Fundamental Limits on the Precision of In-memory Architectures

(Invited Talk)

Sujan K. Gonugondla, Charbel Sakr, Hassan Dbouk, and Naresh R. Shanbhag
(gonugon2,sakr2,hdbouk2,shanbhag)@illinois.edu
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

ABSTRACT

This paper obtains the fundamental limits on the computational precision of in-memory computing architectures (IMCs). Various compute SNR metrics for IMCs are defined and their interrelationships analyzed to show that the accuracy of IMCs is fundamentally limited by the compute SNR (SNR_a) of its analog core, and that activation, weight and output precision needs to be assigned appropriately for the final output SNR $\text{SNR}_\text{T} \rightarrow \text{SNR}_\text{a}$. The minimum precision criterion (MPC) is proposed to minimize the output and hence the column analog-to-digital converter (ADC) precision. The charge summing (QS) compute model and its associated IMC QS-Arch are studied to obtain analytical models for its compute SNR, minimum ADC precision, energy and latency. Compute SNR models of QS-Arch are validated via Monte Carlo simulations in a 65 nm CMOS process. Employing these models, upper bounds on SNR_a of a QS-Arch-based IMC employing a 512 row SRAM array are obtained and it is shown that QS-Arch's energy cost reduces by 3.3× for every 6 dB drop in SNR_a, and that the maximum achievable SNR_a reduces with technology scaling while the energy cost at the same SNR_a increases. These models also indicate the existence of an upper bound on the dot product dimension N due to voltage headroom clipping, and this bound can be doubled for every 3 dB drop in SNR_a.

KEYWORDS

in-memory computing, taxonomy of in-memory, in-memory noise, machine learning, accelerator, in-memory precision, in-memory accuracy, compute in-memory

ACM Reference Format:
Sujan K. Gonugondla, Charbel Sakr, Hassan Dbouk, and Naresh R. Shanbhag. 2020. Fundamental Limits on the Precision of In-memory Architectures: (Invited Talk). In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD '20), November 2–5, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3400302.3416344

1 INTRODUCTION

In-memory computing (IMC) [13, 19, 28, 34] has emerged as an attractive alternative to conventional von Neumann (digital) architectures for addressing the energy and latency cost of memory accesses in data-centric machine learning workloads. IMCs embed analog mixed-signal computations in close proximity to the bit-cell array (BCA) in order to execute machine learning computations such as matrix-vector multiply (MVM) and dot products (DPs) as an intrinsic part of the read cycle and thereby avoid the need to access raw data.

IMCs exhibit a fundamental trade-off between its energy-delay product (EDP) and the accuracy or *signal-to-noise ratio* (SNR) of its analog computations. This trade-off arises due to constraints on the maximum bit-line (BL) voltage discharge and due to process variations, specifically spatial variations in the threshold voltage V_t, which limit the dynamic range and the SNR. Additionally, IMCs also exhibit noise due to the quantization of its input activation and weight parameters and due to the column analog-to-digital converters (ADCs). Henceforth, we use "compute SNR" to refer to the computational precision/accuracy of an IMC, and "precision" to the number of bits assigned to various signals.

Today, a large number of IMC prototype ICs have been demonstrated [1, 3, 4, 7, 12, 15–17, 31–33, 36, 38, 40]. While these IMCs have shown impressive reductions in the EDP over a von Neumann equivalent with minimal loss in inference accuracy, it is not clear that these gains are sustainable for larger problem sizes across data sets and inference tasks. Unlike digital architectures whose compute SNR can be made arbitrarily high by assigning sufficiently high precision to various signals, IMCs need to contend with both quantization noise as well as analog non-idealities. Therefore, IMCs will have intrinsic limits on their compute SNR. Since the compute SNR trades-off with energy and delay, it raises the following question: *What are the fundamental limits on the achievable computational precision of IMCs?*

Answering this question is made challenging due to the rich design space occupied by IMCs encompassing a huge diversity of available memory devices, bitcell circuit topologies, circuit and architectural design methods. Today's IMCs tend to employ ad-hoc approaches to assign input and ADC precisions or tend to over-provision its analog SNR in order to emulate the determinism of digital computations. An analytical understanding of the relationship between precision, compute SNR, energy, and delay in IMCs, is presently missing.

This paper attempts to fill this gap by: 1) defining compute SNR metrics for IMCs, 2) developing a systematic methodology to obtain

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ICCAD '20, November 2–5, 2020, Virtual Event, USA
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-8026-3/20/11…$15.00
https://doi.org/10.1145/3400302.3416344

a minimum precision assignment for activations, weights and outputs of fixed-point DPs realized on IMCs to meet network accuracy requirements, and 3) employing this methodology to obtain the limits on achievable compute SNR of a commonly employed IMC topology, and quantify it energy vs. accuracy trade-off.

2 NOTATION AND PRELIMINARIES

2.1 General Notation

We employ the term signal-to-quantization noise ratio (SQNR) when *only* quantization noise (denoted as q) is involved. The term SNR is employed when analog noise sources are included and use η to denote such sources. SNR is also employed when both quantization and analog noise sources are present.

2.2 The Additive Quantization Noise Model

Under the additive quantization noise model, a floating-point (FL) signal x quantized to B_x bits is represented as $x_q = x + q_x$, where q_x is the quantization noise assumed to be independent of the signal x.

If $x \in [-x_\mathrm{m}, x_\mathrm{m}]$ and $q_x \sim U[-0.5\Delta_x, 0.5\Delta_x]$ where $\Delta_x = x_\mathrm{m} 2^{-(B_x-1)}$ is the quantization step size and $U[a,b]$ denotes the uniform distribution over the interval $[a,b]$, then the signal-to-quantization noise ratio (SQNR_x) is given by:

$$\mathrm{SQNR}_{x(\mathrm{dB})} = 10\log_{10}(\mathrm{SQNR}_x) = 6B_x + 4.78 - \zeta_{x(\mathrm{dB})} \tag{1}$$

where $\mathrm{SQNR}_x = \frac{\sigma_x^2}{\sigma_{q_x}^2}$, $\sigma_{q_x}^2 = \frac{\Delta_x^2}{12}$, and $\zeta_x(\mathrm{dB}) = 10\log_{10}(\frac{x_\mathrm{m}^2}{\sigma_x^2})$ is the peak-to-average (power) ratio (PAR) of x. Equation (1) quantifies the familiar 6 dB SQNR gain per bit of precision.

2.3 The Dot-Product (DP) Computation

Consider the FL dot product (DP) computation defined as:

$$y_\mathrm{o} = \mathbf{w}^\mathsf{T}\mathbf{x} = \sum_{j=1}^{N} w_j x_j \tag{2}$$

where y_o is the DP of two N-dimensional real-valued vectors $\mathbf{w} = [w_1, \ldots, w_N]^\mathsf{T}$ (weight vector) and $\mathbf{x} = [x_1, \ldots, x_N]^\mathsf{T}$ (activation vector) of precision B_w and B_x, respectively.

In DNNs, the dot product in (2) is computed with $w \in [-w_\mathrm{m}, w_\mathrm{m}]$ (signed weights), input $x \in [0, x_\mathrm{m}]$ (unsigned activations) and output $y \in [-y_\mathrm{m}, y_\mathrm{m}]$ (signed outputs). Assuming the additive quantization noise model from Section 2.2, the fixed-point (FX) computation of the DP (2) is described by:

$$\begin{aligned} y &= \mathbf{w}_q^\mathsf{T}\mathbf{x}_q + q_y = (\mathbf{w} + \mathbf{q}_w)^\mathsf{T}(\mathbf{x} + \mathbf{q}_x) + q_y & (3) \\ &\approx \mathbf{w}^\mathsf{T}\mathbf{x} + \mathbf{w}^\mathsf{T}\mathbf{q}_x + \mathbf{q}_w^\mathsf{T}\mathbf{x} + q_y = y_\mathrm{o} + q_{iy} + q_y & (4) \end{aligned}$$

where $\mathbf{w}_q = \mathbf{w} + \mathbf{q}_w$ and $\mathbf{x}_q = \mathbf{x} + \mathbf{q}_x$ are the quantized weight and activation vectors, respectively, q_{iy} is the total input (weight and activation) quantization noise seen at the output y, and q_y is the additional output quantization noise due to round-off/truncation in digital architectures or from the finite resolution of the column ADCs in IMC architectures.

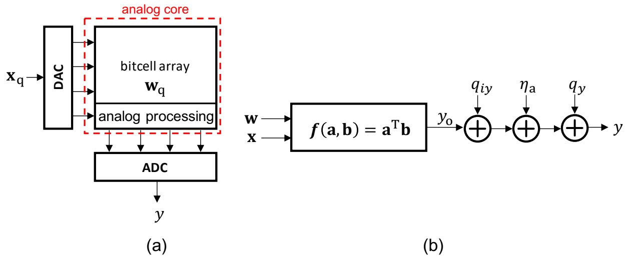


Figure 1: System noise model of IMC: (a) a generic IMC block diagram, and (b) dominant noise sources in fixed-point DP computation on IMCs.

Assuming that the weights (signed) and inputs (unsigned) are i.i.d. random variables (RVs), the variances of signals in (4) are given by:

$$\sigma_{y_\mathrm{o}}^2 = N\sigma_w^2\mathbb{E}[x^2]; \sigma_{q_y}^2 = \frac{\Delta_y^2}{12}; \sigma_{q_{iy}}^2 = \frac{N}{12}\left(\Delta_w^2\mathbb{E}[x^2] + \Delta_x^2\sigma_w^2\right) \tag{5}$$

where σ_w^2 is the variance of the weights, $\Delta_w = w_\mathrm{m}2^{-B_w+1}$, $\Delta_x = x_\mathrm{m}2^{-B_x}$ and $\Delta_y = y_\mathrm{m}2^{-B_y+1}$ are the weight, activation, and output quantization step-sizes, respectively.

3 COMPUTE SNR LIMITS OF IMCS

We propose the system noise model in Fig. 1 for obtaining precision limits on IMC architectures. Such architectures (Fig. 1(a)) accept a quantized input ($\mathbf{x}_q$) and a quantized weight vector ($\mathbf{w}_q$) to implement multiple FX DP computations of (4) in parallel in its analog core. Hence, unlike digital architectures, IMC architectures suffer from both quantization and analog noise sources such as SRAM cell current variations, thermal noise, and charge injection, as well as the limited headroom, which limits its compute SNR.

3.1 Compute SNR Metrics for IMCs

The following equations describe IMC noise model in Fig. 1:

$$y = y_\mathrm{o} + q_{iy} + \eta_\mathrm{a} + q_y; \quad \eta_\mathrm{a} = \eta_\mathrm{e} + \eta_\mathrm{h} \tag{6}$$

where y_o is the ideal DP value defined in (2), q_{iy} is the input quantization noise reflected at the output q_{iy}, η_a is the analog noise term comprising both clipping noise η_h due to limited headroom, and η_e being all other noise sources, and q_y is the quantization noise introduced by the ADC.

We define the following fundamental compute SNR metrics:

$$\mathrm{SQNR}_{q_{iy}} = \frac{\sigma_{y_\mathrm{o}}^2}{\sigma_{q_{iy}}^2}; \mathrm{SNR}_\mathrm{a} = \frac{\sigma_{y_\mathrm{o}}^2}{\sigma_{\eta_\mathrm{a}}^2}; \mathrm{SQNR}_{q_y} = \frac{\sigma_{y_\mathrm{o}}^2}{\sigma_{q_y}^2} \tag{7}$$

where SNR_a is the *analog SNR*, $\mathrm{SQNR}_{q_{iy}}$ is the *propagated SQNR* at the output due to input (weight and activation) quantization noise and is given by:

$$\begin{aligned} \mathrm{SQNR}_{q_{iy}(\mathrm{dB})} &= 6(B_x + B_w) + 4.8 - [\zeta_{x(\mathrm{dB})} + \zeta_{w(\mathrm{dB})}] \\ &\quad - 10\log_{10}\left(\frac{2^{2B_x}}{\zeta_x} + \frac{2^{2B_w}}{\zeta_w}\right) \end{aligned} \tag{8}$$

where $\zeta_{x(\mathrm{dB})} = 10\log_{10}\left(\frac{x_\mathrm{m}^2}{4\mathbb{E}[x^2]}\right)$ and $\zeta_{w(\mathrm{dB})} = 10\log_{10}\left(\frac{w_\mathrm{m}^2}{\sigma_w^2}\right)$ are the PARs of the (unsigned) activations and (signed) weights, respectively, and SQNR_{q_y} is the *digitization SQNR* solely due to ADC

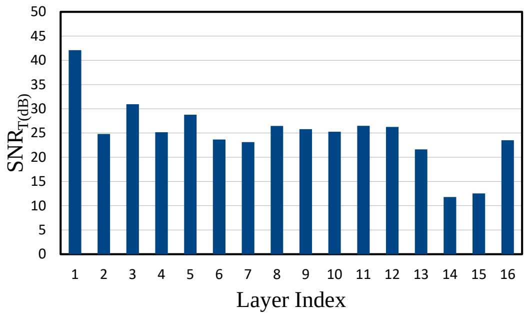


Figure 2: Per-layer $\mathrm{SNR}_{\mathrm{T(dB)}}$ requirements of DP computations in VGG-16 deployed on ImageNet.

quantization noise and is given by:

$$\mathrm{SQNR}_{q_y(\mathrm{dB})} = 6B_y + 4.8 - [\zeta_{x(\mathrm{dB})} + \zeta_{w(\mathrm{dB})}] - 10\log_{10}(N) \quad (9)$$

which is obtained by the substitutions: $B_x \leftarrow B_y$ and $\zeta_{x(\mathrm{dB})} \leftarrow \zeta_{y(\mathrm{dB})} = \zeta_{x(\mathrm{dB})} + \zeta_{w(\mathrm{dB})} + 10\log_{10}(N)$ in (1).

From (6) and (7), it is straightforward to show:

$$\mathrm{SNR_A} = \frac{\sigma^2_{y_o}}{\sigma^2_{q_{iy}} + \sigma^2_{\eta_a}} = \left[\frac{1}{\mathrm{SNR_a}} + \frac{1}{\mathrm{SQNR}_{q_{iy}}}\right]^{-1} \quad (10)$$

$$\mathrm{SNR_T} = \frac{\sigma^2_{y_o}}{\sigma^2_{q_{iy}} + \sigma^2_{\eta_a} + \sigma^2_{q_y}} = \left[\frac{1}{\mathrm{SNR_A}} + \frac{1}{\mathrm{SQNR}_{q_y}}\right]^{-1} \quad (11)$$

where $\mathrm{SNR_A}$ is the pre-ADC SNR and $\mathrm{SNR_T}$ is the total output SNR including all noise sources. Note: (10)-(11) can be repurposed for digital architectures by setting $\mathrm{SNR_a} \rightarrow \infty$ since quantization is the only noise source implying $\mathrm{SNR_A} = \mathrm{SQNR}_{q_{iy}}$. Equations (8)-(9) indicate that $\mathrm{SQNR}_{q_{iy}}$ and SQNR_{q_y} can be made arbitrarily large by assigning sufficiently high precision to the DP inputs (B_x and B_w) and the output (B_y). Thus, from (10)-(11), $\mathrm{SNR_T}$ in IMCs is fundamentally limited by $\mathrm{SNR_a}$ which depends on the analog noise sources as one expects.

3.2 Precision Assignment Methodology for IMCs

Prior work [25, 26], indicates the requirement $\mathrm{SNR}_{\mathrm{T(dB)}} > \mathrm{SNR}^*_{\mathrm{T(dB)}} =$ 10 dB-40 dB (see Fig. 2) for the inference accuracy of an FX network to be within 1% of the corresponding FL network for popular DNNs (AlexNet, VGG-9, VGG-16, ResNet-18) deployed on the ImageNet and CIFAR-10 datasets. To meet this $\mathrm{SNR}_{\mathrm{T(dB)}}$ requirement, digital architectures choose B_x and B_w such that $\mathrm{SQNR}_{q_{iy}} > \mathrm{SNR}^*_{\mathrm{T}}$, and then choose B_y sufficiently high to guarantee $\mathrm{SQNR}_{q_y} \gg \mathrm{SQNR}_{q_{iy}}$ so that $\mathrm{SNR_T} \rightarrow \mathrm{SQNR}_{q_{iy}}$.

In contrast, for IMCs, we first need to ensure that $\mathrm{SNR_a} > \mathrm{SNR}^*_{\mathrm{T}}$ so that $\mathrm{SNR_T}$ can be made to approach $\mathrm{SNR_a}$ with appropriate precision assignment via the following methodology:

(1) Assign sufficiently high values for B_x and B_w per (8) such that $\mathrm{SQNR}_{q_{iy}} \gg \mathrm{SNR_a}$ so that $\mathrm{SNR_A} \rightarrow \mathrm{SNR_a}$ per (10).

(2) Assign sufficiently a high value for B_y per (9) such that $\mathrm{SQNR}_{q_y} \gg \mathrm{SNR_A}$ so that $\mathrm{SNR_T} \rightarrow \mathrm{SNR_A}$ per (11).

For example, if $\mathrm{SQNR}_{q_{iy}(\mathrm{dB})}, \mathrm{SQNR}_{q_y(\mathrm{dB})} \geq \mathrm{SNR}_{\mathrm{a(dB)}} + 9\,\mathrm{dB}$ then $\mathrm{SNR}_{\mathrm{a(dB)}} - \mathrm{SNR}_{\mathrm{T(dB)}} \leq 0.5\,\mathrm{dB}$, i.e., $\mathrm{SNR}_{\mathrm{T(dB)}}$ lies within 0.5 dB of $\mathrm{SNR}_{\mathrm{a(dB)}}$. In this manner, by appropriate choices for B_x, B_w, and B_y, IMCs can be designed such that $\mathrm{SNR_T} \rightarrow \mathrm{SNR_a}$, which is the fundamental limit on $\mathrm{SNR_T}$.

From the above discussion it is clear that the input precisions B_x and B_w are dictated by network accuracy requirements, while the output precision B_y needs to be set sufficiently high to avoid becoming a significant noise contributor. To ensure that a sufficiently high value for B_y, digital architectures employ the bit growth criterion (BGC) described next.

3.3 Bit Growth Criterion (BGC)

The BGC is commonly employed to assign the output precision B_y in digital architectures [9, 25]. BGC sets B_y as:

$$B_y^{\mathrm{BGC}} = B_x + B_w + \log_2(N) \quad (12)$$

Substituting $B_y = B_y^{\mathrm{BGC}}$ from (12) into (9) and employing the relationship $\zeta_{y(\mathrm{dB})} = 10\log_{10}(N) + \zeta_{x(\mathrm{dB})} + \zeta_{w(\mathrm{dB})}$, the resulting SQNR due to output quantization using the BGC is given by:

$$\begin{aligned}\mathrm{SQNR}^{\mathrm{BGC}}_{q_y(\mathrm{dB})} &= 10\log_{10}\left(\frac{\sigma^2_{y_o}}{\sigma^2_{q_y}}\right)\\ &= 6(B_x + B_w) + 4.8 - [\zeta_{x(\mathrm{dB})} + \zeta_{w(\mathrm{dB})}] + 10\log_{10}(N). \quad (13)\end{aligned}$$

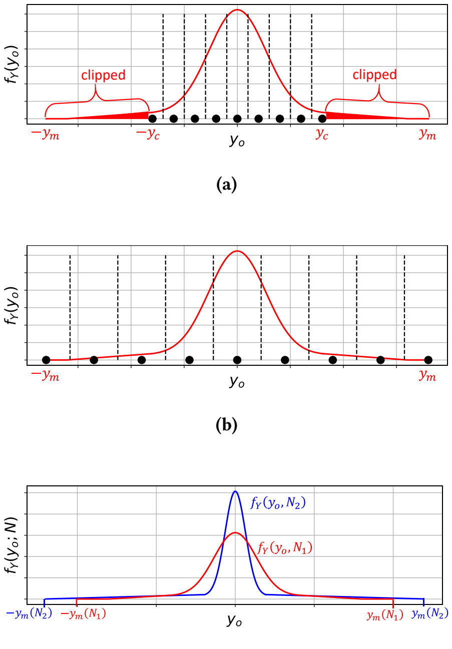
Recall that $\mathrm{SQNR}^{\mathrm{BGC}}_{q_y} \gg \mathrm{SNR_A}$ in order to ensure $\mathrm{SNR_T}$ is close to its upper bound. Comparing (9) and (13), we see that, for high values of DP dimensionality N, BGC is overly conservative since it assigns large values to B_y per (12). Some digital architectures truncate the LSBs to control bit growth. The SQNR of such truncated BGC (tBGC) can be obtained from (9) by setting the value of $B_y < B_y^{\mathrm{BGC}}$.

BGC's high precision requirements is accommodated by digital architectures by increasing the precision of arithmetic units at a commensurate increase in the computational energy, latency, and activation storage costs. However, IMCs cannot afford to use this criterion since B_y is the precision of the BL ADCs which impacts its energy, latency, and area. Indeed, recent works [24] have claimed that BL ADCs dominate the energy and latency costs of IMCs assuming BGC to assign B_y.

In the next section, we propose an alternative to BGC referred to the minimum precision criterion (MPC), that can be employed by both digital and IMC architectures which achieves a desired SQNR_{q_y} with much fewer bits than BGC.

3.4 The Minimum Precision Criterion (MPC)

We propose MPC to reduce B_y without incurring any loss in SQNR_{q_y} compared to BGC. Unlike BGC, MPC accounts for the statistics of y_o to permit controlled amounts of *clipping* to occur. In MPC (see Fig. 3(a)), the output y_o is *clipped* to lie in the range $[-y_c, y_c]$ instead of $[-y_m, y_m]$ as in BGC (see Fig. 3(b)), where $y_c < y_m$ (y_c: *clipping level*), and the B_y bits are employed to quantize this reduced range. The *clipping probability* $p_c = \Pr\{|y_o| > y_c\}$ is kept to a small user-defined value, e.g., $y_c = 4\sigma_{y_o}$ ensures that $p_c < 0.001$



(a)

(b)

(c)

Figure 3: Comparison of BGC and MPC: (a) MPC quantization levels, (b) BGC quantization levels, and (c) distribution $f_Y(y_o)$ of the ideal DP output y_o vs. DP dimensionality N.

if $y_o \sim \mathcal{N}(0, \sigma^2_{y_o})$. The resulting SQNR_y is given by:

$$\text{SQNR}^{\text{MPC}}_{q_y(\text{dB})} = 6B_y + 4.8 - \zeta^{\text{MPC}}_{y(\text{dB})} - 10\log_{10}\left(1 + p_c\frac{\sigma^2_{cc}}{\sigma^2_{q_y}}\right) \quad (14)$$

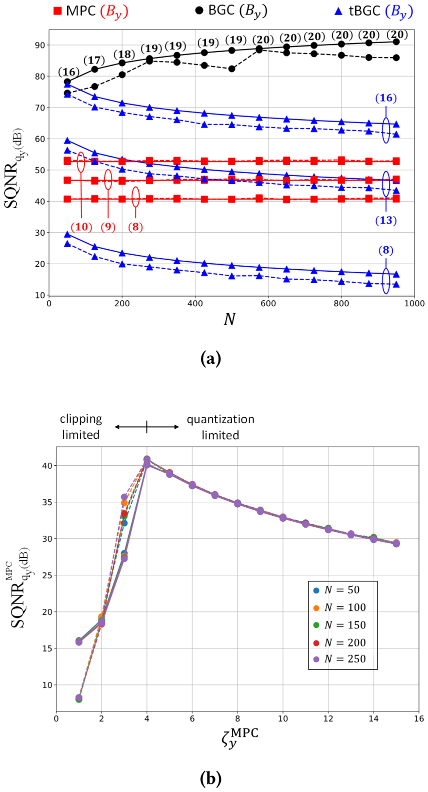
where $\zeta^{\text{MPC}}_{y(\text{dB})} = 10\log_{10}\frac{y_c^2}{\sigma^2_{y_o}}$, and $\sigma^2_{cc} = \mathbb{E}\left[(y_o - y_c)^2 \,\middle|\, |y_o| > y_c\right]$ is the *conditional clipping noise variance*. Setting $y_c = \zeta^{\text{MPC}}_y \sigma_{y_o}$ yields $\zeta^{\text{MPC}}_{y(\text{dB})} = 10\log_{10}(\zeta^{\text{MPC}}_y)^2$ indicating that p_c is a decreasing function of ζ^{MPC}_y. Thus, (14) has the same form as (1) with an additional (last term) *clipping noise factor*.

MPC exploits a key insight (see Fig. 3(c)), which follows from the Central Limit Theorem (CLT) – *in a N-dimensional DP computation (2), σ_{y_o} grows sub-linearly (as $\sqrt{N}$) as compared to the maximum y_m which grows linearly with N*. Furthermore, (14) shows a *quantization vs. clipping noise trade-off* controlled by the clipping level y_c. This trade-off, illustrated in Fig. 3(c), is absent in BGC and tBGC, and is critical to MPC's ability to realize desired values of SQNR_{q_y} with smaller values of B_y.

Assuming $y_o \sim \mathcal{N}(0, \sigma^2_{y_o})$, and substituting $y_c = 4\sigma_{y_o}$, and $p_c = 0.001$ into (14), we obtain the following lower bound:

$$B^{\text{MPC}}_y \geq \frac{1}{6}\left[\text{SNR}_{\text{A(dB)}} + 7.2 - \gamma - 10\log_{10}\left(1 - 10^{-\frac{\gamma}{10}}\right)\right] \quad (15)$$

in order for $\text{SNR}_{\text{A(dB)}} - \text{SNR}_{\text{T(dB)}} \leq \gamma$. For instance, the choice $\gamma = 0.5$ dB yields $B^{\text{MPC}}_y \geq \frac{1}{6}\left[\text{SNR}_{\text{A(dB)}} + 16.3\right]$ which corresponds to $\text{SQNR}^{\text{MPC}}_{y(\text{dB})} \geq \text{SNR}_{\text{A(dB)}} + 9$ dB as discussed in Section 3.2.



(a)

(b)

Figure 4: Trends in $\text{SQNR}_{q_y\text{(dB)}}$ for DP computation with $B_x = B_w = 7$: (a) $\text{SQNR}_{q_y\text{(dB)}}$ vs. N for MPC ($\zeta_y = 4$), BGC, tBGC, and (b) $\text{SQNR}^{\text{MPC}}_{q_y\text{(dB)}}$ vs. ζ^{MPC}_y when $B_y = 8$.

3.5 Simulation Results

To illustrate the difference between MPC, BGC and tBGC, we assume that $\text{SNR}_{\text{a(dB)}} \geq 31$ dB, so that $\text{SNR}_{\text{T(dB)}} \geq 30$ dB provided $\text{SQNR}_{q_{iy}(\text{dB})}, \text{SQNR}_{q_y(\text{dB})} \geq 40$ dB per (10)-(11). We further assume DPs of varying dimension N with 7-b quantized unsigned inputs and signed weights randomly sampled from uniform distributions. Substituting $B_x = B_w = 7$, $\zeta_{x(\text{dB})} = -1.3$ dB, and $\zeta_{w(\text{dB})} = 4.8$ dB into (8), we obtain $\text{SQNR}_{q_{iy}(\text{dB})} = 41$ dB. Thus, all that remains is to assign B_y such that $\text{SQNR}_{q_y(\text{dB})} \geq 40$ dB, for which there are three choices - MPC, BGC and tBGC.

Figure 4(a) compares the SQNR_{q_y} achieved by the three methods. Per (15), MPC meets the $\text{SQNR}_{q_y(\text{dB})} \geq 40$ dB requirement by setting $B_y = 8$ and $\zeta^{\text{MPC}}_y = 4$ independent of N. In contrast, per (12), BGC assigns $16 \leq B_y \leq 20$ as a function of N to achieve the same SNR_{T} as MPC. Furthermore, tBGC meets the SQNR_{q_y} requirement with $11 \leq B_y \leq 13$ but fails to do so with $B_y = 8$. Figure 4(b) shows that $\text{SQNR}^{\text{MPC}}_{q_y(\text{dB})}$ is maximized when $\zeta^{\text{MPC}}_y = 4$, i.e., when clipping level $y_c = 4\sigma_{y_o}$ thereby illustrating MPC's quantization vs. clipping noise trade-off described by (14). Figure 4 also validates the analytical expressions (8), (9), (13), and (14) (bold) by indicating

Table 1: A Taxonomy of IMCs using In-memory Compute Models

		In-memory Compute Model			Analog Core Precision		ADC Precision
		QS	IS	QR	B_x	B_w	B_{ADC}
CMOS	Kang *et al.* [15]	✓		✓	8	8	8
	Biswas *et al.* [1]			✓	8	1	7
	Zhang *et al.* [40]	✓			5	1	1
	Valavi *et al.* [33]			✓	1	1	1
	Khwa *et al.* [16]		✓		1	1	1
	Jiang *et al.* [12]		✓		1	1	3.46
	Si *et al.* [30]	✓		✓	2	5	5
	Jia *et al.* [11]			✓	1	1	8
	Okumura *et al.* [23]		✓		1	T	8
	Kim *et al.* [17]		✓		1	1	1
	Guo *et al.* [8]	✓			1	1	3
	Yue *et al.* [38]	✓		✓	2	5	5
	Su *et al.* [32]	✓			2	1	5
	Dong *et al.* [4]	✓		✓	4	4	4
	Si *et al.* [31]	✓			2	2	5
Beyond CMOS	Chen *et al.* [2]		✓		1	T	3
	Fick *et al.* [5]		✓		A	A	A
	Xue *et al.*[35]		✓		1	3	4
	Yan *et al.*[37]		✓		1	1	1
	Zha *et al.*[39]	✓			1	1	1
	Xue *et al.*[36]		✓		2	4	6

T: Ternary; A: Analog/Continuous-valued

a close match to ensemble-averaged values of SQNR_{q_y} obtained from Monte Carlo simulations (dotted).

Note: the theoretically optimal quantizer given an arbitrary signal distribution is obtained from the Lloyd-Max (LM) algorithm [18]. Unfortunately, the LM quantization levels are non-uniformly spaced which makes it hard to design efficient arithmetic units to process such signals. MPC offers a practical alternative to LM.

4 ANALYTICAL MODELS FOR COMPUTE SNR

This section derives analytical expressions for SNR_{a} of a typical IMC. First, we show that most IMCs can be 'explained' via a few in-memory compute models.

4.1 In-memory Compute Models

All IMCs are viewed as employing one or more *in-memory compute models* defined as a mapping of algorithmic variables y_{o}, x_j and w_j in (2) to physical quantities such as time, charge, current, or voltage, in order to (usually partially) realize an analog BL computation of the multi-bit DP in (2).

Furthermore, we suggest that most IMCs today employ one or more of the following three in-memory compute models (see Fig. 5): (a) *charge summing* (QS) [7, 14, 15, 40]; (b) *current summing* (IS) [12, 16, 17, 30]; and (c) *charge redistribution* (QR) [1, 7, 15, 33], and conjecture that these compute models are in some sense universal in that they represent an approximation to a 'complete set' of practical, i.e., realizable, mappings of variables from the algorithmic to the circuit domain as shown in Table 1.

Henceforth, we discuss the QS model and the corresponding QS-based IMC referred to as QS-Arch in detail since it is very commonly used. Analytical expressions for circuit domain equivalents of η_{e} and η_{h} in (6) for the QS model are presented. These will be combined with algorithm and precision-dependent noise sources q_{iy} and q_y to obtain SNR_{T}.

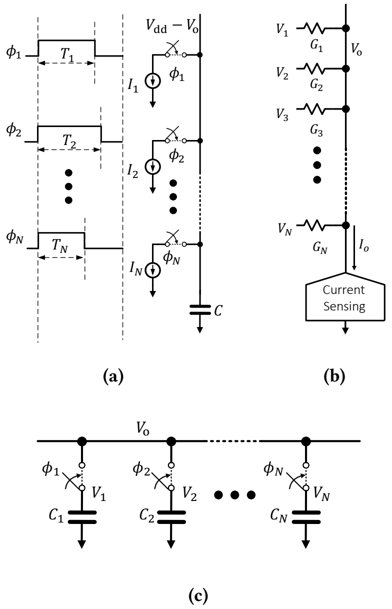


Figure 5: In-memory compute models: (a) charge summing (QS), (b) current summing (IS), and (c) charge redistribution (QR) models.

4.2 The Charge Summing (QS) Model

The QS model (see Fig. 5(a)) realizes the DP in (2) via the variable mapping ($y_{\text{o}} \to V_{\text{o}}$, $w_j \to I_j$, $x_j \to T_j$) where the cell current I_j is integrated over the WL pulse duration T_j ($j = 1, \ldots, N$) on a BL (or cell) capacitor C resulting an output voltage as shown below:

$$(y_{\text{o}} \to V_{\text{o}}) = \frac{1}{C}\sum_{j=1}^{N}(w_j \to I_j)(x_j \to T_j) \tag{16}$$

where V_{o} is the DP output assuming infinite voltage head-room, i.e., no clipping. The cell current I_j depends upon transistor sizes and the WL voltage V_{WL}, and typical values are: C (a few hundred fFs), I_j (tens of μAs), and T_j (hundreds of ps).

Noise Models: The noise contributions in QS arise from the following sources: (1) variations in the pulse-widths T_j of current switch pulses ϕ_j (Fig. 5(a)); (2) their finite rise and fall times (see Fig. 6(b)); (3) spatial variations in the currents I_j; (4) thermal noise in the discharge RC-network; and (5) clipping due to limited voltage head-room. Thus, the analog DP output V_{a} corresponding to $y_{\text{a}} = y_{\text{o}} + \eta_{\text{a}}$ is given by:

$$\begin{aligned}(y_a \to V_{\text{a}}) &= (y_{\text{o}} \to V_{\text{o}}) + (\eta_{\text{e}} \to v_{\text{e}}) + (\eta_{\text{h}} \to v_{\text{c}}),\\ v_{\text{e}} &= v_\theta + \frac{1}{C}\sum_{j=1}^{N} i_j T_j + I_j(t_j - t_{\text{rf}}),\\ v_{\text{c}} &= \min\left(V_{\text{o}},\ V_{\text{o,max}}\right) - V_{\text{o}},\end{aligned} \tag{17}$$

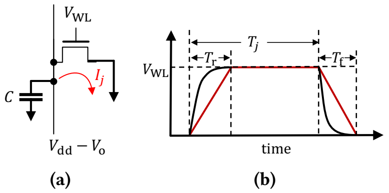


Figure 6: Modeling the discharge process in the QS compute model: (a) cell current I_j, and (b) the word-line voltage pulse V_{WL}.

Table 2: QS Model Parameters in a 65 nm CMOS Process

Parameter	Value	Parameter	Value
k' (μA/V^2)	220	α	1.8
σ_{T0} (ps)	2.3	$\sigma_{V_{\mathrm{t}}}$ (mV)	23.8
$\Delta V_{\mathrm{BL,max}}$ (V)	0.8-to-0.9	V_{WL} (V)	0.4-to-0.8
V_{t} (V)	0.4	T_0 (ps)	100
T (K)	270	k (JK^{-1})	1.38e-23

where $V_{\mathrm{o,max}}$ is the maximum allowable output voltage, and v_{e} and v_{c} are the voltage domain noise due to circuit non-idealities and clipping, respectively, $i_j \sim \mathcal{N}(0, \sigma_{I_j}^2)$ is the noise due to (spatial) current mismatch, and $t_j \sim \mathcal{N}(0, \sigma_{T_j}^2)$ is the noise due to (temporal) pulse-width mismatch, respectively, both of which are modeled as zero mean Gaussian random variables, t_{rf} models the impact of finite rise and fall times of the current switching pulses, and $v_\theta \sim \mathcal{N}(0, \sigma_\theta)$ is the thermal noise. Note: $V_{\mathrm{o,max}}$ can be as high as 0.9 V when $V_{\mathrm{dd}} = 1$ V.

Analytical expressions to estimate the noise standard deviations σ_{I_j}, σ_{T_j}, σ_θ, and t_{rf}, (see appendix) are provided below:

$$\sigma_{I_j} = I_j \left(\frac{\alpha \sigma_{V_{\mathrm{t}}}}{V_{\mathrm{WL}} - V_{\mathrm{t}}} \right) = I_j \sigma_{\mathrm{D}} \tag{18}$$

$$t_{\mathrm{rf}} = T_r - \left(\frac{V_{\mathrm{WL}} - V_{\mathrm{t}}}{V_{\mathrm{WL}}} \right) \frac{T_{\mathrm{r}} + T_{\mathrm{f}}}{\alpha + 1} \tag{19}$$

$$\sigma_{T_j} = \sqrt{h_j} \sigma_{T0}, \quad \sigma_\theta = \sqrt{\frac{kT}{C}} \tag{20}$$

where σ_{D}^2 is normalized current mismatch variance, $T_j = h_j T_0$ is the delay of a h_j-stage WL driver composed unit elements with delay T_0 each, σ_{T0} is the standard deviation of T_0, T_{r} and T_{f} are WL pulse rise and fall times (see Fig. 6(b)), α is a fitting parameter in the α-law transistor equation, $\sigma_{V_{\mathrm{t}}}$ is standard deviation of V_{t} variations, k is the Boltzmann constant, and T is the absolute temperature.

Note that typically the WL voltage V_{WL} is identical for all rows in the memory array with a few exceptions such as [40] which modulate V_{WL} to tune the cell current I_j. The effects of rise/fall times and delay variations can be mitigated by carefully designing the WL pulse generators. Therefore, noise in QS is dominated by spatial threshold voltage variations. Indeed, using the typical values from Table 2, we find that σ_{I_j}/I_j ranges from 8% to 25%, while σ_{T_j}/T_j ranges from 0.5% to 3%.

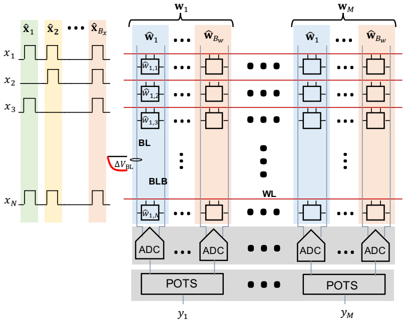


Figure 7: The charge summing IMC (QS-Arch).

Energy and Delay Models: The average energy consumption in the QS model is given by:

$$E_{\mathrm{QS}} = \mathbb{E}\left[V_{\mathrm{a}}\right] V_{\mathrm{dd}} C + E_{\mathrm{su}} \tag{21}$$

where the spatio-temporal expectation $\mathbb{E}\left[V_{\mathrm{a}}\right]$ is taken over inputs (temporal) and over columns (spatial) E_{su} is the energy cost of toggling switches ϕ_js. Equation (21) shows that the energy consumption in the QS model increases with $C \propto$ array size, the supply voltage V_{dd}, and the mean value of the DP $\mathbb{E}\left[V_{\mathrm{a}}\right]$.

The delay of the QS model is given by $T_{\mathrm{QS}} = T_{\max} + T_{\mathrm{su}}$, where T_{su} is the time required to precharge the capacitors and setup currents, and $T_{\max} = \max\{T_j\}$ is the longest allowable pulse-width.

Table 2 tabulates parameters of the QS model in a representative 65 nm CMOS process.

4.3 QS-Arch

The charge summing architecture (QS-Arch) in Fig. 7(b) employs a 6T [8] or 8T [30] SRAM bitcell within the QS model (see Section 4.2). This architecture implements fully-binarized DPs on the BLs by mapping the input bit $\hat{x}_{i,j}$ to the WL access pulse $V_{\mathrm{WL},j}$ while the weights $\hat{w}_{i,j}$ are stored across B_w columns of the BCA so that the BC currents $I_{i,j} \propto \hat{w}_{i,j}$. The output $V_{\mathrm{o}} = \Delta V_{\mathrm{BL}}$ is the voltage discharge on the BL and the capacitance $C = C_{\mathrm{BL}}$ is the BL capacitance in (16). QS-Arch sequentially (bit-serially) processes one multi-bit input vector $\mathbf{x}$ in B_x in-memory compute cycles followed by a digital summing of the binarized DPs to obtain the final multi-bit DP (2). Table 3 summarizes the noise and energy models for QS-Arch.

We derive the analytical expressions of architecture-level noise models for QS-Arch using those of the QS model described in Section 4.2. In QS-Arch, clipping occurs in each of the $B_x \times B_w$ binarized DPs and contributes to the overall clipping noise variance $\sigma_{\eta_{\mathrm{h}}}^2$ at the multi-bit DP output. Circuit noise from each binarized DP is aggregated to obtain the final circuit noise variance σ_e^2. In addition, employing MPC imposed requirement on the final DP output precision B_y (15), we obtain the lower bound on ADC precision B_{ADC}.

Since the multi-bit DP computation in (2) is high-dimensional (N can be in hundreds), it is clear that the limited BL dynamic range e.g., $V_{\mathrm{o,max}}$ in (17), will begin to dominate $\mathrm{SNR_a}$ in (7). It is for this reason that most, if not all, IMCs resort to some form of binarization of the multi-bit DP in (2) prior to employing one of the in-memory compute models (see Table 1). Ultimately, $\mathrm{SNR_a}$ limits

Table 3: Model Parameters for QS-Arch

Bitcell type	6T or 8T	Analog Core Precision	$B_x = 1, B_w = 1$
Energy cost per DP	$E_{\text{QS-Arch}} = B_w B_x (E_{\text{QS}} + E_{\text{ADC}}) + E_{\text{misc}}$	Compute model mapping	$C \to C_{\text{BL}}$ $V_{\text{o}} \to \Delta V_{\text{BL}}$ $T_j \to T_{\text{WL},j}$
$\sigma^2_{q_{iy}}$	$\frac{1}{12} N \Delta_x^2 \sigma_w^2 + \frac{1}{12} N \Delta_w^2 \mathbb{E}\left[x^2\right]$	$\sigma^2_{\eta_{\text{h}}}$	$\frac{4}{9}\left(1-4^{-B_w}\right)\left(1-4^{-B_x}\right)$ $\sum_{k=k_{\text{h}}}^{N} (k-k_{\text{h}})^2 \binom{N}{k} \left(\frac{1}{4}\right)^k \left(\frac{3}{4}\right)^{N-k}$
$\sigma^2_{\eta_{\text{e}}}$	$\frac{N\sigma_{\text{D}}^2\left(1-4^{-B_w}\right)\left(1-4^{-B_x}\right)}{9}$	B_{ADC}	$\geq \min\left(\frac{\text{SNR}_{\text{A(dB)}}+16.2}{6}, \log_2(k_{\text{h}}), \log_2(N)\right)$

$k_{\text{h}} = \frac{\Delta V_{\text{BL,max}}}{\Delta V_{\text{BL,unit}}}$; $\sigma_{\text{D}} = \frac{\sigma_I}{I}$ is the normalized standard deviation of the bit-cell current (18); $(x)_+ = \max(x, 0)$.

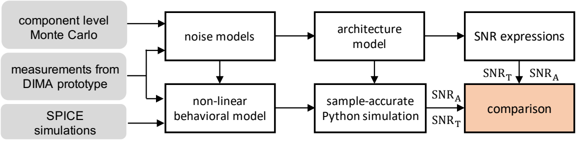


Figure 8: SNR validation methodology.

the number and accuracy of BL computations per read cycle and hence the overall energy efficiency of IMCs.

5 SIMULATION RESULTS

This section describes the noise model validation methodology for validating the noise expressions in Table 3 and simulation results for QS-Arch.

5.1 Noise Model Validation Methodology

Figure 8, we obtain the QS model parameters (Section 4) using Monte Carlo circuit simulations in a representative 65 nm CMOS process, with experimental validation of some of these, e.g., $\sigma_{\eta_{\text{e}}}$, from our IMC prototype ICs [6, 15] when possible.

Incorporating non-linear circuit behavior along with noise models, sample-accurate Monte Carlo Python simulations are employed to numerically calculate SNR values using ensemble averaged (over 1000 instances) statistics. We compare the SNR values obtained through sample-accurate simulations with those obtained by evaluating the analytical expressions in Table 3.

The quantitative results in subsequent sections employ the QS model parameter values in Table 2 along with QS-Arch energy and noise models from Table 3. An SRAM BCA with 512 rows and $C_{\text{BL}} = 270\,\text{fF}$ is assumed throughout. Energy and accuracy of QS-Arch is traded-off by tuning V_{WL}. We assume zero mean signed weights w_j and unsigned inputs x_j drawn independently from two different distributions. We set $B_x = B_w = 6$ everywhere, unless otherwise stated, so that $\text{SQNR}_{q_{iy}\text{(dB)}} = 38.9\,\text{dB} \gg \text{SNR}_{\text{a(dB)}}$ and therefore $\text{SNR}_\text{A} \approx \text{SNR}_\text{a}$ from (10). Next, we show how SNR_A and SNR_T trade-off with N and B_{ADC}.

5.2 SNR Trade-offs in QS-Arch

Figure 9(a) shows that the maximum achievable SNR_A increases with V_{WL}. Further, for a fixed V_{WL}, QS-Arch also exhibits a sharp drop in SNR_A at high values of $N > N_{\max}$, e.g., $\text{SNR}_\text{A} \approx 19.6\,\text{dB}$ for $N \leq 125$ and then drops with increase in N. A key reason for this trade-off is that $\sigma^2_{\eta_\text{h}}$ decreases while $\sigma^2_{\eta_\text{e}}$ increases as V_{WL} is reduced (see Table 3), and since $\sigma^2_{\eta_\text{h}}$ limits N and $\sigma^2_{\eta_\text{e}}$ limits SNR_a. Thus, by controlling V_{WL}, we can trade-off $N_{\max}$ with SNR_A. Specifically, $N_{\max}$ increases by 2× for every 3 dB drop in SNR_A.

In QS-Arch, the minimum value of B_{ADC} (see Table 3) depends upon the minimum of: 1) the MPC term (15); 2) the headroom clipping term; and 3) the small N case where BL discharge ΔV_{BL} has a finite number of discrete levels. Figure 9(b) shows that $\text{SNR}_\text{T} \to \text{SNR}_\text{A}$ of Fig. 9(a) when B_{ADC} is greater than the lower bound (circled) in Table 3 for different values of V_{WL} and N.

5.3 Impact of ADC Precision

Minimizing the column ADC energy is critical to maintain IMC's energy efficiency since each DP in QS-Arch requires $B_x \times B_w$ conversions. Furthermore, ADCs need to operate in a noise-limited regime due to the high PAR of high-dimensional DP outputs combined with severe area constraints imposed by column-pitch matching requirements.

ADC energy costs when operating in the noise-limited regime is modeled as [20, 21]:

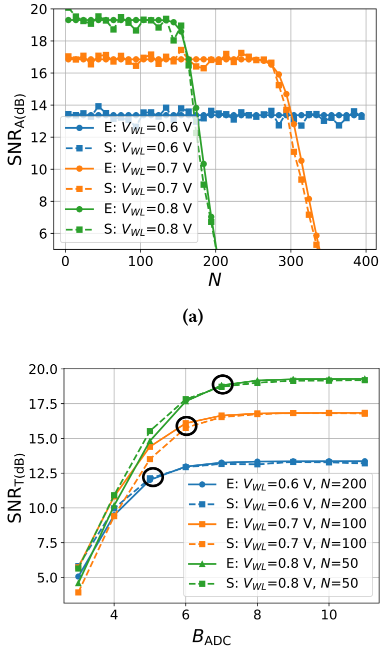
$$E_{\text{ADC}} = \beta 4^{B_{\text{ADC}}} \tag{22}$$

where β is estimated from the Schreier figure of merit [21, 27] which is approximately 180 dB based on recent (2019) ADCs [22] leading to $\beta = 7.5 \times 10^{-4}$ fJ at $V_{\text{dd}} = 1\,\text{V}$.

Figure 10 shows that ADC energy increases with DP dimension N. However, the gap between ADC energy consumption with BGC and MPC begins to increase for $N > 60$. This is because BGC assigns higher values of B_{ADC} as compared to MPC (see Table 3) to achieve the same SNR_T.

5.4 Impact of Technology Scaling

One expects IMCs to exhibit improved energy efficiency and throughput in advanced process nodes due to lower capacitance and lower supply voltage. However, the impact of technology scaling on the analog noise sources also needs to be considered. To study this trade-off, we employ the SNR and energy models from Section 5 (see Table 3) with parameters scaled as per the ITRS roadmap [10]. FDSOI technology is assumed for the 22 nm, 11 nm and 7 nm nodes.



(a)

(b)

Figure 9: Compute SNR trade-offs in the QS-Arch with $B_x = B_w = 6$: (a) SNR$_{\text{A(dB)}}$ vs. N for different values of V_{WL}, and (b) SNR$_{\text{T(dB)}}$ vs. B_{ADC} showing that the expression in Table 3 correctly predicts the minimum ADC precision B_{ADC} (circled). Close match is achieved between expressions in Table 3 (E) and simulations (S) of (17).

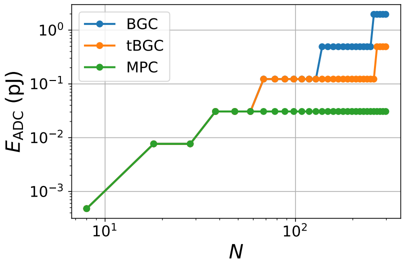


Figure 10: ADC energy in QS-Arch with $B_x = B_w = 6$ at SNR$_{\text{A(dB)}}$ = 16.2 dB as a function of N when B_{ADC} chosen according to BGC (12), tBGC, and the MPC criterion (Table 3) such that SNR$_{\text{T(dB)}}$ is within 0.5 dB of SNR$_{\text{A(dB)}}$.

For a specific node, Fig. 11 shows that the QS-Arch's energy cost reduces by 3.3× for every 6 dB drop in SNR$_{\text{A}}$, but it suffers a catastrophic drop in SNR$_{\text{A}}$ before reaching the input quantization noise limit set by (8). This drop occurs due to an increase in the clipping noise variance $\sigma^2_{\eta_{\text{h}}}$.

Across technology nodes, the maximum achievable SNR$_{\text{A}}$ in QS-Arch *reduces* as technology scales from 65 nm down to 7 nm due to: 1) increased clipping probability caused by lower supply voltages, and 2) increased variations in BL discharge voltage ΔV_{BL} due to smaller $V_{\text{dd}}/V_{\text{t}}$ ratio. As a result, Fig. 11 also shows that the energy consumption, *at the same* SNR$_{\text{A}}$, is in fact *higher* in 11 nm and 7 nm nodes as compared to the 22 nm node due to the need to employ a higher values of V_{WL} to control variations in ΔV_{BL} implying the technology scaling may not be very friendly to IMCs based on QS-Arch.

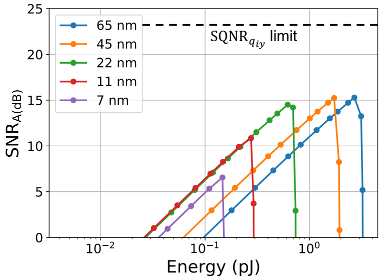


Figure 11: Impact of CMOS technology scaling on the compute SNR vs. energy trade-off in QS-Arch with $B_x = 3$, $B_w = 5$, and $N = 300$.

6 CONCLUSIONS AND SUMMARY

Based on the results presented in the earlier sections, we provide the following IMC design guidelines:

- For IMCs to be useful in realizing DNNs, the compute SNR of their analog core (SNR$_{\text{a}}$) needs to be the range 10 dB – 40 dB or greater depending on the layer.
- The total SNR (SNR$_{\text{T}}$) of DP computations implemented on IMCs is limited by SNR$_{\text{a}}$. Weight, activation, and column ADC precisions need to assigned in accordance with the minimum precision criterion (MPC) in order to minimize the energy and latency overheads, especially of column ADCs.
- For the commonly used IMC QS-Arch, given an array size, there exists a trade-off between the maximum achievable SNR$_{\text{a}}$ and the maximum realizable DP dimension N. Multi-bank IMCs will be required for high-dimensional DPs in order to boost the overall compute SNR.
- Technology scaling will have an adverse impact on QS-Arch's maximum achievable SNR$_{\text{a}}$ and the energy cost incurred for a fixed SNR$_{\text{a}}$.

An overarching conclusion of this paper is that the drive towards minimizing energy and latency using IMCs, runs counter to meeting the compute SNR requirements imposed by applications. This paper quantifies this trade-off through analytical expressions for compute SNR and energy-delay models. It is hoped that IMC designers will employ these models as they seek to optimize the design of IMCs of the future, including the use of algorithmic methods for SNR boosting such as statistical error compensation (SEC) [29].

ACKNOWLEDGMENTS

This work was sponsored by C-BRIC, the Semiconductor Research Corporation (SRC) and the Defense Advanced Research Projects Agency (DARPA).

REFERENCES

[1] Avishek Biswas and Anantha P Chandrakasan. 2018. Conv-RAM: An energy-efficient SRAM with embedded convolution computation for low-power CNN-based machine learning applications. In *IEEE International Solid-State Circuits Conference (ISSCC)*. 488–490.

[2] Wei-Hao Chen et al. 2018. A 65nm 1Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors. In *IEEE International Solid-State Circuits Conference (ISSCC)*. 494–496.

[3] Hassan Dbouk, Sujan K Gonugondla, Charbel Sakr, and Naresh R Shanbhag. 2020. KeyRAM: A 0.34 uJ/decision 18 k decisions/s Recurrent Attention In-memory Processor for Keyword Spotting. In *2020 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 1–4.

[4] Qing Dong et al. 2020. A 351 TOPS/W and 372.4 GOPS Compute-in-Memory SRAM Macro in 7nm FinFET CMOS for Machine Learning Applications. In *IEEE International Solid-State Circuits Conference (ISSCC)*. 242–243.

[5] Laura Fick, David Blaauw, Dennis Sylvester, Skylar Skrzyniarz, M Parikh, and David Fick. 2017. Analog in-memory subthreshold deep neural network accelerator. In *2017 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 1–4.

[6] Sujan Kumar Gonugondla, Mingu Kang, and Naresh Shanbhag. 2018. A 42pJ/decision 3.12 TOPS/W robust in-memory machine learning classifier with on-chip training. In *IEEE International Solid-State Circuits Conference (ISSCC)*. 490–492.

[7] Sujan K Gonugondla, Mingu Kang, and Naresh R. Shanbhag. 2018. A variation-tolerant in-memory machine learning classifier via on-chip training. *IEEE Journal of Solid-State Circuits* 53, 11 (2018), 3163–3173.

[8] Ruiqi Guo, Yonggang Liu, Shixuan Zheng, Ssu-Yen Wu, Peng Ouyang, Win-San Khwa, Xi Chen, Jia-Jing Chen, Xiudong Li, Leibo Liu, Meng-Fan Chang, Shaojun Wei, and Shouyi Yin. 2019. A 5.1pJ/Neuron 127.3us/Inference RNN-based Speech Recognition Processor using 16 Computing-in-Memory SRAM Macros in 65nm CMOS. In *2019 IEEE Symposium on VLSI Circuits*. IEEE, 120–121.

[9] Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. 2015. Deep learning with limited numerical precision. In *International Conference on Machine Learning*. 1737–1746.

[10] ITRS-collaborations. 2015. ITRS Roadmap tables. *ITRS* (2015). http://www.itrs2.net/itrs-reports.html

[11] Hongyang Jia, Yinqi Tang, Hossein Valavi, Jintao Zhang, and Naveen Verma. 2018. A Microprocessor implemented in 65nm CMOS with Configurable and Bit-scalable Accelerator for Programmable In-memory Computing. *arXiv preprint arXiv:1811.04047* (2018).

[12] Zhewei Jiang, Shihui Yin, Mingoo Seok, and Jae-sun Seo. 2018. XNOR-SRAM: In-memory computing SRAM macro for binary/ternary deep neural networks. In *2018 IEEE Symposium on VLSI Technology*. IEEE, 173–174.

[13] Mingu Kang, Sujan Gonugondla, and Naresh R Shanbhag. 2020. *Deep In-memory Architectures for Machine Learning*. Springer.

[14] Mingu Kang, Sujan K. Gonugondla, Min-Sun Keel, and Naresh R. Shanbhag. 2015. An energy-efficient memory-based high-throughput VLSI architecture for convolutional networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[15] Mingu Kang, Sujan K Gonugondla, Ameya Patil, and Naresh R. Shanbhag. 2018. A multi-functional in-memory inference processor using a standard 6T SRAM array. *IEEE Journal of Solid-State Circuits* 53, 2 (2018), 642–655.

[16] Win-San Khwa, Jia-Jing Chen, Jia-Fang Li, Xin Si, En-Yu Yang, Xiaoyu Sun, Rui Liu, Pai-Yu Chen, Qiang Li, Shimeng Yu, et al. 2018. A 65nm 4Kb algorithm-dependent computing-in-memory SRAM unit-macro with 2.3 ns and 55.8 TOPS/W fully parallel product-sum operation for binary DNN edge processors. In *IEEE International Solid-State Circuits Conference (ISSCC)*. 496–498.

[17] Jinseok Kim, Jongeun Koo, Taesu Kim, Yulhwa Kim, Hyungjun Kim, Seunghyun Yoo, and Jae-Joon Kim. 2019. Area-Efficient and Variation-Tolerant In-Memory BNN Computing using 6T SRAM Array. In *2019 IEEE Symposium on VLSI Circuits*. IEEE, 118–119.

[18] S. Lloyd. 1982. Least squares quantization in PCM. *IEEE Transactions on Information Theory* 28, 2 (1982), 129–137.

[19] M. Kang, M.-S. Keel, N. R. Shanbhag, S. Eilert, and K. Curewitz. 2014. An energy-efficient VLSI architecture for pattern recognition via deep embedding of computation in SRAM. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 8326–8330.

[20] Boris Murmann. 2008. A/D converter trends: Power dissipation, scaling and digitally assisted architectures. In *2008 IEEE Custom Integrated Circuits Conference*. IEEE, 105–112.

[21] Boris Murmann. 2015. The race for the extra decibel: a brief review of current ADC performance trajectories. *IEEE Solid-State Circuits Magazine* 7, 3 (2015), 58–66.

[22] Boris Murmann. 2019. ADC performance survey 1997-2019. https://web.stanford.edu/~murmann/adcsurvey.html

[23] Shunsuke Okumura, Makoto Yabuuchi, Kenichiro Hijioka, and Koichi Nose. 2019. A Ternary Based Bit Scalable, 8.80 TOPS/W CNN accelerator with Many-core Processing-in-memory Architecture with 896K synapses/mm2. In *2019 IEEE Symposium on VLSI Circuits*. IEEE, 248–249.

[24] Angad S. Rekhi, Brian Zimmer, Nikola Nedovic, Ningxi Liu, Rangharajan Venkatesan, Miaorong Wang, Brucek Khailany, William J. Dally, and C. Thomas Gray. 2019. Analog/Mixed-Signal Hardware Error Modeling for Deep Learning Inference. In *Proceedings of the 56th Annual Design Automation Conference 2019 (DAC '19)*. Association for Computing Machinery, New York, NY, USA, Article 81, 6 pages. https://doi.org/10.1145/3316781.3317770

[25] Charbel Sakr, Yongjune Kim, and Naresh Shanbhag. 2017. Analytical Guarantees on Numerical Precision of Deep Neural Networks. In *International Conference on Machine Learning*. 3007–3016.

[26] Charbel Sakr and Naresh Shanbhag. 2018. An analytical method to determine minimum per-layer precision of deep neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 1090–1094.

[27] Richard Schreier, Gabor C Temes, et al. 2005. *Understanding delta-sigma data converters*. Vol. 74. IEEE press Piscataway, NJ.

[28] Naresh Shanbhag, Mingu Kang, and Min-Sun Keel. 2017. Compute memory. US Patent 9,697,877, Issued July 4th., 2017.

[29] Naresh R Shanbhag, Naveen Verma, Yongjune Kim, Ameya D Patil, and Lav R Varshney. 2018. Shannon-inspired statistical computing for the nanoscale era. *Proc. IEEE* 107, 1 (2018), 90–107.

[30] Xin Si, Jia-Jing Chen, Yung-Ning Tu, Wei-Hsing Huang, Jing-Hong Wang, Yen-Cheng Chiu, Wei-Chen Wei, Ssu-Yen Wu, Xiaoyu Sun, Rui Liu, et al. 2019. A Twin-8T SRAM Computation-In-Memory Macro for Multiple-Bit CNN-Based Machine Learning. In *IEEE International Solid-State Circuits Conference (ISSCC)*. IEEE, 396–398.

[31] Xin Si, Yung-Ning Tu, Wei-Hsing Huang, Jian-Wei Su, Pei-Jung Lu, Jing-Hong Wang, Ta-Wei Liu, Ssu-Yen Wu, Ruhui Liu, Yen-Chi Chou, Zhixiao Zhang, Syuan-Hao Sie, Wei-Chen Wei, Yun-Chen Lo, Tai-Hsing Wen, Tzu-Hsiang Hsu, Yen-Kai Chen, William Shih, Chung-Chuan Lo, Ren-Shuo Liu, Chih-Cheng Hsieh, Kea-Tiong Tang, Nan-Chun Lien, Wei-Chiang Shih, Yajuan He, Qiang Li, and Meng-Fan Chang. 2020. A 28nm 64Kb 6T SRAM Computing-in- Memory Macro with 8b MAC Operation for AI Edge Chips. In *IEEE International Solid-State Circuits Conference (ISSCC)*. 246–247.

[32] Jian-Wei Su, Xin Si, Yen-Chi Chou, Ting-Wei Chang, Wei-Hsing Huang, Yung-Ning Tu, Ruhui Liu, Ta-Wei Lu, Pei-Jungand Liu, Jing-Hong Wang, Zhixiao Zhang, Hongwu Jiang, Shanshi Huang, Chung-Chuan Lo, Ren-Shuo Liu, Chih-Cheng Hsieh, Kea-Tiong Tang, Shyh-Shyuan Sheu, Sih-Han Li, Heng-Yuan Lee, Shih-Chieh Chang, Shimeng Yu, and Meng-Fan Chang. 2020. A 28nm 64Kb Inference-Training Two-Way Transpose Multibit 6T SRAM Compute-in-Memory Macro for AI Edge Chips. In *IEEE International Solid-State Circuits Conference (ISSCC)*. 240–241.

[33] Hossein Valavi, Peter J Ramadge, Eric Nestler, and Naveen Verma. 2018. A mixed-signal binarized convolutional-neural-network accelerator integrating dense weight storage and multiplication for reduced data movement. In *2018 IEEE Symposium on VLSI Circuits*. IEEE, 141–142.

[34] Naveen Verma, Hongyang Jia, Hossein Valavi, Yinqi Tang, Murat Ozatay, Lung-Yen Chen, Bonan Zhang, and Peter Deaville. 2019. In-memory computing: Advances and prospects. *IEEE Solid-State Circuits Magazine* 11, 3 (2019), 43–55.

[35] Cheng-Xin Xue, Wei-Hao Chen, Je-Syu Liu, Jia-Fang Li, Wei-Yu Lin, Wei-En Lin, Jing-Hong Wang, Wei-Chen Wei, Ting-Wei Chang, Tung-Cheng Chang, et al. 2019. A 1Mb Multibit ReRAM Computing-In-Memory Macro with 14.6 ns Parallel MAC Computing Time for CNN Based AI Edge Processors. In *IEEE International Solid-State Circuits Conference (ISSCC)*. IEEE, 388–390.

[36] Cheng-Xin Xue, Tsung-Yuan Huang, Je-Syu Liu, Ting-Wei Chang, Hui-Yao Kao, Jing-Hong Wang, Ta-Wei Liu, Shih-Ying Wei, Sheng-Po Huang, Wei-Chen Wei, Yi-Ren Chen, Tzu-Hsiang Hsu, Yen-Kai Chen, Yun-Chen Lo, Tai-Hsing Wen, Chung-Chuan Lo, Ren-Shuo Liu, Chih-Cheng Hsieh, Kea-Tiong Tang, and Meng-Fan Chang. 2020. A 22nm 2Mb ReRAM Compute-in-Memory Macro with 121-28TOPS/W for Multibit MAC Computing for Tiny AI Edge Devices. In *IEEE International Solid-State Circuits Conference (ISSCC)*. 244–245.

[37] Bonan Yan, Qing Yang, Wei-Hao Chen, Kung-Tang Chang, Jian-Wei Su, Chien-Hua Hsu, Sih-Han Li, Heng-Yuan Lee, Shyh-Shyuan Sheu, Mon-Shu Ho, et al. 2019. RRAM-based Spiking Nonvolatile Computing-In-Memory Processing Engine with Precision-Configurable In Situ Nonlinear Activation. In *2019 Symposium on VLSI Technology*. IEEE, T86–T87.

[38] Jinshan Yue, Zhe Yuan, Xiaoyu Feng, Yifan He, Zhixiao Zhang, Xin Si, Ruhui Liu, Meng-Fan Chang, Xueqing Li, Huazhong Yang, and Yongpan Liu. 2020. A 65nm Computing-in-Memory-Based CNN Processor with 2.9-to-35.8TOPS/W System Energy Efficiency Using Dynamic-Sparsity Performance-Scaling Architecture and Energy-Efficient Inter/Intra-Macro Data Reuse. In *IEEE International Solid-State Circuits Conference (ISSCC)*. 234–235.

[39] Yue Zha, Etienne Nowak, and Jing Li. 2019. Liquid Silicon: A Nonvolatile Fully Programmable Processing-In-Memory Processor with Monolithically Integrated ReRAM for Big Data/Machine Learning Applications. In *2019 IEEE Symposium on VLSI Circuits*. IEEE, 206–207.

[40] Jintao Zhang, Zhuo Wang, and Naveen Verma. 2017. In-Memory Computation of a Machine-Learning Classifier in a Standard 6T SRAM Array. *IEEE Journal of Solid-State Circuits* 52, 4 (April 2017), 915–924.