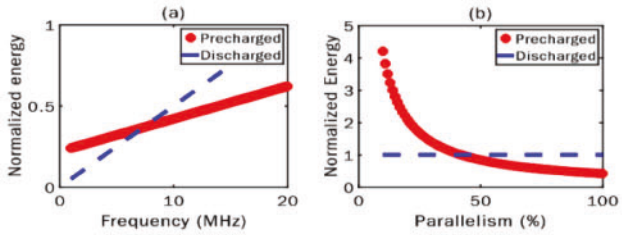



Fig. 5. Energy comparison for CiM using precharged (scheme 1) and discharged RBL (scheme 2) voltage sensing schemes as a (a) function of frequency of CiM operation and (b) parallelism of CiM operation.

### B. Voltage-Based Sensing

Various CiM architectures have used voltage-based sensing for read and compute [4], [15]. Two versions of voltage-based sensing are commonly used: (1) keeping the read bitline (RBL) precharged to $V_{DD}$ during the hold operation and (2) discharging RBL to 0 during hold and charging it to $V_{DD}$ for every read/compute operation. We will refer to these as scheme 1 and scheme 2 respectively. In scheme 1, the RBL charging energy per operation is relatively small compared to scheme 2. However, the array expends leakage energy during hold due to the pre-charged RBLs. Therefore, for low frequency systems, where the number of CiM operations per second is low, the extra energy spent in scheme 2 for charging the bitlines would be balanced by the savings in leakage energy. Fig. 5(a) displays this trade-off for ADRA showing that at frequencies below 7.53 MHz, scheme 2 is more energy efficient.

Another crucial aspect when analyzing the two voltage sensing schemes is the parallelism ($P$) in the CiM operation. Let's define $P$ as $N_{CiM}/N_T$ where $N_{CiM}$ is the number of words in a row on which the CiM operation is being performed parallelly and $N_T$ is the total number of words stored in a row. For example, CiM on a *single* word and on all words would correspond to $P = 1/N_T$ and $P = 1$, respectively. Now, since the wordline is common to all the words in the row, the words not involved in the computation are still accessed. This is especially critical for scheme 1, in which the half-selected words effectively go through a 'pseudo CiM' operation (like pseudo-read [24]). Hence, some energy is wasted in charging the bitlines of the *half-selected* words, as they may be discharged during pseudo-CiM. In contrast, scheme 2 does not suffer from this energy overhead since only the *selected* RBLs are charged during read/compute. Thus, scheme 2 is more suitable for arrays with a low $P$(< ∼42% in Fig. 5(b)).

Fig. 6(c) shows the performance comparison of ADRA compared to the baseline for scheme 1. It is seen that the speedup ranges from 1.57×−1.73× and increases with increase in array size. However, the CiM computation costs 20-23% more energy than the baseline. Examination of the energy components (Fig. 6(b)) of the standard read and the CiM operations shows that the bitline charging energy is the most dominant energy component for both read and CiM. In current-based sensing, the bitline charging energy components for standard read and CiM have negligible difference.

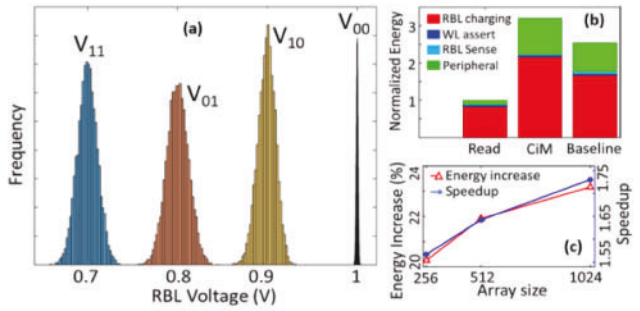
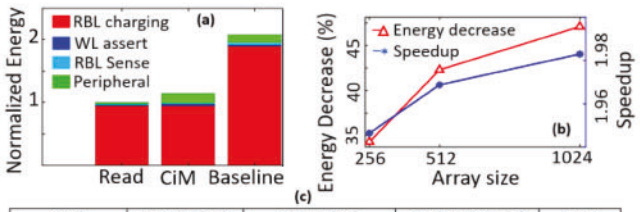



Fig. 6. (a) Frequency of $V_{RBL}$ values for both voltage-based sensing schemes, (b) energy components of read, ADRA-based CiM and baseline and (c) energy decrease and speedup of ADRA-based CiM as a function of array size for precharged RBL voltage sensing (scheme 1).




| Work | IPS-CiM [11] | CiM3D [12] | Push-Rule 6T [5] | ADRA |
|---|---|---|---|---|
| CiM operations | Commutative + Comparison | Comparison (many to one) | ALL | ALL |
| Tech. agnostic | No | No | No | Yes |
| Bitcell | 3T FeFET | 6T + 3T SRAM (3D) | split WL 6T SRAM | 1T FeFET |
| Tech. node | 45 nm | 20 nm | 28 nm | 45 nm |
| Energy | 1.02 | 8.93 | 1.19 | **1** |
| Latency | 1.75 | 12.38 | **0.98** | 1 |
| EDP | 1.78 | 110.55 | 1.16 | **1** |
| Area | **0.91** | 1.52 | 1.56 | 1 |

Fig. 7. (a) energy components of read, ADRA-based CiM and baseline, (b) energy decrease and speedup of ADRA-based CiM as a function of array size for discharged RBL voltage sensing (scheme 2), (c) table comparing ADRA with related CiM works (normalized).

However, for scheme 1, the CiM bitline charging energy is approximately 3x that of read due to the following reason: Let $\Delta$ be the sense margin of the voltage SA. For the standard read operation (in a single ended memory), the bitline will have to discharge by $2\Delta$ to reliably distinguish between the *two* possible values (0 and 1). However, in ADRA, the bitline will need to discharge by $6\Delta$ to reliably distinguish between the *four* possible input vectors. Thus, compared to the baseline (two reads + compute), ADRA exhibits an energy overhead of 1.5× in the bitline energy component. However, since the peripheral circuit energies for both read and compute are similar, ADRA-based CiM using scheme 1 has a net 20-23% energy overhead and 1.57×−1.73× speedup over the baseline, leading to a 23.26 - 28.81% decrease in EDP. Conversely, for scheme 2, the implications of ADRA are different (See Fig. 7 (a-b)). ADRA-based CiM has a speedup of 94.5 - 98.3% and expends 35.5 - 45.8% lesser energy than the baseline, leading to a 66.83 - 72.6% decrease in EDP. This is because the RBLs are charged before the read/CiM, making the RBL energy component similar for standard read and ADRA. Therefore, ADRA's effect on the overall performance are similar to current sensing.

## V. Conclusion

In this brief, we propose ADRA, an asymmetric dual row assertion scheme. It achieves a single-cycle 2-bit read that enables full scale digital CiM of functions with two in- memory operands. ADRA maps each input vector to a unique current or voltage value on the senseline, allowing the CiM of non-commutative functions such as subtraction and comparison in a single cycle, in addition to earlier implemented commutative functions. We evaluate ADRA on a 1T FeFET array for a current-based and two voltage-based sensing schemes. We show that ADRA exhibits 23.2 - 72.6% decrease in EDP compared to the near-memory baseline. Further, comparison with existing CiM works (Fig. 7(c)) displays that ADRA furnishes 13.79 - 99.10% lower EDP, making it an attractive solution for full scale digital CiM.

## References

[1] W. A. Wulf and S. A. McKee, "Hitting the memory wall: Implications of the obvious," *ACM Comput. Arch. News*, vol. 23, no. 1, pp. 20–24, 1995.
[2] S. Yin et al., "XNOR-SRAM: In-memory computing SRAM Macro for binary/ternary deep neural networks," *IEEE J. Solid-State Circuits*, vol. 55, no. 6, pp. 1733–1743, Jun. 2020.
[3] S. Jain et al., "Computing in memory with spin-transfer torque magnetic RAM," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 3, pp. 470–483, Mar. 2018.
[4] A. Agrawal et al., "X-SRAM: Enabling in-memory Boolean computations in CMOS static random access memories," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 12, pp. 4219–4232, Dec. 2018.
[5] S. Jeloka et al., "A 28 nm configurable memory (TCAM/BCAM/SRAM) using push-rule 6T bit cell enabling logic-in-memory," *IEEE J. Solid-State Circuits*, vol. 51, no. 4, pp. 1009–1021, Apr. 2016.
[6] S. Aga et al., "Compute caches," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, 2017, pp. 481–492.
[7] S. K. Thirumala et al., "Non-volatile memory utilizing reconfigurable ferroelectric transistors to enable differential read and energy-efficient in-memory computation," in *Proc. IEEE/ACM Int. Symp. Low Power Electron. Design (ISLPED)*, 2019, pp. 1–6.
[8] J. Borghetti et al., "'Memristive' switches enable 'stateful' logic operations via material implication," *Nature*, vol. 464, pp. 873–876, Apr. 2010.
[9] K. M. Kim and R. S. Williams, "A family of stateful memristor gates for complete cascading logic," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 11, pp. 4348–4355, Nov. 2019.
[10] L. Cheng et al., "In-memory digital comparator based on a single multivalued one-transistor-one-resistor memristor," *IEEE Trans. Electron Devices*, vol. 67, no. 3, pp. 1293–1296, Mar. 2020.
[11] S. K. Thirumala et al., "IPS-CiM: Enhancing energy efficiency of intermittently-powered systems with compute-in-memory," in *Proc. IEEE 38th Int. Conf. Comput. Design (ICCD)*, 2020, pp. 368–376.
[12] A. K. Ramanathan et al., "CiM3D: Comparator-in-memory designs using monolithic 3-D technology for accelerating data-intensive applications," *IEEE J. Explor. Solid-State Computat. Devices Circuits*, vol. 7, no. 1, pp. 79–87, Jun. 2021.
[13] S. Kvatinsky et al., "MAGIC—Memristor-aided logic," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 61, no. 11, pp. 895–899, Nov. 2014.
[14] S. George et al., "Nonvolatile memory design based on ferroelectric FETs," in *Proc. 53rd ACM/EDAC/IEEE Design Autom. Conf. (DAC)*, 2016, pp. 1–6.
[15] D. Reis et al., "Computing in memory with FeFETs," in *Proc. Int. Symp. Low Power Electron. Design (ISLPED)*, 2018, pp. 1–6.
[16] D. Reis et al., "Design and analysis of an ultra-dense, low-leakage, and fast FeFET-based random access memory array," *IEEE J. Explor. Solid-State Computat. Devices Circuits*, vol. 5, no. 2, pp. 103–112, Dec. 2019.
[17] K. Ni et al., "A circuit compatible accurate compact model for ferroelectric-FETs," in *Proc. IEEE Symp. VLSI Technol.*, 2018, pp. 131–132.
[18] S. L. Miller and P. J. McWhorter, "Physics of the ferroelectric nonvolatile memory field effect transistor," *J. Appl. Phys.*, vol. 72, no. 12, p. 5999, 1992.
[19] X. Li et al., "Design of 2T/cell and 3T/cell nonvolatile memories with emerging ferroelectric FETs," *IEEE Design Test*, vol. 36, no. 3, pp. 39–45, Jun. 2019.
[20] "Predictive technology model." Accessed: May 2022. [Online]. Available: http://ptm.asu.edu/
[21] S. Deng et al., "Overview of ferroelectric memory devices and reliability aware design optimization," in *Proc. Great Lakes Symp. VLSI (GLSVLSI)*, 2021, pp. 473–478.
[22] A. K. Saha and S. K. Gupta, "Modeling and comparative analysis of hysteretic ferroelectric and anti-ferroelectric FETs," in *Proc. 76th Device Res. Conf. (DRC)*, 2018, pp. 1–2.
[23] K. Agarwal and S. Nassif, "Characterizing process variation in nanometer CMOS," in *Proc. 44th ACM/IEEE Design Autom. Conf.*, 2007, pp. 396–399.
[24] T.-H. Kim et al., "A 0.2 V, 480 kb subthreshold SRAM with 1 k cells per bitline for ultra-low-voltage computing," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 518–529, Feb. 2008.