# FeFET-Based Logic-in-Memory Supporting SA-Free Write-Back and Fully Dynamic Access With Reduced Bitline Charging Activity and Recycled Bitline Charge

Wenjun Tang, *Graduate Student Member, IEEE*, Mingyen Lee, *Member, IEEE*, Juejian Wu, *Student Member, IEEE*, Yixin Xu, Yao Yu, Yongpan Liu, *Senior Member, IEEE*, Kai Ni, *Member, IEEE*, Yu Wang, *Fellow, IEEE*, Huazhong Yang, *Fellow, IEEE*, Vijaykrishnan Narayanan, *Fellow, IEEE*, and Xueqing Li, *Senior Member, IEEE*

***Abstract*— Bitwise logic-in-memory (BLiM) is a promising approach to efficient computing in data-intensive applications by reducing data movement between memory and processing units. However, existing BLiM techniques have challenges towards higher energy efficiency and speed: (i) DC power in computing and result sensing is significant in most existing RRAM and MRAM based BLiM solutions; (ii) before the computation result could be stored back to the same memory array, existing BLiM has to sense the result first, at the cost of extra power and latency due to the sense amplifiers (SAs). Targeting at higher energy efficiency and speed, this work proposes a new BLiM approach in 2-transistor/ cell (2T/C) and 3T/C topologies based on ferroelectric field-effect transistors (FeFETs), supporting a variety of computing functions. For the first time, this new approach supports SA-free direct write-back, and consumes no static power for computing and sensing with proposed fully dynamic computing and sensing schemes. Another highlight is that this work further minimizes the dynamic power by (i) reducing the chance of bitline charging activities and (ii) recycling the bitline charge in sensing multi-operand operations. Compared with prior BLiM methods based on nonvolatile memories, evaluation shows 3.0x–100x latency and 1.3x–200x energy improvement for typical in-memory XOR operation, which further leads to 3.0x–58x and 3.2x–78x savings of latency and energy, respectively, for the application of advanced-encryption standard (AES).**

Manuscript received 20 September 2022; revised 29 January 2023; accepted 27 February 2023. Date of publication 7 March 2023; date of current version 30 May 2023. This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFA0706100, in part by the NSFC under Grant U21B2030 and Grant 92264204, in part by the Tsinghua University–Daimler Greater China Ltd. Joint Institute for Sustainable Mobility, and in part by the NSF under Grant 2008365 and Grant 2132918. This article was recommended by Associate Editor Y. Zhang. *(Wenjun Tang and Mingyen Lee contributed equally to this work.) (Corresponding author: Xueqing Li.)*

Wenjun Tang, Mingyen Lee, Juejian Wu, Yongpan Liu, Yu Wang, Huazhong Yang, and Xueqing Li are with the Beijing Information Science and Technology National Research Center (BNRist), Department of Electronic Engineering, Tsinghua University, Beijing 100084, China (e-mail: xueqingli@tsinghua.edu.cn).

Yixin Xu and Vijaykrishnan Narayanan are with the Department of Computer Science and Engineering, Penn State University, University Park, PA 16802 USA (e-mail: vijay@cse.psu.edu).

Yao Yu is with Daimler Greater China Ltd., Beijing 100102, China (e-mail: yao.y.yu@mercedes-benz.com).

Kai Ni is with the Department of Microsystems Engineering, Rochester Institute of Technology, Rochester, NY 14623 USA (e-mail: kai.ni@rit.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSI.2023.3251961.

Digital Object Identifier 10.1109/TCSI.2023.3251961

***Index Terms*— Computing-in-memory, processing-in-memory, logic-in-memory, FeFET, ferroelectric, nonvolatile memory.**

## I. Introduction

THE "memory wall" problem has become the performance bottleneck of the conventional Von Neumann architectures in data-intensive applications [1]. The separated computing units and memories, along with limited memory bus bandwidth, cause high latency and energy consumption. One promising approach is computing-in-memory (CiM), which may break the memory access barrier and improve the system performance with reduced data transfer activities and increased processing parallelism [2], [3], [4], [5], [6], [7], [8], [9]. As a subset of CiM, bitwise logic-in-memory (BLiM) has wide applications, e.g., database, encryption, and image processing.

BLiM techniques have been proposed in various types of memories, including SRAM [3], DRAM [10], and emerging nonvolatile memories (NVMs) [4], [5], [6], [7], [8], [9]. NVMs, such as RRAM [4], [5], STT-MRAM [6], [7], [8], and PCM [9], exhibit appealing characteristics including higher density and non-volatility for BLiM, when compared with CMOS-only implementations. Among existing NVM technologies, the emerging ferroelectric FETs (FeFETs) have attracted increasing attention in low-power system designs, because FeFETs exhibit DC-power-free write capability, an ultra-high ON/OFF ratio, and excellent CMOS-process compatibility. Several pioneering FeFET-based memory-logic-synergy works make use of the FeFET device advantages [11], [12], [13], [14], [15], [16]. However, these designs still suffer from high-power sensing complexity and power [11], [15], [16], low generality [12], or non-array access [13]. To achieve lower power, latency, and complexity, this work re-thinks BLiM from a few new perspectives:

First, *SA-free direct write-back capability*. The computing results generated on the bitlines are often needed to be stored to the same memory array. Prior BLiM works usually rely on a complex CiM interface to convert the analog computing results to digital bits, and then apply a subsequent write-back. As the energy and latency costs of such a sense-then-write-back operation are usually high, there is a fundamental question: is there a direct-write-back BLiM paradigm without the costly sensing and write-back driving? The question is well

addressed in this paper by a proposed direct-write-back design without bitline sensing and driving. It is achieved by reusing the remnant bitline charge with a tactical wordline control.

Second, *DC-free dynamic peripherals*. Existing BLiM works rely heavily on the operations in the current domain [11]. The direct compute and sense currents along with one or more current references increase the total energy consumption significantly. In contrast, this work targets at ultra-low-power BLiM in the voltage-charge domain without consuming DC power. Furthermore, as now bitline capacitance charging consumes most power, this paper further pursues bitline charge saving and reusing techniques for even higher energy efficiency.

Third, *support for multi-operand operations*. There are computing scenarios with multiple input operands, such as encryption algorithms and Hamming coding. Prior works limited by large device variations during current-mode sensing usually break the computing into several 2-input tasks, resulting in extra overheads. In this work, the proposed BLiM operations could practically support over 3 activated inputs. It is achieved by a proposed capacitive coupling peripheral circuitry that exploits the FeFET unique ultra-high ON/OFF ratio.

In summary, this work on low-power FeFET-based BLiM has the following highlighted features:

- *BLiM architecture with both 2T/C and 3T/C cell design*: A compact DC-power-free architecture, including the memory array, sensing interface and peripherals, is proposed. The 2T/C design has higher density. The 3T/C design has more flexibility and higher reliability. Both designs are evaluated.
- *Low-power voltage-domain operations*: This voltage-domain computing capability takes advantage of the FeFET features and achieves lower power than the current domain. Bitline charging activities are optimized for even lower computing energy.
- *Direct write-back scheme*: A category of BLiM operations supporting sensing-free direct write-back is proposed, which reuses remnant bitline charge and avoids extra driving. For other operations, a default write-back data path is provided.
- *Low-cost interface*: The dynamic voltage-mode sense amplifier (SA) is area-efficient and low-power for standby and sensing. It also supports multi-variable custom logic and achieves high energy efficiency with charge reuse.
- *Rich functions*: A variety of one- or multi-input BLiM operations are proposed, including basic logic operations COPY, NOT, (N)AND, (N)IMP, XOR2, etc., and custom logic multi-input operations (N)OR, XOR, sum of product (SOP), LUT, etc.
- *Evaluation*: Functionality verification and performance benchmarking of the proposed BLiM show significant energy and latency advantages over prior NVM-based BLiM designs. The proposed method is also applied to AES as a case study.

In the rest of this paper, Section II introduces FeFETs and related CiM progress. Section III presents the proposed BLiM architectures and circuit details. Sections IV and V present the BLiM logic operations. Section VI evaluates the designs. Section VII concludes this work.

## II. Background

This section introduces the FeFET device basics and recent efforts in FeFET-based CiM, and highlights the opportunities and challenges in FeFET-based BLiM.

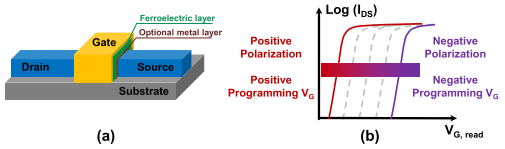



Fig. 1. FeFET [21]. (a) A FinFET structure; (b) Typical I-V curves.

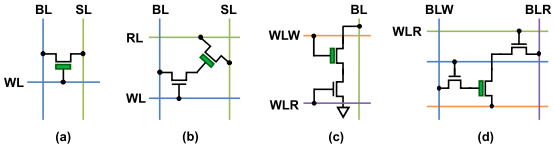



Fig. 2. FeFET memory cells. (a) a 1T/C design [27]; (b) a 2T/C design [28]; (c) another 2T/C design [29]; (d) a 3T/C design [29]. BL: bitline; WL: wordline; SL: select line; RL: read line; WLW: write wordline; WLR: read wordline; BLW: write bitline; BLR: read bitline.

### A. FeFET Device Basics

FeFETs have been proposed as a promising beyond-CMOS device, which is essentially a MOSFET with a ferroelectric layer integrated into the gate [17], [18], [19], [20]. Fig. 1(a) shows the FinFET structure of FeFETs [21]. To achieve ferroelectricity, doped $HfO_2$ is being actively explored and shows excellent compatibility with advanced CMOS technologies [22], [23].

As a nonvolatile device, FeFET stores the data as the polarization of the ferroelectric layer, which tunes the transistor threshold voltage ($V_{TH}$). The polarization can be programmed by applying voltage pulses of a certain amplitude and duration at the gate [24]. Fig. 1(b) shows typical I-V curves corresponding to different polarizations. With this feature, both multi-level and single-level memories could be achieved [25].

Compared with other NVM devices such as RRAM, PCM, and STT-MRAM, FeFET exhibits new opportunities. First, FeFET provides separate read and write paths. During the write process, no DC power is consumed as the FeFET gate acts as a capacitive load, leading to low power compared to two-terminal resistive NVM devices. Second, FeFET can achieve an ultra-high ON/OFF ratio, e.g., $10^6$ [26], which provides better state distinguishability, reduces leakage current, and enables a friendly scaling to a large array. Besides, this ON/OFF difference enables energy-efficient voltage-domain operations with a full swing. Third, FeFET could also work as a transistor apparently, thereby enabling convenient integrated logic and memory.

Several FeFET NVM designs have been reported. Some of them are summarized in Fig. 2. In [27], an ultra-dense AND-type 1T/C design is proposed. The work in [28] proposes a crossbar-style 2T/C design. The work in [29] proposes a 2T/C and a 3T/C design, with balanced write disturbance, power supply overheads, and power consumption. The designs in [29] show the opportunity for efficient BLiM cell design, which is the start point of this work.

### B. Existing FeFET-Based CiM Designs

Several FeFET-based memory-logic synergy designs have been proposed, ranging from application-specific designs to more general ones that focus on basic logics.

First, ternary content-addressable memory (TCAM) is a data-driven approach to search operations in the memory array. Thanks to the three-terminal structure, FeFET-based

TCAMs achieve ultra-high density compared to other designs [12], [30].

Second, nonvolatile flip-flop (nvFF) is capable of restoring the state during a power outage, making it appropriate for nonvolatile computing in edge devices. Benefiting from low-power write operations, FeFET-based nvFF designs could save the backup and restore energy by several orders of magnitude compared with RRAM-based and MTJ-based nvFF [31], [32], [33], [34]. Similarly, nonvolatile SRAM [35] is also proposed to mitigate leakage current by backup and restore operations.

Third, NVMs also show great potential for neural network (NN) accelerators with a large quantity of MAC operations and memory accesses. Compact cells with logic functionalities could be built with FeFETs. Multi-level cells are also promising to design dense and low-power analog synapses. Prior FeFET-based NN accelerator designs include hybrid precision training [36], XNOR cells [37], ternary compute-enabled cells [38], TCAM arrays [39], compact crossbars [40], 3D-NAND structures [41], and modifications of devices [42], [43], etc.

Fourth, FeFET is also promising for application-specific CiM units, including multiplier [44], adder [13], [45], [46], dynamic logic [13], etc. In these works, FeFETs support both computing and data storage with simplified structures, thanks to the CMOS compatibility and the transistor interface.

For general BLiM, prior works explore a wide range of BLiM functionalities with custom SAs. The work in [11] presents an FeFET-based 3T/C BLiM design with a mixed-mode sensing of voltage and current to perform logics including (N)AND, X(N)OR and ADD. The work in [15] and [16] is more flexible, supporting TCAM, BCNN, and CNN acceleration as well as the general BLiM. The work in [14] exploits double-gate control for efficient access and BLiM operations.

Recently, there are several trends of FeFET-based memory-logic or memory-computing synergy designs. First, reconfigurability is explored at both device level [47] and circuit level [48]. Second, the back-end-of-line FeFETs for monolithic 3D integration are exploited with nonvolatile router [49] or in-memory training [50] functionalities. Third, benefitting from the high on-off ratio of FeFET, charge-domain computing is introduced achieving high energy efficiency and computing linearity [51], [52]. Fourth, CiM design and optimization for specific application scenarios are proposed, e.g., reliable BNN [53] and attack-defense GNN [54]. In this work, rich logic functionalities and energy-efficient high-reliability voltage-domain computing are the main focus.

In general, FeFET-based memory-logic synergy has shown promise for low-power memory-oriented applications [55]. However, there is still much space to be explored. Dynamic logic style designs in [13] and [44] suffer from large area. BLiM in [11], [15], [16], and [45] modifies a standard FeFET NVM array to support BLiM but needs complicated SA design, occupies large area, and has a layout fitting issue. What's worse, the current sensing mode introduces significant power consumption [11]. The peripheral in [15] includes multiple operational amplifiers and logic gates, which limits the area efficiency and power efficiency. The work in [45] computes addition without SA, but redundant data storage mapping brings extra overheads.

Faced with these challenges, it could be well understood that a new dense BLiM architecture with a fully dynamic sensing interface and rich computing functions is desired. Furthermore, after eliminating the static power consumption, is it possible to save the dynamic power by reducing bitline charging activities? Is it possible to recycle the remnant bitline charge for further computing? Last and preferably, after a computing operation, is it possible to write back the computing results directly to the array without extra SA sensing and bitline driving? These concerns will be addressed elegantly in this work.

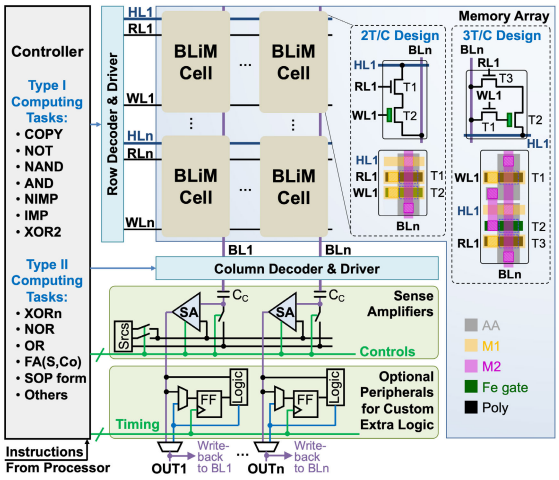



Fig. 3. Proposed FeFET-based BLiM architecture.

## III. Proposed BLiM Architecture and Interface

This section presents the proposed FeFET-based BLiM architecture, including the overall structure, the sensing and computing interface, and optional custom logic support.

### A. Architecture and Functionality Overview

The proposed FeFET-based BLiM architecture is shown in Fig. 3. It includes a memory array, row and column decoders and drivers, a controller, a sensing interface, and optional peripherals. It supports two types of BLiM computing: Type-I operations that process all the inputs in parallel without the need for external logic gates, and Type-II BLiM operations that support more complex logics or multiple inputs.

The computing operations do not destruct the operand data unless a subsequent write-back is applied right to the operand address. Also, as enabled by the structure, the BLiM operations in each column could be performed with full parallelism.

### B. Memory Array

This work adopts single-level per FeFET cell (SLC) to enable a high noise margin for data storage, sensing, and driving. As shown in Fig. 3, for the computable array implementation, we adopt 2-transistor per cell (2T/C) and 3T/C designs. These two cell structures originate from the cache-purpose-only memory array in [29], with a read line RL for reading, a write line WL for writing, and an added horizontal line HL for computing. Although 2T/C and 3T/C designs occupy more area than 1T/C, the computing capability is enhanced with more functionalities, higher reliabilities, and extra direct write-back support.

### C. Dynamic Sensing and Computing Interface

Fig. 4 shows the schematic of the proposed sensing and computing interface. It includes a switched-capacitor ($C_C$)

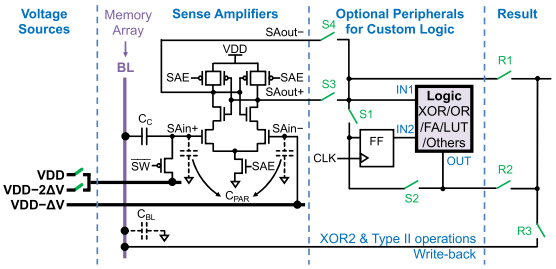



Fig. 4. Schematic of the sensing interface and peripherals.

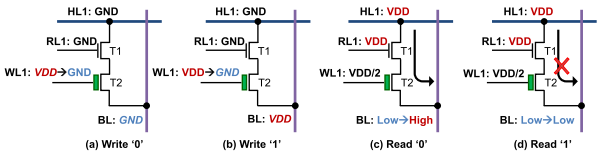



Fig. 5. Write and read operations of 2T/C design. (a) Write '0'; (b) Write '1'; (c) Read '0' by WCB; (d) Read '1' by WCB. Notes: black means wordlines with idle voltages (HL: GND, RL: GND, WL: VDD/2), while red/blue indicate high/low voltages of activated wordlines or bitline output, respectively.

bias scheme, a latch-type voltage-mode SA, optional custom logic control, and the write-back interface. SAs provide a fast-sensing interface for the data interaction with the external, which is controlled by the sense amplifier enabler signal SAE. While the operating flow of sensing and computing will be further explained in detail in Section IV and V, there are a few highlighted features with optimized area, power and latency:

First, the adopted sensing interface operates entirely within the voltage domain, which enables fundamentally higher power efficiency than the prior complicated current-mode SA in [11].

Second, the interface supports multiple computing tasks with only one-time bitline precharge by recycling remnant charge through adaptive $C_C$ bias (controlled by switching signal $\overline{SW}$).

Third, the interface supports optional custom logic for complex computing such as full adder or other look-up-table (LUT) computing (see Section IV-I for details).

Last, as mentioned in Sections I-II and to be further shown in Sections IV-V, for a category of SA-free BLiM computing, there is no need to use the sensing interface, as the remnant bitline charge could be used for subsequent write-back directly.

It is also noted that, we have proposed a switched-capacitor-based dynamic reference generation of SA in our previous work [56], in which the variation of common-mode voltage may affect the performance and reliability. This problem is solved by the proposed input-coupling SA in this work.

### D. Optional Custom Logic Support

While in-memory computing may have flexibility limits, complex logic may benefit from near-memory computing. One of the goals in this work is to exploit the in-memory logic computing capability to reduce the near-memory computing complexity and achieve higher energy efficiency and speed. We propose a low-complexity peripheral circuit design in Fig. 4, which supports a range of multi-variable custom logics.

The implementation makes use of the near-memory logic or LUT. As each column of the memory array has one bitline, a local storage for intermediate variables is needed. One flip-flop (FF) and a LUT or custom combinational logic circuit are added to the peripheral circuit for each column.

To support multiple operands, a multiplexer formed by S1 and S2 selects either the SA output or the previous logic result as the FF input. Either $V_{\mathrm{SAout+}}$ or $V_{\mathrm{SAout-}}$ can be sent to the external or subsequent computing as the normal or inverted logic result, respectively. More details and examples will be provided in Section IV-I.

## IV. Proposed 2T/C BLiM

This section presents the proposed FeFET-based 2T/C BLiM implementation details. For discussion convenience in this paper, a high and low voltage represents the digits '1' and '0', respectively; a negative and positive FeFET polarization leads to positive and negative $V_{TH}$, representing the state '1' and '0', respectively.

### A. Memory Write and Read Operations

The write operation is shown in Fig. 5(a-b). High or low voltage (VDD or GND) corresponding to the write data is set on bitlines. All RLs of the array are grounded during the write process to avoid the influence of HLs. As $V_{GS}$ beyond the coercive voltage $V_{CO}$ will set the polarization of FeFET, the WLs of unselected rows are set to VDD/2 to avoid unexpected write (VDD/2 < $V_{CO}$ < VDD). WLs of selected rows are set to VDD in the first stage and GND for the second stage. Then, the FeFET cell is set to a positive/negative polarization for a GND/VDD bitline voltage in the first/second stage, respectively, where the FeFET $V_{GS}$ exceeds the coercive voltage $V_{CO}$. The set stage is indicated by the italic WL1 and BL voltage in Fig. 5(a)(b). The write of the same data can be applied to multiple destination rows simultaneously, as these extra rows increase the capacitive load slightly but do not affect the correctness of write results.

The voltage-mode read operation could be implemented by either discharging or charging the bitlines through the wordline. The former based on precharged bitlines has been proposed in [29], similar to the CMOS SRAM. The latter is shown in Fig. 5(c-d). The sensing scheme in Fig. 4 supports both charging and discharging methods.

With the sensing circuits, the data can be read out without a full charge or discharge process. Note that when a direct write-back is needed, the write voltage amplitude needs to be sufficiently high and thus a full charge or discharge with longer time may be required.

For the wordline-discharging-bitline (WDB) scheme, the sensing could be carried out in a few stages:

*Stage 1: Initialization*. The HL, RL, and WL of the selected row illustrated in Fig. 5 are set to GND, GND, and VDD/2, respectively, as in the standby state; The bitline (BL) of the memory array is precharged to VDD; SAE is set low, so that the outputs of the SA latch (SAout+ and SAout− in Fig. 4) are both VDD; $\overline{SW}$ is set low and then high to set the input gate voltage $V_{\mathrm{SAin+}}$ to VDD.

*Stage 2: Row selection and bitline discharging.* The RL is set to a VDD pulse. $V_{BL}$ will remain unchanged or decrease if the FeFET is in the OFFor ON state, respectively. More importantly, with the capacitive coupling by $C_C$ in Fig. 4, if $V_{BL}$ decreases to VDD − $2\Delta V$, $V_{\mathrm{SAin+}}$ will decrease by $2\Delta V \times C_C / (C_C + C_{PAR})$, or $\sim 2\Delta V$ if the parasitic $C_{PAR}$ at SA input is much less than $C_C$ in practice; otherwise $V_{\mathrm{SAin+}}$ stays at VDD.

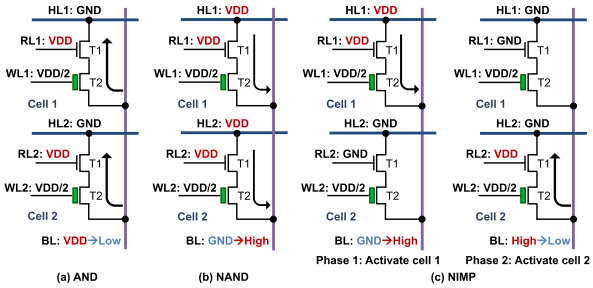



Fig. 6. BLiM operations of (a) and, (b) nand, and (c) nimp.

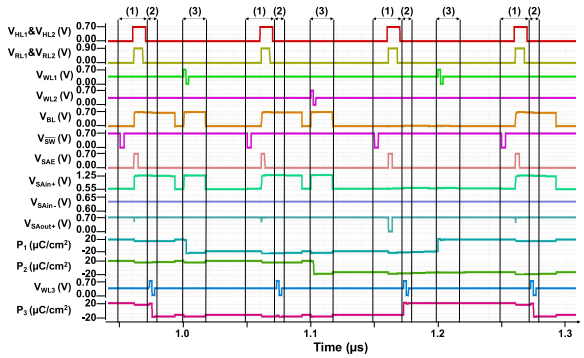



Fig. 7. Proposed 2T NAND2: transient waveforms and FeFET polarization (P.). VDD = 0.7V. (1) NAND between cell 1/2; (2) Write back to cell 3; (3) Set inputs.

*Stage 3: SA sensing and latching*. SAE is set high to compare $V_{\text{SAin+}}$ and $V_{\text{SAin−}}$ (which is VDD $-\Delta V$) and latch the result.

After the memory read, the latched result could be further processed with optional custom logic, and then sent to output, and/or sent to BL for subsequent conventional write-back or further computing. The optional custom logic at the sensing interface provides the opportunity of complex logic such as full adder or other LUT-based computing (see Section IV-I).

It is noted that, *Stage 2* and *Stage 3* could be run again to support more operands after *Stage 3*. This enables the opportunity of multiple computing tasks with only one single BL precharge, as to be further exemplified in Section IV-I.

For the wordline-charging-bitline (WCB) scheme, the sensing procedure consists of a few similar stages:

*Stage 1: Initialization*. The HL, RL, and WL of the selected row illustrated in Fig. 5 are set to VDD, GND, and VDD/2, respectively, as in the standby state; the BL of the memory array is set to GND; SAE is set low, so that the outputs of the SA latch are both VDD; $\overline{\text{SW}}$ is set low and then high to set the input gate voltage $V_{\text{SAin+}}$ to VDD $- 2\Delta V$.

*Stage 2: Row selection and bitline discharging*. The RL is set to a VDD pulse. $V_{\text{BL}}$ will remain unchanged or increase if the FeFET is in the OFFor ON state, respectively. If $V_{\text{BL}}$ increases by $2\Delta V$, $V_{\text{SAin+}}$ will also increase by $\sim 2\Delta V$ (from VDD $- 2\Delta V$ to $\sim$VDD) due to capacitor coupling; otherwise $V_{\text{SAin+}}$ stays at VDD $- 2\Delta V$.

*Stage 3: SA sensing and latching*. SAE is set high to compare $V_{\text{SAin+}}$ and $V_{\text{SAin−}}$ (which is VDD $-\Delta V$) and latch the result.

### *B. Proposed Wordline-Discharging-Bitlines (WDB) Operations*

WDB operations exploit discharging precharged bitlines through the source rows. It generates an AND logic:

$$\text{BL} = \text{A1} \cdot \text{A2} \cdot \ldots \cdot \text{AN}, \tag{1}$$

in which A1, A2, …, AN represent the data of $N$ ($\geq$1) source rows, and BL represents the logic result which can be sensed by SA. As a special case, operation with only one source row corresponds to the read operation.

Fig. 6(a) illustrates the AND operation when $N = 2$ (write-back not shown). The operation can be split into three stages:

*Stage 1:* Precharge all bitlines to VDD. If sensing is required, apply a GND pulse on $\overline{\text{SW}}$ for the initialization of $C_{\text{C}}$.

*Stage 2:* Ground HLs of selected rows. At the same time, turn on the access transistors of the selected rows by setting RLs to VDD. Only when all the selected cells are in the OFF state can the bitline retain VDD, namely A1 = … = AN = 1 and BL = 1. Hold the stage until the bitline can generate a significant voltage difference for SA, or is discharged sufficiently for write-back. Then turn off the access transistors by setting RLs to GND.

*Stage 3:* Enable SA to sense whether the bitline is discharged, then send the result to the external. Alternatively, directly write back the result as to be introduced in Section IV-G.

### *C. Proposed Wordline-Charging-Bitlines (WCB) Operations*

WCB operations exploit charging bitlines through the source row. It generates a NAND logic:

$$\text{BL} = !(\text{A1} \cdot \text{A2} \cdot \ldots \cdot \text{AN}), \tag{2}$$

in which A1, A2, …, AN represent the data of $N$ ($\geq$1) source rows, and BL represents the logic result. Similarly, operation with only one source row achieves the NOT operation. The proposed operation can also be divided into three stages:

*Stage 1:* Ground all BLs to prevent the influence of remnant charge. If sensing is required, apply a GND pulse on $\overline{\text{SW}}$.

*Stage 2:* Set HLs of selected rows to VDD. At the same time, turn on the access transistors of the selected rows by setting RLs to VDD. Only when all the selected cells are in the OFF state can the bitline retain uncharged, namely A1 = … = AN = 1 and BL = 0. Hold the stage until the bitline can generate a significant voltage difference for SA, or is charged sufficiently high for write-back. Then, turn off the access transistors by setting RLs to GND.

*Stage 3:* Enable SA to sense whether the bitline is charged, then send the result to the external. Alternatively, directly write back the result as to be introduced in Section IV-G.

Fig. 6(b) illustrates the proposed NAND operation when $N = 2$ (write-back not shown). Fig. 7 shows NAND transient results, where Step (9) illustrates the proposed direct write-back to be described in Section IV-G.

For a single operation, WDB and WCB have different pros and cons. WCB is more energy-efficient, as the bitlines are not always charged but depend on the ON/OFF state of each FeFET cell in the row being accessed. Consider a single $N$-variable operation with randomly distributed source data and no initial charge on bitlines, when a write-back is needed, WDB always needs a full or high-voltage charging process for the bitlines, while WCB causes a full or high-voltage charging process for only ON-state cells, i.e., $(1 - 1/2^N)$ of charged bitlines on

average. In addition, when only the result sensing is needed, WCB saves charging time and energy because only a sufficient charging of bitline for SA is required. However, WDB achieves higher speed than WCB. For WCB, as the bitline voltage increases, $V_{GS}$ of the access transistor decreases, resulting in a reduced drain-source current, which leads to longer charging latency. In contrast, for WDB, $V_{GS}$ remains VDD during the entire discharging process. Further, WDB can be applied to several specified bits by pre-charging corresponding bitlines (e.g., the lower 8 bits in a 32-bit word), which provides flexibility and higher energy efficiency in some cases (WCB can only be executed on all cells of the selected row).

For logic functionality, WDB and WCB can generate a pair of complementary logic such as {NOT, Read} and {NAND, AND}. Users can choose which strategy to obtain the result considering energy, speed, etc. For multi-input operations, the ON/OFF ratio of the device should be sufficiently high, so that the logic result could be sensed correctly. With a high ON/OFF ratio up to $10^6$ and beyond, FeFETs could support operations with massively-parallel multi-input operations naturally.

### D. Proposed NIMP Operations

The two-variable NIMP (material nonimplication) operation is expressed as:

$$\mathrm{BL} = \mathrm{A2}\ \mathrm{NIMP}\ \mathrm{A1} = \mathrm{A2}\cdot !\mathrm{A1}, \tag{3}$$

in which A1 and A2 represent the data of the two source rows, and BL represents the logic result. Fig. 6(c) gives an example of the proposed NIMP operation (write-back not shown). Different from WDB or WCB operations, the NIMP operation exploits a combined charging-discharging process, which includes three stages:

*Stage 1:* Ground all bitlines to discharge the remnant charge. If sensing is required, apply a GND pulse on $\overline{\mathrm{SW}}$.

*Stage 2:* Set HL1 to VDD. At the same time, turn on access transistor of A1 by setting RL1 to VDD. The bitline is charged when A1 = 0, or remains uncharged when A1 = 1. Then turn off the access transistor of A1 by setting RL1 to GND.

*Stage 3:* Set HL2 to GND. At the same time, turn on access transistor of A2 by setting RL2 to VDD. The bitline is discharged to GND when A2 = 0, or remains the previous voltage when A2 = 1. Therefore, only when A1 = 0 and A2 = 1 can the bitline be charged to a high voltage, otherwise GND. Then turn off the access transistor of A2 by setting RL2 to GND.

*Stage 4:* Enable SA to sense whether the bitline is charged, then send the result to the external. Alternatively, directly write back the result as to be introduced in Section IV-G.

### E. Proposed IMP Operations

Similar to the relation between WDB and WCB operations, the two-variable IMP (material implication) operations can also be implemented by a charging path opposite to NIMP. The IMP logic can be expressed as:

$$\mathrm{BL} = \mathrm{A2}\ \mathrm{IMP}\ \mathrm{A1} = !\mathrm{A2} + \mathrm{A1}, \tag{4}$$

in which A1 and A2 represent the data of the two source rows, and BL represent the logic result. The proposed operation can be split into three stages:

*Stage 1:* Precharge all bitlines to VDD. If sensing is required, apply a GND pulse on $\overline{\mathrm{SW}}$.

*Stage 2:* Set HL1 to GND. At the same time, turn on access transistor of A1 by setting RL1 to VDD. The bitline is discharged when A1 = 0, or remains VDD when A1 = 1. Then turn off the access transistor of A1 by setting RL1 to GND.

*Stage 3:* Set HL2 to VDD. At the same time, turn on access transistor of A2 by setting RL2 to VDD. The bitline is charged to VDD when A2 = 0, or remains the previous voltage when A2 = 1. Therefore, only when A1 = 0 and A2 = 1 can the bitline be discharged to a low voltage, otherwise VDD. Then turn off the access transistor of A2 by setting RL2 to GND.

*Stage 4:* Enable SA to sense whether the bitline is discharged, then send the result to the external. Alternatively, directly write back the result as introduced in Section IV-G.

### F. Other Type-I Logics Operations

The above operations can be divided into four basic operating primitives: (a) initially set the bitline to VDD; (b) initially set the bitline to GND, (c) charge the bitline through an activated cell and (d) discharge the bitline through an activated cell. If we focus on the change of the bitline voltage, these operations can be expressed as assignments: (a) BL:= 1, (b) BL:= 0, (c) BL:= BL + !Ai, and (d) BL:= BL · Ai, where Ai represents the data stored in an activated cell. The combination of such operations can generate a specific range of logics. For example, a logic

$$\begin{aligned}\mathrm{BL} &= !(\mathrm{A1}\cdot\mathrm{A2}\cdot\mathrm{A3})\cdot\mathrm{A4}\cdot\mathrm{A5} = (!\mathrm{A1}+!\mathrm{A2}\\ &\quad +!\mathrm{A3})\cdot\mathrm{A4}\cdot\ \mathrm{A5}\end{aligned} \tag{5}$$

can be implemented as sequence (bcccdd). From this point of view, it is clear that inversing the initial state of bitlines and exchanging (b) and (c) in the process will generate a pair of complementary logic. And in the process, multiple consecutive (b) or consecutive (c) can perform at the same time respectively, as the voltage applied on HLs of activated rows is the same thus no short from VDD to GND is formed.

### G. Direct Write-Back Scheme

The above operations are categorized into Type-I BLiM operations, which means a full swing result can be generated at bitlines with sufficient operation time. The bitline charge is enough to drive write operations of FeFET cells. In other words, direct write-back is possible.

Direct write-back bonds logic operation with a write stage, which reduces an extra memory access instruction and saves energy. For example, a COPY operation is a combination of a read and a direct write-back. Further, the method can write the bitline voltage to multiple destination rows as the write process of an extra destination row consumes negligibly more bitline charge. To achieve a direct write-back, the full-swing bitline voltage is applied in the process by a full charge or discharge of the bitline. At the final stage of the above operations, execute a VDD-GND pulse on one or multiple destination rows so that results can be written back. Fig. 8(a) gives an example of COPY operation. Fig. 9 shows the corresponding transient waveform.

It is noted that, if the logic result is only needed by the external, direct write-back is not executed to save FeFET's lifetime, operation energy and latency. And for a complex logic that a single Type-I operation cannot handle, there are two implementations. The first is to use Type-II operations

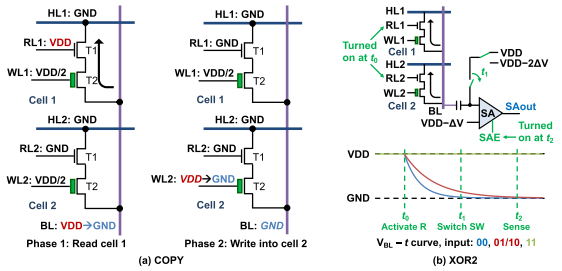



Fig. 8. (a) copy and (b) two-variable xor operations.

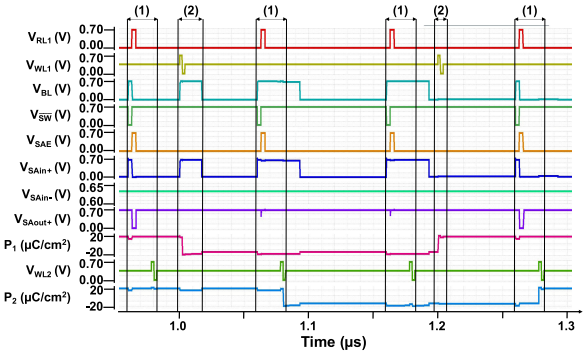



Fig. 9. Proposed 2T copy: transient waveforms and FeFET polarization ($P_1$ and $P_2$) with VDD = 0.7V. (1) copy operation from cell 1 to cell 2; (2) Set inputs.

which can cover arbitrary logic theoretically, as illustrated in Section IV-I. The second is to apply multiple Type-I operations in serial with intermediate result written back, which reduces the area of peripherals for custom logic at the cost of FeFET lifetime.

For WCB operations, due to the drop of a threshold voltage, the voltage on the bitlines cannot be charged to VDD as the wordline. Therefore, it has a higher probability of write failure for the same-VDD-amplitude write pulse [57]. To further improve the performance, an additional pull-up circuit or higher control voltage on RLs is needed for full-swing charging.

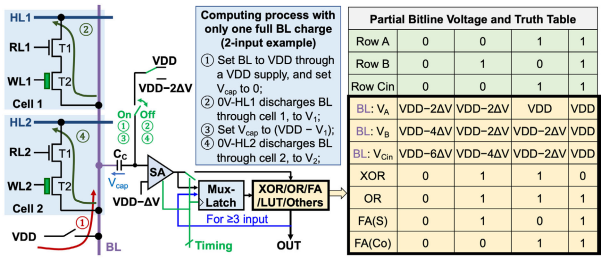
### H. Proposed XOR2 Operation

Exploiting dynamic bias provided by the switched capacitor $C_C$, two-variable XOR (XOR2) can be implemented without an extra logic gate. The method compares bitline voltages at two selected times to generate the logic

$$\text{BL} = \text{A1 XOR A2}, \tag{6}$$

in which A1 and A2 represent the data of the two source rows, and BL represents the logic result.

Fig. 8(b) illustrates the proposed XOR2 operation. First, set $\overline{\text{SW}} = 0$ and precharge all bitlines. Then, activates A1 and A2 simultaneously by setting RL1, RL2 to VDD and HL1, HL2 to GND. At time $t_1$, set $\overline{\text{SW}} = 1$ so that the data input port of SA can trace bitline voltage change, and the results can be sensed at time $t_2$. When inputs are in case '11' or '00', the voltage difference at $t_1$ and $t_2$ is closed to 0, and a positive voltage difference can be observed in case '01' or '10'. Therefore, SA can distinguish the XOR2 logic result. To generate the correct logic, $t_1$ and $t_2$ are selected such that the bitline voltage difference ($V_{t1} - V_{t2}$) is smaller than $\Delta V$ in case '11' or '00', and larger than $\Delta V$ in case '01' or '10'. The result can be written back to the memory array by an additional data path controlled by a transmission gate, as shown in Fig. 4.




Partial Bitline Voltage and Truth Table

| | | | | |
|---|---|---|---|---|
| Row A | 0 | 0 | 1 | 1 |
| Row B | 0 | 1 | 0 | 1 |
| Row Cin | 0 | 0 | 1 | 1 |
| BL: $V_A$ | VDD−2ΔV | VDD−2ΔV | VDD | VDD |
| BL: $V_B$ | VDD−4ΔV | VDD−2ΔV | VDD−2ΔV | VDD |
| BL: $V_{Cin}$ | VDD−6ΔV | VDD−4ΔV | VDD−2ΔV | VDD |
| XOR | 0 | 1 | 1 | 0 |
| OR | 0 | 1 | 1 | 1 |
| FA(S) | 0 | 1 | 0 | 1 |
| FA(Co) | 0 | 0 | 1 | 1 |

Fig. 10. BLiM computing process for x(n)or, (n)or, fa, etc. The bitline voltage is considered as a linear discharge with $\Delta V \ll V_{\text{BL}}$.

### I. Proposed Custom Logic Design: Type-II BLiM Operation

Type-II BLiM operations, referring to a complex custom logic with two or more inputs, are implemented with a flexible peripheral. Type-II operations may contain multiple reads of operands in serial. For full utilization of the bitline charge, an optimized consecutive read scheme is proposed. It exploits dynamic bias by switched capacitor $C_C$ for each bitline and needs less bitline charging in the process. Fig. 10 illustrates the design and the computing process. The complete procedure can be stated as follows:

*Stage 1:* The voltage source for the data input port of SA is set to VDD. Precharge all bitlines to VDD and apply a GND pulse on $\overline{\text{SW}}$. Now the bias $V_{\text{cap}}$ is set to 0.

*Stage 2:* Set HL1 to GND. At the same time, turn on the access transistor of A1 for certain time $\Delta t$. The bitline voltage ($V_{\text{BL}}$) of each column is either discharged below (VDD − $\Delta V$) or kept at VDD depending on the cell state. Then turn off the access transistor of A1.

*Stage 3:* SA senses the result. The result is saved in FF.

*Stage 4:* Apply a GND pulse on $\overline{\text{SW}}$. Now $V_{\text{cap}}$ is set to (VDD $-V_{\text{BL}}$). Because of the charge redistribution, $V_{\text{BL}}$ may be slightly different from the previous stage.

*Stage 5:* Set HL2 to GND and turn on the access transistor of A2 for certain time $\Delta t$. The bitline voltage of each column is either discharged below ($V_{\text{BL}} - \Delta V$) or kept at $V_{\text{BL}}$ depending on the cell state. Then turn off the access transistor of A2.

*Stage 6:* Sense the result with SA, and calculate the result with the logic circuit. For two-variable operations, the BLiM procedure is done. For BLiM of more variables, save the current logic result in FF and repeat *Stage 4* to *Stage 6* until all inputs are read out and calculated. Optionally, the final result can be written back by an additional data path.

Four-variable XOR (XOR4), as an example of a multi-variable function, can be calculated in the process shown in Fig. 11. To implement the function, the custom logic block is configured as a two-variable XOR gate.

As an intuitive extension of the above concept, a read of a single row in the procedure can be replaced by an AND operation of multiple rows, which activates certain rows simultaneously to discharge the bitline. If we configure the custom logic as a two-variable OR gate and store necessary inversion of certain operands by in-memory NOT operations in advance, a sum of product (SOP) form two-level logic

$$\text{Result} = \sum m(a_1, a_2, \ldots, a_{2^N}) \tag{7}$$

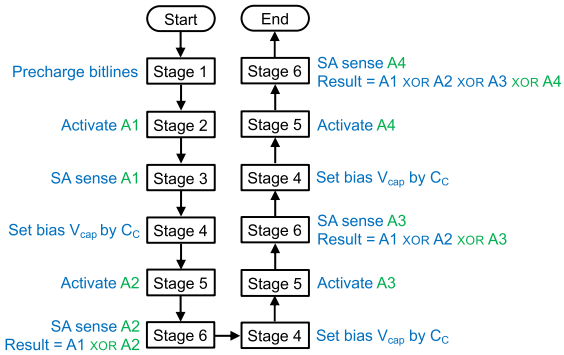



Fig. 11. Flow chart for xor4 computing process.

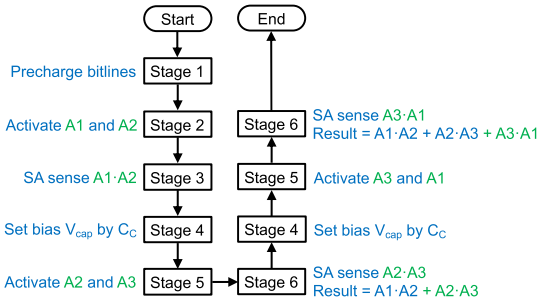



Fig. 12. Flow chart for three-variable majority computing process.

is generated, where $m_i$ stands for a minterm and corresponds to AND operations. Therefore, any Boolean functions can be implemented theoretically. Maximum discharge times of a bitline limit the total occurrences of each input variable in the SOP expression. In addition, the latency of a SOP form is proportional to the number of minterms. As an example, consider a typical complex logic function, which is usually called a three-variable majority:

$$\text{Result} = \text{A1} \cdot \text{A2} + \text{A2} \cdot \text{A3} + \text{A3} \cdot \text{A1}. \tag{8}$$

It can be calculated as shown in Fig. 12, where the custom logic is configured as a two-variable OR gate.

Compared to separate reads, the consecutive reads method saves both energy and latency, as charging bitlines to VDD repeatedly is not required. The switch capacitor $C_C$ can be much smaller than bitline parasitic $C_{BL}$, which means less energy and latency for capacitor charging. For charge utilization, consider a single multi-variable operation with $N$ single-row reads and precharged VDD bitlines initially, separate reads need $\sim (C_{BL} \cdot N \cdot \Delta V)$ more charge than the proposed method in the worst case, as the cost of charging $C_C$ is much less.

However, as the bitline voltage decreases after multiple reads, the discharging speed is lowered. To analyze the issue, we consider the worst case where the FeFETs are all in the ON state, each memory cell has a low constant resistance $R_{ON}$ when activated, and only one row is activated simultaneously. Assume that the switch capacitance $C_C$ is much smaller than bitline capacitance $C_{BL}$, the activation time $\Delta t$ is fixed, and the resistance of each cell can be regarded unchanged during the process. Therefore, the bitline voltage is discharged exponentially as $e^{-t/\tau}$. The maximum number of reads $N_{max}$ is

$$N_{max} = \left\lfloor \frac{\tau}{\Delta t} \log \left( \frac{e^{\Delta t/\tau} - 1}{\Delta V/\text{VDD}} \right) \right\rfloor \tag{9}$$

where $\tau = R_{ON} C_{BL}$ is the time constant of the discharging path and the outer brackets represent the greatest integer less

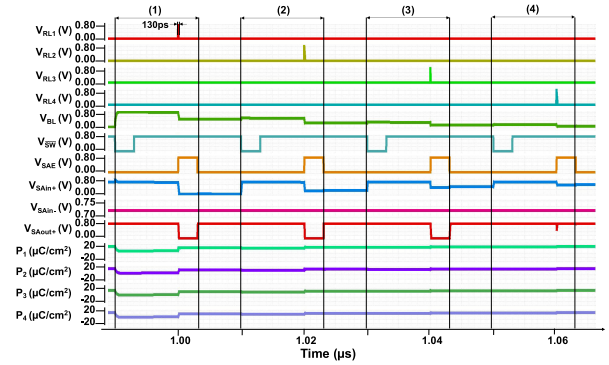



Fig. 13. Transient waveforms of worst-case consecutive read ($N_{max} = 3$, $\Delta t = 130$ ps, VDD = 0.8V). The bottom four curves are the polarization of each cell. (1) First read; (2) Second read; (3) Third read; (4) Fourth read (failed).

than or equal to the content. The activation time $\Delta t$ can be optimized to reach the maximum reads. Fig. 13 shows the simulation result that indicates a maximum number of reads $N_{max} = 3$. Because the transistor works in the saturation region initially, the discharging current keeps in constant. Hence, the actual $N_{max}$ is larger than the estimation using resistance $R_{ON}$ in linear region. To enable a larger number of inputs for a single logic operation, charging bitline to VDD is needed after several consecutive read operations.

For more complex functions such as ADD, data transfers between different columns are required. Simply adding ports that connect the logic block of each column is sufficient, as both two data inputs of the logic block are latched by SA and FF. The SA and FF serve as buffer for intermediate data between operations, which reduce the extra FeFET writes. For improved flexibility and time complexity of logic execution, more buffers can be added as long as the extra area overhead can be tolerated.

### *J. Controller Design for Logic Reconfiguration*

To support the rich logic functionalities with varied control flows, the controller in the proposed BLiM architecture should be well optimized. Besides the basic memory read and write, to handle all the in-memory logic functions, the controller is designed to have the following types of states: (i) instruction decoding and bitline initialization; (ii) BLiM computing; (iii) direct write-back or result readout. In (i), the bitline is discharged or precharged according to the fetched BLiM operation. At the same time, activated rows for each computing state in (ii) are generated and sent to the row decoder sequentially. For example, NOR3 instruction "! (A1 · A2 · A3)" generates a sequence {(A1, A2, A3)}, where only one computing state is required; "A2 NIMP A1" generates {(A1), (A2)}, where the operation needs two computing states to activate A1 and A2 in sequential. In (ii), one BLiM primitive is executed including full WDB (Type-I), full WCB (Type-I), pulsed WDB + SA sensing (Type-I&II), pulsed WCB + SA sensing (Type-I), XOR2 (Type-I), and bias setting (Type-II). One or multiple computing states could be controlled by a finite-state machine (FSM) with reconfigurable LUT logic. The LUT is configured at compile time that includes steps of each logic operation used in a target application (which may include custom logics defined in Section IV-F and Section IV-I). In (iii), the logic result is either directly written back to the FeFET array, transferred to the external, or saved in FF for Type-II operations.

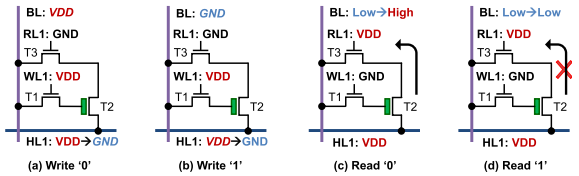



Fig. 14. Write and read operations of 3T/C design. (a) Write '0'; (b) Write '1'; (c) Read '0' by WCB; (d) Read '1' by WCB. Notes: black means wordlines with idle voltages (HL: GND, RL: GND, WL: GND), while red/blue indicate high/low voltages of activated wordlines or bitline output, respectively.

The execution of a single BLiM instruction should not be interrupted. For instructions with a non-write-back state (iii), pipelining between the instructions can be applied to improve the overall throughput. Other possible optimization direction includes instruction reordering for bitline charge recycling, i.e., an instruction with WDB may recycle remnant bitline charge after the previous WCB to further improve energy efficiency. This work focuses on the core BLiM operations and further optimization of instruction execution remains as future work.

## V. Proposed 3T/C BLiM

As mentioned, the 3T/C design has a topology similar to 2T/C in terms of logic operation. In this section, we mainly focus on the differences. The assumption about the meaning of '0' and '1' is the same as the 2T/C design.

### A. Memory Write Operation

Unlike the 2T/C design, the 3T/C design exploits a pulse on the HL to write data rather than the WL. Assume the data to save have been placed on bitlines as VDD or GND. Turn off all RLs of the array and all WLs of unselected rows. Turn on the WLs of selected rows during the process. Set HLs of selected rows to VDD in the first stage and GND for the second stage. Then the FeFET polarization is set to negative/positive for a GND/VDD bitline voltage in the first/second stage, respectively. The simultaneous write of the same data to multiple rows is also applicable for the 3T/C design. Fig. 14 shows the write and read operations.

Compared with the 2T/C design, the write disturb is reduced, thanks to the additional write access transistor. Since the write pulse is set on different terminals of the FeFET, the two designs store complementary data with the same bitline voltage.

### B. BLiM Operations

From the perspective of the BLiM operations ignoring write back, the 3T/C design can work equivalently as the 2T/C design by bypassing (i.e., turning off) the write access transistor. Fig. 15-17 show the transient waveforms of AND, NOT and XOR operation, respectively.

### C. Direct Write-Back Scheme

Corresponding to the write scheme, the direct write-back is performed by setting all RLs to GND, setting the destination WLs to a high voltage and applying a VDD pulse on the HLs of the destination rows. BLiM operations in 3T/C with the same computing procedure as that in 2T/C will write back complementary logic results, where the WCB scheme generates an AND logic, and the WDB scheme generates a NAND logic.

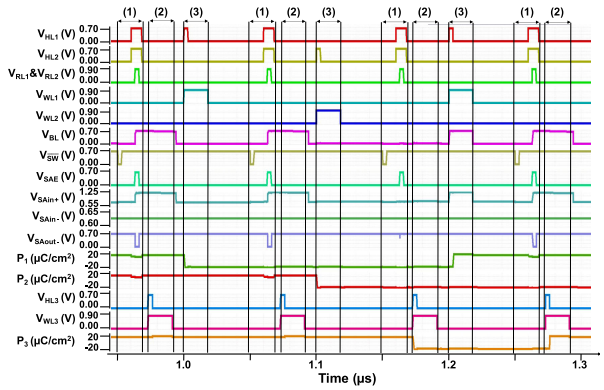



Fig. 15. Proposed 3T and2: transient waveforms and FeFET polarization (P.). VDD = 0.7V. (1) and between cell 1/2; (2) Write back to cell 3; (3) Set inputs.

Slightly different from the 2T/C write-back, the degraded FeFET $V_{GS}$ due to $V_{TH}$ drop can affect the write-back latency and even cause a write failure, especially in the WCB operations. Therefore, we rise the high operating voltage of WLs voltage by 0.2V, and also rise the high operating voltage of RLs voltage by 0.2V in WCB scheme, which can generate sufficient FeFET $V_{GS}$ for write-back. Step (9) in Fig. 15-16 shows the direct write-back scheme of the 3T/C design.

The 3T/C design provides improved reliability regarding the write disturb at the cost of cell density and operation energy, which provides a suitable solution for update-frequent applications. Generally, 2T/C design can be regarded as a compact cell design, and modification of the 2T/C design provides additional functionality or improved performance. Thanks to the memory-embedded transistor interface of FeFET, more cell configurations can be available in the future.

## VI. Evaluation and Discussion

This section evaluates and discusses the proposed designs. In the evaluation, the 10nm PTM FinFET model from [58] is used in all MOSFETs. The FeFET model is the calibrated 10nm model in [59], with 0.01 kinetic coefficient $\rho$ and 10.5nm ferroelectric layer thickness. Besides, a 10fF capacitor is considered as the bitline parasitic capacitance. In the following evaluation, all the data are for a single column.

As the proposed BLiM exploits single-level FeFET for logic operations, and does not rely on the multi-domain dynamics, a single-domain LK model is adopted for efficient circuit behavior analysis and evaluation. The recent multi-domain models can capture more device characteristics, e.g., scalability, variation, stochasticity, and accumulation, more accurately [57], [60]. Several works have presented the difference between the two models [12], [61], where the read/compute latency and energy of the two models are varying in a comparable range (within 2x of energy and within 4x of latency). It is noted that deploying the single-domain model does not change the overall conclusion. As a matter of fact, recent articles indicate that device development such as ferroelectric thickness [62] and gate length [63] scaling, new material [64], and new structure [65] can further improve the FeFET performance, e.g., operating voltage, state switching energy and latency. Therefore, it is expected that the performance of the proposed BLiM may also be updated as the FeFET device continues to be optimized in the future.

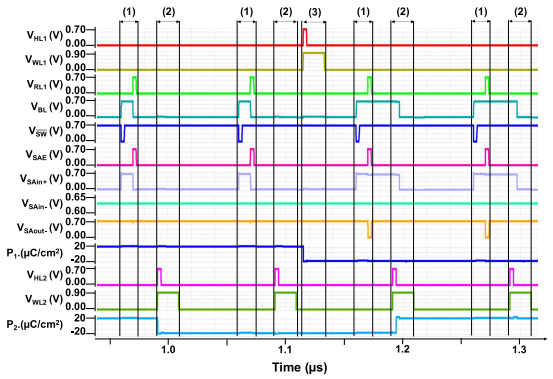



Fig. 16. Proposed 3T not: transient waveforms and FeFET polarization ($P_1$ and $P_2$) with VDD = 0.7V. (1) NOT of cell 1; (2) Write back to cell 2; (3) Set inputs.

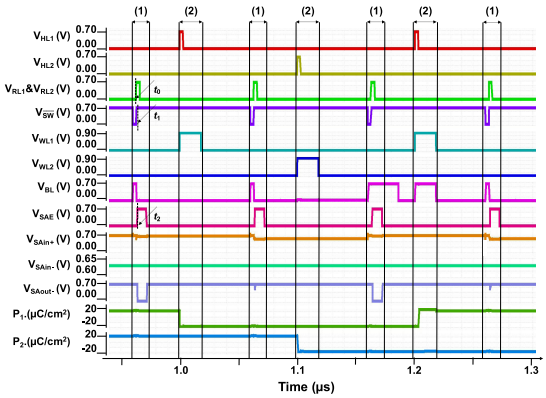


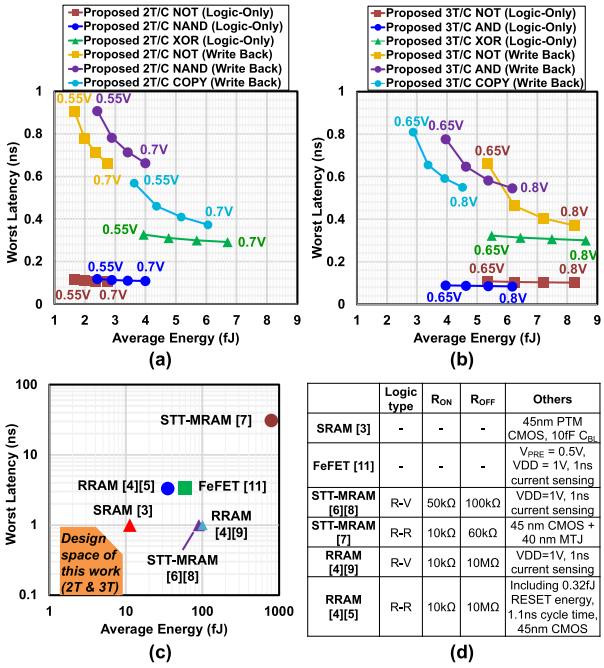

Fig. 17. Proposed 3T xor2: transient waveforms and FeFET polarization ($P_1$ and $P_2$). VDD=0.7V. (1) xor operation between cell 1 and cell 2; (2) Set inputs.

### A. Performance of 2T/C BLiM Operations

Fig. 18(a) shows the latency versus energy of 2T/C BLiM operations including both operation types. All operations are 1-bit two-operand logic and consider all input situations. Energy and latency of sensing peripherals are included. For Type-I operations, the write-back latency depends on the FeFET configuration, which means that the write time could be reduced with device improvement. With fast SA sensing, the latency of logic-only operations is less than 0.35ns. The operating voltage without the need for a direct write-back could be further lowered as long as the sensing peripheral works correctly. Besides, due to the highly parallel activations instead of consecutive reads, Type-I operations save more energy and time compared with Type-II operations in the same operating voltage. The proposed BLiM operations achieve <7fJ energy and <1ns latency in the worst case.

### B. Performance of 3T/C BLiM Operations

3T/C BLiM also includes two types of operations and considers different input situations as shown in Fig. 18(b). Due to the voltage drop of the access transistors, 3T/C needs a higher operating voltage for write and computing operations. It is noted that WCB operations of the 3T/C BLiM (e.g., COPY, AND) require a higher RL and WL voltage beyond VDD. Therefore, more energy and latency are consumed under the same operating voltage compared with corresponding operations of 2T/C (e.g., NOT, NAND).




| | Logic type | $R_{ON}$ | $R_{OFF}$ | Others |
|---|---|---|---|---|
| SRAM [3] | - | - | - | 45nm PTM CMOS, 10fF $C_{BL}$ |
| FeFET [11] | - | - | - | $V_{PRE}$ = 0.5V, VDD = 1V, 1ns current sensing |
| STT-MRAM [6][8] | R-V | 50kΩ | 100kΩ | VDD=1V, 1ns current sensing |
| STT-MRAM [7] | R-R | 10kΩ | 60kΩ | 45 nm CMOS + 40 nm MTJ |
| RRAM [4][9] | R-V | 10kΩ | 10MΩ | VDD=1V, 1ns current sensing |
| RRAM [4][5] | R-R | 10kΩ | 10MΩ | Including 0.32fJ RESET energy, 1.1ns cycle time, 45nm CMOS |

(d)

Fig. 18. Energy-latency evaluation. (a)(b) Proposed BLiM performance; (c) Comparison with in-memory xor2in other works using parameters in (d).

For WDB operations, the bitlines are precharged through the voltage source instead of the access transistor and wordline. Therefore, no RL voltage rise is needed for both 2T/C and 3T/C BLiM. That is why the WDB operations of the proposed two cell designs (e.g., COPY and XOR in 2T/C versus NOT and XOR in 3T/C) reach almost the same energy in the same VDD. Besides, with a higher operating voltage, the 3T/C BLiM has improved latency performance (<8.5ns) but increased energy consumption (<9fJ) compared with the 2T/C BLiM design.

### C. Discussion of Non-Ideal Issues

The proposed BLiM architecture exhibits robustness to multiple non-ideal factors. For Type-I operations, non-ideal issues, such as variation, endurance, temperature and voltage change, mostly affect the latency. The computing accuracy can be guaranteed under non-ideal factors by a sufficiently large on-off ratio of the FeFET device, which is enabled by the voltage-mode digital-like operation. However, with a low VDD, direct write-back has a higher failure probability, especially for WCB operations, as illustrated in Section IV-G. This could be improved with lower-voltage FeFET development.

For Type-II operations, besides the non-ideal factors mentioned above, inaccurate timing control and uncertainty of bitline capacitance are also the main non-ideal factors, which can lead to different BL voltage drops of a single discharge. The dynamic bias adjustment can almost eliminate the error from previous reads in each bias-setting stage (*Stages 1, 4* in Section IV-G). Therefore, BLiM results keep accurate. But the number of consecutive reads reduces since larger voltage margin is needed to tackle these non-ideal issues.

### D. Comparison With Prior BLiM Works

Fig. 18(c) shows the XOR2 performance comparison between the proposed BLiM and prior BLiM works based on RRAM, STT-MRAM, and SRAM. For NVM-based BLiM,

TABLE I
THE COMPARISON BETWEEN FEFET-BASED BLIM

| BLiM | | This Work | | ISLPED'18 [11] | R-FeFET [14] | FeMAT [15] |
|---|---|---|---|---|---|---|
| Structure | | 2T/cell | 3T/cell | 3T/cell | 4T/cell, with 2 R-FeFETs | 3T/cell |
| Configuration | | 10nm PTM, ρ=0.01Ω·m, $T_{FE}$=10.5nm | | 45nm PTM, $T_{FE}$=5.4nm | Assume that each half of a differential cell has the same $R_{on}$ and $R_{off}$ as 2T/C in this work | 45nm PTM |
| Two-input BLiM | (N)AND | Energy: 2.4fJ@0.55V, 4fJ@ 0.7V; Interface: Voltage-mode SA; Latency: <0.12ns (parallel) | Energy: 4fJ@0.65V, 6.2fJ@ 0.8V; Interface: Voltage-mode SA; Latency: <0.1ns (parallel) | Energy: ~56fJ (1.0V VDD, 33μA $I_{REF}$, 1ns duration assumed*); Interface: current-mode SA; Latency: – (not reported)* | Energy: 2x of our 2T/C AND; Interface: Reconfigurable voltage-mode SA with add-on logic gates; Latency: same as our 2T/C AND (Assume the same sensing interface as this work) | Energy: 10.6fJ (without peripherals), 130.6fJ (with peripherals); Interface: OpAmp-based current-to-voltage converter with add-on logic gates; Latency: 0.688ns (1V VDD, 0.1V for read) |
| | (N)OR | Energy: 3.8fJ@0.55V – 6.4fJ@0.7V; Interface: Voltage-mode SA with peripherals; Latency: <0.3ns (serial) | Energy: 5.5fJ@0.65V, 8.8fJ@ 0.8V; Interface: Voltage-mode SA with peripherals; Latency: <0.25ns (serial) | Energy: – (dynamic only)*; Interface: voltage-mode SA; Latency: 2.4ns read latency | | |
| | X(N)OR | Energy: 4fJ@0.55V, 6.7fJ@ 0.7V; Interface: Voltage-mode SA; Latency: <0.33ns (parallel) | Energy: 5.5fJ@0.65V, 8.6fJ@ 0.8V; Interface: Voltage-mode SA; Latency: <0.33ns (parallel) | Energy: approx. NAND + NOR; Interface: mixed-mode SA; Latency: voltage-mode read (2.4ns) + current-mode (not reported)* | | |
| | Others | COPY and NOT (support direct write-back), ADD, CUSTOM; Interface: same as above. | | ADD, NOT and to-be-added logic; Interface: mixed current-mode SA + custom logic (PTL) for ADD | ADD, NOT and to-be-added logic; Interface: same as above. | ADD, NOT and to-be-added logic; Interface: same as above. |
| Multi-input BLiM support | | (N)AND (Type-I); OR (Type-II with OR gate); NOR (Type-II with NOR gate); XOR (Type-II with XOR gate); Two-level SOP logic (Type-I NOT + Type-II with OR gate) | | (N)AND with adjusted references; (N)OR | (N)AND and (N)OR | (N)AND with adjusted references; (N)OR with adjusted references; |

*/–: Incomplete data. Minimal SA power is calculated based on provided 33μA $I_{REF}$ plus 23μA average $I_{SENSE}$ from 1.0V supply for assumed 1ns duration time.

we mainly focus on two operation schemes: (i) R-R logic, in which input and output are both resistance of the NVMs; (ii) R-V logic, which uses NVMs as inputs and generates voltage outputs by the sensing interface [66].

The proposed FeFET-based design outperforms others with lower energy and latency. With voltage-mode sensing, the energy is reduced by 1.3x–200x compared with other current-mode or mixed-mode R-V BLiM. The high write energy efficiency of FeFETs is another advantage, which further improves energy performance with write-back, compared to R-R BLiM designs that suffer from writes with DC currents. For latency, previous BLiM works have not fully exploited the parallelism in the operations or suffered from complex operating steps. By contrast, the proposed BCW and WCB charging schemes and simple sensing steps speed up logic operations by 3.0x–100x. The optimization of charge usage also enhances the low-power and low-latency features.

A detailed comparison with the prior FeFET-based general BLiM designs [11], [14], [15] is shown in Table I. The design in [11] achieves (N)AND and X(N)OR by current-mode and mixed-mode sensing, respectively. FeMAT in [15] converts the sense line current to voltage for sensing by a total of three operational amplifiers. These complicated sensing modes will lower the energy efficiency significantly. While [14] proposes voltage-mode sensing for logic operations, the differential cell design doubles the area and energy consumption. Moreover, the extra logic gates in [14] and [15], as well as the sensing buffers and inverters in [11] occupy large area and causes additional energy. For multi-input operations, the energy and latency performance will be worse due to complicated sensing schemes. On the other hand, this work achieves balanced area and energy with voltage-mode operations and low-cost peripheral design. In addition, the proposed design schemes of direct write-back and selective wordline-bitline charging also improve the energy efficiency. Furthermore, the peripherals enable optimized operations for more general logic functions.

### E. AES Implementation and Benchmarking

AES is a widely-used encryption algorithm. This algorithm consists of four submodules that can be operated recurrently and in parallel, as shown in Fig. 19(a). In the benchmarking, an efficient implementation of AES in NVMs, named "AES in-memory" (AIM), is considered as the criterion [67].

The proposed BLiM can accelerate several custom instructions in the AES process like in-memory XOR2 and XOR4. The dedicated architecture and data organization for AES are shown in Fig. 19(b). We consider the encryption of a 128-bit data block, with each bit of a byte stored in different arrays. Independent custom logic (Section III-D) is designed for each column of a data block. Additional S-Boxes and M2 LUTs are included for maximized parallelism. The data path of the four transformations in AES is shown in Fig. 19(c).

The latency and energy consumption of each operation for FeFETs have been discussed in Section VI. For a quantitative application benchmark, all instructions are split into several atomic operations shown in Table II. XOR2 and XOR4 refer to 2-operand and 4-operand in-memory XOR logic, respectively, with results written back to the array after the computation. An operand includes data of 1 byte. All in-memory operations in AES can have the maximum parallelism that equals the number of SAs. We count the used operations in AES and sum up their latency and energy, with the latency and energy of LUTs and controller scheduling excluded for all compared works. Fig. 19(d) shows the comparison between different BLiM designs, where the latency and energy savings reach at least 3.0x and 3.2x, respectively, for the proposed BLiM design. Such savings of energy and latency in the

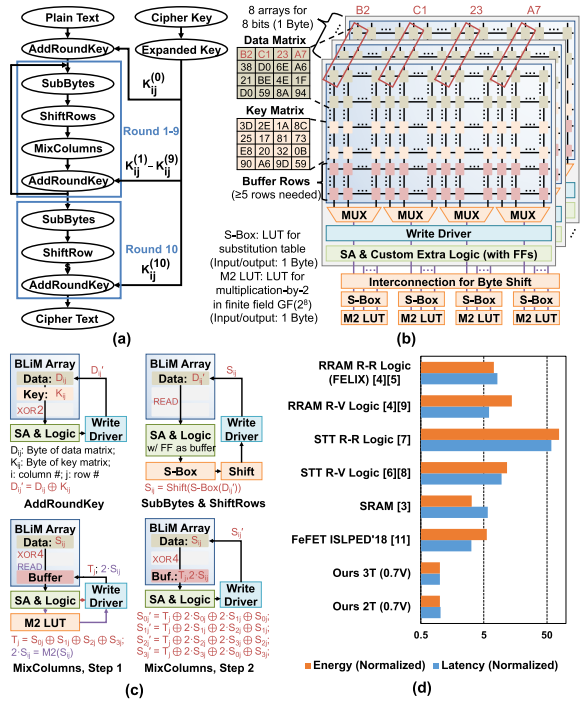



Fig. 19. AES evaluation. (a) AES overview; (b) AES architecture for proposed BLiM; (c) AES data path of each transformation; (d) BLiM evaluation for 128-bit AES using AIM implementation, normalized to proposed 3T BLiM design.

TABLE II

TOTAL ATOMIC OPERATIONS OF EACH 128-BIT INPUT

| Atomic Operation | Details | | Count |
|---|---|---|---|
| Read | 8-bit memory read | | 304 |
| LUT + Write | Search 8-bit inputs in other memory blocks, then write the search results into the memory | | 304 |
| XOR2 | Execute in-memory XOR2, and write back to the memory | | 176 |
| XOR4 | This work | Execute in-memory XOR4, and write back to the memory | 180 |
| | Others | Execute three {in-memory XOR2 + write-back} in serial | |

benchmarking are achieved because (i) multiple-operands operation is possible with only one-time bitline charging, (ii) only dynamic power is consumed, which avoids the high power caused by the static current of current-domain operations. This result shows great potential of FeFETs for BLiM applications.

### F. Future Work

The proposed BLiM primitives have covered most bitwise logics. Arithmetic operations, such as addition and multiplication, can also be implemented by the combination of BLiM primitives. However, more optimization for the computing process and architecture support needs to be explored. To further optimize the performance of applications, memory controller and software also play an important role. With the proposed adaptive WCB/WDB operations and different custom logic for Type-II operations, more compile-time instruction reordering strategies for specified algorithms can be designed.

## VII. CONCLUSION

In this paper, we present a FeFET-based low-power BLiM approach. The proposed BLiM architecture could be applied in both 2T/C and 3T/C memory arrays. A simplified switched-capacitor-based sensing interface along with optional peripherals are designed for efficient and general multiple-input logic operations. The voltage-mode operation and sensing schemes are designed to avoid DC power consumption. For more energy and latency savings, we further propose several techniques: (i) SA-free direct write-back for several logic operations, which reuses the remnant bitline charge; (ii) continuous read for Type-II operations that reduces the total bitline charging activities. With these proposed techniques, circuit and application evaluations have shown advantages in both energy and latency, compared with prior BLiM works.

## REFERENCES

[1] J. Backus, "Can programming be liberated from the Von Neumann style? A functional style and its algebra of programs," *Commun. ACM*, vol. 21, no. 8, pp. 613–641, Aug. 1978.

[2] M. Kang, M.-S. Keel, N. R. Shanbhag, S. Eilert, and K. Curewitz, "An energy-efficient VLSI architecture for pattern recognition via deep embedding of computation in SRAM," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2014, pp. 8326–8330.

[3] 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.

[4] S. Gupta et al., "FELIX: Fast and energy-efficient logic in memory," in *Proc. Int. Conf. Comput.-Aided Design*, Nov. 2018, pp. 1–7.

[5] S. Gupta, M. Imani, H. Zhao, F. Wu, J. Zhao, and T. Š. Rosing, "Implementing binary neural networks in memory with approximate accumulation," in *Proc. ACM/IEEE Int. Symp. Low Power Electron. Design*, Aug. 2020, pp. 247–252.

[6] 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.

[7] Y. Pan et al., "An STT-MRAM based reconfigurable computing-in-memory architecture for general purpose computing," *CCF Trans. High Perform. Comput.*, vol. 2, no. 3, pp. 272–281, Sep. 2020.

[8] Y. Zhao et al., "An STT-MRAM based in memory architecture for low power integral computing," *IEEE Trans. Comput.*, vol. 68, no. 4, pp. 617–623, Apr. 2019.

[9] S. Li et al., "Pinatubo: A processing-in-memory architecture for bulk bitwise operations in emerging non-volatile memories," in *Proc. 53rd Annu. Design Autom. Conf.*, Jun. 2016, pp. 1–6.

[10] S. Angizi and D. Fan, "ReDRAM: A reconfigurable Processing-in-DRAM platform for accelerating bulk bit-wise operations," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2019, pp. 1–8.

[11] D. Reis et al., "Computing in memory with FeFETs," in *Proc. Int. Symp. Low Power Electron. Design*, Jul. 2018, pp. 1–6.

[12] X. Yin et al., "An ultra-dense 2FeFET TCAM design based on a multi-domain FeFET model," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 66, no. 9, pp. 1577–1581, Sep. 2019.

[13] X. Yin, X. Chen, M. Niemier, and X. S. Hu, "Ferroelectric FETs-based nonvolatile logic-in-memory circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 1, pp. 159–172, Jan. 2019.

[14] 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)*, Jul. 2019, pp. 1–6.

[15] X. Zhang, X. Chen, and Y. Han, "FeMAT: Exploring in-memory processing in multifunctional FeFET-based memory array," in *Proc. IEEE 37th Int. Conf. Comput. Design (ICCD)*, Nov. 2019, pp. 541–549.

[16] X. Zhang et al., "Re-FeMAT: A reconfigurable multifunctional FeFET-based memory architecture," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 41, no. 11, pp. 5071–5084, Nov. 2022.

[17] S. Dunkel et al., "A FeFET based super-low-power ultra-fast embedded NVM technology for 22nm FDSOI and beyond," in *IEDM Tech. Dig.*, Dec. 2017, p. 19.

[18] K. Chatterjee et al., "Self-aligned, gate last, FDSOI, ferroelectric gate memory device with 5.5-nm $Hf_{0.8}Zr_{0.2}O_2$, high endurance and breakdown recovery," *IEEE Electron Device Lett.*, vol. 38, no. 10, pp. 1379–1382, Oct. 2017.

[19] S. Slesazeck, U. Schroeder, and T. Mikolajick, "Embedding hafnium oxide based FeFETs in the memory landscape," in *Proc. Int. Conf. IC Design Technol. (ICICDT)*, Jun. 2018, pp. 121–124.

[20] M. Trentzsch et al., "A 28nm HKMG super low power embedded NVM technology based on ferroelectric FETs," in *IEDM Tech. Dig.*, Dec. 2016, p. 11.

[21] A. Sharma and K. Roy, "1T non-volatile memory design using sub-10nm ferroelectric FETs," *IEEE Electron Device Lett.*, vol. 39, no. 3, pp. 359–362, May 2018.

[22] J. Müller et al., "Nanosecond polarization switching and long retention in a novel MFIS-FET based on ferroelectric $HfO_2$," *IEEE Electron Device Lett.*, vol. 33, no. 2, pp. 185–187, Feb. 2012.

[23] J. Müller et al., "Ferroelectricity in $HfO_2$ enables nonvolatile data storage in 28 nm HKMG," in *Proc. Symp. VLSI Technol. (VLSIT)*, Jun. 2012, pp. 25–26.

[24] M. Jerry et al., "Ferroelectric FET analog synapse for acceleration of deep neural network training," in *IEDM Tech. Dig.*, Dec. 2017, pp. 6.2.1–6.2.4.

[25] K. Ni et al., "Fundamental understanding and control of device-to-device variation in deeply scaled ferroelectric FETs," in *Proc. Symp. VLSI Technol.*, Jun. 2019, pp T40–T41.

[26] H. Mulaosmanovic et al., "Impact of read operation on the performance of $HfO_2$-based ferroelectric FETs," *IEEE Electron Device Lett.*, vol. 41, no. 9, pp. 1420–1423, Sep. 2020.

[27] K. Ni et al., "Write disturb in ferroelectric FETs and its implication for 1T-FeFET AND memory arrays," *IEEE Electron Device Lett.*, vol. 39, no. 11, pp. 1656–1659, Nov. 2018.

[28] S. George et al., "Nonvolatile memory design based on ferroelectric FETs," in *Proc. 53rd Annu. Design Autom. Conf.*, Jun. 2016, pp. 1–6.

[29] X. Li et al., "Design of 2T/cell and 3T/cell nonvolatile memories with emerging ferroelectric FETs," *IEEE Des. Test*, vol. 36, no. 3, pp. 39–45, Jun. 2019.

[30] S. Lim et al., "Cross-coupled ferroelectric FET-based ternary content addressable memory with energy-efficient match line scheme," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 2, pp. 806–818, Feb. 2023.

[31] X. Li et al., "Enabling energy-efficient nonvolatile computing with negative capacitance FET," *IEEE Trans. Electron Devices*, vol. 64, no. 8, pp. 3452–3458, Aug. 2017.

[32] X. Li et al., "Advancing nonvolatile computing with nonvolatile NCFET latches and flip-flops," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 64, no. 11, pp. 2907–2919, Nov. 2017.

[33] X. Li et al., "Lowering area overheads for FeFET-based energy-efficient nonvolatile flip-flops," *IEEE Trans. Electron Devices*, vol. 65, no. 6, pp. 2670–2674, Jun. 2018.

[34] A. A. Saki et al., "A family of compact non-volatile flip-flops with ferroelectric FET," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 11, pp. 4219–4229, Nov. 2019.

[35] X. Li et al., "Design of nonvolatile SRAM with ferroelectric FETs for energy-efficient backup and restore," *IEEE Trans. Electron Devices*, vol. 64, no. 7, pp. 3037–3040, Jul. 2017.

[36] X. Sun, P. Wang, K. Ni, S. Datta, and S. Yu, "Exploiting hybrid precision for training and inference: A 2T-1FeFET based analog synaptic weight cell," in *IEDM Tech. Dig.*, Dec. 2018, pp. 3.1.1–3.1.4.

[37] X. Chen et al., "Design and optimization of FeFET-based crossbars for binary convolution neural networks," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2018, pp. 1205–1210.

[38] S. K. Thirumala, S. Jain, S. K. Gupta, and A. Raghunathan, "Ternary compute-enabled memory using ferroelectric transistors for accelerating deep neural networks," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2020, pp. 31–36.

[39] A. F. Laguna, X. Yin, D. Reis, M. Niemier, and X. S. Hu, "Ferroelectric FET based in-memory computing for few-shot learning," in *Proc. Great Lakes Symp. VLSI*, May 2019, pp. 373–378.

[40] T. Soliman et al., "Efficient FeFET crossbar accelerator for binary neural networks," in *Proc. IEEE 31st Int. Conf. Appl.-Specific Syst., Archit. Processors (ASAP)*, Jul. 2020, pp. 109–112.

[41] P. Wang et al., "Drain-erase scheme in ferroelectric field effect transistor—Part II: 3-D-NAND architecture for in-memory computing," *IEEE Trans. Electron Devices*, vol. 67, no. 3, pp. 962–967, Mar. 2020.

[42] V. P.-H. Hu et al., "Split-gate FeFET (SG-FeFET) with dynamic memory window modulation for non-volatile memory and neuromorphic applications," in *Proc. Symp. VLSI Technol.*, Jun. 2019, pp. T134–T135.

[43] C. Chen et al., "Bio-inspired neurons based on novel leaky-FeFET with ultra-low hardware cost and advanced functionality for all-ferroelectric neural network," in *Proc. Symp. VLSI Technol.*, Jun. 2019, pp. T136–T137.

[44] M. Li et al., "Nonvolatile and energy-efficient FeFET-based multiplier for energy-harvesting devices," in *Proc. 25th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2020, pp. 562–567.

[45] E. T. Breyer et al., "Compact FeFET circuit building blocks for fast and efficient nonvolatile logic-in-memory," *IEEE J. Electron Devices Soc.*, vol. 8, pp. 748–756, Apr. 2020.

[46] D. Reis, M. T. Niemier, and X. S. Hu, "A computing-in-memory engine for searching on homomorphically encrypted data," *IEEE J. Explor. Solid-State Comput. Devices Circuits*, vol. 5, pp. 123–131, 2019.

[47] S. Thirumala, A. Raha, S. Gupta, and V. Raghunathan, "Exploring the design of energy-efficient intermittently powered systems using reconfigurable ferroelectric transistors," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 30, no. 4, pp. 365–378, Apr. 2022.

[48] C. Marchand et al., "A FeFET-based hybrid memory accessible by content and by address," *IEEE J. Explor. Solid-State Comput. Devices Circuits*, vol. 8, pp. 19–26, 2022.

[49] Y. Luo et al., "A compute-in-memory hardware accelerator design with back-end-of-line (BEOL) transistor based reconfigurable interconnect," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 12, no. 2, pp. 445–457, Jun. 2022.

[50] W. Shim and S. Yu, "Ferroelectric field-effect transistor-based 3-D NAND architecture for energy-efficient on-chip training accelerator," *IEEE J. Explor. Solid-State Comput. Devices Circuits*, vol. 7, pp. 1–9, 2021.

[51] X. Ma et al., "CapCAM: A multilevel capacitive content addressable memory for high-accuracy and high-scalability search and compute applications," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 30, no. 11, pp. 1770–1782, Nov. 2022.

[52] G. Yin et al., "Enabling lower-power charge-domain nonvolatile in-memory computing with ferroelectric FETs," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 7, pp. 2262–2266, Jul. 2021.

[53] M. Yayla et al., "Reliable binarized neural networks on unreliable beyond von-neumann architecture," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 69, no. 6, pp. 2516–2528, Jun. 2022.

[54] L. Mankali et al., "Leveraging ferroelectric stochasticity and in-memory computing for DNN IP obfuscation," *IEEE J. Explor. Solid-State Comput. Devices Circuits*, vol. 8, pp. 102–110, 2022.

[55] X. Li and L. Lai, "Nonvolatile memory and computing using emerging ferroelectric transistors," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2018, pp. 750–755.

[56] M. Lee et al., "FeFET-based low-power bitwise logic-in-memory with direct write-back and data-adaptive dynamic sensing interface," in *Proc. ACM/IEEE Int. Symp. Low Power Electron. Design*, Aug. 2020, pp. 127–132.

[57] S. Deng et al., "A comprehensive model for ferroelectric FET capturing the key behaviors: Scalability, variation, stochasticity, and accumulation," in *Proc. IEEE Symp. VLSI Technol.*, Jun. 2020, pp. 1–2.

[58] *Predictive Technology Model*. Accessed: Sep. 20, 2022. [Online]. Available: http://ptm.asu.edu/

[59] A. Aziz et al., "Physics-based circuit-compatible SPICE model for ferroelectric transistors," *IEEE Electron Device Lett.*, vol. 37, no. 6, pp. 805–808, Jun. 2016.

[60] K. Ni, M. Jerry, J. A. Smith, and S. Datta, "A circuit compatible accurate compact model for ferroelectric-FETs," in *Proc. IEEE Symp. VLSI Technol.*, Jun. 2018, pp. 131–132.

[61] D. Reis, M. Niemier, and X. S. Hu, "The implications of ferroelectric FET device models to the design of computing-in-memory architectures," *J. Integr. Circuits Syst.*, vol. 16, no. 1, pp. 1–8, Apr. 2021.

[62] A. I. Khan, A. Keshavarzi, and S. Datta, "The future of ferroelectric field-effect transistor technology," *Nature Electron.*, vol. 3, no. 10, pp. 588–597, Oct. 2020.

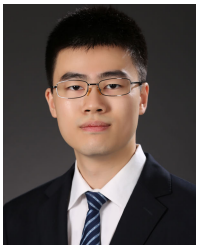
[63] S. Jindal et al., "Scaling behavior of ferroelectric FET with reduction in number of domains in ferroelectric layer," *Jpn. J. Appl. Phys.*, vol. 61, May 2022, Art. no. SC1030.

[64] D. Das et al., "A Ge-channel ferroelectric field effect transistor with logic-compatible write voltage," *IEEE Electron Device Lett.*, vol. 44, no. 2, pp. 257–260, Feb. 2023.

[65] Z. Jiang et al., "Asymmetric double-gate ferroelectric FET to decouple the tradeoff between thickness scaling and memory window," in *Proc. IEEE Symp. VLSI Technol. Circuits (VLSI Technol. Circuits)*, Jun. 2022, pp. 395–396.

[66] W. Shen et al., "Stateful logic operations in one-transistor-one-resistor resistive random access memory array," *IEEE Electron Device Lett.*, vol. 40, no. 9, pp. 1538–1541, Sep. 2019.

[67] M. Xie, S. Li, A. O. Glova, J. Hu, and Y. Xie, "Securing emerging nonvolatile main memory with fast and energy-efficient AES in-memory implementation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 11, pp. 2443–2455, Nov. 2018.

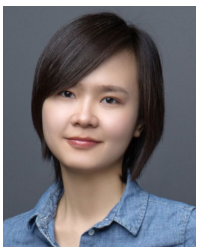


**Wenjun Tang** (Graduate Student Member, IEEE) received the B.S. degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2021, where he is currently pursuing the Ph.D. degree. His current research interests include in-memory and in-sensor computing circuit designs for edge AI.

**Mingyen Lee** (Member, IEEE) received the B.S. degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2021, where he is currently pursuing the master's degree. His research interests include energy-area-efficient computing-in-memory designs.

**Juejian Wu** (Student Member, IEEE) received the B.S. degree from Tsinghua University, Beijing, China, where he is currently pursuing the Ph.D. degree. His research interests include emerging memory circuit design, memory architecture, and computing-in-memory.

**Yixin Xu** is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Penn State University, University Park, PA, USA. Her research interests include circuit and computer architecture designs, especially with emerging memory technologies.

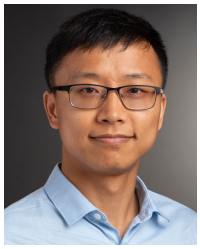


**Yao Yu** received the B.S. degree from the Beijing Institute of Technology and the M.S. degree from the Karlsruhe Institute of Technology. She has been with the Department of Research and Development Automated Driving System, Daimler Greater China Ltd., Beijing, China, since 2018. She is currently developing an automated driving system data recorder.

**Yongpan Liu** (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees from Tsinghua University, China, in 1999, 2002, and 2007, respectively.

He is currently a Full Professor (Cheung Kong Scholar) with the Department of Electronic Engineering, Tsinghua University.

Prof. Liu is a Program Committee Member of ISSCC, A-SSCC, and DAC.



**Kai Ni** (Member, IEEE) received the B.S. degree in electrical engineering from the University of Science and Technology of China, in 2011, and the Ph.D. degree in electrical engineering from Vanderbilt University, Nashville, TN, USA, in 2016. Since then, he has been a Post-Doctoral Associate with the University of Notre Dame. He is currently an Assistant Professor of electrical and microelectronic engineering with the Rochester Institute of Technology. His current research interests include nanoelectronic devices empowering unconventional computing, AI accelerator, and 3D memory technology.

**Yu Wang** (Fellow, IEEE) received the B.S. and Ph.D. (Hons.) degrees from Tsinghua University, Beijing, in 2002 and 2007, respectively.

He is currently a tenured Professor with the Department of Electronic Engineering, Tsinghua University. His research interests include brain-inspired computing, application-specific hardware computing, parallel circuit analysis, and power/reliability-aware system design methodology.

**Huazhong Yang** (Fellow, IEEE) received the B.S. degree in microelectronics and the M.S. and Ph.D. degrees in electronic engineering from Tsinghua University, Beijing, China, in 1989, 1993, and 1998, respectively.

In 1993, he joined the Department of Electronic Engineering, Tsinghua University, where he has been a Full Professor since 1998. His research interests include wireless sensor networks, data converters, energy-harvesting circuits, nonvolatile processors, and brain-inspired computing.

**Vijaykrishnan Narayanan** (Fellow, IEEE) received the B.S. degree in computer science and engineering from the University of Madras, Chennai, India, in 1993, and the Ph.D. degree in computer science and engineering from the University of South Florida, Tampa, FL, USA, in 1998.

He is currently the Robert Noll Chair Professor of computer science and engineering and electrical engineering with Penn State University, University Park, PA, USA. His current research interests include power-aware and reliable systems, embedded systems, nanoscale devices, interactions with system architectures, reconfigurable systems, computer architectures, network-on-chips, and domain-specific computing.

**Xueqing Li** (Senior Member, IEEE) received the B.S. and Ph.D. degrees from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2007 and 2013, respectively.

From 2013 to 2017, he was a Post-Doctoral Researcher with the Department of Computer Science and Engineering, Penn State University, University Park, PA, USA. He joined the Department of Electronic Engineering, Tsinghua University, as an Assistant Professor, in 2018, where he is currently an Associate Professor. His research interests include high-performance data converter circuit design, emerging memory, and memory-oriented computing with CMOS and beyond-CMOS technologies.