Comprehending In-memory Computing Trends via Proper Benchmarking

Naresh R. Shanbhag and Saion K. Roy

University of Illinois at Urbana-Champaign, Urbana, IL, USA

1. Introduction

Since its inception in 2014 [1], the modern version of in-memory computing (IMC) has become an active area of research in integrated circuit design globally for realizing artificial intelligence and machine learning workloads. Since 2018, > 40 IMC-related papers have been published in top circuit design conferences demonstrating significant reductions (>20X) in energy over their digital counterparts especially at the bank-level. Today, bank-level IMC designs have matured but it is not clear what the limiting factors are. This lack of clarity is due to multiple reasons including: 1) the *conceptual complexity of IMCs* due to its full-stack (devices-to-systems) nature, 2) the presence of a *fundamental energy-efficiency vs. compute SNR trade-off* due to its analog computations, and 3) the *statistical nature of machine learning workloads*. The absence of a rigorous benchmarking methodology for IMCs – a problem facing machine learning ICs in general [2] – further obfuscates the underlying trade-offs. As a result, it has become difficult to evaluate the novelty of IMC-related ideas being proposed and therefore gauge the true progress in this exciting field.

At their core, IMCs are decision(inference)-making machines. Ideally, one should benchmark IMCs using system-level metrics such as energy-per-inference (decision energy), latency-per-inference (decision latency), inference throughput, and inference accuracy. Furthermore, since energy, latency, and accuracy trade-off with each other in all decision-making systems, IMC designers need to quantify this trade-off. To do so, requires one to map complete deep nets onto IMCs accounting for all associated overheads when quantifying these metrics. However, much of the reported IMC metrics today are at the bank-level. Therefore, in this paper, we will focus primarily on bank-level benchmarking of IMCs.

Specifically, we propose a rigorous benchmarking methodology for IMCs, and employ it to analyze data collected from 50+ publications spread across CICC, VLSI and ISSCC since 2018 to explain trends and identify challenges in IMC design. Though much of the discussion is on SRAM-based IMCs [3]-[30], comparisons are drawn with eNVM-based IMCs [31]-[42] and with recent digital accelerators [45]-[57].

2. A Hierarchical View of IMCs

IMC being a full-stack technology makes it hard to comprehend various factors contributing to its overall efficiency. To address this conceptual difficulty, we propose a hierarchical view of IMCs shown in Fig. 1 where a basic building block of an IMC is referred to as an *ADC column*. An ADC column includes the circuitry and computations that precede the input to a single ADC and computes an N-dimensional dot-product. These include row drivers, bitcells, precharge, and summing circuitry. An ADC column can be arrayed to generate an IMC bank, which in turn can be arrayed to obtain a multi-bank IMC processor. This view of IMCs in Fig. 1 emphasizes focusing on a single ADC column when evaluating IMCs since the bank-level and eventually the processor-level properties are inherited from those of the ADC column.

ADC Column: We begin by defining the key *ADC column parameters* (see Fig. 1) which a designer can choose. Note: these parameters describe an ADC column's ***per read cycle*** functionality, e.g., B_X and B_W are the precisions of the input and weights, respectively, spread over R rows and C columns in diverse ways, and employed during a *single read cycle* to compute an N-dimensional dot-poduct. Per read cycle functionality allows one to distinguish between IMCs realizing true multi-bit computations within a bitcell vs. those that compose multi-bit dot-products from binary bitcell computations. Similarly, R_C (C_C) refer to the actual number of rows (columns) activated during one read cycle. Commonly known as *row (column) parallelism*, IMCs strive to maximize their values to amortize the cost of a memory read access over as many computations as possible. The ADC precision B_{ADC}, when properly chosen, provides a measure of accuracy.

Next, we define key *ADC column metrics* (see Fig. 1) that result from

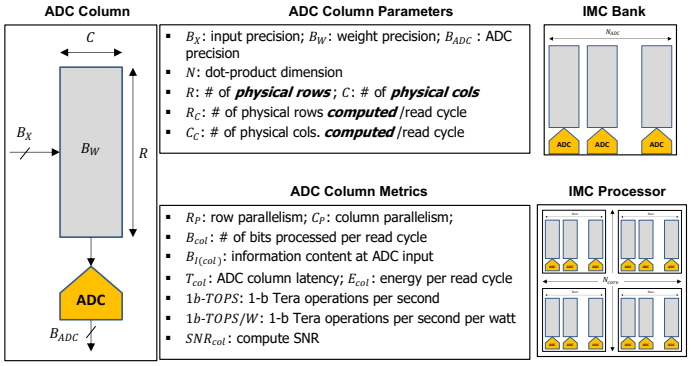

Fig. 1. An IMC processor viewed as an array of N_{core} IMC banks with each bank comprising an array of N_{ADC} parallel ADC columns. An ADC column represents the fundamental unit for constructing IMCs.

parameter choices made and the specifics of the implementation. These include: 1) row ($R_P = R_C/R$) and column ($C_P = C_C/C$) parallelism factors; 2) the number of bits processed per read cycle ($B_{col} = C_C\, R_C\, B_X = NB_W\, B_X$); 3) the information content at the input of the ADC:

$$B_{I(col)} = B_X + B_W + \log_2(N) \tag{1}$$

with exceptions for 1-b operands); 4) energy per column including ADC energy (E_{col}); 5) latency (T_{col}); 6) 1-b Tera OPS (1b-TOPS = $2B_{col}/T_{col}$); 7) 1-b Tera OPS/W (1b-TOPS/W), and 8) compute SNR (SNR_{col}). Of these, metrics 1)-3) are usually available in publications since these refer to design choices. Metrics 4)-7) are hard to measure and are almost never reported. We would like to emphasize the importance of characterizing all eight metrics to fully explain the benefits of any new IMC design method. Note: we employ bit-normalized metrics (1b-TOPS,1b-TOPS/W, and 1b-TOPS/mm^2) to enable comparison across IMCs with diverse arithmetic precisions.

Bank: An IMC bank (core) comprises N_{ADC} ADC columns operating in parallel. Thus, the bank-level parameters: 1) the number of columns ($N_{col} = N_{ADC}\, C$); and 2) the number of rows ($N_{row} = R$), are directly inherited from those of a single ADC column. Some of the bank-level metrics such as: 1) the number of bits processed per read cycle $B_{core} = N_{ADC}\, B_{col}$, and 2) the information content ($B_{I(core)} = N_{ADC}\, B_{I(col)}$) are also inherited from those of a single ADC column, while others viz. 3) energy efficiency (1b-TOPS/W); 4) throughput (1b-TOPS); 5) latency (T_{core}); 6) area (A_{core}); 7) compute density (1b-TOPS/mm^2) are reported directly. Note: we use the same notation for energy efficiency, throughput, and compute density for both ADC column-level and bank-level metrics to avoid unnecessary proliferation of symbols.

While one can extrapolate all bank-level metrics from those of an ADC column, such extrapolated metrics will not include the overheads incurred due to scale-up. However, quantifying the gap between extrapolated bank-level metrics and directly measured ones does provide useful information regarding the quality of the scale-up process.

Processor: IMC processors or multi-bank IMCs [4]-[6],[11],[12],[15],[18],[23],[25] have recently appeared. These report metrics at the system-level, e.g., decision energy, throughput, and accuracy, but also separately at the bank-level. This should enable one to compare the efficiency of the scale-up process. However, most do not map complete networks and instead map a few layers running the rest in software to obtain accuracy numbers. Furthermore, most IMC processor works exploit sparsity to enhance energy efficiency and the reported bank level numbers are for a specified level of sparsity. The true bank level metrics with full utilization are missing. For these reasons, we restrict our attention to bank-level metrics even for IMC processors.

3. Proposed IMC Benchmarking Strategy

We employ the following strategy when benchmarking IMCs:

1) All comparisons are made at the bank-level for IMCs and at the core level for digital accelerators even for multi-bank IMCs and multi-core digital processors. While we have

attempted to include as many SRAM- and eNVM-based IMC designs, we have considered only the most recent digital accelerators (since 2019).

2) We begin top-down by scaling the reported bank-level 1b-TOPS/W metrics by the precision factor, i.e., either $B_X B_W$ or $2B_X B_W$ to obtain 1b-TOPS/W to ensure that a multiply-accumulate is treated as two OPs per standard practice.
3) Next, we analyze the ADC column architecture to determine the column parameters dot product length N, R_C, C_C, B_X, B_W, and B_{ADC} per Fig. 1, and derive metrics R_P, C_P, B_{col}, and $B_{I(col)}$.
4) Using the reported value of T_{core} we compute 1b-TOPS= $2B_X B_W R_C N_{ADC}/T_{core}$. This computed throughput is compared with reported value. In case of a discrepancy (a rare occurrence), we prioritize the computed value.
5) When the total power is reported, we normalize the bottom-up computed 1b-TOPS by the reported power and compare it with the reported 1b-TOPS/W. In case of a discrepancy (a rare occurrence), we prioritize the computed value.
6) In those cases, where ADC column metrics are not reported we defer to the reported bank-level metrics. Furthermore, we verify the reported 1b-TOPS/mm^2 by normalizing the computed 1b-TOPS by the bank area, which is estimated from the die photo when it is not reported.
7) We do not scale the efficiency values by the ADC precision or normalize w.r.t. to the technology nodes to preempt controversy and only benchmark reported data.

Next, we present the trends in SRAM-based IMCs.

4. SRAM-based IMCs

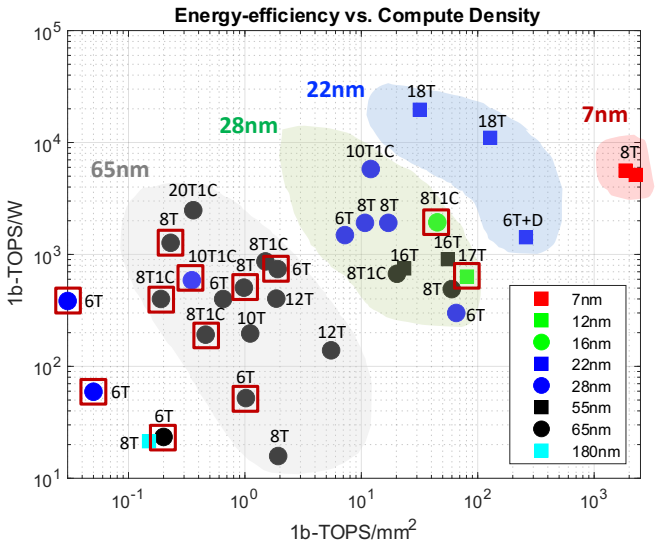

Fig. 2. 1b-TOPS/W vs. 1b-TOPS/mm^2 for SRAM-based IMCs categorized w.r.t. the technology node and bitcell architecture. All the metrics are per bank including those for IMC processors (red boxes).

Figure 2 shows that the energy efficiency 1b-TOPS/W and the compute density 1b-TOPS/mm^2 both improve with technology scaling. This lays to rest the concern that IMCs may be at a disadvantage in advanced nodes due to their analog-heavy computations. A couple of outliers are the 16nm 8T1C design [5] and the 12nm design [25] whose bank-level metrics seem to be worse than other 22nm and 28nm designs simply because these are multi-bank IMC processors. Another trend, seen in the negative slope of the shaded (same technology node) regions, is the role of the bitcell (BC) architecture. BCs employing 6T+ topologies, e.g., 8T, 10T, 8T1C, 10T1C and 18T, achieve higher 1b-TOPS/W but at the expense of 1b-TOPS/mm^2.

A key reason underlying the trends in Fig. 2 can be found in Fig. 3 where one sees designs employing complex BCs generally achieving higher throughput (1b-TOPS). This is to be expected since one primary benefit of transitioning to 6T+ BCs is the ability to reduce the cell currents without being severely impacted by spatial variations thereby enabling an increase in the dot product dimension

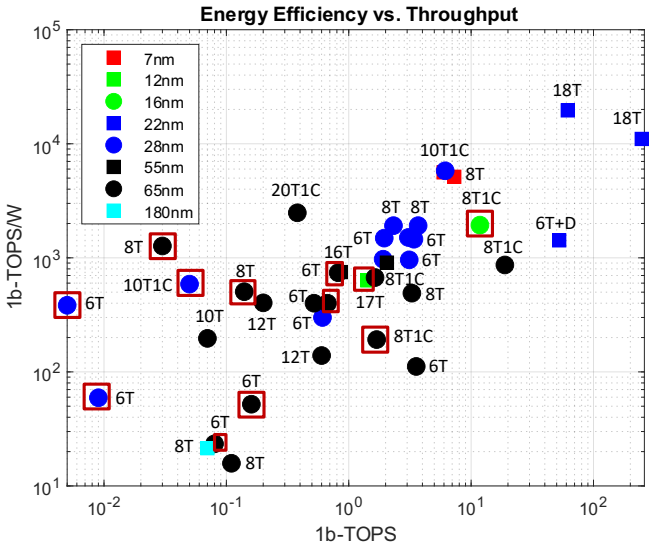

Fig. 3. 1b-TOPS/W vs. 1b-TOPS for SRAM-based IMCs. All metrics are per bank including those for IMC processors (red boxes).

N within the headroom constraints. Complex BCs such as 8T1C and 18T leverage this fact to achieve higher 1b-TOPS compared to 6T and 8T. Designs with 6T+ BCs that buck these trends, i.e., showing lower 1b-TOPS, do so for various reasons, e.g., the 12T design in [30] has low array utilization due to the use of a high ADC column muxing ratio of 1:64, the 8T design in [17] reports higher latency T_{core} to improve accuracy. Finally, the 6T+D design [7] achieves higher 1b-TOPS by performing digital accumulation thereby avoiding voltage headroom constraints.

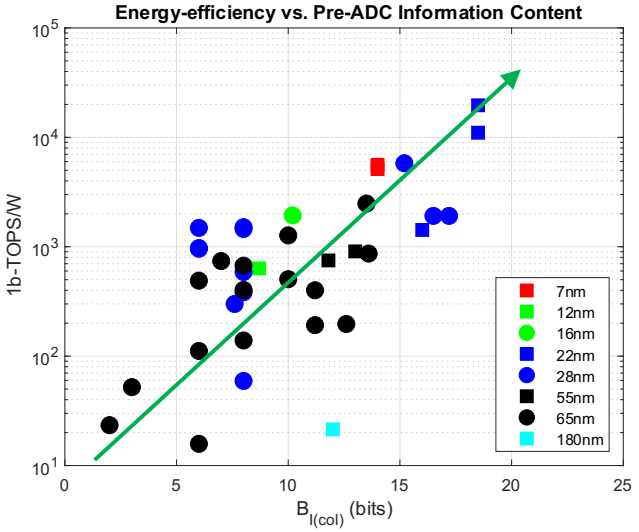

Fig. 4. 1b-TOPS/W vs. $B_{I(col)}$ for SRAM-based IMCs.

It would be useful if the bank-level trends in Figs. 2-3 could be predicted by specific ADC column level metrics. Indeed, it turns out (see Fig. 4) that energy efficiency tracks the information content $B_{I(col)}$ at the ADC input very closely. We observe a $10\times$ improvement in 1b-TOPS/W per bit increase in $B_{I(col)}$. This trend clearly points out to the importance of maximizing the information content at the ADC input to maximize energy efficiency.

5. Comparison Across Architectures

In this section, we overlay energy-efficiency, compute density and throughput metrics for SRAM-based IMCs, eNVM-based IMCs, eDRAM-based IMCs, and digital accelerators. Recall – we always

compare bank/core-level metrics throughout this paper even when comparing processor architectures.

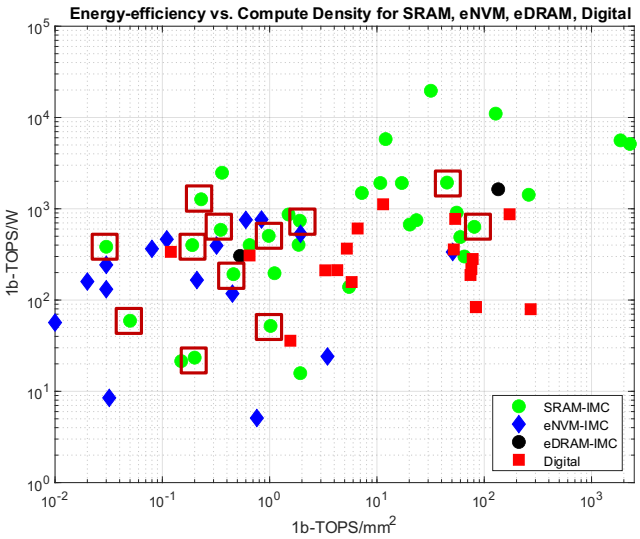

Fig. 5. 1b-TOPS/W vs. 1b-TOPS/mm^2 categorized with respect to SRAM-, eNVM-, eDRAM-based IMCs and digital accelerators. The centroids (1b-TOPS/mm^2, 1b-TOPS/W) for each cluster are at: SRAM (146.2, 2011.1), eNVM (3.7, 282.5) and digital (54.6, 349.7).

Energy-efficiency vs. Compute Density: Figure 5 shows that SRAM-based IMCs achieve both the highest bank-level energy-efficiency and the highest compute density compared to eNVM-, eDRAM-based IMCs, and digital accelerators. The best (average) reported 1b-TOPS/W for SRAM-based IMCs [8] is about $17\times$ ($5.7\times$) higher than that for digital accelerators [47] and about $26\times$ ($7\times$) higher than that for eNVM-based IMCs [31]. However, as Fig. 5 shows, the efficiency gap between the best reported metric for an SRAM-based IMC processor (16 bank, 16nm) [5] and that of a comparable digital accelerator [57] (16 PE, 16nm) reduces to $3\times$ ($1.7\times$ when comparing to [47]). This clearly indicates that the cost of scaling-up IMC designs is quite high today. Though both SRAM-based IMCs and digital accelerators benefit from technology scaling (see Fig. 1), their slopes may be different. Therefore, it will be interesting to see if this efficiency gap widens or reduces in the future.

Figure 5 also indicates that the best (average) reported compute density for SRAM IMCs is about $8.4\times$ [14] ($2.7\times$) greater than that of digital accelerators [55] and about $45\times$ [32] ($39.5\times$) when compared to eNVMs. Furthermore, eNVM-based IMCs and digital accelerators seem to be roughly on par in terms of their energy-efficiencies. However, eNVM-based IMCs have roughly an order-of-magnitude *lower* compute densities compared to digital accelerators. This is quite surprising since one of the much-touted advantages of eNVM devices is their density. One exception is the PCM crossbar [32] utilizing multi-bit weight storage with analog compensation techniques to achieve accurate MVM operation in presence of analog non-idealities. Thus, today, eNVM-based IMCs lag *both* SRAM-based IMCs and digital accelerators in *both* energy-efficiency and compute densities.

Energy-efficiency vs. Throughput: The trends in Fig. 5 can be explained by comparing the energy-efficiency with throughput achieved by the three types of architectures. Figure 6 shows that digital accelerators achieve the highest throughput. The best (average) reported throughput for digital accelerators is $53\times$ [52] ($79\times$) higher than that of SRAM-based IMC [8], and roughly about $363\times$ ($292\times$) compared to that of eNVM-based IMCs [32], and that too at compute densities (Fig. 5) comparable to that of SRAM-based IMCs.

Not shown in Fig. 6 is the role of compute accuracy in determining the throughput. Digital accelerators can achieve arbitrarily high throughput via scale-up without compromising on their accuracy. IMCs, be it SRAM or eNVM-based ones, need to work much harder to preserve accuracy during scale-up. Specifically, in case of eNVM-based IMCs, the throughput is limited by its lower array utilization which is required to preserve its accuracy due to the presence of analog non-idealities, e.g., the eNVMs in [38][42] which have higher 1b-TOPS with multi-bit input/weight achieve a low accuracy of ~ 90%-92% on MNIST. Compute accuracy in presence of analog non-idealities such as wire parasitics and conductance mismatch is negatively impacted. Additionally, the area overhead of

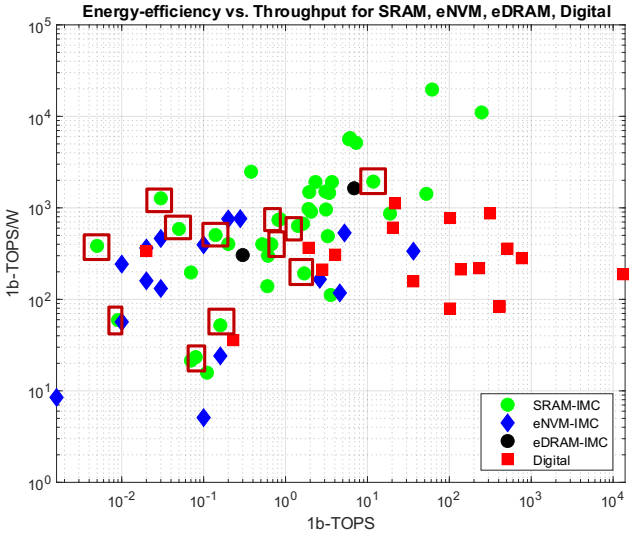

Fig. 6. 1b-TOPS/W vs. 1b-TOPS categorized with respect to SRAM-, eNVM-, eDRAM-based IMCs and digital accelerators. The centroids (1b-TOPS, 1b-TOPS/W) for each cluster are at: SRAM (11.5, 1881.7), eNVM (3.1, 282.5) and digital (904, 349.7).

peripherals (DAC, sensing, and ADC) tends to be large, e.g., 90% of the total bank area in [41], due to the need for high-sensitivity CMOS read-out circuitry required to maintain the fidelity of analog computations. This further reduces their 1b-TOPS/mm^2 in Fig. 5.

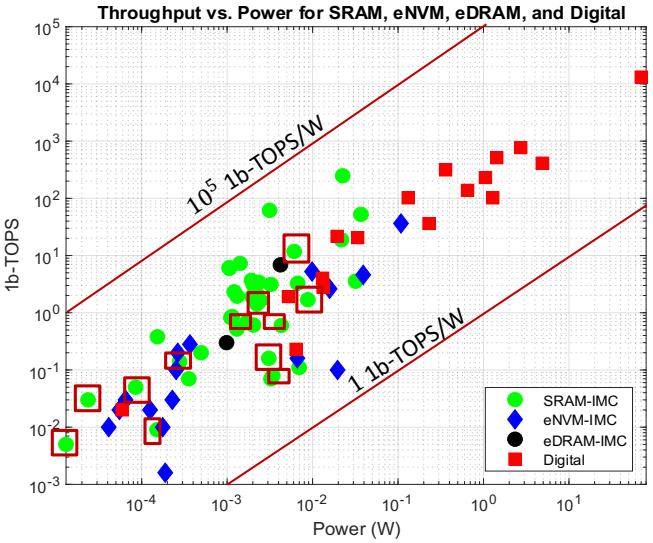

Fig. 7. 1b-TOPS vs. power (W) categorized with respect to SRAM-, eNVM-, eDRAM-based IMCs and digital accelerators. The centroids (mW, 1b-TOPS) for each cluster are at: SRAM (4.9, 11.5), eNVM (12.6, 3.1) and digital (4800,904).

Throughput vs. Power: Finally, it is instructive to compare the throughput achieved at a specific power consumption. Figure 7 shows that all architectures lie between the 10^5 1b-TOPS/W and 1 1b-TOPS/W iso-efficiency lines with throughput increasing in proportion to power consumption. Digital accelerators lie towards the upper end of the throughput and power axes, followed by SRAM-based IMCs and then eNVM-based ones.

Embedded DRAM (eDRAM)-based IMCs have recently appeared [43][44]. These employ charge redistribution methods for computing dot products. The design in [43] repurposes the 1T1C eDRAM array for IMC has low array utilization (lower 1b-TOPS) to avoid destructive reads whereas [44] proposes a 3T dynamic analog RAM using multi-bit weights, and adaptive ADC skipping techniques to achieve higher TOPS while preserving the accuracy. These emerging methods seem to be promising since they achieve energy-efficiencies and compute densities comparable to mainstream SRAM-based IMCs in spite of being implemented in 65nm CMOS.

FeFETs-based IMCs [33][34] are being proposed but we do not include them in our charts since the reported metrics are based on simulations. A complete FeFET-based IMC has yet to be published.

6. Measuring IMC Accuracy

As mentioned in Section 1, IMCs being decision-making machines exhibit an inherent trade-off between energy-efficiency, latency, and accuracy. Therefore, a notable omission in Figs. 2-4 is the accuracy achieved at a specified level of 1b-TOPS/W or 1b-TOPS. This omission is a glaring weakness in IMC design today since no papers measure or report the compute SNR SNR_{col} of an ADC column (see Fig. 1) alongside energy-efficiency in their comparison tables. Unlike digital accelerators that can realize an arbitrary level of accuracy simply by scaling up precision, IMCs trade-off their compute SNR for energy-efficiency gains. Thus, it is critically important for IMC works to report the 3-tuple (energy-efficiency, throughput, accuracy).

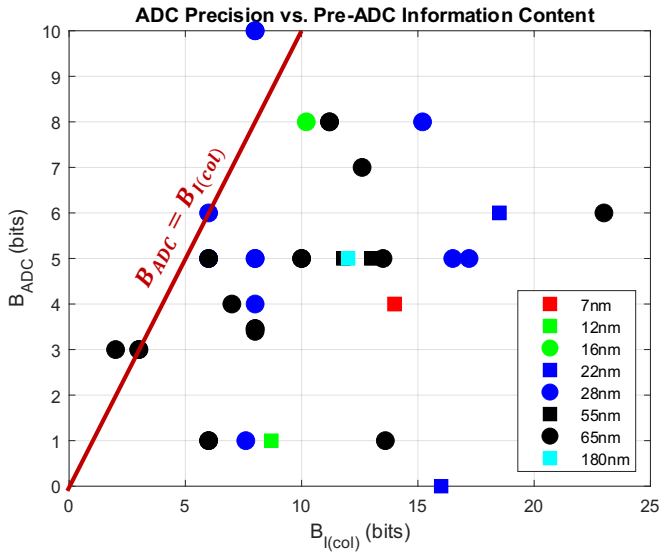

Fig. 8. B_{ADC} vs. $B_{I(col)}$ for SRAM IMCs categorized with respect to technology node.

IMC's accuracy needs to be reported at both the bank-level (compute accuracy SNR_{col}) and at the network level (e.g., probability of correct decision P_d).

Compute Accuracy (SNR_{col}): In the following, we assume a fixed-point digital computation as the baseline, i.e., the ideal output $y_o = \mathbf{w}^{\mathrm{T}}\mathbf{x}$ of a $B_X \times B_W$-bit N-dimensional dot-product computation with an output precision of $B_X + B_W + \log_2 N$, i.e., $B_{I(col)}$. Thus, IMCs can be compared directly with digital accelerators, e.g., with an 4b x 4b N-dimensional digital dot-product computation, both in terms of energy-efficiency, throughput, and accuracy.

The compute SNR SNR_{col} of an ADC column is given by [58]:

$$SNR_{col} = [1/SNR_A + 1/SNR_{ADC}\,)]^{-1} = \sigma_{y_o}^2/\sigma_e^2 \qquad (2)$$

where SNR_A is the *analog (pre-ADC)* SNR representing the *true upper bound* on SNR_{col} and $e = y_o - y_{imc}$ is the error between the IMC (ADC) output y_{imc} and the ideal digital output y_o. The analog SNR SNR_A, which includes all analog non-idealities as seen at the ADC input, will drop as a function of the dot-product dimension N, B_X, and B_W since the increased output dynamic range still needs to fit within headroom constraints. This is the intrinsic SNR_{col} vs. energy-efficiency trade-off in IMCs. The term SNR_{ADC} in (1) is the ADC signal-to-noise ratio.

To minimize the column ADC's energy overhead, one needs to design ADCs with minimum required SNR_{ADC}, i.e., fully exploit the SNR vs. energy trade-off exhibited by ADCs. The ADC's SNR can be tuned by adjusting the input clipping range [58][65] and its precision B_{ADC}, among others, to account for the fact that for large N, the ADC input signal has a distribution with a small variance relative to its dynamic range. This eventually results in ADCs being noise-limited [66]. Ideally, the SNR_{ADC} needs to be sufficiently greater than SNR_A so that $SNR_{col} \rightarrow SNR_A$, e.g., if $SNR_{ADC} > SNR_A + 9$dB, then SNR_{col} lies within 0.5dB of SNR_A. Since measuring SNR_A directly is difficult, IMC designers can measure SNR_{col} directly by comparing the ADC column output y_{imc} with the ideal digital computation y_o per (2) and increase SNR_{ADC} until SNR_{col} saturates.

A valid question to ask is: *what is a good SNR_{col} to achieve*? Prior work [59][60] indicates that the signal-to-quantization-noise ratio (SQNR) of fixed-point dot-product computations needs to fall in the range $[10\,\mathrm{dB}, 40\,\mathrm{dB}]$ in order for the inference accuracy of a fixed-point implementation to be within 1% of the corresponding floating-point implementation for popular DNNs (AlexNet, VGG-9, VGG-16, ResNet-18) deployed on the ImageNet and CIFAR-10 datasets. Therefore, IMC designs need to ensure that $SNR_{col} \in [10\,\mathrm{dB}, 40\,\mathrm{dB}]$ to be assured that network accuracy will be close to that of floating-point implementations of the same network.

Currently, the approach described above is not followed and SNR_{col} is typically not reported in IMC publications ([10][18] being rare exceptions). Therefore, as a proxy for SNR_{col}, we compare B_{ADC} with $B_{I(col)}$ in Fig. 8, where we find that $B_{ADC} < B_{I(col)}$ and in fact in many cases $B_{ADC} \ll B_{I(col)}$. Note that in an ideal noiseless scenario, i.e., $SNR_A \rightarrow \infty$, one expects to see $SNR_{col} \rightarrow \infty$ when $B_{ADC} \geq B_{I(col)}$. In practice, SNR_A is finite and $B_{ADC} \ll B_{I(col)}$ indicating that bank-level compute accuracy SNR_{col} is quite limited in most cases and IMC designers rely on the inherent error-tolerance of machine learning algorithms to achieve the desired level of network accuracy.

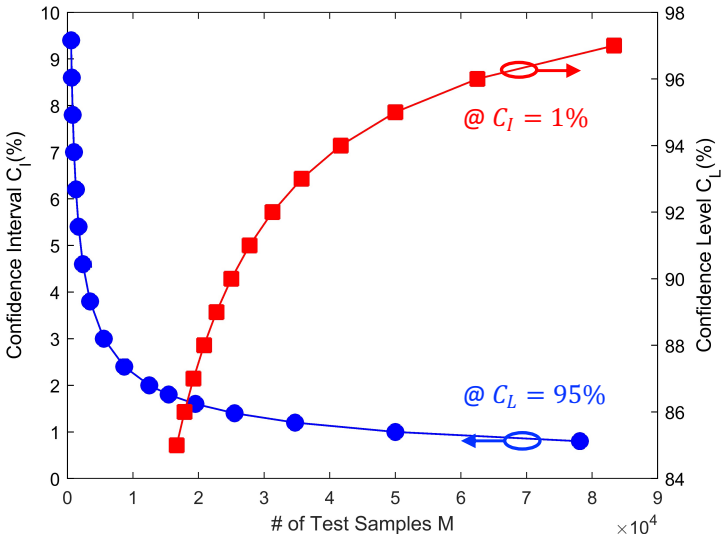

Fig. 9. Confidence interval (C_I) and confidence level (C_L) vs. number of test samples M.

Network Accuracy (P_d): IMC papers typically report accuracies for specific datasets, e.g., VGG-16 on CIFAR-10. However, it is not clear whether the entire network was mapped onto the IMC or just a few layers with the rest implemented in software. A more serious issue is that IMC papers do not report the number of input samples used during chip testing to obtain network accuracy. The reason being in a limited sample scenario, the accuracy P_d (probability of correct decision) itself is a random variable with a mean and variance. For this variance to be small, it is important that two conditions be satisfied: 1) the input vectors be randomly sampled from the dataset, and 2) the number of samples be sufficiently large. If any of these two conditions are violated, then the reported accuracy will be either biased or have a large variance.

For example, it is well-known that some inputs are easy to classify compared to others since their feature vectors lie far from the decision boundary of the network. If such 'friendly' inputs are cherry-picked for testing an IMC then its reported accuracy will be artificially inflated. For instance, one can operate the IMC at a low compute SNR to achieve high energy-efficiency. Then choose 'friendly' inputs to achieve high accuracy in the presence of a perturbed decision boundary caused by the IMC's low compute SNR.

A second issue is the large variance in P_d if the number of test samples are small. IMCs report one accuracy number obtained ostensibly by testing over what is most likely a small subset of the dataset. Thus, the reported accuracy is one sample of a potentially large variance random variable. For this reason, IMC works need to report the confidence interval ($0 < C_I < 1$) and confidence level ($0 < C_L < 1$) when reporting P_d, e.g., "we achieve an accuracy of $90\% \pm 1\%$ at a confidence level of $> 95\%$".

Assuming the misclassification errors are independent Bernoulli coin flips, one can show via the Chebyshev inequality that the minimum number of test samples M needed to achieve a detection accuracy $P_d \pm C_I$ with a confidence level of C_L is given by:

$$M > 1/[4(1 - C_L)C_I^2\,] \quad (4)$$

Equation (3) indicates that C_I and C_L trade-off with each other when the number of test samples M is fixed. Figure 9 shows that we need to test at least 50,000 test images to claim that the true accuracy lies within $\pm 1\%$ (confidence interval) of the reported accuracy with a confidence level of $> 95\%$. We suggest fixing the confidence level to a preset value, e.g., 95% or higher, and then reporting the measured accuracy along with the associated confidence interval. If very few test vectors are chosen, then the confidence interval will be large, e.g., $C_I = 3\%$ if only 5500 test images are chosen.

Finally, it is highly recommended that as a community we establish a benchmark set of pre-trained fixed-point networks (*IMC test networks*) with an associated subset of pre-selected 'hard' inputs (*IMC test set*) per dataset. The size of the IMC test set can be chosen to meet commonly agreed upon specifications on C_L and C_I. Such a step will ensure uniformity and confidence in the reported IMC accuracies.

7. Discussion and Summary

This paper has presented a hierarchical view of IMCs that employs an ADC column as the basic unit and associated metrics. Motivated by this view, an IMC benchmarking methodology was presented and employed to analyze 40+ IMC IC works that appeared in CICC, VLSI, ISSCC (and ESSCIRC) since 2018. Comparisons were made between SRAM-, eNVM-based and recent digital accelerators. Key findings of our study are summarized below:

1) Bank-level SRAM-based IMC design methods are mature and have demonstrated clear wins in terms of energy-efficiency ($17\times$ in 1b-TOPS/W) and compute density ($8\times$ in 1b-TOPS/mm^2) over digital accelerators when the comparisons are made at the bank-level. However, the bank-level gap in energy-efficiency reduces to $< 2\times$ when comparing IMC processors with digital accelerators.
2) Despite their analog-heavy computation and contrary to conventional wisdom, SRAM-based IMCs benefit from technology scaling, i.e., their energy-efficiency and compute density both improve in advanced nodes. 6T+ SRAM bitcells tend to enhance energy-efficiency in the same technology node but at a cost to compute density.
3) eNVM-based IMCs lag behind both SRAM-based IMCs and digital accelerators in terms of energy-efficiency and compute density due to the challenges associated with compute accuracy that arise when enhancing row-parallelism.
4) Compute accuracy of IMCs is a neglected issue. Energy-efficiency, compute density and throughput need to be measured at a pre-specified accuracy. Metrics such as compute SNR SNR_{col} need to be quantified and employed to predict accuracy of networks being mapped.
5) Testing of IMCs to obtain network accuracy needs to report confidence levels and confidence intervals to be rigorous. We suggest fixing the confidence level to a preset value, e.g., 95% or higher, and then reporting the measured accuracy along with the associated confidence interval.
6) An IMC benchmark suite is much needed for consistency and to evaluate progress in the field. Such a suite will consist of a set of pre-trained fixed-point networks (*IMC test networks*) with an associated subset of pre-selected 'hard' inputs (*IMC test set*) per dataset. The size of the IMC test set can be chosen to meet commonly agreed upon specifications on the confidence level, e.g., > 95%, and confidence interval, e.g., < 1%. Such a step will ensure uniformity and confidence in the reported IMC accuracies.

A criticism of this paper can be that the causes underlying the data-driven trends presented here has not been fully investigated or elaborated upon. Doing so requires a deeper study of various IMC compute models and circuit methods (see [61][62]) which is beyond the scope of this paper.

Despite much progress since the publication of [1], the area of IMC design remains full of potential and numerous opportunities. Most of these will require collaborations between researchers in relevant areas. Future opportunities for IMC research include [*area required to collaborate are indicated in square brackets*]:

1) Fully understanding and quantifying the fundamental efficiency vs. accuracy trade-offs in various IMCs (see [63][61]). This includes minimizing ADC energy by tailoring its SNR to that of the column computation (see [58][65][66]). [*circuits, statistical analysis*]
2) Developing algorithmic approaches such as statistical error compensation (SEC) [27][64] to enhance the accuracy of IMCs beyond what is possible via purely circuit optimization. [*circuits, communications*]
3) Designing IMCs on emerging devices such as MRAM, FeFET and others. Specifically, tailoring device properties to enable massive row-parallelism with minimal impact on accuracy. [*semiconductor devices, circuits*]
4) Developing methodologies to scale-up IMCs without losing their bank-level efficiencies while meeting end-to-end network/application-level accuracy requirements. [*circuits, microarchitecture, statistical analysis*]
5) Developing efficient application-to-IMC architecture mapping methods (see [67]) to fully exploit the inherent parallelism in multi-bank IMC processors while meeting accuracy requirements. [*compilers, microarchitecture*]
6) Exploring the design of hybrid IMC and spatial architectures that leverage their respective strengths to effectively exploit parallelism, reuse, and sparsity opportunities afforded by AI workloads. [*circuits, microarchitecture, machine learning*]
7) Developing machine learning algorithms intrinsically tailored for IMCs. [*machine learning, microarchitecture*]
8) Exploring applications beyond AI where high precision (> 12b) computation is required, e.g., signal processing, communications, security, scientific computing, and others. [*applications*, *algorithms*]

While the full-stack nature of IMCs provides numerous opportunities to devices researchers, analog/mixed-signal designers, architects, system, and algorithm designers, it also presents a formidable challenge in quantifying progress in the field. It is hoped that this paper brings together the IMC design community to engage in a robust discussion on the topic of establishing rigorous benchmarking and evaluation strategies for IMCs so as to ensure its continued growth moving forward.

8. Acknowledgements

The authors gratefully acknowledge many fruitful discussions with Mingu Kang, Naveen Verma, Boris Murmann, Pavan Hanumolu, and Ali Keshavarzi. We thank Han-Mo Ou, Hyungyo Kim, Hassan Dbouk, and Shuo Li for assisting in the IC data collection effort.

This work was supported by DARPA via the ERI FRANC program, and by the Semiconductor Research Corporation (SRC) via the Center on Brain-Inspired Computing (C-BRIC).

References:

[1] M. Kang et al., “An energy-efficient VLSI architecture for pattern recognition via deep embedding of computation in SRAM”, ICASSP, May 2014.

[2] G. W. Burr, et al., "Fair and comprehensive benchmarking of machine learning processing chips", IEEE Design & Test, 2021.

[3] J.-W. Su et al., “A 28nm 384kb 6T-SRAM computation-in-memory macro with 8b precision for AI edge chips”, ISSCC, 2021.

[4] J. Yue et al., “A 2.75-to-75.9TOPS/W computing-in-memory NN processor supporting set-associate block-wise zero skipping and ping-pong CIM with simultaneous computation and weight updating”, ISSCC, 2021.

[5] H. Jia et al., “A programmable neural-network inference accelerator based on scalable in-memory computing”, ISSCC, 2021.

[6] R. Guo et al., “A 5.99-to-691.1TOPS/W tensor-train in-memory-computing processor using bit-level-sparsity-based optimization and variable-precision quantization”, ISSCC, 2021.

[7] Y.-D. Chih et al., “A 89 TOPS/W and 16.3 TOPS/mm^2 all digital SRAM-based full precision compute-in-memory in 22nm for machine-learning edge applications”, ISSCC, 2021.

[8] I. A. Papistas et al., “A 22 nm, 1540 TOP/s/W, 12.1 TOP/s/mm^2 in-memory analog matrix-vector-multiplier for DNN acceleration”, CICC, 2021.

[9] R. A. Rasul et al., “A 128x128 SRAM macro with embedded matrix-vector multiplication exploiting passive gain via MOS capacitor for machine learning application”, CICC, 2021.

[10] J. Lee et al., “Fully row/column-parallel in-memory computing SRAM macro employing capacitor-based mixed-signal computation with 5-b inputs”, VLSI, 2021.

[11] S. Yin et al., “PIMCA: A 3.4-Mb programmable in-memory computing accelerator in 28nm for on-chip DNN inference”, VLSI, 2021.

[12] R. Guo et al., “A 6.54-to-26.03 TOPS/W computing-in-memory RNN processor using input similarity optimization and attention-based context-breaking with output speculation”, VLSI, 2021.

[13] J.-W. Su et al., “A 28nm 64Kb inference-training two-way transpose multibit 6T SRAM compute-in-memory macro for AI edge chips”, ISSCC, 2020.

[14] Q. Dong et al., “A 351TOPS/W and 372.4GOPS compute-in-memory SRAM macro in 7nm FinFET CMOS for machine-learning applications”, ISSCC, 2020.

[15] J. Yue et al., ISSCC, “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”, 2020.

[16] X. Si et al., “A 28nm 64Kb 6T SRAM computing-in-memory macro with 8b MAC operation for AI edge chips”, ISSCC, 2020.

[17] C. Yu et al., “A 16K current-based 8T SRAM compute-in-memory macro with decoupled read/write and 1-5bit column ADC”, CICC, 2020.

[18] H. Jia et al., “A programmable heterogeneous microprocessor based on bit-scalable in-memory computing”, JSSC, 2020.

[19] Z. Jiang et al., “C3SRAM: an in-memory-computing SRAM macro based on robust capacitive coupling computing mechanism”, JSSC, 2020.

[20] X. Si et al., “A twin-8T SRAM computation-in-memory macro for multiple-bit CNN-based machine learning”, ISSCC, 2019.

[21] J. Yang et al., “Sandwich-RAM: an energy-efficient in-memory BWN architecture with pulse-width modulation”, ISSCC, 2019.

[22] T. Chen et al., “An SRAM-based accelerator for solving partial differential equations”, CICC, 2019.

[23] R. Guo et al., “A 5.1pJ/Neuron 127.3us/Inference RNN-based speech recognition processor using 16 computing-in-memory SRAM macros in 65nm CMOS”, VLSI, 2019.

[24] J. Kim et al., “Area-efficient and variation-tolerant in-memory BNN computing using 6T SRAM array”, VLSI, 2019.

[25] S. Okumura et al., “A ternary based bit scalable, 8.80 TOPS/W CNN accelerator with many-core processing-in-memory architecture with 896K synapses/mm^2”, VLSI, 2019.

[26] W.-S. Khwa et al., “A 65nm 4Kb algorithm-dependent computing-in-memory SRAM unit-macro with 2.3ns and 55.8TOPS/W fully parallel product-sum operation for binary DNN edge processors”, ISSCC, 2018.

[27] S. K. Gonugondla et al., “A 42pJ/decision 3.12TOPS/W robust in-memory machine learning classifier with on-chip training”, ISSCC, 2018.

[28] A. Biswas and A. P. Chandrakasan, “Conv-RAM: an energy-efficient SRAM with embedded convolution computation for low-power CNN-based machine learning applications”, ISSCC, 2018.

[29] H. Valavi et al., “A mixed-signal binarized convolutional-neural-network accelerator integrating dense weight storage and multiplication for reduced data movement”, VLSI, 2018.

[30] Z. Jiang et al., “XNOR-SRAM: in-memory computing SRAM macro for binary/ternary deep neural networks”, VLSI, 2018.

[31] C.-X. Xue et al., “A 22nm 4Mb 8b-precision ReRAM computing-in-memory macro with 11.91-195.7 TOPS/W for tiny AI edge devices”, ISSCC, 2021.

[32] R. Khaddam-Aljameh et al., “HERMES core – a 14nm CMOS and PCM-based in-memory compute core using an array of 300ps/LSB linearized CCO-based ADCs and local digital processing”, VLSI, 2021.

[33] C. Matsui et al., “Energy-efficient reliable HZO FeFET computation-in-memory with local multiply & global accumulate array for source-follower & charge-sharing voltage sensing”, VLSI, 2021.

[34] D. Saito et al., “Analog in-memory computing in FeFET-based 1T1R array for edge AI applications”, VLSI, 2021.

[35] W. Li et al., “Secure-RRAM: a 40nm 16kb compute-in-memory macro with reconfigurability, sparsity control, and embedded security”, CICC, 2021.

[36] J.-H. Yoon et al., “A 40-nm, 64-Kb, 56.67 TOPS/W voltage-sensing computing-in-memory/digital RRAM macro supporting iterative write with verification and online read-disturb detection”, JSSC, 2021.

[37] P. Deaville et al., “A maximally row-parallel MRAM in-memory-computing macro addressing readout circuit sensitivity and area”, ESSCRRIC, 2021.

[38] Q. Liu et al., “A fully integrated analog ReRAM based 78.4TOPS/W compute-in-memory chip with fully parallel MAC computing”, ISSCC, 2020.

[39] C.-X. Xue et al., “A 22nm 2Mb ReRAM compute-in-memory macro with 121-28TOPS/W for multibit MAC computing for tiny AI edge devices”, ISSCC, 2020.

[40] C.-X. Xue et al., “Embedded 1-Mb ReRAM-based computing-in-memory macro with multibit input and weight for CNN-based AI edge processors”, JSSC, 2020.

[41] S. Yin et al., “High-throughput in-memory computing for binary deep neural networks with monolithically integrated RRAM and 90-nm CMOS”, TED, 2020.

[42] R. Mochida et al., “A 4M synapses integrated analog ReRAM based 66.5 TOPS/W neural-network processor with cell current controlled writing and flexible network architecture”, VLSI, 2018.

[43] S. Xie et al., “eDRAM-CIM: compute-in-memory design with reconfigurable embedded-dynamic-memory array realizing adaptive data converters and charge-domain computing”, ISSCC, 2021.

[44] Z. Chen et al., “A 65nm 3T dynamic analog RAM-based computing-in- memory macro and CNN accelerator with retention enhancement, adaptive analog sparsity and 44TOPS/W system energy efficiency”, ISSCC, 2021.

[45] A. Agarwal et al., “A 7nm 4-Core AI chip with 25.6TFLOPS hybrid FP8 training, 102.4TOPS INT4 inference and workload-aware throttling”, ISSCC, 2021.

[46] J.-S. Park et al., “A 6K-MAC feature-map-sparsity-aware neural processing unit in 5nm flagship mobile SoC”, ISSCC, 2021.

[47] H. Mo et al., “A 28nm 12.1TOPS/W dual-mode CNN processor using effective-weight-based convolution and error-compensation-based prediction”, ISSCC, 2021.

[48] J. Park et al., “A 40nm 4.81TFLOPS/W 8b floating-point training processor for non-sparse neural networks using shared exponent bias and 24-way fused multiply-add tree”, ISSCC, 2021.

[49] Z. Tan et al., “A 400MHz NPU with 7.8TOPS2/W high-performance- guaranteed efficiency in 55nm for multi-mode pruning and diverse quantization using pattern-kernel encoding and reconfigurable MAC”, CICC, 2021.

[50] J. Lee et al., “A 13.7 TFLOPS/W floating-point DNN processor using heterogeneous computing architecture with exponent-computing-in-memory”, VLSI, 2021.

[51] S. Kang et al., “GANPU: A 135TFLOPS/W multi-DNN training processor for GANs with speculative dual-sparsity exploitation”, ISSCC, 2020.

[52] Y. Jiao et al., “A 12nm programmable convolution-efficient neural-processing-unit chip achieving 825TOPS”, ISSCC, 2020.

[53] C.-H. Lin et al., “A 3.4-to-13.3TOPS/W 3.6TOPS dual-core deep-learning accelerator for versatile AI applications in 7nm 5G smartphone SoC”, ISSCC, 2020.

[54] J. Oh et al., “A 3.0 TFLOPS 0.62V scalable processor core for high compute utilization AI training and inference”, VLSI, 2020.

[55] J.-H. Kim et al., “Z-PIM: an energy-efficient sparsity aware processing-in-memory architecture with fully-variable weight precision”, VLSI, 2020.

[56] J. Wang et al., “A compute SRAM with bit-serial integer/floating-point operations for programmable in-memory vector acceleration”, ISSCC, 2019.

[57] B. Zimmer et al., “A 0.11 pJ/Op, 0.32-128 TOPS, scalable multi-chip-module-based deep neural network accelerator with ground-reference signaling in 16nm”, VLSI, 2019.

[58] S. K. Gonugondla et al., “Fundamental limits on the precision of in-memory architectures”, ICCAD, 2020.

[59] C. Sakr et al., “Analytical guarantees on numerical precision of deep neural networks”, ICML, 2017.

[60] C. Sakr et al., “An analytical method to determine minimum per-layer precision of deep neural networks”, ICASSP, 2018.

[61] S. K. Gonugondla et al., “Fundamental limits on energy-delay-accuracy of in-memory architectures in inference applications”, IEEE Transactions on CAD, 2021 (to appear).

[62] R. Sehgal and J. P. Kulkarni, “Trends in analog and digital intensive compute-in-SRAM designs”, AICAS, 2021.

[63] M. Kang et al., “Deep in-memory architectures for machine learning–accuracy versus efficiency trade-offs”, IEEE Transactions on Circuits and Systems I: Regular Papers, 2020.

[64] N. R. Shanbhag et al., “Shannon-inspired statistical computing for the nanoscale era”, Proceedings of the IEEE, 2018.

[65] C. Sakr and N. R. Shanbhag, “Signal processing methods to enhance the energy efficiency of in-memory computing architectures,” IEEE Transactions on Signal Processing, 2021 (to appear).

[66] B. Murmann, “Mixed-signal computing for deep neural network inference,” IEEE Transactions on VLSI, January 2021.

[67] M. Kang, et al., “An energy-efficient programmable mixed-signal accelerator for machine learning algorithms,” IEEE MICRO, 2019.