A Blueprint for Accurate, Energy-Efficient DNN Inference via Capacitive In-Memory Processing

Kai-Uwe Demasius[1] Alexander Lowa[1] Boris Murmann[2]
[1]SEMRON, Dresden, Germany
[2]University of Hawaii at Manoa, Hawaii, USA

***Abstract*—This paper presents a hardware–software co-design methodology for accurate and energy-efficient DNN inference on capacitive processing-using-memory (PUM) systems. We investigate the fundamental algorithm–hardware interactions that limit analog accelerators, focusing on accumulator resolution and noise resilience. To eliminate impractical requirements for peripheral circuitry, we introduce a precision-aware training framework that minimizes the effective resolution of ADCs and enhances robustness to circuit-level non-idealities such as device variability and read-out noise. Leveraging bit-slicing and accumulator-aware optimizations, we show that ADC resolution can be reduced from 19 to 7 bits on ResNet18 without accuracy degradation. We further provide a system-level analysis linking neural network characteristics back to compute-cell parameters and ADC design constraints, establishing the operational limits of capacitive charge-domain inference. Finally, we present an energy model demonstrating how reduced ADC precision and improved charge-domain efficiency yield substantial system-level energy savings. Together, these results offer a blueprint for scalable, low-power analog DNN accelerators, pushing the boundaries of fully on-chip inference.**

I. Introduction

The computational workload of modern DNNs is dominated by matrix-vector multiplications (MVM), which arise in convolutional, fully connected, and attention layers [1]. State-of-the-art models involve billions of multiply-accumulate (MAC) operations across millions or even billions of parameters [2], creating a strong need for hardware that can deliver both high throughput and energy efficiency. To address this, memory-centric architectures such as processing-using-memory (PUM) and processing-near-memory (PNM) reduce data movement by embedding computation within or close to memory arrays, both of which fall under the broader category of processing-in-memory (PIM) [3, 4]. Among PUM approaches, charge-domain techniques stand out for their superior energy efficiency. For instance, SRAM-based macros using 6T/8T bitcells in 7nm nodes have demonstrated efficiencies as high as 5616 TOPS/W/b [5], or using 10T3C bitcells at 28 nm achieving 8161 TOPS/W/b [6] for common vision workloads like ResNet-18 [7] on ImageNet [8]. At the device level, emerging memcapacitive arrays like CapRAM [9] promise efficiencies exceeding 29600 TOPS/W/b, leveraging principles such as charge screening and adiabatic recovery. In contrast, resistive PUM macros such as ReRAM and PCM based ones typically lag behind, reaching only 336–1344 TOPS/W/b due to device variability and the high-precision ADC overheads required for reliable readout [10, 11]. Despite their efficiency, charge-domain PUM architectures face a critical bottleneck: limited on-chip memory density. Existing macros typically provide only kilobytes to a few megabytes of weight storage, orders of magnitude below the capacity required for modern DNNs with billions of parameters. Fetching weights from off-chip DRAM undermines the core energy advantage of PUM, incurring substantial access energy and latency penalties. The problem is especially severe in SRAM-based designs: storing 1 GB in 6T SRAM is estimated to require $\sim$400 mm^2 of die area and cost hundreds of dollars [12]. These limitations have motivated a shift toward denser memory technologies compatible with 3D integration. For example, CapRAM supports monolithic vertical stacking over logic, akin to Bit Cost Scalable (BiCS) NAND-Flash [13], offering high-density, low-energy analog MACs within a compact footprint suitable for edge deployment. Yet, high efficiency for the analog dot product alone does not suffice, the dominant energy bottleneck in PUM is often not the MAC itself, but rather the ADCs [14, 15, 16]. These converters must digitize high-resolution analog column sums, and their energy cost scales exponentially with resolution.

In this work, we derive design rules for scalable, precision-aware capacitive PUM systems suitable for DNN inference. Our key contributions are:

1) We derive theoretical limits on analog column sums and show how bit-slicing reduces ADC precision demands, albeit at the cost of additional compute cycles.
2) We extend the A2Q+ [17] algorithm to support slice-wise regimes, enabling tight ADC precision budgets without degrading inference accuracy.
3) We develop an analytical framework to quantify the effects of geometrical variation, programming imprecision, and readout noise on signal fidelity in capacitive charge-domain systems.
4) We demonstrate that capacitive PUM architectures can approach the fundamental noise limit more effectively than resistive counterparts, unlocking new opportunities for ultra-efficient on-chip inference.

Together, these contributions form a blueprint for next-generation PUM accelerators that can eventually support billion-parameter DNNs on-chip within tight edge energy budgets. Fig. 1 provides a high-level view of the capacitive accelerator and its digital periphery for accommodating slice-wise quantization (Sec. III-C).

II. Preliminaries

This chapter reviews the key concepts underlying our analysis. We first outline the computational demands of DNN inference, focusing on the central role of matrix-vector multiplications. Next, we discuss quantization techniques that reduce inference cost by encoding weights and activations in low-precision formats. Finally, we introduce charge-based PUM architectures that perform MVMs efficiently using capacitive memories.

A. Modern DNN Inference

At its core, inference in modern DNNs reduces to repeated matrix–vector multiplications. Convolutional, fully connected, and attention layers can all be expressed in this form [1], where a weight matrix multiplies an activation vector to produce the next-layer output, typically followed by a nonlinearity. The computational cost of inference is therefore proportional to the number of MAC operations, which scales with both the depth of the network and the dimensionality of its layers. State-of-the-art models contain millions to billions of parameters [2], leading to inference workloads dominated by MVMs. Therefore, efficient hardware execution is critical. Algorithmic techniques such as quantization reduce the arithmetic precision of weights and activations while preserving accuracy [18, 19, 20]. A widely adopted configuration uses 4-bit weights and 8-bit activations with high bit accumulations, achieving high accuracy across diverse tasks [19, 20, 21]. Weight quantization is generally more robust than activation quantization [22, 23], since weights are static and can be quantized offline with minimal accuracy loss [24, 25, 26]. In contrast, activations vary dynamically at runtime, making aggressive low-bit quantization substantially more challenging [18, 27, 28]. Such quantization strategies form the algorithmic foundation for capacitive PUM systems.

B. Quantizing Weights and Activations

Inference efficiency hinges on quantization of weights and activations. Configurations like W4A8 (Section II-A) highlight the precision–efficiency tradeoff. Next, we examine the strategies that realize these representations.

Different strategies introduce quantizers at distinct stages of the training–deployment pipeline. Post-training quantization (PTQ) [29, 30, 31] applies discretization to a frozen model without further optimization, but typically requires higher precision or extensive calibration to preserve accuracy [32, 33]. At the other extreme, fully quantized training (FQT) [34, 35, 36] enforces low precision across all tensors, including gradients, ensuring close alignment with hardware arithmetic, though at the cost of noisy weight updates. A compromise is quantization-aware training (QAT) [37, 38, 39], in which the forward pass is quantized during fine-tuning while the backward pass remains in full precision. For sub-8-bit arithmetic, QAT consistently outperforms PTQ and avoids the optimizer instabilities common in FQT [36, 40]. For these reasons, we adopt QAT as the quantization strategy in this work.

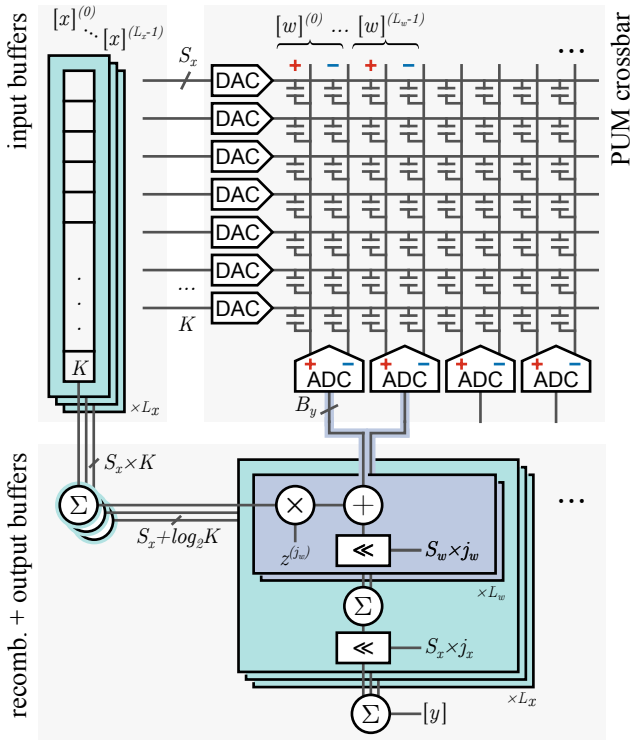

Fig. 1. Periphery from quantized input $[x]$ to quantized output $[y]$. Shown is the crossbar with differential weights, and the digital operations needed for slicing recombination. Note how a single network weight requires multiple bitlines due to differential weights (+ & -) and weight slicing (L_w)

Quantization during training is commonly simulated by a uniform affine mapping from high-precision to low-precision values. As shown in Equation (1), the operator is defined by a positive scale factor $\Delta \in \mathbb{R}_+$ and an integer zero-point $z \in \mathbb{Z}$. Inputs are rounded to the nearest integer using half-way rounding $\lfloor \cdot \rceil$, then clipped via $\mathrm{clip}(\cdot, n, p)$ to the representable range determined by the target bitwidth $B \in \mathbb{N}$. For signed formats we set $n = -2^{B-1}$, $p = 2^{B-1} - 1$; for unsigned formats $n = 0$, $p = 2^B - 1$. For a vector $\boldsymbol{u} \in \mathbb{R}^K$ we define

$$\mathcal{Q}_B(\boldsymbol{u}) := \mathrm{clip}\Big(\Big\lfloor \frac{\boldsymbol{u}}{\Delta} \Big\rceil + z,\ n,\ p\Big). \tag{1}$$

For brevity, the quantized vector is written as $[\boldsymbol{u}] := \mathcal{Q}_B(\boldsymbol{u}) \in \mathbb{Z}^K$, and its de-quantized counterpart is written as $\hat{\boldsymbol{u}} := \Delta\big([\boldsymbol{u}] - z\big)$, so that $\boldsymbol{u} \approx \hat{\boldsymbol{u}}$. Each quantized entry is bounded in magnitude. For any index $i \in \{1, \ldots, K\}$, the output of (1) satisfies

$$|[\boldsymbol{u}]_i| \ \le\ 2^{B-\delta_u} - \bar{\delta}_u.$$

Here, the indicator $\delta_{(\cdot)} = 1$ for signed representations and 0 otherwise, while its complement $\bar{\delta}_{(\cdot)} = 1 - \delta_{(\cdot)}$ applies in the unsigned case. During QAT, the forward pass substitutes each floating-point tensor with its de-quantized version. Gradients are evaluated with the straight-through estimator (STE) [41], enabling simultaneous learning of network weights and quantization parameters [39, 42, 43].

C. Charge Domain PUM using Capacitive Devices

The general concept of charge domain PUM [6, 44, 45, 46] builds up on the charge-voltage relationship of a capacitor $Q = C \times V_{in}$, where the inputs of the MVM are encoded as the voltage V_{in} and the weights are stored as a capacitance C. The charge Q is accumulated on a bitline over many capacitive cells and delivers the final result of the MVM according to $\sum Q_i$. This charge is converted into a voltage and then converted back to the digital domain, by using an analog-to-digital converter.

The implementation of the variable capacitance can differ depending on the concept. In SRAM based accelerators, the capacitor is realized by MIM or MOM capacitors in the back-end-of-line (BEOL), with a 6T or 8T SRAM cell storing a 1 bit value of the weight [6]. There are several implementations of memcapacitive systems, where a variable capacitance can be implemented with an integrated storage capability, e.g., ferroelectric based [47], MOS capacitor based [48, 49], or charge screening based memcapacitors [9]. Charge screening based memcapacitors (CapRAM) have the advantage of obtaining a large on/off ratio compared to other implementations (in single crystalline silicon up to 60). They are based on three layers: an input gate, a shielding layer, and a read-out electrode, which is connected to the bitline.

To control the capacitive coupling from the input electrode to the read-out electrode, a memory layer is placed between the input gate and the shielding layer, which may be based on either a ferroelectric or a charge-trapping mechanism. Such a memcapacitive implementation not only enables integrated storage capability, but also offers the advantage of higher memory density compared to SRAM-based designs by supporting cost-effective 3D monolithic scaling, similar to 3D NAND flash [13]. For the following analysis, we assume a general memcapacitive device with a maximum capacitance of $C_{max} = 1\,\mathrm{fF}$ and an area of $1\,\mu\mathrm{m}^2$, which is capable of storing 2^{B_w} states, corresponding to a B_w bit weight value.

Each memcapacitive device has a certain ON/OFF ratio (ratio between the maximum capacitance $C_{\max}$ and the minimum capacitance $C_{\min}$). An analysis of how this affects the overall readout performance is delivered later.

D. Column Sum Resolution vs ADC Energy

The power efficiency of an ADC [50, 51, 52] is typically quantified by the Walden figure of merit (FOM) or the Schreier FOM, where the former is an empirical measure for circuits that are not fully limited by thermal noise (i.e., ADCs with low to moderate bitwidths). The Walden FOM is defined as follows, where B_y is the ADC's bitwidth:

$$FOM_W = \frac{E_{conv}}{2^{B_y}} \qquad [FOM_W] = \frac{\mathrm{J}}{\text{conv.-step}} \tag{2}$$

Therefore, the conversion energy scales with 2^{B_y}. During the past 20 years, the Walden FOM for ADCs has improved by a factor of 7500 [53], partly due to technology scaling and partly due to design improvements. In the early 2000s, when the technology node was around 350 nm, a typical value for the Walden FOM was in the range of $1\,\mathrm{pJ/conv.\text{–}step}$; in 2019 numbers of $0.4\,\mathrm{fJ/conv.\text{–}step}$ were obtained [53]. According to the thermal limit of the Schreier FOM, an increase of the total energy per conversion proportional to 4^{B_y} is required [51], and it becomes the dominant energy contribution for higher bitwidth ($> 10\,\mathrm{bit}$). A detailed analysis on the thermal energy limit for a capacitive PUM system follows in Sec. IV-B.

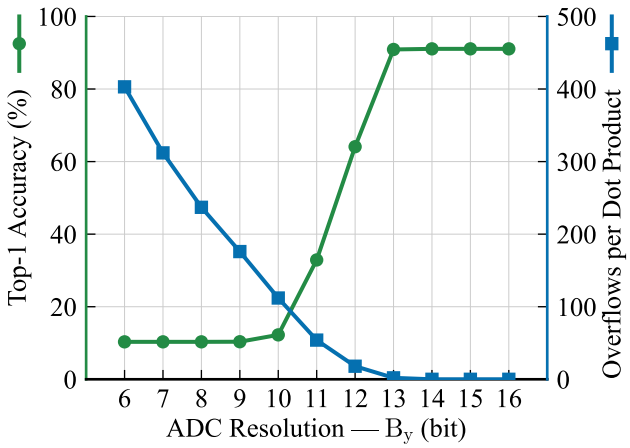

Fig. 2. Top-1 classification accuracy and average number of accumulator overflows per dot product as a function of ADC resolution B_y. Results are shown for a single-layer quantized neural network trained on MNIST [54] with 8-bit weights and binary inputs. At low ADC resolutions, frequent overflow severely degrades accuracy, while increasing B_y beyond 12 bits eliminates overflows and restores full accuracy.

III. Reducing ADC Resolution Requirements

To ensure reliable inference on PUM accelerators, the ADC resolution must balance two competing requirements: it must be wide enough to avoid overflow, yet narrow enough to minimize energy consumption. In this chapter, we derive tight analytical bounds on the crossbar column sum that characterize this trade-off and thereby determine the minimum ADC resolution under the full-precision guarantee (FPG). The FPG requires that every valid dot-product during inference maps to a unique digital code, ensuring functional correctness without overflow errors [16, 55]. While overflow is negligible with high-precision accumulators ($\geq$ 16 bits), reducing the accumulator bitwidth causes overflow probability to grow exponentially, leading to severe accuracy degradation [17, 56]. This effect is illustrated in Fig. 2, which shows results for a quantized single-layer DNN on MNIST [54].

Several works have attempted to circumvent the FPG requirement, but often under unrealistic assumptions. A natural strategy is to rescale the ADC range to match the input quantizer of the subsequent layer [55]. Conceptually, this aligns the ADC output with the network activations. However, practical deployment would require scaling not only per crossbar but also per output column to accommodate channel-wise weight quantization [19, 21, 39]. Consequently, each bitline could demand vastly different ADC ranges, from as little as 0.02% of the dynamic range up to nearly the full scale, rendering fine-grained per-column scaling infeasible in hardware.

A more practical alternative is to align the theoretical maximum accumulator value with the chosen ADC resolution, thereby avoiding impractical hardware specialization. To this end, we first derive conservative upper bounds on column sums directly from operand bitwidths. We then tighten these bounds via bit-slicing, partitioning the computation into lower-resolution segments that better exploit ADC precision. Finally, we reformulate the bounds in terms of slice-wise weight contributions, which allows optimization methods such as QAT to guarantee that, for a fixed ADC resolution, overflow is provably avoided during inference.

For concreteness, we quantify these bounds under the quantization setting introduced in Sec. II-A, namely W4A8: 4-bit signed weights and 8-bit unsigned activations. We assume a column length of $K = 128$, consistent with widely adopted PUM configurations [14, 57, 58].

A. Initial Bound from Operand Data Types

A conservative upper bound on the column-wise accumulation in analog crossbars can be directly inferred from the representational limits of the quantized operands.

Let $K \in \mathbb{N}$ denote the crossbar column length. Let $[\boldsymbol{w}], [\boldsymbol{x}] \in \mathbb{Z}^K$ be the quantized weights and activations, represented with bitwidths B_w and B_x, respectively, as described in Sec. II-B. One column of the PUM crossbar computes a dot product

$$[y] = \big\langle [\boldsymbol{x}], [\boldsymbol{w}] \big\rangle = \sum_{i=1}^{K} [x]_i \, [w]_i. \tag{3}$$

which must lie within the representable range of the column ADC. To satisfy the FPG, the result must obey the constraint:

$$-2^{B_y-1} \le [y] \le 2^{B_y-1} - 1, \tag{4}$$

where B_y is the effective resolution (in bit) of the ADC output. Since input values can vary arbitrarily at inference time, we bound the worst-case accumulation magnitude using the triangle inequality:

$$|y| = |\sum_{i=1}^{K} [x]_i [w]_i| \le \sum_{i=1}^{K} |[x]_i| \, |[w]_i| \tag{5}$$

$$\le K \cdot (2^{B_x-\delta_x} - \bar{\delta}_x) \cdot (2^{B_w-\delta_w} - \bar{\delta}_w) \tag{6}$$

$$\le 2^{B_y-1} - 1. \tag{7}$$

To streamline subsequent derivations, we define the maximum value per product as $G_p = (2^{B_x-\delta_x} - \bar{\delta}_x) \cdot (2^{B_w-\delta_w} - \bar{\delta}_w)$. Solving for the minimum required resolution B_y, we obtain:

$$B_y \ \ge \ \big\lceil 1 + \log_2\big(K \cdot G_p + 1\big) \big\rceil \tag{8}$$

Here, $\lceil \cdot \rceil$ denotes ceiling rounding. In the W4A8 case ($K = 128$, $B_w = 4$, $B_x = 8$), this yields $B_y = 19$, which far exceeds typical PUM ADC resolutions (≤ 10 bit) due to energy-area scaling [14, 15, 16], as elaborated in Sec. II-D.

$y = x \times w$ $\quad x = x^{(1)} \times 2^1 + x^{(0)}$ $\quad w = w^{(1)} \times 2^1 + w^{(0)}$

Slicing Scheme		Operations cycle 1	Operations cycle 2	Bits per MAC	Conversions per Row
Unsliced		$x \times w$		4	1
Input only		$x^{(1)} \times w$	$x^{(0)} \times w$	2	2
Weight only	col 1	$x \times w^{(1)}$		2	2
	col 2	$x \times w^{(0)}$			
Both	col 1	$x^{(1)} \times w^{(1)}$	$x^{(1)} \times w^{(1)}$	1	4
	col 2	$x^{(1)} \times w^{(0)}$	$x^{(1)} \times w^{(0)}$		

Fig. 3. Bit-slicing trade-offs for a 2 bit × 2 bit dot product. Operands are split into high $(x^{(1)}, w^{(1)})$ and low $(x^{(0)}, w^{(0)})$ bits. Each column shows the products evaluated in one crossbar column and one cycle. Fewer bits per slice relax the ADC resolution but increase cycles and conversions.

B. Improving Bound via Bit-Slicing

As 19 bit resolution is impractical in our setting, we aim to reduce the column-sum resolution without sacrificing exactness. To achieve this, we decompose both inputs and weights into lower-bitwidth components [58, 59, 60]. Specifically, let the original bitwidths of activations and weights be B_x and B_w. Choose divisors $S_x \mid B_x$ and $S_w \mid B_w$ where S_x and S_w are the slice widths of activations and weights respectively and set the slice counts $L_x = B_x/S_x$ and $L_w = B_w/S_w$. For $[u] \in \mathbb{Z}$ define the slicing operator:

$$\mathcal{S}_{B,S}([u]) := \big([u]^{(0)}, \ldots, [u]^{(L-1)}\big) \tag{9}$$

$$\text{so that } [u] = \sum_{j=0}^{L-1} 2^{jS} [u]^{(j)}, \tag{10}$$

where the most-significant slice is interpreted as signed whenever $[u]$ is signed. Input activations and weights are sliced with $\mathcal{S}_{B_x,S_x}([\boldsymbol{x}])$ and $\mathcal{S}_{B_w,S_w}([\boldsymbol{w}])$, respectively. Substituting the slice expansions into the dot product of Sec. III-A yields:

$$[y] = \sum_{j_x=0}^{L_x-1} \sum_{j_w=0}^{L_w-1} 2^{j_x S_x + j_w S_w} \big\langle [\boldsymbol{x}]^{(j_x)}, [\boldsymbol{w}]^{(j_w)} \big\rangle. \tag{11}$$

Each inner product on the right involves only S_x bit and S_w bit operands. Applying the full-precision guarantee to the term indexed by (j_x, j_w) therefore requires a minimum column sum resolution of:

$$B_y^{(j_x,j_w)} \ \ge \ \Big\lceil 1 + \log_2\big(K \cdot G_p^{(j_x,j_w)} + 1\big) \Big\rceil \tag{12}$$

with the per product maximum $G_p^{(j_x,j_w)} = (2^{S_x-\delta_x^{(j_x)}} - \bar{\delta}_x^{(j_x)}) \cdot (2^{S_w-\delta_w^{(j_w)}} - \bar{\delta}_w^{(j_w)})$. Thus the column sum resolution depends on the slice widths rather than on the full operand widths.

Returning to our illustrative W4A8 dot-product example ($K = 128$; $B_w = S_w = 4$, signed; $B_x = 8$, unsigned) with the conventional choice of $S_x = 1$ [14, 58, 61], the required

output precision B_y is reduced from 19 bits to 12 bits. With $S_x = 1$, this scheme effectively decouples the input precision from the resolution constraints imposed by the ADC.

Nevertheless, bit-slicing introduces trade-offs. Fig. 3 visualizes these trade-offs for a simplified 2 bit activation and 2 bit weight example. Finer slicing reduces the ADC resolution requirement. However, it proportionally increases the number of conversions by a factor of $L_x L_w$. The slices can be processed temporally, spatially, or through a hybrid scheme. Contemporary PUM designs typically adopt combinations of these processing strategies. Recent advancements also exploit bit-level sparsity by skipping zero-valued slices, significantly accelerating computations compared to naive bit-sliced implementations [62, 63, 64].

C. Slice-Wise Resolution Tightening

While a column-sum resolution of $B_y = 12$ bit represents a substantial reduction from the original 19-bit baseline, it remains costly (Sec. II-D). To push beyond coarse bit-slicing, we extend A2Q+ [17] by enforcing slice-specific accumulator constraints.

Restating the column-sum bound from Eq. (8), the quantized weight vector $[\boldsymbol{w}]$ must satisfy

$$\begin{aligned}\sum_{i=1}^{K} |[x]_i|\,|[w]_i| &\leq (2^{B_x-\delta_x} - \bar{\delta}_x)\cdot \|[\boldsymbol{w}]\|_1 \\ &\leq 2^{B_y-1} - 1.\end{aligned} \tag{13}$$

which imposes the global budget on the ℓ_1–norm of the quantized weight vector,

$$\|[\boldsymbol{w}]\|_1 \leq \frac{2^{B_y-1}-1}{2^{B_x-\delta_x} - \bar{\delta}_x}. \tag{14}$$

A2Q+ strengthens this constraint by introducing a zero-centering condition on the weights, $\sum_i [w]_i = 0$. This property allows the accumulator's full dynamic range to be utilized and removes dependence on the input sign, yielding a less restrictive bound:

$$\|[\boldsymbol{w}]\|_1 \;\leq\; \frac{2^{B_y}-2}{2^{B_x}-1}, \tag{15}$$

as formalized in Proposition 3.1 of [17]. Compared to our original bound, this budget can be up to $4\times$ larger in low-bit activation regimes, significantly alleviating pressure on model weights. To enforce the bound during training without degrading accuracy, A2Q+ applies a symmetric quantization-aware reparameterization:

$$\hat{\boldsymbol{w}} = \Delta \cdot \mathcal{Q}_{B_w}(\boldsymbol{w}), \tag{16}$$

$$\boldsymbol{w} = \frac{\boldsymbol{v} - \mu_{\boldsymbol{v}}}{\|\boldsymbol{v} - \mu_{\boldsymbol{v}}\|_1} \cdot \min(g,\, G_y), \tag{17}$$

$$\mu_{\boldsymbol{v}} = \frac{1}{K}\sum_{i=1}^{K} v_i, \quad G_y = \Delta \cdot \frac{2^{B_y}-2}{2^{B_x}-1}. \tag{18}$$

Here, the subtraction of $\mu_{\boldsymbol{v}}$ enforces zero-centering directly in the reparameterization, ensuring that the quantized weights satisfy the improved bound. Instead of using the round to nearest, A2Q+ rounds the scaled floating point weights towards zero, which ensures $\|[\hat{\boldsymbol{w}}]\|_1 \leq \|\boldsymbol{w}\|_1$, which we denote by $\lfloor\cdot\rfloor$. The formulation in Eq. (17) draws inspiration from the weight normalization reparameterization introduced in [65], where a weight vector is expressed as $\boldsymbol{w} = g \cdot \boldsymbol{v}/\|\boldsymbol{v}\|$, thereby enabling the decoupled optimization of magnitude g and direction $\boldsymbol{v}/\|\boldsymbol{v}\|$.

We extend this quantization-aware scheme to operate on individual bit slices. Using the slicing operator from Sec. III-B, the sliced representation of the quantized weights $[\boldsymbol{w}]$ is given by

$$\mathcal{S}_{B_w,S_w}([\boldsymbol{w}]) = \left([\boldsymbol{w}]^{(0)}, \ldots, [\boldsymbol{w}]^{(L_w-1)}\right). \tag{19}$$

During training, each latent slice $\boldsymbol{w}^{(j)}$ with $j \in \{0, \ldots, L_w - 1\}$ is parameterized by a distinct latent direction vector $\boldsymbol{v}^{(j)} \in \mathbb{R}^K$ and magnitude $g^{(j)}$, while sharing a common step size $\Delta \in \mathbb{R}$. Each slice is independently projected onto the A2Q+ feasible set defined by the slice-wise ℓ_1–norm budget

$$G_y = \Delta \cdot \frac{2^{B_y}-2}{2^{S_x}-1} \tag{20}$$

and quantized according to Eq. (16). In contrast to the original quantizer, we reintroduce mean subtraction by modifying the A2Q+ update rule to

$$\hat{\boldsymbol{w}}^{(j)} = \Delta \cdot \mathcal{Q}_{S_w}\left(\boldsymbol{w}^{(j)}\right) + \mu_{\boldsymbol{v}^{(j)}} \tag{21}$$

In the original A2Q+ formulation, mean subtraction implicitly constrains the weight vector to lie on a $(K-1)$-dimensional hyperplane, thereby reducing the model's degrees of freedom [66]. This reduction may impair expressivity, particularly for small K. By reintroducing the mean, we preserve the full dimensionality of the weight vector, at the cost of only one additional digital dot product per analog matrix–vector multiplication.

Initialization begins with per-channel calibration using the Optimally Clipped Tensors And Vectors (OCTAV) algorithm [42], which determines the full-precision scaling factor Δ. The quantized weights are then partitioned via $\mathcal{S}_{B_w,S_w}(\cdot)$, and the EP-init procedure [17] is applied to initialize $(\boldsymbol{v}^{(j)}, g^{(j)})$ in accordance with Eq. (20). In line with that framework, both the scaling factor Δ and the slice-wise magnitude $g^{(j)}$ are parameterized exponentially. A complete derivation and theoretical motivation for EP-init is provided in Colbert et al. [17].

Finally, we adopt the regularization of [17] penalty to prevent slice-wise magnitudes from saturating when $g^{(j)} > G_y$:

$$R \;=\; \sigma(g - G_y)\,, \qquad \sigma(z) = \max\{z, 0\}. \tag{22}$$

The overall regularization contribution is then defined as $\mathcal{L}_{\text{reg}} = \sum_l \sum_i \sum_j R_{l,i,j}$, where j indexes the slice of the i-th output channel in the l-th layer. During training, this penalty is intorduced such that $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda \mathcal{L}_{\text{reg}}$, where $\mathcal{L}_{\text{task}}$ is the specific task loss and λ is a constant scalar.

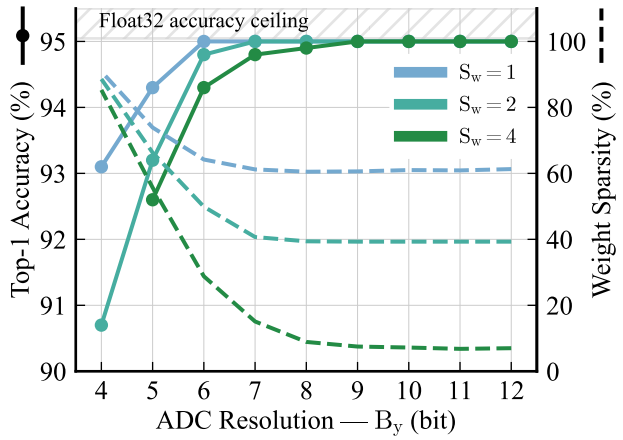

Fig. 4. Top-1 accuracy of ResNet18 on CIFAR-10 as a function of ADC resolution for the W4A8 configuration under varying weight slice widths. Naive clipping fails once resolution falls below the theoretical bound B_y from Eq. (12), whereas our slice-wise A2Q+ extension sustains high accuracy. Notably, narrower slice widths enable stronger recovery at fixed sparsity levels.

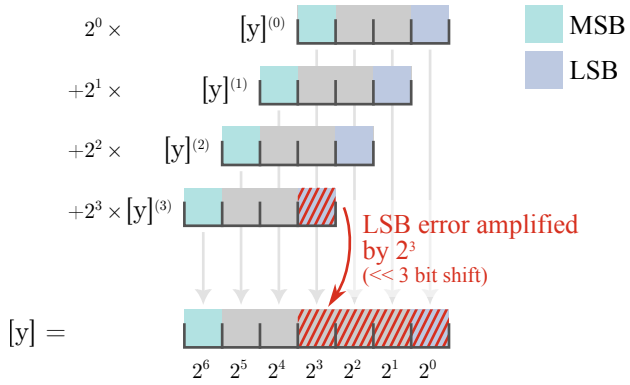

Fig. 5. Example of an erroneous bit-serial MAC operation. A single-bit error in the MSB slice can severely distort the result by amplifying the impact of the error and suppressing the contribution of lower-significance slices. In extreme cases, the full contribution of the LSB slices may be erased due to the left-shift amplification.

D. Empirical Evaluation

We assess the effectiveness of our approach on a ResNet18 model [7] pretrained on CIFAR-10 [67]. The evaluation considers ADC resolutions between 3 and 12 bits under the configuration outlined in Sec. III-B, with weights fixed to 4 bits ($B_w = 4$). The weight slice width is varied as $S_w \in \{1, 2, 4\}$, while the activation slice width remains fixed at $S_x = 1$. In total, 60 settings are explored, each repeated three times with different random seeds to ensure statistical robustness. To adapt ResNet18 for CIFAR-10, we replace the first convolution with a 3×3 kernel, stride 1, and padding 1, and remove the initial max-pooling layer to preserve spatial resolution. Fine-tuning is performed using SGD with momentum over 100 epochs, batch size 256, weight decay $1e-5$, and an initial learning rate of $1e-3$, decayed by a factor of 0.1 every 30 epochs. Further, we set the scale scalar of the regularization term defined in Eq. (22) as $\lambda = 1e-3$, following [17].

Naive quantization causes severe accuracy degradation once the ADC resolution drops 1 bit below the threshold B_y. This can be alleviated by reducing the slice width ($S_w = 1$ or 2) or by applying slice-aware training strategies. As shown in Fig. 4, our extension of A2Q+ preserves high accuracy across all resolutions. For instance, with $S_w = 4$ and $B_y = 8$, an effective allocation of only ~ 2 bits per weight is sufficient to match the full-precision baseline. Overall, the proposed method reduces the required column-wise accumulation resolution from 19 bits to as low as 7–9 bits, without significant loss in accuracy.

These results are consistent with prior findings [68, 69, 70] showing that 2-bit quantization can preserve accuracy in large-scale architectures such as Transformers [71] and ImageNet-trained networks [8]. Our framework generalizes these insights by indirectly supporting adaptive, per-weight bitwidth assignments, e.g., mixing 2- and 4-bit weights, while respecting slice-level and global ℓ_1 constraints. This added flexibility yields a strictly more expressive quantization scheme than uniform-bit methods, suggesting improved performance in more complex settings. Future work should validate these findings on a broader set of architectures, hardware configurations, and tasks to further assess generality.

IV. Linking Hardware Parameters with DNN Noise Tolerance

In this chapter, we establish quantitative limits on the tolerance of DNNs to hardware-induced noise, focusing on weight-storage imperfections (stemming from geometrical variation and programming noise) and on ADC read-out noise in capacitive PUM devices. These limits are evaluated across crossbar dimensions ranging from $K = 128$ to $K = 8192$ and for multiple operand bitwidths. The derived constraints, most notably the requirement that both weight variation and read-out noise remain within the ($3\sigma \leq 0.5$ LSB) bound, serve as a fundamental criterion for reliable inference and form the basis for the upper bound on energy efficiency, which will be addressed in the following chapter. Building on the framework introduced in Sec. III, we consider bit-sliced arithmetic for analog MVMs, where errors in the most significant bit (MSB) slices are particularly detrimental due to power-of-two weighting. A single MSB error can be amplified into large activation deviations (see Fig. 5), propagating through the network and significantly degrading inference accuracy. To mitigate this, we impose strict constraints on least significant bit (LSB) detection errors during every bit-slice cycle. Furthermore, to safeguard inference accuracy in models such as ResNet-18 [7], we employ a two-pronged strategy that combines noise-aware training, achieved by injecting Gaussian noise into the dot-product formulation of Eq. (11) as in [16], thereby enhancing network robustness, with a strict requirement that the combined weight-storage and read-out noise remains below 0.5 LSB at the three-sigma level, consistent with [16, 72].

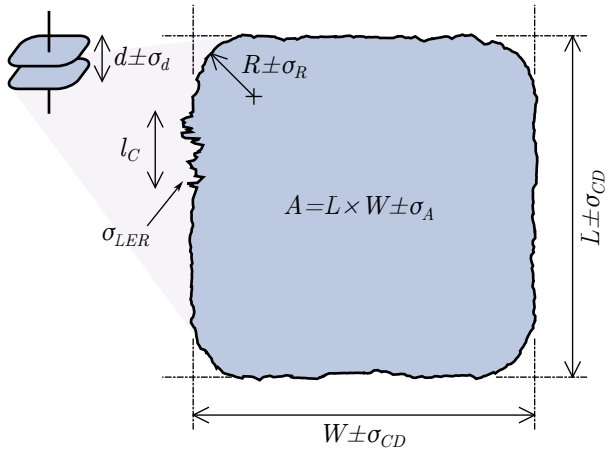

Fig. 6. Plate capacitor with geometrical variation in dielectric thickness σ_d and area variation due to critical dimension variation (σ_{cd}), line edge roughness σ_{LER} with a correlation length l_C and corner radius variation σ_R.

Under the quantization and bit-slicing scheme of Sec. III, we assume a W4A8 MVM is decomposed into $L_x = 8$ smaller W4A1 MVMs, and, following conservative assumptions, show that the required ADC resolution B_y increases by at most one bit whenever the crossbar size K doubles.

A. Geometrical Variations

Each capacitance depends on the surface area (A), the dielectric constant (ϵ_r) and the thickness (d) of the insulator according to the formula $C = \epsilon_0\epsilon_r \cdot \frac{A}{d}$. The capacitance is therefore prone to geometrical variations of the area and thickness, as shown in Fig. 6. Here we give an estimate of the variations that one can expect for an exemplary area of $1\,\mu\text{m}^2$ and a dielectric thickness of $10\,\text{nm}$.

In CMOS processing, the physical gate shape deviates from its drawn geometry due to limitations in lithography and etching. The most significant sources of this variation are critical dimension (CD) variability, line edge roughness (LER), and corner rounding. CD variation refers to fluctuations in the printed gate length and width, typically caused by exposure and etch nonuniformity. LER introduces short-range, stochastic waviness along gate edges because of resist behavior and etch chemistry. Corner rounding arises from optical diffraction limits and causes curved transitions at originally square corners.

To quantify the effect of these mechanisms on the gate area, we begin with the nominal gate area expression $A = L \cdot W$ and propagate variation in both length and width. For a nominally square gate of $L = W = 1\,\mu\text{m}$, the standard deviation of area due to CD variation is:

$$\sigma_{A,\text{CD}} = \sqrt{W^2\sigma_L^2 + L^2\sigma_W^2} = \sqrt{2} \cdot L \cdot \sigma_{\text{CD}}\,. \tag{23}$$

A 1σ CD variation of $\sigma_{\text{CD}} = 5\,\text{nm}$ is typical for 180 nm DUV processes [73], which yields $\sigma_{A,\text{CD}} \approx 7071\,\text{nm}^2$.

For LER, we estimate area variation based on edge waviness with finite spatial correlation. The contribution to area variation from LER on all four sides is approximated as:

$$\sigma_{A,\text{LER}} = \sqrt{4L \cdot l_C} \cdot \sigma_{\text{LER}}\,. \tag{24}$$

Here, σ_{LER} is the 1σ edge roughness amplitude and l_c the correlation length. At the $180\,\text{nm}$ node, typical values are $\sigma_{\text{LER}} = 2\,\text{nm}$ and $l_c = 42\,\text{nm}$ [74], resulting in $\sigma_{A,\text{LER}} \approx 820\,\text{nm}^2$.

Corner rounding reduces gate area by replacing square corners with quarter-circles. The nominal area lost per device is:

$$\Delta A = 4R^2\left(1 - \frac{\pi}{4}\right). \tag{25}$$

For a rounding radius of $R = 60\,\text{nm}$, as commonly observed in deep ultra violet (DUV) lithography [75], this yields a nominal loss of about $3090\,\text{nm}^2$. If the rounding radius itself varies locally with $\sigma_R \approx 2\,\text{nm}$, the corresponding standard deviation in area is:

$$\sigma_{A,\text{corner}} = 4 \cdot 2R\left(1 - \frac{\pi}{4}\right) \cdot \sigma_R \approx 206\,\text{nm}^2. \tag{26}$$

Assuming these sources are uncorrelated, the total 1σ area variation becomes:

$$\sigma_A = \sqrt{\sigma_{A,\text{CD}}^2 + \sigma_{A,\text{LER}}^2 + \sigma_{A,\text{corner}}^2} \approx 7121\,\text{nm}^2. \tag{27}$$

This corresponds to a relative 1σ variation of approximately 0.71% for a $1\,\mu\text{m}^2$ gate, where the CD variation is the dominant source of variation.

In more advanced lithography such as ArF immersion lithography at the $22\,\text{nm}$ node, these absolute variation become less significant due to better depth of focus. Typical values for immersion lithography are $\sigma_{\text{CD}} \approx 0.5\,\text{nm}$, $\sigma_{\text{LER}} \approx 2.5\,\text{nm}$, and $R \approx 7\,\text{nm}$, with l_c around 20–30 nm [76, 77, 78]. Applying the same formulas, total 1σ area variation for a nominal $1 \times 1\,\mu\text{m}^2$ gate reduces to around $1118\,\text{nm}^2$ (0.11%), assuming the same capacitance area.

Film thickness variation at the local (intra-die) scale is a critical contributor to device mismatch. For high-uniformity processes such as atomic layer deposition (ALD), local 1σ thickness variation is typically $\sim 0.3\,\text{nm}$ for Al_2O_3 and HfO_2, based on metrology studies [79]. The correlation length is in the range of $8\,\text{nm}$ to $10\,\text{nm}$.

Low-pressure chemical vapor deposition (LPCVD) processes also exhibit strong local control, with 1σ values of $\sim 0.6\,\text{nm}$ for thin amorphous silicon films [80] and correlation length of around $\sim 15\,\text{nm}$. Similar results are obtained for LPCVD oxides [81].

Over a $1 \times 1\,\mu\text{m}^2$ gate area, the effective variation is reduced by spatial averaging. Assuming a correlation length of $10\,\text{nm}$ and a local 1σ variation of $0.3\,\text{nm}$ for ALD grown layers, the number of uncorrelated segments is approximately $N = \left(\frac{1000nm}{10nm}\right)^2 \approx 10000$, resulting in an average thickness variation of:

$$\sigma_{\text{d}} = \frac{0.3\,\text{nm}}{\sqrt{10000}} \approx 0.003\,\text{nm}. \tag{28}$$

This illustrates how even sub-nanometer local variation can effectively average out across larger device areas, especially when the correlation length is much smaller than the feature size.

The total variation in gate capacitance arises from local fluctuations in both gate area and dielectric thickness. For a parallel plate capacitor with $C = \varepsilon_0 \varepsilon_r \cdot \frac{A}{d}$, the combined 1σ variation can be estimated using first-order error propagation:

$$\left(\frac{\sigma_C}{C}\right)^2 = \left(\frac{\sigma_A}{A}\right)^2 + \left(\frac{\sigma_d}{d}\right)^2 . \tag{29}$$

Here, we use a 1σ gate area variation of $\sigma_A = 7121\,\mathrm{nm}^2$ over $A = 1\,\mu\mathrm{m}^2 = 10^6\,\mathrm{nm}^2$, and a dielectric thickness variation of $\sigma_d = 0.003\,\mathrm{nm}$ over $d = 10\,\mathrm{nm}$. Thus, the expected total capacitance variation 1σ is approximately 0.71%, with the area variation contributing the dominant share.

B. Programming Variation

Compared to SRAM based approaches with MIM or MOM capacitors, which only encounter geometrical variation, a programmable memcapacitor exhibits also programming noise, which is caused by shot noise and drift depending on the memory mechanism. For charge trap memories, like SONOS transistors, a variation of $\sigma_{ID} = 20\,\mathrm{nA}$ for a drain current of $I_D = 400\,\mathrm{nA}$ were obtained [82]. This corresponds to a threshold voltage variation (σ_{VT}) in the subthreshold regime of:

$$\sigma_{VT} = \frac{\sigma_{\mathrm{ID}}}{I_{\mathrm{D}}} \cdot \frac{kT}{\eta q} . \tag{30}$$

With k being the Boltzmann constant, T the temperature, $\eta = 1.4$ the gate efficiency and q the electron charge. For $T = 300\,\mathrm{K}$ we get $\sigma_{\mathrm{VT}} = 0.9\,\mathrm{mV}$. In CapRAM the capacitive change between a maximum (C_{max}) and minimum (C_{min}) capacitance is based on shifting of capacitance-voltage (CV) curves and usually changes in a voltage window of $V_{\mathrm{W}} \approx 1.5\,\mathrm{V}$. Therefore, the programming accuracy of the capacitance is:

$$\frac{\sigma_{\mathrm{C,pgr}}}{C_{\mathrm{max}} - C_{\mathrm{min}}} = \frac{\sigma_{\mathrm{VT}}}{V_{\mathrm{W}}} \approx 0.062\,\% . \tag{31}$$

This high programming accuracy can be utilized to compensate also the geometrical variations.

Based on the required weight bitwidth S_{W} and the matrix size K the required $\sigma_{C,\mathrm{pgr}}$ can be estimated. To reduce the requirements for the weight bitwidth S_w, the weight is split into a positive and negative weight. This is also helpful for readout noise. The required $\sigma_{C,\mathrm{pgr}}$ for achieving $3\sigma \leq 0.5\,\mathrm{LSB}$ error is (for $S_{\mathrm{w}} > 1\,\mathrm{bit}$):

$$\frac{\sigma_{C,\mathrm{pgr}}}{C_{\mathrm{max}} - C_{\mathrm{min}}} = \frac{1}{6\sqrt{K} \cdot (2^{S_{\mathrm{w}}-1} - 1)} . \tag{32}$$

Fig. 7 shows the required programming variation versus the matrix size for different weight bitwidth. For smaller matrix sizes ($K < 256$) σ variations of $> 0.12\,\%$ are sufficient for 4 bit weights, while for larger matrices ($K > 1024$) the SONOS programming accuracy is not sufficient and likely more weight

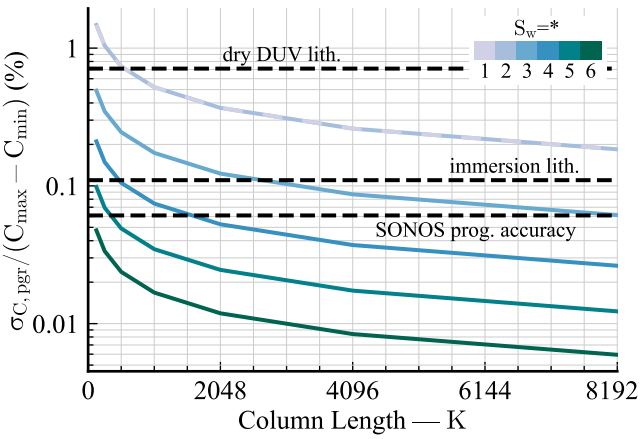

Fig. 7. Required capacitance variance to achieve a 3σ variance of 0.5 LSB for the weights for different vector sizes K and weight slice bit precisions S_{w}. The limit for the geometrical variation for DUV lithography and immersion lithography is indicated by a dashed line, as well as the SONOS programming variation due to noise and drift (a capacitance area of $1\,\mu\mathrm{m}^2$ is assumed). Note, that the curves for $S_{\mathrm{w}} = 1\,\mathrm{bit}$ and $S_w = 2\,\mathrm{bit}$ are overlapping due to differential programming.

slices are necessary (e.g. 2 bit weights for an $K = 8192$ matrix).

Another key aspect for any memcapacitive device is drift over time, which is highly dependent on temperature. The general requirement for a non-volatile memory is to meet data retention for 10 years at 85°C. In in-memory computing systems the drift requirements are much more tight, since the error can add up over the analog accumulator and the error is highly dependent on the charge state of the SONOS memory. The drift in terms of capacitance depends also on the shape of the CV curve, as shown in Fig. 11. A more flat CV curves leads to less capacitive drift than a steep one. Also, differential weights programmed around the center help to compensate drift to some extend. Assuming a maximum error of 0.5 LSB over the whole accumulator leads to a maximum allowed drift of (under the assumption of equal capacitance change per cell):

$$\Delta_C = \frac{C_{max} - C_{min}}{(2^{S_{\mathrm{w}}-1} - 1) \cdot K} \tag{33}$$

Under the assumption of a capacitance change between maximum and minimum of 90% within 1.5V this would lead to a maximum allowed voltage drift of:

$$\Delta_V = \frac{0.9 \cdot 1.5V}{(2^{S_{\mathrm{w}}-1} - 1) \cdot K} \tag{34}$$

For a 256 array and 4 Bit weights this leads to $\Delta_V = 0.7mV$. In a SONOS memory the drift can be greatly improved by emptying the flat trap states, which decay faster [82]. Even with this improvement it is expected that the retention at room temperature can be only meet for a few hours, until a refresh is necessary. A significant improvement towards weeks and even years can be expected by using band engineered SONOS (BE-SONOS), which reduces charge leakage by significant amount[83].

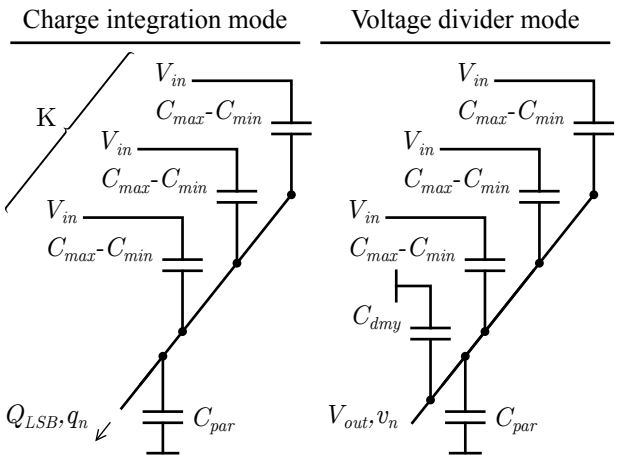

Fig. 8. Voltage divider and charge integration mode for a capacitive array including parasitic capacitance. The dummy capacitors C_{dmy} are included for the voltage divider to compensate nonlinearity.

C. Read Non-Idealities

Capacitive in-memory computing architectures are subject to various non-idealities during read-out operations that can significantly impair computational accuracy. Among these, the two most critical factors are read-out noise and read-out nonlinearity. These limitations arise from both intrinsic physical noise sources, circuit-level imperfections such as parasitic components, and fundamental device characteristics.

A fundamental source of noise in any capacitor is thermal noise, commonly referred to as kTC noise, which originates from the thermal agitation of charge carriers. The root-mean-square (RMS) voltage of this noise is given by $V_{\mathrm{rms}} = \sqrt{kT/C}$, where k is the Boltzmann constant, T is the absolute temperature and C is the capacitance. This expression illustrates that lower capacitance values lead to higher noise amplitudes, making small capacitive elements particularly vulnerable, but small capacitive elements are necessary for high weight storage density.

In practical capacitive memory arrays, additional noise contributions arise from parasitic capacitances, which are unintentionally formed due to routing, interconnects, and device layout. These parasitics do not carry a signal, but contribute fully to the thermal noise budget, degrading the signal-to-noise ratio (SNR). Because of this, minimizing parasitic effects is a key optimization target in the design of high-density capacitive analog compute-in-memory arrays. Also, the minimum possible programmed capacitance C_{min} has no additional information, but adds noise similar to the parasitic capacitance. A higher ratio of maximum C_{max} to minimum capacitance C_{min} (ON/OFF ratio) of a memcapacitive device helps to improve the overall read-out noise.

The schematic in Fig. 8 shows a representative capacitive memory array with labeled parasitic components. These parasitics interact with the programmed memcapacitors during readout and influence both signal fidelity and energy efficiency.

Capacitive arrays can be read using two principal modes: charge integration and voltage division, each interacting differently with noise and nonlinearity.

In the charge integration mode, the bitline (BL) is held at virtual ground by a transimpedance amplifier. When an input voltage is applied, all memory cell charges, including those from parasitic capacitance and the minimum capacitance of each device, are integrated in the amplifier. This means that the resulting signal includes both useful charge and unwanted noise contributions.

Assuming that each memcapacitor can be programmed between a minimum capacitance C_{min} and a maximum capacitance C_{max}, with S_w bit of weight resolution, the corresponding capacitive resolution is (for $S_{\mathrm{w}} > 1\,\mathrm{bit}$) $C_{\mathrm{LSB}} = (C_{\mathrm{max}} - C_{\mathrm{min}}) \,/\, \left(2^{S_{\mathrm{w}}} - 1\right)$. One way to increase the overall signal-to-noise ratio is the use of differential weights that are separated on two bit lines (see Fig. 1). With separation of the positive and negative range, the capacitive step increases by a factor of two: $C_{\mathrm{LSB}} = (C_{\mathrm{max}} - C_{\mathrm{min}}) \,/\, \left(2^{S_{\mathrm{w}}-1} - 1\right)$. Together with an input voltage V_{in} which is discretized into S_{x} bit, the charge corresponding to one LSB (least significant bit) of output is (for $S_w > 1\,\mathrm{bit}$):

$$Q_{\mathrm{LSB}} = \frac{C_{\mathrm{max}} - C_{\mathrm{min}}}{2^{S_{\mathrm{w}}-1} - 1} \cdot \frac{V_{\mathrm{in}}}{2^{S_{\mathrm{x}}} - 1}\,. \tag{35}$$

The noise on the bitline depends on the total capacitance seen at the node. As mentioned in the beginning of Sec. IV, the total ADC bitwidth B_y depends on the given matrix size K and the bitwidth of the weights and inputs. Therefore, the number of capacitive cells that effectively contribute to the result (N_{cap}) are given as (for $S_{\mathrm{w}} > 1\,\mathrm{bit}$):

$$N_{\mathrm{cap}} = \frac{2^{B_{\mathrm{y}}}}{2^{S_{\mathrm{w}}} \cdot 2^{S_x}}\,. \tag{36}$$

The total charge noise is therefore under the worst-case condition of a maximum accumulator:

$$q_{\mathrm{n}} = \sqrt{kT(K \cdot C_{\mathrm{min}} + C_{\mathrm{par}} + N_{\mathrm{cap}} \cdot C_{\mathrm{max}})}\,. \tag{37}$$

In the voltage divider mode, the memory cell operates as one part of a capacitive voltage divider in conjunction with the parasitic bitline capacitance. The output voltage is given by:

$$V_{\mathrm{out}} = \frac{C_{\mathrm{mem}}}{C_{\mathrm{mem}} + C_{\mathrm{par}}} \cdot V_{\mathrm{in}}\,. \tag{38}$$

This read-out mode is inherently nonlinear, especially when the ratio between C_{mem} and C_{par} varies significantly. To counteract this non-linearity, programmable dummy capacitors with a total capacitance of C_{dmy} can be added to the bitline, where the number of dummy capacitors is N_{cap}. These dummy capacitors are programmed in the opposite direction to C_{mem} and connected to ground during readout, effectively compensating for parasitic variations and eliminating the dependence on C_{mem} in the denominator. The LSB output voltage is then given as (for $S_{\mathrm{w}} > 1\,\mathrm{bit}$):

$$V_{\mathrm{LSB}} = \frac{C_{\mathrm{max}} - C_{\mathrm{min}}}{K \cdot C_{\mathrm{min}} + C_{\mathrm{par}} + C_{\mathrm{dmy}}} \cdot \frac{1}{2^{S_{\mathrm{w}}-1} - 1} \cdot \frac{V_{\mathrm{in}}}{2^{S_{\mathrm{x}}} - 1}\,. \tag{39}$$

The corresponding RMS noise voltage is:

$$v_{\mathrm{n}} \approx \sqrt{\frac{kT}{K \cdot C_{\mathrm{min}} + C_{\mathrm{par}} + N_{\mathrm{cap}} C_{\mathrm{max}}}} \,. \tag{40}$$

The resulting voltage V_{out} can be processed via a low-noise amplifier or digitized directly using an ADC. While amplification allows better control over signal levels, direct conversion demands higher voltage resolution. A significant advantage of the voltage divider method is that it does not require a virtual ground amplifier, simplifying circuit implementation.

There is no difference in signal-to-noise ratio between the voltage divider and the charge integrator mode. In order to improve the signal-to-noise values to the required 3σ margin for a half-LSB transition, averaging or oversampling is necessary. The number of necessary averages N_{av} corresponds to:

$$N_{\mathrm{av}} = \left(6 \cdot \frac{q_n}{Q_{\mathrm{LSB}}}\right)^2 . \tag{41}$$

Reformulating this formula delivers:

$$N_{\mathrm{av}} = 36 \cdot kT \cdot (K \cdot C_{\mathrm{min}} + C_{\mathrm{par}} + N_{\mathrm{cap}} \cdot C_{\mathrm{max}}) \cdot \left(2^{S_w - 1} - 1\right)^2 \cdot \left(2^{S_x} - 1\right)^2 \cdot \frac{1}{\left(C_{\mathrm{max}} - C_{\mathrm{min}}\right)^2 \cdot V_{\mathrm{in}}^2} \,. \tag{42}$$

Fig. 9 reveals the number of averages N_{av} for different capacitive ON/OFF ratios ($C_{\mathrm{max}}/C_{\mathrm{min}}$) and matrix sizes (for $C_{\mathrm{par}} = K \cdot C_{\mathrm{min}}$, $S_x = 1\,\mathrm{bit}$ and $S_w = 4\,\mathrm{bit}$, $C_{\mathrm{max}} = 1\,\mathrm{fF}$ and $V_{\mathrm{in}} = 400\,\mathrm{mV}$). A higher ON/OFF ratio reduces the number of averages, while a larger matrix increases it linearly. It is expected for large arrays that the ON/OFF ratio needs to be at least > 50 to not exceed 100 - 200 averages, for smaller arrays in the range of $K < 512$ an ON/OFF ratio of 5-10 is sufficient. The improvement in terms of N_{av} with respect to the ON/OFF ratio saturates at the lower end due to the bitwidth requirement of the ADC (B_{y}) and the amount of capacitance C_{max}. Also, there might be an optimal point of splitting large vector-matrix multiplications into smaller ones to reduce the number of averages for throughput reasons, while making compromises on the parameter density because of the ADC overhead area.

For ferroelectric implementations as memcapacitive arrays it will become crucial to improve the ON/OFF ratio, e.g. MFM (Metal-Ferroelectric-Metal) capacitors suffer a low ON/OFF ratio, due to the only minor change of capacitance with the programmed dielectric constant ($\approx 10\%$)[47], which can lead to hundreds of averages even for small arrays. Therefore, a non-linear semiconductor needs to be added to these stacks to improve the ON/OFF ratio. A ferroelectric memory compared to a SONOS memory has the advantage of a larger dielectric constant (6-7x larger) and therefore larger maximum capacitance per area.

Fig. 10 reveals the number of averages for different crossbar sizes versus different multiplied bitwidth $S_{\mathrm{x}} \times S_{\mathrm{w}}$. Reducing the bitwidth reduces the number of averages exponentially, whereas the case $S_{\mathrm{w}} = 1\,\mathrm{bit}$ is equal to $S_{\mathrm{w}} = 2\,\mathrm{bit}$ due to a non-differential implementation for $S_{\mathrm{w}} = 1\,\mathrm{bit}$. Therefore, it

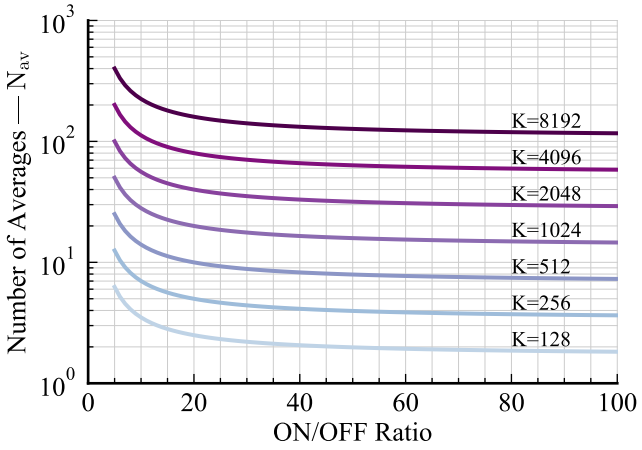

Fig. 9. Dependence of the number of averages N_{av} on the ON/OFF ratio of the memcapacitive device for different vector sizes K. An $S_w = 4\,\mathrm{bit}$ weight slice and $S_{\mathrm{x}} = 1\,\mathrm{bit}$ input slice is assumed.

is always more beneficial to do more input bit-slicing, because additional input slices increase time linearly, while additional input bits increase N_{av} exponentially (the same also holds true for the overall energy efficiency). For weight slices, there is a compromise between the parameter density and the number of averages N_{av}.

When it comes to the latency with regards to the required number of averages, Fig. 10 reveals a plot for different matrix sizes and multiplied bitwidth $S_{\mathrm{x}} \times S_{\mathrm{w}}$ for 20ns per average and 80ns per average. The latency can easily go up to $8\mu s$ for large matrices, but even with a $8\mu s$ and a 8192x8192 matrix, there are still 16.7 TOPS performed, under the assumption that all columns are calculated at the same time. A 8192x8192 matrix is estimated to take up an area of 0.37 mm² in a 64 layer 3D NAND flash type of array, which corresponds to 45.1 TOPS/mm², which is competitive to other resistive IMC implementations (few TOPS/mm²)[10]. The ADC area needs to be taken into account, where likely multiplexing becomes necessary to perform a 8192 column operation, where an estimated 10x multiplexing leads to 4.51 TOPS/mm². With reduced number of averages by improving parasitic capacitance values in the array and reducing the time spend for each average this number would be improved by a lot.

Another important consideration for accurate analog computation is the mitigation of readout nonlinearity. As previously described, one compensation technique for the voltage divider mode involves the use of programmable dummy capacitors to linearize the read-out path by canceling out parasitic and memcapacitor-dependent variations. However, another significant source of nonlinearity lies in the inherent characteristics of the memcapacitor device itself. CapRAM devices typically exhibit a nonlinear capacitance-voltage (C-V) characteristic, which is engineered to achieve programmable capacitive states through voltage-induced shifts. For example, the capacitance may change by more than 90 % within a voltage window

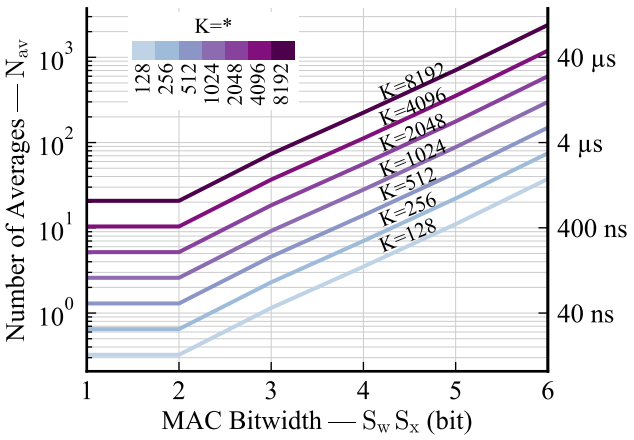

Fig. 10. Dependence of the number of averages N_{av} on the MAC bitwidth according to S_{w} and S_{x} and different vector sizes K. The ON/OFF ratio is assumed to be 10 here. Note, that $S_w = 1\,\mathrm{bit}$ is equal to $S_{\mathrm{w}} = 2\,\mathrm{bit}$, because $S_{\mathrm{w}} = 1\,\mathrm{bit}$ is a non-differential weight. The right axis shows the corresponding conversion time for an averaging cycle time of 40ns

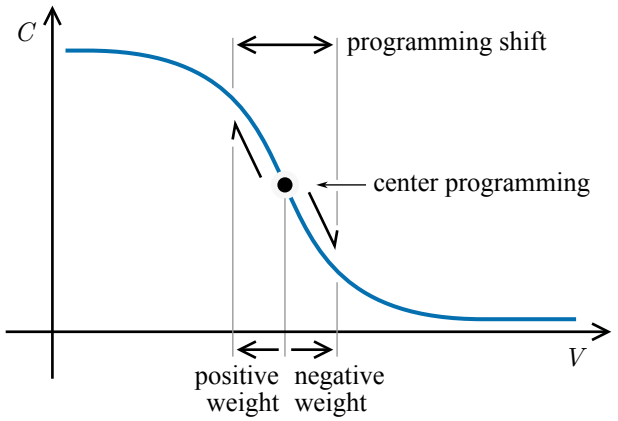

Fig. 11. Center programming of negative and positive weights allows for symmetric slope errors to cancel out.

of approximately $1.5\,\mathrm{V}$. Given a nominal read-out voltage of $400\,\mathrm{mV}$, this results in a significant voltage-dependent deviation in capacitance, which translates into output errors depending on the programmed state. An approach to mitigate effects, like mismatch and sensitivity to temperature, is the use of differential weight cells, in which a weight is encoded across two capacitors—one positive and one negative—operating in a differential manner. This technique not only reduces mismatch and sensitivity to temperature variations, but also significantly reduces read non-linearity.

The key idea is to program both the positive and negative capacitors symmetrically around the center of the C-V curve, where the slope is steepest. If the C-V curve is symmetric, then variations in the slope with respect to voltage cancel out to first order, thereby linearizing the effective weight read-out. This concept is illustrated in Fig. 11.

One trade-off of the center-symmetric programming strategy is an increase in noise, since the overall capacitance increases, while the amount of signal stays the same (center programming corresponds to an effective increase in C_{min}).

V. Translating Requirements to Energy Efficiency

As mentioned in Sec. II-D, the best in class Walden FOM ranges nowadays in the range of $0.4\,\mathrm{fJ/step}$ [53]. For the following analysis it is assumed to be $1\,\mathrm{fJ/step}$. The required bitwidth for the ADC is determined in Sec. III and is dependent on the matrix size and the bitwidth of the input and weight. For $S_x = 1$ bit and $S_w = 4$ bit, meaning a W4A1 MVM, the bitwidth requirement for the ADC depends linearly on the matrix size K and is $B_y = 8$ bit for $K = 128$ and $B_y = 14$ bit for $K = 8192$ with potential of reducing this number with further research. Assuming $1\,\mathrm{fJ/step}$ there is a constant contribution of $8\,\mathrm{fJ/OP}$ per capacitive cell from the Walden FOM, independent of the matrix size including the conversion of $L_x = 8$ input slices, which corresponds to $0.25\,\mathrm{fJ \cdot b/OP}$.

The thermal contribution is caused by the energy supplied to the capacitive device, including the number of averages. For any potential amplification following the bit line, a dynamic amplifier [84, 85] would be an energy-efficient approach due to its low static power consumption.

The thermal contribution from the capacitive device is given by number of averages, the $L_x = 8$ input slices and the fact that the capacitive device performs two operations (summation and multiplication). Also, one has to consider that each weight requires two capacitors, one positive and one negative:

$$E_{\mathrm{cap}} = \frac{8 \cdot N_{av} \cdot Q_{LSB} \cdot V_{in} \cdot 2^{B_y}}{K} + 8 \cdot N_{av} \cdot C_{min} \cdot V_{in}^2 \,. \tag{43}$$

The first term in the sum considers all the energy for the supplying the charge for 2^{B_y} LSB charge levels for the given amount of averages and bit-slices, while the second term considers the energy for charging all capacitors with C_{min}, due to limited ON/OFF ratio of the capacitive device.

Fig. 12 reveals the energy E_{cap} for different matrix sizes and ON/OFF ratios. High energy efficiencies are obtained for smaller arrays and decrease for larger arrays, due to the increased bitwidth requirement for the ADC and the Schreier FOM. A higher ON/OFF ratio helps to improve the energy efficiency.

Fig. 13 shows the total energy efficiency E_{tot}, which includes the Walden FOM. For small arrays and therefore low ADC bitwidth the Walden FOM is dominating the energy per operation while for larger arrays the thermal limit of the capacitive device becomes dominating, which is even more dominating for lower ON/OFF ratios. For an array size of $K = 256$ energy efficiencies of $9.5\,\mathrm{fJ/OP}$ ($0.3\,\mathrm{fJ \cdot b/OP}$) are obtained, while for $K = 4096$ the efficiency is $32.4\,\mathrm{fJ/OP}$ ($1.01\,\mathrm{fJ \cdot b/OP}$), assuming an ON/OFF ratio of 50 and $L_x = 8$ lower-precision W4A1 MVMs.

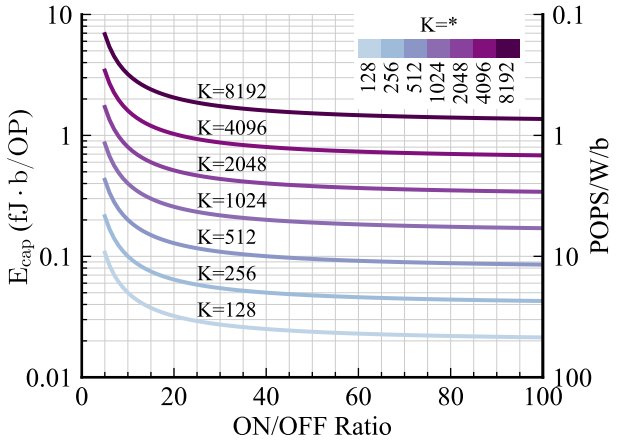

Fig. 12. Capacitive thermal energy consumption per operation (E_{cap}) versus ON/OFF ratio of the capacitive device for different vector sizes K. We assume $L_x = 8$ input slices resulting in 8× W4A1 MVMs.

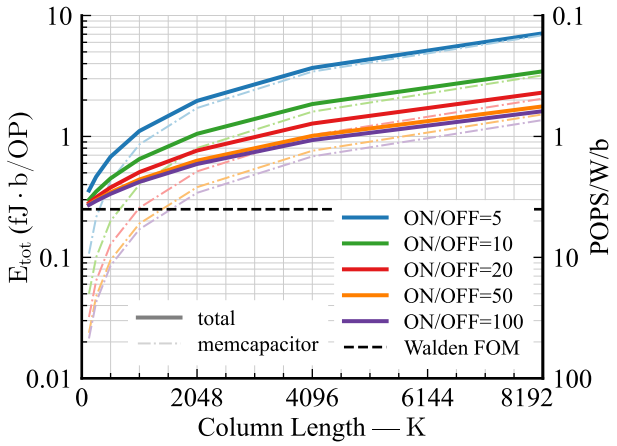

Fig. 13. Total energy per operation including the Walden FOM of the ADC versus vector size K for different ON/OFF ratios. The dashed lines show the capacitive thermal contribution, which dominates for larger matrix sizes due to the increased ADC bitwidth requirement B_y. We assume $L_x = 8$ input slices resulting in 8× W4A1 MVMs.

Improvements on energy efficiency can be obtained by choosing smaller crossbar sizes and/or more weight slices at the cost of lower parameter density. Another optimization potential are improvements on the quantisation to achieve lower σ requirements for the LSB error and/or lower ADC bitwidth B_y. Fig. 14 shows the dependency of the energy per operation on the ADC bitwidth for a $K = 8192$ matrix for an ON/OFF ratio of 50. A typical 2^{2B_y} behavior, according to the Schreier FOM, is visible and a reduction by just 2 bit from 14 bit reduces the energy per operation from $1.78\,\text{fJ} \cdot \text{b/OP}$ to $0.22\,\text{fJ} \cdot \text{b/OP}$.

In fact, the theoretical limit of energy efficiency can be calculated based on the signal-to-noise ratio, given that the amount of energy sourced in a capacitor is:

$$E_{\text{cap}} = C \cdot {V_{\text{in}}}^2 \,. \tag{44}$$

Together with the charge noise, we get the following number

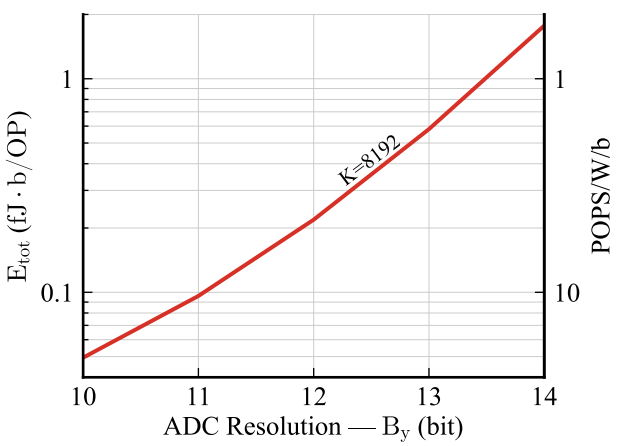

Fig. 14. Total energy per operation for a column length of $K = 8192$ and reduced bitwidth for the ADC B_y. We assume $L_x = 8$ input slices resulting in 8× W4A1 MVMs (in all other plots $B_y = 14$ bit was assumed for $K = 8192$).

of levels ($3\sigma \leq 0.5$ LSB):

$$2^{B_y} = \frac{C \cdot V_{\text{in}}}{6 \cdot \sqrt{kTC}} \,. \tag{45}$$

Reformulating this with the energy, we get:

$$E_{\text{cap}} = 36 \cdot 2^{2B_y} \cdot kT \,. \tag{46}$$

This formula represents again the exponential dependency on the bit precision for the readout B_y. For purely resistive crossbar arrays, we get similar results with a given measurement time T_{meas}:

$$E_{\text{res}} = \frac{{V_{\text{in}}}^2}{R} \cdot T_{meas} \,. \tag{47}$$

For the number of levels, we get the following, based on thermal noise:

$$2^{B_y} = \frac{V_{\text{in}}}{6 \cdot \sqrt{4kTR\Delta f}} \,. \tag{48}$$

Assuming the effective noise bandwidth Δf is limited by the amount of measurement time according to $\Delta f = 1/2T_{meas}$, we get:

$$E_{\text{res}} = 36 \cdot 2^{2B_y} \cdot 2 \cdot kT \,. \tag{49}$$

This proves, at the same bitwidth B_y, resistive devices are at least 2 times less energy efficient than capacitive devices. In many resitive devices, like RRAM or transistors in weak inversion, there is also shot noise present due to potential barriers, which leads to worse energy efficiency numbers, according to the following relationship we get:

$$2^{B_y} = \frac{I \cdot T_{meas}}{6\sqrt{qI \cdot T_{meas}}} \,. \tag{50}$$

Where I is the current through the device, which yields according to $E_{\text{shot}} = I \cdot T_{\text{meas}} \cdot V$, with the applied read-out voltage V:

$$E_{\text{shot}} = 36 \cdot 2^{2B_y} \cdot q \cdot V \,. \tag{51}$$

For a read-out voltage of $V = 0.4V$, the results are 15.4× than the capacitive case.

VI. Conclusion and Future Work

In this work, we investigated the core trade-offs in capacitive PUM systems, focusing on input and weight precision, matrix size, and the constraints imposed by both quantization and physical nonidealities. Our analysis revealed that meeting the commonly adopted fidelity constraint of $3\sigma \leq 0.5$ LSB places significant demands on weight programming accuracy, particularly as matrix size increases. To compensate for increased programming noise and geometric variation in larger arrays, additional weight slices may be required to preserve inference fidelity. On the read-out side, the output capacitance fundamentally limits the achievable SNR, making oversampling essential for maintaining accuracy at larger scales. We showed that increasing the ON/OFF ratio of the memcapacitive device substantially reduces the number of required averaging cycles, with an ON/OFF ratio exceeding 50 becoming necessary for matrices of a few thousand rows to achieve acceptable energy efficiency. ADC resolution was identified as a dominant contributor to read-out energy. In this paper, we operated under the FPG framework to preserve exact digital equivalence. However, this constraint imposes high bitwidth and energy demands, especially at scale.

Future work should investigate quantization strategies that relax the FPG constraint, enabling even lower ADC resolutions without significant loss in model accuracy. Additionally, developing methods that relax the strict $3\sigma \leq 0.5$ LSB bound could quadratically reduce the number of required averages, thereby improving energy efficiency and easing the burden on weight write accuracy. Finally, exploring whether column-sum resolution tightening can be enforced using PTQ as a more cost-effective alternative to QAT, especially for large models where full QAT is prohibitively expensive [21], would be valuable. These directions promise to unlock further efficiency gains and broaden the scalability of capacitive PUM architectures for deep learning inference.

References

[1] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, *Efficient processing of deep neural networks*. Springer, 2020.

[2] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, "Scaling laws for neural language models," 2020. [Online]. Available: https://arxiv.org/abs/2001.08361

[3] O. Mutlu, S. Ghose, J. Gómez-Luna, and R. Ausavarungnirun, "A modern primer on processing in memory," in *Emerging computing: from devices to systems: looking beyond Moore and Von Neumann*. Springer, 2022, pp. 171–243.

[4] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie, "Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 27–39, 2016.

[5] M. E. Sinangil, B. Erbagci, R. Naous, K. Akarvardar, D. Sun, W.-S. Khwa, H.-J. Liao, Y. Wang, and J. Chang, "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 Journal of Solid-State Circuits*, vol. 56, no. 1, pp. 188–198, 2021.

[6] J. Lee, B. Zhang, and N. Verma, "A switched-capacitor sram in-memory computing macro with high-precision, high-efficiency differential architecture," in *2024 IEEE European Solid-State Electronics Research Conference (ESSERC)*, 2024, pp. 357–360.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," 2015. [Online]. Available: https://arxiv.org/abs/1409.0575

[9] K.-U. Demasius, A. Kirschen, and S. Parkin, "Energy-efficient memcapacitor devices for neuromorphic computing," *Nature Electronics*, vol. 4, pp. 748–756, 2021. [Online]. Available: https://www.nature.com/articles/s41928-021-00649-y

[10] R. Khaddam-Aljameh, M. Stanisavljevic, J. Fornt Mas, G. Karunaratne, M. Brändli, F. Liu, A. Singh, S. M. Müller, U. Egger, A. Petropoulos, T. Antonakopoulos, K. Brew, S. Choi, I. Ok, F. L. Lie, N. Saulnier, V. Chan, I. Ahsan, V. Narayanan, S. R. Nandakumar, M. Le Gallo, P. A. Francese, A. Sebastian, and E. Eleftheriou, "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 Journal of Solid-State Circuits*, vol. 57, no. 4, pp. 1027–1038, 2022.

[11] W. Li, P. Xu, Y. Zhao, H. Li, Y. Xie, and Y. Lin, "Timely: Pushing data movements and interfaces in pim accelerators towards local and in time domain," in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2020, pp. 832–845.

[12] J. Shapiro, "Scaling the memory wall," https://www.luxcapital.com/news/scaling-the-memory-wall, July 2023, accessed: 2025-06-15.

[13] A. Nitayama and H. Aochi, "Bit cost scalable (bics) technology for future ultra high density storage memories," in *2013 Symposium on VLSI Technology*, 2013, pp. T60–T61.

[14] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 14–26, 2016.

[15] X. Qiao, X. Cao, H. Yang, L. Song, and H. Li, "Atomlayer: A universal reram-based cnn accelerator with atomic layer computation," in *Proceedings of the 55th Annual Design Automation Conference*, 2018, pp. 1–6.

[16] W. Zhang, S. Ando, Y.-C. Chen, and K. Yoshioka,

"Asim: Improving transparency of sram-based analog compute-in-memory research with an open-source simulation framework," *arXiv preprint arXiv:2411.11022*, 2024.

[17] I. Colbert, A. Pappalardo, J. Petri-Koenig, and Y. Umuroglu, "A2q+: Improving accumulator-aware weight quantization," *arXiv preprint arXiv:2401.10432*, 2024.

[18] R. Krishnamoorthi, "Quantizing deep convolutional networks for efficient inference: A whitepaper," *arXiv preprint arXiv:1806.08342*, 2018.

[19] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. Van Baalen, and T. Blankevoort, "A white paper on neural network quantization," *arXiv preprint arXiv:2106.08295*, 2021.

[20] S. Sharify, U. Saxena, Z. Xu, I. Soloveychik, X. Wang *et al.*, "Post training quantization of large language models with microscaling formats," *arXiv preprint arXiv:2405.07135*, 2024.

[21] Y. Li, R. Gong, X. Tan, Y. Yang, P. Hu, Q. Zhang, F. Yu, W. Wang, and S. Gu, "Brecq: Pushing the limit of post-training quantization by block reconstruction," 2021. [Online]. Available: https://arxiv.org/abs/2102.05426

[22] C. Lee, J. Jin, T. Kim, H. Kim, and E. Park, "Owq: Outlier-aware weight quantization for efficient fine-tuning and inference of large language models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 12, 2024, pp. 13 355–13 364.

[23] J. Lin, J. Tang, H. Tang, S. Yang, W.-M. Chen, W.-C. Wang, G. Xiao, X. Dang, C. Gan, and S. Han, "Awq: Activation-aware weight quantization for on-device llm compression and acceleration," *Proceedings of Machine Learning and Systems*, vol. 6, pp. 87–100, 2024.

[24] D. Du, Y. Zhang, S. Cao, J. Guo, T. Cao, X. Chu, and N. Xu, "Bitdistiller: Unleashing the potential of sub-4-bit llms via self-distillation," *arXiv preprint arXiv:2402.10631*, 2024.

[25] H. Wang, S. Ma, L. Dong, S. Huang, H. Wang, L. Ma, F. Yang, R. Wang, Y. Wu, and F. Wei, "Bitnet: Scaling 1-bit transformers for large language models," *arXiv preprint arXiv:2310.11453*, 2023.

[26] A. Abdolrashidi, L. Wang, S. Agrawal, J. Malmaud, O. Rybakov, C. Leichner, and L. Lew, "Pareto-optimal quantized resnet is mostly 4-bit," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3091–3099.

[27] G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, and S. Han, "Smoothquant: Accurate and efficient post-training quantization for large language models," in *International Conference on Machine Learning*. PMLR, 2023, pp. 38 087–38 099.

[28] C. Guo, J. Tang, W. Hu, J. Leng, C. Zhang, F. Yang, Y. Liu, M. Guo, and Y. Zhu, "Olive: Accelerating large language models via hardware-friendly outlier-victim pair quantization," in *Proceedings of the 50th Annual International Symposium on Computer Architecture*, 2023, pp. 1–15.

[29] M. Nagel, M. v. Baalen, T. Blankevoort, and M. Welling, "Data-free quantization through weight equalization and bias correction," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1325–1334.

[30] M. Nagel, R. A. Amjad, M. van Baalen, C. Louizos, and T. Blankevoort, "Up or down? adaptive rounding for post-training quantization," 2020. [Online]. Available: https://arxiv.org/abs/2004.10568

[31] J. Zhang, Y. Zhou, and R. Saab, "Post-training quantization for neural networks with provable guarantees," *SIAM Journal on Mathematics of Data Science*, vol. 5, no. 2, pp. 373–399, 2023.

[32] J. Liu, L. Niu, Z. Yuan, D. Yang, X. Wang, and W. Liu, "Pd-quant: Post-training quantization based on prediction difference metric," 2023. [Online]. Available: https://arxiv.org/abs/2212.07048

[33] Y. Jeon, C. Lee, and H.-y. Kim, "Genie: Show me the data for quantization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 12 064–12 073.

[34] C. Sakr and N. Shanbhag, "Per-tensor fixed-point quantization of the back-propagation algorithm," *arXiv preprint arXiv:1812.11732*, 2018.

[35] H. Xi, C. Li, J. Chen, and J. Zhu, "Training transformers with 4-bit integers," *Advances in Neural Information Processing Systems*, vol. 36, pp. 49 146–49 168, 2023.

[36] X. Sun, N. Wang, C.-Y. Chen, J. Ni, A. Agrawal, X. Cui, S. Venkataramani, K. El Maghraoui, V. V. Srinivasan, and K. Gopalakrishnan, "Ultra-low precision 4-bit training of deep neural networks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1796–1807, 2020.

[37] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," *Advances in neural information processing systems*, vol. 28, 2015.

[38] J. Choi, Z. Wang, S. Venkataramani, P. I.-J. Chuang, V. Srinivasan, and K. Gopalakrishnan, "Pact: Parameterized clipping activation for quantized neural networks," *arXiv preprint arXiv:1805.06085*, 2018.

[39] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned step size quantization," *arXiv preprint arXiv:1902.08153*, 2019.

[40] W. Guo, D. Liu, W. Xie, Y. Li, X. Ning, Z. Meng, S. Zeng, J. Lei, Z. Fang, and Y. Wang, "Towards accurate and efficient sub-8-bit integer training," *arXiv preprint arXiv:2411.10948*, 2024.

[41] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *arXiv preprint arXiv:1308.3432*, 2013.

[42] C. Sakr, S. Dai, R. Venkatesan, B. Zimmer, W. Dally, and B. Khailany, "Optimal clipping and magnitude-aware differentiation for improved quantization-aware training," in *International Conference on Machine Learning*. PMLR,

2022, pp. 19 123–19 138.

[43] Y. Bhalgat, J. Lee, M. Nagel, T. Blankevoort, and N. Kwak, "Lsq+: Improving low-bit quantization through learnable offsets and better initialization," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 696–697.

[44] S. Lee and Y. Kim, "Charge-domain static random access memory-based in-memory computing with low-cost multiply-and-accumulate operation and energy-efficient 7-bit hybrid analog-to-digital converter," *Electronics*, vol. 13, no. 3, 2024. [Online]. Available: https://www.mdpi.com/2079-9292/13/3/666

[45] K. Yoshioka, S. Ando, S. Miyagi, Y.-C. Chen, and W. Zhang, "A review of sram-based compute-in-memory circuits," *Japanese Journal of Applied Physics*, vol. 63, no. 12, p. 120802, Dec. 2024. [Online]. Available: http://dx.doi.org/10.35848/1347-4065/ad93e0

[46] Z. Chen, Z. Wen, W. Wan, A. R. Pakala, Y. Zou, W.-C. Wei, Z. Li, Y. Chen, and K. Yang, "Pico-ram: A pvt-insensitive analog compute-in-memory sram macro with in-situ multi-bit charge computing and 6t thin-cell-compatible layout," 2024. [Online]. Available: https://arxiv.org/abs/2407.12829

[47] Q. Zheng, Z. Wang, N. Gong, Z. Yu, C. Chen, Y. Cai, Q. Huang, H. Jiang, Q. Xia, and R. Huang, "Artificial neural network based on doped hfo2 ferroelectric capacitors with multilevel characteristics," *IEEE Electron Device Letters*, vol. 40, no. 8, pp. 1309–1312, 2019.

[48] D. Kwon and I.-Y. Chung, "Capacitive neural network using charge-stored memory cells for pattern recognition applications," *IEEE Electron Device Letters*, vol. 41, no. 3, pp. 493–496, 2020.

[49] T. You, L. P. Selvaraj, H. Zeng, W. Luo, N. Du, D. Bürger, I. Skorupa, S. Prucnal, A. Lawerenz, T. Mikolajick, O. G. Schmidt, and H. Schmidt, "An energy-efficient, bifeo3-coated capacitive switch with integrated memory and demodulation functions," *Advanced Electronic Materials*, vol. 2, no. 3, p. 1500352, 2016. [Online]. Available: https://advanced.onlinelibrary.wiley.com/doi/abs/10.1002/aelm.201500352

[50] R. Walden, "Analog-to-digital converter survey and analysis," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 4, pp. 539–550, 1999.

[51] B. Murmann, "The race for the extra decibel: A brief review of current adc performance trajectories," *IEEE Solid-State Circuits Magazine*, vol. 7, no. 3, pp. 58–66, 2015. [Online]. Available: https://ieeexplore.ieee.org/document/7168508

[52] B. E. Jonsson, "Using figures-of-merit to evaluate measured a/d-converter performance," in *Proceedings of the IEEE International Workshop on ADC Modelling and Testing (IWADC) & ADC Forum*. IEEE, 2011, location and page numbers not specified.

[53] X. Tang, J. Liu, Y. Shen, S. Li, L. Shen, A. Sanyal, K. Ragab, and N. Sun, "Low-power sar adc design: Overview and survey of state-of-the-art techniques," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 6, pp. 2249–2262, 2022.

[54] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE signal processing magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[55] T. P. Xiao, B. Feinberg, C. H. Bennett, V. Prabhakar, P. Saxena, V. Agrawal, S. Agarwal, and M. J. Marinella, "On the accuracy of analog neural network inference accelerators," *IEEE Circuits and Systems Magazine*, vol. 22, no. 4, pp. 26–48, 2023.

[56] R. Ni, H.-m. Chu, O. Castañeda Fernández, P.-y. Chiang, C. Studer, and T. Goldstein, "Wrapnet: Neural net inference with ultra-low-precision arithmetic," in *International Conference on Learning Representations ICLR 2021*. OpenReview, 2021.

[57] S. Negi, U. Saxena, D. Sharma, and K. Roy, "Hcim: Adc-less hybrid analog-digital compute in memory accelerator for deep learning workloads," in *Proceedings of the 30th Asia and South Pacific Design Automation Conference*, 2025, pp. 648–655.

[58] A. Ankit, I. E. Hajj, S. R. Chalamalasetti, G. Ndu, M. Foltin, R. S. Williams, P. Faraboschi, W.-m. W. Hwu, J. P. Strachan, K. Roy *et al.*, "Puma: A programmable ultra-efficient memristor-based accelerator for machine learning inference," in *Proceedings of the twenty-fourth international conference on architectural support for programming languages and operating systems*, 2019, pp. 715–731.

[59] M. Le Gallo, S. Nandakumar, L. Ciric, I. Boybat, R. Khaddam-Aljameh, C. Mackin, and A. Sebastian, "Precision of bit slicing with in-memory computing based on analog phase-change memory crossbars," *Neuromorphic Computing and Engineering*, vol. 2, no. 1, p. 014009, 2022.

[60] S. Diware, A. Gebregiorgis, R. V. Joshi, S. Hamdioui, and R. Bishnoi, "Unbalanced bit-slicing scheme for accurate memristor-based neural network architecture," in *2021 IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS)*. IEEE, 2021, pp. 1–4.

[61] A. Nag, R. Balasubramonian, V. Srikumar, R. Walker, A. Shafiee, J. P. Strachan, and N. Muralimanohar, "Newton: Gravitating towards the physical limits of crossbar acceleration," *IEEE Micro*, vol. 38, no. 5, pp. 41–49, 2018.

[62] D. Im, G. Park, Z. Li, J. Ryu, and H.-J. Yoo, "Sibia: Signed bit-slice architecture for dense dnn acceleration with slice-level sparsity exploitation," in *2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, 2023, pp. 69–80.

[63] M. Shi, V. Jain, A. Joseph, M. Meijer, and M. Verhelst, "Bitwave: Exploiting column-based bit-level sparsity for deep learning acceleration," in *2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 2024, pp. 732–746.

[64] Y. Chen, J. Meng, J.-s. Seo, and M. S. Abdelfattah, "Bbs: Bi-directional bit-level sparsity for deep learning acceleration," in *2024 57th IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2024, pp. 551–564.

[65] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *Advances in neural information processing systems*, vol. 29, 2016.

[66] G. Yang, J. Pennington, V. Rao, J. Sohl-Dickstein, and S. S. Schoenholz, "A mean field theory of batch normalization," *arXiv preprint arXiv:1902.08129*, 2019.

[67] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[68] Y. Li, S. Xu, B. Zhang, X. Cao, P. Gao, and G. Guo, "Q-vit: Accurate and fully quantized low-bit vision transformer," *Advances in neural information processing systems*, vol. 35, pp. 34 451–34 463, 2022.

[69] D. Du, G. Gong, and X. Chu, "Model quantization and hardware acceleration for vision transformers: A comprehensive survey," *arXiv preprint arXiv:2405.00314*, 2024.

[70] M. Sun, H. Ma, G. Kang, Y. Jiang, T. Chen, X. Ma, Z. Wang, and Y. Wang, "Vaqf: Fully automatic software-hardware co-design framework for low-bit vision transformer," *arXiv preprint arXiv:2201.06618*, 2022.

[71] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023. [Online]. Available: https://arxiv.org/abs/1706.03762

[72] C.-C. Chang, S.-T. Li, T.-L. Pan, C.-M. Tsai, I.-T. Wang, T.-S. Chang, and T.-H. Hou, "Device quantization policy in variation-aware in-memory computing design," *Scientific reports*, vol. 12, no. 1, p. 112, 2022.

[73] G. Vandenberghe, T. Marschner, K. G. Ronse, R. J. Socha, and M. V. Dusa, "CD control comparison for sub-0.18-um patterning using 248-nm lithography and strong resolution enhancement techniques," in *Optical Microlithography XII*, L. V. den Hove, Ed., vol. 3679, International Society for Optics and Photonics. SPIE, 1999, pp. 228 – 238. [Online]. Available: https://doi.org/10.1117/12.354336

[74] J. Croon, G. Storms, S. Winkelmeier, I. Pollentier, M. Ercken, S. Decoutere, W. Sansen, and H. Maes, "Line edge roughness: characterization, modeling and impact on device behavior," in *Digest. International Electron Devices Meeting,*, 2002, pp. 307–310.

[75] V. Moroz, M. Choi, and X.-W. Lin, "Systematic study of the impact of curved active and poly contours on transistor performance," in *Design for Manufacturability through Design-Process Integration III*, V. K. Singh and M. L. Rieger, Eds., vol. 7275, International Society for Optics and Photonics. SPIE, 2009, p. 72751B. [Online]. Available: https://doi.org/10.1117/12.814369

[76] Y. Ban, Y. Ma, H. J. Levinson, Y. Deng, J. Kye, and D. Z. Pan, "Modeling and characterization of contact-edge roughness for minimizing design and manufacturing variations in 32-nm node standard cell," in *Design for Manufacturability through Design-Process Integration IV*, M. L. Rieger and J. Thiele, Eds., vol. 7641, International Society for Optics and Photonics. SPIE, 2010, p. 76410D. [Online]. Available: https://doi.org/10.1117/12.846654

[77] C. Wang, R. L. Jones, E. K. Lin, W. li Wu, J. S. Villarrubia, K.-W. Choi, J. S. Clarke, B. J. Rice, M. Leeson, J. Roberts, R. Bristol, and B. Bunday, "Line edge roughness characterization of sub-50nm structures using CD-SAXS: round-robin benchmark results," in *Metrology, Inspection, and Process Control for Microlithography XXI*, C. N. Archie, Ed., vol. 6518, International Society for Optics and Photonics. SPIE, 2007, p. 65181O. [Online]. Available: https://doi.org/10.1117/12.725380

[78] T. Fujiwara, T. Toki, D. Tanaka, M. Sato, J. Kosugi, R. Tanaka, N. Sakasai, T. Ohashi, R. Nakasone, and A. Tokui, "Process window control using CDU master," in *Optical Microlithography XXV*, W. Conley, Ed., vol. 8326, International Society for Optics and Photonics. SPIE, 2012, p. 83260Q. [Online]. Available: https://doi.org/10.1117/12.916227

[79] P. A. Premkumar, A. Delabie, L. N. J. Rodriguez, A. Moussa, and C. Adelmann, "Roughness evolution during the atomic layer deposition of metal oxides," *Journal of Vacuum Science & Technology A*, vol. 31, no. 6, p. 061501, 07 2013. [Online]. Available: https://doi.org/10.1116/1.4812707

[80] A. V. Novak, V. R. Novak, and D. I. Smirnov, "Evolution of surface morphology during the growth of amorphous and polycrystalline silicon films," *Journal of Surface Investigation: X-ray, Synchrotron and Neutron Techniques*, vol. 11, no. 5, pp. 1014–1021, 2017. [Online]. Available: https://link.springer.com/article/10.1134/S1027451017050317

[81] S. Assali, A. Attiaoui, S. Mukherjee, J. Nicolas, and O. Moutanabbir, "Teos layers for low temperature processing of group iv optoelectronic devices," *Journal of Vacuum Science & Technology B*, vol. 36, no. 6, p. 061204, 10 2018. [Online]. Available: https://doi.org/10.1116/1.5047909

[82] T. P. Xiao, B. Feinberg, C. H. Bennett, V. Prabhakar, P. Saxena, V. Agrawal, S. Agarwal, and M. J. Marinella, "On the accuracy of analog neural network inference accelerators," *IEEE Circuits and Systems Magazine*, vol. 22, no. 4, pp. 26–48, 2022.

[83] S.-Y. Wang, H.-T. Lue, P.-Y. Du, C.-W. Liao, E.-K. Lai, S.-C. Lai, L.-W. Yang, T. Yang, K.-C. Chen, J. Gong, K.-Y. Hsieh, R. Liu, and C.-Y. Lu, "Reliability and processing effects of bandgap-engineered sonos (be-sonos) flash memory and study of the gate-stack scaling capability," *IEEE Transactions on Device and Materials Reliability*, vol. 8, no. 2, pp. 416–425, 2008.

[84] B. J. Hosticka, "Dynamic cmos amplifiers," *IEEE Journal of Solid-State Circuits*, vol. SC-15, no. 5, pp. 887–

894, October 1980.
[85] M. S. Akter, K. A. A. Makinwa, and K. Bult, "A capacitively degenerated 100-db linear 20–150 ms/s dynamic amplifier," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 4, pp. 1115–1126, 2018.