Received 18 January 2024; revised 3 May 2024 and 15 June 2024; accepted 12 July 2024. Date of publication 25 July 2024;
date of current version 7 August 2024.

Digital Object Identifier 10.1109/OJSSCS.2024.3432468

A Readout Scheme for PCM-Based Analog In-Memory Computing With Drift Compensation Through Reference Conductance Tracking

ALESSIO ANTOLINI[1], ANDREA LICO[1] (Graduate Student Member, IEEE), FRANCESCO ZAVALLONI[1], ELEONORA FRANCHI SCARSELLI[1] (Member, IEEE), ANTONIO GNUDI[1], MATTIA LUIGI TORRES[2], ROBERTO CANEGALLO[2], AND MARCO PASOTTI[2]

[1]ARCES-DEI, University of Bologna, 40126 Bologna, Italy

[2]Smart Power Technology R&D, STMicroelectronics, 20041 Agrate Brianza, Italy

CORRESPONDING AUTHOR: A. ANTOLINI (e-mail: alessio.antolini2@unibo.it)

This work was supported in part by the ECSEL Joint Undertaking (JU) under Grant 101007321. The JU receives support from the European Union's Horizon 2020 Research and Innovation Programme and France, Belgium, Czech Republic, Germany, Italy, Sweden, Switzerland, Türkiye.

ABSTRACT This article presents a readout scheme for analog in-memory computing (AIMC) based on an embedded phase-change memory (ePCM). Conductance time drift is overcome with a hardware compensation technique based on a reference cell conductance tracking (RCCT). Accuracy drop due to circuits mismatch and variability involved in the computational chain are minimized with an optimized iterative program-and-verify algorithm applied to the phase-change memory (PCM) devices. The proposed AIMC scheme is designed and manufactured in a 90-nm STMicroelectronics CMOS technology, with the aim of adding a signed multiply-and-accumulate (MAC) computation feature to a Ge-Rich GeSbTe (GST) embedded PCM array. Experimental characterizations are performed under different operating conditions and show that the mean MAC decrease in time is approximately null at room temperature and reduced by a factor of 3 after 64-h bake at 85 °C. Based on several MAC operations, the estimated 512 × 512 matrix–vector-multiplication (MVM) accuracy is 97.4%, whose decrease in time is less than 3% in the worst case.

INDEX TERMS Analog in-memory computing (AIMC), artificial intelligence, drift compensation, matrix–vector multiplication (MVM), phase-change memory (PCM).

I. INTRODUCTION

THE RISING spread of data-centric applications, such as deep learning and artificial intelligence, has dictated new constraints in the field of computing architectures and algorithms. Among innovative solutions recently being researched, in-memory computing (IMC) has been proposed as a valid strategy to overcome traditional Von Neumann architectures, where physically separated processing and memory units implicate a massive amount of both energy consumption and computing latency due to internal data transfers [1], [2], [3], [4]. To address this issue, the IMC strategy intends the execution of computing tasks directly within the memory array avoiding data to be conveyed between memory and processing units [5]. This can be achieved on the one hand with fully digital architectures using typically static random access memories (SRAMs) [6], [7], [8], [9]. On the other side, analog IMC (AIMC) [10] allows all the computations to be carried out in an analog way, and, according to this approach, the computational function emerges from the physical features of the memory devices, thus requiring the employment of dedicated memories and architectures.

A representative AIMC task is the matrix–vector multiplication (MVM), which is the kernel operation performed in the inference of deep neural networks (DNNs) [11] and by linear algebra accelerators [12]. An MVM can be computed ideally in one step taking advantage of nonvolatile memories (NVMs) arranged in a cross-point structure [13], [14], such as Flash [15] and magneto-resistive random access memory (MRAM) [16]. Among resistive NVMs, phase-change memory (PCM) has established itself as a promising technology for the AIMC context as it grants data to be stored with high density [17], [18]. This can be achieved thanks to the increasing scalability of its memory cells; furthermore, multiple bits can be stored in a single device through a set of intermediary resistive states [19].

In the PCM-based AIMC scenario, MVM can be implemented in an analog fashion using Ohm and Kirchhoff laws, with the matrix coefficients being mapped in the PCM cells in the form of conductance levels [20], [21]. Although many advances have been recently carried out to extend the application panorama of PCM from conventional digital storage to AIMC, some challenges are still open [22]. Among these, the retention of the states stored in PCM devices is deeply affected by the conductance drift, so that the generic cell conductance tends to randomly decrease in time [23], with a negative impact on the overall computation. As drift plays a primary role in narrowing the accuracy of PCM-based hardware accelerators [24], different approaches have been proposed to address this issue.

This article expands the analysis of the hardware drift compensation scheme based on reference cell conductance tracking (RCCT) first presented in [25], whose effect was combined with a software device-aware training to enhance the accuracy of DNNs in [26]. In particular, this article motivates the employment of the conductance-ratio RCCT with respect to other possible compensation solutions, showing its potential in mean drift recovery for low-power PCM-based AIMC. Moreover, the recognized increase of matrix coefficients spread related to conductance-ratio RCCT is minimized involving multiple PCM reference cells. Furthermore, a thorough characterization of the corresponding prototype is performed to estimate the accuracy of 512 × 512-sized MVM operations, even with a temperature-induced drift acceleration.

This article is organized as follows. Section II describes the drift phenomenon through an experimental characterization of Ge-rich GeSbTe (GST) PCM cells of a 90-nm CMOS STMicroelectronics embedded memory IP, and discusses the state-of-the-art solutions for drift compensation. Section III motivates and describes the proposed hardware scheme for drift compensation. Section IV is focused on its circuit-level implementation, while Section V illustrates the related test prototype, designed and fabricated in the same previous technology, along with its characterization results. Section VI concludes this article.

II. PCM CELLS CONDUCTANCE DRIFT

A. ANALYSIS OF PCM CELLS CONDUCTANCE TIME BEHAVIOR

Short-term conductance time drift manifests itself as a slow but steady increase of the resistivity of the PCM amorphous material, which several studies have ascribed to defects annihilation [27], [28] and phase relaxation [29]. The measured PCM cell conductance $g(t)$ has been shown to follow a power law in time

$$g(t) = g_0 \left(\frac{t}{t_0}\right)^{-\nu} \tag{1}$$

where $g_0 := g(t_0)$ is the initial conductance at arbitrary time t_0 after programming; $\nu > 0$ is the drift coefficient, which is a cell-to-cell variable and depends on the operating temperature [30].

As a preliminary step toward the goals of this work, an analysis of the time behavior of Ge-rich GeSbTe (GST) PCM devices was carried out on a 128-kB embedded PCM (ePCM) memory IP in a 90-nm STMicroelectronics CMOS technology [31]. Conductance measurements were performed by applying a voltage V_R on each cell [as outlined by Fig. 1(a)] and reading its corresponding current.

As first discussion, Fig. 1(b) shows the current of two cells, programmed with the same initial conductance value g_0, being monitored for an hour at room temperature (RT). Curves highlight first the evanescent trend of the drift (as modeled by the previous equation), whose randomness introduces a conductance variability from cell to cell; furthermore, each device is subject to low-frequency (flicker) noise, which originates from traps in the amorphous phase of the material [32] and results in additional fluctuations of its current. Both effects turn into a decrease of the MVM computation accuracy in PCM-based AIMC systems, as the matrix coefficients are typically implemented by the cells conductance.

To analyze the time drift dependence on the initial conductance g_0 and on temperature, 1000 PCM cells programmed to different g_0 values were measured after 12 h ($t = t_a$) at T_1 = RT. The drift coefficient $\nu_1 := \nu|_{T=T_1}$ was evaluated by inverting (1), that is

$$\nu_1 = \frac{\ln(g_0) - \ln(g(t_a))}{\ln\left(\frac{t_a}{t_0}\right)}. \tag{2}$$

The drift coefficient was then evaluated at higher temperature $T_2 = 85$ °C. As the employed measurement setup is suitable for measures performed at RT only, the drift coefficient $\nu_2 := \nu|_{T=T_2}$ was estimated, as suggested in [23], with two measures (t_a, t_b) before and one measure (t_c) after a 12-h annealing of the testchip at T_2, so that

$$\nu_2 = \frac{\ln\left(\frac{g(t_a)}{g(t_c)}\right) + \nu_1 \ln\left(\frac{t_a}{t_0}\right)}{\ln\left(\frac{t_c - t_b}{t_0}\right)}. \tag{3}$$

Both coefficient sets are reported in Fig. 1(c), where ν_1 and ν_2 are plotted for each cell as a function of its initial conductance value $g(t_0)$ to highlight two relevant aspects in the context of AIMC. First of all, the drift coefficient depends on the initial conductance value $g(t_0)$, as it is attributable to the amount of amorphous phase resulting from the cell programming state [33]. Second, the drift is accelerated by temperature [34], as highlighted by higher ν-values at 85 °C with respect to those at RT. Therefore, in a

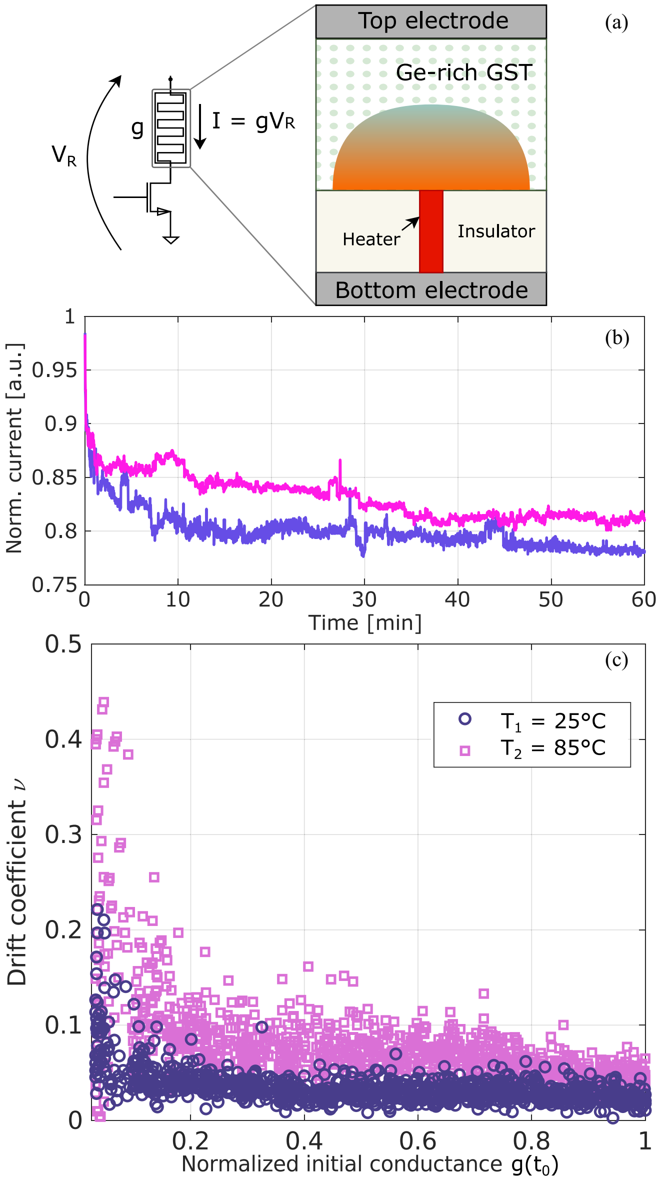

FIGURE 1. **(a) Employed measurement setup employed and sketch of the internal structure of a GST PCM cell. (b) Normalized current of two sample PCM devices programmed to the same initial conductance g_0 and measured for an hour at RT. (c) Drift coefficients ν_1 at T_1 and ν_2 at T_2 as a function of the initial conductance value $g(t_0)$ for a set of 1000 PCM cells. Measures are taken in a 12-h time window at RT (blue circles) and at 85 °C (purple squares).**

multilevel AIMC approach, the overall retention is difficult to evaluate; furthermore, temperature fluctuations during computations make complex to accurately describe the behavior of cells over time and their effect on the computing accuracy.

These considerations are supported by various works highlighting the impact of drift on the classification accuracy of some well-known neural networks [24], [35]. For this reason, the PCM cell drift has been handled from the technology itself to the application level with different strategies, whose overview is provided hereafter.

B. STATE-OF-THE-ART DRIFT MITIGATION TECHNIQUES

Due to the intrinsic physical origin of the drift, optimizations concerning the device materials have been investigated. In [36] an optimized Ge/N-doped GST alloy achieves a fast programming time and high crystallization temperature; Koelmans et al. [37] proposed to project the phase configuration of phase-change material onto a stable resistive thin film within the device, so that the readout current is considerably less sensitive to the lattice rearrangement process of the amorphous phase. In [38], a read/program scheme realizes PCM drift compensation and features specific current pulses to suppress data decoding failure for multilevel storage applications.

In the AIMC context, drift mitigation is typically addressed at the application level. In [35], stored coefficients are modified during the inference execution; alternatively, other works overcome drift issue with post-processing compensations based on its statistical description [39], [40], or adopting device-aware DNNs training techniques [41], [42], [43] to increase the resilience of the neural inference accuracy.

A representative hardware compensation mechanism to address generic NVMs retention issues is the RCCT. Lin et al. [44] proposed to improve retention in ReRAM-based systems with a reference array stored in a system register, which is read to correct neural inference operations. RCCT has been initially proposed for PCM-based multilevel storage in [38] and exploits the resistance of reference cells as an adaptive-sensing threshold.

However, programming cells at different times due to standard write operations, may result in overlapping intermediate conductance states. As a result, all cells within a given reference must have the same time synchronization, and any update necessarily involves a read and rewrite process of the entire group. Incidentally, the AIMC context does not require frequent write operations, as the MVM weights stored in the memory array are typically defined offline and infrequently updated [41]. This could give to RCCT the opportunity to be an effective strategy to overcome the drift impact on analog computations. Furthermore, thanks to the on-chip nature of the RCCT, its drift mitigation turns to be more resilient to further nonidealities, such as chip-to-chip variability and operating conditions variations, that can complicate accurate modeling of PCM drift for post-processing compensations.

III. HARDWARE DRIFT COMPENSATION FOR MULTILEVEL AIMC

A. MEAN CONDUCTANCE DRIFT RECOVERY

According to the AIMC paradigm, the MVM output $\mathbf{z} = \mathbf{W}\mathbf{x}$, defined as

$$\begin{pmatrix} z_1 \\ \vdots \\ z_m \end{pmatrix} = \begin{pmatrix} w_{1,1} & \cdots & w_{1,n} \\ \vdots & \ddots & \vdots \\ w_{m,1} & \cdots & w_{m,n} \end{pmatrix} \times \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \tag{4}$$

can be implemented with the matrix coefficients $w_{i,j}$ being stored in the memory in the form of conductance $g_{i,j}$,

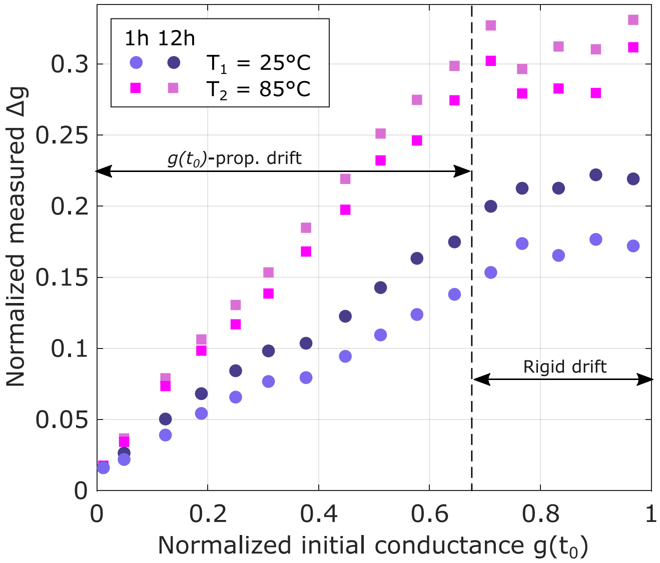

FIGURE 2. Mean conductance decrease Δg for 16 groups of cells as a function of their initial conductance. Curves refer to Δg evaluated after $t = 1$ and $t = 12$ h at RT (blue circles) and at 85 °C (purple squares).

exploiting one or more cells per element. Each z_i of the output array $\mathbf{z}$ is computed as the multiply-and-accumulate (MAC) operation

$$z_i = \mathbf{w}_i \cdot \mathbf{x} = \sum_{j=1}^{n} w_{i,j} x_j \tag{5}$$

where $\mathbf{w}_i = (w_{i,1}, \ldots, w_{i,n})$, and $\mathbf{x} = (x_1, \ldots, x_n)$ is the input vector.

Fig. 2 reports the mean conductance decrease $\Delta g := g(t_0) - g(t)$ of 16 groups of cells with different initial conductance $g(t_0)$. Measures were performed on the same test vehicle of Section II-A. Curves refer to $\Delta g(t, T)$ measured after $t_1 = 1$ h and $t_2 = 12$ h at RT (T_1) and at $T_2 = 85$ °C. Results identify two possible behaviors of the mean conductance decrease. Cells with normalized initial conductance ranging from 0 to 0.7 approximately exhibit a $g(t_0)$-proportional drift, so that one can assume $\Delta g(t, T, g_0) = k g(t_0)$, where $k = k(t, T)$ depends on time and temperature. The conductance decrease of cells with higher initial conductance shows a negligible correlation between $g(t_0)$ and Δg, which can thus be assumed rigid in time and temperature, i.e., $\Delta g(t, T, g_0) = h(t, T)$.

In the last condition, drift can be mitigated by implementing each matrix coefficient through the difference of two PCM devices, so that the mean value of the (i, j)th matrix coefficient $w_{i,j} := g_{i,j} - g_R$ has ideally no dependence on drift, as in this case

$$\begin{aligned} w_{i,j}(t, T) &= \big(g_{i,j}(t_0) - h(t, T)\big) - (g_R(t_0) - h(t, T)) \\ &= g_{i,j}(t_0) - g_R(t_0) \end{aligned} \tag{6}$$

where g_R is the reference cell which implements the RCCT.

On the other side, using an RCCT based on conductance ratio is the most effective method to attenuate the drift when it exhibits a $g(t_0)$-proportional behavior. In this case, each coefficient

$$w_{i,j} := \frac{g_{i,j}}{g_R} \tag{7}$$

turns to be ideally constant on average in time and temperature, as

$$w_{i,j}(t, T) = \frac{g_{i,j}(t_0)(1 - k(t, T))}{g_R(t_0)(1 - k(t, T))} = \frac{g_{i,j}(t_0)}{g_R(t_0)} \tag{8}$$

recovers the mean value of the cell conductance decrease for each matrix weight.

The feasibility of the first solution is validated by circuital simulations in [45] using a binary programming of PCM devices. In the multilevel context, as shown in Fig. 2, the range of programmed conductances $g(t_0)$ where cells exhibit a $g(t_0)$-proportional drift is wider than the one where the drift is rigid. Therefore, the implementation of each matrix coefficient using a conductance ratio as in (8) would mitigate the drift of a wider range of programmed conductances $g(t_0)$. Moreover, operating at lower conductances allows for the reduction of the overall power consumption. The hardware implementation proposed in this work is designed to operate under this latter condition.

The assertion of (8) is valid in the ideal case of $\Delta g(t, T, g_0) = k(t, T) g(t_0)$. A more realistic expression of $w_{i,j}$ in time can be obtained combining (1) and (7)

$$w_{i,j}(t, T) = \frac{g_{i,j}}{g_R} \left(\frac{t}{t_0} \right)^{\nu_i - \nu_R} \tag{9}$$

which exhibits an equivalent weight drift coefficient reduced to $\nu_i - \nu_R$. The value of the reference cell conductance is crucial for the effectiveness of the drift compensation, since an ideal compensation would be achieved only if $\nu_i = \nu_R \ \forall i$, so that (9) is equal to (8), which is not feasible due to the variability of drift coefficients. Results in [25] and [26] have shown that a $g_R = g^{\mathrm{M}}$, where g^{M} is the average conductance among the employed $g_{i,j}$ values, maximizes the drift mitigation.

B. SPREAD MITIGATION WITH MULTIPLE REFERENCE CELLS

Previous analyses on drift compensation consider only the mean value of cells conductances (i.e., of the matrix coefficients). As anticipated with of Fig. 1(b), it is necessary to also include in this evaluation the low-frequency conductance fluctuations and drift coefficient variability of PCM cells, which lead conductances with the same programming level $g(t_0)$ to increase their standard deviation over time.

As each weight $w_{i,j}$ is derived from the ratio of two noisy sources, its equivalent spread is increased with respect to a single PCM device. Let us define $\sigma_g(t)$ as the conductance spread in a generic time instant t of cells programmed to the same level, and $\sigma_w(t)$ as the standard deviation of the conductance-ratio equivalent coefficient distribution. As proven in the Appendix, the increase of σ_w can be

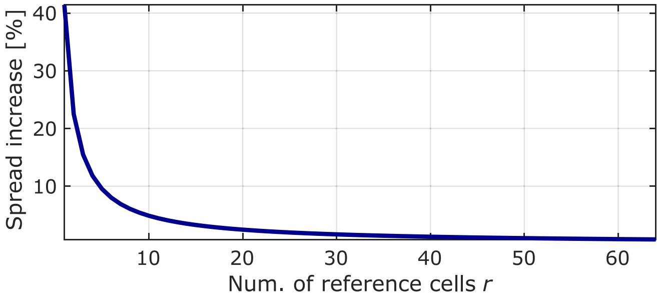

FIGURE 3. Spread increase of MVM coefficients as a function of the number of reference cells, according to (10).

estimated, assuming equal $\sigma_g(t)$ for all conductance levels, as $\sigma_w(t) = \sqrt{2}\sigma_g(t)$. This limitation is overcome in this work by implementing the reference conductance g_R with multiple PCM devices as in this case σ_w is reduced to

$$\sigma_w(t) = \sigma_g(t)\sqrt{1+\frac{1}{r}} \tag{10}$$

where $r \geq 1$ is the number of PCM cells used as reference conductance.

It is evident that the more r increases, the more σ_w tends to be equal to σ_g, which implies the compensation method to not introduce significant overhead in terms of coefficients spread. Exploiting a reference sector implies a decrease of the area efficiency, as a portion of the memory array is thus excluded from computations. However, as reported in Fig. 3, σ_w tends to σ_g with few reference cells compared to common memory IP sizes. As stated in Section V, the current implementation involves $r = 8$ bitlines (BLs) as reference, with a theoretical increase of the coefficient spread around 6% only.

IV. READOUT SCHEME FOR AIMC

A. READOUT SCHEME ARCHITECTURE

The proposed readout circuit is conceived to perform a signed $(n \times m)$-sized MVM operation as in (4) with a PCM conductance-ratio RCCT to mitigate the effect of drift on the computing accuracy. As shown in Fig. 4, the AIMC readout scheme is expected to be interfaced on $2n + r$ BLs of the ePCM array, and it is conceived to prevent the memory IP from being modified. As explained hereafter, the first $2n$ BLs are used to read the matrix coefficients, whereas r BLs are used as reference for RCCT.

The detailed circuital implementation of the readout scheme is reported in Fig. 5. The scheme takes as input the array $\mathbf{x} = (x_1, \ldots, x_n)$, where x_j are digital signed strings. The absolute value of each x_j is internally converted to an analog value V_j, while the sign bits x_j^S are directly connected to the corresponding readout circuit. Each matrix element $w_{i,j}$ of $\mathbf{W}$ is expressed as in (7) $w_{i,j} = g_{i,j}/g_R$, and its sign is represented by $g_{i,j}^S$. PCM devices implementing the coefficients on the same row $\mathbf{w}_i$ are located on the ith wordline (WL) along with the r reference cells. A dedicated memory sector of r-BLs contains the reference cells for RCCT.

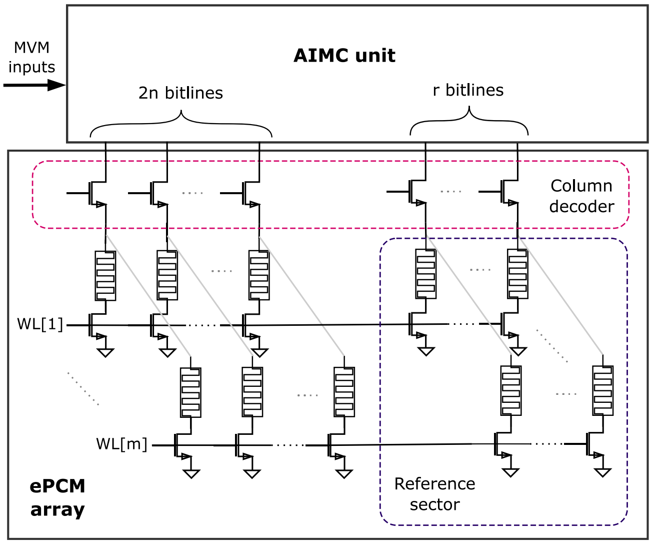

FIGURE 4. Schematic of the ePCM and its interface with the AIMC unit.

The reference BL biasing circuit sets the BL read voltage V_R of the reference conductance g_R. According to the waveforms of Fig. 5, when the START signal switches to logic low, a current $I_R = g_R V_R$ is integrated on capacitance C_R, generating a ramp signal $V_{\text{RAMP}}(t)$ starting from V_0

$$V_{\text{RAMP}}(t) = \frac{I_R}{C_R}t + V_0. \tag{11}$$

The same reference read voltage V_R is applied on each weight cell $g_{i,j}$ through n BL biasing circuits. Each corresponding current $I_j = g_{i,j}V_R$ is then sourced to or sunk from the output node according to the sign of the product $w_{i,j}x_j$, which is obtained combining the corresponding sign cell $g_{i,j}^S$ value and x_j^S sign bit, as further described. The jth current path is then enabled for a time window T_{ON_j}, which begins at the START falling edge and ends when the output of the jth comparator switches to logic low, i.e., when $V_{\text{RAMP}}(t) = V_j$. According to (11), the time window

$$T_{\text{ON}_j} = \frac{C_R\left(V_j - V_0\right)}{g_R V_R} \tag{12}$$

is the time-coded version of the input x_j. As widely demonstrated by [5], the adoption of time modulation of inputs with cells being read at fixed voltage addresses cells I–V characteristic nonlinearity.

The amount of charge Q_i at the output node

$$Q_i = \sum_{j=1}^{n} \pm I_{i,j}T_{\text{ON}_j} = C_R \sum_{j=1}^{n}\left[\pm\frac{g_{i,j}}{g_R}\left(V_j - V_0\right)\right] \tag{13}$$

implements the analog computation of MAC z_i as in (5), where $w_{i,j}$ is represented as the conductance ratio $g_{i,j}/g_R$ and each x_j is mapped into $(V_j - V_0)$. The RCCT is circuitally achieved with the slope of ramp $V_{\text{RAMP}}(t)$ decreasing in

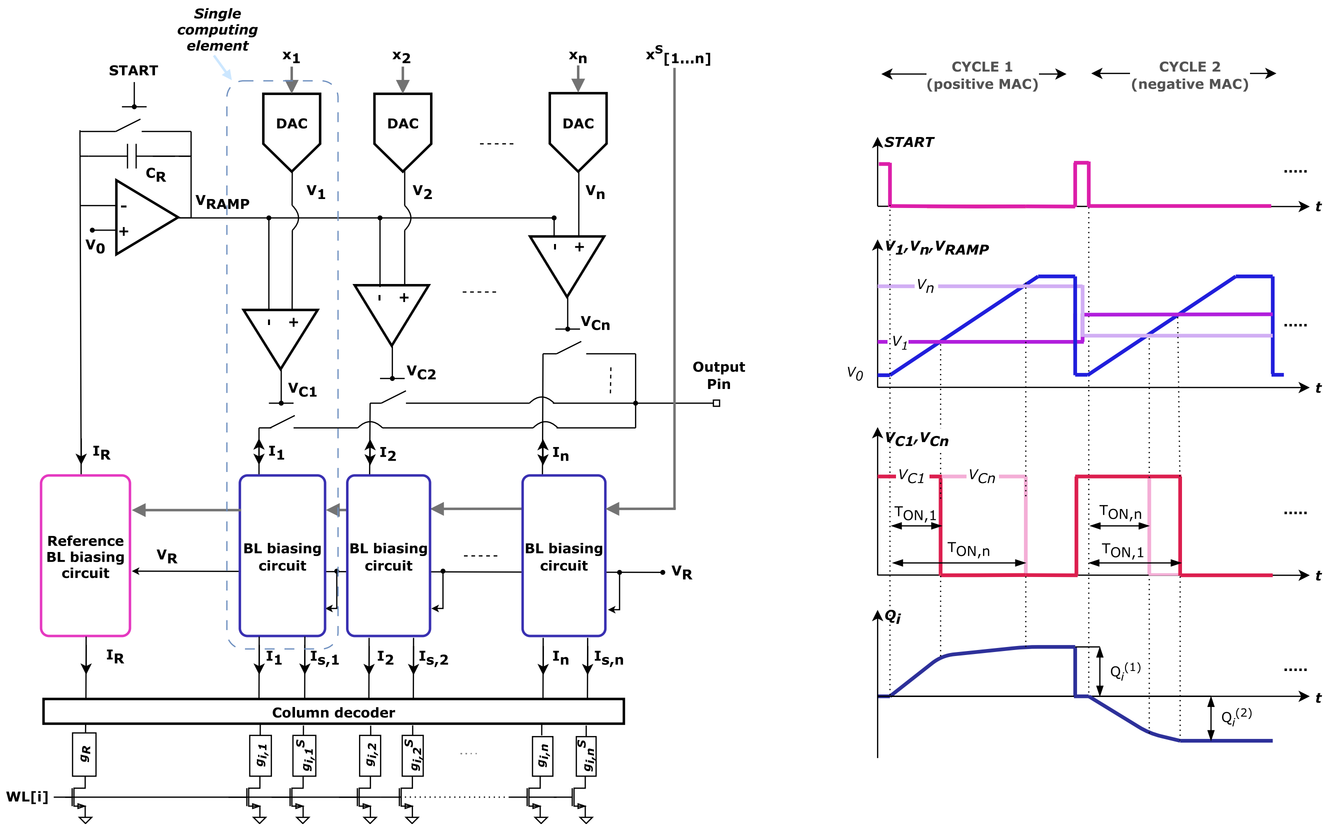

FIGURE 5. (Left) Circuital realization of the AIMC unit. (Right) Waveforms of two consecutive MAC operations, one positive and one negative.

accordance with the reference conductance drift; this leads to an increase of T_{ON_j}, which compensates for the drift-induced drop of weight cells conductances.

It is worth noting that applying the MVM inputs on the BLs requires a sequential activation of m consecutive WLs to perform a full MVM operation of size $(n \times m)$. However, the proposed compensation method does not lose generality, as it is applicable also when MVMs are executed in a single step by applying the inputs on the WLs.

B. BITLINE BIASING CIRCUIT

The BL biasing circuit, shown in Fig. 6 (left), is implemented with a current–voltage mirror (CVM) and a voltage regulator. The voltage of the source terminal of transistor M_4 (i.e., the ith BL' node) is forced to the read voltage V_R using the voltage regulator. Feedback from the source of M_4 is provided to the noninverting input of the amplifier, while the inverting input is connected to V_R. A single-voltage regulator can be shared among more than one CVM.

The CVM, composed of the two coupled transistor (M_1–M_4 and M_2–M_3), is a well-known circuit topology widely used in bandgap reference circuits [46], [47], [48]. Transistors M_2–M_3 are arranged as current mirror to grant $I' = I$; further, transistors M_1–M_4, sharing the same gate voltage and having matched drain currents, provide a voltage mirror to have equal potentials on BL and BL' nodes. In other words, the CVM provides a voltage feedback that equalizes the gate-source voltages of transistors M_1 and M_4. Thus, neglecting the voltage drop through the column selector M_S, the reference voltage V_R is applied to the PCM cell as well.

The drain–source voltages of the paired transistors (M_1–M_4) and (M_2–M_3) are equal if $V_X = V_Y$ (such that $V_{\mathrm{BL}} = V_{\mathrm{BL'}}$), which is only in the ideal case granted by the circuit topology. Nevertheless, channel-length modulation makes I'_j being different from I_j, leading to inequalities between the V_{GS} voltages of M_1 and M_4. This translates into a systematic voltage offset error on BL and BL' nodes, afflicting the accuracy of the MVM computation. As explained in Section V-B, the voltage offset is compensated in this work by tuning the MVM weights cells conductance during the programming phase.

The mirroring factor α of M_1–M_4 and M_2–M_3 can be chosen to get $I'_j < I$, with an advantage in terms of power consumption. Considering $\widehat{I} = g^{\mathrm{M}} V_R$, where g^{M} is the average conductance, the form factors of M_2 and M_4 have been chosen to satisfy

$$V_{\mathrm{DD}} - |V_{\mathrm{GS}_2}|_{I=\widehat{I}} = V_{\mathrm{GS}_4}|_{I'=\widehat{I}/\alpha} + V_R \qquad (14)$$

which implies $V_X = V_Y$ when $I = \widehat{I}$, and thus voltage offset is minimized when $g_{i,j} = g^{\mathrm{M}}$.

The reference BL biasing circuit of Fig. 5 is equal to the one shown in Fig. 6 (left), with $g_{i,j} = g_R$ and $I_j = I_R$. The employment of more reference cell conductances g_R

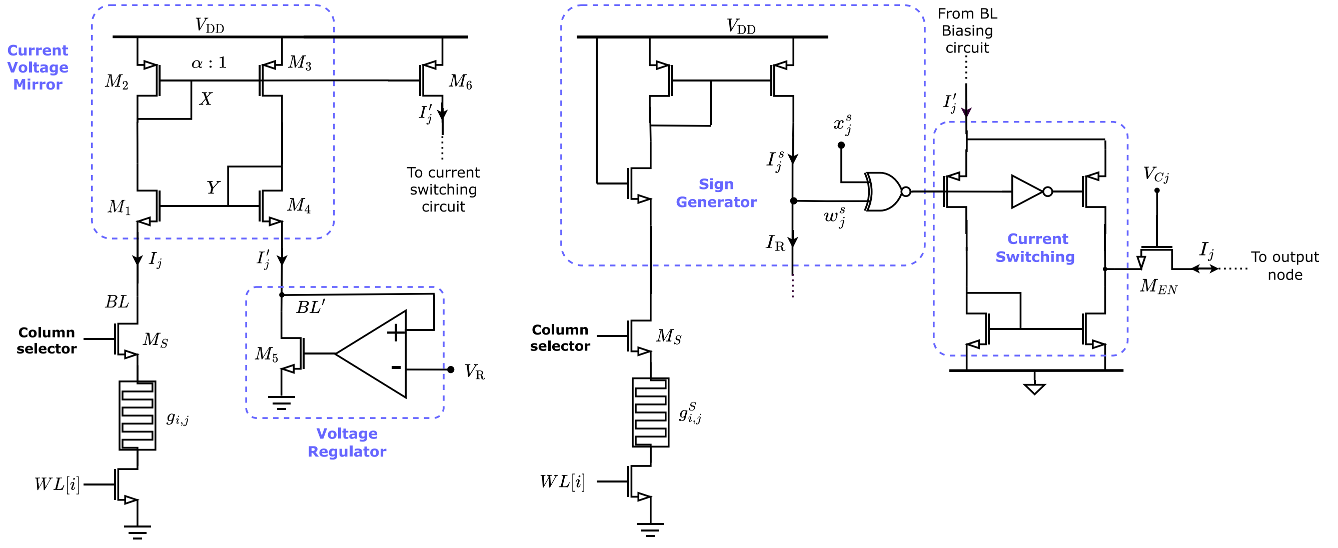

FIGURE 6. (Left) Schematic of BL biasing circuit. (Right) Schematic of the sign generation circuit.

is supported by summing r cell currents of the reference BL nodes, which are then scaled with a mirroring factor α equal to r, and then integrated on the ramp capacitance C_R. As a result, the expression of the integration time window T_{ON_j} in (12) is still valid, so that the drift compensation mechanism is not affected.

C. SIGN GENERATOR CIRCUIT

While the value of I_j represents the MVM coefficient $w_{i,j}$ magnitude, its direction encodes the sign of the MAC product $w_{i,j}x_j$. The sign of $w_{i,j}$ is stored in an additional cell $g^s_{i,j}$, whose current $I^s_j = g^s_{i,j}V_R$ is read with a sign generator circuit through its BL node [see Fig. 6 (right)]. Each sign current I^s_j is compared with a mirrored version of the reference current I_R. The result of the current comparison is a logic signal w^s_j that represents the sign of $w_{i,j}$: if $I^s_j < I_R$, this being indicative of a positive sign, w^s_j voltage value is logic low, whereas, when $I^s_j > I_R$ (i.e., a negative sign), w^s_j is logic high. Since the sign cell does not require high programming accuracy, a fine control of the sign BL voltage is not required.

The sign of the whole product $w_{i,j}x_j$ is the output of the XOR between w^s_j and the input sign bit x^s_j. As a final step, the current $\pm I_j = g_{i,j}V_R$ is sourced to or sunk from the output node according to the output of the sign bit generator, and the current flow is modulated in time with the output of the jth comparator on the gate of M_{EN}.

V. IMPLEMENTATION AND EXPERIMENTAL RESULTS

A. TESTCHIP IMPLEMENTATION

The test vehicle, designed in a 90-nm STMicroelectronics CMOS technology, is mainly intended to validate the drift compensation technique, and includes a single 128-kB ePCM array [31] interfaced with the AIMC scheme.

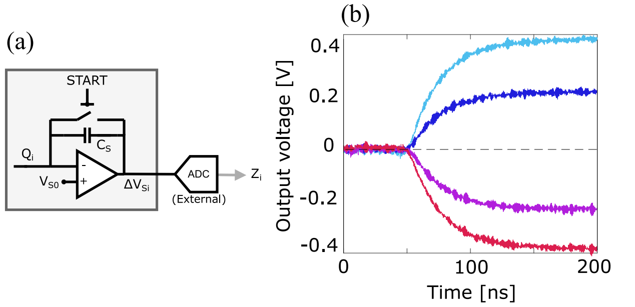

FIGURE 7. (a) Output charge Q_i (13) is internally converted with an integrator to ΔV_{Si} and then accessed through an external ADC. (b) Measured waveforms of ΔV_{Si} of two positive and two negative MAC.

As illustrated in Fig. 7(a), the electrical charge Q_i in (13) is converted by an analog integrator with feedback capacitance C_S, to the voltage variation

$$\Delta V_{S_i} := V_{S_i} - V_{S0} = \frac{C_R}{C_S}\sum_{j=1}^{n}\left[\pm\frac{g_{i,j}}{g_R}\left(V_j - V_0\right)\right]. \quad (15)$$

Some waveforms of ΔV_S are reported in Fig. 7(b). The integrator has been included in the macro and is synchronous to the START signal employed in the computation, and its output is converted with an external 12-bit ADC to the digital string z_i, which corresponds to the single MAC operation defined in (5).

The prototype includes a readout scheme with $n = 12$ inputs, with 4-bit magnitude plus sign. The limited size n of the operation is due to a constraint on the area of the feedback capacitance of the integrator. The circuit layout is indeed reported in Fig. 8, with a zoom on the AIMC scheme. Each single computing element, which includes the DAC, the comparator, and the BL biasing circuit (Fig. 5), uses one BL for the weight cell and one BL for the sign cell. Its layout has been conceived to fit the equivalent

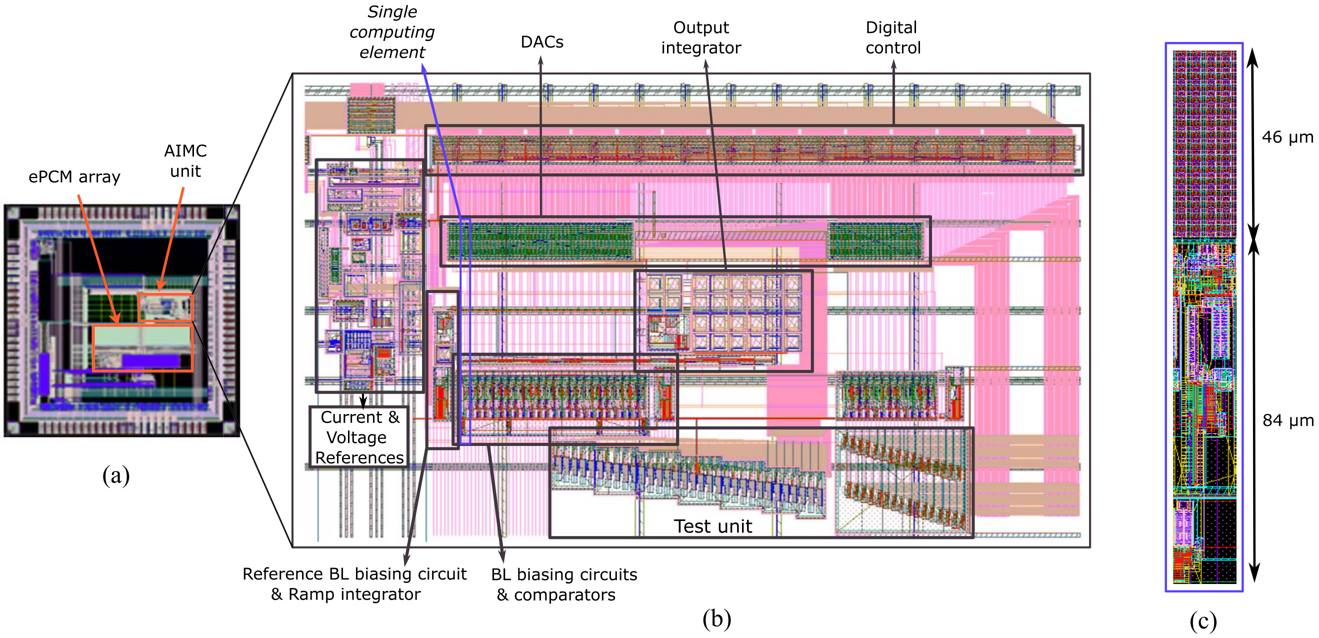
ePCM array
AIMC unit
(a)
Single computing element
DACs
Output integrator
Digital control
Current & Voltage References
Test unit
Reference BL biasing circuit & Ramp integrator
BL biasing circuits & comparators
(b)
46 μm
84 μm
(c)

FIGURE 8. (a) Testchip micrograph. (b) Layout of the AIMC unit. (c) Layout detail and height occupation of a single computing element.

pitch of four BLs of the ePCM, with an overall 130-μm height corresponding to a 17% increase of the memory IP height. A more optimized layout to fit only two memory BLs could be designed with approximately doubled height and same area occupation. The integrator is the largest block due to the large required capacitance, and this limitation, as well as the reduced number of inputs n, is specific to the presented prototype and could be overcome using more efficient specific charge-based ADCs (as, for example, in [21] and [49]).

Circuits have been designed with nominal $V_{\mathrm{DD}} = 1.2$ V and $V_R = 0.3$ V, with PCM cells currents ranging from hundreds of nA to tens of μA. The minimum time required to perform a single MAC operation is 150 ns, while the maximum output voltage ΔV_S is ± 400 mV. The energy consumption of the AIMC scheme is dominated by the weight and sign cells currents, which becomes even more significant with increased MAC size n. In the current implementation with $n = 12$, the mean energy required to perform a MAC operation is approximately 41 pJ (including all circuits of Fig. 5, and excluding the output integrator, which adds approximately 10 pJ). The RCCT implies the use of $r = 8$ reference cells plus the ramp integrator and a BL biasing circuit. Area and consumption of the RCCT-related circuits are 18% and 21% of the total area and consumption of the AIMC scheme with $n = 12$, respectively. It should be noted that this is not an issue, as RCCT area and consumption do not scale with the MVM size n, and r can be left unchanged even with increased n.

The macro integrates also a test unit [see Fig. 8(b)], which is used to generate the ramp of (11) with a programmable current source to disable the RCCT feature.

B. PROGRAMMING ALGORITHM WITH WEIGHTS TUNING

In previous results [25], [26], each cell was programmed with an iterative program-and-verify SET stair-case (SSC) procedure [50] verifying the measured conductance is equal to a target value up to a predefined tolerance. Differently, to prevent circuit nonidealities from affecting the computing accuracy, in this work, the matrix coefficients $g_{i,j}$ are programmed by targeting a MAC output value. To this purpose, the r reference cells are first programmed so that $g_R = g^{\mathrm{M}}$, defined as the average among the $g_{i,j}$ values. Then, all but the jth input x_j are set to 0, whereas x_j is chosen equal to an arbitrary reference value $\bar{x}$; then, in this condition, the measured MAC output becomes $z = w_{i,j}\bar{x} = (g_{i,j}/g_R)\bar{x}$ which is proportional to the weight $w_{i,j}$, and used as a decision parameter in the algorithm. As a result, the voltage offset on BL nodes and other circuit nonidealities (such as BL voltage drop, mismatches between the mirroring coefficients and on the integrating capacitances) are accounted in the programming phase and thus compensated by tuning the cells conductances. A diagram of the programming procedure is reported in Fig. 9. The amount of time required to program each PCM device is in the order of 10^{-2} s, which is considerably higher with respect to binary programming. This is due to the programming algorithm to be executed by an external microprocessor; a significant speed-up could be achieved with the integration of an embedded microcontroller in the testchip. However, programming time and endurance are not severe constraints in many applications involving AIMC systems thanks to infrequent MVM weights update.

The distributions of 4000 normalized MAC outputs targeting 16 different values are shown in Fig. 10. Targets were

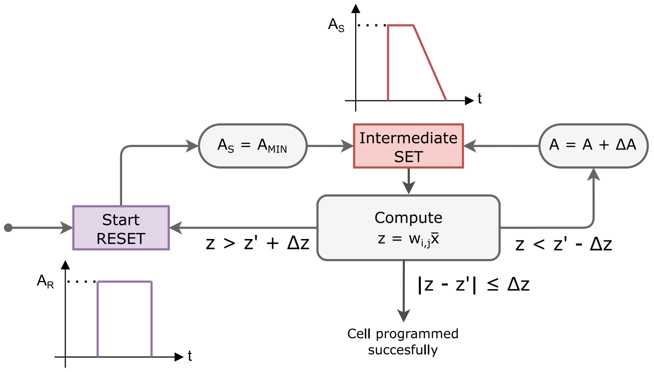

FIGURE 9. Diagram of the algorithm employed to program matrix coefficients. Once the reference has been set, the procedure begins with the application of a start RESET pulse. Then, an SSC sequence begins with a SET pulse of amplitude $A = A_{MIN}$. At each step, the readout scheme computes $z = w_{i,j}\bar{x}$. If the result is below a predefined target interval $z' \pm \Delta z$, an amplitude-increased SET pulse is applied; otherwise, the procedure is restarted.

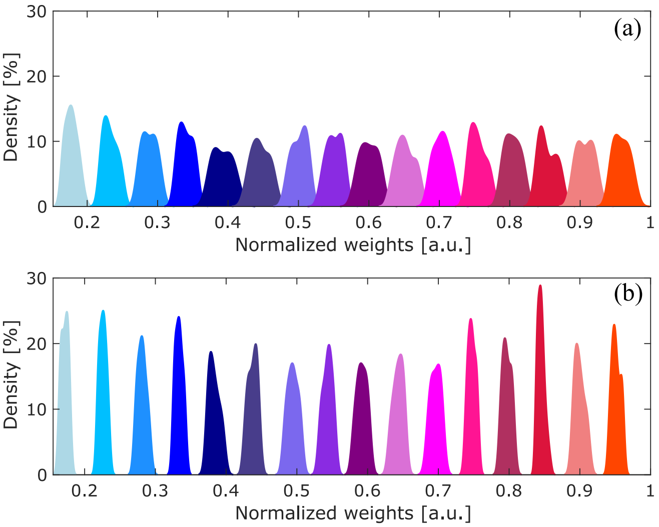

FIGURE 10. Normalized measured MAC coefficients when each cell is programmed targeting (a) conductance level $g_{i,j}$ and (b) MAC result $z = w_{i,j}\bar{x}$. Coefficients are measured as MAC result of the AIMC scheme.

chosen to span the $g(t_0)$-proportional drift interval shown in Fig. 2. Plot (a) shows MAC results when cells were programmed with a standard iterative SSC sequence reading their measured conductance, whereas in the scenario reported in plot (b), the iterative procedure of Fig. 9 was employed. It is evident that the latter weight distributions exhibit a lower spread σ_w.

C. DRIFT COMPENSATION

The drift compensation technique was first tested by evaluating the evolution in time of the normalized MAC outputs shown in Fig. 10(b), i.e., when the MAC output is proportional to a single $w_{i,j}$, $z = w_{i,j}\bar{x}$. This test quantifies the worst-case scenario in terms of drift, as its effect on MAC results is not mitigated by combining different inputs and weights. The MAC outputs were monitored just after programming (t_0), after a 12-h time interval first at RT (t_1), then after the testchip being additionally annealed for a total amount of 64 h at 85 °C (t_2) to accelerate the cells drift. Additional intermediate measures are also shown; all measures have been taken at RT.

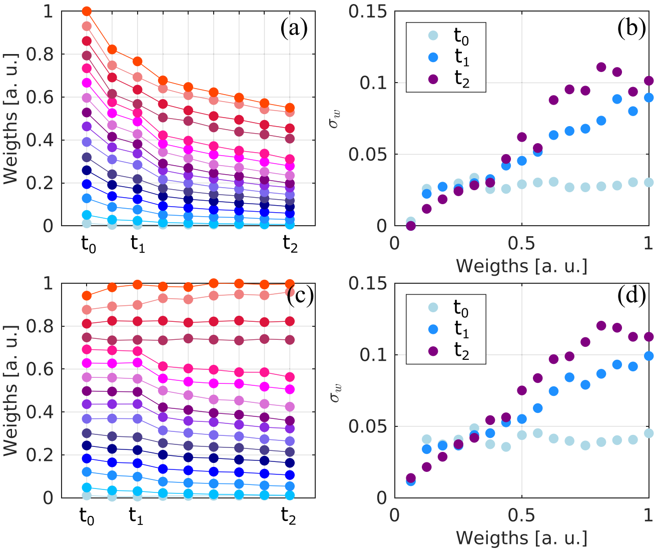

FIGURE 11. Measured normalized outputs z after programming (t_0), after 12 h at RT (t_1), and after additional 64-h bake at 85 °C (t_2) (a) without drift compensation, i.e., with constant reference current; (c) with RCCT compensation with PCM reference current. Plots (b) and (d) report the spread of the MAC weights in t_0, t_1, and t_2.

All MAC were computed first with a ramp signal V_{RAMP} generated with an external current constant in time, thus expecting no drift compensation. Results are reported in Fig. 11(a), which reports the mean values of measured weights $w_{i,j}$. Each curve is averaged on all weights with the same initial value. As expected, MAC outputs tend to decrease in time under the effect of cells conductance drift, which becomes even more evident after the bake. Fig. 11(b) shows the weights standard deviation σ_w, where different curves refer to σ_w evaluated in t_0, t_1, and t_2. The programming algorithm grants approximately equal σ_w in t_0 for all levels, whereas it tends to increase in t_1 and t_2 under the effect of random drift.

The same measurements were repeated with the reference ramp being generated using $r = 8$ PCM reference cells, whose programmed conductances span the whole range of values used in the weight cells $g_{i,j}$. This yields the normalized reference conductance $g_R(t_0) = 0.5$, which maximizes the drift mitigation as shown in [25] and [26]. Fig. 11(c) reports the mean values of compensated weights $w_{i,j}$, with the compensation mechanism recovering the drop in time of results in agreement with (9). The values of coefficient spread σ_w, reported in Fig. 11(d), do not significantly differ from the uncompensated scenario, which underlines the improvement achieved by involving more cells as reference conductance.

The MAC error induced by conductance drift in time, defined as $z(t_0) - z(t)$, is quantified in Fig. 12(a) and (b), and underlines how the compensation scheme confines the results drop under 4% in t_1 and under 18% in t_2, whereas it is

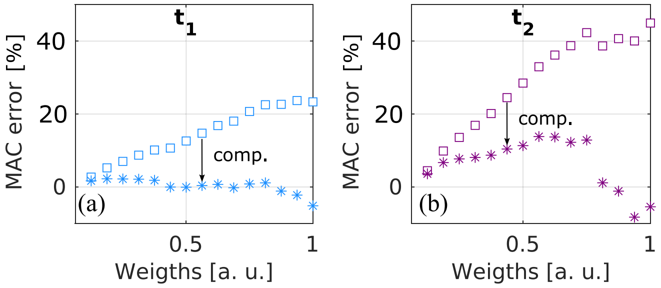

FIGURE 12. MAC error as a function of the normalized weights, evaluated in (a) t_1 and (b) t_2 in the uncompensated (squares) and compensated (stars) scenario.

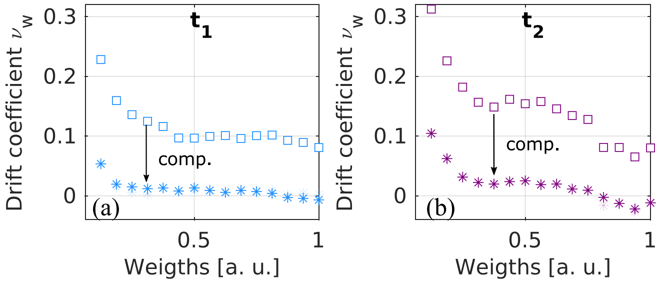

FIGURE 13. Mean drift coefficient ν_w as a function of the normalized weights, evaluated in (a) t_1 and (b) t_2 without RCCT (squares) and with RCCT (stars).

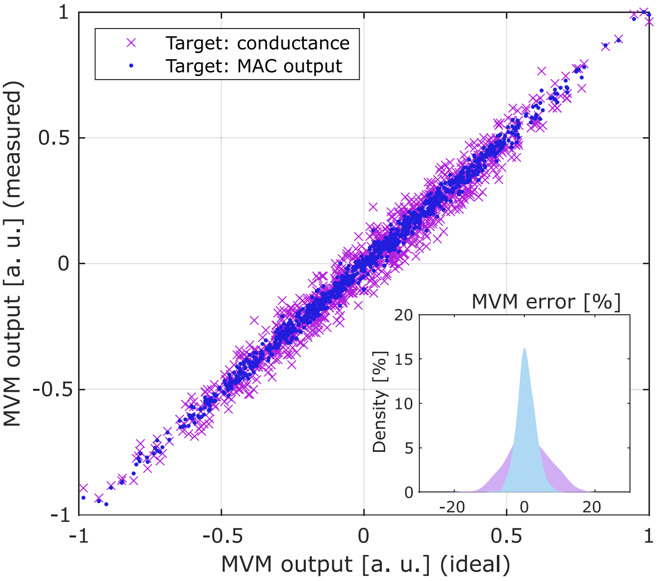

FIGURE 14. Set of 4000 random MVM operations performed immediately after the programming phase. Purple crosses refer to the programming algorithm targeting the cell conductance, whereas the blue dots represent the same set of operations performed targeting the MAC output. The corresponding MVM error distributions are reported in the inset.

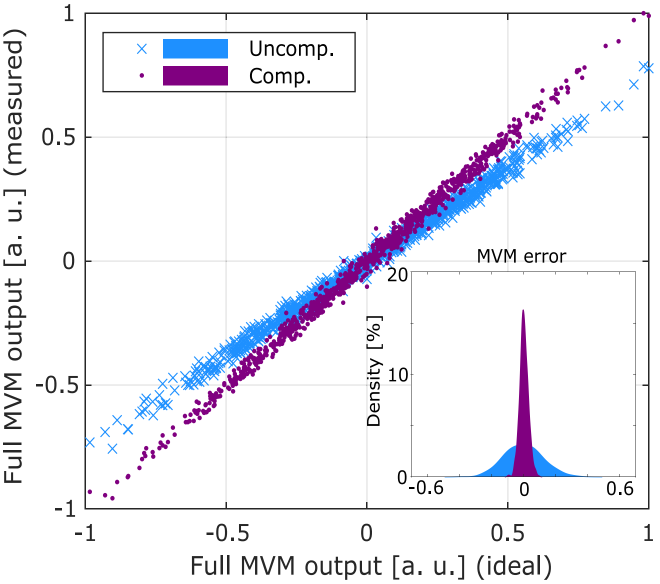

FIGURE 15. Results of MVM operations after 12 h at RT (t_1), without any drift compensation, i.e., with constant reference current (purple crosses) and with PCM reference RCCT compensation (blue dots). The corresponding MVM error distributions are plotted in the inset.

approximately ×3 higher in case no hardware compensation is adopted. A negative MAC error indicates in both t_1 and t_2 a slight overcompensation for the three highest weight levels. A further result is provided by Fig. 13(a) and (b), where the mean equivalent drift coefficient of MVM weights is plotted as a function of the programming weights values in t_1 and in t_2. It is evident that, in the uncompensated case, matrix coefficients tend to assume the drift coefficient of PCM cells (see Fig. 1), whereas ν_w is considerably reduced by the RCCT compensation as expected by (9).

D. MVM OPERATION

To assess the accuracy of MVM operations, a set of ideal MVM results $\mathbf{z}^{\mathrm{id}}$ was defined to cover the full output range, in which random 4-bit signed inputs $\mathbf{x}$ and weights $\mathbf{W}$ are considered, so that $\mathbf{z}^{\mathrm{id}} = \mathbf{W}\mathbf{x}$. Weights were then programmed in the PCM array, and operations were performed on the testchip exploiting the whole $n = 12$ size of the operands. To make the analysis more suitable for the AIMC context, multiple MAC operations are repeated on the chip to extend the results to an MVM as in (4) with $n = m = 512$, which is a standard size of matrices for neural inference.

As a first test, the impact on MVM operations of the proposed multilevel programming procedure, described in Section V-B, was evaluated. The corresponding normalized results $\mathbf{z}$ are plotted in Fig. 14 as a function of $\mathbf{z}^{\mathrm{id}}$, where crosses and dots are referred to the programming algorithm targeting cell conductance and MAC output, respectively. The distributions of the relative MVM error ε, defined as $\varepsilon = \mathbf{z} - \mathbf{z}^{\mathrm{id}}$, are reported in the subplot. Results show that more accurate computations are achieved when using the programming procedure targeting the MAC output, which implies the bounds of ε to be almost halved, and the standard deviation $\sigma(\varepsilon)$ decrease from 5.9% to 2.6%. The MVM accuracy, defined as $1 - \sigma(\varepsilon)$, increases thus from 94.1% to 97.4%.

To test the drift compensation of MVM operations, the outputs reported in Fig. 14 were monitored in time. Results are reported in Figs. 15 and 16, where all the measured

TABLE 1. Comparison table of recent drift compensation techniques.

Reference	[38] (2017)	[44] (2019)	[41] (2020)	[35] (2021)	[45] (2021)	**This work**	
Memory type	PCM	ReRAM	PCM	PCM	PCM	**PCM**	
Technology	GST 90 nm	*n.a.*	GST 90 nm	GST 40 nm	*n.a.*	**GST 90 nm**	
Drift compensation type	RCCT	Network update	Offline training	Network update	RCCT	**RCCT**	
Application	Multilevel starge	Inference	Inference	Inference	Inference	**MVM**	**Inference**
Weights	Multilevel	Multilevel	Multilevel	Multilevel	Binary	**Multilevel**	
Benchmark	Bit Error Rate	DNN accuracy	DNN accuracy	DNN accuracy	DNN accuracy	**MVM accuracy**	**DNN accuracy**
Validation	On-chip	Simulation	Simulation	Simulation	Simulation	**On-chip**	**Simulation**
Max accuracy drop	*n.a.*	8.2%	6%	1%	0.3%	**2.3%**	**3%**
High temp. included	no	yes	no	no	no	**yes**	

* Full MVM operations of size $n = m = 512$ are obtained combining several MAC operations.

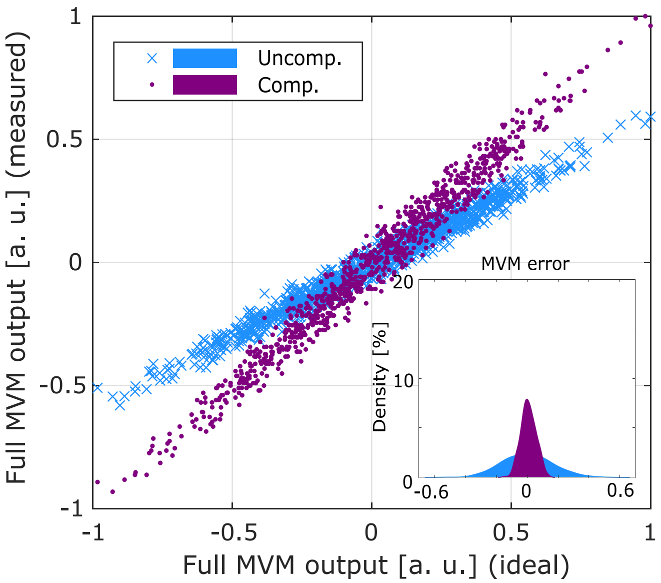

FIGURE 16. Results of MVM operations after 64-h bake at 85 °C (t_2), without any drift compensation, i.e., with constant reference current (purple crosses) and with PCM reference RCCT compensation (blue dots). The corresponding MVM error distributions are plotted in the inset.

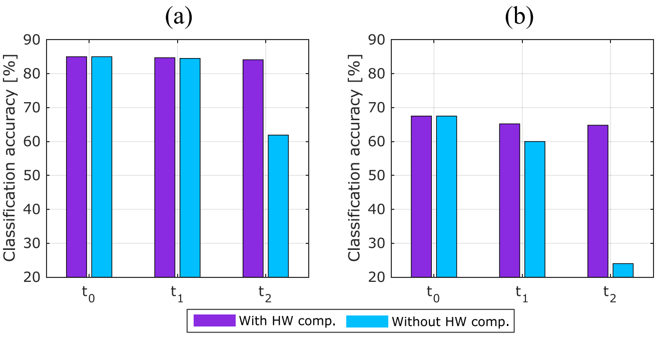

FIGURE 17. Simulated accuracy when conductance drift is applied to the DNN weights during inference, both with and without RCCT compensation. Plots refer to (a) VGG-8 and (b) Lenet-5 DNNs.

normalized MVM outputs $\mathbf{z}$ in t_1 (12 h at RT) and t_2 (64 h at 85 °C after t_1) are plotted as a function of their ideal value $\mathbf{x}^{\mathrm{id}}$. The distributions of the corresponding MVM error are shown in the insets. In t_1 ε varies in the uncompensated case between –49.7% and +50.8%, with $\sigma(\varepsilon) = 11.7\%$, while ε range is reduced to –13.7% and +16.8% and $\sigma(\varepsilon) = 2.7\%$ with RCCT. In case the MVM operations are performed in t_2, ε varies from –53.7% to 54.2%, with a standard deviation of 15.8% with no compensation, while, when RCCT is used, $\sigma(\varepsilon)$ is reduced to 4.9%. Accordingly, the MVM accuracy $1 - \sigma(\varepsilon)$ is equal to 97.2% and to 95.1% at the end of the first and second scenarios, respectively; with respect to the highest value of 97.4% achieved in t_0 (see Section V-B), the MVM accuracy has decreased by 0.2% and 2.3% only. The RCCT compensation keeps the computation error fairly invariant in both the considered scenarios, and, as shown in Figs. 15 and 16, keeps the output voltage swing constant in time.

E. RESULTS DISCUSSION

Table 1 reports a comparison of the most recent drift compensation techniques, evaluated on different technologies and targets, namely, multilevel storage [38] and neural inference [35], [41], [44]. In the first case, a specific drift-resilient procedure with readout code correction is evaluated in terms of bit error rate for PCM multilevel storage. In the other scenarios, drift mitigation is assessed with neural network offline training or update, and the accuracy time drop is targeted. The most drift resilient solution is proposed in [45], which is however achieved with PCM binary programming and without on-chip validation.

Although the current design is conceived to implement only a conductance-ratio RCCT, the results presented in this work are in line with those proposed by the state-of-the-art in terms of achievable MVM accuracy drop. Moreover, the proposed analyses are also supported by on-chip results extended to the drift-accelerated scenario with high temperature. These two conditions are considered only in [44], which however offers a slightly lower performance.

To extend this comparison to the application level, a simulation of inference tasks with MVMs being executed on the proposed testchip (already presented in [26]) has been implemented exploiting the experimental characterization of the AIMC unit immediately after programming (t_0), after 18 h at RT (t_1), and after a 24-h bake at 90 °C (t_2). As depicted in Fig. 17, results show that the proposed hardware compensation technique reduces the long-term effects of cells conductance drift. Indeed, the adoption of the conductance-ratio RCCT yields the classification accuracy drop to be negligible in t_1, whereas in t_2 is kept under 0.2% for a VGG-8 DNN, and 3% for a Lenet5 DNN; if no RCCT compensation was adopted, the accuracy loss would be around 22% and 36%, respectively.

To conclude, the conductance-ratio RCCT provides a DNN accuracy recovery comparable to other solutions. Moreover, the effectiveness of the hardware compensation approach is independent on the particular use case, as it includes operating conditions, such as supply-voltage variations, working temperature, and chip-to-chip variations. Nonetheless, it is reasonable to affirm that greater resilience to drift can be achieved by combining strategies that involve different levels of abstraction, from hardware implementation to neural network topology and their respective training.

VI. CONCLUSION

In this work, a peripheral unit adding AIMC function to a 128-kB ePCM macrocell was presented. The scheme implements a hardware drift compensation mechanism, which is based on an RCCT. Moreover, the unit is conceived to operate with signed inputs and coefficients and does not require any modification to the internal structure of the ePCM. The additional spread induced by the RCCT scheme is overcome using a set of reference PCM cells. Experimental results show that the drift-induced MAC operations drop is reduced by compensation scheme by a factor of 3. MAC outputs are then extended to evaluate MVM operations of size 512, whose accuracy reaches 97.4%, with a negligible decrease in time even after a 64-h bake at 85 °C.

APPENDIX

Let us consider two independent Gaussian variables x and y, and let μ_x, μ_y and σ_x^2, σ_y^2 be their mean values and variances, respectively. Let z be a random variable derived from the ratio of the previous ones, so that $z := x/y$. Under proper assumptions [51], z can be approximated as a Gaussian variable with mean value $\mu_z = \mu_x/\mu_y$ and variance

$$\sigma_z^2 = \frac{\mu_x^2}{\mu_y^2}\left(\frac{\sigma_x^2}{\mu_x^2} + \frac{\sigma_y^2}{\mu_y^2}\right). \tag{16}$$

In the framework of this work, x and y identify a PCM cell conductance g and the reference cell conductance g_R, respectively. Normalizing each conductance with its mean value, so that $\mu_x = \mu_y = 1$, an MVM coefficient w implemented as $w := g/g_R$ shows a variance $\sigma_w^2 = \sigma_g^2 + \sigma_{g_R}^2$. In case g_R is implemented by a single PCM device, one can assume $\sigma_{g_R}^2 \simeq \sigma_g^2$, which yields to

$$\sigma_w^2 = 2\sigma_g^2. \tag{17}$$

Alternatively, if g_R is implemented by r PCM devices, then $g_R = \sum_{i=1}^{r} g_{R_i}/r$. In this case, $\sigma_{g_R}^2 \simeq \sigma_g^2/r$, so that the variance of the MVM coefficient is reduced with respect to the previous one to

$$\sigma_w^2 = \sigma_g^2\left(1 + \frac{1}{r}\right) \tag{18}$$

which proves (10).

ACKNOWLEDGMENT

The authors wish to thank Marcella Carissimi, Chantal Auricchio, and Laura Capecchi from STMicroelectronics (Agrate Brianza, Italy) for their fundamental contribution to the testchip design and development.

REFERENCES

[1] H. Amrouch et al., "Brain-inspired computing: Adventure from beyond CMOS technologies to beyond von Neumann architectures," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, 2021, pp. 1–9.

[2] N. Verma et al., "In-memory computing: Advances and prospects," *IEEE Solid-State Circuits Mag.*, vol. 11, no. 3, pp. 43–55, Aug. 2019.

[3] S. Gao, F. Yang, L. Zhao, and Y. Zhao, "Current research status and future prospect of the in-memory computing," in *Proc. IEEE 14th Int. Conf. ASIC (ASICON)*, 2021, pp. 1–4.

[4] A. Garofalo et al., "A heterogeneous in-memory computing cluster for flexible end-to-end inference of real-world deep neural networks," 2022, *arXiv:2201.01089*.

[5] W. Haensch, T. Gokmen, and R. Puri, "The next generation of deep learning hardware: Analog computing," *Proc. IEEE*, vol. 107, no. 1, pp. 108–122, Jan. 2019.

[6] S. Yin, Z. Jiang, J.-S. Seo, and M. Seok, "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.

[7] S. Yin, Z. Jiang, M. Kim, T. Gupta, M. Seok, and J.-S. Seo, "Vesti: Energy-efficient in-memory computing accelerator for deep neural networks," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 1, pp. 48–61, Jan. 2020.

[8] A. Kneip, M. Lefebvre, J. Verecken, and D. Bol, "IMPACT: A 1-to-4b 813-TOPS/W 22-nm FD-SOI compute-in-memory CNN accelerator featuring a 4.2-POPS/W 146-TOPS/mm^2 CIM-SRAM with multi-bit analog batch-normalization," *IEEE J. Solid-State Circuits*, vol. 58, no. 7, pp. 1871–1884, Jul. 2023.

[9] G. Desoli et al., "16.7 a 40-310TOPS/W SRAM-based all-digital up to 4b in-memory computing multi-tiled NN accelerator in FD-SOI 18nm for deep-learning edge applications," in *Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC)*, 2023, pp. 260–262.

[10] Z. Wan, T. Wang, Y. Zhou, S. S. Iyer, and V. P. Roychowdhury, "Accuracy and resiliency of analog compute-in-memory inference engines," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 18, no. 2, pp. 1–23, Mar. 2022. [Online]. Available: https://doi.org/10.1145/3502721

[11] J. Hartmann, P. Cappelletti, N. Chawla, F. Arnaud, and A. Cathelin, "Artificial intelligence: Why moving it to the edge?" in *Proc. IEEE 47th Eur. Solid State Circuits Conf. (ESSCIRC)*, 2021, pp. 1–6.

[12] Z. Sun, G. Pedretti, E. Ambrosi, A. Bricalli, W. Wang, and D. Ielmini, "Solving matrix equations in one step with cross-point resistive arrays," *Proc. Nat. Acad. Sci.*, vol. 116, no. 10, pp. 4123–4128, 2019.

[13] S. Ricci, P. Mannocci, M. Farronato, and D. Ielmini, "In-memory computing with crosspoint resistive memory arrays for machine learning," in *Proc. Annu. Meeting Italian Electron. Soc.*, 2023, pp. 35–40.

[14] P. Mannocci and D. Ielmini, "A generalized block-matrix circuit for closed-loop analogue in-memory computing," *IEEE J. Exp. Solid-State Computat. Devices Circuits*, vol. 9, no. 1, pp. 47–55, Jun. 2023.

[15] F. Merrikh-Bayat, X. Guo, M. Klachko, M. Prezioso, K. K. Likharev, and D. B. Strukov, "High-performance mixed-signal neurocomputing with nanoscale floating-gate memory cell arrays," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4782–4790, Oct. 2018.

[16] P. Yao et al., "Fully hardware-implemented memristor convolutional neural network," *Nature*, vol. 577, no. 7792, pp. 641–646, 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-1942-4

[17] G. W. Burr et al., "Phase change memory technology," *J. Vac. Sci. Technol. B*, vol. 28, no. 2, pp. 223–262, Mar. 2010. [Online]. Available: https://doi.org/10.1116/1.3301579

[18] G. W. Burr et al., "Recent progress in phase-change memory technology," *IEEE J. Emerg. Select. Topics Circuits Syst.*, vol. 6, no. 2, pp. 146–162, Jun. 2016.

[19] N. Papandreou et al., "Programming algorithms for multilevel phase-change memory," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2011, pp. 329–332.

[20] D. Ielmini and G. Pedretti, "Device and circuit architectures for in-memory computing," *Adv. Intell. Syst.*, vol. 2, no. 7, 2020, Art. no. 2000040. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/aisy.202000040

[21] R. Khaddam-Aljameh et al., "HERMES-core—A 1.59-TOPS/mm2 PCM on 14-nm CMOS in-memory compute core using 300-ps/LSB Linearized CCO-based ADCs," *IEEE J. Solid-State Circuits*, vol. 57, no. 4, pp. 1027–1038, Apr. 2022.

[22] A. Athmanathan, M. Stanisavljevic, N. Papandreou, H. Pozidis, and E. Eleftheriou, "Multilevel-cell phase-change memory: A viable technology," *IEEE J. Emerg. Select. Topics Circuits Syst.*, vol. 6, no. 1, pp. 87–100, Mar. 2016.

[23] D. Ielmini and Y. Zhang, "Analytical model for subthreshold conduction and threshold switching in chalcogenide-based memory devices," *J. Appl. Phys.*, vol. 102, no. 5, Sep. 2007, Art. no. 54517. [Online]. Available: https://doi.org/10.1063/1.2773688

[24] M. Bertuletti, I. Muñoz-Martín, S. Bianchi, A. G. Bonfanti, and D. Ielmini, "A Multilayer neural accelerator with binary activations based on phase-change memory," *IEEE Trans. Electron Devices*, vol. 70, no. 3, pp. 986–992, Mar. 2023.

[25] A. Antolini et al., "An embedded PCM peripheral unit adding analog MAC in memory computing feature addressing non linearity and time drift compensation," in *Proc. IEEE 48th Eur. Solid State Circuit Conf. (ESSCIRC)*, 2022, pp. 1–4.

[26] A. Antolini et al., "Combined HW/SW drift and variability mitigation for PCM-based analog in-memory computing for neural network applications," *IEEE J. Emerg. Select. Topics Circuits Syst.*, vol. 13, no. 1, pp. 395–407, Mar. 2023.

[27] N. Ciocchini, E. Palumbo, M. Borghi, P. Zuliani, R. Annunziata, and D. Ielmini, "Modeling resistance instabilities of set and reset states in phase change memory with Ge-rich GeSbTe," *IEEE Trans. Electron Devices*, vol. 61, no. 6, pp. 2136–2144, Jun. 2014.

[28] M. Boniardi and D. Ielmini, "Physical origin of the resistance drift exponent in amorphous phase change materials," *Appl. Phys. Lett.*, vol. 98, no. 24, 2011, Art. no. 243506.

[29] I. V. Karpov, M. Mitra, D. Kau, G. Spadini, Y. A. Kryukov, and V. G. Karpov, "Fundamental drift of parameters in chalcogenide phase change memory," *J. Appl. Phys.*, vol. 102, no. 12, Dec. 2007, Art. no. 124503. [Online]. Available: https://doi.org/10.1063/1.2825650

[30] D. Ielmini, A. L. Lacaita, and D. Mantegazza, "Recovery and drift dynamics of resistance and threshold voltages in phase-change memories," *IEEE Trans. Electron Devices*, vol. 54, no. 2, pp. 308–315, Feb. 2007.

[31] M. Carissimi et al., "An extended temperature range ePCM memory in 90-nm BCD for smart power applications," in *Proc. IEEE 48th Eur. Solid State Circuits Conf. (ESSCIRC)*, 2022, pp. 373–376.

[32] P. Fantini, G. Betti Beneventi, A. Calderoni, L. Larcher, P. Pavan, and F. Pellizzer, "Characterization and modelling of low-frequency noise in PCM devices," in *Proc. IEEE Int. Electron Devices Meeting*, 2008, pp. 1–4.

[33] L. Laurin et al., "Unveiling retention physical mechanism of Ge-rich GST ePCM technology," in *Proc. IEEE Int. Rel. Phys. Symp. (IRPS)*, 2023, pp. 1–7.

[34] F. G. Volpe, A. Cabrini, M. Pasotti, and G. Torelli, "Drift induced rigid current shift in Ge-rich GST phase change memories in low resistance state," in *Proc. 26th IEEE Int. Conf. Electron., Circuits Syst. (ICECS)*, 2019, pp. 418–421.

[35] X. Sun et al., "PCM-based analog compute-in-memory: Impact of device non-idealities on inference accuracy," *IEEE Trans. Electron Devices*, vol. 68, no. 11, pp. 5585–5591, Nov. 2021.

[36] H. Y. Cheng et al., "A high performance phase change memory with fast switching speed and high temperature retention by engineering the GexSbyTez phase change material," in *Proc. Int. Electron Devices Meeting*, 2011, pp. 3.4.1–3.4.4.

[37] W. W. Koelmans et al., "Projected phase-change memory devices," *Nat. Commun.*, vol. 6, pp. 2041–1723, Sep. 2015.

[38] W.-S. Khwa et al., "A resistance drift compensation scheme to reduce MLC PCM raw BER by over 100× for storage class memory applications," *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 218–228, Jan. 2017.

[39] S. Kariyappa et al., "Noise-resilient DNN: Tolerating noise in PCM-based AI accelerators via noise-aware training," *IEEE Trans. Electron Devices*, vol. 68, no. 9, pp. 4356–4362, Sep. 2021.

[40] A. Bhattacharjee, A. Moitra, Y. Kim, Y. Venkatesha, and P. Panda, "Examining the role and limits of batchnorm optimization to mitigate diverse hardware-noise in in-memory computing," in *Proc. Great Lakes Symp. VLSI*, 2023, pp. 619–624.

[41] V. Joshi, M. Le Gallo, and S. Haefeli, "Accurate deep neural network inference using computational phase-change memory," *Nat. Commun.*, vol. 11, p. 2473, May 2020.

[42] D. Joksas et al., "Nonideality-aware training for accurate and robust low-power memristive neural networks," *Adv. Sci.*, vol. 9, no. 17, 2022, Art. no. 2105784.

[43] S. Diware, A. Gebregiorgis, R. V. Joshi, S. Hamdioui, and R. Bishnoi, "Mapping-aware biased training for accurate memristor-based neural networks," in *Proc. IEEE 5th Int. Conf. Artif. Intell. Circuits Syst. (AICAS)*, 2023, pp. 1–5.

[44] Y.-H. Lin et al., "Performance impacts of analog ReRAM non-ideality on neuromorphic computing," *IEEE Trans. Electron Devices*, vol. 66, no. 3, pp. 1289–1295, Mar. 2019.

[45] I. Muñoz-Martín, S. Bianchi, O. Melnic, A. G. Bonfanti, and D. Ielmini, "A drift-resilient hardware implementation of neural accelerators based on phase change memory devices," *IEEE Trans. Electron Devices*, vol. 68, no. 12, pp. 6076–6081, Dec. 2021.

[46] B. Razavi, *Design of Analog CMOS Integrated Circuits*, 1st ed. New York, NY, USA: McGraw-Hill, 2001.

[47] P. R. Gray, P. J. Hurst, S. H. Lewis, and R. G. Meyer, *Analysis and Design of Analog Integrated Circuits*, 5th ed. Hoboken, NJ, USA: Wiley, 2001.

[48] Y.-H. Lam and W.-H. Ki, "CMOS bandgap references with self-biased symmetrically matched current–voltage mirror and extension of sub-1-V design," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 6, pp. 857–865, Jan. 2010.

[49] R. Vignali et al., "Designing circuits for AiMC based on non-volatile memories: A tutorial brief on trade-offs and strategies for ADCs and DACs co-design," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 71, no. 3, pp. 1650–1655, Mar. 2024.

[50] A. Antolini et al., "Characterization and programming algorithm of phase change memory cells for analog in-memory computing," *Materials*, vol. 14, no. 7, p. 1624, 2021.

[51] E. Díaz-Francés and F. J. Rubio, "On the existence of a normal approximation to the distribution of the ratio of two independent normal random variables," *Stat. Papers*, vol. 54, pp. 309–323, May 2013.

ALESSIO ANTOLINI received the Ph.D. degree in electronic engineering from the University of Bologna, Bologna, Italy, in March 2023.

He is a Junior Assistant Professor with the Electrical, Electronic and Information Engineering Department (DEI) "Guglielmo Marconi," University of Bologna. Since November 2019, he has been with the "Ercole De Castro" Advanced Research Centre on Electronic Systems for Information and Communication Technologies, where his activities are focused on the physical characterization of phase-change memory devices, and on the design of integrated circuits for analog in-memory computing architectures.

ANDREA LICO (Graduate Student Member, IEEE) received the M.Sc. degree in electronics engineering from the University of Bologna, Bologna, Italy, in 2021, where he is currently pursuing the Ph.D. degree in electronics with the Electrical, Electronic and Information Engineering Department.

In 2021, he joined the Advanced Research Center on Electronic Systems, University of Bologna, as a Research Fellow. His research activities are focused on the design of analog integrated circuit for analog in-memory computing architectures based on phase-change memory.

FRANCESCO ZAVALLONI received the Dr.Eng. degree (Hons.) in electronic engineering from the University of Bologna, Bologna, Italy, in 2021, where he is currently pursuing the Ph.D. degree under the ET-IT Ph.D. programme, which is developed inside the Advanced Research Center on Electronic System.

His research activity focuses on analog in-memory computing, PCM-based programming algorithms, neural networks, and modeling PCM-based system behavior.

ELEONORA FRANCHI SCARSELLI (Member, IEEE) received the M.S. degree in electrical engineering and the Ph.D. degree in electrical engineering and computer science from the University of Bologna, Bologna, Italy, in 1992.

From 1994 to 2004, she was a Research Assistant with the Faculty of Engineering, University of Bologna, where she has been an Associate Professor since 2005 and currently teaching the design of digital integrated circuits. She is a member of the Scientific Committee of the joint STMicroelectronics-ARCES Laboratory. Her main research activities include architecture and circuits for low-power sensing and actuating IoT nodes and for in-memory computing based on phase-change memory.

ANTONIO GNUDI received the Ph.D. degree in electrical engineering from the University of Bologna, Bologna, Italy, in 1989.

He has been an Associate Professor of Electronics with the University of Bologna since 1998. He has been involved in modeling and numerical simulation of electron devices by means of both semi-classical and quantum approaches. Investigated devices include nanometer-size post-CMOS structures (nanowire and double-gate FETs, silicon, and III–V FETs), energy-efficient transistors (tunnel and superlattice FETs), and 2-D channel transistors (graphene and TMD). Recently, he has also been involved in the design of circuits and architectures for PCM-based analog in-memory computing applications.

MATTIA LUIGI TORRES received the M.Sc. degree in electronics engineering from the Politecnico di Milano, Milan, Italy, in 2020.

He is currently an HW Design Engineer with the SmartPower TRD Group, STMicroelectronics, Agrate Brianza, Italy. His main topics of research are focused on PCM memory design and testing.

ROBERTO CANEGALLO received the master’s degree in electrical engineering from the University of Pavia, Pavia, Italy, in 1992, and the Ph.D. degree from the University of Bologna, Bologna, Italy, in 2005.

Since 1992, he has been with STMicroelectronics, Agrate Brianza, Italy, where he worked in technology research and development on multilevel flash memories for five years. In 1998, he was with the Research Laboratory jointly managed by STMicroelectronics and Bologna University as a Project Manager for innovative design of IC solutions. He is currently a Senior Member of the Technical Staff with STMicroelectronics and the Program Manager of the Research and Development Smart Power Technology Joint STMicroelectronics-ARCES University of Bologna Laboratory. He has authored many scientific articles and holds European and U.S. patents. His main activities are in the field of sensors, power electronics, and ultralow-power solutions for Internet of Things application in smart homes.

MARCO PASOTTI received the master’s degree in electronic engineering from the Universitá degli Studi di Pavia, Pavia, Italy, in 1991.

In 1994, he joined STMicroelectronics, Agrate Brianza, Italy, collaborating to the design of digital and mixed analog/digital IC for image acquisition and processing. In 1996, he was engaged in the design of flash memory for analog applications, multilevel storage, and applications to multimedia products. Since 2005, he has been leading the team in designing cost-effective eNVM solutions and, lately, embedded Flash and PCM memories for consumer applications. He is currently a senior member of the technical staff and his current interests are in high-performances embeddable NVM memories, analog in-memory computing, new technologies for nonvolatile memories, and all related application aspects.

Open Access funding provided by ‘Alma Mater Studiorum - Università di Bologna’ within the CRUI CARE Agreement