# IMPACT: A 1-to-4b 813TOPS/W 22nm FD-SOI Compute-In-Memory CNN Accelerator Featuring a 4.2POPS/W 146TOPS/mm$^2$ CIM-SRAM With Multi-Bit Analog Batch-Normalization

Adrian Kneip, *Graduate Student Member, IEEE,* Martin Lefebvre, *Graduate Student Member, IEEE,* Julien Verecken and David Bol, *Senior Member, IEEE*

***Abstract*—Amid a strife for ever-growing AI processing capabilities at the edge, compute-in-memory (CIM) SRAMs involving current-based dot-product (DP) operators have become excellent candidates to execute low-precision convolutional neural networks (CNNs) with tremendous energy efficiency. Yet, these architectures suffer from noticeable analog non-idealities and a lack of dynamic range adaptivity, leading to significant information loss during ADC quantization that hinders CNN performance with digital batch-normalization (DBN). To overcome these issues, we present IMPACT, a 1-to-4b mixed-signal accelerator in 22nm FD-SOI intended for low-precision edge CNNs. It includes a novel 72kB dual-supply CIM-SRAM macro with 6T-based DP operators as well as a multi-bit analog batch-normalization (ABN) unit to bypass the ADC quantization issue. IMPACT embeds the macro within a highly-parallel, channel- and precision-adaptive digital datapath that handles memory transfers and provides input-reshaping capability. Finally, a co-designed CIM-aware CNN training framework accounts for the macro's analog impairments, wherein non-linearity and variability. Measurement results showcase a 4b-normalized computing efficiency of 813TOPS/W at 64MHz for the whole accelerator. Taken aside, the CIM-SRAM macro achieves peak energy and area efficiencies of 4.2POPS/W and 146TOPS/mm$^2$ respectively, surpassing all existing low-precision CIM designs to date.**

***Index Terms*—Computing In-Memory (CIM), SRAM, Current-based Dot-Product (DP), Convolutional Neural Networks (CNNs), Analog Batch-Normalization (ABN), Hardware-Aware Training, 22nm FD-SOI.**

## I. Introduction

As of today, ever more edge applications strive for extensive Machine-Learning (ML) capabilities, ranging from smart sensing to object detection or mobile healthcare monitoring [1], [2]. Yet, in the ongoing end-of-scaling era [3], conventional systems collapse on the von-Neumann bottleneck [4] and can hardly offer enough energy efficiency to run such data-intensive ML algorithms on tiny edge nodes. Therefore, compute in-memory (CIM) accelerators have rapidly become an attractive solution for executing convolutional neural networks (CNNs) thanks to genuine weights stationarity and an excellent fit for massively parallel dot-product (DP) operations.

Nonetheless, SRAM-based CIM architectures suffer from an inherent trade-off between signal-to-noise ratio (SNR) and computing efficiency, which depends on the DP operator type as depicted in Fig. 1. Analog CIM-SRAMs involving dynamic current-based architectures provide both high energy- and area-efficiency, enabled by the use of 6T or 8T foundry bitcells [5], [6], [7]. Yet, they suffer from significant non-linearity and local mismatch, generally restricting their computing precision below a 4b resolution. Charge- [8], [9] or time-based [10], [11] operators have proven to cope with this SNR degradation, suiting designs up to 8b precision while preserving great energy efficiency. These architectures, however, pay the price in terms of bitcell area or sequential input processing, decreasing their area efficiency and throughput [12]. Finally, a recent strife towards digital CIM has emerged to remove analog non-idealities altogether [13], [14], reaching precisions above 8b at the expense of a 10× lower computing efficiency compared to analog CIM.

A large fraction of today's research focuses on enabling moderate-to-high SNR architectures. However, several challenges remain to be lifted in current-based architectures to harness their excellent efficiency while preserving CNN accuracy [15], [16]. Namely, today's big challenge remains to

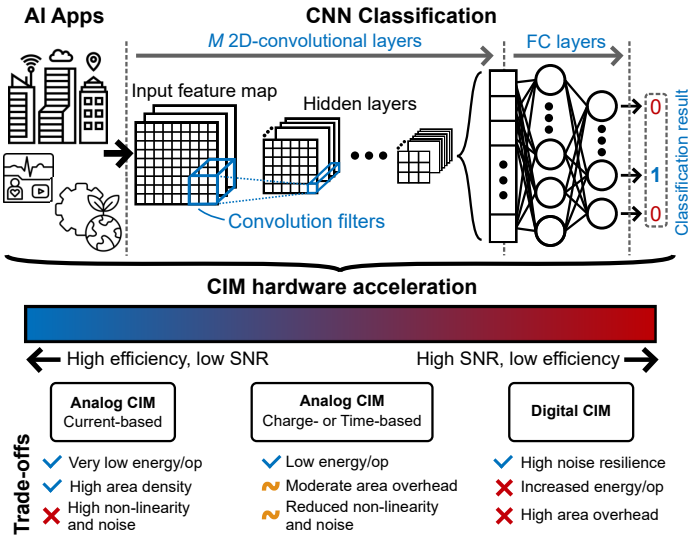


Fig. 1. Ever-growing AI edge applications rely upon classification algorithms such as CNNs, demanding energy-efficient hardware acceleration. Computing in-memory (CIM) offers an enticing solution that spans a broad, architecture-dependent trade-off between computing accuracy and efficiency.

This work was supported by the *Fonds National de la Recherche Scientifique* (FNRS), Belgium, under CDR Grant J.0014.20 and A. Kneip's Research Fellowship. A. Kneip, M. Lefebvre and D. Bol are with the ICTEAM Institute, UCLouvain, Belgium. J. Verecken is with intoPIX, Belgium (corresponding author: adrian.kneip@uclouvain.be).

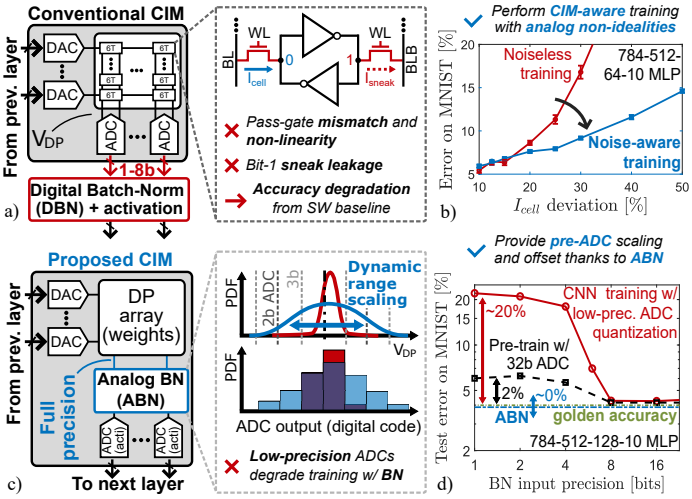


Fig. 2. Challenges and solutions in low-precision, current-based CIM-SRAMs. a) Conventional architecture with 6T-based DP operators, depicting the analog non-idealities altering CNN training accuracy. b) Accuracy degradation on MNIST due to non-idealities and noise. Combining modeling and noise-aware training partly recovers the loss. c) Providing ABN capability before the ADC conversion enables dynamic range scaling adaptability of the DP distribution, leading to d) golden accuracy recovery compared to DBN training with low-precision ADCs.

combine hardware understanding and innovation with an error-resilient software training, hence closing the loop on the full design stack. As such, recent studies on non-idealities affecting current-based DP operators [16], [17] and rapid progress in quantization- and noise-aware training [18], [19], [20] might pave the way for a new wave of low-precision CIM-SRAM designs.

In this paper, we revisit current-based CIM-SRAM design with a close hardware-software development to deliver IMPACT, a 1-to-4b **I**n-**M**emory com**P**uting accelerator with **A**nalog bat**C**h normaliza**T**ion in 22nm FD-SOI. After deriving the main challenges faced by low-precision CIM architectures in Section II, we present IMPACT's novel 6T current-based analog CIM-SRAM macro, featuring multi-bit analog batch-normalization (ABN) ability. Then, Section IV discusses IMPACT's highly-parallel and load-adaptive digital datapath, ensuring an efficient I/O transfer to/from the macro. Finally, Section V presents chip measurement results together with our custom CIM-aware CNN training framework. IMPACT achieves a 4b-normalized total peak efficiency of 813TOPS/W at 64MHz. Independent characterization of the CIM-SRAM macro show a peak energy and area efficiency of 4.2POPS/W and 146TOPS/mm$^2$ with 4b inputs and weights, surpassing all existing low-precision designs to date.

## II. Challenges of Low-SNR CIM Design

To begin with, let us properly grasp the key challenges faced by current-based CIM-SRAM accelerators. The conventional 6T-based architecture is depicted in Fig. 2-a), and is similar to [5]. Sets of DACs/ADCs are respectively responsible for the macro's I/O conversion between analog and digital domains, with the ADC precision ranging from 1 to 8b. Besides, a set of digital post-processing operations is usually applied to the CIM-SRAM outputs for CNN applications, amongst which digital batch-normalization (DBN).

### A. Overcoming Analog Non-Idealities

The analog DP operation with 6T transistors and capacitive discharge has been previously detailed in the literature [21], [16]. In brief, the input-controlled activation of multiple horizontal wordlines (WLs) in parallel lead to the differential discharge of the vertical, precharged bitlines (BL and BLB) by the sum of individual bitcell currents $I_{cell}$. The 0/1 value stored within the bitcell therefore acts as +1/-1 weights, discharging either BL or BLB. The resulting DP voltage in column $j$ can therefore be expressed as a non-linear function $f_{nl}$ of these quantities

$$V_{DP,j} = f_{nl}(T_{DP}, \mathrm{Y}_{DP,j}) \simeq \frac{T_{DP}}{C_{BL}} I_{cell} \sum_{i=0}^{N_{on}-1} \mathrm{X}_i \mathrm{W}_{i,j}, \quad (1)$$

where $\mathrm{Y}_{DP,j} = \mathrm{Y}_{BL,j} - \mathrm{Y}_{BLB,j} = \mathbf{X} \cdot \mathbf{W}_j$ is the expected DP result, $T_{DP}$ the DP operation time, $C_{BL}$ the BL capacitance and $N_{on}$ the number of activated bitcells. Still, Eq. (1) practically suffers from several analog non-idealities. As depicted in Fig. 2-a), mismatch and non-linearity of individual $I_{cell}$ currents distort the DP transfer function, introducing uncertainty on the result [16]. This phenomenon is worsened by the use of reduced WL voltage (typically, half of $V_{DD}$) to prevent data overwrite from sneak bit-1 leakage [5].

These non-idealities degrade the eventual inference accuracy of the mapped CNN, as demontrated in Fig. 2-b) for a simple 784-512-64-10 multi-layer perceptron (MLP) trained on MNIST with 1b inputs and weights. The accuracy severely drops above a 20% deviation on $I_{cell}$ when considering ideal 32b ADCs. Still, this drop can be partly recovered by including non-linearities and injecting noise during training, acting as weights regularization. In this way, the CNN model accounts for the specificities of the given architecture and improves its generalization ability, providing greater robustness against noise [22].

### B. The ADC Precision Bottleneck

Another key challenge is linked to the finite precision of ADCs when using DBN, which is key to the fast and stable convergence of quantized training [23]. As observed in Fig. 2-d), a drop in test accuracy appears below the 8b ADC precision, despite quantization-aware training based on [24]. This degradation comes from a lack of dynamic range scaling of the analog DP result: here, the narrow DP distribution depicted in Fig. 2-c), typical of CNN first layers, does not fit the full-scale ADC input. This leads to important information loss during ADC quantization, which hinders the DBN training ability due to a resulting poor diversity of input data. Pre-training the network with full-precision ADCs before low-precision fine-tuning alleviates the accuracy drop, but does not reach the golden 32b-ADC accuracy level. Yet, extending the analog DP range on a fixed basis would not help as high-valued DP results would then saturate, prohibiting the mapping of deeper CNN layers.

This issue begs for an adaptive ADC dynamic range able to fit different distribution types, and therefore suggests two possible solutions. A first naive way would be to increase

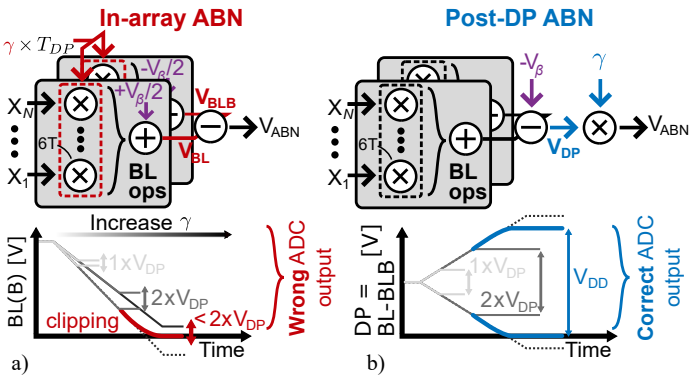


Fig. 3. Comparing qualitative implementations of ABN a) in-array and b) post-DP. The in-array solution suffers from dynamic range clipping, leading to erroneous DP values and wrong ADC quantization. On the contrary, post-DP ABN amplifies the voltage difference up to $V_{DD}$, preventing ADC non-monotonicity.

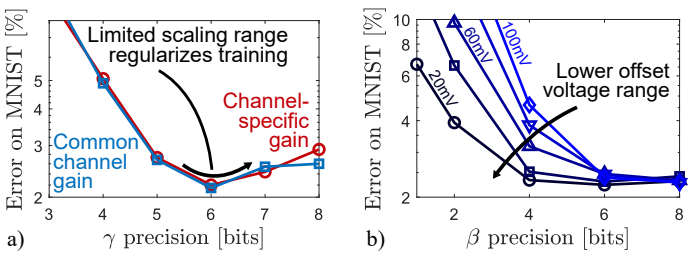


Fig. 4. Evolution of the error on MNIST for the 784-512-256-10 MLP when quantizing a) the ABN gain $\gamma$, b) its offset $\beta$, with a 6b gain. Ideal ABN is modeled as per Eq. (2). Limiting the voltage range of $V_\beta$ improves the accuracy at low $\beta$ precision, suiting hardware mapping.

the ADC precision to 8b or above to accommodate enough resolution for both the narrow and wide ranges of analog DP distribution. However, this solution wastes precision bits to perform scaling, thus incurring significant power and area overheads for low-precision ($\leq$ 4b) design targets. Instead, we propose to bring the DBN scaling and offsetting abilities prior to the ADC conversion, thus performing analog BN (ABN) directly in the voltage domain as

$$V_{ABN} = \gamma\,(V_{DD}/2 + \Delta V_{DP}) + \beta. \qquad (2)$$

$\gamma$ and $V_\beta$ are respectively the ABN scaling and voltage offset parameters. Fig. 2-d) demonstrates a correct recovery of the golden accuracy level with ABN used during training, making it a highly desirable feature for low-precision CIM macros.

### C. ABN Design Constraints

Although promising, the ABN concept has to be considered within the tight design constraints of CIM-SRAM arrays. Earlier works proposed ABN solutions for neural networks employing binary outputs, folding the ABN function into a simple offset applied to the ADC [8]. However, such technique cannot be applied to the multi-bit case, which requires an explicit gain factor. By representing the analog DP operation from Eq. (1) as a series of multiply and accumulate operations in Fig. 3, one can distinguish two conceptual ideas to apply Eq. (2) to the DP result: either (i) performing *in-array ABN* during the DP operation by first scaling both one-sided BL and BLB results, then offsetting the differential DP value ; either (ii) successively applying the offset and gain factors in a *post-DP ABN* fashion.

Let us now consider ways to implement these ideas, knowing the DP operator architecture. *In-array ABN* could entirely rely on the existing 6T-based DP operators, tuning the integration time $T_{DP}$ in Eq. (1) to get a gain factor while using extra bitcells to generate a differential offset. However, one-sided clipping of the BL(B) voltage might occur due to the limited dynamic range, as depicted in Fig. 3-a). Such clipping introduces erroneous DP results at high $\gamma$, resulting in non-monotonic ADC conversions that propagate errors through subsequent computations. Despite being modeled, such non-monotonicity hinders the training ability of the MLP, leading to more than 10% accuracy degradation. In contrast, a *post-DP ABN* solution would directly amplify the voltage difference $\Delta V_{DP}$ up to full-scale, preserving correct ADC decisions as shown in Fig. 3-b). Therefore, this second solution makes a better candidate for implementing in-memory ABN in the case of 6T-based DP operators. Still, it requires to design an additional ABN gain stage, as proposed in Section III.

Finally, let us study the minimum gain and offset resolution needed by the post-DP ABN solution, considering the ideal model from Eq. (2), to limit hardware overheads. Fig. 4-a) showcases that an optimum value of 6b (i.e., $\gamma \leq 64$) gives the smallest accuracy loss. Bounding the scaling range acts as regularization during training, helping with network training convergence. Interestingly, using a common gain $\gamma$ for all output channels rather than channel-specific gains does not alter the obtained accuracy. This relaxes design constraints, only requiring to modify the global timing of the ABN operation for all columns of the macro. Regarding the channel-specific offset in Fig. 4-b), limiting the available voltage range is critical to reduce the minimum required precision. With a 20-to-40mV range, a 4b $\beta$ precision (including the sign bit) is sufficient to avoid significant accuracy loss with minimum hardware overhead.

## III. CIM-SRAM Macro Architecture

The proposed IMPACT accelerator is embedded within the ICare micro-controller, represented in Fig. 5-a). On top of the accelerator, ICare implements clock and power supply generation blocks, an analog front-end for ECG applications [25] and a Cortex-M4 CPU handling chip-level data transfers. The block-level diagram of IMPACT in Fig. 5-b) showcases 4×16kB local SRAM memories (LMEMs) for I/O data storage, used in ping-pong configuration between successive CNN layers to avoid unnecessary transfers. A massively-parallel, precision- and channel-adaptive digital datapath handles data transfers between the LMEMs and the 1-to-4b analog CIM-SRAM macro at the core of the accelerator.

Looking at Fig. 5-c), the custom CIM-SRAM macro features a 1152×512 DP array using 6T bitcells from the foundry. A dual-supply strategy with high/low $V_{DDH/L}$ levels achieves two goals. First, it provides reduced WL voltage to prevent the data overwrite issue during the analog DP operation. Second, it enables a power division strategy, where the power-hungry,

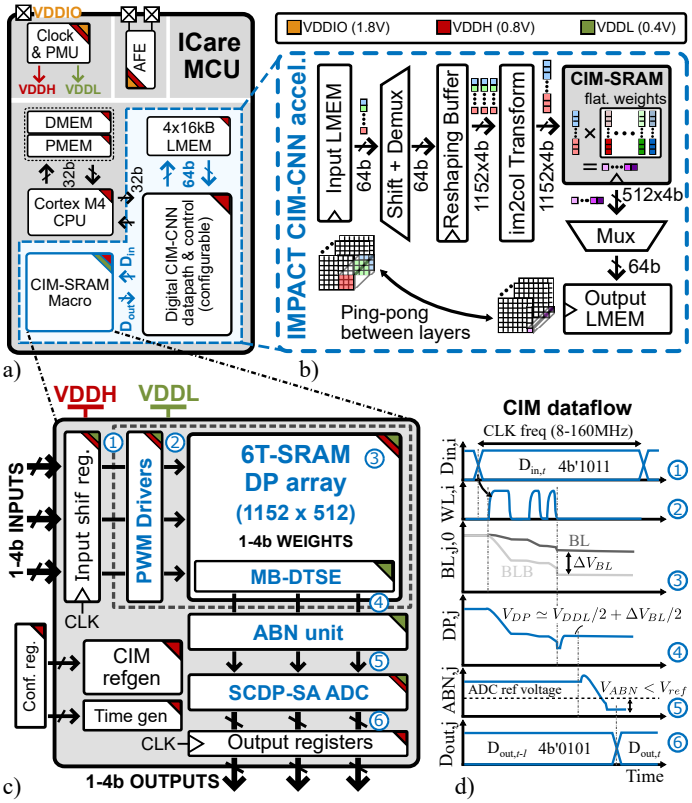


Fig. 5. Top-down architecture of a) the ICare micro-controller unit, focusing on b) the proposed IMPACT accelerator. It provides a massively-parallel, reconfigurable datapath wrapped around c) a custom analog CIM-SRAM macro with multi-bit ABN. d) The macro's 4b operation is qualitatively depicted.

highly-parallel datapath is supplied at $V_{DDL} = 0.4$V while bitcells, periphery and control signals stay at $V_{DDH} = 0.8$V. A qualitative example of the macro's internal operations is shown in Fig. 5-d). First, the 4b digital inputs (1) are converted into a series of pulse-width modulated (PWM) WL signals with binary encoding (2). Read-disturbance issues are avoided by driving the WL to a reduced $V_{DDL} = V_{DDH}/2$ voltage [16], yielding a 3-$\sigma$ deviation on the accessed bitcell nodes below 2.5mV at maximum array utilization. After the 6T-based DP operation (3), single-ended conversion of the differential $\Delta V_{BL}$ happens within the multi-bit differential-to-single-ended (MB-DTSE) blocks (4). The ensuing DP result finally undergoes the ABN operation (5) before a final ADC conversion (6), involving self-calibrating dual-phase sense amplifiers (SCDP-SAs). Below, we detail the different blocks taking part in the analog dataflow, together with their key design challenges. Beforehand, let us note that the macro uses an almost-standard write interface, with $V_{DDH}$ voltage applied to the WL and BL instead of $V_{DDL}$ during conventional write operations. Write operations are sequentially applied to each row by using the input shift-register, removing the need for a dedicated WL decoder. Furthermore, standard read interface was removed to gain area and design time, as it is not necessary for correct CIM-SRAM operations.

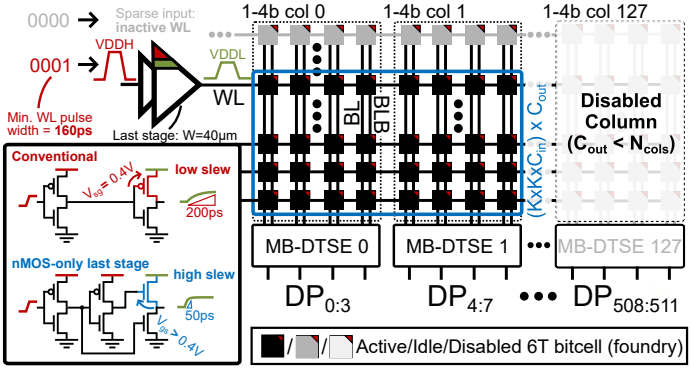


Fig. 6. Architecture of the analog DP array based on 6T foundry bitcells, assembled in sets of four columns to store up-to 4b weights horizontally. Dual-supply WL drivers with an nMOS-only last stage enables super-fast slew despite the reduced WL voltage.

### A. Scalable-Range Current-Based Analog DP

The architecture of the 6T-based DP operator is depicted in Fig. 6, together with its input WL drivers. In order to perform convolutions as a sequence of DP operations, all filter weights of a single output channel are stored per column within $K \times K \times C_{in}$ vertical rows, with $C_{in} \leq 128$ the number of input channel and $K = 3$ the kernel size. Weight bits map to 1-to-4 adjacent columns depending on the weight precision $r_w$, with sets of four columns connecting to a single MB-DTSE unit. In that way, the input-dependent WL pulses are shared across $C_{out}$ different output channels, with $C_{out} \leq 512/r_w$. Inputs are encoded in a pulse-width modulated (PWM) manner as a series of pulses, based on the target input precision. Such encoding takes genuine advantage of input sparsity, as zero-valued input bits correspond to no WL toggle. After precharging BL/BLB to $V_{DDL}$, the DP operation happens as described in Section II.A, with a baseline $T_{DP}$ duration set to 1.5ns. This value is chosen to span half of the $Y_{BL}$ input range, as seen in Fig. 7-a). Yet, such DP duration imposes a tiny LSB pulse width below 100ps in the case of 4b inputs, putting harsh constraints on the WL driver sizing. Limiting the total driver's area overhead to 20% of the DP array, a conventional last stage inverter built with low-$V_t$ devices and supplied at $V_{DDL}$ only achieves a 200ps slew, prohibiting the generation of the sub-100ps LSB pulse. The main bottleneck of the inverter comes from the poor driving strength of its pull-up pMOS, whose $V_{sg}$ is stuck at a low $V_{DDL}$. To solve this, we take advantage of the availability of the higher $V_{DDH}$ supply to create the nMOS-only last stage shown in Fig. 6. The $V_{gs}$ of the pull-up nMOS now evolves dynamically from $V_{DDH}$ to $V_{DDL}$ during the 0-to-1 WL transition. This vastly improves the initial driving strength of the driver while always keeping the nMOS in saturation, reducing the slew down to 50ps at half the size.

The transfer function resulting from the DP operation is reported in Fig. 7-a), highlighting several deterministic and stochastic non-idealities. On the one hand, the DP operator suffers from expected non-linearity due to finite dynamic range and exacerbated dependence of the pull-down current to $V_{BL(B)}$ in such advanced technology node. Furthermore, charge-injection from multi-WL deactivation at the end of the

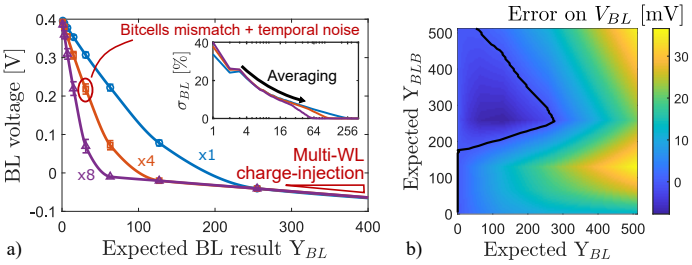


Fig. 7. a) Analog non-idealities alter the one-sided, post-layout DP transfer function ($Y_{BLB} = 0$). Tunable DP duration offers resilience against global variations and pre-gain capability. b) Nominal error on the BL voltage due to charge-injection and sneak bit-1 leakages in differential 6T-based DP operators.

DP operation adds a linearly-scaling offset to both BL and BLB. Its impact is best witnessed at high $Y_{BL}$, as the negative injection brings the voltage of the fully-discharged BL(B) below 0V. On the other hand, various sources of variability have to be carefully considered. To partly cope with global process and temperature variations, we introduce a 3b tunable configuration on the $T_{DP}$ duration. Interestingly, this feature also provides the ability to zoom on a more restricted range of $Y_{BL}$ values, serving as a kind of pre-amplification stage that can be included during offline CNN training. Because of the limited gain range, the saturation issue faced with in-array gain discussed in Section II.C is avoided. Nevertheless, cell-to-cell mismatch remains a dominant contributor to DP variability. Looking at the relative deviation of the DP voltage $\sigma_{DP}$ in Fig. 7-a)'s inset, the error scales with the DP duration but averages out with higher values of $Y_{BL}$ as more bitcells become passing. In order match the actual hardware response, CIM-aware training thus ought to consider these sources of distortion through an accurate modeling.

Furthermore, an error map of the obtained BL voltage with non-zero $Y_{BLB}$ values is shown in Fig. 7-b), underlining yet another source of differential non-linearity. Two regions can be distinguished: negative voltage errors, happening at high $Y_{BLB}$ and low $Y_{BL}$, respectively, and positive ones that increase with $Y_{BL}$ and $Y_{BLB}$. Negative errors are related to the charge-injection issue, while positive errors come from sneak-1 leakage $I_{sneak}$, depicted back in Fig. 2-a). Indeed, when the BL voltage drops close to 0V, bitcells storing a bit-1 becomes weakly passing, injecting current onto the BL and preventing its DC state to reach a full zero. In the end, the charge-injection and sneak leakage effects counteract each other, such that a minimum error appears around $Y_{BLB} \simeq 256$.

### *B. Binary-Weighted Differential-to-Single-Ended Unit*

After the DP operation, sets of four adjacent BL/BLB column pairs enter the MB-DTSE unit. Each unit is composed of four DTSE sub-units, each embedding banks of binary-valued MoM-capacitors as illustrated in Fig. 8-b). The DTSE unit works in two phases described in Fig. 8-c), similar to [26]: during the charge phase (CH), the full capacitance values are connected to BL and BLB to sample the differential voltage, while the output is reset. Then, a precision-dependent subset of the banks is connected to the output node during the share phase (SH), with a flip of the BLB-side capacitance to perform

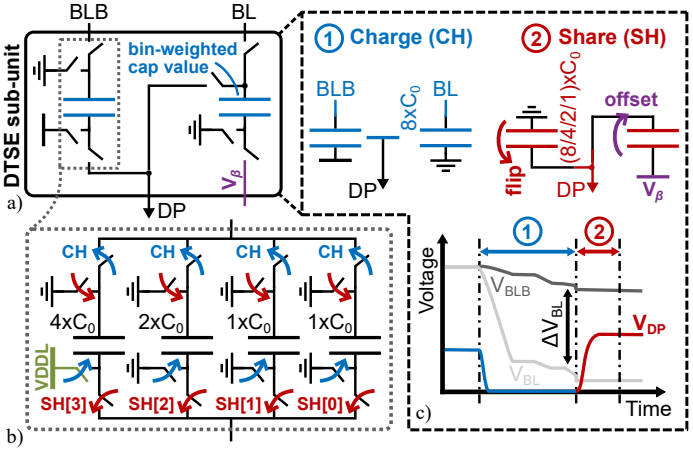


Fig. 8. a) Architecture of the DTSE sub-unit, highlighting b) the binary-valued MoM-capacitance bank controlled by the weights configuration. c) Qualitative depiction of the two-phase DTSE operations.

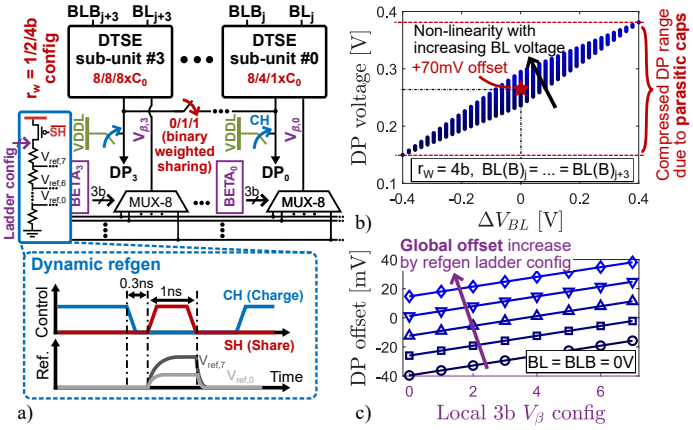


Fig. 9. a) Architecture of the MB-DTSE block, combining DTSE conversion with binary-weighted inter-column charge-sharing to accommodate for 1-to-4b weights precision $r_w$. A 3b column-wise offset voltage $V_\beta$ is selected from 8 global references generated by a dynamically-enabled resistive ladder. b) The post-layout transfer function of the MB-DTSE, altered by parasitic capacitances. c) Configuration of the 3b $V_\beta$ value, performing the ABN offset stage. A 3b tuning of the resistive ladder provides global calibration.

negative charge accumulation. As shown in Fig. 9-a), each MoM-bank configuration is a function of the weights precision $r_w$, which also controls the sharing of adjacent DTSE output nodes. This all results in an ideal DP voltage given by

$$V_{DP} = \frac{V_{DDL}}{2} + \sum_{i=0}^{r_w-1} \frac{2^i}{2^{r_w}-1} \times \frac{\Delta V_{BL,i}}{2}. \qquad (3)$$

Nonetheless, the validity of Eq. (3) is altered by the load and inner parasitic capacitances of the DTSE units. As seen in Fig. 9-b), these capacitances compress the effective DP dynamic range and offset the result towards the output node's precharge value. More dramatically, they introduce a direct dependence on $V_{BL}$ in Eq. 3, leading to the observed non-linearity. Due to the pitch-matched layout constraint, increasing the unit MoM capacitance $C_0 = 1$fF would also increase the routing parasitics, thus not solving the linearity issue. Therefore, we instead directly model the non-linearity at training time.

Moreover, we take advantage of the MB-DTSE block to perform the ABN offset stage. Applying a voltage $V_\beta$ on the

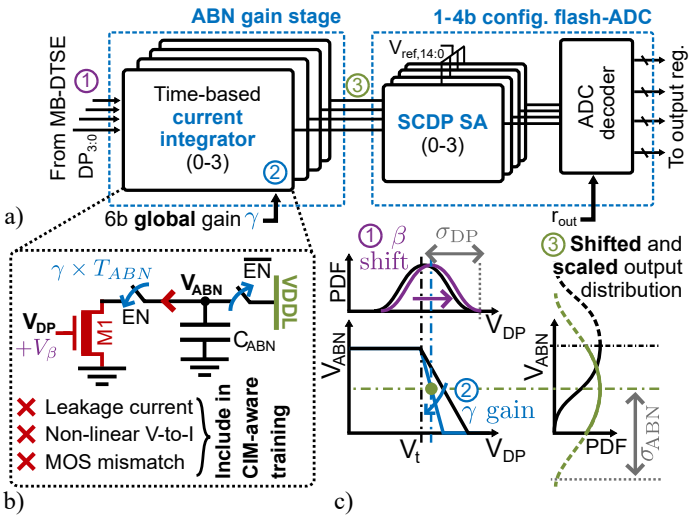


Fig. 10. a) Block diagram of the combined ABN and ADC chain across a set of four adjacent columns. b) Architecture of the ABN gain stage, built out of a simple current-controlled integrator with several known non-idealities. c) Ideal shaping of the DP distribution by the proposed ABN stage.

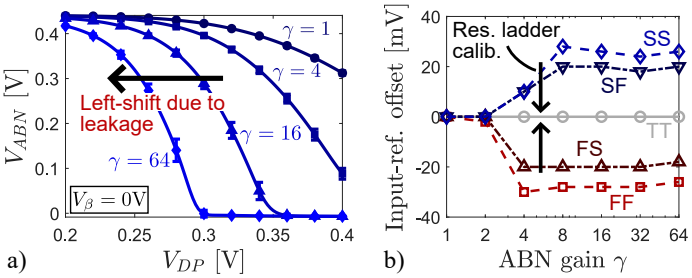


Fig. 11. a) Post-layout response of the ABN gain stage. M1's leakage transforms the gain effect into a voltage shift at high $\gamma$, adding non-linearity. b) Distortion due to PVT variations is dealt with by adapting the resistive ladder.

bottom plate of the BL-side capacitance during the DTSE share phase induces a positive offset on the DP node. A 3b programmable $\beta$ latch selects the column-wise $V_\beta$ from eight reference levels $V_{ref}^{7:0}$. These are generated by a resistive ladder, dynamically activated during the MB-DTSE operation only to avoid unnecessary quiescent power. Negative offsets are obtained by connecting $V_{DDL}$ to the BLB-side's MSB capacitor, and ground to the other ones, as seen in Fig. 8-b). The resulting DP offset shown in Fig. 9-c) has a 20mV range, which can be globally calibrated. Note that we use a 3b offset due to pitch-fitting routing constraints, altough a 4b configuration should be preferred based on discussions from Section II.C.

### C. ABN Gain Stage and ADC Co-Design

With the offset readily applied in the MB-DTSE unit, we only need an additional gain stage to implement the whole ABN function. This unit can be closely co-designed with the ADCs, which architecture can enable different implementations of the gain. A linear gain could simply be obtained by using single-slope ADCs with tunable ramp, similar to [7]. Yet, generating a 6b ramp involves numerous counter toggles, thereby incurring a large power overhead. Instead, we propose the system depicted in Fig. 10-a) that relies upon

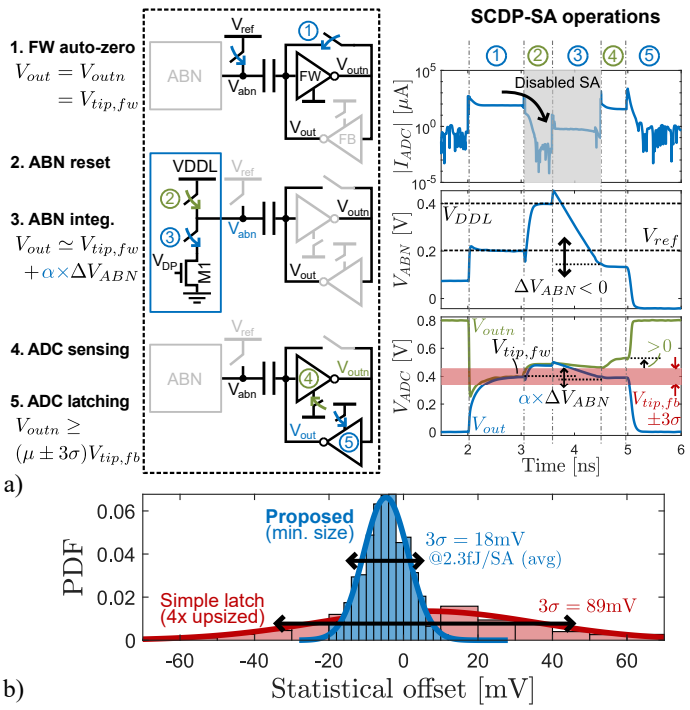


Fig. 12. a) Architecture and operations of the SCDP-SA, used within 1-to-4b flash ADCs and closely co-designed with the ABN gain stage. Disabling the SA during the ABN operation prevents high short-circuit current without loosing the AZ information. b) SCDP-SAs achieve a 18mV 3$\sigma$ input offset with sub-ns and 2.3fJ/SA decision.

a current-controlled integrator followed by dense 1-to-4b flash ADCs. The DP-controlled current integrator shown in Fig. 10-b) provides simple voltage scaling ability by tuning the integrator's discharge time $T_{ABN}$. The expected impact on the DP distribution is depicted in Fig. 10-c), assuming an ideal current source: after properly offsetting the input DP distribution, an increase of the integrator gain $\gamma$ extends the ABN node discharge time, resulting in a larger voltage drop at fixed DP voltage. This behavior matches an eventual increase of the standard deviation of the ABN output distribution. However, to avoid scaling the common-mode voltage, one has to recentre the input distribution around the halfway point of the ABN dynamic range. This additional shift can be encompassed within the total $V_\beta$ offset and considered at training time. On a side note, the ideal ABN unit works similarly to a ReLU function with tunable slope, favouring sparse activations.

Nevertheless, the actual transfer function of the gain stage in Fig. 11-a) gets heavily affected by M1's leakage current, non-linearity and variability. Namely, leakage transforms gain scaling into an additional offset at high $\gamma$, as the transfer function swing lies within M1's weak-inversion range. Consequently, the gain cannot linearly grow larger than a factor 6, well-below the expected 64. Hence, one has to rely upon the non-linearity the ABN response to achieve a higher gain, scaling the common-mode alongside. This asks for a dedicated model of the ABN layer implemented within the hardware-aware CNN training, including both circuits non-idealities and variability. Finally, the transfer function shift induced by global process variations can be modelled as an input-referenced

offset on $V_{DP}$, as in Fig. 11-b). The -30/+25mV offset in SS/FF corners can be compensated by calibrating the MB-DTSE resistive ladder appropriately.

Every CIM-SRAM column includes four ABN-SA pairs, computing up to 2b results. In the 4b case, the results of four neighbouring columns are joined together, imposing $r_w = r_{out}$ to simplify the design. The 15-level reference voltages are generated by another dynamic resistive ladder, with configurable connection to the SAs controlled by the target output precision $r_{out}$. Furthermore, each ABN output directly feeds the input of the SCDP-SAs, as shown in Fig. 11-a). The SA architecture merges the fast and low-power conversion of conventional latch-based SAs with the resilience to offset of single-ended auto-zeroed (AZ) topologies [27]. The SA consists of two latched forward (FW) and feedback (FB) inverters and a 2fF auto-zero (AZ) input MoM capacitance $C_{az}$. The intertwined sequence of operations of ABN and ADC blocks occur along five phases, detailed in Fig. 12-a). First, FW offset compensation takes place while disabling the FB inverter, setting $V_{out}$ to the FW tipping point. Afterwards, ABN operations occur during phases 2 and 3 with both inverters disabled, such that $V_{out}$ tracks the changes in $V_{ABN}$ through the AZ capacitance, with a downscaling factor $\alpha$ due to parasitic charge-sharing. Besides, shutting both inverters down during ABN prevents unnecessary DC currents to flow through while the $V_{out}$ node slowly changes. Once the ABN ends, the offset-free FW inverter is first enabled to probe the $\alpha \times \Delta V_{ABN}$ difference, ensuring that its output $V_{outn}$ resolves in the correct direction. Eventually, if $\alpha \times \Delta V_{ABN}$ is large enough, the resulting change in $V_{outn}$ will overcome the offset deadband of the FB inverter, thereby leading to a correct full-scale decision in the final latching phase. Given that the FW inverter amplifies the small input difference in the sensing stage, only a few mV of voltage swing on $\alpha \times \Delta V_{ABN}$ are necessary for $V_{outn}$ to overcome the FB's deadband, vastly decreasing the overall offset. In the end, the proposed SCDP-SA achieves a sub-ns, 2.3fJ decision. Using only minimum-size transistors, its 3-$\sigma$ input-referred offset deviation remains below 18mV, outperforming 4×-upsized conventional latches by 3.5×.

## IV. CIM-CNN Accelerator Dataflow

In order to leverage the full potential of the CIM-SRAM macro, efficient I/O data transfers between the macro and LMEMs have to be carried out. Moreover, in order to execute convolutional layers efficiently, one has to apply the *im2col* transform [28] to input image data so as to transform convolutions into a series of equivalent dot-product operations, suiting the weight-stationary CIM-SRAM. However, computing this transform with the CPU would waste precious clock cycles and energy. Instead, we propose a dedicated digital datapath to handle CIM I/O transfers, embedding reconfigurable *im2col* input transform acceleration. Below, we detail IMPACT's accelerator-level dataflow and underline the optimal LMEM mapping to minimize I/O transfer latency.

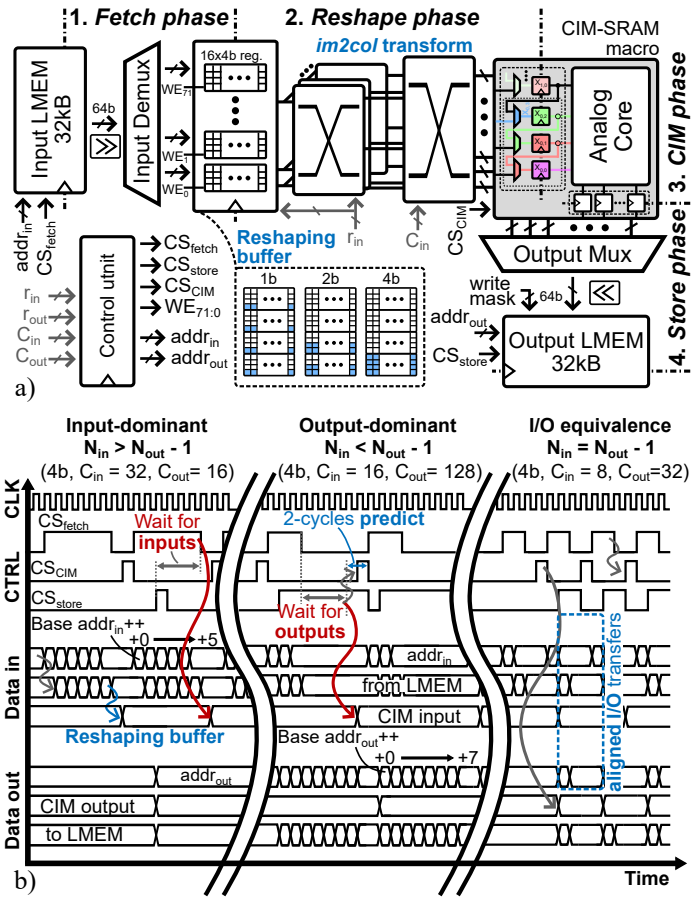


Fig. 13. a) Datapath of the IMPACT accelerator, emphasizing the channel- and precision-adaptive data transform between the input LMEM and the CIM-SRAM macro. b) Depiction of the accelerator's dataflow under various transfer scenarii.

### A. Reconfigurable Datapath Architecture

The architecture of the IMPACT's overall datapath is depicted in Fig. 13-a). The accelerator can be either used in *fully-connected (FC)* or *2D-convolutional (Conv2D)* mode, with the *im2col* transform only used by the second. Seen in a sequential way, IMPACT operates along four phases, detailed below for convolutional mode.

1) *Fetch phase:* First, a full $K \times K \times C_{in}$ image patch is transferred from the input LMEM to the reshaping buffer. This buffer consists of 72 banks of 16×4b registers, each with an individual write-enable signal $\text{WE}_i$. The number of 64b transfer cycles between the LMEM and the buffer depends on the data arrangement within the LMEM as well as on the layer parameters. Assuming a kernel size $K = 3$ and a channel-first input data storage, as depicted in Fig. 14-a), the number of input transfer cycles is given by

$$N_{in} = K \times \text{ceil}\left(\frac{K \times r_{in} \times C_{in}}{\text{BW}}\right), \qquad (4)$$

with the LMEM bandwidth $\text{BW} = 64\text{b}$. This number excludes edge cases, which are discussed in Section IV.B. The 64b data are then properly shifted based on the current fetch address $addr_{in}$ before being written to the reshaping buffer. As depicted in Fig. 13, the written

banks depend upon the input precision $r_{in}$, enabling the precision-wise scaling of $N_{in}$.

2) *Reshape phase:* Once all inputs have been fetched, the highly-parallel *im2col* unit takes the 1152×4b buffered data and reshapes them according to the target input precision and number of channels. The *im2col* stage is bypassed in FC mode given that the flattened input vector is readily available in the input LMEM. In convolutional mode, the transform outputs a channel-last vector to perform the DP operation with the in-memory stored weights, as presented in Section III.A. Importantly, moving to this channel-last format enables the input shift register, reusing two-third of input data thanks to kernel-wise shifting between subsequent CIM-SRAM operations along the same image row.
3) *CIM phase:* With new inputs ready, the CIM-SRAM operation takes place according to details presented in Section III, storing its results into up to 512×2b output registers based on the target precision and channel configuration. Note that the CIM-SRAM accelerator does not support 4b outputs in the case of binary weights due to output registers limitation.
4) *Store phase:* Eventually, registered CIM-SRAM outputs are sequentially transferred by packs of 64b to the output LMEM, with the appropriate shift based on $addr_{out}$. A write mask avoids overwriting results along the convolution operation when $r_{out} \times C_{out} < 64$b, as a single 64b LMEM word then contains multiple convolution results. The number of output transfer cycles is

$$N_{out} = \text{ceil}\left(\frac{r_{out} \times C_{out}}{\text{BW}}\right). \quad (5)$$

Performed sequentially, these four phases would yield a total of $N_{tot} = 2 + N_{in} + N_{out}$ cycles, incurring significant latency. Fortunately, the fetch, store and compute stages can be pipelined to maximize the overall throughput. IMPACT's control unit generates the fetch/store addresses and control signals as a function of the mapped CNN layer parameters. Three different scenarii follow, whose dataflow are depicted in Fig. 13-b) with example configurations.

i) *Input-dominant:* Taking advantage of pipelining, the fetch request for the next input image patch is issued at the end of the buffer-to-CIM transfer stage. If $N_{in} > N_{out} - 1$, storage of the current CIM-SRAM outputs will be done before all new input data could be fetched, incurring an output transfer stall. The overall latency is thus the sum of the input transfer cycles, buffer-to-CIM and CIM computation cycles, yielding a total delay of $N_{in} + 2$ cycles between consecutive CIM-SRAM operations. This number could be reduced to $N_{in} + 1$ by issuing the next fetch request concurrently to $\text{CS}_{CIM}$. However, we kept the first option to avoid the need to handle some complex configuration cases within the control unit.
ii) *Output-dominant:* In typical CNNs, several layers aim at extending the output dimensionality, such that $C_{out} > C_{in}$. If the increase is such that $N_{in} < N_{out} - 1$, fetching of the next input data will be over before all current outputs are safely stored in the output LMEM. The next CIM-SRAM operation is then stalled until store operations are over, as well as input LMEM fetches. Thanks to the clocked buffering of CIM outputs, the last store command is issued while the next CIM operation takes place. As a result, the CIM-to-CIM delay is bounded to $N_{out} + 1$ cycles.
iii) *I/O equivalence:* When $N_{in} = N_{out} - 1$, store operations take exactly as long as input transfers, including the buffer-to-CIM cycle. The CIM-to-CIM delay thus reaches $N_{in} + 2$ cycles.

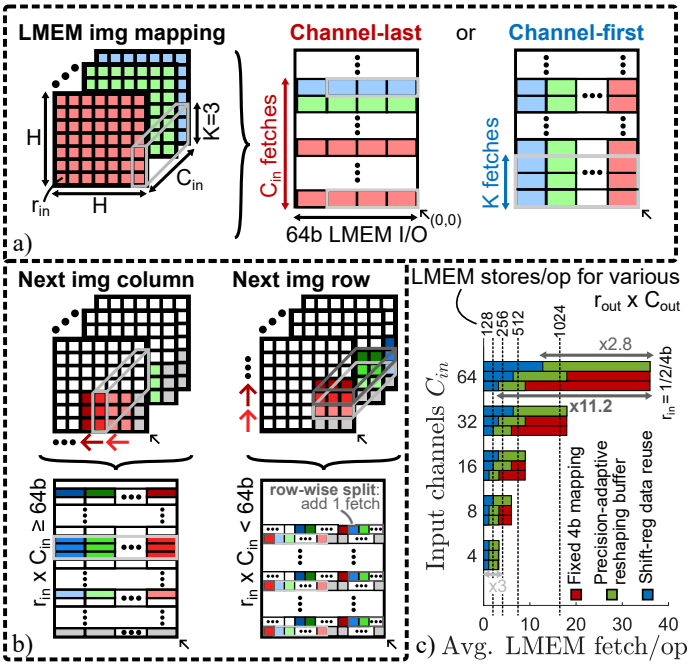


Fig. 14. Mapping of multi-channel input data for 2D convolutional operations. a) Channel-first mapping requires fewer transfers than channel-last. b) Data fetches along row and columns, including edge cases. c) Comparison of fetch cycles with different optimizations, relative to output transfers.

Mapping a complete neural network on IMPACT can be done by performing this sequence of operations until all inputs to layer $l$ have been processed, before loading the weights of layer $l+1$ from an external memory. The input and output LMEMs are then swapped between layers in a ping-pong manner to reuse layer $l$'s outputs as inputs to layer $l+1$, when there is no need for additional inter-layer processing. However, this is only valid for layers whose size is smaller than the CIM-SRAM capacity (i.e. $C_{in} \leq 128$, $C_{out} \leq 128$ with 4b weights). Still, IMPACT could map larger layers by splitting their processing into different slices, successively loading the corresponding weights and storing temporary output results into an off-chip DRAM. This process would be similar to previous accelerators [12], [24] but would induce significant power and latency overheads related to the additional DRAM accesses. With sufficient silicon area available, a solution might be to design a multi-core CIM-CNN accelerator with inter-macro communication similar to [33], being able to map larger layers directly on-chip.

### B. LMEM Data Mapping

The operations described above stand correct provided that input and output data are properly mapped within the LMEMs. Fig. 14-a) compares the channel-last and channel-first storage of an input image of size $\mathrm{H}\times\mathrm{H}$, featuring $C_{in}$ input channels. In the channel-last case, the number of fetch cycles would be given by

$$N_{in} = \mathrm{ceil}\left(\frac{r_{in}\times C_{in}\times \mathrm{H}\times \mathrm{H}}{\mathrm{BW}}\right). \tag{6}$$

Comparing Eq. (4) and (6) where $K=3$ and $\mathrm{BW}=64$b, the channel-first solution always outperforms the channel-last one for $\mathrm{H}\geq 4$, which is safely assumed here. Besides, the channel-last mapping cannot take advantage of input data reuse, wasting LMEM accesses. As for the output image, individual pixels are successively stored in the same channel-first way to minimize $N_{out}$, as given by Eq. (5). Furthermore, using an identical mapping allows to directly reuse the output data of layer $l$ as input data of layer $l+1$, exchanging input and output LMEM in a ping-pong fashion. This makes channel-first mapping even more critical at large H.

The full convolution operation happens by first progressing along the image columns, then its rows, as showcased in Fig. 14-b). To further minimize transfer delays and the total time taken by the convolution, pixel information is stored within the LMEM in a channel-row-column format. This structure enables down to a single-cycle fetch per column of the image patch when $K\times r_{in}\times C_{in} < 64$b per . Besides, providing column-wise data reuse with the CIM-SRAM's input shift register reduces the amount of fetches to a single column slice rather, changing Eq. (4) into $N_{in} = K\times\mathrm{ceil}\left(r_{in}\times C_{in}/\mathrm{BW}\right)$. Fetching $K$ columns is only necessary at the start of a new image row, as the base address of input data fetches is reset and offset by $i\times r_{in}\times C_{in}$, with $i$ the row index. However, for $r_{in}\times C_{in} < 64$b, this offset can lead to a row-wise split of the channels to be fetched across two SRAM words, depicted in Fig. 14-b). These edge cases are handled by the control unit, adding one additional fetch cycle when required.

Finally, we evaluate in Fig. 14-c) the average number of LMEM fetches between successive CIM-SRAM operations across the full duration of a convolution operation, including the new-row overhead cycles as well as edge cases. We also showcase how the precision-adaptive reshaping buffer and input shift-register improve the transfer latency in some, but not all, layer configurations. The improvement is at its highest for large layers and small bitwidth, with up to 11.2× less fetch cycles. This vastly decreases the total input LMEM access energy; however, the actual throughput gain is eventually limited by the output-dominant boundary set by the $r_{out}\times C_{out}$ configuration. Finally, the number of fetch cycles converges to a single one with a low number of input channels thanks to the channel-first mapping.

## V. Training & Measurement Results

The proposed IMPACT accelerator has been fabricated in 22nm FD-SOI and embedded within the ICare micro-controller. The custom test setup depicted in Fig. 15-a) allows

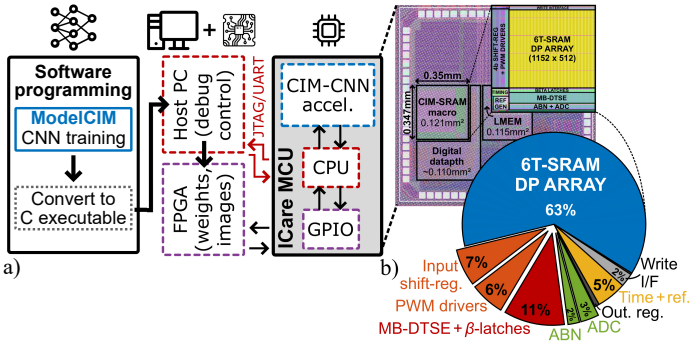


Fig. 15. a) Measurement setup, spanning from offline CNN training to on-chip mapping. b) ICare die micro-photograph, highlighting the IMPACT accelerator and the layout of the CIM-SRAM macro, as well as the macro's area breakdown.

to characterize the accelerator with user-specified input/weight distributions, but also to map the CNN layer weights obtained from the offline CIM-aware training. The chip micro-photograph in Fig. 15-b) highlights the accelerator's area, reaching $0.121\mathrm{mm}^2$ for the CIM-SRAM macro and $0.11\mathrm{mm}^2$ for the digital datapath, excluding LMEMs. An area breakdown showcases that the DP array amounts to 63% of the total macro area. The MB-DTSE unit is the biggest source of overhead due to its MoM-cap layout, while the dense ABN and ADC units combined only occupy 5% of the silicon surface. Below, we successively characterize the macro and accelerator, before addressing the CNN training framework and comparing our results to the state-of-the-art. The results are finely reported for a single test chip, however coarse tests suggest similar trends for other chips.

### A. CIM-SRAM Characterization

The functionality and performance of the CIM-SRAM macro are independently tested through a specific CIM test mode. In that mode, the macro continuously operates on data pre-stored within the reshaping buffer, allowing accurate power extraction. Using uniformly distributed inputs and weights, the measured 4b transfer function of the macro is reported in Fig. 16-a), at nominal supply ($V_{DDL/H} = 0.4/0.8$V), 32MHz and the $T_{DP}$ multiplier set to 3. Changing the ABN gain configuration $\gamma$ yields the expected scaling and offsetting effects, although the actual gain is limited to 4× following the discussion in Section III.C. Moreover, the aggregated deviation across the macro's columns (spatial), averaged across 100 successive temporal samples, ranges from a maximum of 3.3 to 4.2LSB for expected DP values (normalized to 1b) yielding $D_{out}$ results in the middle of the output dynamic. Furthermore, the deviation scales up with $\gamma$ as the ABN gain amplifies noise sources prior to the ABN unit, namely the analog DP mismatch seen in Section III.A.

Keeping fixed supply voltages, the maximum operating frequency of the macro is found in Fig. 16-b) when sweeping both the ABN gain and DP duration. Given that the DP operation takes a larger fraction of the CIM-SRAM cycle, especially at low $\gamma$, the maximum operating frequency drops faster with increasing $T_{DP}$. However, the drop in energy efficiency at the corresponding maximum frequency in Fig. 16-

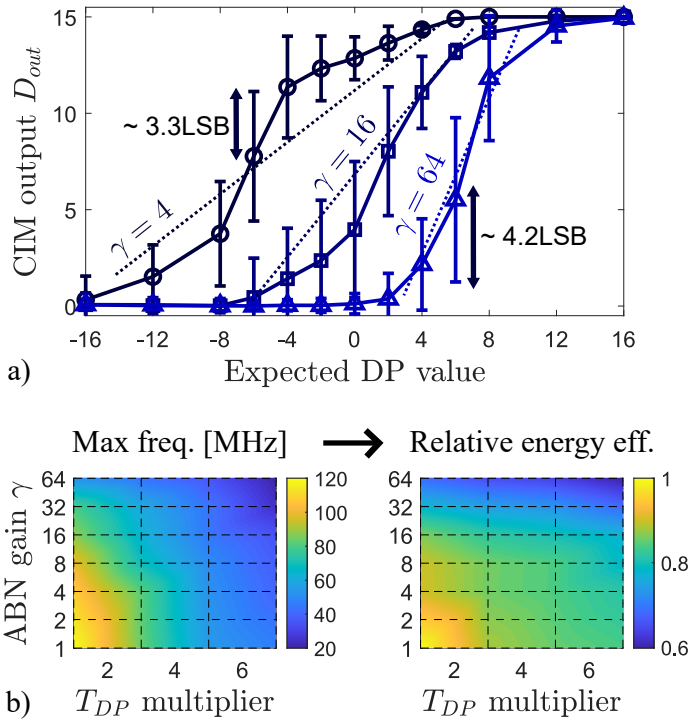


Fig. 16. a) Measured 4b transfer function of the CIM-SRAM macro with increasing ABN gain at nominal supplies $V_{DDL/H}$ = 0.4/0.8V, averaged across 100 successive temporal samples with the same inputs and weights. The high 3.3-to-4.2LSB deviation on $D_{out}$ is the sum of non-linear effects, spatial and low-frequency temporal noises obtained across the macro's output columns for uniformly distributed input and weights. b) The maximum operating frequency drops with increasing DP and/or ABN durations. Increasing these parameters also diminishes the macro's energy efficiency.

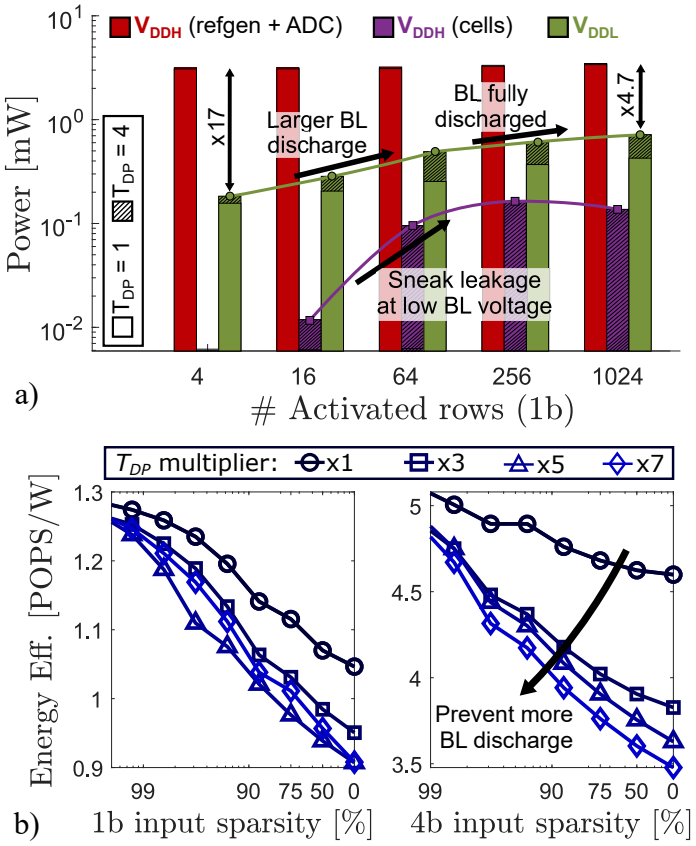


Fig. 17. Evolution of the macro's a) power breakdown with more rows activated, for low and medium DP durations, b) energy efficiency for different levels of input sparsity with 1b and 4b inputs, at $\gamma = 32$, nominal supply and 64MHz. A 0% sparsity corresponds to 1024 activated rows.

b) stagnates once maximum BL(B) discharge is reached, given that the outputs of the MB-DTSE and ABN blocks remain mostly unchanged. On the contrary, increasing $\gamma$ leads to a larger discharge of the ABN node, continuously yielding additional ADC toggles that directly affect the energy/op metric. Altogether, the time-dependent current-based operation offers a direct trade-off between operating frequency, energy/op and dynamic range scaling.

Moreover, at nominal voltages and fixed timing parameters, the computing efficiency of the CIM-SRAM macro strongly varies with the actual hardware utilization per channel configuration. Besides, it also depends upon the sparsity of the input data vector. Ignoring any I/O transfer latency at this stage, the 4b-normalized area and energy efficiency of the macro are respectively given by

$$\text{AE} = \frac{2 \times (K \times K \times C_{in}) \times C_{out} \times f_{clk}}{4/r_{in} \times 4/r_w \times \text{macro area}} \tag{7}$$

and

$$\text{EE} = \frac{2 \times (K \times K \times C_{in}) \times C_{out} \times f_{clk}}{4/r_{in} \times 4/r_w \times P_{tot}}. \tag{8}$$

$f_{clk}$ and $P_{tot}$ are respectively the clock frequency and average CIM-SRAM power consumption. While area efficiency scales linearly with the number of channels in Eq. (7), the static power overheads of the macro make Eq. (8) heavily non-linear. Fig. 17-a) demonstrates this by measuring the CIM-SRAM's power with an increasing number of activated rows (i.e., non-zero input channels) and different DP duration for 1b inputs. Breaking the power down between supplies, the $V_{DDH}$ contribution vastly dominates when few rows are used, due to the fixed current drawn by the resistive ladders and control drivers, sized for the $1152 \times 512$ array. However, the $V_{DDL}$ contribution rises quickly as more rows become activated, toggling more WLs and increasingly discharging the BL(B)s as a result. The effect is larger with a longer DP duration, given that the BL(B) discharge at fixed input dimension increases. The $V_{DDL}$ surge slows down above 64 active rows in parallel, as BL(B) are fully discharged and do therefore not impact the total power anymore. Still, $V_{DDL}$ only amounts to 18% of the total power at maximum array utilization. Another interesting point is the rising contribution from the separated bitcell supply, also set to $V_{DDH}$ as described in Section III. This relates to the sneak leakages on BL(B) that lead to low DC currents between bitcells, namely as the BL(B) voltage approaches 0V. This last point explains why the effect is only significant for a long DP duration.

Eventually, Fig. 17-b) highlights the energy efficiency improvement with an increasingly sparse input ratio, at 4b. The ABN gain $\gamma$ is set to its median value (32), while different DP durations are tested. Once again, the improvement is more significant for a long DP duration thanks to increased savings on the BL(B) precharge energy. In the end, the 4b CIM-

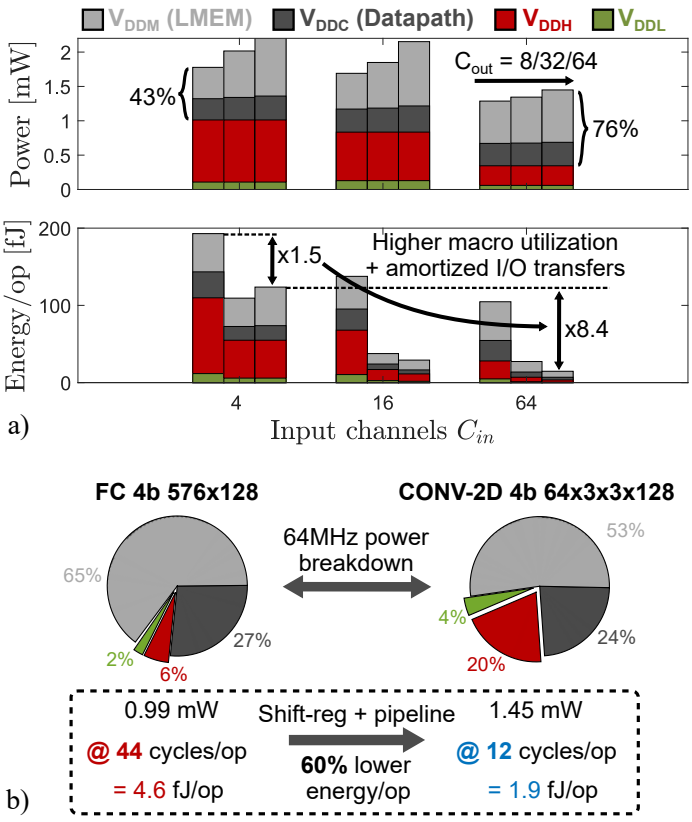


Fig. 18. Power breakdown of the full CIM-CNN accelerator at 4b and 64MHz. a) Increasing the number of channels reduces the CIM-SRAM's relative contribution to the total energy/op. b) At fixed I/O size, convolution mode achieves a 60% lower energy/op than its fully-connected counterpart.

SRAM's peak power efficiency varies from 4.2 to 3.6POPS/W from fast to slow, assuming a 50% input sparsity.

### B. CIM-CNN Accelerator Characterization

Although the CIM-SRAM macro showcases an excellent computing efficiency when considered alone, the total efficiency of the IMPACT accelerator will be hindered by the pre-/post-processing steps, I/O transfers and regular LMEM accesses. Hence, for convolutional layers, the overall 4b energy efficiency of the accelerator becomes

$$\mathrm{EE}_{tot} = \frac{2 \times (K \times K \times C_{in}) \times C_{out}}{P_{macro} + P_{datapath} + P_{LMEM}} \times \frac{f_{clk}}{N_{cycles}}, \quad (9)$$

where $P_{macro}$, $P_{datapath}$ and $P_{LMEM}$ are respectively the average power of the macro, datapath and LMEMs, and $N_{cycles}$ is the critical number of cycles per configuration, derived in Section IV.B. For lower computing precisions, Eq. (9) scales similarly to Eq. (8). The accelerator's measured power breakdown is shown in Fig. 18-a) for different input and output channel configurations, at 4b and 64MHz. The CIM-SRAM timing parameters ($T_{DP}$, $\gamma$) are set to (3, 16) and nominal supplies are used ($V_{DDH/M/C}$ = 0.8V, $V_{DDL}$ = 0.4V). The breakdown reveals that the sum of datapath and LMEM power fractions grows from 43% with few channels to 76% at maximum size. This mainly relates to the higher hardware utilization of the macro, amortizing its energy/op

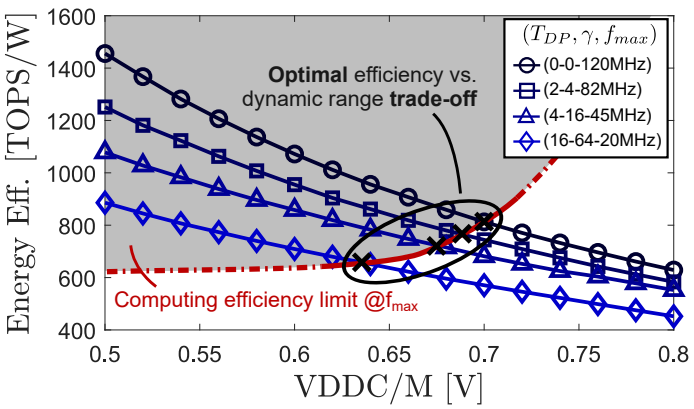


Fig. 19. Maximum energy efficiency point of the CIM-CNN accelerator under various CIM-SRAM configurations and voltage scaling of the digital part, at 4b and 50% input sparsity.

(i.e., 1/EE) by one to two order(s) of magnitude. Meanwhile, the total energy associated with data transfers scales linearly with the number of LMEM accesses, providing no benefits at first glance. However, scaling $C_{out}$ at fixed $C_{in}$ does not necessarily increase the number of cycles per operation, thanks to pipelining. Consequently, the energy fraction associated with data transfers is configuration-dependent, as seen for instance with $C_{in} = 64$ and $C_{out}$ changing from 8 to 32.

Moreover, Fig. 18-b) compares the power breakdown between fully-connected (FC) and convolutional modes for a constant number of activated inputs. While they avoid the need to perform the *im2col* transform, FC operations on a single image cannot take advantage of data reuse nor pipelined operations, amounting to more data transfers between the macro and LMEMs compared to the convolutional mode. As a result, the power contribution of the CIM-SRAM macro drops below 10% while that of LMEM accesses take up to 65%. The eventual result is a 60% lower energy/op compared to the convolutional case at 64MHz.

Previous results suggest that I/O transfers remain the major bottleneck to enable the full efficiency of the CIM-SRAM macro. Yet, the maximum frequency of the datapath logic and SRAM at their nominal 0.8V supply is generally higher than that of CIM-SRAM accelerator. As such, two ways exist to improve the energy efficiency of the CIM-CNN accelerator: either (i) by working at the datapath's maximum frequency and allowing multiple cycles to perform the CIM-SRAM operation, or (ii) by reducing the LMEM and datapath's voltage to fit the maximum frequency imposed by the CIM-SRAM macro. The first solution is enticing to improve the overall throughput when the CIM-SRAM is configured in a low-speed setup. However, it requires to handle stalls from the macro and has little impact on the energy/op, and was therefore not implemented in this work. Instead, we measured the improvement in energy efficiency brought by the second solution in Fig. 19 for different critical configurations of the CIM-SRAM macro, based on Fig. 16-a). It highlights the energy efficiency boundary of the accelerator, ranging between 650 and 813TOPS/W. These points are the trade-off optima between computing efficiency and dynamic range allocation of the IMPACT accelerator.

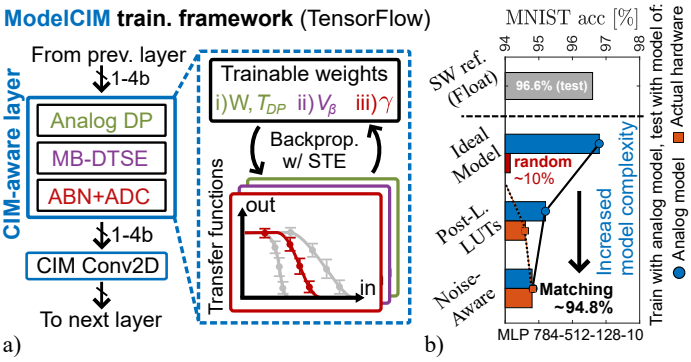


Fig. 20. ModelCIM framework for CIM-aware CNN training, implemented with the TensorFlow/Keras back-end [30]. a) Representation of a CIM-aware layer, with trainable weights and transfer function models. b) Improvement of MNIST accuracy matching with increasing model complexity.

### C. CIM-Aware Training

In order to map an actual CNN on the IMPACT accelerator, a dedicated CNN training is necessary to account for the high level of non-linearity and variability faced by the CIM-SRAM macro. Therefore, we developed the custom CIM-aware ModelCIM framework shown in Fig. 20-a) to include these analog non-idealities during CNN training. The ModelCIM framework replaces each subsequent operation by a model of its analog equivalent, which applies here to the DP, MB-DTSE and ABN blocks. Actual transfer functions, including deterministic trends such as non-linearities and layout asymmetries, are modeled by means of accurate post-layout look-up tables. Meanwhile, stochastic deviations are accounted for by adding noise to each block, following its statistical distribution of outputs due to mismatch and intrinsic noise. Finally, quantization-aware training relying on straight-through estimators (STEs [29]) and adapted PWL drivers/ADC models accounts for the target bit-precision. The test accuracy obtained on MNIST by training a 784-512-128-64-10 MLP with the ModelCIM framework is shown in Fig. 20-b). It comes to little surprise that the matching in accuracy between successive models and the actual hardware representation improves with the complexity of the model, with best results obtained for a noise-aware training with post-layout look-up tables. Still, the framework is unable to fully compensate for the accuracy loss on the MLP. Indeed, a 2% residual degradation remains due to the accumulation of non-linear behaviors in the CIM-SRAM.

### D. State-of-the-Art Comparison

Finally, IMPACT and its CIM-SRAM macro are compared to various works across the state-of-the-art in Table I. The macro on its own achieves a 595.1kB/mm$^2$ density thanks to its dense array of 6T bitcells from the foundry and minimized periphery overhead. Moreover, it showcases excellent peak area and energy efficiencies of 146TOPS/mm$^2$ and 4.2POPS/W, normalized to 4b computing, while providing the multi-bit ABN feature. Although these metrics scale with the macro's timing configuration, they still exceed that of any previous designs across all CIM types. This improvement in terms of raw performance is the combination of using (i) dense 6T current-based DP operators, involving the switching of small capacitances compared to charge-domain operations [12], [35], while avoiding multiple sequential BL(B) switching as in high-precision current-domain designs [36], (ii) relaxed ADC precision thanks to the multi-bit ABN implementation, yielding to a better amortization of the ADC conversion energy, (iii) an advanced technology node. However, these improvements are traded for a reduced maximum computing precision, bounded by the CIM-SRAM's high non-linearity and variability. The full IMPACT accelerator achieves a 1.5TOPS peak throughput and 813TOPS/W peak energy efficiency, assuming 50% input sparsity. In that regard, Jia et al. achieve a higher throughput thanks to a multi-core 8b CIM accelerator [31], thereby also extending its reconfigurability to various CNN tasks. The CIM sub-system of the DIANA chip [32] also achieves enhanced throughput by implementing the clock-boosted digital datapath technique described in Section V.B. However, they require extensive LMEM capacity to store the flattened input image in order pre-process the *im2col* transform. Still, IMPACT outperforms both works in terms of computing efficiency at the cost of reduced inference accuracy, despite using the post-silicon knowledge to fine-tune training parameters such as the chip corner and the levels of injected noise. The resulting impact on accuracy remains below 2% for the low-complexity MNIST dataset, but might worsen for more complex datasets that are increasingly sensitive to linearity degradation. Relying on chip-wise training with the full hardware in the loop could provide a solution to further improve the resilience to such accuracy loss, although limited by the inherent non-idealities of the proposed architecture. This paves the way for future works detailing the robustness of hardware-aware training to circuit-level non-idealities depending on the use case. Once again, these points underline the trade-off between computing SNR (and thus, inference accuracy) and efficiency presented back in Section I, with the proposed work purposefully attempting to push back the design boundary of the latter.

## VI. Conclusion

In this work, we presented IMPACT, a 1-to-4b CIM-CNN accelerator in 22nm FD-SOI that exploits hardware-software co-design to maximize energy efficiency while preserving computational accuracy. To that end, we simultaneously introduce a novel low-precision 6T-based analog CIM-SRAM macro and a custom CIM-aware CNN training framework to model hardware impairments. The 1152×512 macro features multi-bit analog batch-normalization (ABN) to overcome the ADC precision bottleneck in low-precision CIM-based CNNs, as well as an ultra-low power datapath enabled by a dual-supply strategy. The IMPACT accelerator embeds the macro within a highly-parallel and reconfigurable digital datapath, providing efficient pipelined I/O data transfers with *im2col* input pre-processing capability. Measurements results showcase a peak accelerator energy efficiency of 813TOPS/W at 64MHz with 4b inputs and weights. Considered alone, the CIM-SRAM macro achieves peak 4b energy and area efficiencies of 4.2POPS/W and 146TOPS/mm$^2$, surpassing all previous low-precision designs to date at the cost of 1.5%

TABLE I
STATE-OF-THE-ART COMPARISON

| | Digital CIM | | Charge-based Analog CIM | | | | Current-based Analog CIM | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | [13] | [14] | [33] | [34] | [35] | [9] | [36] | [7] | [6] | [37] | [32] | **This work** |
| **Year** | 2022 | 2022 | 2021 | 2021 | 2021 | 2022 | 2021 | 2021 | 2021 | 2022 | 2022 | 2022 |
| **Technology** | 5nm FinFET | 28nm CMOS | 16nm FinFET | 28nm CMOS | 28nm CMOS | 22nm CMOS | 28nm CMOS | 28nm CMOS | 7nm FinFET | 65nm CMOS | 22nm FD-SOI | 22nm FD-SOI |
| **Bitcell type** | 12T | 8T | 8T1C | 10T1C | 8T1C | 9T+C | 6T+LC | 9T | 8T | 8T | 12T | 6T |
| **On-chip CIM size** | 8kB | 2kB | 576kB | 424kB | 36kB | 2×16kB | 48kB | 2kB | 16×0.5kB | 2kB | 72kB | 72kB |
| **Density [kB/mm$^2$]** | 601 | 40.8 | 23 | 21.8 | 70.6 | 131 | - | 109.2 | 156.3 | 3.4 | 31.4 | **595.1** |
| **Supply voltage [V]** | 0.5-0.9 | 0.5-0.9 | 0.8 | 1 | 0.6 | 0.7-1 | 0.85 | 0.6 | 0.8-1 | 0.9 | 0.6-0.9 | 0.4/0.8 |
| **Max precision (in/w/out)** | 4/4/14b | 4/1/4b | 8/8/8b | 2/1/1b | 5/1/8b | 8/8/8b | 8/8/20b | 1/1/7b | 4/4/4b | 8/8/8b | 7/1.5/6b | 4/4/4b |
| **CIM sub-system** | No | No | Yes | Yes | No | Yes | No | No | No | Yes | Yes | Yes |
| **Peak Throughput[1] [TOPS]** | 0.73-2.9[2] | 0.03-1.25[2] | 11.8 | 0.3 | 0.38[2] | 2.4-4[2] | 0.83[2] | 0.37-0.02[2] | 5.9-7.2[2] | 1[3] | 8.6-11.9 | 0.3-1.5[4]/2.9-17.6[2,4] |
| **Peak Macro EE[1] [TOPS/W]** | 254-42 | 62-30 | 207 | 36.8 | 363 | 129-62 | 91 | 91-3 (1-4b) | 611-436 | 79.4[3] | - | **2600-4200**[4] |
| **Peak Accel. EE[1] [TOPS/W]** | - | - | 121 | 27.3 | - | - | - | - | - | 17.9[3] | 196-116 | **650-813**[4] |
| **Peak AE[1] [TOPS/mm$^2$]** | 55-221[2] | 0.61-25[2] | 2.67 | 0.01 | 0.75 | 9.6-16[2] | - | 6.8-0.4 (1-4b)[2] | 115-141[2] | 0.17[3] | 5.8-8.3 | **1.1-6.4/24-146**[2,4] |
| **MNIST acc. [%]** | - | - | - | - | - | - | - | - | 99.6 | 99.3 | - | 98.1[5] |
| **ABN unit** | No | No | No | No | No | No | No | No | No | No | No | Multi-bit |

[1] Normalized to 4b operations [2] Excluding I/O transfer cycles [3] Average for MNIST execution [4] Changes w/ timing parameters [5] Simulated w/ post-silicon charac. on adapted LeNet-5
Blue (resp. red) is used to highlight strong (resp. weak) performance across designs.

accuracy degradation on the MNIST dataset. These results demonstrate that analog current-based CIM-SRAMs can be enabled for low-precision CNN applications as long as careful co-design is carried out. Nevertheless, accuracy degradation is likely to further ramp up for more complex datasets, setting an application-dependent limit to the use of such high-performance current-domain CIM-SRAM macros. Finally, the $\times 5$ gap between the accelerator- and macro-level computing efficiencies underlines that data transfers remain a system-level bottleneck, requiring extensive future research effort to fully enable low-precision CIM-SRAM hardware.

## ACKNOWLEDGMENT

The authors would like to thank ECS team members for their help with the ICare chip design and proofreading of this work, in particular Dr. R. Dekimpe, Dr. R. Saeidi and M. Gonzalez.

## REFERENCES

[1] S. Murshed *et al.*, "Machine Learning at the Network Edge: A Survey," *ACM Computing Surveys*, vol. 54, no. 8, pp. 1–37, 2021.
[2] B. Silva *et al.*, "Mobile-health: A review of current state in 2015," *Elsevier Journal of Biomedical Informatics*, vol. 56, no. 8, pp. 265–272, 2015.
[3] R. S. Williams, "What's Next? [The end of Moore's law]," *IEEE Computing in Sciences & Engineering*, vol. 19, no. 2, pp. 7–13, 2017.
[4] M. Horowitz, "Computing's energy problem (and what we can do about it)," in *IEEE ISSCC*, 2014, pp. 10–14.
[5] J. Zhang *et al.*, "A Machine-Learning Classifier Implemented in a Standard 6T SRAM Array," in *IEEE VLSI*, 2016, pp. 1–2.
[6] M. Sinangil *et al.*, "A 7-nm Compute-in-Memory SRAM Macro Supporting Multi-Bit Input, Weight and Output and Achieving 351 TOPS/W and 372.4 GOPS," *IEEE JSSC*, vol. 56, no. 1, pp. 188–198, 2021.
[7] V. Rajanna *et al.*, "SRAM with In-Memory Inference and 90Activity Reduction for Always-On Sensing with 109 TOPS/mm2 and 749-1,459 TOPS/W in 28nm," in *IEEE ESSCIRC*, 2021, pp. 127–130.
[8] H. Valavi *et al.*, "A 64-Tile 2.4-Mb In-Memory-Computing CNN Accelerator Employing Charge-Domain Compute," *IEEE JSSC*, vol. 54, no. 6, pp. 1789–1799, 2019.
[9] H. Wang *et al.*, "A 32.2 TOPS/W SRAM Compute-in-Memory Macro Employing a Linear 8-bit C-2C Ladder for Charge Domain Computation in 22nm for Edge Inference," in *IEEE VLSI*, 2022, pp. 36–37.
[10] J. Song *et al.*, "TD-SRAM: Time-Domain-Based In-Memory Computing Macro for Binary Neural Networks," *IEEE TCAS-I*, vol. 68, no. 8, pp. 3377 – 3387, 2022.
[11] P.-C. Wu *et al.*, "A 28nm 1Mb Time-Domain Computing-in-Memory 6T-SRAM Macro with a 6.6ns Latency, 1241GOPS and 37.01TOPS/W for 8b-MAC Operations for Edge-AI Devices," in *IEEE ISSCC*, 2022, pp. 190–192.
[12] H. Jia *et al.*, "A Programmable Heterogeneous Microprocessor Based on Bit-Scalable In-Memory Computing," *IEEE JSSC*, vol. 55, no. 9, pp. 2609–2621, 2020.
[13] H. Fujiwara *et al.*, "A 5-nm 254-TOPS/W 221-TOPS/mm$^2$ Fully-Digital Computing in-Memory Macro Supporting Wide-Range Dynamic-Voltage-Frequency Scaling and Simultaneous MAC and Write Operations," in *IEEE ISSCC*, 2022, pp. 186–188.
[14] D. Wang *et al.*, "DIMC: 2219TOPS/W 2569F$^2$/b Digital In-Memory Computing Macro in 28nm Based on Approximate Arithmetic Hardware," in *IEEE ISSCC*, 2022, pp. 266–268.
[15] N. Verma *et al.*, "In-Memory Computing: Advances and Prospects," *IEEE SSCM*, vol. 11, no. 3, pp. 43–55, 2019.
[16] A. Kneip *et al.*, "Impact of Analog Non-Idealities on the Design Space of 6T-SRAM Current-Domain Dot-Product Operators for In-Memory Computing," *IEEE TCAS-I*, vol. 68, no. 5, pp. 1931–1944, 2021.
[17] P. Houshmand and A. Raychowdhury, "A Practical Design-Space Analysis of Compute-in-Memory With SRAM," *IEEE TCAS-I*, vol. 69, no. 4, pp. 1466 – 1479, 2022.
[18] V. Joshi, M. Le Gallo, S. Haefeli *et al.*, "Accurate deep neural network inference using computational phase-change memory," *Nat Comm*, vol. 11, no. 2473, pp. 1–13, 2020.
[19] C. Zhou *et al.*, "Noisy Machines: Understanding Noisy Neural Networks and Enhancing Robustness to Analog Hardware Errors Using Distillation," *arXiv preprint arXiv:2001.04974v1*, pp. 1–14, 2020.
[20] M.-S. Le *et al.*, "PR-CIM: a Variation-Aware Binary-Neural-Network Framework for Process-Resilient Computation-in-memory," *arXiv preprint arXiv:2110.09962v1*, pp. 1–8, 2021.
[21] M. Kang *et al.*, "A Multi-Functional In-Memory Inference Processor Using a Standard 6T SRAM Array," *IEEE JSSC*, vol. 53, no. 2, pp. 642–655, 2018.
[22] X. Peng *et al.*, "DNN+NeuroSim V2.0: An End-to-End Benchmarking Framework for Compute-in-Memory Accelerators for On-Chip Training," *IEEE TCAD*, vol. 40, no. 11, pp. 2306–2319, 2021.
[23] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, pp. 1–11, 2015.
[24] B. Moons *et al.*, "Minimum Energy Quantized Neural Networks," in *IEEE Asilomar*, 2017, pp. 1921–1925.
[25] R. Dekimpe and D. Bol, "Mixed-Signal Compensation of Tripolar Cuff Electrode Imbalance in a Low-Noise ENG Analog Front-End," in *IEEE ESSCIRC*, 2022, pp. 445–448.

[26] M. Lefebvre *et al.*, "A 0.2-to-3.6TOPS/W Programmable Convolutional Imager SoC with In-Sensor Current-Domain Ternary-Weighted MAC Operations for Feature Extraction and Region-of-Interest Detection," in *IEEE ISSCC*, 2021, pp. 118–120.
[27] Y. Chae *et al.*, "A 6.3 W 20 bit Incremental Zoom-ADC with 6 ppm INL and 1 V Offset," *IEEE JSSC*, vol. 48, no. 12, pp. 3019–3027, 2013.
[28] K. Chellapilla *et al.*, "High performance convolutional neural networks for document processing," in *Tenth international workshop on frontiers in handwriting recognition*. Suvisoft, 2006, pp. 1–7.
[29] Y. Bengio *et al.*, "Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation," *arXiv preprint arXiv:1308.3432*, pp. 1–12, 2013.
[30] M. Abadi *et al.*, "TensorFlow: a system for Large-Scale machine learning," in *USENIX OSDI*, 2016, pp. 265–283.
[31] H. Jia *et al.*, "Scalable and Programmable Neural Network Inference Accelerator Based on In-Memory Computing," *IEEE JSSC*, vol. 57, no. 1, pp. 198–211, 2022.
[32] P. Houshmand *et al.*, "DIANA: An End-to-End Hybrid DIgital and ANAlog Neural Network SoC for the Edge," *IEEE JSSC (Early Access)*, pp. 1–13, 2022.
[33] J. Lee *et al.*, "A Programmable Neural-Network Inference Accelerator Based on Scalable In-Memory Computing," in *IEEE ISSCC*, 2021, pp. 236–238.
[34] S. Yin *et al.*, "PIMCA: A 3.4-Mb Programmable In-Memory Computing Accelerator in 28nm for On-Chip DNN Inference ," in *IEEE VLSI*, 2021, pp. 1–2.
[35] J. Lee *et al.*, "Fully Row/Column-Parallel In-memory Computing SRAM Macro employing Capacitor-based Mixed-signal Computation with 5-b Inputs," in *IEEE VLSI*, 2021, pp. 1–2.
[36] J.-W. Su *et al.*, "A 28nm 384kb 6T-SRAM Computation-in-Memory Macro with 8b Precision for AI Edge Chips," in *IEEE ISSCC*, 2021, pp. 250–251.
[37] J. Yue *et al.*, "STICKER-IM: A 65 nm Computing-in-Memory NN Processor Using Block-Wise Sparsity Optimization and Inter/Intra-Macro Data Reuse," *IEEE JSSC (Early Access)*, pp. 1–14, 2022.

**Adrian Kneip** (Graduate Student Member, IEEE) received the M.Sc. degree (summa cum laude) in Electrical Engineering from the Université catholique de Louvain (UCLouvain), Louvain-la-Neuve, Belgium, in 2019. He is currently working there towards the PhD degree with the Electronics Circuits and Systems group, under the supervision of Prof. D. Bol. His research interests include the modelling and design of mixed-signal ultra-low-power systems and machine-learning hardware, with special dedication to SRAM memories and analog/mixed-signal in-memory computing. He serves as a reviewer for different IEEE conferences and journals, including IEEE TCAS-I/-II, ISCAS and APCAS. Since 2020, he also serves as IEEE Student Branch Representative for the Benelux Section.

**Martin Lefebvre** (Graduate Student Member, IEEE) received the M.Sc. degree (summa cum laude) in Electromechanical Engineering from the Université catholique de Louvain (UCLouvain), Louvain-la-Neuve, Belgium, in 2017, where he is currently pursuing the Ph.D. degree, under the supervision of Prof. D. Bol. His current research interests include hardware-aware machine learning algorithms, low-power mixed-signal vision chips for embedded image processing, and ultra-low-power current reference architectures. He serves as a reviewer for various conferences and journals, including IEEE Trans. on Biomed. Circuits and Syst., IEEE Trans. on VLSI Syst., Int. Symp. on Circuits and Syst. and Asia Pacific Conf. on Circuits and Syst.

**Julien Verecken** received the M.Sc. degree (summa cum laude) in Electrical Engineering from the Université catholique de Louvain (UCLouvain), Louvain-la-Neuve, Belgium, in 2020, where he was conducting research until 2022 under the supervision of Prof. D. Bol. His research interests include embedded deep learning for computer vision and the energy-efficient design of application-specific instruction-set processors for ultra-low-power microcontrollers for capsule endoscopy. He is now working in the field of image processing and compression as an FPGA/ASIC design engineer at intoPIX, Mont-Saint-Guibert, Belgium.

**David Bol** (Senior Member, IEEE) received the M.Sc. degree in Electromechanical Engineering and the Ph.D degree in Engineering Science from Université catholique de Louvain (UCLouvain), Louvain-la-Neuve, Belgium in 2004 and 2008, respectively. In 2005, he was a visiting Ph.D student at the CNM National Centre for Microelectronics, Sevilla, Spain, in advanced logic design. In 2009, he was a postdoctoral researcher at intoPIX, Louvain-la-Neuve, Belgium, in low-power design for JPEG2000 image processing. In 2010, he was a visiting postdoctoral researcher at the UC Berkeley Laboratory for Manufacturing and Sustainability, Berkeley, CA, in life-cycle assessment of the semiconductor environmental impact. He is now an Associate Professor at UCLouvain. In 2015, he participated to the creation of e-peas semiconductors, Louvain-la-Neuve, Belgium. Prof. Bol leads the Electronic Circuits and Systems (ECS) group focused on ultra-low-power design of integrated circuits for the IoT and biomedical applications including computing, power management, sensing and wireless communications. He is engaged in a social-ecological transition in the field of ICT research. He co-teaches four M.S. courses in Electrical Engineering at UCLouvain on digital, analog and mixed-signal integrated circuits and systems as well as sensors, with two B.S. courses including the course on Sustainable Development and Transition. Prof. Bol has authored or co-authored more than 150 technical papers and conference contributions and holds three delivered patents. He (co-)received three Best Paper/Poster/Design Awards in IEEE conferences (ICCD 2008, SOI Conf. 2008, FTFC 2014). He serves as a reviewer for various journals and conferences such as IEEE J. of Solid-State Circuits, IEEE Trans. on VLSI Syst., IEEE Trans. on Circuits and Syst. I/II. Since 2008, he presented several invited papers and keynote tutorials in international conferences including a forum presentation at IEEE ISSCC 2018. On the private side, he pioneered the parental leave for male professors in his faculty to spend time connecting to nature with his family.