

© 2024 Saion Kumar Roy

ENERGY-EFFICIENT RESISTIVE
IN-MEMORY COMPUTING ARCHITECTURES

BY

SAION KUMAR ROY

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois Urbana-Champaign, 2024

Urbana, Illinois

Doctoral Committee:

Professor Naresh Shanbhag, Chair
Professor Pavan Hanumolu
Assistant Professor Saugata Ghose
Associate Professor Jae-sun Seo, Cornell University

ABSTRACT

Embedded non-volatile memory (eNVM)-based in-memory computing (IMC) architectures have garnered significant interest due to their high memory density and non-volatility in realizing efficient implementations of deep neural networks (DNNs). Their high density enables on-chip storage of DNN models, thereby reducing the energy and latency costs associated with data movement. Additionally, their non-volatile nature supports efficient activity-driven duty-cycled operations. However, these benefits have not been realized in real-life eNVM-based IMC integrated chip (IC) prototypes, which significantly lag behind SRAM-based IMCs and digital accelerators in terms of energy efficiency and compute density. This lag is primarily due to the presence of large read-out circuits required to achieve high bank-level accuracy. Moreover, the fundamental energy-accuracy trade-offs of eNVM-based IMCs are not well understood, as most research focuses on achieving high network-level accuracy, which masks bank-level accuracy due to the error tolerance of DNNs and the use of noise-aware training. This dissertation aims to evaluate the intrinsic energy-accuracy trade-offs of eNVM-based IMCs and develop techniques to enhance bank-level accuracy.

First, we introduce a comprehensive benchmarking methodology for IMCs that quantifies the significant lag in energy efficiency and compute density of eNVM-based IMCs compared to SRAM-based IMCs and digital accelerators. Second, to understand this lag, which arises from the low bank-level accuracy of resistive IMCs, we use the compute signal-to-noise-plus-distortion ratio (SNDR) to measure accuracy and identify its dependence on device, circuit, and architectural parameters. This is achieved by developing a chip-verified behavioral model, which reveals that the maximum achievable SNDR is limited by pre-analog-to-digital-converter (ADC) analog non-idealities. Additionally, we find that increasing the device-level conductance contrast improves the maximum achievable SNDR only up to a certain point, beyond

which it saturates. Third, building on insights from the behavioral model, we propose two bank-level accuracy-boosting techniques at the circuit and algorithmic levels, demonstrating a compute SNDR improvement from 2.7 to 6 dB in a 22 nm MRAM-based IMC prototype. This boost is traded off to achieve a $5\times$ reduction in energy per 1b-operation, with modest energy and area overhead. Furthermore, our work shows that the bank-level accuracy boost correlates well with network-level accuracy without resorting to noise-aware training.

Lastly, we explore the security vulnerabilities of eNVM-based IMC architectures against model extraction attacks (MEAs). These attacks can reconstruct private training data from trained model parameters, leaking sensitive user information. While the presence of analog noise in eNVM-based IMC computation might suggest intrinsic robustness to MEAs, we demonstrate that this is not the case. Our results underscore the critical importance of investigating the security vulnerabilities of IMCs in general and eNVM-based IMCs in particular.

To Kuhu and my parents for their love and support.

ACKNOWLEDGMENTS

*The cost of a thing is the amount of what I will call life which is required to be exchanged for it, immediately or in the long run.*¹

Embarking on a Ph.D. is a monumental task and a life-changing experience that transforms an individual into a critical thinker, problem solver, and excellent communicator. Six years ago, I began my Ph.D. under the guidance of Prof. Naresh Shanbhag with the aspiration of mastering these skills. As I finalize my dissertation, I am proud to say that I have undergone a profound transformation, acquiring these abilities through an immensely challenging yet rewarding journey.

Words cannot adequately express my gratitude to Prof. Shanbhag for offering me this opportunity and for his unwavering guidance and support throughout the process. His mentorship has been instrumental in honing my thinking, writing, and communication skills, always pushing me to strive for excellence. I deeply admire and appreciate Prof. Shanbhag's dedication to my growth and development.

I wish to extend my sincere gratitude to Prof. Pavan Hanumolu, Prof. Saugata Ghose, and Prof. Jae-sun Seo for serving on my doctoral committee and for their invaluable insights and comments. I am especially thankful to Prof. Pavan Hanumolu for our fruitful interactions and his exceptionally perceptive feedback on circuit design and analysis.

I would like to express my gratitude to Dr. Mike Burkland from Raytheon, Dr. Steven Soss from Global Foundries (GF), Peter Deaville, Bonan Zhang, Prof. Naveen Verma from Princeton University, Han-Mo Ou, Mostafa Gamal Ahmed, Prof. Hanumolu, and Prof. Shanbhag from UIUC for their collaboration DARPA FRANC project. I also want to thank Dr. Avishek Biswas, Dr. Hetul Sanghvi, and Dr. Mahesh Mehendale from Texas Instruments for

¹A quote from *Walden* by Henry David Thoreau

providing me with the internship opportunity in Summer 2021, which was a wonderful learning experience.

My interactions with the outstanding researchers in Prof. Shanbhag’s lab have been pivotal to my progress. Their inspiring dedication to excellence and their thoughtful, constructive feedback have continually pushed me to improve. I am deeply grateful to Sujan Gonugondla, Charbel Sakr, Ameya Patil, Hassan Dbouk, Haocheng Hua, Henfei Geng, Ji Shi, Michael Tuttle, Kuk-Hwan Kim, Shuo Li, Mostafa Gamal Ahmed, Jeff Teng, Soonha Hwang, Han-Mo Ou, Hyungyo Kim, Kaining Zhuo, Vignesh Sundaresha, Mihir Kavishwar, Prithvi Nathan, Omar Bekdache, Jeter Hwang, Nathan Chiang, and Ryan Fu for making the most challenging phase of my life both memorable and fulfilling.

The research presented in this dissertation was funded in part by C-BRIC under JUMP and COCOSYS under JUMP 2.0, supported by SRC and DARPA, as well as DARPA’s FRANC program. I feel fortunate to have been part of the SRC community as a graduate student; interacting with faculty members and students from various universities involved in these programs has been a formative experience. I am deeply grateful to ECE Illinois for providing me with the opportunity to pursue a Ph.D. degree and to the Coordinated Science Laboratory (CSL) for hosting me and my work.

I would like to express my deepest gratitude to my friends, whose unwavering support has been instrumental in boosting my morale during the toughest times. Special thanks to Soumajit Dutta, Anamitra Chaudhuri, Kaustav Chakraborty, Sarthak Saha, Arpan Dasgupta, Alex Spencer, Ankit Sharma, Sagar Kumashi, and Souvik Roy. I am also grateful to Saransh Sharma and Saurav Maji for their unending banter and for motivating me to pursue the quest for excellence. I am profoundly indebted to my parents, Supriyo Roy and Sujata Roy, whose sacrifices have made it possible for me to embark on and complete this Ph.D. journey. Finally, and most importantly, I am deeply thankful to Kuheli Pratihari for standing by my side throughout this relentless journey, helping me stay strong during the lows and celebrate the highs, always pushing toward excellence.

TABLE OF CONTENTS

CHAPTER 1	INTRODUCTION	1
1.1	Accuracy of eNVM-based IMCs	3
1.2	Thesis Organization	6
CHAPTER 2	BENCHMARKING IN-MEMORY COMPUTING ARCHITECTURES	9
2.1	A Compositional View of IMCs	11
2.2	IMC Data Analysis Strategy	14
2.3	SRAM-based IMC Trends	15
2.4	Comparison Across Architectures	17
2.5	Measuring IMC Accuracy	23
2.6	Discussion and Summary	29
CHAPTER 3	ENERGY-ACCURACY TRADE-OFFS IN RESISTIVE IN-MEMORY COMPUTING ARCHITECTURES	32
3.1	Background and Preliminaries	34
3.2	Non-ideal Behavior in Resistive IMC Architecture	37
3.3	Simulation Results	45
3.4	Accuracy Limits of Resistive Crossbars	56
3.5	Discussion and Conclusion	66
3.6	Addendum: Circuit Derivations for Non-idealities in Parallel-bar Architectures	67
CHAPTER 4	A COMPUTE SNDR-BOOSTED MRAM BASED IMC PROTOTYPE IN 22 nm FD-SOI	72
4.1	Proposed Current Sensing (CS) Architecture	74
4.2	Statistical Error Compensation for BL/SL Resistances	80
4.3	The SEC-enabled MRAM IMC Architecture	82
4.4	The Compute SNDR Methodology	86
4.5	Measured Results	88
4.6	Conclusion	93
CHAPTER 5	ON THE SECURITY VULNERABILITIES OF MRAM-BASED IN-MEMORY COMPUTING ARCHITECTURES	95
5.1	Threat Model for eNVM-based IMCs	98

5.2	A Statistical MEA Framework for eNVM-based IMCs	99
5.3	Proposed MEAs for eNVM-IMCs	102
5.4	Measured Results	110
5.5	Energy-Accuracy-Security Trade-offs in IMCs	113
5.6	Conclusion	119
CHAPTER 6 CONCLUSION AND FUTURE WORK		120
6.1	Summary of Contributions	120
6.2	Future Prospects	121
REFERENCES		124

CHAPTER 1

INTRODUCTION

Embedded non-volatile memory (eNVM) technologies have become a dynamic area of research for the past two decades, primarily due to their key advantages of non-volatility and high memory density. These devices fill the gap between SRAM and DRAM, achieving higher density than SRAM and lower read/write latency compared to DRAM [1], as shown in Fig. 1.1. Their applications span at different levels of the memory hierarchy from radiation-hardened space applications [2] to low-power Internet-of-Things (IoT) [3]. The high density eNVMs offer a solution for reducing the data movement costs, which dominates the energy consumption of computing systems, and their non-volatility enables energy-efficient duty cycled operation. Due to their scalability in advanced technology nodes, eNVMs are being actively explored as potential replacements for Flash memory and DRAM [3, 4], which face significant scaling challenges while maintaining low cost, low latency, high reliability, and energy efficiency [5].

Device-level eNVM research made significant strides between 2005 and 2015, marked by the development of the first HfO_2/Ti device using conventional fab materials [6] and the introduction of the first 40 nm node embedded ReRAM product [7]. This progress laid the foundation for exploring eNVM technologies for in-memory computing (IMC), which aims to reduce data movement costs by performing compute operations within the memory. Alongside the first proposal of SRAM-based analog IMC [8], eNVM-based IMCs [9] emerged as an energy-efficient matrix-vector multiplication (MVM) unit, a crucial compute kernel in deep neural networks (DNNs). Early behavioral models for eNVM-based IMCs promised $5.5\times$ and $7.5\times$ gains in energy efficiency and compute density, respectively [10], compared to contemporary digital accelerators [11]. These behavioral models captured the ideal functionality of eNVM-based IMCs by leveraging the proportionality of the equivalent array conductance on dot-product, performing MVM opera-

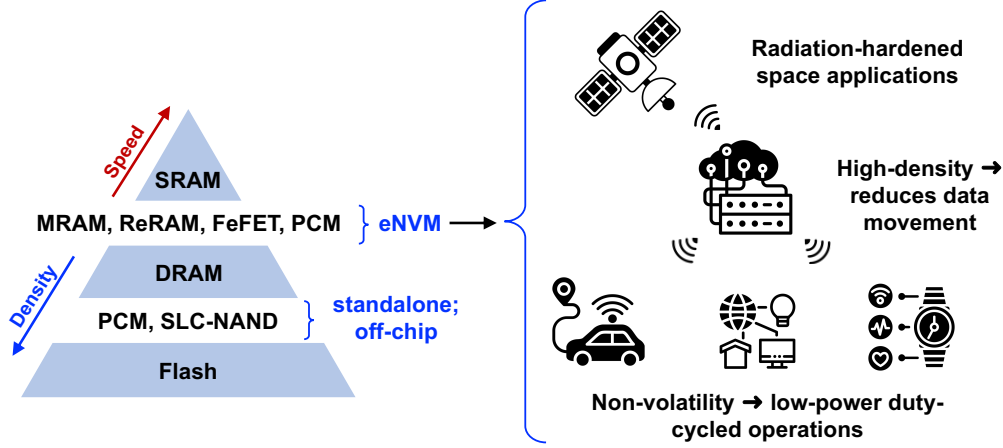


Figure 1.1: Relative position of eNVM in the memory landscape and their applications.

tions at high energy-efficiencies, thereby accelerating DNN workloads which were becoming immensely popular in various applications including computer vision [12], natural language processing [13], and speech recognition [14].

However, as integrated chip (IC) prototypes of these IMCs began to appear around 2017, the performance gains for eNVM-based IMCs [15, 16] were less impressive than expected when compared to contemporary digital accelerators [17] and SRAM-based IMCs [18]. This was surprising, given that the high memory density and non-volatility of eNVMs were anticipated to deliver substantial improvements in both energy efficiency (TOPS/W) and compute density (TOPS/mm²). This gap in key IMC metrics continues to grow as of today, with eNVM-based IMCs [19, 20] with the highest TOPS/W and TOPS/mm² significantly lagging behind the state-of-the-art digital accelerators [21] and SRAM-based IMCs [22]. One apparent reason for this lag is the requirement for large read-out circuits which occupy over 80-90% of the entire IMC bank area as observed from the chip layouts in the published works [20, 23, 24, 25, 26] (see Fig. 1.2). The need for large read-out circuits indicates the presence of dominant analog non-idealities, which, if left unaddressed, can lead to a drop in accuracy. Not surprisingly, a majority of eNVM-based IMC works employ compensation schemes at all levels of the design hierarchy, spanning device-, circuit-, architecture-, and algorithm-level innovations to achieve high accuracy. However, these schemes often come at a high cost in terms of energy efficiency and compute density. Thus, eNVM-based IMCs suffer from a low compute accuracy problem which prevents

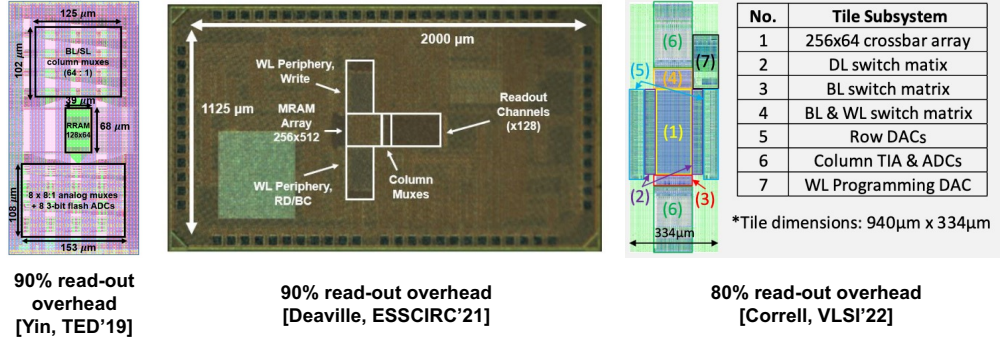


Figure 1.2: Large area overhead in eNVM-based IMCs (figure courtesy [26, 23, 20]).

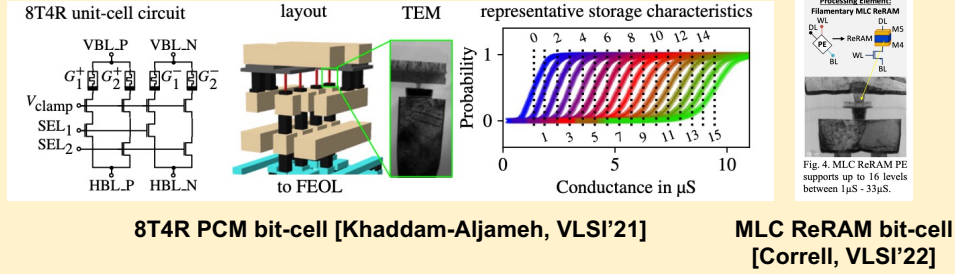
them from becoming competitive with other AI accelerators.

1.1 Accuracy of eNVM-based IMCs

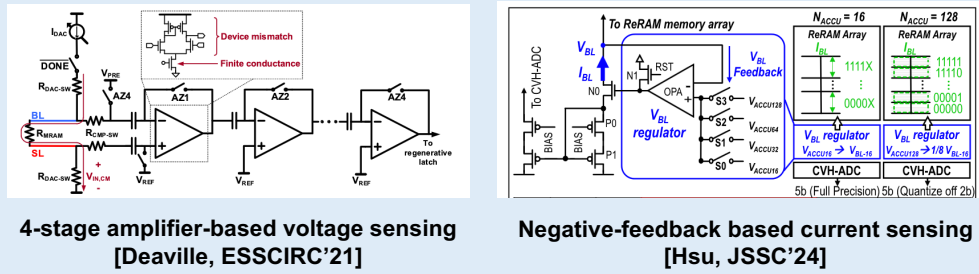
Figure 1.3 illustrates the cross-stack design innovations employed by eNVM-IMCs today to enhance their accuracy. These innovations address the non-idealities associated with resistive in-memory computation. Various memory technologies, such as phase change memory (PCM) [19], ferro-electric FET (FeFET) [30], resistive RAM (ReRAM) [15, 31], and spin transfer torque magnetic RAM (STT-MRAM) [23, 28], are utilized in eNVM-based IMCs, each offering a different conductance range for storing weights. The *conductance contrast* $g_{\text{on}}/g_{\text{off}}$ is defined as the ratio of on-state conductance (g_{on}) to off-state conductance (g_{off}). This ratio is in the range of 1.6 – 2 for MRAM with $g_{\text{on}} = 222 \mu\text{S}$ [32], 12 for ReRAM with $g_{\text{on}} = 40 \mu\text{S}$ [33], and 10^3 for both FeFET [34] and PCM [35] with $g_{\text{on}} = 1 \mu\text{S}$. A high contrast in conductance states is desirable as it increases the signal range at the dot-product level, leading to high accuracy when multiple bit-cells are activated simultaneously. Additionally, memories with high $g_{\text{on}}/g_{\text{off}}$ ratios, such as PCM and ReRAM, are used for multi-bit weight storage. These are implemented using complex bit-cell configurations like 8-Transistor-4-Resistor (8T4R) [19] or multi-level cells (MLC) [20], which pack more bits into the dot-product operation, thereby increasing throughput (TOPS), TOPS/W, and TOPS/mm².

At the circuit level, a sensing circuit is required to accurately measure the equivalent array conductance G_{eq} , which is proportional to the dot-

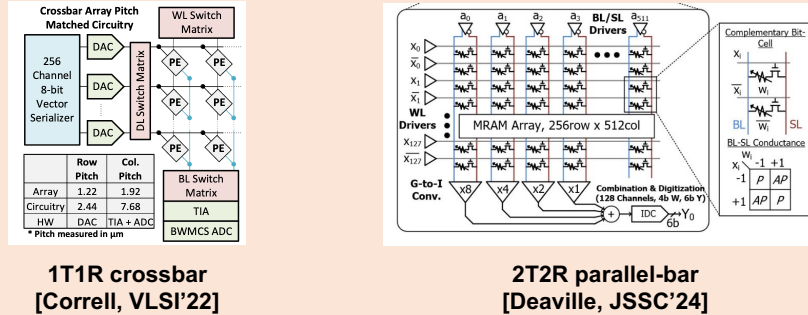
Device-level Innovations



Circuit-level Innovations



Architecture-level Innovations



Algorithmic-level Innovations

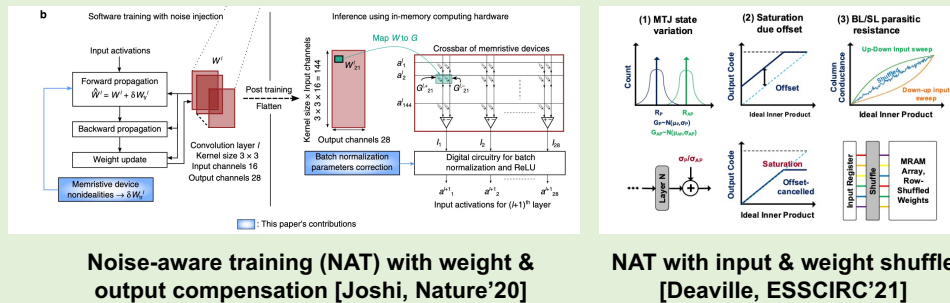


Figure 1.3: Full stack accuracy enhancing innovations in eNVM-based IMC spanning devices [19, 20], circuits [23, 27], architectures [20, 28], and algorithms [29, 23].

product. Current-sensing involves fixing a reference voltage across the array and sensing the current drawn [27, 36, 37, 38, 24, 39, 28], whereas the voltage-sensing measures the voltage drop across the array when driven by a current source [26, 23, 40]. In both cases, analog non-idealities in the sensing circuits play a crucial role in determining accuracy. For example, in current sensing, a fixed bit-line voltage [27] is essential in reducing spatial mismatch across ADC columns, while a high-gain multi-stage amplifier is necessary to sense small voltage changes in the array resistance [23].

At the architecture level, two main configurations of eNVM-based IMCs are utilized: cross-bars, where bit-lines (BLs) run orthogonal to source-lines (SLs), and parallel-bars, where BLs run parallel to SLs. While cross-bars [41, 42, 20] are popular in the device community, parallel-bar architectures [15, 23, 24, 25, 31, 43] has proved to be more scalable due to the inherent isolation between the ADC columns. This scalability is crucial for multi-bank IMC designs, enhancing their overall efficiency and performance.

At the algorithm level, a majority of resistive IMCs employ noise-aware training (NAT) [29, 19, 23, 24, 25] to improve network-level accuracy. This task-specific approach re-trains the network by incorporating the noise statistics of the IMC during training. NAT primarily involves updating the weights stored in the memory, but some designs also employ input shuffling [23] or output correction [29, 19] to improve the network-level accuracy.

The aforementioned cross-stack innovations highlight the issue of low bank-level accuracy in eNVM-based IMCs, which needs improvement to achieve accuracy comparable to digital fixed-point network-level standards. Unfortunately, most designs report only network-level accuracy after applying noise-aware training (NAT), obscuring the intrinsic bank-level accuracy and the underlying energy-accuracy trade-off. This underscores the need to quantify bank-level accuracy and comprehensively analyze intrinsic trade-offs to better understand and enhance the performance of eNVM-based IMCs.

Most analyses of resistive IMCs are based on crossbar architectures [44, 45, 46, 47, 48, 49, 50, 51], which are more prevalent in the device research community [30, 41, 52, 53, 54, 55, 34, 42, 19, 20, 56, 33]. However, the behavioral models used in these analyses often fall short of capturing all sources of non-idealities [10, 44]. When these models are used for architecture exploration of DNNs, they present an overly optimistic picture of the practical applications of crossbars. Furthermore, parallel-bar resistive IMC architectures, which

are more suited to scaling up and are commonly used in modern resistive IMCs [31, 37, 38, 57, 15, 23, 24, 39, 40, 26, 58, 59, 60, 43], have not been adequately modeled. This indicates a significant gap in our understanding of the energy-accuracy trade-offs in present-day resistive IMCs and raises the following questions:

- *What are the fundamental energy-accuracy trade-offs intrinsic to resistive IMCs?*
- *What are the limits of compute accuracy of resistive IMCs? And how can we approach it?*
- *Do the analog non-idealities make resistive IMCs secure?*

This dissertation aims to address these questions by proposing: (1) a comprehensive benchmarking methodology for the IMC design space, (2) an in-depth analysis of the energy-accuracy trade-offs in resistive IMCs, (3) a technique to boost compute accuracy for resistive IMCs, verified with a 22 nm FD-SOI chip prototype, and (4) an assessment of the security vulnerabilities of resistive IMCs.

1.2 Thesis Organization

The contributions of this dissertation are summarized as follows:

In **Chapter 2**, we propose a new benchmarking methodology for IMCs. This methodology includes a compositional view that enables the parsing of an IMC design into its canonical components, a set of benchmarking metrics for evaluating performance, efficiency, and accuracy, and a systematic strategy for analyzing the data and metrics reported in publications. We applied this benchmarking methodology to a database of over 70 IC designs published since 2018. The results show that SRAM-based IMCs exhibit superior energy efficiency and compute density at the bank level compared to digital accelerators. Conversely, resistive IMCs lag behind SRAM-based IMCs in both energy efficiency and compute density, and, surprisingly, they also fall behind digital accelerators in terms of compute density. Additionally, the benchmarking reveals a significant omission in recent IMC works: the evaluation of the bank-level accuracy metric, signal-to-noise-plus-distortion ratio (SNDR), and the corresponding energy-SNDR trade-off for different IMCs.

Chapter 3 determines the fundamental limits on the compute accuracy of resistive IMCs by employing statistical signal and noise models. This chapter uses compute SNDR to quantify the accuracy of IMCs and identify the device, circuit, and architectural parameters that affect it. We further analyze the fundamental limits on the SNDR of MRAM, ReRAM, and FeFET-based IMCs using parameter variation and noise models validated against measured results from a recent MRAM-based IMC prototype in a 22 nm process. Our analysis reveals that at high output signal magnitudes, the maximum achievable SNDR is limited by pre-ADC array non-idealities such as conductance variations, parasitic resistances, and current mirror mismatches. Conversely, at small signal magnitudes, ADC thermal noise becomes the limiting factor for SNDR. Furthermore, for large dot-product (DP) dimensions ($N > 50$), FeFET-based IMCs achieve the highest maximum SNDR, followed by ReRAM, and then MRAM. The increase in conductance contrast ($g_{\text{on}}/g_{\text{off}}$) enhances the maximum achievable SNDR only until it reaches a value of approximately 12. Lastly, we extend our analysis to resistive crossbar architecture to obtain similar conclusions, with maximum achievable SNDR being limited by pre-ADC non-idealities. The system-level accuracy in both cases strongly depends on the bank-level SNDR, indicating that optimizing SNDR is crucial for maximizing network-level accuracy.

In **Chapter 4**, we leverage the insights from the behavioral model in **Chapter 3** to propose two SNDR boosting techniques for resistive IMCs. At the circuit level, we introduce offset-compensating current sensing (OCCS) to reduce static ADC column mismatch. At the algorithmic level, we address non-linearity arising from bit-line/source-line (BL/SL) wire parasitics using a low-overhead statistical error compensation (SEC) block. We demonstrate the efficacy of these SNDR boosting techniques on an MRAM IMC macro in a 22 nm FD-SOI process, achieving an SEC-enabled SNDR boost of 2.7 to 6 dB across different operating points. This boost can be traded off to achieve a substantial $5\times$ reduction in energy per 1b-operation, with a modest SEC energy overhead of 0.8 % and an area overhead of 12.2 %. Furthermore, the work shows an SEC-enabled increase in neural network (NN) accuracy from 74.8 % to 82.0 % for CIFAR-10 by mapping the last layer of ResNet-20, without resorting to noise-aware training.

In **Chapter 5**, we study the vulnerability of resistive IMC architectures to model extraction attacks (MEAs). These attacks allow the retrieval of pri-

vate training data from trained model parameters, thereby leaking sensitive user information. The presence of analog noise in resistive IMC computation suggests that they may be intrinsically robust to MEA. However, we show that this conjecture is false. Specifically, we consider the scenario where an attacker aims to retrieve model parameters via input-output query access and propose three attacks that exploit the statistics of the IMC computation based on the behavioral model in **Chapter 3**. We demonstrate the efficacy of these attacks in extracting the model parameters of the last layer of a ResNet-20 network from the bitcell array of the MRAM-based IMC prototype described in **Chapter 4**. Employing the proposed MEAs, the attacker obtains a CIFAR-10 accuracy within 0.1% of that of a $N = 64$ dimensional, $7\text{ b} \times 4\text{ b}$ fixed-point digital baseline. *To the best of our knowledge, this is the first work to demonstrate MEAs for resistive IMC on a real-life IC prototype.* Our results indicate the critical importance of investigating the security vulnerabilities of IMCs in general and resistive IMCs in particular.

Lastly, **Chapter 6** concludes this dissertation and proposes future research directions.

CHAPTER 2

BENCHMARKING IN-MEMORY COMPUTING ARCHITECTURES

In this chapter, we propose a benchmarking methodology to compare present-day in-memory computing (IMC) works. Since its inception in 2014 [8], IMC has become an active area of research in the integrated circuit design community globally for realizing artificial intelligence and machine learning workloads. IMCs address head-on the well-known data movement cost in modern AI workloads. These workloads typically require the storage and processing of large models, e.g., 175 B for GPT-3 [61] natural language model, to generate a single inference. The emergence of deep learning in 2012 [62] has only aggravated this long-standing problem, and IMCs are viewed as a potential solution.

The data movement problem was baked into the von Neumann architecture in 1945 [63] via the physical separation of the processor and memory leading to an energy cost of a byte access from a 32 KB SRAM to be $> 100\times$ that of an 8 b multiply-accumulate (MAC) in 45 nm node [64]. This problem has been a subject of much investigation in the microarchitecture research community over the past few decades resulting in a body of work referred to as processor-in-memory (PIM). Computational RAM [65] implements a 128×64 -bit SRAM array with 64 bit-serial column-pitch matched digital processors. Intelligent RAM [66] employs programmable vector processing along side 13 MB of embedded DRAM to alleviate the memory bandwidth problem. Smart Memory [67] employs a modular tiled architecture with each tile consisting of a 128×144 -bit SRAM array and a 6-input, 4-output reconfigurable PLA. However, in each of these, the separation between low-swing memory access and the full-swing digital domains is preserved. In contrast, Kerneltron [68] employs a specialized charge-injection based DRAM array (256×128 -b) to compute a matrix-vector multiply (MVM) *in analog* required to implement a support vector machine. It represents a true precursor of today's IMCs.

The concept of SRAM-based IMC was first proposed in [8], which demonstrated that a multi-bit, high-dimensional dot product computation, can be efficiently realized during a *single* memory read using a standard 6T SRAM bitcell array (BCA). IMC design concepts popular today, such as pulse-width modulated activations and charge-domain summing, were also proposed and subsequently validated via multiple prototype IC designs in 65 nm CMOS [69, 70, 71, 72].

Since 2017, IMC design has become an active area of research with > 50 IMC-related papers published in top circuit design conferences demonstrating significant reductions ($> 13\times$) in energy over their digital counterparts especially at the bank-level. Today, the energy-efficiency benefits of bank-level IMC designs have saturated and 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 caused by its full-stack (devices-to-systems) nature; 2) the presence of a fundamental energy efficiency vs. compute signal-to-noise-plus-distortion ratio (SNDR) 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 [73] – 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.

In this chapter, we propose a rigorous benchmarking methodology for IMCs comprising: 1) a compositional view of IMCs that enables one to parse an IMC design into its canonical components; 2) a set of benchmarking metrics to quantify the performance, efficiency, and accuracy of IMCs; and 3) a strategy for analyzing the reported IMC data and metrics [74]. We apply the proposed benchmarking methodology on an extensive database of metrics collected from > 70 machine learning IC designs published in major circuits conferences since 2018, in order to comprehend trends and identify challenges in IMC design. Though much of the discussion is on SRAM-based IMCs [75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112], comparisons are drawn with eNVM-based IMCs [38, 19, 30, 113, 59, 40, 23, 42, 37, 36, 26, 15, 60, 43, 114, 115, 116, 24], eDRAM-based IMCs [117, 118, 119], and with recent digital accelerators [120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 17, 21, 132, 133, 134, 135, 136, 137]. The database

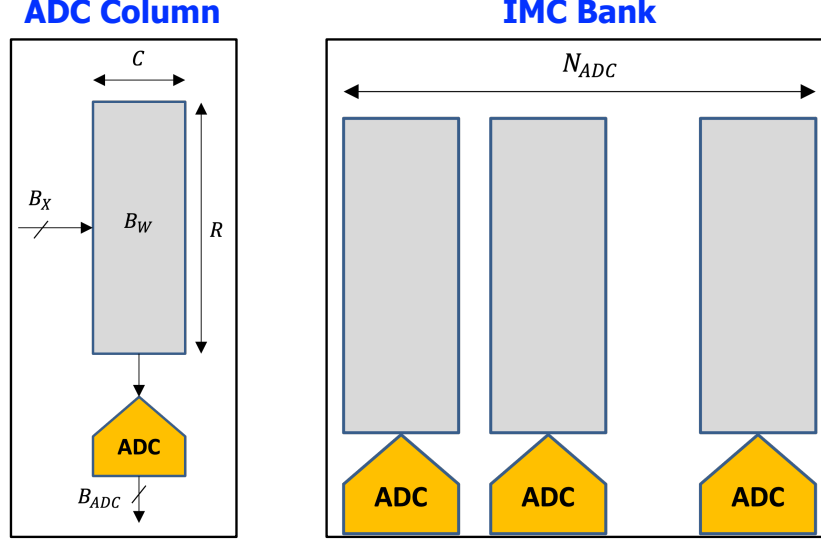


Figure 2.1: An IMC processor viewed as an array of N_{core} IMC banks with each bank comprising an array of N_{ADC} parallel ADC columns. An ADC column represents the fundamental unit for constructing IMCs.

and Python code to generate the plots in this chapter can be downloaded from <https://github.com/UIUC-IMC/UIUC-IMC-Benchmarking>.

2.1 A Compositional View of IMCs

The full-stack nature of IMC makes it hard to comprehend the complex interplay between various factors contributing to its overall efficiency. To address this conceptual difficulty, we propose a compositional view of IMCs shown in Fig. 2.1 where an IMC bank is composed of N_{ADC} ADC columns with each ADC column implementing an N -dimensional dot product per read cycle. Furthermore, an IMC processor (not shown) can be composed by integrating N_{core} IMC banks. Typical range of values in today's IMC designs are: $N_{\text{core}} \in \{2, 4, 8, 16\}$, $N_{\text{ADC}} \in \{16, 32, 64, 128, 256\}$, and $N \in \{64, 128, 256, 512, 1024, 2048\}$.

The compositional view of IMCs shown in Fig. 2.1 emphasizes the role of a single ADC column when evaluating IMCs since the bank-level and processor-level metrics are inherited from those of the ADC column. Thus, the canonical component of an IMC is an ADC column, which is described next.

2.1.1 ADC Column

An ADC column (see Fig. 2.1) computes a $B_X \times B_W$ -b N -dimensional dot product in a single read cycle employing circuitry (row drivers, bitcells, precharge and the summing circuitry) spread over R physical rows and C columns of the bitcell array preceding the input of a single B_{ADC} -b ADC. *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.

Thus, the key ADC column parameters are: 1) input (B_X) and weight (B_W) precisions; 2) dot product dimension (N); 3) number of physical rows (R) and columns (C); and 4) row (column) parallelism R_C (C_C) refers to the actual number of rows (columns) activated during one read cycle.

Next, we define key ADC column metrics (see Fig. 2.1) that result from parameter choices made and the specifics of the implementation. These include: 1) the number of bits processed per read cycle:

$$B_{\text{col}} = C_C R_C B_X = N B_W B_X \quad (2.1)$$

; 2) the information content at the input of the ADC:

$$B_{I(\text{col})} = B_X + B_W + \log_2(N) \quad (2.2)$$

with exceptions for 1-b operands; 3) column energy (E_{col}); 4) column latency (T_{col}); 5) bit-level throughput (1b-TOPS= $2B_{\text{col}}/T_{\text{col}}$); 6) per bit energy efficiency (1b-TOPS/W); 7) bit-level compute density (1b-TOPS/mm²); and 8) compute SNDR (SNDR_{col}). Note: we employ bit-normalized metrics (1b-TOPS, 1b-TOPS/W, 1b-TOPS/mm²) to enable comparison across IMCs with diverse arithmetic precisions.

IMCs strive to maximize N to amortize the cost of a memory read access over as many computations as possible in order to minimize the energy per 1 b OP E_{op1} given by:

$$E_{\text{op1}} = \frac{1}{1\text{b-TOPS/W}} = \frac{E_{\text{col}}}{2B_X B_W N} = \frac{E_{\text{bl}} + E_{\text{ADC}}}{2B_X B_W N} \quad (2.3)$$

where E_{bl} is the energy of the bitline computations in the ADC column prior to the ADC, and E_{ADC} is the ADC energy. It is clear that IMCs can maxi-

mize their 1b-TOPS/W metric by implementing multi-bit high-dimensional computations per read cycle.

Of the eight ADC column metrics mentioned above, 1)-7) are usually available in publications. Metric 8) is hard to measure and is almost never reported. We emphasize the importance of characterizing all eight ADC column metrics to fully explain the benefits of any new IMC design method.

2.1.2 Bank

An IMC bank (see Fig. 2.1) comprises N_{ADC} ADC columns operating in parallel. Thus, all bank-level parameters are directly inherited from those of a single ADC column, e.g. the number of physical columns in the bank is given by $N_{\text{col}} = N_{\text{ADC}}C$. Similarly, bank level metrics such as the number of bits processed per read cycle $B_{\text{core}} = N_{\text{ADC}}B_{\text{col}}$, and the information content ($B_{I(\text{core})} = N_{\text{ADC}}B_{I(\text{col})}$) 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/mm²) 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.

2.1.3 Processors

An IMC processor comprises N_{core} banks connected with an on-chip network. IMC processors or multi-bank IMCs [76, 77, 78, 82, 83, 86, 89, 94, 96, 105, 114, 111, 104, 119] 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.

2.2 IMC Data Analysis Strategy

Our data analysis strategy leverages the compositional view of IMCs outlined in Section 2.1. 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-, eNVM-, and eDRAM-based IMC designs, we have considered only the most recent digital accelerators (since 2019). Instead of simply plotting reported data from comparison tables, we employ the following strategy:

1. We begin top-down by scaling the reported bank-level TOPS/W metrics by the precision factor, i.e., either $B_X B_W$ or $2B_X B_W$ to obtain 1b-TOPS/W in order to ensure that a multiply-accumulate is treated as two OPs per standard practice.
2. We identify and analyze the ADC column to determine the column parameters N , R_C , C_C , B_X , B_W , and B_{ADC} per Fig. 2.1, and derive metrics R_P , C_P , B_{col} , and $B_{I(\text{col})}$.
3. Using the reported value of T_{core} we compute 1b-TOPS as:

$$1\text{b-TOPS} = 2B_X B_W R_C N_{\text{ADC}} / T_{\text{core}}. \quad (2.4)$$

This bottom-up computed throughput is compared with the top-down (reported) value. In case of a discrepancy (a rare occurrence), we prioritize the computed value. Works where the discrepancy is $> 10\times$ are not reported as the ADC column information is unspecified in those cases and hence cannot be verified.

4. 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.
5. We verify the reported 1b-TOPS/mm² by normalizing the bottom-up computed 1b-TOPS by the bank area. The latter is estimated from the

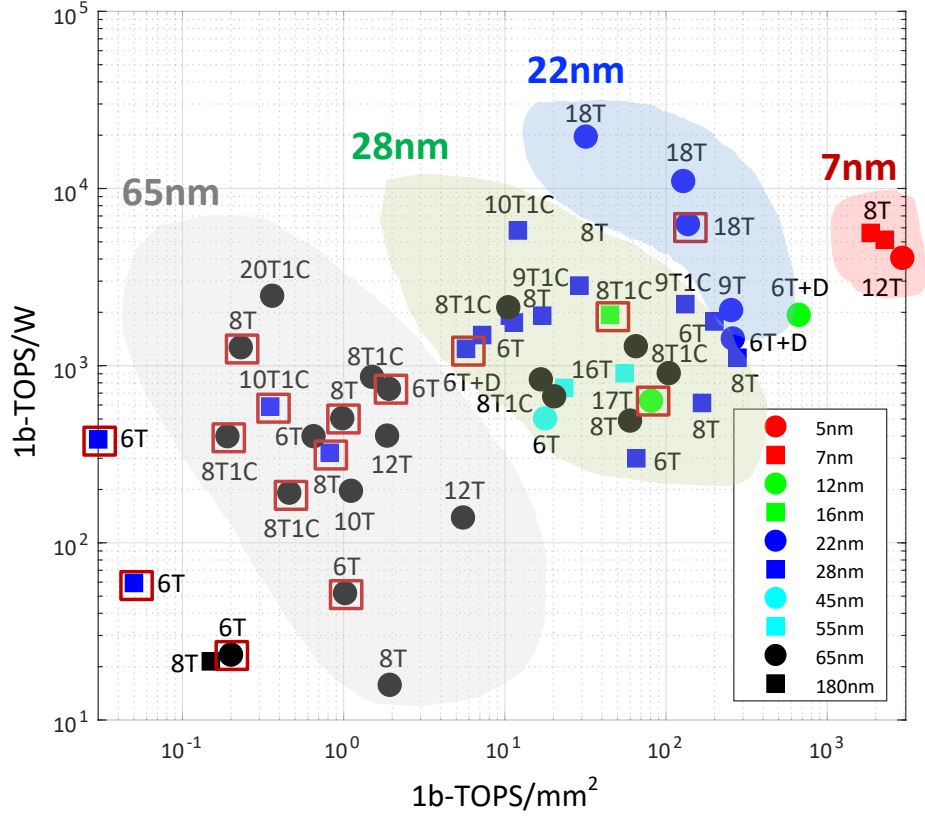


Figure 2.2: 1b-TOPS/W vs. 1b-TOPS/mm² for SRAM-based IMCs categorized w.r.t. the technology node and bitcell architecture. All the metrics are per bank, including those for IMC processors (red boxes).

die photo when it is not reported.

6. We do not scale the efficiency values by the ADC precision or normalize w.r.t. to the technology nodes to preempt controversy.

Next, we apply the data analysis strategy to SRAM-based IMCs first and then compare SRAM-, eNVM-, eDRAM- based IMCs, and digital accelerators.

2.3 SRAM-based IMC Trends

Figure 2.2 shows that the energy efficiency (1b-TOPS/W) and the compute density (1b-TOPS/mm²) both improve with technology scaling. This lays to rest the concern that IMCs may be at a disadvantage in advanced nodes

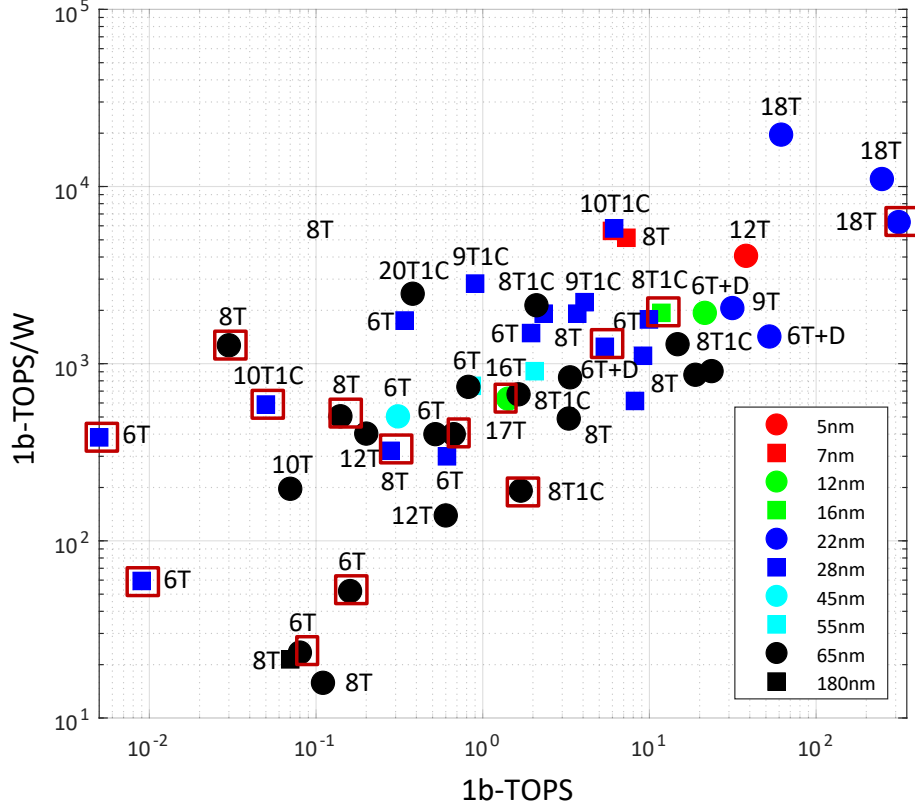


Figure 2.3: 1b-TOPS/W vs. 1b-TOPS for SRAM-based IMCs. All metrics are per bank, including those for IMC processors (red boxes).

due to their analog-heavy computations. A couple of outliers are the 16nm 8T1C design [77] and the 12nm design [96], 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 iso-technology 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/mm².

A key reason underlying the trends in Fig. 2.2 can be found in Fig. 2.3, 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 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 [100] has low array utilization due to the use of a high ADC column muxing ratio of 1:64, while the 8T design in [88] reports higher latency T_{core} to improve accuracy.

Finally, the 6T+D designs [79, 102] (6T cells with embedded digital logic) achieve higher 1b-TOPS by performing digital accumulation, thereby avoiding voltage headroom constraints. Such designs offer an interesting hybrid of analog and digital computation within the IMC framework. They can be a serious contender, provided the energy and area overhead of digital processing can be minimized.

It would be useful if the bank-level trends in Figs. 2.2-2.3 could be predicted by specific ADC column level metrics. Indeed, it turns out (see Fig. 2.4) that energy efficiency tracks the information content $B_{I(\text{col})}$ at the ADC input (see (2.2)) very closely. We observe a $10\times$ improvement in 1b-TOPS/W for every bit increase in $B_{I(\text{col})}$. This trend clearly points out the importance of maximizing $B_{I(\text{col})}$ to maximize energy efficiency. Equation (2.2) shows that $B_{I(\text{col})}$ is most effectively maximized by realizing multi-bit arithmetic ($B_X, B_W > 1$) in the BC, and by increasing the dot product dimension N . However, the energy efficiency gain will be limited by the increase in the ADC precision required to preserve accuracy, thereby indicating the fundamental energy vs. SNDR trade-off in IMCs.

2.4 Comparison Across Architectures

In this section, we compare energy efficiency, compute density, and throughput metrics for SRAM-, eNVM-, eDRAM-based IMCs, and digital accelerators. Recall that we always compare bank/core-level metrics throughout this chapter, even when comparing processor architectures.

2.4.1 Energy efficiency vs. Compute Density

Figure 2.5 shows that SRAM-based IMCs achieve both the highest bank-level energy efficiency and the highest compute density compared to eNVM-,

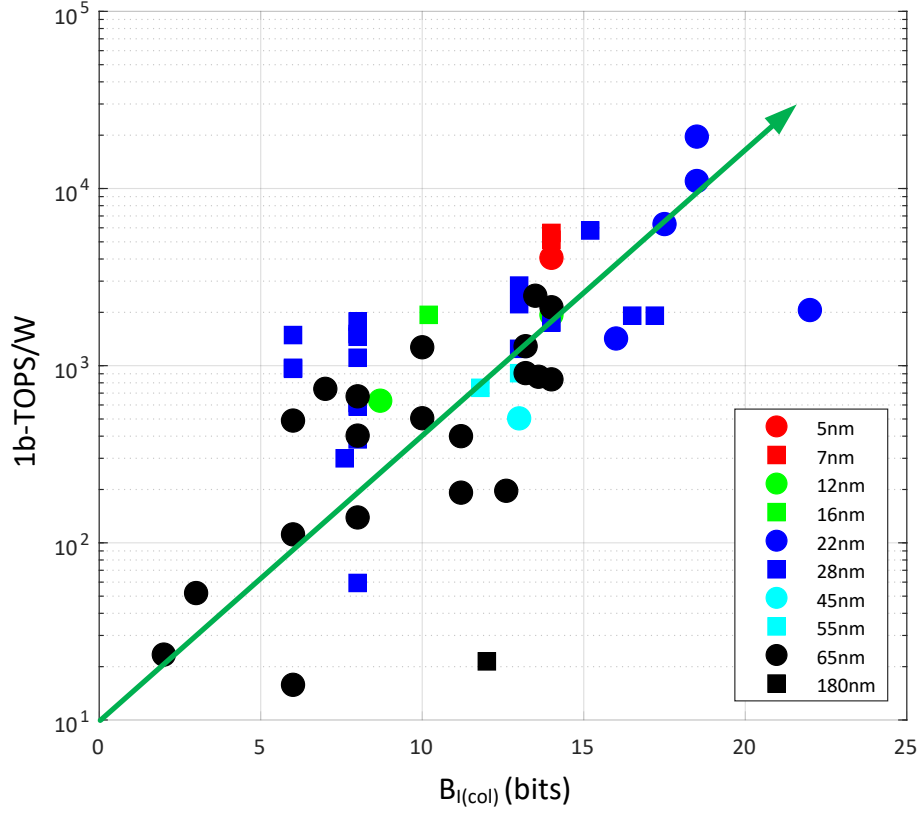


Figure 2.4: 1b-TOPS/W vs. $B_{I(\text{col})}$ for SRAM-based IMCs.

eDRAM-based IMCs, and digital accelerators. The best (average) reported 1b-TOPS/W for SRAM-based IMCs is about $13 \times$ [22] ($5 \times$) higher than that for digital accelerators [136], about $14.2 \times$ [43] ($4.8 \times$) higher than that for eNVM-based IMCs, and about $12 \times$ [119] ($2.44 \times$) higher than that for eDRAM-based IMCs. The latter seems to be a promising new entrant into the IMC space [118, 119, 117] achieving energy-efficiencies and compute densities comparable to mainstream SRAM-based IMCs in spite of being implemented in 65 nm CMOS. These employ charge redistribution methods for computing dot products. The design in [118] repurposes the 1T1C eDRAM array for in-memory compute but has low array utilization (lower 1b-TOPS) to avoid destructive reads whereas [119] proposes a 3T dynamic analog RAM using multi-bit weights, and adaptive ADC skipping techniques to achieve higher TOPS while preserving the accuracy. A 2T1C bitcell based eDRAM design proposed in [117] demonstrates bit-line swing aware computation accompanied by a compact Schmitt trigger-based ADC.

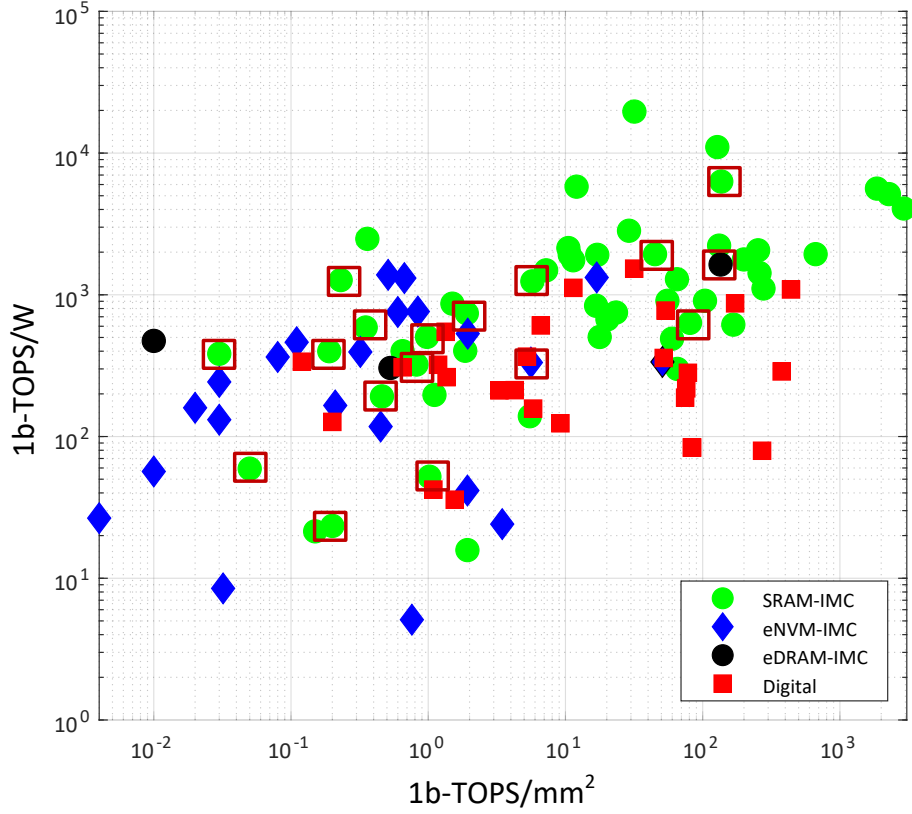


Figure 2.5: 1b-TOPS/W vs. 1b-TOPS/mm² categorized with respect to SRAM-, eNVM-, eDRAM-based IMCs and digital accelerators. The centroids (1b-TOPS/mm², 1b-TOPS/W) for each cluster are at: SRAM (195.8, 1966.5), eNVM (3.9, 406.4), eDRAM (45.17, 804.3) and digital (68.4, 393.6).

However, as Fig. 2.5 shows, the efficiency gap between a comparable multi-bank SRAM-based IMC processor (16 bank, 16 nm) [77] and a digital accelerator [17] (16 PE, 16 nm) reduces to $3\times$, as predicted in [138]. This gap further reduces to $1.26\times$ when compared to [136], which is a 5 nm design. 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. 2.1), their slopes may be different. Therefore, it will be interesting to see if this efficiency gap widens or reduces in the future. The large bank size in [105] (1152×1024) seems to be the critical distinguishing factor enabling high energy efficiency, high compute density, and high throughput. Higher input precision [105] implemented using multi-bit DACs also plays an important role in achieving higher 1b-TOPS.

Figure 2.5 also indicates that the best (average) reported compute density for SRAM IMCs is about $6.6 \times [102]$ ($2.86 \times$) greater than that of digital accelerators [21], about $57.4 \times [19]$ ($50.2 \times$) when compared to eNVMs, and about $21.6 \times [119]$ ($4.33 \times$) when compared to eDRAM-based IMCs. 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 areal density. One exception is the PCM crossbar [19] utilizing multi-bit weight storage with analog compensation techniques to achieve accurate MVM operation in the 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. This result can be viewed as a future research opportunity or may indicate a fundamental limitation of eNVM-based IMCs. Preliminary work [139] indicates the latter to be the case for resistive crossbars – an assertion supported by recent crossbar IMCs [42, 116] that allocate 80% of the bank area to the sensing circuitry/peripherals.

2.4.2 Energy efficiency vs. Throughput

The trends in Fig. 2.5 can be explained by comparing energy efficiency with throughput achieved by the four types of architectures. Figure 2.6 shows that digital accelerators achieve the highest throughput. The best (average) reported throughput for digital accelerators is $42.2 \times [127]$ ($39.13 \times$) higher than that of SRAM-based IMC [105], roughly about $363 \times [19]$ ($204.2 \times$) compared to that of eNVM-based IMCs, and about $1885.7 \times [119]$ ($272.3 \times$) compared eDRAM-based IMCs. Note that this huge throughput advantage of digital accelerators is achieved at a reasonable ($2.86 \times$ on average) expense in compute density, as indicated in Fig. 2.5. Clearly, digital accelerators are the leading contenders in the domain of deep learning accelerators today.

Not shown in Fig. 2.6 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 they SRAM or

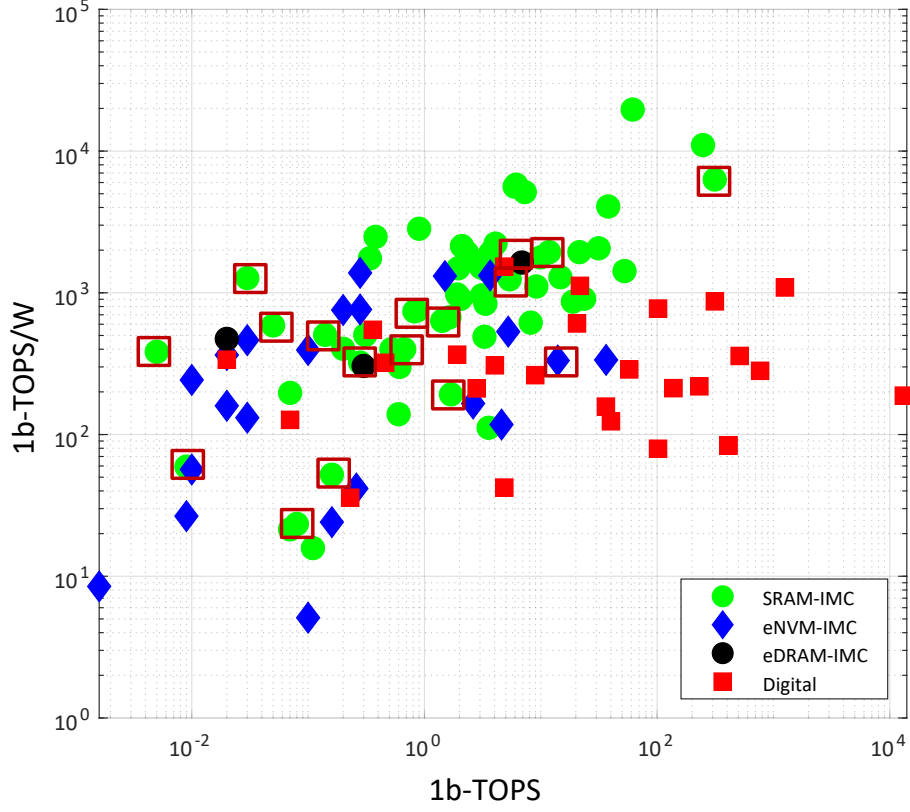


Figure 2.6: 1b-TOPS/W vs. 1b-TOPS categorized with respect to SRAM-, eNVM-, eDRAM-based IMCs and digital accelerators. The centroids (1b-TOPS, 1b-TOPS/W) for each cluster are at: SRAM (16.7, 1880.3), eNVM (3.2, 406.4), eDRAM (2.4, 804.3) and digital (653.5, 393.6).

eNVM or eDRAM-based, need to work much harder to preserve accuracy during scale-up. Specifically, in the 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 [42, 15] which have higher 1b-TOPS with multi-bit input/weight achieve a low accuracy of 90%-92% on MNIST.

Achieving high compute accuracy in eNVM-based IMCs is hard because the read-out circuit needs to sense small changes in small resistances, e.g., a $10\text{ k}\Omega$ cell resistance appears as a $40\ \Omega$ resistor to the read-out circuitry when computing a 256-dimensional dot product. Additionally, the compute SNDR is further reduced due to analog non-idealities such as wire parasitics and conductance mismatch. These necessitate high-sensitivity CMOS read-out circuitry to maintain the fidelity of analog computations. Thus, the area

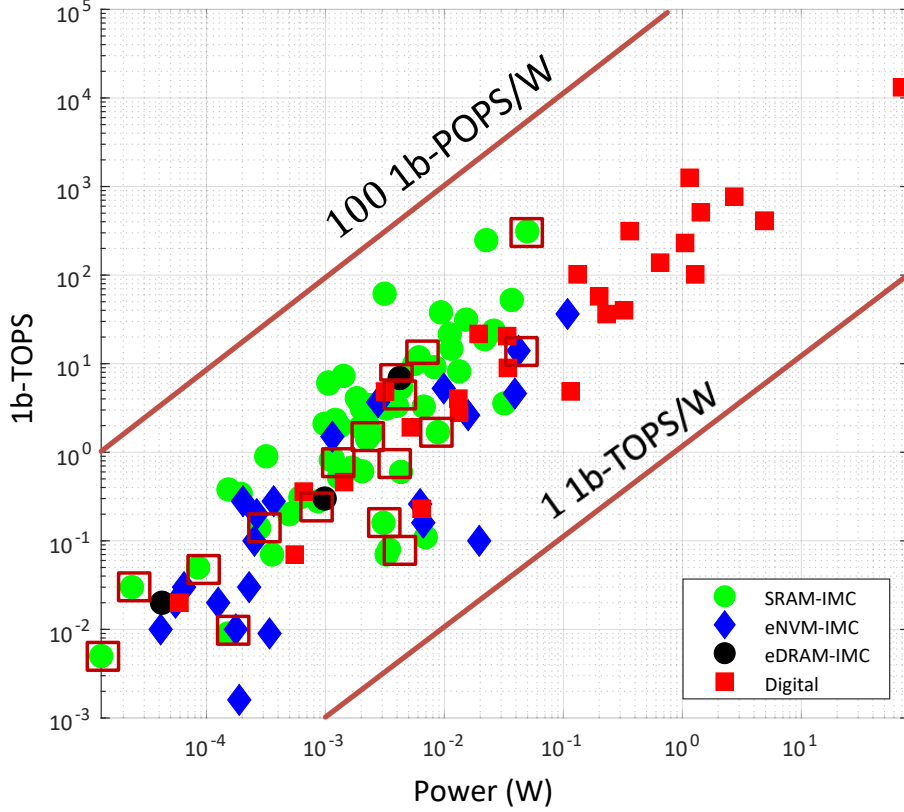


Figure 2.7: 1b-TOPS vs. power (W) categorized with respect to SRAM-, eNVM-, eDRAM-based IMCs and digital accelerators. The centroids (mW , 1b-TOPS) for each cluster are at: SRAM (6.3, 16.7), eNVM (11.5, 3.2), eDRAM (1.7, 2.4) and digital (3323.3, 653.5).

overhead of peripherals (DAC, sensing, and ADC) tends to be large, e.g., 90% of the total bank area in [26]. This further reduces their 1b-TOPS/ mm^2 in Fig. 2.5.

2.4.3 Throughput vs. Power

Finally, it is instructive to compare the throughput achieved at a specific power consumption. Figure 2.7 shows that all architectures lie between the 100 1b-TOPS/ W (10^5 1b-TOPS/ W) and 1 1b-TOPS/ W iso-efficiency contours 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 and then eNVM and eDRAM-based IMCs. FeFETs-based IMCs [30, 113] 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 prototype IC has yet to be published.

2.5 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. 1-7 is the accuracy achieved at a specified efficiency or throughput. This omission is a glaring weakness in IMC design today, since no papers measure or report the compute SNDR SNDR_{col} of an ADC column (see Fig. 2.1) alongside energy efficiency in their comparison tables. Unlike digital accelerators that can realize an arbitrary level of accuracy simply by scaling up their arithmetic precision, IMCs trade-off their compute SNDR for energy efficiency gains. Thus, it is critically important for IMC works to report the 3-tuple (energy efficiency, throughput, accuracy) for their ADC column/bank. An IMC's accuracy needs to be reported at both the bank-level (compute accuracy SNDR_{col}) and at the network level.

2.5.1 Digital Baseline

Quantifying the accuracy of an ADC column/bank requires a reference to compare with. Since IMCs have digital interfaces with analog computations sandwiched in between, we choose a fixed-point digital computation as the baseline whose functionality is defined below:

$$y_o = \mathbf{w}^T \mathbf{x} \quad (2.5)$$

where $\mathbf{x}$ is an N dimensional B_X -bit input/activation vector, $\mathbf{w}$ is an N dimensional B_W -bit weight vector, and y_o is a scalar output with precision B_y given by (2.2), i.e., $B_y = B_X + B_W + \log_2(N)$ (commonly known as the *bit growth model*). This enables IMCs to be compared directly with digital accelerators in terms of energy efficiency, throughput, and accuracy.

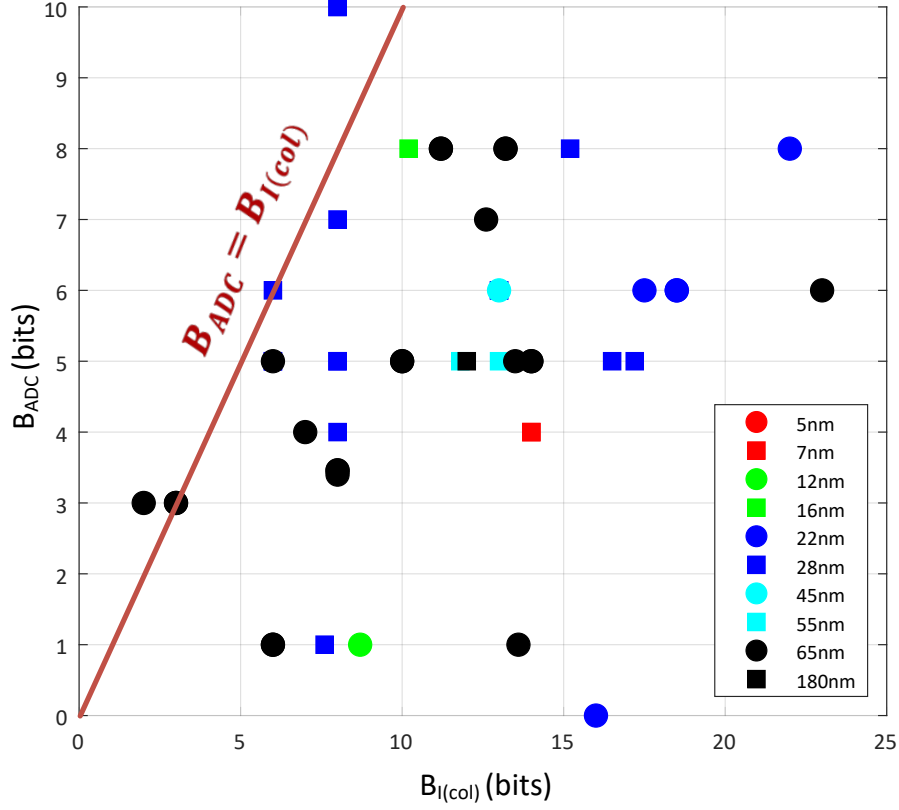


Figure 2.8: B_{ADC} vs. $B_{I(col)}$ for SRAM IMCs categorized with respect to technology node.

2.5.2 ADC Column Accuracy Metric

The compute SNDR of an ADC column ($SNDR_{col}$) can be written as [140]:

$$SNDR_{col} = \left[\frac{1}{SNDR_a} + \frac{1}{SNDR_{ADC}} \right]^{-1} = \sigma_{y_o}^2 / \sigma_e^2 \quad (2.6)$$

where $SNDR_a$ is the (analog) SNDR of the signal feeding into the ADC, $e = y_o - y_{imc}$ is the error between the IMC (ADC) output y_{imc} , and the fixed-point digital output y_o computed using bit growth. Note: in defining the compute SNDR in this manner, we have adopted a broader definition of non-idealities (similar to the SNDR metric for ADCs) that includes both systematic and random variations, e.g., spatial variations in the BCs, coupling noise, ADC thermal noise, ADC quantization/clipping noise, as well as any non-linearity in y_{imc} .

The analog SNDR ($SNDR_a$) reduces as a function of the dot product di-

mension N , B_X , and B_W since the increased output dynamic range needs to fit within headroom constraints. This leads to the fundamental SNDR (SNDR_a) vs. energy efficiency trade-off in the pre-ADC column computation circuitry.

The term SNDR_{ADC} in (2.6) is the ADC’s SNDR. Interestingly, SNDR_{ADC} and its energy consumption are both exponential functions of its effective number of bits (ENOB) [138]. Since the ADC’s energy consumption tends to dominate the overall energy efficiency of an IMC, it is of prime importance to minimize the required ENOB. However, doing so reduces SNDR_{ADC} leading to the ADC dominating SNDR_{col} in (2.6). This points out the complex interplay between the pre-ADC column computation and the ADC which determines the overall SNDR_{col} vs. energy efficiency trade-off in IMCs.

2.5.3 Optimizing ADC Column Accuracy vs. Energy Trade-off

To optimize the ADC column’s accuracy vs. energy trade-off, one needs to design ADCs with the minimum SNDR_{ADC} required to ensure that SNDR_{col} in (2.6) is sufficiently high to meet the network accuracy requirements. A valid question to ask is: what is a sufficiently high value for SNDR_{col} ?

Our prior work [141, 142] indicates that the signal-to-quantization-noise ratio (SQNR) of fixed-point dot product computations needs to fall in the range [10 dB, 40 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 $\text{SNDR}_{\text{col}} \in [10 \text{ dB}, 40 \text{ dB}]$ to be assured that network accuracy will be close to that of floating-point implementations of the same network, though quantization-aware training (QAT) [143] can reduce the SNDR_{col} requirements.

IMC designs employing the charge redistribution model of computation [89] using MOM capacitors tend to have high values of SNDR_a because bitcell computations are made full-swing (digital) and capacitor mismatch is $< 1\%$. In such designs, SNDR_{ADC} determines the overall accuracy SNDR_{col} of the ADC column. In contrast, in IMCs using the charge summing model of

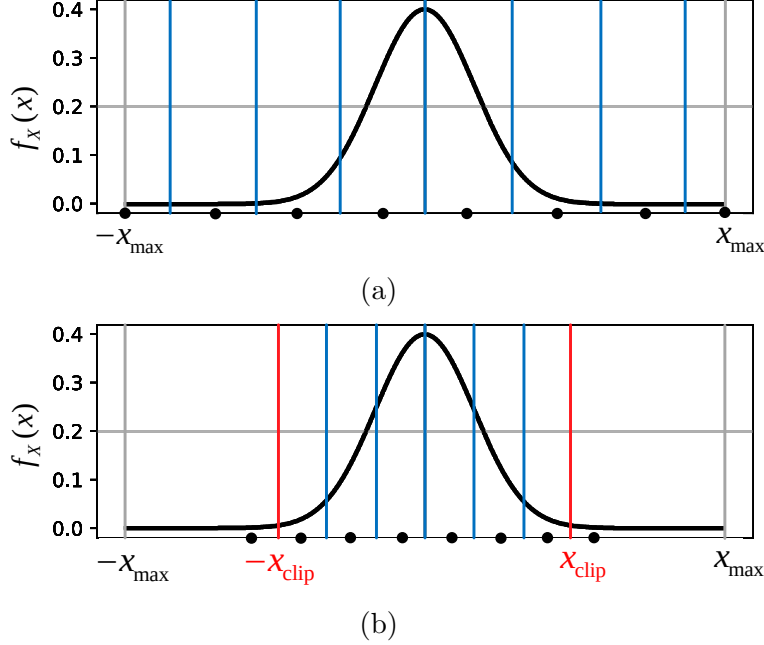


Figure 2.9: Quantizing the column ADC input with $B_{\text{ADC}} = 3$ -b using: (a) full-range quantizer, and (b) Optimal Clipping Criterion (OCC)[144]. Here $f_X(x)$ is the input signal distribution.

computation [88], SNDR_a is limited because the bitcell computations rely on cell currents which are subject to spatial variations and non-linearity. In such IMCs, SNDR_{ADC} needs to be sufficiently greater than SNDR_a so that $\text{SNDR}_{\text{col}} \rightarrow \text{SNDR}_a$, e.g., if $\text{SNDR}_{\text{ADC}} > \text{SNDR}_a + 9 \text{ dB}$, then SNDR_{col} lies within 0.5 dB of SNDR_a .

Once the minimum required SNDR_{ADC} is determined, one needs to find the smallest ADC precision B_{ADC} needed to meet this requirement. Noting the fact that for large dot product dimension N , the ADC input signal has a distribution $f_X(x)$ with a small variance relative to its dynamic range, i.e., a large *peak-to-average power ratio* (PAPR) [138], one can minimize B_{ADC} by allowing for the input signal into the ADC (see Fig. 2.9) to be clipped. Doing so enables the ADC to quantize only the most informative part of the input signal range. The Optimal Clipping Criterion (OCC) [144] does just that. It generates a theoretically optimal clipping range $\pm x_{\text{clip}}$ that maximizes the signal-to-quantization noise ratio (SQNR), bringing it to within a fraction of a dB of that of the Lloyd-Max [145] quantizer. Column ADCs operate in a thermal noise-limited regime for large N , implying that its energy consumption quadruples with each additional bit of ENOB [138].

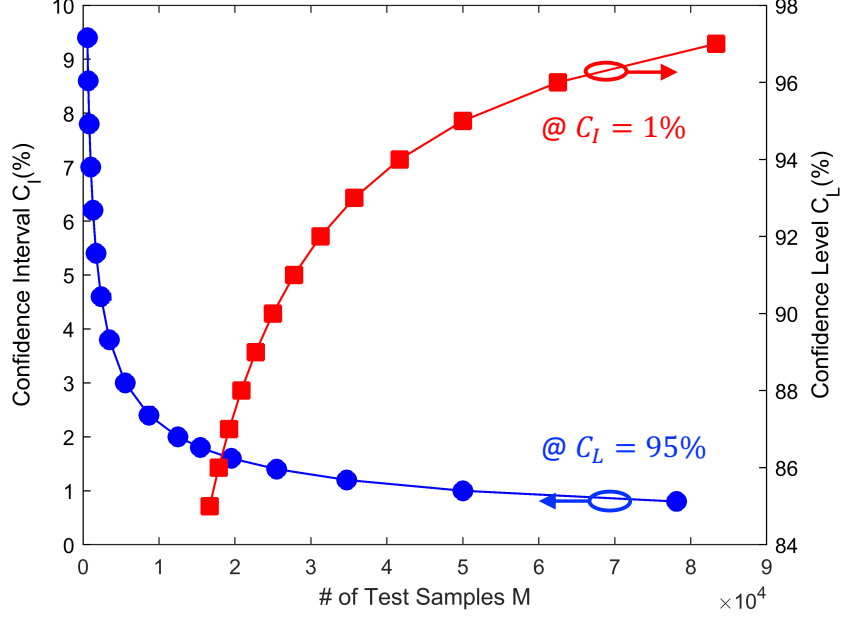


Figure 2.10: Confidence interval (C_I) and confidence level (C_L) vs. number of test samples M .

Therefore, it is critical to employ methods such as OCC [144] to maximize SNDR_{ADC} with as small a B_{ADC} (hence ENOB) as possible.

Currently, the approach described above is not followed and SNDR_{col} is typically not reported in IMC publications ([81, 89] being rare exceptions). Therefore, as a proxy for SNDR_{col} , we compare B_{ADC} with $B_{I(\text{col})}$ in Fig. 2.8, where we find that $B_{\text{ADC}} < B_{I(\text{col})}$ and in fact in many cases $B_{\text{ADC}} \ll B_{I(\text{col})}$. In an ideal noiseless scenario, i.e., $\text{SNDR}_a \rightarrow \infty$, one expects to see $\text{SNDR}_{\text{col}} \rightarrow \infty$ when $B_{\text{ADC}} \geq B_{I(\text{col})}$. Since in practice $B_{\text{ADC}} \leq B_{I(\text{col})}$ as indicated by Fig. 2.8, bank-level compute accuracy SNDR_{col} is most likely dominated by SNDR_{ADC} (e.g., < 20 dB in [81, 89]), and, hence IMC designers rely on the inherent error-tolerance of machine learning algorithms and QAT to achieve the desired level of network accuracy.

2.5.4 Network Accuracy (P_d)

Since it is challenging to design IMCs with sufficient capacity to implement an entire modern deep net, IMC papers report measured network level accuracies by mapping a few layers on the IC with the rest implemented off-chip. This decouples the reported accuracy and the reported energy efficiency metrics,

making it hard to judge the true performance of the design. Note that this off-chip processing can also be leveraged to compensate for errors due to IMC’s analog computation and artificially boost network accuracy. This is one more reason to measure and report SNDR_{col} as suggested in Section 2.5.3.

Another issue with reporting IMC accuracy is the difficulty of testing over an entire data set. Today’s solution is to downsample the dataset. However, 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 SNDR 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 SNDR.

A more serious issue is that most IMC papers do not report the number of input samples used during chip testing to obtain network accuracy. This is a problem because the measured network accuracy $\hat{p}_d$ (probability of correct decision) given by:

$$\hat{p}_d = \frac{N_c}{N_t} \quad (2.7)$$

where N_c and N_t are the number of correctly classified and the total number input test images, respectively, is an *estimate* of the unknown *true accuracy* p_d . In a limited sample scenario, the measured accuracy $\hat{p}_d$ is a random variable with a mean and variance. Therefore, 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.

Since the number of test samples is limited, the measured accuracy $\hat{p}_d$ incurs a large variance since it is one sample of a potentially large variance random variable. One way around this problem is for IMC papers to report the confidence interval ($0 < C_I < 1$) and confidence level ($0 < C_L < 1$) when reporting $\hat{p}_d$, e.g., an “accuracy of $90\% \pm 1\%$ at a confidence level of 95%” implies that the true IMC accuracy p_d lies within an interval of $\pm 1\%$ around $\hat{p}_d$ with probability 0.95. 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 $\hat{p}_d \pm C_I$ with a confidence level of C_L is given by:

$$M > \frac{1}{4(1 - C_L)C_I^2} \quad (2.8)$$

Equation (2.8) indicates that C_I and C_L trade-off with each other when the number of test samples M is fixed. Figure 2.10 shows that we need to test with at least 50,000 test images to claim that the true accuracy lies within $\pm 1\%$ (confidence interval) of the measured 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.

2.5.5 IMC Benchmarking Dataset

Finally, it is highly recommended that the IMC community establish a benchmark dataset 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 and energy efficiency.

2.6 Discussion and Summary

This chapter 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 50+ IMC IC works that have appeared in CICC, VLSI, ISSCC (and ESSCIRC) since 2018. Comparisons were made between SRAM-, eNVM-based, and recent digital accelerators. The key findings of our work are summarized below:

1. Bank-level SRAM-based IMC design methods are mature and have demonstrated clear wins (see Fig. 2.11) in terms of energy efficiency

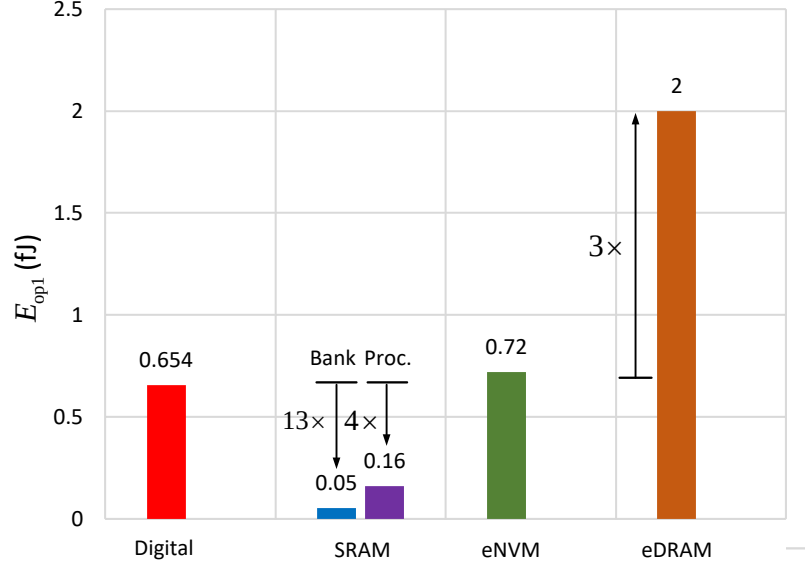


Figure 2.11: Energy per 1b OP comparison

($13 \times$ in 1b-TOPS/W) and compute density ($6.6 \times$ in 1b-TOPS/mm²) over digital accelerators when the comparisons are made at the bank-level. However, the bank-level gap in energy efficiency reduces to $< 4 \times$ when comparing IMC processors with digital accelerators. This gap will change as IMC and digital accelerator designers continue to innovate.

2. eNVM and eDRAM-based IMCs lag significantly behind (see Fig. 2.11 and (2.3)) both SRAM-based IMCs and digital accelerators in terms of energy efficiency and compute density. A key reason being the challenge in achieving high row-parallelism which limits the TOPS and hence TOPS/W. For eNVM-based IMCs, high row-parallelism is difficult to realize because it limits the achievable compute SNDR.
3. Despite their analog-heavy computation and contrary to conventional wisdom, SRAM-based IMCs benefit from technology scaling (see Fig. 2.2), i.e., their energy efficiency and compute density both improve in advanced nodes. Furthermore, 6T+ SRAM bitcells tend to enhance energy efficiency in the same technology node by enabling high row-parallelism but at a cost in compute density.
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 SNDR SNDR_{col} need to be quantified and employed to predict accuracy of networks being mapped.

5. Network accuracy needs to be accompanied by the associated confidence level and confidence interval to quantify statistical significance. 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 per (2.8).
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 and confidence interval. Such a step will ensure uniformity and confidence in the reported IMC accuracies.

One criticism of this work is that the causes underlying the data-driven trends presented here have not been fully investigated or elaborated upon. Doing so requires a deeper study of various IMC compute models and circuit methods (see [146, 147]), which is a good topic for future research.

CHAPTER 3

ENERGY-ACCURACY TRADE-OFFS IN RESISTIVE IN-MEMORY COMPUTING ARCHITECTURES

In this chapter, we provide a rationale for the energy efficiency (TOPS/W) and compute density (TOPS/mm²) gap between resistive IMCs and SRAM-based IMCs and digital accelerators, as shown in **Chapter 2**. A key reason for this gap is due to the need for using high-sensitivity, hence large area, read-out circuitry, to sense small changes in the column conductance, which is already quite small due to the highly parallelized read access pattern of IMCs, e.g., detecting a $\Delta G = 80 \mu\text{S}$ conductance change in a nominal array resistance of $G = 20 \text{ mS}$ corresponding to a unit change in the output of a high-dimensional (128) dot-product (DP). As a result, the peripherals occupy the bulk of the total IMC area, e.g., $> 90\%$ in [26], thereby compromising the compute density. Furthermore, the sensing problem becomes even more acute if the array is operated at low current levels in order to maximize energy efficiency, e.g., to achieve an energy-per-1b operation (E_{op1}) of 0.5 fJ in computing a 128-dimensional binary DP requires sensing with a $< 10 \mu\text{A}$ bit-line current and a $< 1 \text{ mV}$ bit-line voltage. This is because the presence of analog non-idealities such as bit-cell conductance variations, wire parasitics, current mirror mismatch, and analog-to-digital-converter (ADC) noise conspire to reduce the computational accuracy (compute accuracy). This is in contrast to digital accelerators, where compute accuracy can be enhanced by increasing the bit precision of the arithmetic. Therefore, achieving the dual objectives of energy efficiency and compute accuracy is a fundamental challenge faced by resistive IMCs today.

Today, the intrinsic compute accuracy problem of resistive IMCs is overcome by exploiting the intrinsic error-tolerance of deep neural networks coupled with the use of noise-aware training (NAT) [19, 23, 148], and by reporting network-level accuracy instead of at the bank-level. This approach is network and data-set specific, making it hard to generalize across tasks, and it does not address the key underlying reasons for the low computational

accuracy of resistive IMCs. Moreover, developing a comprehensive understanding of the limits on computational accuracy of resistive IMCs is critical in order to be able to scale-up today’s single-bank macro-level designs to multi-bank system architectures.

Resistive IMCs can be classified into two types of architectures: cross-bars [30, 41, 52, 53, 54, 55, 34], where bit-lines (BLs) cross source-lines (SLs) with the bit-cell (BC), and parallel-bars [31, 37, 38, 57, 15, 23, 24, 39, 40, 26, 59, 60, 43], where BLs run parallel to SLs with BC between them. While there has been some work on analyzing the compute accuracy of resistive *crossbars* architectures [44, 45, 46, 47, 48, 49, 50, 51] due to their popularity in the device community, not much work has been done to develop a comprehensive understanding of the compute accuracy limits of resistive parallel-bar architectures which are more practical due to the ease of scalability arising from the intrinsic isolation between ADC columns.

In this work, we address this modeling gap by proposing a compute signal-to-noise-plus-distortion ratio (SNDR) metric to quantify the bank/array-level compute accuracy of a resistive parallel-bar IMC (henceforth referred to as resistive IMC) [149]. Thereafter, we extend the analysis to crossbar architectures verified with circuit simulations in 22 nm process node (referred to as resistive crossbars).

Specifically, we make the following contributions:

- We develop a behavioral model of a resistive IMC bank to evaluate its SNDR, and validate this model against measured results from our recent 22 nm MRAM-based IMC [58] prototype. This model is available at: <https://github.com/saionkr2/eNVM-IMC-Modeling>.
- We employ the behavioral model to study the dependence of compute SNDR on various design parameters at the device-level (conductance contrast), circuit-level (reference voltage, ADC precision), and the architectural-level (DP dimension).
- We consider MRAM-, ReRAM-, and FeFET-based IMCs, and demonstrate that FeFET achieves the highest SNDR followed by ReRAM and MRAM when the DP dimension $N > 50$.
- We show, somewhat surprisingly, that the maximum achievable SNDR for resistive IMCs lies in the range of 18 dB-to-22 dB, and that this

SNDR is achieved when the device conductance contrast ($g_{\text{on}}/g_{\text{off}}$) increases to 12. This result points to the importance of reducing the impact of conductance variations and wire parasitics in order to reap the benefits of high conductance contrast.

- We quantify the energy-SNDR trade-off at the bank-level and the energy-accuracy trade-off at the system level for a ResNet-20 (CIFAR-10) and find that the IMC design parameters that maximize the bank-level SNDR are close to those that maximize the network level accuracy.
- Lastly, we extend our analysis to the resistive crossbars to obtain the dependence of the bank-level SNDR on device, circuit, and architecture parameters. Similar to parallel-bars, the maximum SNDR is limited by the pre-ADC non-idealities. Furthermore, our results indicate that the SNDR is predictive of network-level accuracy for both architectures.

3.1 Background and Preliminaries

We provide the background in terms of the structure and function of parallel-bar resistive IMC architecture commonly used today [31, 37, 38, 57, 15, 23, 24, 39, 40, 26, 58, 59, 60, 43].

3.1.1 Architecture Overview

Figure 3.1 shows a resistive parallel-bar architecture comprising of M ADC columns with bitlines (BLs) parallel to sourcelines (SLs) connecting 2T2R *differential* bitcells (BC). Binary differential inputs are given along the wordlines (WLs) pairs which drive the gate of NMOS access transistor in the BCs. The BCs store the weight values in terms of the cell conductance value G_{bc} and its complement G_{bcb} .

We choose the 2T2R architecture because it is commonly employed in high-dimensional parallel-bar architectures today [23, 24, 39, 26, 59]. In contrast to 1T1R architectures [40], a 2T2R architecture provides a linear mapping between the column conductance and DP output, and permits the representation of positive and negative inputs and weights in a single physical column instead of two physical columns required in 1T1R designs [15]. An

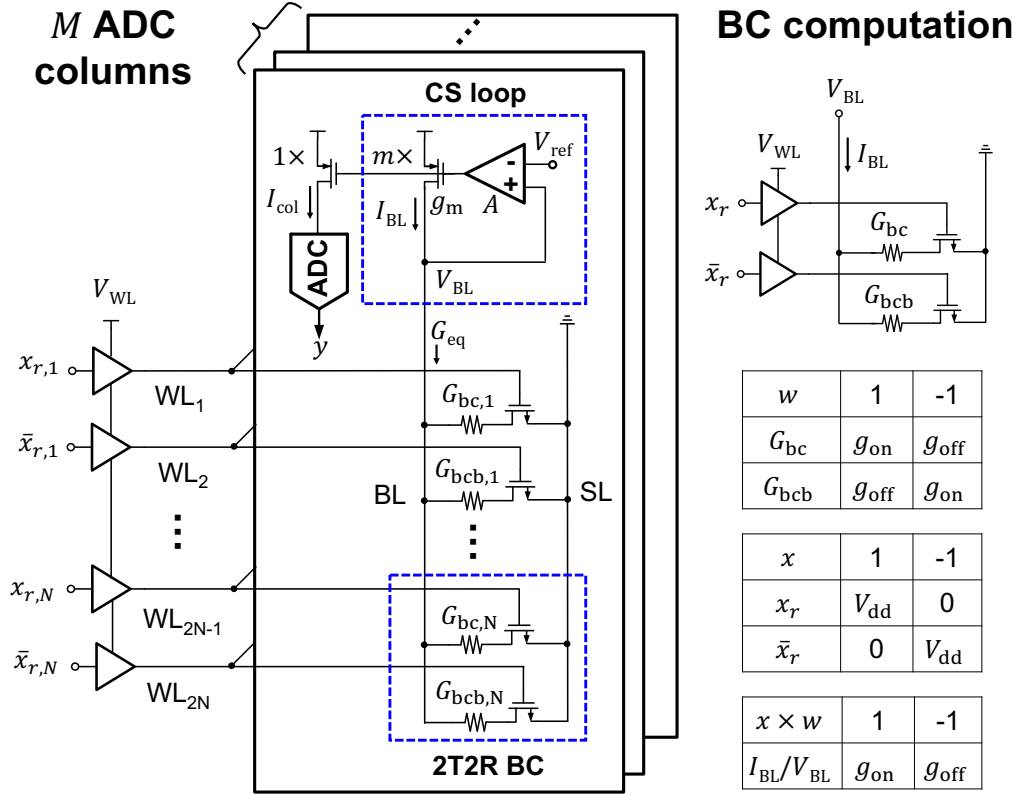


Figure 3.1: Resistive parallel-bar IMC architecture.

ADC column strives to compute the ideal DP y_o given by:

$$y_o = \mathbf{w}^T \mathbf{x}, \quad (3.1)$$

where $\mathbf{x}$ and $\mathbf{w}$ are the N -dimensional binary antipodal input and weight vectors, respectively, i.e., $x_k, w_k \in \{\pm 1\} (k = 1, \dots, N)$.

The current sensing loop (CS loop) of the ADC column measures the column conductance G_{eq} by asserting a voltage V_{BL} on the BL and sensing the resulting current $I_{BL} = V_{BL}G_{eq}$. The BL current I_{BL} is mirrored to generate the input current I_{col} into the column ADC which is digitized into a B_{ADC} -bit output code $y = Q[I_{col}]$.

When the M ADC columns are activated simultaneously with the same input vector $\mathbf{x}$, the parallel-bar IMC computes M , N -dimensional DPs in parallel, or a $M \times N$ -dimensional matrix-vector multiplication (MVM) in one read cycle.

3.1.2 BC Computation

The parallel-bar architecture (see Fig. 3.1) maps the binary antipodal input and weights $x_k, w_k \in \{\pm 1\} (k = 1, \dots, N)$ in (3.1) to differential input $(x_{r,k}, \bar{x}_{r,k})$ and differential conductance $(G_{bc,k}, G_{bcb,k})$, respectively, as follows:

$$x_{r,k} = \frac{1 + x_k}{2}; \quad \bar{x}_{r,k} = \frac{1 - x_k}{2} \quad (3.2)$$

$$G_{bc,k} = g_{on} \left(\frac{1 + w_k}{2} \right) + g_{off} \left(\frac{1 - w_k}{2} \right) \quad (3.3)$$

$$G_{bcb,k} = g_{on} \left(\frac{1 - w_k}{2} \right) + g_{off} \left(\frac{1 + w_k}{2} \right) \quad (3.4)$$

where g_{on} (g_{off}) represents the high or ON (low or OFF) conductance state. The *conductance contrast* g_{on}/g_{off} ranges from 1.6 for MRAM with $g_{on} = 222 \mu S$ [32], 12 for ReRAM with $g_{on} = 40 \mu S$ [33] to 10^3 for FeFET with $g_{on} = 1 \mu S$ [34].

The input and weight-dependent *conductance state* of the k -th BC is denoted as $G_k \in \{g_{on}, g_{off}\}$, and is defined as:

$$G_k = \begin{cases} g_{on} & \text{if } w_k x_k = 1 \\ g_{off} & \text{if } w_k x_k = -1 \end{cases}. \quad (3.5)$$

e.g., if $x_k = +1$, then from (3.2), $x_{r,k} = 1$ and $\bar{x}_{r,k} = 0$, thereby selecting the upper 1T1R device of the k -th BC (see Fig. 3.1). From (3.3), the conductance $G_{bc,k}$ of the selected 1T1R device takes value g_{on} if $w_k = +1$ and g_{off} if $w_k = -1$.

3.1.3 ADC Column Computation

When an input vector $\mathbf{x} \in \{\pm 1\}^N$ is applied to a N -row ADC column, the column conductance G_{eq} (see Fig. 3.1) is given by:

$$G_{eq} = \sum_{k=1}^N G_k. \quad (3.6)$$

From (3.6), it is clear that G_{eq} takes discrete values in the range $[Ng_{\text{off}}, Ng_{\text{on}}]$ with a fixed step size $\Delta g = g_{\text{on}} - g_{\text{off}}$.

The CS loop senses G_{eq} by asserting a BL voltage V_{BL} close to V_{ref} with the SL grounded. A high loop gain $\text{LG} = Ag_m/G_{\text{eq}}$ in the feedback loop is required to ensure $V_{\text{BL}} = V_{\text{ref}} \times \text{LG}/(1 + \text{LG}) \approx V_{\text{ref}}$ across all M ADC columns, where A is the differential amplifier gain, and g_m is the transconductance of the PMOS current source. From (3.1) and (3.6), the ideal values of BL current I_{BL} , ADC column current I_{col} , and its range I_{FR} are given by:

$$I_{\text{BL}} = V_{\text{ref}}G_{\text{eq}}; \quad (3.7)$$

$$I_{\text{col}} = I_{\text{sig}} = \frac{I_{\text{BL}}}{m} \propto y_o; \quad (3.8)$$

$$I_{\text{FR}} = \frac{V_{\text{ref}}N\Delta g}{m}, \quad (3.9)$$

where m is the current scaling factor, and I_{sig} denotes the signal current. The ADC quantizes I_{sig} to generate an output code $y = Q[I_{\text{sig}}]$, which equals the ideal DP y_o in (3.1) when the ADC precision $B_{\text{ADC}} = 1 + \log_2(N)$ and step size $I_{\text{step}} = V_{\text{ref}}\Delta g/m$.

In practice, however, due to the presence of process variations and noise, $I_{\text{col}} \neq I_{\text{sig}}$, which leads to a finite compute SNDR at the IMC output. Next, we model the impact of these analog non-idealities on I_{col} in order to quantify their impact on the compute SNDR.

3.2 Non-ideal Behavior in Resistive IMC Architecture

Recent resistive IMC designs have experimentally observed the impact of analog non-idealities on their compute accuracy [23, 24, 39]. In this section, we model these non-idealities and analyze their impact on the compute accuracy.

Figure 3.2 shows that the dominant non-idealities in resistive IMCs comprises: 1) CV - spatial BC conductance variations (δG_{bc} , δG_{bcb}); 2) PC: non-zero parasitic conductances on the BL (g_{bl}) and SLs (g_{sl}); 3) MM: current mirror mismatch between ADC columns; 4) AN: ADC thermal (AT), quantization (AQ), and clipping (AC) noise; and 5) SD: supply voltage drop across ADC columns due to layout-dependent power grid resistance. In this

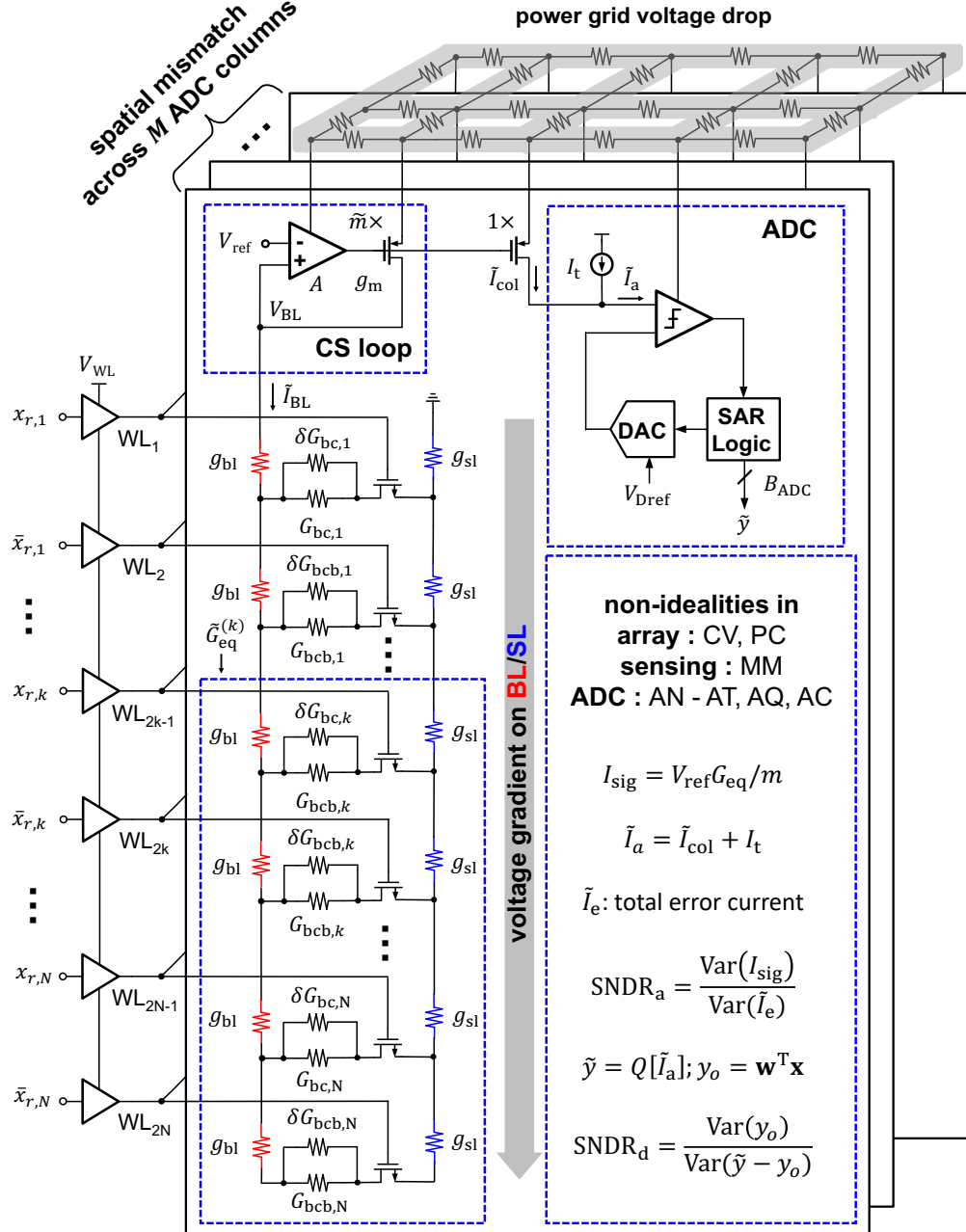


Figure 3.2: Non-idealities in resistive parallel-bar IMC.

work, we focus on the non-idealities 1)-4), since these are intrinsic to an ADC column, forming a fundamental compute unit of a resistive IMC [74].

Since our work considers inference mode operation only of resistive IMCs, we do not consider non-idealities such as cycle-to-cycle variations, which manifest during write access and can be remedied using adaptive write techniques [29].

3.2.1 Modeling Non-idealities

Non-idealities in an ADC column (see Fig. 3.2) can be clustered into **array non-idealities** (CV and PC), **CS loop non-idealities** (MM), and **ADC non-idealities** (AN). Figure 3.3 shows the individual impact of the ADC non-idealities on the column current distribution corresponding to the ideal DP y_o (3.1) of dimension N , assumed to be the Binomial distribution $\text{Bi}(N, 0.5)$. Each DP code c has multiple weight and activation pairs corresponding to a unique circuit configuration and hence a unique state $\mathcal{S}^c = \{(\mathbf{x}, \mathbf{w}) : \mathbf{w}^T \mathbf{x} = c\} \subset \mathcal{S}$, where $\mathcal{S}$ is the set of all possible states obtained by choosing $\mathbf{w}, \mathbf{x} \in [\pm 1]^N$. Next, we describe the impact of non-idealities on the ideal current states.

Array Non-idealities

Conductance variations due to manufacturing tolerances within each BC and non-zero BL/SL wire parasitic conductances transform the ideal column conductance G_{eq} in (3.6) into its non-ideal value $\tilde{G}_{\text{eq}}$.

Conductance variations (CV) in the k -th BC δG_k can be modeled as [150]:

$$\delta G_k \sim \begin{cases} \mathcal{N}(0, \sigma_{\text{off}}^2) & \text{if } G_k = g_{\text{off}}; \\ \mathcal{N}(0, \sigma_{\text{on}}^2) & \text{if } G_k = g_{\text{on}}; \end{cases} \quad (3.10)$$

where it has been observed that the standard deviations of σ_{off} and σ_{on} are proportional to g_{off} and g_{on} , respectively. This leads to a spread in the ideal current levels, where the middle codes (Fig. 3.3) have the highest spread since the number of states for those codes is the largest, e.g., for $c = 32, N = 64$, $|\mathcal{S}^c| \approx 10^{18}$.

Additionally, the presence of parasitic conductances (PC) on the BL and SL viz. g_{bl} and g_{sl} , respectively, distorts the linear relationship between the column conductance G_{eq} and BC conductance state G_k in (3.6) to generate a non-ideal column conductance $\tilde{G}_{eq}$.

In order to capture this non-linear effect, we compute $\tilde{G}_{eq}$ via the following descending recursion ($k = N, \dots, 1$):

$$\tilde{G}_{eq,k} = \begin{cases} \left(\tilde{G}_{eq,k+1} \parallel (g_{bl} \parallel g_{sl}) \right) \parallel (g_{bl} \parallel g_{sl}) \\ \quad + \tilde{G}_k \parallel (g_{bl} \parallel g_{sl}) & \text{if } x_k = 1 \\ \left(\tilde{G}_{eq,k+1} + \tilde{G}_k \right) \parallel \frac{(g_{bl} \parallel g_{sl})}{2} & \text{if } x_k = -1 \end{cases}, \quad (3.11)$$

where $\tilde{G}_{eq,k}$ is the equivalent conductance of the array looking down into the terminals of the k -th BC (see Fig. 3.2), $\tilde{G}_{eq,N+1} = 0$, and $\tilde{G}_k = G_k + \delta G_k$ is the non-ideal conductance state of the k -th BC. Note that $\tilde{G}_{eq} = \tilde{G}_{eq,1}$. The derivation for (3.11) is shown in Section 3.6.2.

The presence of wire parasitics PC has the effect of shrinking the signal range below its ideal value I_{FR} (3.9), due to a drop in the equivalent conductance $\tilde{G}_{eq} < G_{eq}$ (3.11). As seen in Fig. 3.3, the signal range shrinks as the DP dimension N increases. Furthermore, a spread in the output currents begins to appear, with the spread for the middle codes being the highest due to the large number of contributing states. In the limiting case when N approaches infinity, the non-ideal column conductance $\tilde{G}_{eq}$ (3.11) approaches $\sqrt{(g_{bl} \parallel g_{sl}) g_{on}}/2$ (shown in Section 3.6.2), thereby limiting the maximum attainable SNDR in resistive IMC architectures. This contrasts with the linear dependence of the ideal column conductance on N , i.e., $G_{eq} \propto N$.

CS Loop Non-idealities

The non-idealities in the current sensing loop arise from the mismatch in the current mirror transistor (see Fig. 3.2) that generates a non-ideal ADC input current $\tilde{I}_{col}$, which appears as an overall mismatch across ADC columns.

The CS loop generates a non-ideal BL current $\tilde{I}_{BL} = V_{ref} \tilde{G}_{eq}$, which is scaled down by a factor of m via a $m : 1$ PMOS current mirror (see Fig. 3.2) to generate the input current $\tilde{I}_{col}$ into the ADC. We assume that W/L mismatch in a $1 : 1$ current mirror converts it into a $1 : 1 + \eta$ current mirror, where

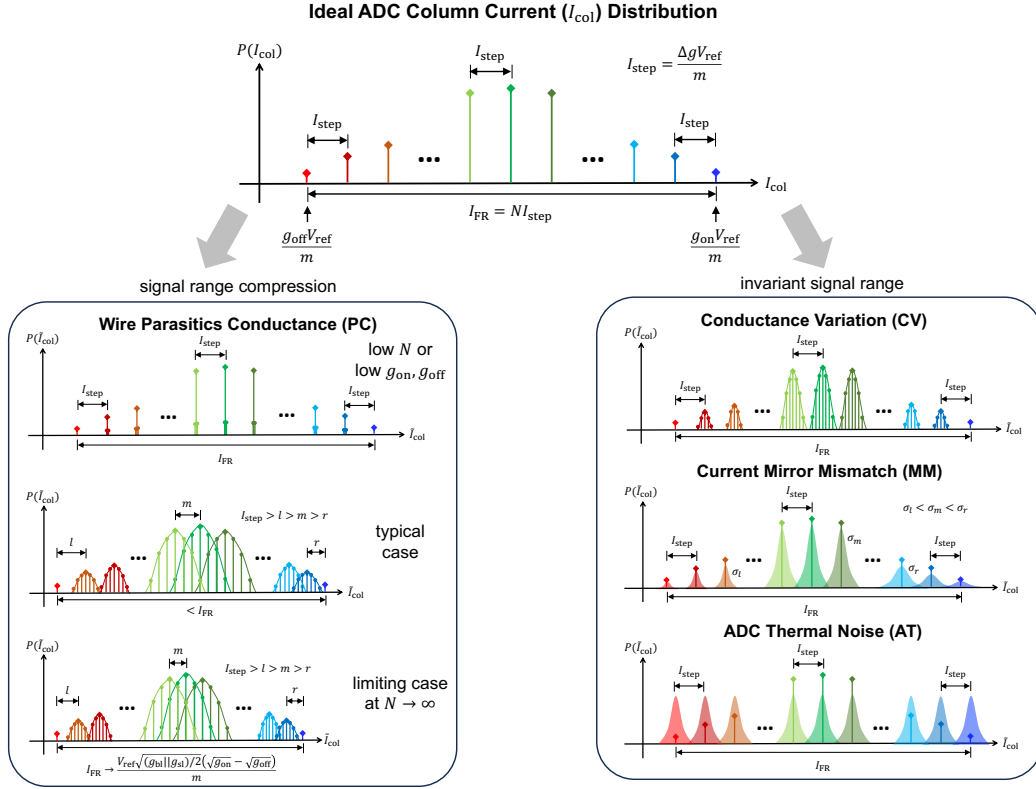


Figure 3.3: Impact of analog non-idealities on ADC column current distribution in resistive IMC architectures.

$\eta \sim \mathcal{N}(0, \sigma_f^2)$ is the mismatch in a unit finger. Assuming further that the mismatch across the m fingers of the current mirror is uncorrelated, we obtain the following ADC column current $\tilde{I}_{\text{col}}$ model (derived in Section 3.6.3):

$$\tilde{I}_{\text{col}} = \frac{\tilde{I}_{\text{BL}}}{\tilde{m}} = \frac{\tilde{G}_{\text{eq}} V_{\text{ref}}}{m(1 + \eta/\sqrt{m})}. \quad (3.12)$$

where $\tilde{m} : 1$ is the current mirroring ratio with a mismatch, and $\tilde{G}_{\text{eq}}$ is given by (3.11). This mismatch is a multiplicative non-ideality appearing across ADC columns, leading to a spread for each DP code in proportion to the corresponding current (Fig. 3.3). Thus, all the key non-idealities in the circuit blocks preceding the ADC are accounted for.

ADC Non-idealities

We consider a successive approximation (SAR) ADC since it is the most commonly used ADC architecture in IMC designs due to its compact layout and energy efficiency. As mentioned earlier, we model three noise sources: 1) thermal, 2) quantization, and 3) clipping noise, and quantify their impact while converting $\tilde{I}_{\text{col}}$ to the output code $\tilde{y}$.

Thermal noise in the ADC comparator (see Fig. 3.2) is well-modeled [151] as a Gaussian distributed noise current $I_t \sim \mathcal{N}(0, \sigma_t^2)$ at the ADC input as follows (Fig. 3.3):

$$\tilde{I}_a = \tilde{I}_{\text{col}} + I_t; \quad \tilde{y} = Q[\tilde{I}_a], \quad (3.13)$$

where $\tilde{I}_a$ is the ADC input current, Q is the quantizer function, and $\tilde{y}$ is the ADC output code. The quantizer introduces two additional noise sources – quantization and clipping noise – described next.

Since the ideal output y_o (see (3.1)) of high-dimensional dot products tend to have large peak-to-average ratios ($\max(y_o)/\sigma_{y_o}$), it is now a common practice [26] to clip the input range of the ADC input to $[I_{\text{clp}}^{(\min)}, I_{\text{clp}}^{(\max)}]$. In doing so, the most informative part of the input signal is uniformly quantized while clipping the extreme (unlikely) values, thereby maximizing the signal-to-quantization noise ratio (SQNR). Previous work [144] has shown the theoretically optimum (in the SQNR sense) clipping level for a zero mean Gaussian distributed signal y_o lies between $3\sigma_{y_o}$ -to- $4\sigma_{y_o}$ from the mean and is a function of B_{ADC} .

Table 3.1: Device parameters for resistive IMC architecture.

Parameter	Variable	Value		
		MRAM [32]	ReRAM [33]	FeFET [34]
on-state conductance	$g_{\text{on}} (\mu\text{S})$	222	40	1
off-state conductance	$g_{\text{off}} (\mu\text{S})$	133	3.3	0.001
BC conductance variation	$\left(\frac{\sigma}{\mu}\right)_{\text{bc}} (\%)$	4 [32, 33]		

Table 3.2: Behavioral model parameters for resistive IMC architecture.

Parameter	Variable	Value
BL wire conductance	$g_{\text{bl}} (\text{S})$	2.85
SL wire conductance	$g_{\text{sl}} (\text{S})$	6.25
CS loop amplifier gain	$A (\text{dB})$	60
CS loop PMOS g_m	$g_m (\mu\text{S})$	100
mirroring ratio	m	20
mirroring mismatch	$\sigma_f (\%)$	5
ADC thermal noise	$\sigma_t (\text{nA})$	6.4

In this work, we sweep the clipping levels symmetrically around the mean $\mathbb{E}[\tilde{I}_{\text{col}}]$ to obtain the SNDR-optimal clipping levels. Consistent with [144], we find that the optimum clipping levels $I_{\text{clp}}^{(\text{min})}$ and $I_{\text{clp}}^{(\text{max})}$ lie between 3-to-4 standard deviations away from the mean $\mathbb{E}[\tilde{I}_{\text{col}}]$. This form of clipped quantization is realized in a SAR ADC [24] by tuning the digital-to-analog converter (DAC) LSB size. The SAR logic iterates through B_{ADC} cycles to obtain a B_{ADC} -bit output $\tilde{y}$ whose analog equivalent $\text{DAC}(\tilde{y})$ is closest in value to the ADC input $\tilde{I}_{\text{a}}$. Note: Our work assumes that commonly employed design methods such as transistor up-sizing, time averaging, and ADC post-processing are been employed to alleviate device-level noise such as thermal, shot, and flicker noise in the array (ADC thermal noise is included). Therefore, we focus primarily on the circuit/architectural non-idealities that limit compute SNDR.

3.2.2 Compute SNDR

We define two compute SNDR metrics: 1) the analog pre-ADC SNDR (SNDR_{a}), and 2) the post-ADC digital SNDR (SNDR_{d}) defined below:

$$\text{SNDR}_{\text{a}} = \frac{\text{Var}(I_{\text{sig}})}{\text{Var}(\tilde{I}_{\text{e}})}; \quad \text{SNDR}_{\text{d}} = \frac{\text{Var}(y_{\text{o}})}{\text{Var}(e)}, \quad (3.14)$$

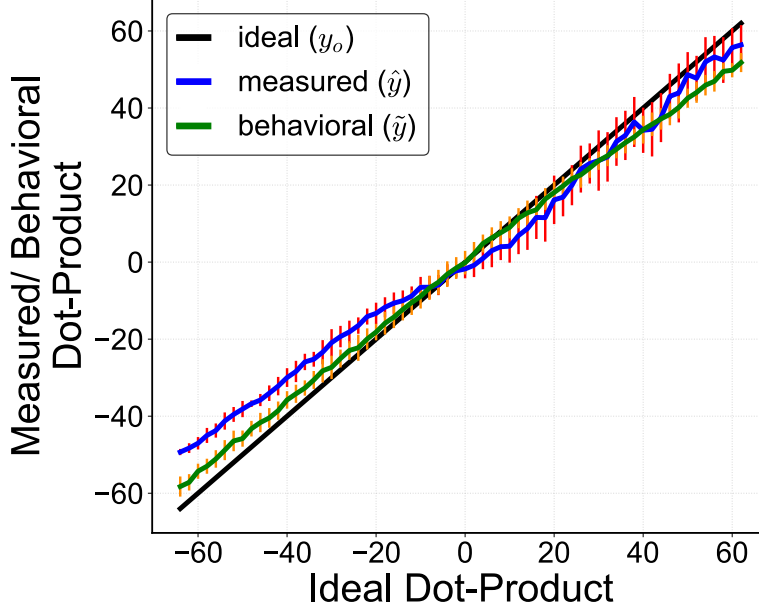


Figure 3.4: Behavioral model validation showing the 1σ error-bar of the behavioral model output $\tilde{y}$ (using parameters in Table 3.2) correlates well with the measured output $\hat{y}$ of the MRAM-based IMC prototype [58] in 22 nm FDSOI process, and the ideal value y_o , with $N = 64$, $B_{\text{ADC}} = 6$, and $V_{\text{ref}} = 20$ mV.

where $\text{Var}(x)$ is the variance of the random variable x , I_{sig} is the ideal signal current (3.7), y_o is the ideal DP output (3.1), $\tilde{I}_e$ is the total current error defined as the difference between the signal current I_{sig} and a range-adjusted non-ideal current at the input of the ADC obtained by matching the mean and the variance of $\tilde{I}_a$ to I_{sig} . Similarly, the error e for computing SNDR_d , is defined as:

$$e = \tilde{y} - y_o. \quad (3.15)$$

where $\tilde{y}$ is the actual DP output and y_o is the ideal output. Note: y_o also represents the output of a fixed-point digital architecture with bit-growth at the output, i.e., with an output precision of $1 + \log_2(N)$.

The analog SNDR, SNDR_a , factors out the impact of the ADC on the compute accuracy so as to provide a clear picture of the impact of array noise and variations on compute accuracy. Furthermore, since $\text{SNDR}_a \geq \text{SNDR}_d$, SNDR_a sets an upper bound on SNDR_d [152]. Thus, the compute SNDR (SNDR_a and SNDR_d) are a task-agnostic bank-localized metrics of accuracy that IMC designers can use to ensure that their IMC designs deliver the

desired network level accuracy, e.g., classification accuracy.

3.2.3 Resistive IMC Behavioral Model and its Validation

Section 3.2.1 collectively describe a behavioral model of a general resistive IMC. A Python code of our model can be downloaded from <https://github.com/saionkr2/eNVM-IMC-Modeling>. Tables 3.1 and 3.2 list the default device and typical model parameters employed in the behavioral model.

Figure 3.4 shows that our behavioral model’s predicted output $\tilde{y}$ of an ADC column is highly correlated with the measured output $\hat{y}$ of the MRAM-based IMC prototype [58] fabricated and tested in a commercial 22nm FDSOI process. Both the simulation and experimental data were obtained under identical conditions, i.e., employing: 1) a fixed weight vector $\mathbf{w}$; 2) random input vectors $\mathbf{x}$ which span the full range $[-N, N]$ of the ADC output (30 inputs per output); and a 3) fixed output offset and gain calibration [58].

3.3 Simulation Results

This section employs the behavioral model presented in 3.2.3, with model parameter values from Tables 3.1 and 3.2 for compute SNDR analysis. For the sake of fair comparison, we employ the behavioral model parameters in Table 2 for all three device types, assuming they are fabricated in the same CMOS node and that the differences in their BEOL processes have negligible impact. The parasitic conductances that arise from long BL/SL interconnects within the memory array are assumed to be fixed for a specific process node. In the following, all compute SNDR_a and SNDR_d values are obtained empirically by averaging the behavioral model predictions over 100k samples.

3.3.1 Simulation Methodology

We employ the simulation methodology described in Fig. 3.5 for evaluating bank-level compute SNDR (both SNDR_a , SNDR_d) and network-level accuracy. First, we choose a resistive IMC architecture with an antipodal weight

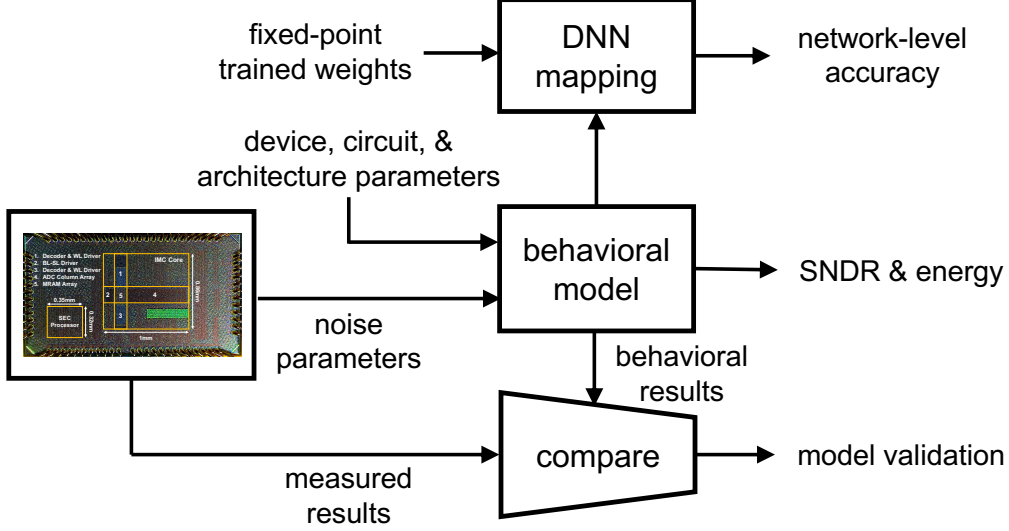


Figure 3.5: Simulation Methodology.

and input encoding, as shown in Fig. 3.1. The behavioral model is implemented using Python to emulate the behavior of the resistive array, the sensing circuit, and ADC. In order to verify our behavioral model, we use the noise and design parameters from our 22 nm chip [58] (3.2) as inputs to the model. We then compare the dot-product outputs from our behavioral model and the measured results from the chip. Thereafter, the behavioral model takes user-defined design parameters and computes the SNDR over 100 k instances of inputs and weights. The energy per 1 b operation E_{op1} is computed for the given design using the energy model described in (2.3) of **Chapter 2**.

Next, we analyze the impact of analog non-idealities on the network-level accuracy while mapping DNNs to resistive IMCs. First, we perform DNN training with fixed-point inputs and weights using Pytorch libraries. Thereafter, we map the layer-wise weights to resistive IMCs defined with noise parameters in Table 1 and user-defined design parameters. The final network-level accuracy is evaluated over the CIFAR-10 dataset with the ResNet-20 network mapped to resistive IMC.

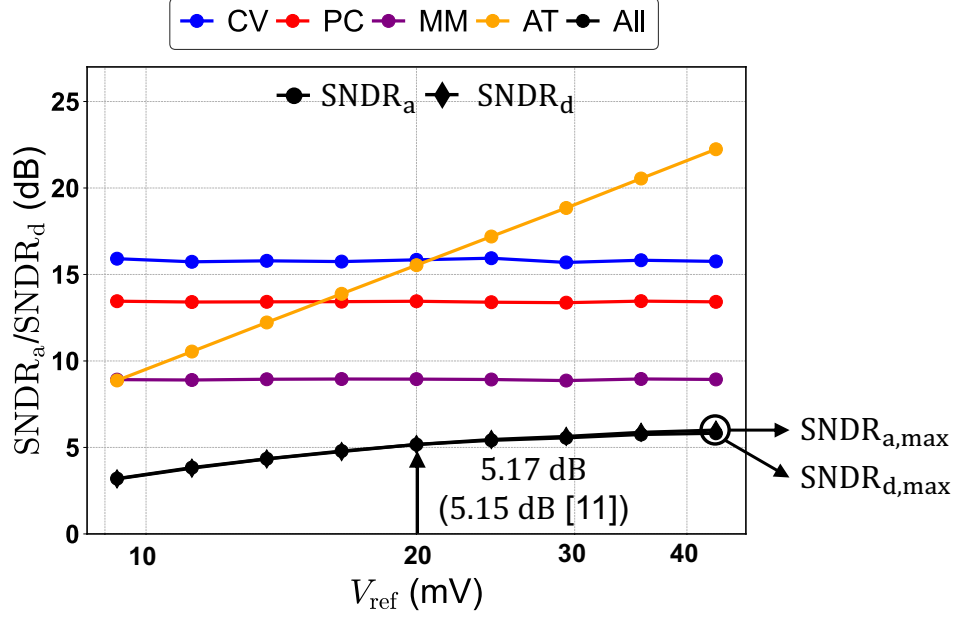


Figure 3.6: Impact of individual non-idealities on the compute SNDR of an MRAM IMC having ADC column with $N = 64$ and $B_{\text{ADC}} = 6$.

3.3.2 SNDR vs. V_{ref}

The reference voltage V_{ref} in the CS loop is a critical IMC design parameter since, as (3.7) and (3.14) indicate, V_{ref} directly sets the signal current I_{sig} and hence the SNDR and energy consumption.

Figure 3.6 studies the impact of the non-idealities modeled in Section 3.2 on the SNDR as a function of V_{ref} . We find that the compute SNDR_a in the presence of non-idealities from the array (CV and PC) and the CS loop (MM) remains more or less constant as V_{ref} , and hence I_{sig} , is increased. The reason being that the array and CS loop non-idealities share the same path as the input signal, and hence, any attempt to scale-up the signal current I_{sig} (numerator for SNDR_a in (3.14)) proportionally scales-up the denominator thereby keeping the SNDR_a constant.

However, in the presence of ADC thermal noise (orange curve), SNDR_a monotonically increases with V_{ref} until it begins to level off when $V_{\text{ref}} \geq 25$ mV. This trend occurs because thermal noise I_t is independent of the signal (see (3.13)), because it is the dominant noise source at low signal (and V_{ref}) levels, and because the array and CS loop non-idealities begin to dominate at high V_{ref} , thereby limiting SNDR_a to SNDR_{max} = 6 dB.

In summary, the non-idealities, in decreasing order of their impact on

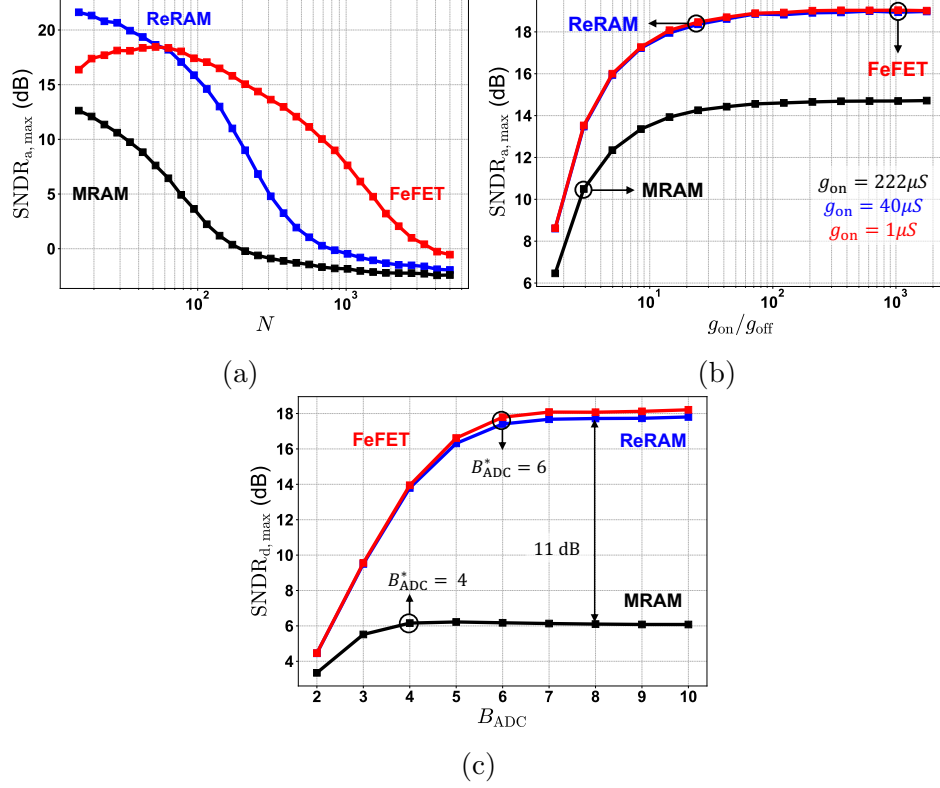


Figure 3.7: Maximum SNDR trends for ReRAM, MRAM and FeFET: (a) $SNDR_{a,max}$ vs. N , (b) $SNDR_{a,max}$ vs. g_{on}/g_{off} for $N = 64$, and (c) $SNDR_{d,max}$ vs. B_{ADC} for $N = 64$.

$SNDR_a$, are: 1) current mirror mismatch (MM), 2) BL/SL parasitic wire conductances (PC), and 3) conductance variations (CV), with the exception that the ADC thermal noise (AT) dominates only at lower values of V_{ref} . Furthermore, Fig. 3.6 also shows that $SNDR_d$ is upper bounded by $SNDR_a$.

The $SNDR_d$ measurement from the prototype MRAM-based IMC [58] (5.15 dB) at $V_{ref} = 20$ mV was found to be close to the value predicted by the behavioral model (5.17 dB), which further provides confidence in our behavioral model and its predictions.

3.3.3 $SNDR_a$ vs. Dot-product Dimension N

It is quite well-known [152] that the compute SNDR of any IMC, either SRAM- or eNVM-based, trades-off with the dot-product dimension N . This trade-off occurs primarily because IMCs pack multiple output levels within a limited (current or voltage) swing range. Hence, we study this trade-off for

MRAM-, ReRAM-, and FeFET-based resistive IMCs.

Figure 3.7(a) plots the maximum achievable value of SNDR_a ($\text{SNDR}_{a,\max}$), obtained by sweeping V_{ref} as done in Fig. 3.6, for the three (MRAM-, ReRAM-, and FeFET-based) resistive IMCs, as a function of N . All three exhibit a roll-off in $\text{SNDR}_{a,\max}$ with an increase in N . This roll-off in $\text{SNDR}_{a,\max}$ occurs because G_{eq} , and hence I_{sig} per (3.6)-(3.7), increases linearly with N , but $\tilde{G}_{\text{eq}}$ (due to PC) from (3.11) increases sub-linearly with N , thereby increasing the gap between the ideal signal current I_{sig} and the actual current $\tilde{I}_{\text{col}}$ (see (3.12)) into the ADC input. This is also indicated in Figure 3.3 (left) shows this phenomenon where the signal separation I_{step} reduces as N increases.

Specifically, $\text{SNDR}_{a,\max}$ peaks at $N = 50$ for FeFET-based architectures before rolling off. In contrast, MRAM- and ReRAM-based architectures see a $\text{SNDR}_{a,\max}$ roll-off starting from $N \geq 5$. The differences in the roll-off effect between the three architectures can be explained from (3.11) and Table 3.1 and 3.2. At low values of N , the signal current $\tilde{I}_{\text{col}}$ for the FeFET-based architecture is small due to the small device conductance (g_{on} and g_{off} per Table 3.1) compared to $(g_{\text{bl}}||g_{\text{sl}})/2$ ($= 1\text{ S}$ per Table 3.2). Thus, $\text{SNDR}_{a,\max}$ increases with N as the rate of increase in $\tilde{G}_{\text{eq}}$ w.r.t. N remain high (see (3.11)) until $\tilde{G}_{\text{eq}}$ becomes comparable with $(g_{\text{bl}}||g_{\text{sl}})/2$. In contrast, MRAM and ReRAM devices have conductances much greater than $(g_{\text{bl}}||g_{\text{sl}})/2$ and, therefore, experience a roll-off with N at smaller values.

3.3.4 SNDR_a vs. Conductance Contrast $g_{\text{on}}/g_{\text{off}}$

The conductance contrast ($g_{\text{on}}/g_{\text{off}}$) leads to a larger separation $\Delta g = g_{\text{on}} - g_{\text{off}}$ between the two states (see (3.6)), thereby enhancing the compute SNDR_a . At the same time, a lower g_{on} reduces the BL current I_{BL} at a fixed value V_{ref} , thereby increasing the energy efficiency. Hence, we now investigate the impact of conductance contrast and g_{on} on the compute SNDR_a .

Figure 3.7(b) shows that $\text{SNDR}_{a,\max}$ initially increases with the ratio $g_{\text{on}}/g_{\text{off}}$ for all three architectures but then saturates for $g_{\text{on}}/g_{\text{off}} > 12$, indicating the limited benefits of improving the conductance contrast ratio beyond ≈ 12 . The $g_{\text{on}}/g_{\text{off}}$ ratio in Fig. 3.7(b) is swept by keeping the on-state conductance g_{on} fixed and lowering the off-state conductance g_{off} . Doing so leads

to a low σ_{off} (see (3.10)), thereby reducing the conductance variation δG_k for BCs storing g_{off} . Additionally, lowering g_{off} increases Δg , leading to an increased separation in the SNDR_{a} optimal current levels, thereby reducing the error current $\tilde{I}_{\text{e}}$ (see (3.14)) and its variance. These effects initially lead to an increasing SNDR_{a} with the $g_{\text{on}}/g_{\text{off}}$ ratio. However, $\text{SNDR}_{\text{a,max}}$ begins to saturate as the conductance contrast increases beyond 12. This is because the signal range reaches its maximum of $[0, Ng_{\text{on}}]$ leading to constant signal and noise variance, hence SNDR_{a} , in (3.14). Furthermore, $\text{SNDR}_{\text{a,max}}$ for MRAM-based architecture is lower than that for the ReRAM-based one due to the smaller effect of wire parasitics at ReRAM's lower g_{on} (see Fig. 3.3(left)). However, comparing ReRAM to FeFET, even though g_{on} is lower for FeFET, $\text{SNDR}_{\text{a,max}}$ for both architectures is similar because the signal range reduces with smaller g_{on} , leading to an increased impact of thermal noise.

3.3.5 SNDR_{d} vs. ADC Precision B_{ADC}

ADC precision B_{ADC} for resistive IMCs is a critical design parameter that affects the overall compute density, energy efficiency, and compute SNDR_{d} . In this study, we determine SNDR_{d} -optimal clipping levels using the optimal clipping criterion (OCC) [144] in order to maximize the ADC's SQNR, and hence the compute SNDR_{d} of the IMC.

Figure 3.7(c) shows that $\text{SNDR}_{\text{d,max}}$ increases with B_{ADC} up to a point and then saturates. This trend, also observed in SRAM-based IMCs [144] and resistive crossbars [153], occurs because of the inherent trade-off between ADC quantization/clipping noise, which dominates at lower values of B_{ADC} , and the pre-ADC non-idealities (array and CS loop non-idealities, and ADC thermal noise), which dominate at higher values of B_{ADC} . Furthermore, the minimum ADC precision (B_{ADC}^*) before $\text{SNDR}_{\text{d,max}}$ saturates is 4 b for MRAM-based architectures and around 6 b for ReRAM and FeFET-based ones. This difference arises primarily due the high conductance and low conductance contrast of the MTJ devices (see Table 3.2), which limits the achievable SNDR_{d} , e.g., Fig. 3.7(c) shows a 11 dB – 12 dB gap between the MRAM-based and ReRAM and FeFET-based architectures.

Table 3.3: Energy model parameters.

Parameter	Variable		Value		
			MRAM	ReRAM	FeFET
supply voltage	V_{DD} (V)		0.8	0.8	5
average conductance	$\mathbb{E}[\tilde{G}_{\text{eq}}]$ (mS)	$N = 32$	5.3	0.68	0.016
		$N = 64$	9.2	1.3	0.032
		$N = 512$	13.0	4.5	0.245
latency	T_{core} (ns)		20		

3.3.6 SNDR_d vs. Energy-per-1b Operation

We use the generic IMC energy model in (2.3) of **Chapter 2** of compute energy per 1 b operation (OP) E_{op1} , as follows:

$$E_{\text{op1}} = \frac{E_{\text{bl}} + E_{\text{ADC}}}{2B_X B_W N} \quad (3.16)$$

where E_{bl} is the pre-ADC energy of the ADC column and sensing blocks, E_{ADC} is ADC energy, B_X , and B_W are the input and weight precision, respectively, and N is the DP dimension. The average pre-ADC energy is given by:

$$E_{\text{bl}} = V_{\text{DD}} V_{\text{ref}} \mathbb{E}[\tilde{G}_{\text{eq}}] T_{\text{core}} \quad (3.17)$$

where $\mathbb{E}[\tilde{G}_{\text{eq}}]$ is the non-ideal conductance $\tilde{G}_{\text{eq}}$ averaged over 100 k samples, and T_{core} is the read latency of the ADC column. We employ the ADC energy model from [154]:

$$E_{\text{ADC}} = k_1 B_{\text{ADC}} + k_2 4^{B_{\text{ADC}}} \quad (3.18)$$

where $k_1 = 100$ fJ, $k_2 = 1$ aJ are the empirically obtained parameters from published ADCs [155, 156]. Table 3.3 lists the energy model parameters including, V_{DD} per device type [34, 32, 33], $\tilde{G}_{\text{eq}}$ computed from the behavioral model, and T_{core} from [58].

Figure 3.8 shows that SNDR_d increases with E_{op1} for all three device types and for all values of DP dimension N , thereby pointing to the well-known accuracy vs. energy-efficiency trade-off in IMCs. More interestingly, the SNDR_d stops improving even if E_{op1} is allowed to increase beyond a certain point. This upper bound in SNDR_d occurs when non-idealities other than the

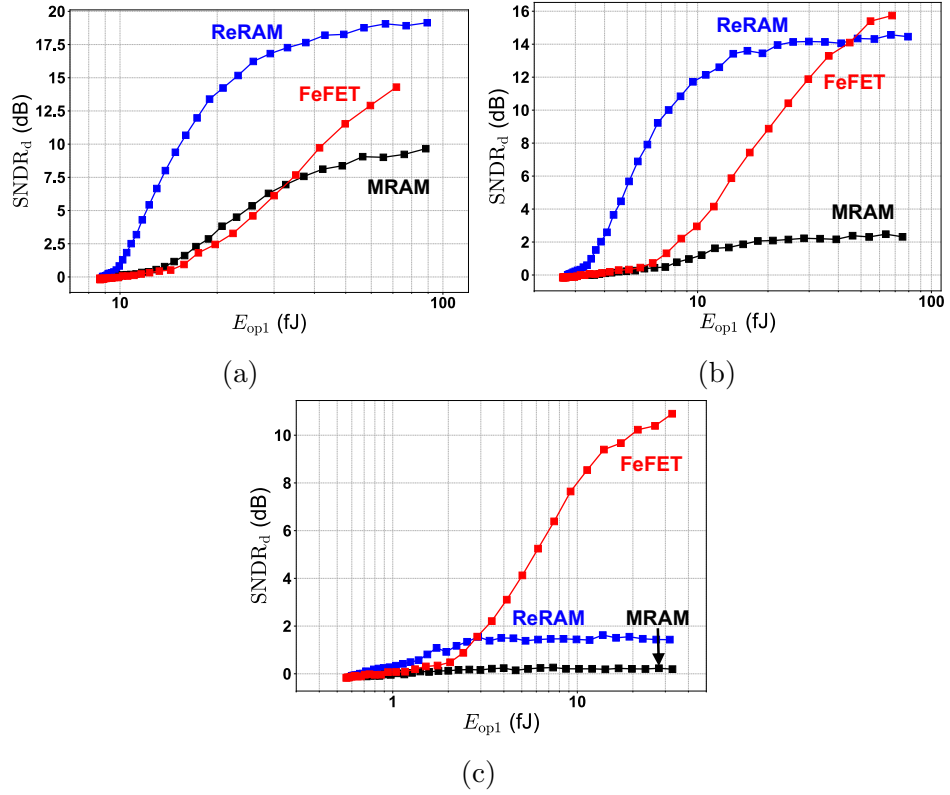


Figure 3.8: SNDR_d vs. Energy-per-1 b operation (E_{op1}) for resistive IMCs with $B_{\text{ADC}} = 6$: (a) $N = 32$, (b) $N = 128$, and (c) $N = 512$. Note: E_{op1} is obtained by sweeping $V_{\text{ref}} \in [10 \text{ mV}, V_{\text{DD}}]$.

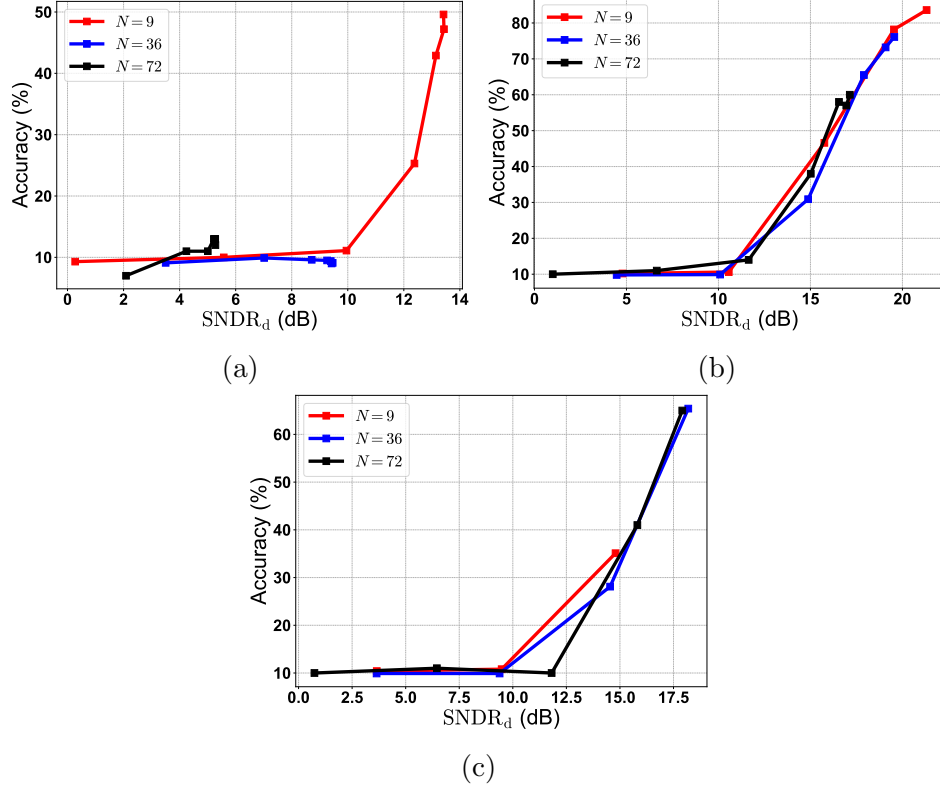


Figure 3.9: ResNet-20 (CIFAR-10) accuracy (%) vs. SNDR_d $N = 9, 36, 144$ and $B_{\text{ADC}} = 7$ using: (a) MRAM, (b) ReRAM, and (c) FeFET.

ADC thermal noise begin to dominate. These non-idealities were previously shown to be proportional to the signal current (see Fig. 3.6).

For $N = 32$ (see Fig. 3.8(a)), ReRAM-based architectures achieve the highest SNDR_d at iso- E_{op1} whereas Fig. 3.8(c) shows that FeFET-based architectures have superior accuracy at iso- E_{op1} when $N = 512$. These trends in SNDR_{d,max} can be inferred from Fig. 3.7(a) where ReRAM's SNDR_{a,max} rolls-off for $N > 5$ compared to FeFET's roll-off that begins at $N > 50$.

3.3.7 System Level Accuracy

We map the 18 convolutional layers of ResNet-20, except the first and last layers, to a non-ideal resistive parallel-bar architecture (Fig 3.2). The intermediate convolutional layers with 3×3 kernels and 16, 32, 64 input channels are structurally similar to MVMs of dot-product dimensions 144, 288, and 576, respectively. These MVMs are mapped to multiple resistive parallel-bar banks, each with ADC columns having 512 rows (following [58]) and

the same V_{ref} . However, as SNDR drops with N (Fig. 3.7(a)), we choose $N = 9, 36, 72$ to maximize SNDR_d while meeting the required dot product dimensions of ResNet-20. We trained the network with antipodal 8 b inputs and 1 b weight, to achieve a digital baseline accuracy of 89.2 % on CIFAR-10. During inference, we compute the network accuracy in the presence of all analog non-idealities as described in Section 3.2 across V_{ref} sweep for each N .

Figure 3.9 shows that the network-level accuracy increases with the bank-level SNDR_d for all three device types. The maximum accuracy drops with an increase in N for MRAM, ReRAM, and FeFET, with the peak for FeFET at $N = 36$ and $N = 9$ for MRAM and ReRAM. This is consistent with the $\text{SNDR}_{a,\text{max}}$ roll-off w.r.t N in Fig. 3.7(a). Figure 3.9(b) reveals that a minimum SNDR_d of 20 dB is required for all the layers to achieve a maximum CIFAR-10 accuracy of 84.5 %. This SNDR_d is achieved only by the ReRAM architecture with $N = 9$, at the cost of requiring more banks to realize a high-dimensional dot product. Correspondingly, numerous ReRAM IMCs have been reported with $N \leq 9$ [31, 37, 38, 43, 40, 59]. Furthermore, MRAM-based architectures achieve a maximum accuracy of only 52 % since its maximum SNDR_d is limited to 13 dB. These results indicate a strong correlation between the task-agnostic SNDR_d and the network-level accuracy and the limitations of device types.

System Level Accuracy vs. Energy

Figure 3.10 shows that an increase in E_{op1} leads to increased network-level accuracy, but the trend depends on N and device type. The system-level energy-accuracy trade-off is consistent with the bank-level energy- SNDR_d in Fig. 3.9. The increase in E_{op1} does not lead to an increase in accuracy if compute- SNDR_d saturates, as seen in Fig. 3.10(a) for the case of MRAM where accuracy remains constant beyond a limit for all N . Figures 3.10(b,c) show the network accuracy increase beyond 60% only at $E_{\text{op1}} > 100$ fJ both for ReRAM and FeFET. The maximum accuracy of 84.5 % in all cases is achieved for ReRAM for $N = 9$ with $E_{\text{op1}} = 120$ fJ. Our results indicate the requirement of boosting SNDR_d at low energy design points to boost the network-level accuracy.

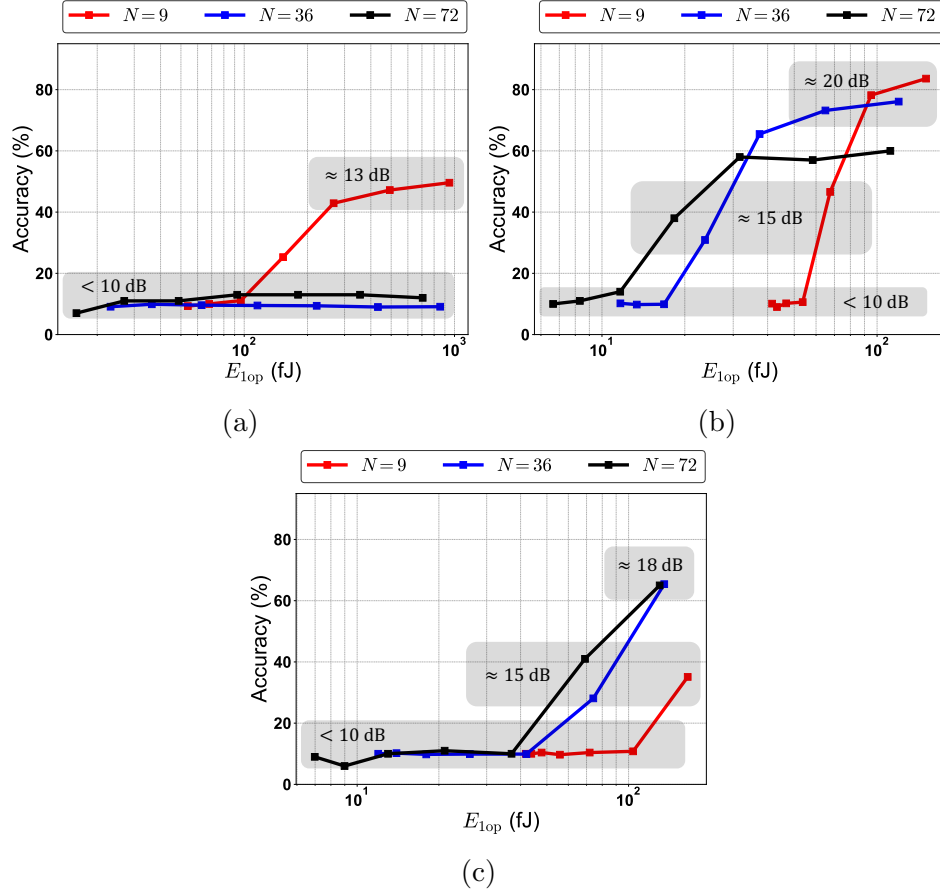


Figure 3.10: ResNet-20 (CIFAR-10) accuracy (%) vs. at E_{op1} $N = 9, 36, 72$ and $B_{ADC} = 7$ using: (a) MRAM, (b) ReRAM, and (c) FeFET.

3.3.8 Device Choice for Resistive IMCs

Here, we answer the question: What is an ideal eNVM device for resistive IMC design? In order to compute a high-dimensional dot-product accurately, a good eNVM device needs low conductance states $g_{\text{on}}, g_{\text{off}} \ll (g_{\text{bl}} || g_{\text{sl}})/2 \approx 1 \text{ S}$ and high conductance contrast $g_{\text{on}}/g_{\text{off}} > 12$ to overcome the non-linearity due to BL/SL wire parasitics, and minimize conductance variations to maximize SNDR_{d} .

3.4 Accuracy Limits of Resistive Crossbars

In this section, we develop an analytical framework for determining the fundamental limits on the bank-level computational accuracy of ReRAM, MRAM, and FeFET crossbars in terms of *compute* SNDR [157]. Although multiple approaches to improve the computational accuracy of resistive crossbars have been proposed [44, 158, 159, 45, 46, 47, 29, 48, 160, 51], these tend to be either empirical design approaches or simulation-based, and therefore unable to pinpoint the precise limits on accuracy or the key contributors to such limits.

We derive expressions for the SNDR based on circuit architecture and noise parameters, such as input digital-to-analog converter (DAC) mismatch, bit-cell conductance variations, output clipping noise, and analog-to-digital converter (ADC) quantization noise. We validate these expressions through simulations in a commercial 22 nm node and employ them to obtain limits on the maximum achievable compute SNDR as a function of the sensing resistance R_s , dot-product dimension N , ADC precision, and the resistive contrast $r_{\text{off}}/r_{\text{on}}$ (equivalent to the conductance contrast $g_{\text{on}}/g_{\text{off}}$) of the device. Finally, we map a ResNet-20 (CIFAR-10) network onto resistive crossbars and demonstrate that the circuit parameter values maximizing array-level compute SNDR also maximize network-level accuracy.

3.4.1 Resistive Crossbar Architecture

Resistive crossbar architectures employ bit-lines (BLs) perpendicular to source-lines (SLs) with one bitcell (BC) between each BL-SL pair. The conductance

of the bitcell is proportional to the weight value stored in the cell. The weight value can be binary (MRAM) or multi-bit (ReRAM, FeFET). The BC has a 1T1R structure, i.e., a transistor followed by a resistive memory device. Input vector $\mathbf{x}$ is provided via voltage DACs on the BLs while the weight vector $\mathbf{w}$ is stored in a row. Figure 3.11(a) shows the circuit model of a voltage-driven crossbar comprising a transimpedance amplifier (TIA) at the SL, which performs current-to-voltage conversion, followed by an ADC. Word-lines (WLs) connecting the gate terminals of the MOSFETs in the BCs spread across the columns. Multiple (M) WLs are activated simultaneously in order to compute M , N -dimensional dot-products, or a $M \times N$ matrix-vector multiplication. IMCs strive to have their ADC output approach the ideal N -dimensional dot-product $y_o = \mathbf{w}^T \mathbf{x}$.

The resistive contrast of a memory device is defined as $r_{\text{off}}/r_{\text{on}}$ (same as conductance contrast $g_{\text{on}}/g_{\text{off}}$) where $r_{\text{on}} = 1/g_{\text{on}}$ and $r_{\text{off}} = 1/g_{\text{off}}$ are the low (on) and the high (off) resistance values of the device, respectively. We make the following assumptions for our SNDR analysis:

- the memory device takes binary states r_{on} and r_{off} .
- the sensing resistance R_s is equal to the input impedance of the TIA.

In the case of voltage DAC driven crossbars, the j^{th} BL is driven by $V_{\text{DC}} + V_j$, where V_{DC} denotes the bias voltage for the BL. The input dependent voltage V_j is given by $x_j V_{\text{lsb}}$ with x_j representing the j^{th} element of the input vector $\mathbf{x}$, and V_{lsb} corresponding to the DAC LSB voltage. The TIA holds the SL at V_{DC} , thereby ensuring V_j voltage across the j^{th} BC in each row.

The current on the SL will not be zero for zero dot-product as the resistance values are finite. In order to remove the non-zero DC current, the following practices are commonly used [42, 54]:

- Apply complementary DAC voltage on adjacent columns. Input to adjacent columns are the same but with opposite sign i.e. $V_{2k-1} = -V_{2k} \forall k \in \{1, 2, \dots, N\}$.
- Employ two bitcells (G_{2k-1}, G_{2k}) to store a single weight bit, where b_{2k}

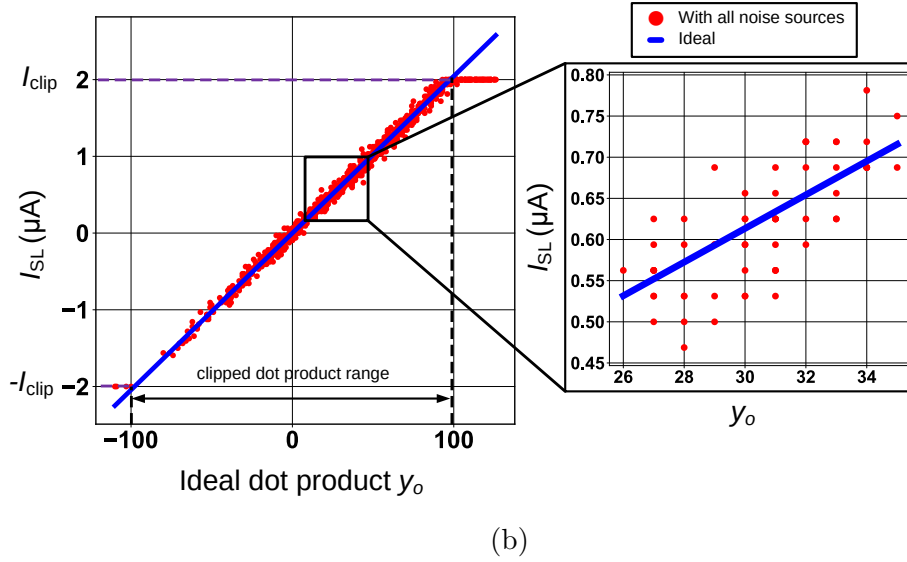
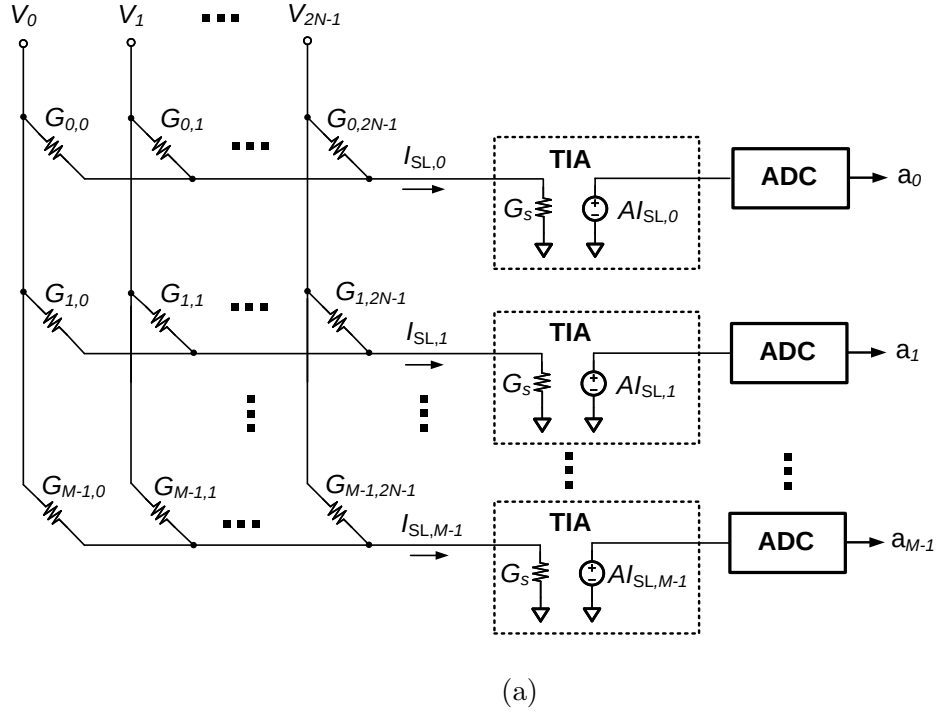


Figure 3.11: The voltage-drive resistive crossbar model: (a) circuit schematic, and (b) the impact of various noise sources on the SL current I_{SL} of a ReRAM crossbar with $N = 512$, $R_s = 316 \Omega$, and a 6 b ADC. The noise in I_{SL} leads to incorrect ADC outputs (red dots in the inset).

denotes the value stored in bitcell pair (G_{2k-1}, G_{2k}) given by:

$$b_{2k} = \begin{cases} 1, & \text{if } G_{2k-1} > G_{2k} \\ 0, & \text{if } G_{2k-1} = G_{2k} \\ -1, & \text{if } G_{2k-1} < G_{2k} \end{cases} \quad (3.19)$$

3.4.2 Compute SNDR Analysis

Signal

Applying KCL and KVL at the BLs and SLs in Figure 3.11(a), we obtain an expression for the SL current accounting for all current paths as follows:

$$I_{\text{sig}} = \left[\frac{R_{\text{arr}}}{R_{\text{arr}} + R_s} \right] \left(\sum_{k=1}^N V_{2k} \Delta G_{2k} \right) = S_I I_{\text{ideal}}, \quad (3.20)$$

where ΔG_{2k} is expressed as $G_{2k-1} - G_{2k}$ and I_{ideal} denotes the ideal SL current given by $\left(\sum_{k=1}^N V_{2k} \Delta G_{2k} \right)$ when $R_s = 0$. The Thevenin resistance looking into the array at the SL (R_{arr}) and the *current scaling factor* (S_I) are as shown below:

$$R_{\text{arr}} = \frac{1}{\sum_{j=1}^{2N} G_j}; S_I = \left[\frac{R_{\text{arr}}}{R_{\text{arr}} + R_s} \right]. \quad (3.21)$$

The parallel-bar equivalent for array resistance R_{arr} is the $1/G_{\text{eq}}$ as defined in (3.6). The expression in (3.20) can be interpreted as a fraction S_I of I_{ideal} flowing through the sensing circuit with R_s and R_{arr} appearing in parallel. This current division arising due to a finite sensing resistance R_s is distinct for crossbars as described in (3.20). In contrast, in parallel-bars, the total current flowing through the array resistance $1/G_{\text{eq}}$ is sensed directly by the current sensing circuit (3.8) leading to a higher SNDR.

On increasing the number of columns (dot product dimension) $2N(N)$, S_I decreases as $R_{\text{arr}} \propto \frac{1}{N}$. From (3.21), it is observed that reducing R_s increases S_I and hence the current. However, lowering R_s below 500Ω is challenging due to significant area overhead.

Analog Non-idealities

The SL current in presence of noise can be written as:

$$I_{\text{SL}} = I_{\text{sig}} + I_{\text{nb}} + I_{\text{nd}} + I_{\text{nc}} + I_{\text{nq}}, \quad (3.22)$$

where I_{sig} is the signal current given by (3.20). The noise sources (see Fig. 3.11(b)) are defined as:

- I_{nb} (*bitcell conductance variation*): is the noise current appearing on the SL due to variation in r_{on} and r_{off} values.
- I_{nd} (*input DAC mismatch*): is the input dependent noise current on SL due to the W/L mismatch in the DAC fingers with respect to the reference.
- I_{nc} (*clipping noise*): refers to the noise that arises due to clipping of the I_{SL} beyond the range $(-I_{\text{clip}}, I_{\text{clip}})$.
- I_{nq} (*ADC quantization noise*) $\sim U(-\frac{I_{\text{clip}}}{2^{B_{\text{ADC}}}}, \frac{I_{\text{clip}}}{2^{B_{\text{ADC}}}})$, where $U(L, H)$ denotes the uniform noise between L and H with B_{ADC} being the precision of the ADC.

We define compute SNDR at the SL as:

$$\text{SNDR} = \frac{\mathbb{E}[I_{\text{sig}}^2]}{\mathbb{E}[I_{\text{nb}}^2] + \mathbb{E}[I_{\text{nd}}^2] + \mathbb{E}[I_{\text{nc}}^2] + \mathbb{E}[I_{\text{nq}}^2]}, \quad (3.23)$$

where $\mathbb{E}$ is the expectation taken over the joint distribution of data and noise.

Note: We have ignored the impact of parasitic BL/SL resistances (PC) in this analysis. Therefore, the SNDR estimates will be optimistic.

Noise Models

The W/L mismatch in the DAC fingers lead to input dependent noise in the DAC output voltage. We combine the mismatch of the complementary $2k^{\text{th}}$ DAC pair into one random variable as:

$$\delta V_{2k} \sim \mathcal{N}\left(0, \sqrt{2|x_{2k}|} \left(\frac{\sigma}{\mu}\right)_{V_{\text{lsb}}} V_{\text{lsb}}\right), \quad (3.24)$$

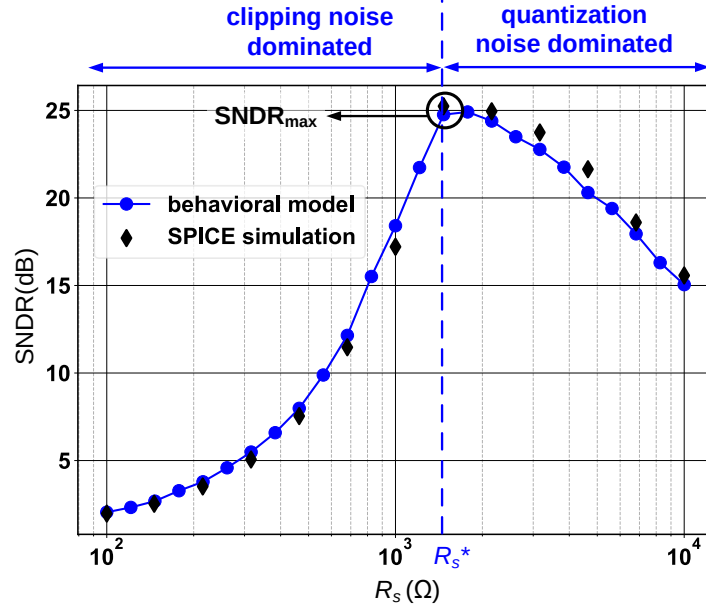


Figure 3.12: SNDR vs. R_s for ReRAM with $N = 512$ and a 6 b ADC. The SNDR achieves a maximum value $\text{SNDR}_{\max}$ at $R_s = R_s^*$ when the clipping and quantization noise are balanced.

where $\left(\frac{\sigma}{\mu}\right)_{V_{\text{isb}}}$ corresponds to the standard deviation to mean ratio of mismatch in a single DAC finger, and $x_{2k} \in \{-2^{B_x-1}, \dots, 2^{B_x-1} - 1\}$. We, then obtain I_{nd} by substituting $V_{2k} \leftarrow V_{2k} + \delta V_{2k}$ in (3.20) as:

$$I_{\text{nd}} = \left[\frac{R_{\text{arr}}}{R_{\text{arr}} + R_s} \right] \left(\sum_{k=1}^N \delta V_{2k} \Delta G_{2k} \right). \quad (3.25)$$

Conductance variations arise during the fabrication process causing g_{on} and g_{off} to exhibit spatial variations. Denoting δG_{2k} as the variation in the difference of conductances of the $2k^{\text{th}}$ bitcell pair, we get:

$$\delta G_{2k} \sim \mathcal{N} \left(0, \left(\frac{\sigma}{\mu} \right)_{\text{bc}} \sqrt{\left(\frac{1}{r_{\text{on}}} \right)^2 + \left(\frac{1}{r_{\text{off}}} \right)^2} \right), \quad (3.26)$$

where $\left(\frac{\sigma}{\mu}\right)_{\text{bc}}$ is the standard deviation to mean ratio of bitcell conductance variation. I_{nb} is determined by substituting $\Delta G_{2k} \leftarrow \Delta G_{2k} + \delta G_{2k}$ in (3.20)

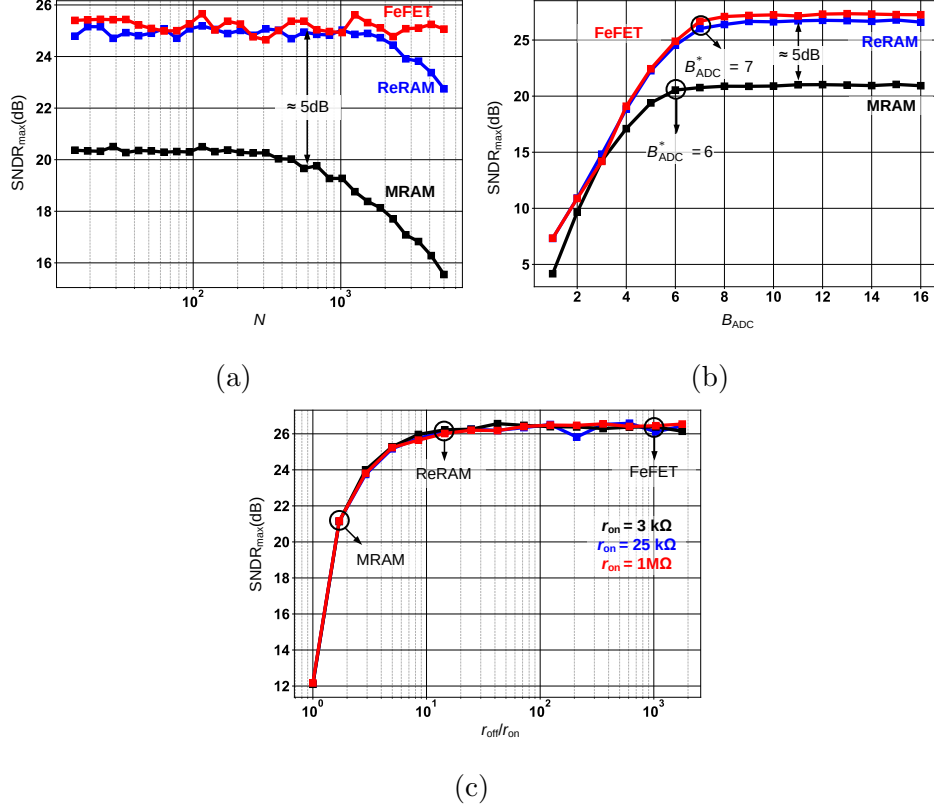


Figure 3.13: Maximum SNDR trends for ReRAM, MRAM and FeFET: (a) SNDR_{max} vs. N for $B_{\text{ADC}} = 6$ and $R_{s,\text{min}} = 1 \text{ k}\Omega$, (b) SNDR_{max} vs. B_{ADC} for $N = 512$, and (c) SNDR_{max} vs. $r_{\text{off}}/r_{\text{on}}$ for $N = 512$ and $B_{\text{ADC}} = 7$.

as:

$$I_{\text{nb}} = \left[\frac{R_{\text{arr}}}{R_{\text{arr}} + R_s} \right] \left(\sum_{k=1}^N V_{2k} \delta G_{2k} \right). \quad (3.27)$$

Note: from (3.20) conductance variations will also cause variations in R_{arr} and hence in I_{sig} independent of the input vector $\mathbf{x}$.

3.4.3 Simulation Results

Behavioral Models and Their Validation

The signal and noise models from Section 3.4.2 were verified with SPICE simulations of a 1T1R crossbar array with the MOSFETs from a commercial 22 nm process and resistances describing the bitcell state. The DAC input

uses a signed 5 b number with $V_{\text{lsb}} = 3 \text{ mV}$ and $\left(\frac{\sigma}{\mu}\right)_{V_{\text{lsb}}} = 4\%$. The $\left(\frac{\sigma}{\mu}\right)_{\text{bc}}$ for conductance variation was chosen as 4%. The clipping range of $[-2 \mu\text{A}, 2 \mu\text{A}]$ is considered for the ADC. We, then model the DAC mismatch (3.24) as noise proportional to the input which is added to the BL voltage. Conductance variation (3.26) was modeled as noise in the bitcell resistance. SPICE simulations return the SL current, which is sampled across different inputs. Next, we add clipping and quantization noise to the sampled currents in software. The empirical estimate of compute SNDR was obtained by averaging over 10000 (1000) samples in behavioral (SPICE) simulations. Fig. 3.12 shows that the two estimates match well, thereby validating our behavioral models which are employed for obtaining the subsequent results in this chapter.

SNDR Dependence on Circuit, and Device Parameters

Figure 3.12 shows that the maximum SNDR (SNDR_{max}) is achieved when the ADC clipping (I_{nc}) and quantization (I_{nq}) noise variances are equal at the SNDR-optimum value for the sensing resistance ($R_s = R_s^*$). This trade-off between clipping and quantization noise occurs because the current scaling factor S_I in (3.21) increases (decreases) as R_s decreases (increases) leading to clipping (quantization) when $R_s < R_s^*$ ($R_s > R_s^*$).

Figure 3.13(a) shows that the SNDR_{max} rolls-off as the dot-product dimension N increases beyond 500 for MRAM and 2000 for ReRAM. This roll-off occurs because the value of R_s^* reduces as N increases and at some point reaches the minimum allowable value of $R_{s,\text{min}}$ (assumed to be 1 k Ω). For higher values of N , $R_s = R_{s,\text{min}} \neq R_s^*$. In case of FeFET, the R_s^* value is large ($\approx 10 \text{ k}\Omega$) even for large N ($\approx 10^3$) and therefore, no roll-off is observed.

Figure 3.13(b) shows that the minimum ADC precision (B_{ADC}^*) required for achieving SNDR_{max} increases from 6 b for MRAM to 7 b for ReRAM and FeFET. This increase in B_{ADC}^* occurs due to an increase ($\approx 5 \text{ dB}$) in the SNDR_{max} for ReRAM and FeFET compared to MRAM, as shown in Fig. 3.13(a). Also, increasing B_{ADC} (lowering quantization noise) does not increase SNDR_{max} since DAC mismatch and conductance variations begin to dominate.

Figure 3.13(c) demonstrates that the SNDR_{max} increases with resistive contrast $r_{\text{off}}/r_{\text{on}}$ until a point (12-to-15) and then saturates. The reason

being that both R_{arr} and S_I in (3.21) increase with $r_{\text{off}}/r_{\text{on}}$. This leads to the the signal I_{sig} (3.20), DAC mismatch I_{nd} (3.25), and conductance variations I_{nb} (3.27) all increasing proportionally in (3.23). When $r_{\text{off}}/r_{\text{on}} < 12$ -15 ($r_{\text{off}}/r_{\text{on}} > 12$ -15) $\mathbb{E}(I_{\text{nc}}^2)$ and $\mathbb{E}(I_{\text{nq}}^2)$ terms in (3.23) are larger (smaller) compared to $\mathbb{E}(I_{\text{nb}}^2)$ and $\mathbb{E}(I_{\text{nd}}^2)$ resulting in the increasing (constant) SNDR_{max} . Figure 3.13(c) also indicates that SNDR_{max} only depends on the resistive contrast and not on the absolute value of r_{on} as the effect of increasing r_{on} affects both signal and noise equally.

System Level Accuracy

The compute SNDR maximizing values R_s^* predicted by our analytical framework at the array-level also maximizes network accuracy. We map a ResNet-20 comprising of three residual blocks each with 6 convolutional layers (3×3 kernels), and having 16, 32, and 64 input channels respectively. With 5 b signed inputs and ternary weights, the network achieves a 84.94% accuracy on CIFAR-10. Following [29], we map the convolutional layers implementing dot products of dimension $N = 144, 288, 576$ to three crossbar arrays consisting of 288, 576, 1152 columns with sensing resistances R_{s1} , R_{s2} , and R_{s3} respectively.

Figure 3.14 plots the empirical ResNet-20 accuracy over a sweep of (R_{s2}, R_{s3}) while setting $R_{s1} = R_{s1}^*$. The maximum accuracy $\tilde{\mathcal{A}}$ is achieved at $(\tilde{R}_{s2}, \tilde{R}_{s3})$ (indicated by blue dot), which can only be found via exhaustive search. Interestingly, the SNDR optimal points (R_{s2}^*, R_{s3}^*) (indicated by red dot) lie very close to $(\tilde{R}_{s2}, \tilde{R}_{s3})$ for all cases. Furthermore, ResNet-20 accuracy with SNDR optimal parameters $\mathcal{A}^*$ is within 1% of the empirical maximum $\tilde{\mathcal{A}}$. This shows that maximizing the array-level compute SNDR also maximizes the system-level network accuracy.

Considering the case of ReRAM in Fig. 3.14(b), the maximum CIFAR-10 accuracy of 84.52% is achieved via the exhaustive search at $(\tilde{R}_{s2} = 825 \Omega, \tilde{R}_{s3} = 383 \Omega)$ and SNDR analysis returns an accuracy of 84.11% at $(R_{s2}^* = 681 \Omega, R_{s3}^* = 562 \Omega)$. Similar observations are made for MRAM (see Fig. 3.14(a)) and FeFET (see Fig. 3.14(c)), indicating the generality of our analysis across devices. Notably, the maximum accuracy for our crossbar implementations without retraining is within 2% of the fixed-point baseline value of 84.94%.

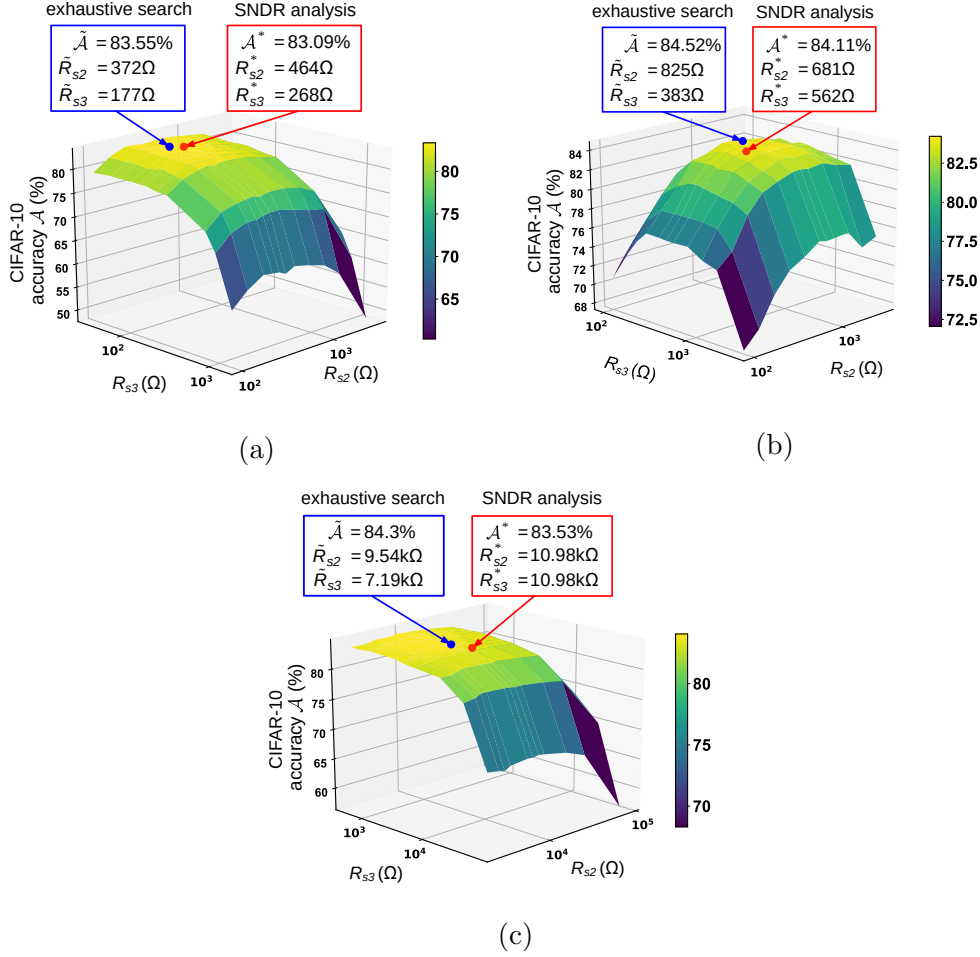


Figure 3.14: Surface plot of ResNet-20 (CIFAR-10) accuracy $\mathcal{A}$ vs. (R_{s2}, R_{s3}) with $R_{s1} = R_{s1}^*$: (a) MRAM ($R_{s1}^* = 464 \Omega$), (b) ReRAM ($R_{s1}^* = 835 \Omega$), and (c) FeFET ($R_{s1}^* = 8.28 \text{ k}\Omega$). The maximum network accuracy $\tilde{\mathcal{A}}$ and the corresponding circuit parameters $(\tilde{R}_{s2}, \tilde{R}_{s3})$ obtained via exhaustive sweep (blue dot) are both close to those obtained by employing compute SNDR analysis (red dot), i.e., $\mathcal{A}^*$ and (R_{s2}^*, R_{s3}^*) .

3.5 Discussion and Conclusion

This is the first work to analyze the non-idealities for a resistive IMC. Our work quantifies the compute SNDR dependencies and energy-SNDR trade-offs, enabling us to identify the dominant non-idealities and hint at design techniques to compensate for those. We show that the SNDR_d saturates at high signal levels since the pre-ADC analog non-idealities scale with the signal level. Furthermore, SNDR_d drops at low signal magnitudes due to a fixed ADC thermal noise floor. Overall, the maximum achievable SNDR_d of resistive IMC architectures without using any noise compensation is limited to the range 18 dB-to-22 dB, which is barely sufficient to achieve neural network accuracy of fixed-point digital accelerators.

A criticism of our analysis is the lack of an analytical model for analog non-idealities, which are mathematically intractable to formulate for resistive IMCs as the noise sources are correlated. We use a chip-verified behavioral model in our work, which includes all the major non-idealities except the power supply noise and thermal noise in current sensing, which need a design-specific analysis for resistive IMCs due to their design and layout dependence.

Nevertheless, this work provides insights into the fundamental limits of compute SNDR (both SNDR_a , SNDR_d) for resistive architectures, which explains the requirement for large read-out circuits for resistive IMCs. To achieve an energy efficiency of, say 8 fJ-per-1b operation, resistive IMC architectures are unable to achieve a compute SNDR_d higher than 10 dB (Fig. 3.8) which is much less than the 22 dB threshold required to achieve a network-level accuracy within 5% of that of an equivalent fixed-point digital architecture. This is why current-day resistive IMC designs employ various noise compensation methods at the device, circuit, and algorithmic levels to enhance accuracy at the expense of huge area and energy, thereby making them lag behind SRAM IMCs and digital accelerators (see Fig. 2.5 of **Chapter 2**).

Lastly, the crossbar analysis in this chapter enables designers to obtain SNDR optimal resistive IMC crossbar parameters without relying on expensive trial and error. Our crossbar framework also provides insights such as: 1) increasing device level resistive contrast provides diminishing returns since the input DAC and conductance mismatch begin to dominate; 2) inference accuracy is maximized when quantization and clipping noise in the ADC are

balanced. Similar to the parallel-bars, the SNDR for crossbars saturates at higher signal levels as array, sensing, and ADC noise scales proportionately. However, unlike parallel-bars with binary WL inputs, crossbars typically have multi-bit inputs along the BL, which suffer from DAC mismatch, leading to a dominant error current on SLs. Furthermore, the lack of isolation between ADC columns in crossbars and asymmetric BL and SL voltage gradient is likely to have an increased impact on wire parasitics at the output compared to parallel-bar architectures. A direct comparison between crossbar and parallel-bar architectures is a potential future research direction.

3.6 Addendum: Circuit Derivations for Non-idealities in Parallel-bar Architectures

3.6.1 BL/SL Wire Parasitics

The recursive expression in (3.11), capturing the impact of wire parasitic and BC conductance variation, is formulated starting from the bottom part of the array, i.e., $k = N$ with the N^{th} equivalent conductance given by:

$$\tilde{G}_{\text{eq},N} = \begin{cases} \tilde{G}_N \parallel (g_{\text{bl}} \parallel g_{\text{sl}}), & \text{if } x_k = -1 \\ \tilde{G}_N \parallel \frac{(g_{\text{bl}} \parallel g_{\text{sl}})}{2}, & \text{if } x_k = 1 \end{cases} \quad (3.28)$$

where $(g_{\text{bl}} \parallel g_{\text{sl}}) = g_{\text{bl}} \parallel g_{\text{sl}}$. Subsequently, the non-ideal conductance at each k^{th} stage, $\tilde{G}_{\text{eq},k}$ is computed from $\tilde{G}_{\text{eq},k+1}$. Figure 3.15(a) shows the case of $x_k = 1$ when the upper BC in the 2T2R cell is activated. The non-ideal conductance $\tilde{G}_k = G_k + \delta G_k$, following (8), is in parallel with $\tilde{G}_{\text{eq},k+1} \parallel (g_{\text{bl}} \parallel g_{\text{sl}})$ leading to,

$$\tilde{G}_{\text{eq},k} = \left[\left(\tilde{G}_{\text{eq},k+1} \parallel (g_{\text{bl}} \parallel g_{\text{sl}}) \right) + \tilde{G}_k \right] \parallel (g_{\text{bl}} \parallel g_{\text{sl}}). \quad (3.29)$$

On the other hand, Fig. 3.15(b) shows the scenario for $x_k = -1$, where the bottom BC is activated, leading to a net non-ideal equivalent conductance of,

$$\tilde{G}_{\text{eq},k} = \left(\tilde{G}_{\text{eq},k+1} + \tilde{G}_k \right) \parallel \frac{(g_{\text{bl}} \parallel g_{\text{sl}})}{2}. \quad (3.30)$$

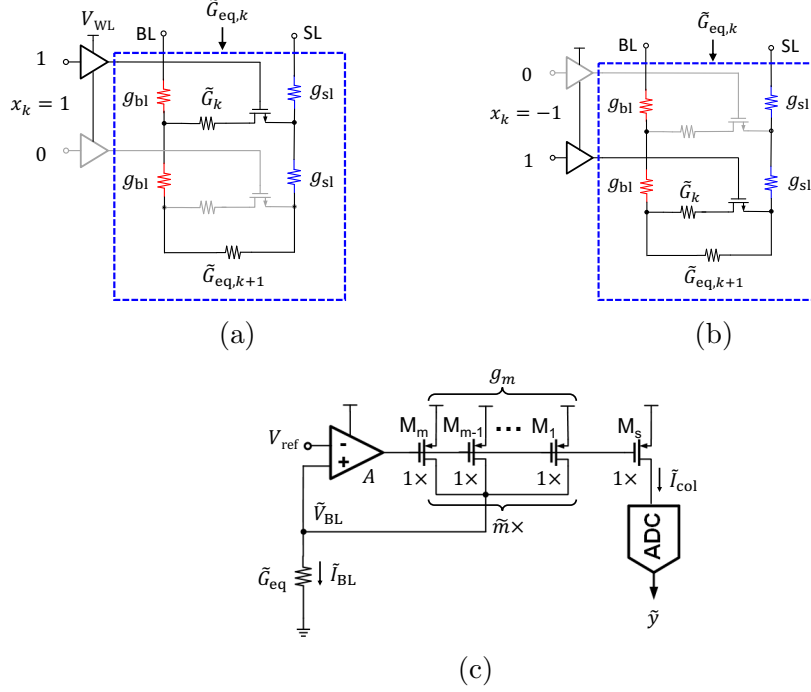


Figure 3.15: ADC column non-idealities highlighting: a) wire parasitics with $x_k = 1$ b) $x_k = -1$, and c) current mirror mismatch.

Thereafter, solving the descending recursion in (3.11), we obtain $\tilde{G}_{eq} = \tilde{G}_{eq,1}$.

3.6.2 Impact of Wire Parasitics at the Limit

The bit-line (g_{bl}) and source-line (g_{sl}) wire conductances introduce a non-linearity between the BL current $\tilde{I}_{BL}$ and ideal dot-product y_o . The non-ideal $\tilde{G}_{eq}$ (evaluated using (5)) increases sub-linearly with N as opposed to the linear increase in ideal $G_{eq} = \sum_{k=1}^N G_k$. This non-linearity leads to a low SNDR_a at large N (Fig. 3.7(a)), as the difference between G_{eq} and $\tilde{G}_{eq}$ increases. We now analyze the limiting value of the equivalent conductance $\tilde{G}_{eq}$.

We assume that for large N , the recursive expression in (5) evaluating $\tilde{G}_{eq}$, converges to a limiting value, i.e., $\tilde{G}_{eq} = \tilde{G}_{eq}^{(1)} = \tilde{G}_{eq}^{(2)}$. The maximum limiting value of $\tilde{G}_{eq}$ arises when $x_k w_k = +1 \forall k \in \{1, \dots, N\}; G_k = g_{on}$. The last

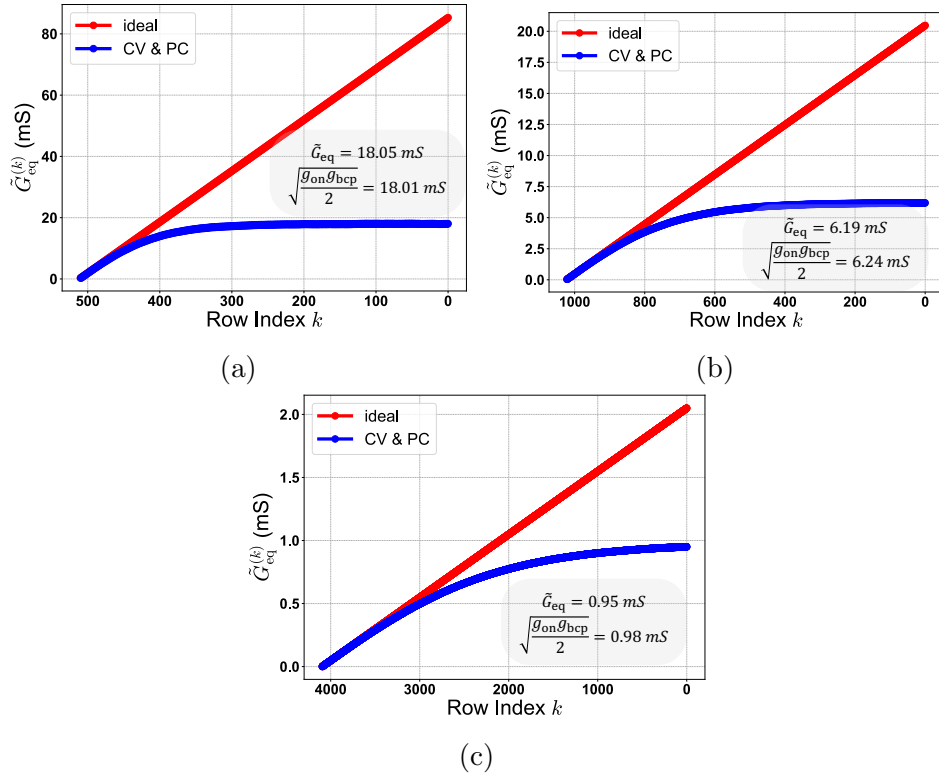


Figure 3.16: Limiting value of equivalent conductance G_{eq} for (a) MRAM, (b) ReRAM, and (c) FeFET

stage of recursion evaluates the maximum limiting value as:

$$\begin{aligned} \tilde{G}_{\text{eq}} \parallel \frac{(g_{\text{bl}} \parallel g_{\text{sl}})}{2} + g_{\text{on}} &= \tilde{G}_{\text{eq}} \\ \implies \tilde{G}_{\text{eq}}^2 - g_{\text{on}} \tilde{G}_{\text{eq}} - g_{\text{on}} \frac{(g_{\text{bl}} \parallel g_{\text{sl}})}{2} &= 0. \end{aligned} \quad (3.31)$$

Solving (3.31) we obtain $\tilde{G}_{\text{eq}} \approx \sqrt{\frac{g_{\text{on}}(g_{\text{bl}} \parallel g_{\text{sl}})}{2}}$. Similarly, the minimum limiting value of $\sqrt{\frac{g_{\text{off}}(g_{\text{bl}} \parallel g_{\text{sl}})}{2}}$ is achieved when $G_k = g_{\text{off}} \forall k$. We validate the analytical limit by comparing it with the limiting value obtained from behavioral models for MRAM, ReRAM, and FeFET, as shown in Fig. 3.16. The $\tilde{G}_{\text{eq}}$ converges to the analytical limits for all the device types for large N . However, we observe that a larger N is required to attain the limiting value as g_{on} increases from MRAM to ReRAM and FeFET. The impact of wire parasitic non-linearity on the overall SNDR_a is smaller (Fig. 6(a)) for smaller g_{on} (ReRAM, FeFET), as the difference $G_{\text{eq}} - \tilde{G}_{\text{eq}}$ is smaller as compared to the case when g_{on} is larger (MRAM).

3.6.3 Current Mirroring Mismatch

The non-ideal BL current, $\tilde{I}_{\text{BL}} = V_{\text{BL}} \tilde{G}_{\text{eq}}$ is downscaled by a factor of m before ADC conversion, as shown in Fig. 3.15(c). The W/L mismatch in the 1 : 1 fingers of the current mirror leads to a mirroring non-ideality that captures the spatial mismatch between ADC columns. It is modeled as an additive noise $\eta \sim \mathcal{N}(0, \sigma_f^2)$ on each finger i of the PMOS device in the CS loop, given by:

$$(1 + \eta_i) \tilde{I}_{\text{col}} \forall i \in \{1, 2, \dots, m\}, \quad (3.32)$$

where $\tilde{I}_{\text{col}}$ is the scaled-down current going into the ADC. The total mismatch from all the fingers appears at $\tilde{I}_{\text{col}}$ and is given by:

$$\begin{aligned} \sum_{i=1}^m (1 + \eta_i) \tilde{I}_{\text{col}} &= \left(m + \sum_{i=1}^m \eta_i \right) \tilde{I}_{\text{col}} = \tilde{I}_{\text{BL}} \\ \implies (m + \sqrt{m} \eta) \tilde{I}_{\text{col}} &= \tilde{I}_{\text{BL}} \\ \implies \tilde{I}_{\text{col}} &= \frac{\tilde{I}_{\text{BL}}}{m(1 + \eta/\sqrt{m})}. \end{aligned} \quad (3.33)$$

The summation of m random variable $\sum_{i=1}^m \eta_i$ is replaced by $\sqrt{m}\eta$ as the noise in each finger is assumed to be uncorrelated.

CHAPTER 4

A COMPUTE SNDR-BOOSTED MRAM BASED IMC PROTOTYPE IN 22 nm FD-SOI

In this chapter, we build on the insights from the behavioral modeling in **Chapter 3** to propose techniques for boosting bank-level accuracy in resistive IMCs, which have low intrinsic compute SNDR. To achieve high energy-efficiency of < 0.5 fJ per 1b operation assuming a 10 ns read latency and 1 V supply, the nominal current through an ADC column needs to be kept low, e.g., $< 10 \mu\text{A}$. Low current levels with bit-line (BL) voltage of < 1 mV can be challenging to sense in the presence of noise and offsets in the current sensing (CS) readout circuit. Furthermore, the CS circuit needs to sense small changes in the current in the presence of a number of circuit non-idealities (see Fig. 4.1), including: 1) static mismatch across ADC columns; 2) voltage gradient along the bit-line/source-line (BL/SL) arising from non-zero wire resistances; 3) thermal noise in the current sensing circuit within an ADC column; and 4) a supply voltage gradient across ADC columns due to power grid resistance.

All the above mentioned problems worsen as the DP dimension, and hence the array size, increases. As a result, the high-sensitivity readout peripherals in eNVM IMCs today occupy 80 – 90% of the total IMC macro area [26, 20, 23, 24, 39] thereby negating the areal density advantages of eNVM devices.

Consequently, several recent works on resistive IMC architectures have favored lower DP dimensions, e.g., [36, 40, 38, 37, 60, 59, 43] have $N < 10$ to preserve computational accuracy. However, an IMC supporting a small DP dimension requires either a higher frequency of weight rewrites and IMC invocations or multiple banks when mapping neural networks (NNs), which tend to have DP dimensions in the hundreds or thousands, leading to diminished system-level throughput and energy efficiency or compute density [59]. Furthermore, in order to achieve high accuracy with large DP dimensions, current IMC research [36, 37, 43, 59, 40, 60, 19, 23, 24] employ noise-aware training [19, 23, 24, 39, 148] and report network-level accuracy for a specific

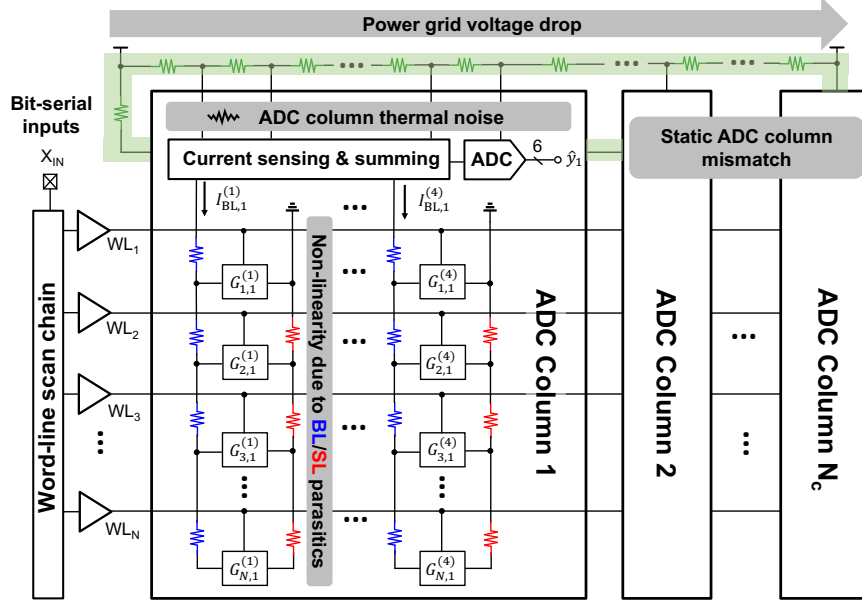


Figure 4.1: Design challenges in eNVM IMCs.

network and dataset. Though quite effective in achieving high network accuracy, sometimes close to that of a floating-point/software implementation, this approach suffers from three critical problems: 1) it is hardware-specific, i.e., it requires specific knowledge of the noise statistics of the IMC hardware; 2) it is task-specific, i.e., it needs to be trained for a specific network and dataset, raising questions about the generalization capacity of the resulting model; and 3) it obscures the fundamental trade-off between energy-efficiency and computational accuracy intrinsic to all IMCs making it hard to evaluate and to explain IMC designs.

To address these limitations, we propose a task-agnostic bank-level accuracy metric, viz. the compute signal-to-noise-plus-distortion ratio (SNDR), which precisely quantifies the bank-level accuracy of the matrix-vector multiply (MVM) computation intrinsic to an IMC bank. Employing a bank-level accuracy metric such as SNDR enables us to quantify the fundamental accuracy vs. energy trade-off inherent in IMCs.

Furthermore, we propose a compute SNDR measurement methodology to characterize the bank-level accuracy of IMCs and two compute SNDR-boosting design methods as follows:

1. an offset-compensating current sensing (OCCS) architecture to minimize the static ADC column mismatch.
2. a statistical error compensation (SEC) algorithm to compensate for the

non-linearity caused by BL/SL parasitics. Our in-situ SEC method is orthogonal to the noise-aware training approach, requiring no network-specific information.

We incorporate the two SNDR-boosting methods [58] into the design of an MRAM IMC in a 22 nm FD-SOI process and, employing the SNDR measurement methodology, we experimentally demonstrate an SEC-enabled SNDR-boost of 2.7-to-6 dB at iso-energy. This SNDR boost is then traded off with energy to achieve a $5\times$ reduction in the energy consumption at iso-SNDR. Furthermore, we show that the bank-level SNDR can be a good proxy for network-level accuracy by mapping the last layer of a ResNet-20 (CIFAR-10) onto the MRAM IMC. The SEC-enabled SNDR-boost is shown to enhance network accuracy from 74.8 % to 82.0 % without any noise-aware training of the network parameters.

4.1 Proposed Current Sensing (CS) Architecture

Current sensing (CS) techniques [36, 60, 37, 19, 23, 24, 39] strive to estimate the column conductance $G_{\text{eq}} \propto y_o$ via sensing the BL current $I_{\text{BL}} = V_{\text{BL}}G_{\text{eq}}$, where V_{BL} is a fixed BL voltage across G_{eq} . This section describes prior CS architectures employed in resistive IMCs, followed by our proposed OCCS architecture.

4.1.1 Existing CS Architectures

Self-biased PMOS

Fig. 4.2(a) shows a low-area solution for current sensing, i.e., a diode-connected PMOS [36, 37, 38, 60]. Assuming square-law behavior and ignoring channel length modulation, the current I_{BL} is given by:

$$I_{\text{BL}} = \frac{k_p}{2} \left(\frac{W}{L} \right) \left(V_{\text{DD}} - V_t - \frac{I_{\text{BL}}}{G_{\text{eq}}} \right)^2 \quad (4.1)$$

where k_p is a process parameter and V_t is the threshold voltage. Equation (4.1) indicates that I_{BL} , and hence V_{BL} , is susceptible to process, voltage,

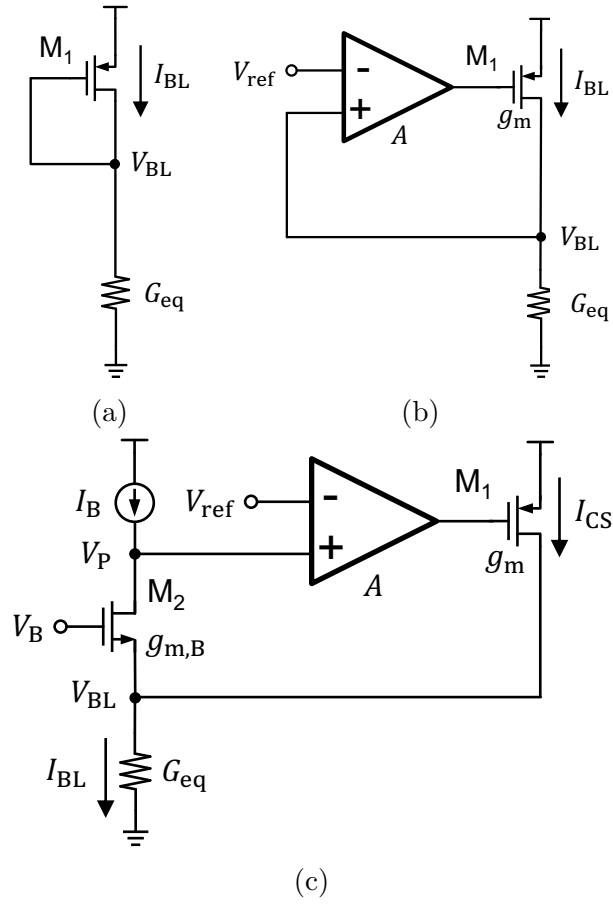


Figure 4.2: Current sensing architectures for resistive IMCs using: (a) a self-biased PMOS, (b) a negative feedback-based, and (c) a common gate-based feedback-based sensing.

and temperature (PVT) variations as it depends on both k_p and V_t . This leads to a static mismatch across ADC columns. Furthermore, the relationship between I_{BL} and G_{eq} is intrinsically non-linear, thereby introducing a non-linearity in the DP output.

The BL current I_{BL} can be reduced to enhance energy efficiency by reducing V_{DD} at the expense of increased impact of PVT variations or by reducing the DP dimension N , which reduces G_{eq} . Therefore, resistive IMCs employing diode-connected sensing [36, 37, 38, 60] rely on small dot-product dimensions ($N < 10$) to achieve high energy efficiency while preserving their accuracy.

Negative Feedback-based Sensing

Negative feedback-based sensing (see Fig. 4.2(b)) removes the dependence of the I_{BL} on device parameters k_p and V_t . The BL current is given by $I_{BL} = V_{BL}G_{eq}$ where V_{BL} is set by an external reference voltage V_{ref} shared across the ADC columns, thereby reducing the static mismatch. However, a large loop gain $LG = Ag_m/G_{eq}$ is required to ensure $V_{BL} = V_{ref} \times LG/(1 + LG) \approx V_{ref}$, where A is the differential amplifier gain, and g_m is the transconductance of PMOS M_1 , e.g., to keep the V_{BL} within 1% of V_{ref} , a loop gain of 100 is needed. Assuming $I_{BL} = 20 \mu A$, $G_{eq} = 20 mS$ and an overdrive voltage for M_1 $V_{ov} = 200 mV$, a differential amplifier gain of $A = 80 dB$ is required.

The energy-efficiency can be enhanced by lowering V_{ref} , which reduces V_{BL} and hence I_{BL} . However, $V_{ref} < 10 mV$ makes the amplifier offset V_{os} (typically in the range 5 – 10 mV) and thermal noise affects the stability of V_{BL} and thereby introduces static mismatch across ADC columns.

Common Gate-based Feedback-based Sensing

Common gate (CG)-based feedback-based sensing (CGFBS) (Fig. 4.2(c)) [24] with the current source I_B biasing M_2 in saturation, leads to the sensed current I_{CS} given by:

$$I_{CS} = V_{BL}G_{eq} - I_B. \quad (4.2)$$

Introducing the CG amplifier into the feedback loop yields several advantages. First, it diminishes the sensitivity of the sensed current to amplifier

offset voltage by attenuating variations in V_P through M_2 before they manifest at V_{BL} . Second, as V_{BL} is isolated from V_{ref} , it facilitates a low V_{BL} , thereby enhancing energy efficiency. Third, it alleviates the gain requirements on the differential amplifier, as the CG stage contributes additional gain. To achieve a loop gain of $LG = 80\text{ dB}$, the differential amplifier gain can be reduced to $A = 54\text{ dB}$ since the CG stage augments the loop gain by a factor of $g_{m,B}r_o \approx 20$, where $g_{m,B}$ denotes the transconductance and r_o represents the output resistance of the CG transistor M_2 .

However, this method, similar to the self-biased PMOS method in Fig. 4.2(a), suffers from the impact of PVT variations and thermal noise since $V_{BL} = V_B - V_t - V_{ov}$ is directly dependent on the threshold voltage V_t . Offset compensation employed in [24] includes a capacitive biasing in the CG stage to reduce deviations in V_{BL} and a negative feedback-based capacitive storage to compensate the intrinsic offset of the differential amplifier. But, at lower values of V_{BL} , the residual offset after compensation is dominant due to the increased impact of non-idealities.

4.1.2 Proposed OCCS Architecture

Our proposed offset-compensating current sensing (OCCS) architecture depicted in Fig. 4.3(a) addresses the challenges from prior designs by substituting transistor M_2 in Fig. 4.2(c), with a level-shifting resistor R_B . This modification ensures that $V_{BL} = V_P - I_B R_B$ and the sensed current I_{CS} (equivalent to (4.2)) remains independent of transistor parameters, rendering them robust to PVT variations while enabling a low V_{BL} for power savings. However, the amplifier offset voltage V_{os}^a , mismatch in level-shifting resistor ΔR_B , and mismatch in bias current ΔI_B still influences $V_{BL} = V_{ref} + V_{os}^a - (I_B + \Delta I_B)(R_B + \Delta R_B)$, detrimentally impacting DP accuracy. To remedy this, we propose compensating $V_{os} = V_{os}^a - I_B \Delta R_B$ while using a large device for I_B current source, keeping ΔI_B low, thereby achieving V_{BL} close to $V_{ref} - I_B R_B$ across CS loops. To do so, we compensate the total offset at V_P , keeping $V_{ref} + V_{os}$ close to V_{ref} , thereby permitting positive and negative offset cancellation. Next, we describe our offset-compensation scheme, which reduces the overall static ADC column mismatch.

Offset-compensating CS Loop

Figure 4.3(b) shows the schematic of our proposed feedback-based sensing equipped with offset compensation. We reduce the overall static ADC column mismatch by employing a two-step compensation technique.

Current sensing (Fig. 4.3(b)) is carried out in two phases: 1) auto-zero (AZ) phase ($S_1 = 1$), and 2) evaluation (E) phase ($S_1 = 0$).

During the AZ phase, the outer feedback loop is open, rendering $I_{CS} = 0$. The matched pairs (M_3, M_4) and (M_5, M_6) ensure the current I_{os} through M_3 and M_6 are equal, resulting in no additional current flowing into R_B . The bias current I_B set by tuning the external resistance R_{ext} is mirrored into CS loop via the matched pair (M_8, M_9) resulting in

$$V_P = I_B (R_B + 1/G_{eq}^{(az)}), \quad (4.3)$$

independent of the amplifier offset voltage. The pair (M_8, M_9) is sized to be large, keeping the bias current mismatch ΔI_B low. Similarly, the BL voltage V_{BL} is set to $I_B/G_{eq}^{(az)}$, where $G_{eq}^{(az)} = Ng_{ap}$ is the minimum conductance the CS loop is required to sense. This is achieved by providing a fixed WL pattern during the AZ phase.

In the presence of an amplifier offset voltage (V_{os}^a) and mismatch in level-shifting resistor ΔR_B , the AZ loop increases/decreases I_{os} to $I_{os} \pm \Delta I$, ensuring $\Delta I R_C$ equals $V_{os} = V_{os}^a - I_B \Delta R_B$. The ΔI_{os} set by AZ loop adapts to R_C mismatch, thus compensating net CS loop offset voltage.

To realize this function, in the zero offset scenario ($V_{os} = 0$), the internal feedback loop sets the gate voltage of M_3 and M_4 to establish an I_{os} satisfying the zero offset condition $V_P = V_{ref} - I_{os} R_C = (I_B - I_{os}) R_C \approx I_B R_B$, assuming $R_B > 1/G_{eq}^{(az)}$ in (4.3), which is typical since column conductances are in the order of 20 mS.

To satisfy this zero offset condition while being able to cancel $V_{os} \in [\pm 100 \text{ mV}]$, we choose $R_B = 0.8 R_C$ with identical resistor types and $I_{os} = 0.2 I_B$ by sizing the matched pair (M_3, M_4) $5\times$ smaller with respect to (M_7, M_9) , thereby setting $V_P (\approx 0.8 V_{ref})$ to be approximately 100 mV below V_{ref} .

In the E phase, the outer loop is closed, and the gate voltages of M_3 and M_4 are maintained at the AZ voltage held across the C_{os} capacitor. Column conductance $G_{eq}^{(e)}$ increases from $G_{eq}^{(az)}$ to a higher value in the $[Ng_{ap}, Ng_p]$ range,

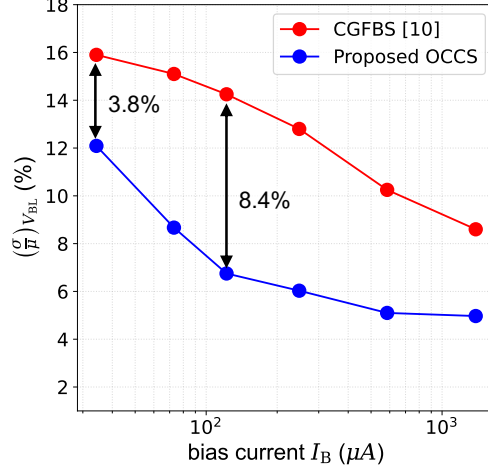


Figure 4.4: Simulated variations in V_{BL} vs. bias current I_B for OCCS and CGFBS [24].

increasing the sensed current I_{CS} through M_1 , given by $I_{CS} = V_{BL}G_{eq}^{(e)} - I_B$.

We employ a single-stage telescopic cascode amplifier to ensure the stability of the feedback loops in both the AZ and E phases. The transistor sizing is chosen to keep the thermal noise low while satisfying the pitch-matching constraints of the ADC column. Monte Carlo simulation results in Fig. 4.4 show that the proposed OCCS scheme consistently achieves lower V_{BL} variation (σ/μ) ranging from 3.8 %-to-8.4 % while incurring an area overhead of $\sim 17\%$ when compared to CGFBS [24], at the same value of I_B and V_{BL} at a fixed $G_{eq} = 10$ mS. Next, we propose an algorithmic approach to compensate for the non-linearity arising from BL/SL wire parasitic.

4.2 Statistical Error Compensation for BL/SL Resistances

The BL/SL wire resistances arising from the routing in the memory array create voltage gradients in the BL and SL voltages (see Fig. 4.5(a)), leading to non-linearity in the sensed current I_{CS} (see Fig. 4.5(b)). This is due to cell currents being dependent on the location of the activated bit-cells (BCs). To address this issue, we propose an algorithmic approach, viz. statistical error compensation, to compensate for this non-linearity. We employ the following

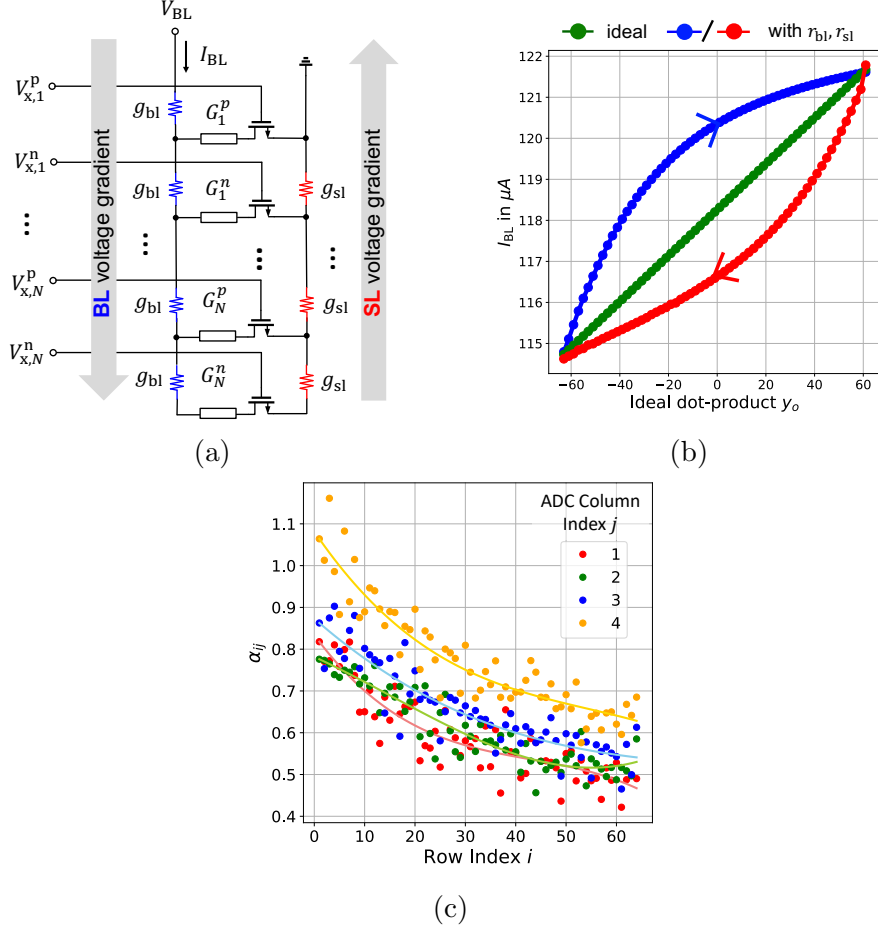


Figure 4.5: Non-linearity due to BL/SL wire parasitics for an ADC column: (a) circuit schematic, (b) simulated BL current I_{BL} vs. ideal dot-product y_o for $N = 64$ (ideal curve is offset and gain compensated), and (c) correlating the α signal model with measured data from four ADC columns.

update rule [161] to minimize the mean squares error (MSE) $\|\mathbf{y}_o - \hat{\mathbf{y}}\|^2$:

$$\gamma_i[t+1] = \gamma_i[t] + \mu \sum_{j=1}^{N_C} e_j[t] \text{sgn}(x_i[t] w_{i,j}), \quad (4.4)$$

where $\mathbf{e} = [e_1, e_2, \dots, e_{N_C}]^T = \mathbf{y}_o - \hat{\mathbf{y}}$ is the error vector for N_C ADC columns, and μ is the update step size.

4.2.1 Non-linearity due to Wire Parasitics

Figure 4.5(b) shows the simulated BL current I_{BL} vs. ideal dot-product y_o (also see [23]). We choose the weight vector $\mathbf{w} = \{1\}^N$ and sweep the activations in order to compute all dot-product values $y_o \in [-N, N]$. The weights are coded into the array using differential BCs as $(G^p, G^n) = (g_p, g_{ap})$, where g_p and g_{ap} are the high and low conductance states of an MTJ, respectively, and (G^p, G^n) , represent the conductance state of the differential BC. In the absence of BL/SL wire parasitics (green line in Fig. 4.5(b)), $I_{BL} = V_{BL}G_{eq}$ is a linear function of the equivalent column conductance $G_{eq} \in [Ng_{ap}, Ng_p]$. However, in the presence of r_{bl}, r_{sl} , the linear relationship no longer holds, as seen in Fig. 4.5(b). The key reason for this phenomenon is that the BCs in the lower part of the array experience a smaller voltage drop than those in the upper part of the array and, hence, contribute a smaller current when activated.

For example, when the input activation is reduced from all ‘1’s towards the all ‘-1’ pattern, starting with the upper BCs switching first from a high conductance state g_p to a low conductance state g_{ap} . As a result, I_{BL} experiences a sharper drop than when BCs in the lower part of the array begin to switch (red plot in Fig. 4.5(b)). The reverse is true (blue plot line in Fig. 4.5(b)) when the input activation is applied from all ‘-1’s to an all ‘1’s pattern again, starting with the upper BCs making this switch first. In both cases, the maximum deviation from the ideal current (green plot in Fig. 4.5(b)) occurs at $y_o = 0$.

4.3 The SEC-enabled MRAM IMC Architecture

The SEC-enabled MRAM IMC architecture in Fig. 4.6 comprises an IMC core and a digital SEC processor. These are described next.

4.3.1 MRAM IMC Core Architecture

The MRAM IMC architecture (Fig. 4.6) is a 512×512 memory array with 128 ADC columns. Each ADC column comprises $B_w = 4$ physical columns each with its own current sensing OCCS loop, differential BCs with comple-

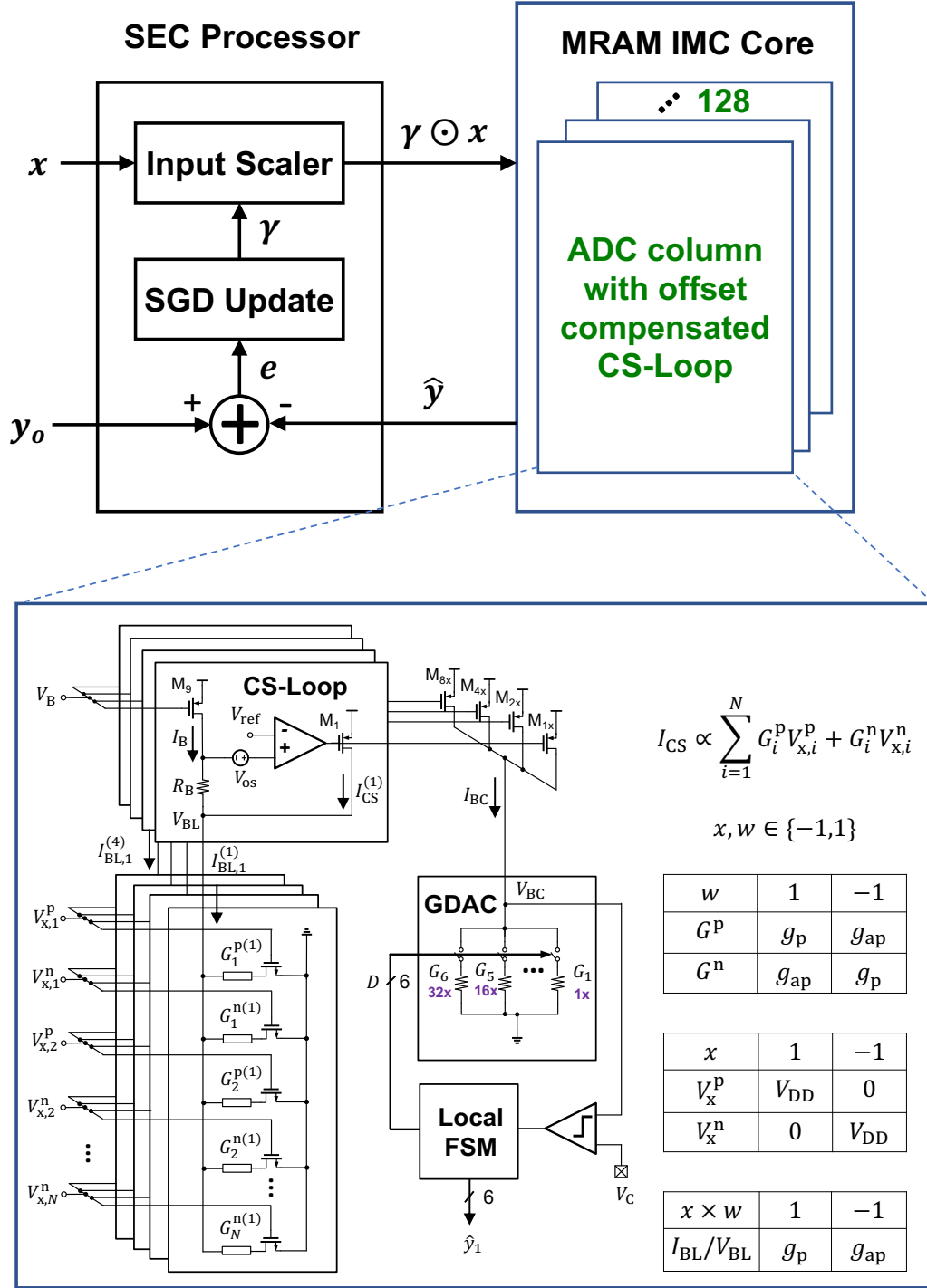


Figure 4.6: Proposed MRAM IMC architecture incorporating the SEC and OCCS design methods.

mentary conductance states $(G^p, G^n) \in \{(g_p, g_{ap}), (g_{ap}, g_p)\}$, and differential input wordlines $(V^p, V^n) \in \{(V_{DD}, 0), (0, V_{DD})\}$ to implement a signed bipolar multiplication.

Each physical column strives to generate an output current I_{CS} proportional to the ideal N -dimensional binarized DP y_o as follows:

$$I_{CS} = V_{BL}G_{eq} - I_B \propto y_o = \sum_{i=1}^N w_i x_i \quad (4.5)$$

where I_B is a bias current and $w_i, x_i \in \{-1, 1\}$, with

$$\begin{aligned} G_{eq} &= \sum_{i=1}^N G_i^p V_i^p + G_i^n V_i^n \\ G_i^p &= g_p \left(\frac{1 + w_i}{2} \right) + g_{ap} \left(\frac{1 - w_i}{2} \right) \\ G_i^n &= g_{ap} \left(\frac{1 + w_i}{2} \right) + g_p \left(\frac{1 - w_i}{2} \right) \end{aligned} \quad (4.6)$$

$$V_i^p = V_{DD} \left(\frac{1 + x_i}{2} \right); V_i^n = V_{DD} \left(\frac{1 - x_i}{2} \right) \quad (4.7)$$

The use of differential BCs generates a linearly stepped conductance range $[Ng_{ap}, Ng_p]$ in proportion to y_o .

The output currents from four adjacent physical columns are mirrored and combined in a binary-weighted manner during the evaluation phase of the OCCS loop to generate the final output current I_{BC} of one ADC column:

$$I_{BC} = \sum_{k=1}^4 I_{CS}^{(k)} 2^{k-1}, \quad (4.8)$$

which is then digitized using a 6 b SAR ADC. In this process, each ADC column computes a $1 \text{ b} \times 4 \text{ b}$ N -dimensional DP. Multi-bit inputs can be processed in a bit-serial format.

The SAR ADC comprising of a conductance DAC (GDAC), a 1 b comparator, and SAR logic determines a 6 b code D such that $I_{BC} = V_{BC}G_{DAC}(D)$. The control signals for the ADC are generated through a finite state machine (FSM) per ADC column.

The fixed-point SEC algorithm is realized as a digital processor (see Fig. 4.7) that interacts with the MRAM IMC core. To enable fast convergence, a user

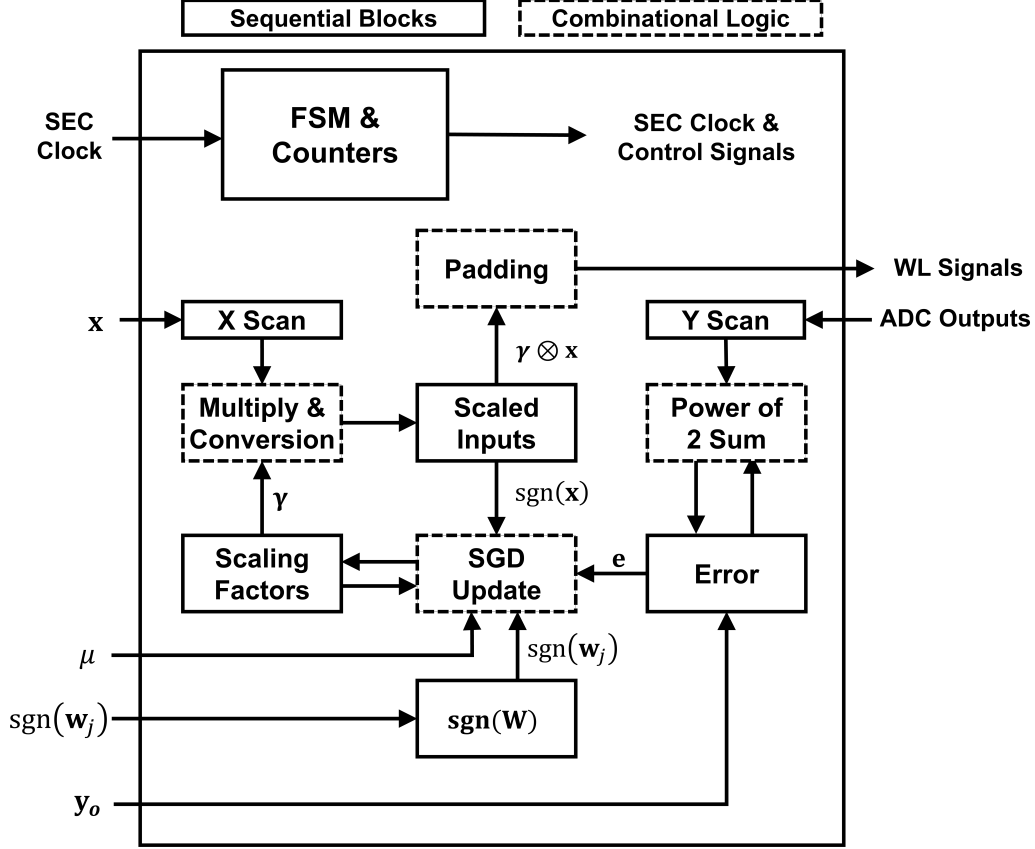


Figure 4.7: The SEC processor architecture.

switch selects between two values of the update step size μ with a ratio of 4 to be used during different stages of training. The SEC processor can be configured for training with data from selected ADC columns to prevent low accuracy columns from impairing the convergence dynamics.

4.3.2 Power-grid Routing

Power supply noise arises in the ADC column array due to the presence of wire parasitics in the power grid routing, leading to a gradient in the supply voltage (see Fig. 4.1). This gradient across ADC columns becomes significant when multiple ADC columns are activated simultaneously, consuming large currents and leading to a data-dependent ADC column mismatch. We perform a careful power routing with multiple vias spread across the power grid to reduce the wire resistance, thereby reducing the V_{DD} gradient for all CS loops.

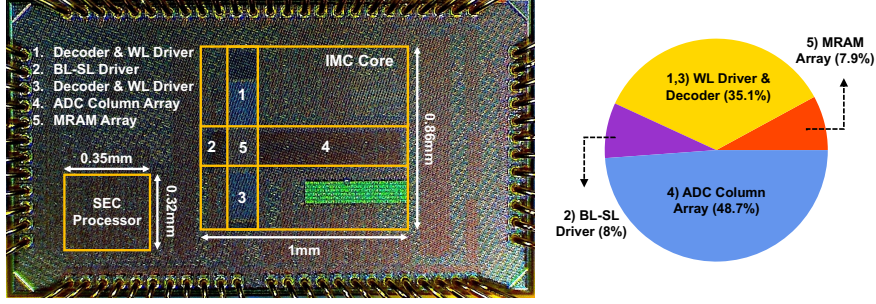


Figure 4.8: Chip photograph and IMC core area breakdown.

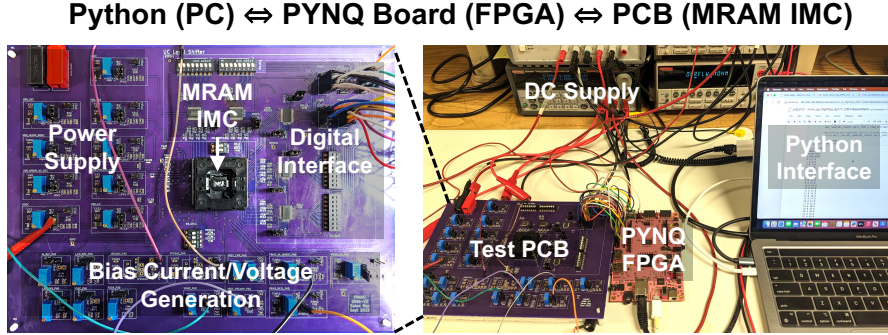


Figure 4.9: Measurement setup comprising a Python interface on host PC, a PYNQ FPGA, and a test PCB housing the MRAM IMC.

4.4 The Compute SNDR Methodology

In our work, we utilize SNDR to characterize a complete ADC column, recognizing it as the fundamental building block of an IMC, with its accuracy directly impacting overall IMC performance. Unlike traditional ADC characterization with full-scale sinusoidal inputs, we determine the SNDR characterization inputs for an ADC column within an IMC by identifying those that activate the maximum number of ADC column states. In order to compute the bank level SNDR, we employ post-calibrated ADC outputs where the calibration parameters are computed off-chip using a test dataset. This section describes our calibration scheme followed by the compute SNDR methodology.

Table 4.1: Chip performance summary.

Performance Summary			
Technology node		22 nm	
Supply voltage (V)		0.8	
Number of ADC columns		128	
IMC array size		512×512	
Frequency (MHz)		8.3, 16.6 (Core), 25 (SEC)	
SEC area overhead		12.2%	
Dot-product dimension (N)		64	128
SEC energy overhead $B_x, B_w, B_{\text{ADC}} = (7, 4, 6)$		0.8%	1.8%
SNDR _{col} (dB)	Before SEC	5.15	0.77
	After SEC	11.1	3.5
Accuracy @ 90% C_L	Before SEC	$74.8 \pm 5\%$	
	After SEC	$82.0 \pm 5\%$	

4.4.1 SNDR Evaluation

The raw ADC output $\hat{y}$ (Fig. 4.6) is calibrated off-chip on a per-column basis to:

$$\tilde{y} = \theta(a\hat{y} + b) \quad (4.9)$$

where $\theta = 1$ (SEC OFF) and $\theta = 1/\alpha_N$ (SEC ON). The parameters a and b are chosen to minimize the mean squared error (MSE) with respect to the ideal dot-product y_o with $\theta = 1$. To find the minimum MSE (MMSE) values of a and b , we collect the raw ADC outputs $\hat{y}$ using a test dataset and compute:

$$a = \frac{\text{Cov}(y_o, \hat{y})}{\text{Var}(\hat{y})}; b = \mathbb{E}[y_o] - a\mathbb{E}[\hat{y}]. \quad (4.10)$$

Henceforth, we use the calibrated ADC outputs $\tilde{y}$ for SNDR analysis.

An ADC column in the MRAM IMC has multiple states determined by the weights $\mathbf{w}$ stored and the inputs $\mathbf{x}$ applied. The SNDR characterization of an ADC column needs to activate all its states of operation in order to accurately capture its computational accuracy.

In the following, for simplicity, we assume each ADC column comprises a single physical column and computes an N -dimensional binary DP $y_o = \mathbf{w}^T \mathbf{x}$. Each weight and activation pair corresponds to a unique circuit configuration and, hence, a unique state $(\mathbf{w}, \mathbf{x}) \in \mathcal{S}$, where $\mathcal{S}$ is the set of all possible

states obtained by choosing $\mathbf{w}, \mathbf{x} \in [\pm 1]^N$. Clearly, the total number of states $|\mathcal{S}| = 2^{2N}$ is a prohibitively large number that makes it challenging to activate all of them for SNDR characterization.

To reduce the complexity of SNDR characterization, we consider subsets $\mathcal{S}^c = \{(\mathbf{x}, \mathbf{w}) : \mathbf{w}^T \mathbf{x} = c\} \subset \mathcal{S}$, i.e., we lump all states that lead to the same output $y_o = c$. Note that $\bigcup_{c=1}^N \mathcal{S}^c = \mathcal{S}$. Next, assuming all states $(\mathbf{x}_j, \mathbf{w}_j) \in \mathcal{S}^c$ are equally likely, we compute the per code mean squared error (MSE) for $y_o = c$ as:

$$\text{MSE}_{\text{code}}(c) = \frac{1}{|\mathcal{S}^c|K} \sum_{k=1}^K (\tilde{y}_k - c)^2, \quad (4.11)$$

where we allow for K repeated activations per state in order to average over noise samples.

The column SNDR is then calculated as:

$$\text{SNDR}_{\text{col}} = \frac{\text{Var}(y_o)}{\sum_{c \in \mathcal{Y}} \text{MSE}_{\text{code}}(c) P_Y(c)}. \quad (4.12)$$

where P_Y is the probability distribution of the ideal output y_o which is reasonably assumed to be the binomial distribution $\text{Bi}(N, 0.5)$. However, covering all the states in the subset $\mathcal{S}^c$ remains challenging for practical IMCs, e.g., for $c = 32, N = 64$, $|\mathcal{S}^c| \approx 10^{18}$. Therefore, we fix the weight vector $\mathbf{w} = \{1\}^N$, and randomly select 30 states from $\mathcal{S}^c$ for each code c . We choose $K = 10$, amounting to 300 measurements per code per ADC column.

4.5 Measured Results

This section describes the measured results of the prototype IC, including the ADC column SNDR characterization and the SEC-enabled SNDR boost at the bank-level and network-level. The MRAM IMC chip, fabricated in a commercial 22 nm FD-SOI process, is packaged with a 64-pin QFN. Figure 4.8 shows the die photograph with the area breakdown of the IMC core, and Table 4.1 shows the chip performance summary. The ADC column array, occupying 48.7% of the total area, dominates the core area compared to the 7.9% share of 32 kB 1T1R MRAM array. The measurement setup is shown in Fig. 4.9 with the Python interface on PC communicating with Xilinx PYNQ-

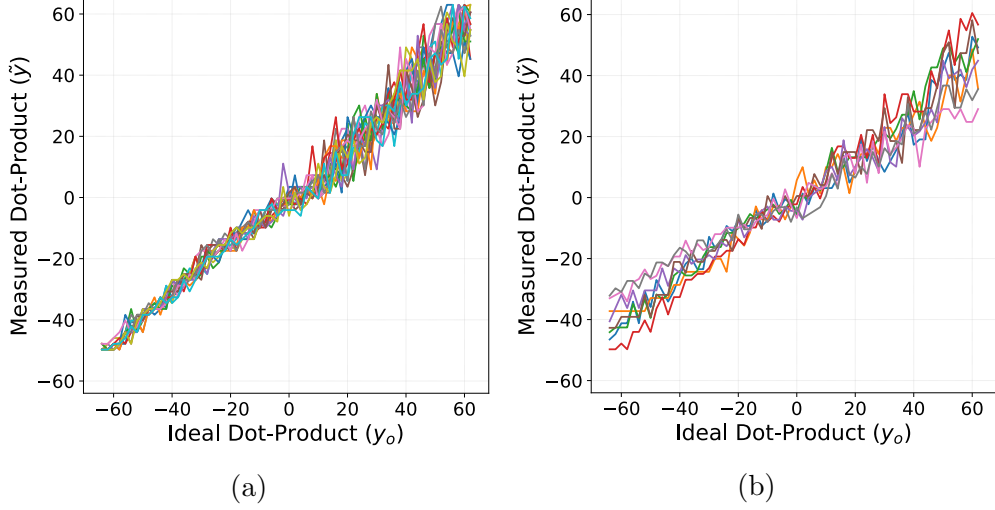


Figure 4.10: Measured results obtained by activating a single state per code: (a) impact of noise in a single ADC column using ten repeated inputs, and (b) static mismatch across eight ADC columns.

Z2 development board equipped with an FPGA that communicates to the MRAM IMC mounted on a test PCB.

4.5.1 ADC Column SNDR Characterization

Figure 4.10(a) shows that the noise in a single ADC column increases with y_o . The dominant noise in each CS loop (see Fig. 4.3(b)) arises from the PMOS bias transistor M_9 and matched pair (M_5, M_6) used for offset compensation. The total noise i_n injected by the dominant sources at V_P node appears as a noise i_{nCS} in sensed current I_{CS} , expressed as:

$$i_{nCS} = i_n(1 + R_B G_{eq}), \quad (4.13)$$

which increases with $G_{eq} \propto y_o$, thereby increasing noise in the measured dot-product $\tilde{y}$. On the other hand, the static ADC column mismatch (see Fig. 4.3(b)) exhibits higher variations at the two extremes of y_o . One reason being that the post-ADC MMSE calibration step (see (4.10)) is w.r.t. $y_o = 0$.

Figure 4.11(a) shows the histogram of 300 measurements for three ideal dot-products $y_o = \pm(N-2), 0$ on a single ADC column. These values are chosen to ensure that multiple states are activated within each subset $\mathcal{S}^c$. The standard deviation (σ) of the measured output $\tilde{y}$ increases with y_o for both

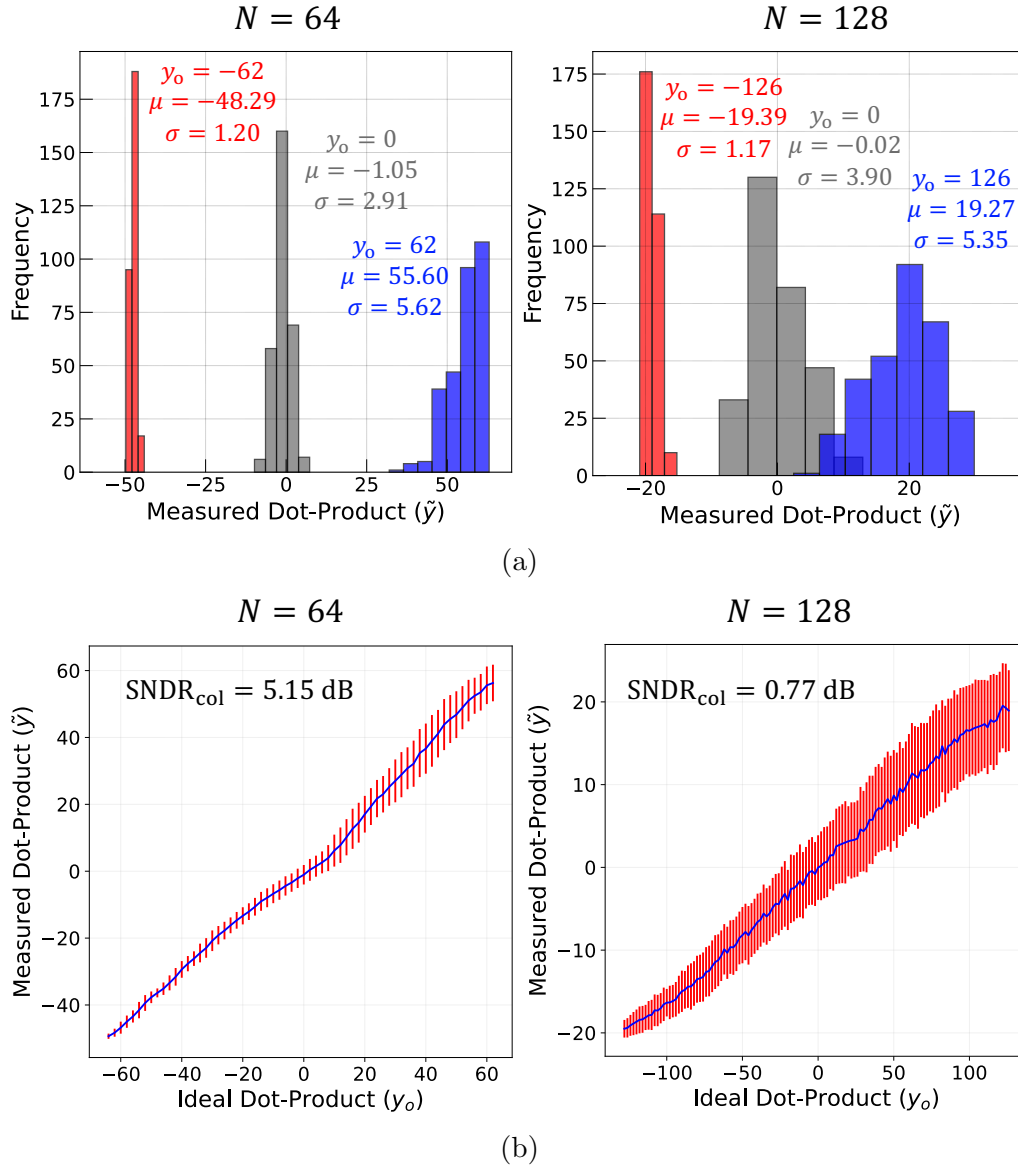


Figure 4.11: Measured ADC column output $\tilde{y}$ with $N = 64, 128$: (a) histograms for $y_o = \pm(N - 2), 0$, and (b) the $1\text{-}\sigma$ error bars.

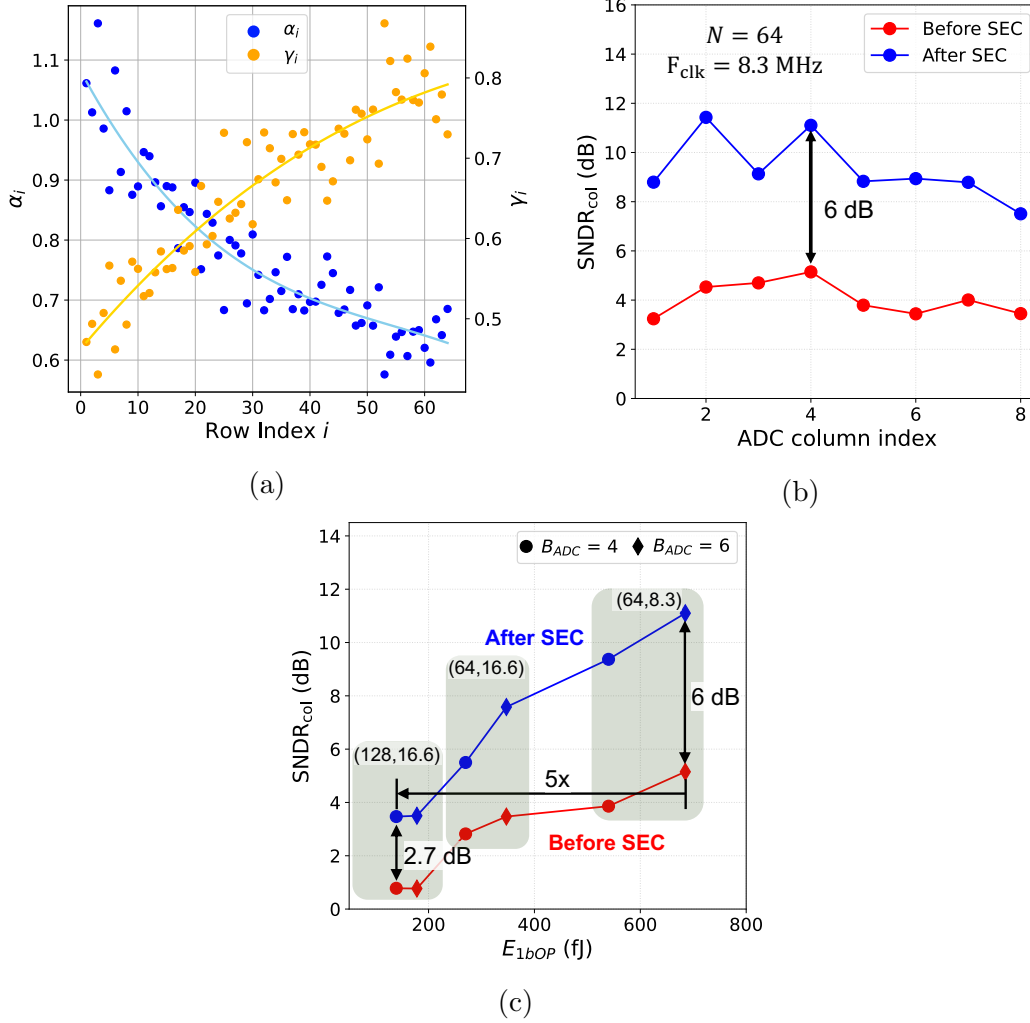


Figure 4.12: Measured results in SEC-enabled mode: (a) learned input scaling parameters γ_i , (b) SNDR boost across ADC columns, and (c) SNDR-energy trade-off for a single ADC column achieved by changing (N, F_{clk}) configurations.

$N = 64, 128$ for the same reason as it does in Fig. 4.10. Figure 4.11(b) shows error bars for measured dot-products $\tilde{y}$ with MMSE optimal gain and offset parameters (a,b) of (0.95, 11.09) for $N = 64$ and (0.2, 4.89) for $N = 128$. The SNDR_{col} drop from 5.15 dB to 0.77 dB as the inner dimension N increases from 64 to 128. This occurs due to the increased net BL/SL wire resistance and increased noise at higher G_{eq} .

4.5.2 SEC-enabled SNDR Boost

To mitigate the impact of BL/SL wire resistances, we employ input scaling with γ_i , obtained through on-chip SEC training (Fig. 4.12(a)). This scaling results in a 4-to-6 dB enhancement in SNDR across ADC columns (Fig. 4.12(b)), thereby highlighting the pervasive effect of BL/SL resistances on SNDR_{col} . The average SNDR over 8 ADC columns increases from 4 dB to 9.15 dB in the presence of SEC.

Next, we measure the SNDR-energy trade-off using the following energy model per 1b compute:

$$E_{\text{op1}} = \frac{V_{\text{DD}} I_{\text{col}} N_{\text{cycles}}}{2N F_{\text{clk}}}, \quad (4.14)$$

where I_{col} is the measured column current, N_{cycles} is the number of clock cycles per IMC computation, and F_{clk} is the clock frequency. Figure 4.12(c) shows the enhancement of SNDR_{col} via SEC, ranging from 2.7-to-6 dB across various (N, F_{clk}) configurations and ADC precisions of $B_{\text{ADC}} = 4$ b and 6 b. This improvement in SNDR can be traded-off for a $5\times$ reduction in energy per 1b-operation (E_{op1}) including a 0.8% energy overhead of SEC (with a corresponding area overhead of 12.2%). The SEC processor's area overhead is primarily attributed to the extensive use of registers, which occupy more than 80% of its total area. However, this area overhead can be reduced to less than 3% by employing an SRAM. Notably, the SNDR boost facilitated by SEC is less pronounced for $(N = 128, F_{\text{clk}} = 16.6 \text{ MHz})$, suggesting an increased impact of noise sources. Additionally, increasing B_{ADC} from 4 b to 6 b does not yield a corresponding improvement in SNDR_{col} for $N = 128$, underscoring the fact that ADC quantization noise does not dominate in this scenario.

4.5.3 Network Level Accuracy

To evaluate network-level accuracy, we deploy the last layer of a ResNet-20 model (CIFAR-10) onto the MRAM array. The last layer (fully connected), computing an MVM, allows us to examine the influence of additive noise arising from analog non-idealities in MRAM IMC. Noise in the last layer has a significant impact on network accuracy, as it directly impacts the softmax outputs and, hence, the predicted label. Note that we evaluate the Top-1 inference accuracy for CIFAR-10 in this work.

The network is trained with 4 b quantized weights and floating-point inputs, resulting in a digital baseline with 91.1 % inference accuracy using 1000 test inputs quantized to 7 b. Our measurements show an accuracy of 74.8% with a confidence interval $C_I = \pm 5\%$ and a confidence level $C_L = 90\%$ [74]. We emphasize that we do not use noise-aware training [19, 23, 24, 39, 148] to boost the inference accuracy. Activating on-chip SEC increases the inference accuracy to 82.0 % without requiring network-specific noise-aware training. This observed 7.2 % improvement aligns closely with the average SEC-enabled SNDR boost of 3 dB across the ADC columns, validating the effectiveness of our SEC method in enhancing inference accuracy. However, the still persistent gap between the accuracy of fixed-point digital compute and that of post-SEC IMC highlights the need to explore SNDR-boosting methods further.

4.6 Conclusion

This chapter describes the first work to characterize the bank-level compute SNDR for an MRAM-based IMC. We proposed two SNDR-boosting methods: OCCS to reduce the static ADC column mismatch, and an algorithmic SEC technique to compensate for non-linearity due to BL/SL wire parasitics. We demonstrate a substantial SEC-enabled SNDR boost ranging from 2.6-to-6 dB. This boost can be traded off to achieve a noteworthy $5\times$ reduction in energy consumption. The average SEC-enabled SNDR boost of 3 dB correlates with a discernible increase in the inference accuracy of ResNet-20 (CIFAR-10), elevating it from 74.8 % to 82.0 %.

While OCCS and SEC methods are particularly advantageous for MRAM-based IMCs, where the lower on-off ratio exacerbates analog non-idealities

and lowers compute SNDR, they can also benefit other eNVM technologies like ReRAM and FeFET [149]. Notably, these accuracy-boosting methods become increasingly crucial for higher dot-product dimensions, where the limited signal swing can degrade SNDR even with a higher on-off ratio. Additionally, today’s resistive IMCs have a small fraction ($< 10\%$) devoted to the memory array, with the rest dedicated to peripherals. Hence, to fully utilize the potential of eNVM’s high memory density within the context of an IMC, future research needs to prioritize reducing the size of sensing circuits while employing accuracy-boosting techniques such as SEC and noise-aware training.

CHAPTER 5

ON THE SECURITY VULNERABILITIES OF MRAM-BASED IN-MEMORY COMPUTING ARCHITECTURES

In the previous chapters, **3** and **4**, we have established that resistive IMCs have low compute SNDR and rely on accuracy boosting techniques to achieve high network-level accuracy. When deployed on Edge devices, resistive IMCs enable on-chip storage of DNN model parameters reducing the latency and energy costs associated with off-chip data movement. However, storing DNN parameters on-chip expands the attack surface of Edge devices, posing security threats such as model extraction attacks [162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172]. Attackers with access to a victim Edge device can exploit controllable inputs and corresponding observable outputs, creating a strong attack scenario for mounting MEAs.

Considering the Edge application of resistive IMCs, we ask the following question in this chapter:

Does the low compute accuracy of resistive IMC make them secure?

The presence of analog noise in resistive IMC computation suggests that they may be intrinsically secure to model extraction attacks, which are a critical vulnerability in current AI accelerators. These attacks allow the attacker to retrieve DNN model parameters [171, 168, 167, 164, 162] and reconstruct DNN architectures [170, 165, 172] from various signatures. Since it is possible to infer the training data from the retrieved model, e.g., via techniques such as membership inference attacks [173, 174, 175] or reconstruction attacks [176, 177], MEAs can lead to the leakage of sensitive private data that may have been used for training the models.

While MEAs for digital accelerators have been studied extensively [162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172], not much work has been done on understanding the security vulnerabilities of IMCs. Part of the reason being that real-life IMC platforms are unavailable for security researchers to work with. Hence, in [178, 179], MEAs for resistive RAM (ReRAM)-based

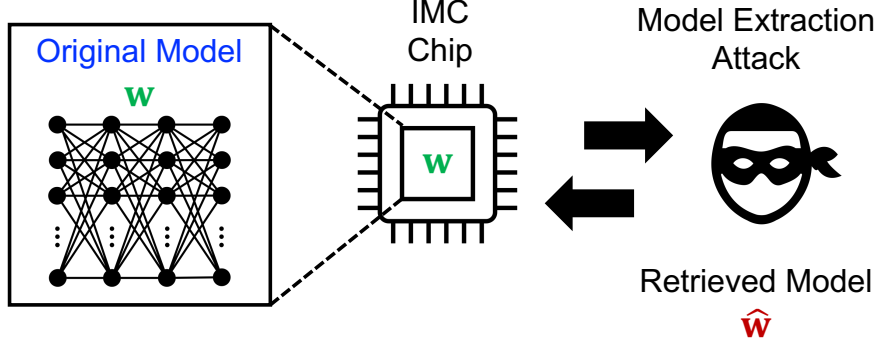


Figure 5.1: Model extraction attack (MEA) on IMCs.

IMCs were developed by employing circuit simulations of the IMC array and their power supply grid. The resulting supply current traces were analyzed to construct MEAs, e.g., [178] tries to infer the network architecture (a coarse parameter), while [179] exploits power side-channel to extract model parameters. A key limitation of [179] is the small dot-product (DP) dimension of 16 which makes it hard to scale their method to higher DP dimensions of 64 or 128 found in today’s eNVM-based IMC prototypes [19, 23, 24, 25, 26, 114]. The reason being the severely limited computational accuracy of underlying the matrix-vector multiply (MVM) operation due to the presence of analog non-idealities (noise, distortion, and process, voltage, and temperature variations) in eNVM-based IMCs, which can obfuscate the model retrieval process. Furthermore, to date, none of the previous works have demonstrated the efficacy of their attacks on real-life IC prototypes of eNVM-based IMCs.

In this chapter, we develop a statistical framework to systematically study the security vulnerability of eNVM IMCs, employ it to develop MEAs for MRAM-based IMCs, and use them to extract DNN model parameters from a MRAM-based IMC prototype in a 22 nm FD-SOI process under a strong attack scenario where the attacker strives to retrieve the model parameters $\mathbf{w}$ stored in the memory via input-output queries as shown in Fig. 5.1 [180].

Specifically, the contributions of this chapter are:

- **A Statistical MEA framework:** We propose a statistical framework to study the security vulnerabilities of eNVM-based IMCs. This framework models the statistics of IMC computation using via circuit-aware stochastic non-linear functions.
- **MEAs for MRAM-based IMCs:** We employ the MEA framework

Table 5.1: Threat models for eNVM-based IMCs.

Threat Model	Attacker Privilege			
	Read-Write	Input	Output	Side-channel
Passive	×	×	×	✓
Active 1	×	✓	×	✓
Active 2	×	✓	✓	×

to construct three query-based attacks: *basis vector* (BV), *least squares* (LS), and *stochastic gradient descent* (SGD) attack.

- **Efficacy Metrics for MEAs:** We propose multiple metrics to quantify the efficacy of MEAs for IMCs. At the bank-level, the bit-error rate (BER) metric is used to measure the fidelity of retrieved model. At the network level, we propose mismatch probability to compare the fidelity of predictions, in addition to the prediction accuracy.
- **Measured Results from IMC prototype:** We apply the three MEAs to a MRAM-based IMC prototype in 22 nm FD-SOI, and demonstrate their efficacy using all three metrics. Specifically, we find that the strongest attack (SGD) is able to achieve an inference accuracy within 0.1% of the fixed-point digital baseline model.
- **Energy-accuracy-security trade-offs in IMC:** We analyze the security vulnerabilities of the MRAM-based IMC prototype at different compute SNDR and energy under various operating conditions. Our findings indicate that the efficacy of attacks is inversely proportional to the compute SNDR, with attacks failing to achieve high inference accuracy in the low-bias operating condition.

These results confirm the existence of vulnerability of MRAM-based IMCs to MEAs, even in the presence of their intrinsic noise. *To the best of our knowledge, this is the first work to experimentally demonstrate the efficacy of MEAs for eNVM-based IMCs on a real-life IC prototype.*

5.1 Threat Model for eNVM-based IMCs

MEAs aim to retrieve DNN model parameters by analyzing input-output queries, side-channel signatures, or both. The DNN maps an input $\mathbf{u} \in \mathbb{R}^D$ to a label $v \in [C]$, using the function $f_\theta(\mathbf{u}) : \mathbb{R}^D \rightarrow [C]$, where $[C] = \{1, 2, \dots, C\}$ with C being the number of classes, and θ is the set of model parameters comprising weight matrices of each layer in the network. The trained model parameters include multiple MVM operations with weight matrices and non-MVM operations such as ReLU and batch normalization. In MEAs, the attacker strives to retrieve an estimate $\hat{\theta}$ of the model parameter θ stored in multiple IMC banks. At the DP level, the attack involves retrieving an estimate $\hat{\mathbf{w}}$ of the weight vector $\mathbf{w}$ stored in the memory. We assume that the attacker has access to the non-MVM parameters such as bias and batchnorm parameters. This is a reasonable assumption to make since these computations involving these parameters are computed in a SIMD processor associated with the IMC bank [89] and, hence, can be retrieved using standard MEAs for digital computations. Thus, the attacker is able to reconstruct a DNN model $\hat{f}$ from weight estimates $\hat{\mathbf{w}}$. A successful attack would result in predictions $\hat{f}(\mathbf{u})$ identical to $f_\theta(\mathbf{u})$ with high probability.

In all attack scenarios, we assume that the weights are initially written into the non-volatile memory in a secure environment and subsequently utilized solely for inference purposes. The attacker lacks direct read/write access to the memory but can execute the IMC operation during the inference stage. In the following, we describe the specific threat models (see Table 5.1) depending on the information available to the attacker:

1. **Passive:** The attacker can only measure side-channel information such as power consumption [181] or electromagnetic emanations [170, 171] during the MVM operation and/or during the bus transfers to and from the IMC. This is a typical scenario for Edge devices deploying eNVM-based IMCs.
2. **Active 1:** The attacker can control the IMC inputs and observe side-channel information. Hence, the attacker also knows the dot-product input dimension N . This is a commonly used threat model for MEAs on Edge devices such as microcontrollers [171] and FPGA [165].
3. **Active 2:** The attacker can control the inputs into the IMC, and observe

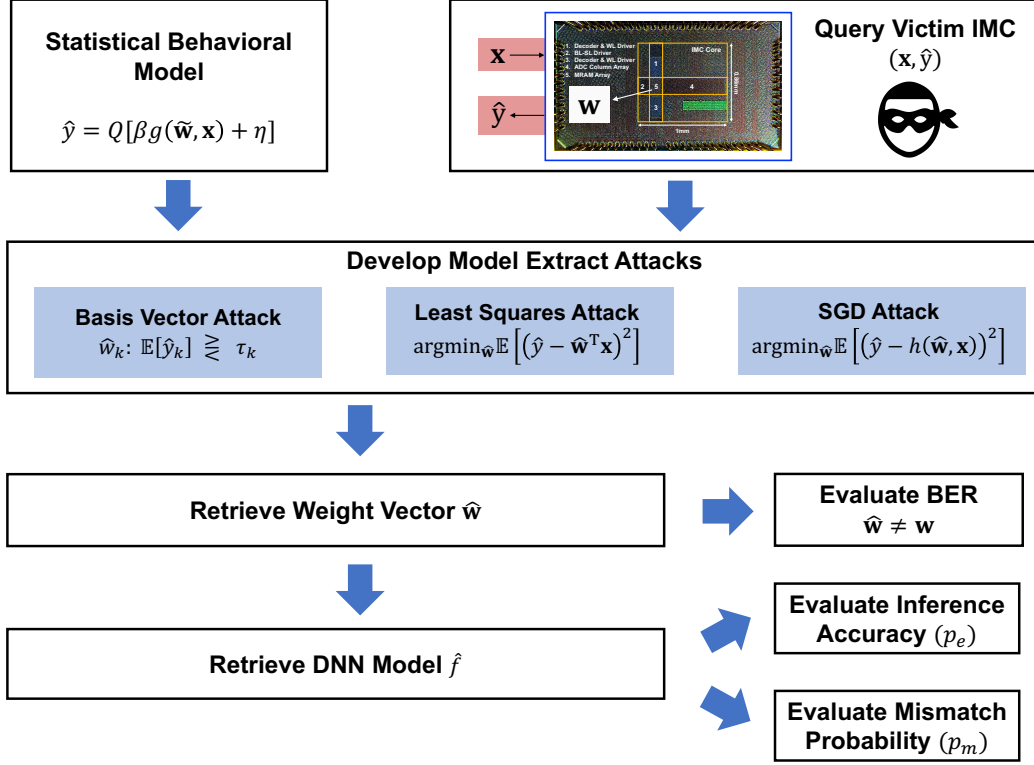


Figure 5.2: A statistical framework for constructing MEAs for eNVM-based IMCs.

the outputs of the IMC. This threat model has been employed in the cloud [168, 167].

In this work, we consider only the *Active 2* threat model which is a strong attack scenario. This allows us to assess the true security vulnerabilities of eNVM-based IMC arising only due to its analog non-idealities and not being limited by the measurement noise in the side-channel signatures.

5.2 A Statistical MEA Framework for eNVM-based IMCs

Unlike digital architectures where the computation is deterministic and where the input-output queries made by the attacker can reveal the DNN mapping [170, 172] or per-layer computations [168, 169, 164, 167, 162], the intrinsic noise in IMCs makes their outputs stochastic. The signal-to-noise and distortion ratio (SNDR) is a metric to quantify the computational ac-

curacy (compute SNDR) in IMCs. In eNVM-based IMCs, the low compute SNDR [38, 19, 30, 113, 59, 40, 42, 37, 36, 15, 60, 43, 114, 115, 116, 43, 23, 24, 25, 26] stems from the fundamental challenge in sensing low resistances in the presence of the analog non-idealities [149]. Hence, traditional MEAs that exploit deterministic side-channel leakage models, such as Hamming weight [182] or Hamming distance [183], to estimate model parameters [171, 165, 179], cannot be applied directly to IMCs.

To address the challenge of constructing MEAs for eNVM-IMCs, we propose the statistical framework shown in Fig. 5.2 which comprises the following steps:

1. **Query Attack:** The attacker queries the victim IMC chip to extract input-output query pairs $(\mathbf{x}, \hat{y})$.
2. **Statistical Behavioral Model:** The attacker then formulates a generic behavioral model of eNVM-based IMCs that describes the IMC output $\hat{y}$ in terms of its inputs $\mathbf{x}$, weights $\mathbf{w}$, and various analog non-idealities [149].
3. **Develop MEAs:** The attacker employs the statistical behavioral model to develop a variety of MEAs. and tests them out on the input-output query pairs $(\mathbf{x}, \hat{y})$.
4. **Apply MEAs:** The attacker tests the MEAs on the input-output query pairs $(\mathbf{x}, \hat{y})$. First, the weight vector is retrieved ($\hat{\mathbf{w}}$) and then the DNN model $\hat{f}$ is reconstructed.
5. **Evaluate MEA Efficacy:** Though this last step is not available to the attacker since it requires the knowledge of the true model parameters, we employ it to evaluate the efficacy of the MEAs. The efficacy of the MEAs is evaluated at the bank-level and at the network (DNN) by employing both fidelity and accuracy metrics.

The statistical behavioral model in Step 2 of the MEA construction framework (see Fig. 5.2) captures the stochasticity in the IMC output due to the following non-idealities [149]:

1. **Conductance variation (CV):** These are manufacturing-induced spatial conductance variations that appear in each BCs, and are fixed for a specific ADC column.

2. **Parasitic conductance (PC):** Non-zero routing resistances in the eNVM array create a voltage gradient along the BLs and SLs, introducing an input and weight dependent non-linearity in the BL current.
3. **Current mirror mismatch (MM):** The W/L mismatch in the current mirror of the read-out circuit leads to variations in the current scaling factors across ADC columns.
4. **ADC thermal noise (AT):** This is the input-referred thermal noise in the comparator in a SAR ADC.

These non-idealities result a noisy version $\hat{y}$ of the ideal DP y_o in (3.1) being computed by the IMC (statistical behavioral model) as defined below:

$$\hat{y} = Q[\beta g(\tilde{\mathbf{w}}, \mathbf{x}) + \eta], \quad (5.1)$$

where $\tilde{\mathbf{w}}$ is the weight vector in presence of conductance variations (CV) whose k -th element $\tilde{w}_k = w_k + \delta w_k$ is stored in the k -th BC, $g(\cdot)$ is a non-linear function of $\tilde{\mathbf{w}}$ and $\mathbf{x}$ capturing the impact of wire parasitics (PC), β is the multiplicative factor modeling current mirror mismatch (MM) across ADC columns, $\eta \sim \mathcal{N}(0, \sigma_t^2)$ is the ADC input referred thermal noise (AT), and $Q[\cdot]$ is the ADC quantizer function.

The statistical behavioral model in (5.1) simplifies to $\hat{y} = y_o$ for an ideal DP computation. This ideal DP model, which is employed for model extraction in digital implementations [168, 164, 167, 170], cannot be used here due to noise in eNVM-based IMC computations. Figure 5.2 shows that the attacker can use to statistical behavioral model to formulate MEAs for eNVM-based IMCs.

5.2.1 Efficacy of MEAs

The weights $\hat{\mathbf{w}}$ retrieved from the IMC bank via MEAs can be used to reconstruct the model $\hat{f}$ (retrieved model) whose predictions can be compared with the fixed-point model f (ideal model). Hence, we employ the following metrics to evaluate the efficacy of our proposed MEAs in Section 5.3:

1. **Bit-error Rate:** This is a bank-level metric. We determine the closeness (fidelity) between the retrieved weight $\hat{\mathbf{w}} \in \{\pm 1\}^N$ and the stored

weight $\mathbf{w} \in \{\pm 1\}^N$ where N is the dot-product dimension. For a single ADC column, we compute the bit-error rate for $\hat{\mathbf{w}}$ as:

$$\text{BER} = \frac{\|\hat{\mathbf{w}} - \mathbf{w}\|_2^2}{4N}. \quad (5.2)$$

A low BER indicates fewer errors in the model $\hat{\mathbf{w}}$ retrieved from the IMC bank;

2. **Accuracy:** The misclassification probability over the inference set $\mathcal{D}$ of the ideal model f and the retrieved model $\hat{f}$ is given by [167, 162]:

$$p_e = \Pr\{v \neq v_t\}; \quad p_{\hat{e}} = \Pr\{\hat{v} \neq v_t\} \quad (5.3)$$

where v_t is the true label, v ($\hat{v}$) and p_e ($p_{\hat{e}}$) are the predicted label and misclassification error, respectively, of the fixed-point (retrieved) model f ($\hat{f}$). The misclassification error provides a measure of the accuracy of the model;

3. **Mismatch Probability:** This is a network-level metric. The mismatch probability p_m [141] between the predicted labels $\hat{v}$ and v of the retrieved model $\hat{f}$ and the ideal model f , respectively, is defined as:

$$p_m = \Pr\{\hat{v} \neq v\}. \quad (5.4)$$

The mismatch probability p_m captures the ability of the retrieved model $\hat{f}$ to replicate the predictions, both correct and incorrect, of the ideal model f .

Achieving high classification accuracy at the network level indicates a successful model retrieval, potentially enabling financial exploitation or inferring private training data via attacks like membership inference [173] or data reconstruction [176]. Additionally, a low p_m implies high retrieval fidelity which allows the attacker to generate effective white box adversarial attacks [184].

5.3 Proposed MEAs for eNVM-IMCs

Figure 5.3 illustrates the three proposed attacks based on the statistical framework in Sec. 5.2 targeted towards retrieving estimates of weight vectors

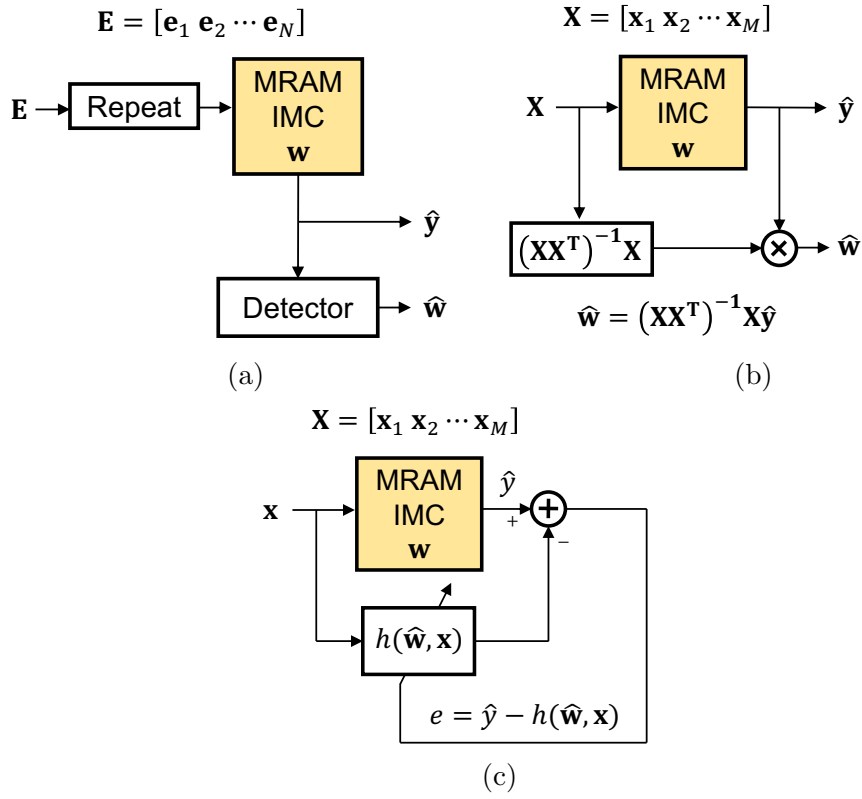


Figure 5.3: Proposed model extraction attacks: (a) basis vector, (b) least square, and (c) stochastic gradient descent attack.

$\hat{\mathbf{w}}$ for a given ADC column storing $\mathbf{w} \in \{\pm 1\}^N$. All the attacks can be extended to retrieve the weights from multiple ADC columns in the IMC, and therefore, the DNN model $\hat{f}$.

5.3.1 Basis Vector (BV) Attack

In this attack, we use the property of the basis vectors to isolate each weight element. We attempt to retrieve the k -th element of $\mathbf{w}$ using the k -th basis vector $\mathbf{e}_k$ with $e_{k,k} = 1$ and $e_{k,j} = 0 \ \forall \ j \neq k$. Only N input vectors are required to retrieve the N -dimensional $\mathbf{w}$ in the ideal case of $\hat{y} = y_o$. However, in the presence of IMC noise, $> N$ samples of $\hat{y}$ ((5.1)) are required to achieve accurate retrieval, either using $N_{\text{rep}} > 1$ repeated queries for each input vector, or using $M > N$ input vectors, or both, resulting in $M \times N_{\text{rep}}$ IMC queries. Figure 5.3(a) shows the attack setup, which comprises the following:

1. **Input generation:** We use the antipodal representation of $\mathbf{e}_k$, i.e., $\mathbf{1} - 2\mathbf{e}_k$, to obtain $\hat{y}_k$ for a single ADC column following (5.1) as:

$$\hat{y}_k = \frac{Q[\beta g(\tilde{\mathbf{w}}, \mathbf{1}) + \eta_1] - Q[\beta g(\tilde{\mathbf{w}}, (\mathbf{1} - 2\mathbf{e}_k)) + \eta_k]}{2}, \quad (5.5)$$

where, $\eta_1, \eta_k \sim \mathcal{N}(0, \sigma_t^2)$ from (5.1), and $\mathbf{1}$ is the all ones input. We extend the basis computation in (5.5) over input set $\mathbf{E} = \{\mathbf{e}_k\} \ \forall \ k \in \{1, 2, \dots, N\}$, while using N_{rep} repeated queries of $\mathbf{E}$ resulting in a total of $N \times N_{\text{rep}}$ IMC queries.

2. **Output averaging:** We average the IMC output $\hat{y}$ over N_{rep} repeated queries of $\mathbf{E}$. The averaged output $\tilde{y}_k$ for input $\mathbf{e}_k$ is given by:

$$\tilde{y}_k = \frac{\sum_{l=1}^{N_{\text{rep}}} \left(\hat{y}_k^{(l)} \right)}{N_{\text{rep}}}, \quad (5.6)$$

where $\hat{y}_k^{(l)}$ corresponds to the IMC output in the l -th repetition.

3. **Detection:** The averaged output $\tilde{y}_k$ is compared with a threshold τ_k to

estimate w_k as follows:

$$\hat{w}_k = \begin{cases} 1 & \tilde{y}_k \geq \tau_k \\ -1 & \tilde{y}_k < \tau_k \end{cases}. \quad (5.7)$$

Extending the detection (5.7) in for all $k \in \{1, 2, \dots, N\}$ we retrieve $\hat{\mathbf{w}}$.

Note: As N_{rep} increases, $\tilde{y}_k$ approaches the expected value $\mathbb{E}[\hat{y}_k]$ since the expectation of $Q[\cdot]$ function in (5.1) can be approximated as $\beta g(\tilde{\mathbf{w}}, \mathbf{x})$ for a coarse quantizer, and the non-linear function $g(\cdot)$ can be linearized when the input is a basis vector. Doing so reveals the need for a per-dimension threshold τ_k for retrieving w_k . In our proposed BV attack, we assume identical thresholds $\tau_k = \tau \forall k \in \{1, 2, \dots, N\}$ in (5.7) and sweep τ to obtain its optimal value ($\tau = 0$) which minimizes the BER.

5.3.2 Least Square (LS) Attack

In this attack, we minimize the squared error of retrieval using the linear least square (LS) solution. Figure 5.3(b) shows our proposed LS attack, which includes the following steps:

1. **Input-output generation:** We generate M input vectors, each of dimension N stored in a matrix $\mathbf{X} \in \{\pm 1\}^{N \times M}$ with i.i.d. $x \in \{\pm 1\} \sim \text{Be}(0.5)$ and apply it to the input of the IMC, resulting in outputs $\hat{\mathbf{y}} \in \mathbb{R}^{M \times 1}$.
2. **LS solution:** We estimate $\hat{\mathbf{w}}$ via the LS solution minimizing the L2 squared error $\|\hat{\mathbf{y}} - \hat{\mathbf{w}}^T \mathbf{X}\|_2^2$, as follows:

$$\hat{\mathbf{w}} = (\mathbf{X}\mathbf{X}^T)^{-1} \mathbf{X}\hat{\mathbf{y}}. \quad (5.8)$$

In order to compute the inverse, $\mathbf{X}\mathbf{X}^T \in \mathbb{R}^{N \times N}$ needs to be full rank matrix requiring $\mathbf{X}$ to have the minimum of N linearly independent columns, resulting in $M \geq N$. As M increases, the empirical solution in (5.8) approaches the minimum mean square error (MMSE) estimate given by $\mathbb{E}[\mathbf{x}\mathbf{x}^T]^{-1} \mathbb{E}[\mathbf{x}\hat{\mathbf{y}}]$. For $x \sim \text{Be}(0.5)$ the autocorrelation matrix $\mathbb{E}[\mathbf{x}\mathbf{x}^T]$ is identity $\mathbf{I}$, leading to $\hat{\mathbf{w}} = \mathbb{E}[\mathbf{x}\hat{\mathbf{y}}]$. This highlights that the LS solution uses the joint statistics of

$(\mathbf{x}, \hat{y})$ as opposed to estimating $\mathbb{E}[\hat{y}]$ through repeated queries as in the BV attack.

5.3.3 Stochastic Gradient Descent (SGD) Attack

In developing the stochastic gradient descent (SGD) attack (see Fig. 5.3(c)), the attacker assumes a specific form for the non-linear function $h(\hat{\mathbf{w}}, \mathbf{x})$ to estimate $g(\tilde{\mathbf{w}}, \mathbf{x})$ in (5.1). The SGD attack minimizes the mean squared error $L(\hat{\mathbf{w}}) = \mathbb{E}[(\hat{y} - h(\hat{\mathbf{w}}, \mathbf{x}))^2]$ iteratively as follows:

1. **Input-output generation:** Random inputs $\mathbf{x} \in \{\pm 1\}^N$ with i.i.d. $x \sim \text{Be}(0.5)$ are fed to the IMC to obtain $\hat{y}$. A total of M unique $(\mathbf{x}, \hat{y})$ input-output pairs are used for the SGD update.
2. **Initialization:** Randomly initialize $\hat{\mathbf{w}}$ with $\hat{w} \sim \text{Be}(0.5)$ and $\mathbf{a}, \mathbf{b}$ with $a, b \sim \mathcal{N}(0, 1)$.
3. **SGD update:** The error $e = \hat{y} - h(\hat{\mathbf{w}}, \mathbf{x})$ computed with a non-linear model:

$$h(\hat{\mathbf{w}}, \mathbf{x}) = \hat{\mathbf{w}}^T \mathbf{x} + (\mathbf{a} \odot \hat{\mathbf{w}}^2)^T \mathbf{x}^2 + (\mathbf{b} \odot \hat{\mathbf{w}}^3)^T \mathbf{x}^3, \quad (5.9)$$

where $\mathbf{a}, \mathbf{b} \in \mathbb{R}^N$ are the quadratic and the cubic non-linear scaling parameters, and $\odot$ is the element-wise multiply operation, and $\mathbf{z}^n = \mathbf{z} \odot \mathbf{z} \odot \dots \odot \mathbf{z}$ (n times) is the element-wise n -way exponentiation of vector $\mathbf{z}$.

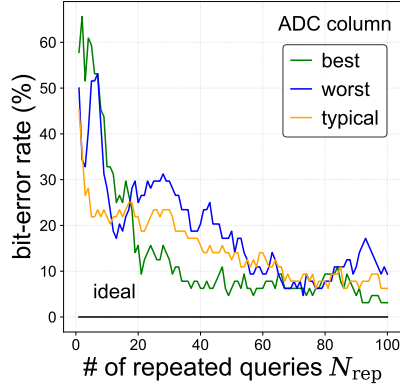
The update rule for $\hat{\mathbf{w}}, \mathbf{a}$, and $\mathbf{b}$, derived from the fundamental SGD equation, is given by:

$$\hat{\mathbf{w}} := \hat{\mathbf{w}} + \mu e (\mathbf{x} + 2\mathbf{a} \odot \hat{\mathbf{w}} \odot \mathbf{x}^2 + 3\mathbf{b} \odot \hat{\mathbf{w}}^2 \odot \mathbf{x}^3) \quad (5.10)$$

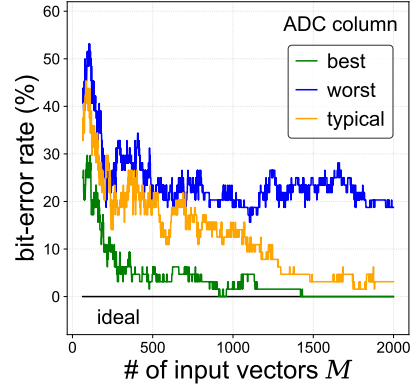
$$\mathbf{a} := \mathbf{a} + \mu e (\hat{\mathbf{w}}^2 \odot \mathbf{x}^2); \mathbf{b} := \mathbf{b} + \mu e (\hat{\mathbf{w}}^3 \odot \mathbf{x}^3) \quad (5.11)$$

where μ is the learning rate. The updates are iteratively computed for T epochs comprising the same set of M inputs per epoch.

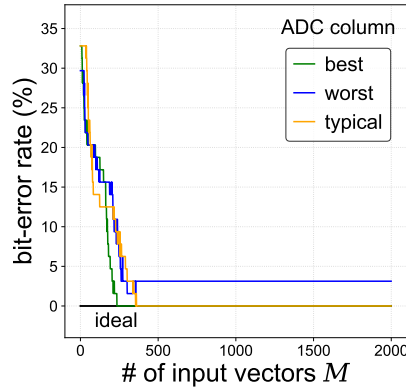
Note that in all the attack methodologies, including BV, LS, and SGD attacks, we binarize the retrieved weights $\hat{\mathbf{w}}$ to $\{\pm 1\}$ to match the quantization scheme of the binary antipodal weights stored in the MRAM array.



(a)

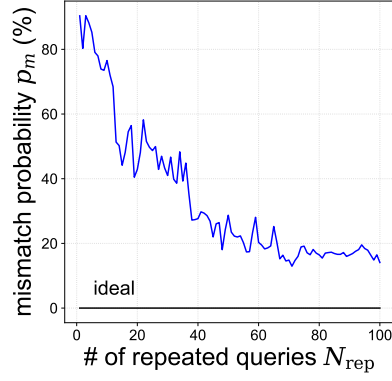


(b)

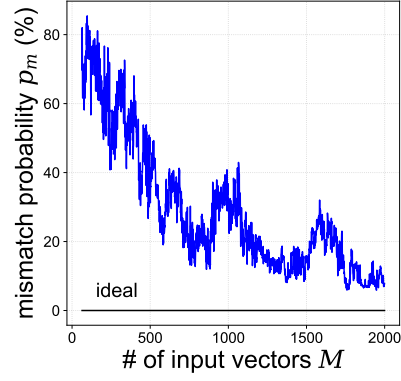


(c)

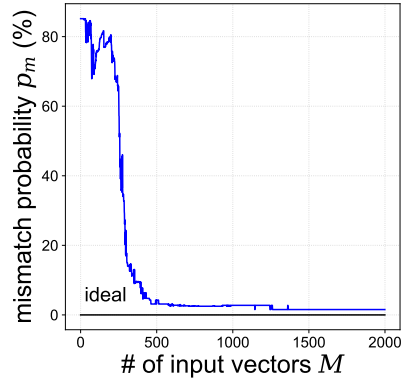
Figure 5.4: ADC column bit-error rate: (a) basis vector, (b) least square, and (c) SGD attack.



(a)

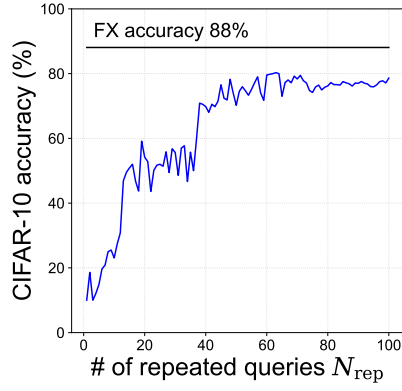


(b)

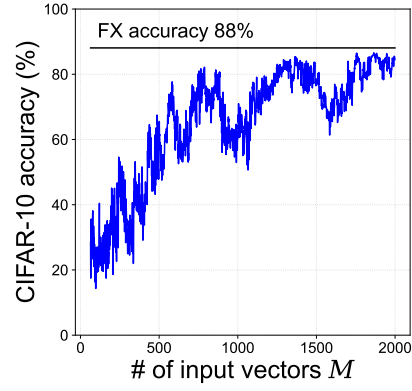


(c)

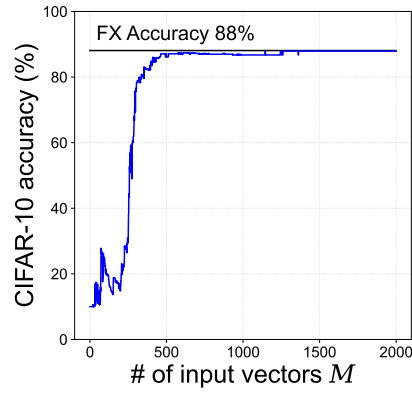
Figure 5.5: Probability of mismatch between predictions of the retrieved vs. original ResNet-20 model on CIFAR-10: (a) basis vector, (b) least square, and (c) SGD attack.



(a)



(b)



(c)

Figure 5.6: Inference accuracy of retrieved ResNet-20 model on CIFAR-10: (a) basis vector, (b) least square, and (c) SGD attack.

5.4 Measured Results

In this work, we apply our proposed attacks on an MRAM IMC prototype IC in 22 nm FD-SOI process, following the *Active 2* threat model (see Table 5.1). We perform model extraction for a ResNet-20 architecture trained over the CIFAR-10 dataset, focusing on recovering the weights for the final layer (64×10) stored in MRAM-based IMC. For this purpose, we used a trained network with 4 b quantized weights and floating-point inputs, achieving a digital baseline accuracy of 88.02% using 10000 test inputs quantized to 7 b. In our case, the retrieved model $\hat{f}$ includes the retrieved last layer weights and the fixed point weights of previous layers. We evaluate the efficacy of our attack over the bank-level and the network-level metrics, as described in Sec. 5.2.1.

5.4.1 Attack Setup

We use the same measurement setup as shown in Fig. 4.9 of **Chapter 4** which includes a Python interface, a PYNQ FPGA to facilitate communication with the MRAM-based IMC and a PCB that houses the MRAM-based IMC prototype IC. The power supply and the bias voltages are generated on the PCB, providing a digital interface for input/output communication with the chip. In this work, the attacker queries the chip, providing the input vectors and observing the corresponding output.

The MRAM-based IMC prototype IC has a storage capacity of 256 kB and a configuration of 512×512 , with a 512 b input bus. This bus activates the word line (WL) in the MRAM memory array, computing an equivalent conductance G_{eq} proportional to the dot product y_o (see (3.1)). The IMC comprises 128 ADC columns, each computing 64-dimensional DP resulting in a 6 b ADC output $\hat{y}$ (see (5.1)).

5.4.2 Bank-level Efficacy

Figure 5.4(a) shows the reduction in the BER for the best, worst, and typical case ADC columns as we increase the number of repeated queries in the basis vector attack. We choose $\tau = 0$ for all the rows ($\tau_k = 0 \forall k \in \{1, 2, \dots, N\}$

in (5.5)), since it achieves the lowest BER of $\approx 4\%$ or < 3 errors in a 64-dimensional DP. This indicates that $\mathbb{E}[\hat{y}_k]$ is close to w_k for all k . The sample mean $\tilde{y}_k$ in (5.6) approaches $\mathbb{E}[\hat{y}_k]$ as N_{rep} increases, leading to low BER. The BV attack can be improved by choosing non-identical values of τ_k .

Figure 5.4(b) and (c) show the BER computed by averaging of $\hat{\mathbf{w}}$ over $N_{\text{rep}} = 100$ separate runs with the same set of M randomly generated input vectors. The BER drops for both LS (Fig. 5.4(b)) and SGD (Fig. 5.4(c)) attacks as the number of input vectors M increase. The minimum BER varies across ADC columns as their SNDR are different, leading to different noise levels in the query outputs $\hat{y}$. The BER for LS attack achieves a perfect retrieval for the best case ADC column with 875 IMC queries, whereas it levels off at the $\approx 4\%$ for $M > 1400$ in the typical case and only reaches a minimum of 15% with 2000 queries in the worst case.

On the one hand, LS performs better than the BV attack in terms of BER and number of queries, as it utilizes the joint statistics of the input and output, as opposed to only outputs in BV, but fails to achieve a low BER across multiple ADC columns. This is because minimizing the squared error with a linear model ($\hat{\mathbf{w}}^T \mathbf{x}$) fails to capture the non-linearity due to wire parasitics. On the contrary, the SGD attack with the non-linear model $h(\hat{\mathbf{w}}, \mathbf{x})$ in (5.9) can close the gap in BER achieving a perfect reconstruction (Fig. 5.4(c)) in the typical (best) case with a significantly lower number $M = 355$ ($M = 235$) input vectors. In the worst case, the SGD attack achieves a BER = 1.5% or one error in retrieving a 64-dimensional DP.

5.4.3 Network-level Efficacy

Next, we use the $\hat{\mathbf{w}}$ estimates from 40 ADC columns, to retrieve the last layer weights (64×10), and therefore the DNN model $\hat{f}$, and compare its predictions with that of the ideal (fixed-point) model f .

Mismatch Probability: Figure 5.5 shows the drop in mismatch probability p_m ((5.4)) as the number of IMC queries increase for the three attacks. The BV attack (Figure 5.5(a)) attains a minimum p_m of 12.94% with 70 repeated queries of the basis set $\mathbf{E}$, which reduces to 5.83% in case of LS attack (Fig. 5.5(b)) with 1883 queries, and further to 1.54% for SGD attack (Fig. 5.5(c)) with 1145 queries.

In all cases, the minimum p_m values are tied to the minimum BER achieved across ADC columns (Fig. 5.4). The large BER in the BV and LS attacks contributes to their p_m being larger than that of the SGD attack. The latter achieves the lowest p_m owing to perfect retrieval of $\hat{\mathbf{w}}$ from 60% of the ADC columns, and minimum BER of 1.5% for the remaining ones, indicating a high fidelity of model extraction where there is a close agreement between the outcomes of the retrieved model $\hat{f}$ and the ideal model f .

Note that the mismatch probability p_m serves as a proxy for the misclassification probability $p_{\hat{e}}$ ((5.3)) of the retrieved model since it is upper bounded as [141]:

$$p_{\hat{e}} \leq p_e + p_m, \quad (5.12)$$

where p_e is the misclassification probability of the fixed-point model in (5.3). Thus, a small p_m predicts a small p_e , and since the SGD attack achieves the smallest p_m , we expect to perform better than the BV and LS attacks in terms of inference accuracy, as shown next.

Inference Accuracy: Figure 5.6 contrasts the network-level efficacy of the three attacks via CIFAR-10 accuracy achieved by the retrieved model. Similar to the trends in Fig. 5.5 for p_m , the inference accuracy for the BV attack (Fig. 5.6(a)) maxes out at 80% with 60 repeated queries of the basis set **E**. The accuracy increases to 86.49% with 1755 queries for the LS attack (Fig. 5.6(b)), and to 87.91% with only 1145 queries for the SGD attack (Fig. 5.6(c)), coming within a 0.1% margin of the fixed-point digital baseline.

These results satisfy the inequality in (5.12) where the classification error $p_{\hat{e}}$ of the retrieved model remains within the upper limit defined by the sum of the p_e of the ideal model f and the mismatch probability p_m . Interestingly, the SGD attack comes close to satisfying the equality in (5.12) with $p_{\hat{e}} = 12.09\% < 13.52\%$ indicating the scenario where the majority of the misclassified samples of the ideal model f do not overlap with the errors introduced by incorrect weight retrieval. These findings underscore the susceptibility of MRAM-based IMCs to MEAs when evaluated for both the network-level fidelity and the accuracy metrics. However, this susceptibility observed at the highest bank-level SNDR for MRAM-based IMC raises an important question: *Does lowering the SNDR make MRAM-based IMCs more secure?* We address this question in the following section.

5.5 Energy-Accuracy-Security Trade-offs in IMCs

In this section, we evaluate the efficacy of the proposed MEAs discussed in Section 5.3 under various operating conditions for our MRAM-based IMC prototype, examining the trade-off in compute SNDR, energy, and security against MEAs.

5.5.1 Experimental Setup

IMCs are known to prescribe to the energy-accuracy trade-off, as shown in Fig. 4.12(c) of **Chapter 4**. This trade-off is characterized by adjusting operating conditions such as the bias current (I_B) of the CS loop, clock frequency (F_{clk}), and ADC precision (B_{ADC}). For our MRAM-based IMC prototype, changing these parameters impacts the compute SNDR and energy as follows:

- **Bias current I_B :** This sets the bit-line voltage (V_{BL}) in the CS loop, which determines the signal current (I_{CS}) as described by (4.5). Lowering I_B reduces the signal current I_{CS} and thereby the energy per 1b compute (E_{op1}) following (4.14). Our previous analysis in Section 3.3.2 indicates that the SNDR_a and, therefore, SNDR_d drops at low signal magnitudes due to the presence of ADC thermal noise.
- **Clock frequency F_{clk} :** This determines the duration of the auto-zero (AZ) phase in the offset compensation (Fig. 4.3 (b)). The voltage sampled on the C_{os} capacitor affects the accuracy of offset compensation, which decreases if the sampling duration is shorter than the settling time of the inner loop. Additionally, higher F_{clk} reduces the latency of computation, thereby reducing E_{op1} .
- **ADC precision B_{ADC} :** This determines the number of output levels to which the CS loop current (I_{CS}) is quantized. Reducing B_{ADC} lowers SNDR_d when quantization noise is dominant. Furthermore, similar to increasing F_{clk} , reducing B_{ADC} decreases the number of clock cycles N_{cycles} , reducing E_{op1} ((4.14)).

Based on the impact of the parameters described above, we choose the following operating conditions:

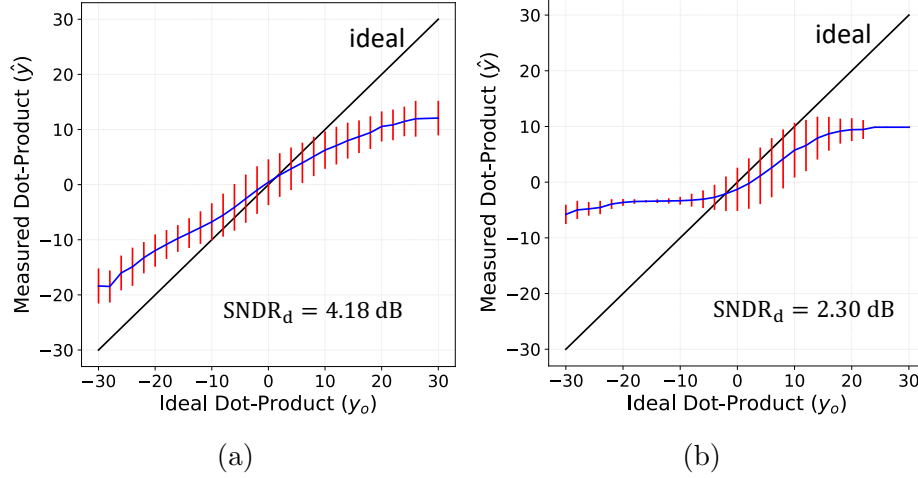


Figure 5.7: Measured dot-product ($\hat{y}$) 1- σ error bars for random inputs with $x \in \{\pm 1\} \sim \text{Be}(0.5)$ for an ADC column under: (a) HB-LF and (b) LB-LF operating conditions.

1. **High bias, low frequency (HB-LF):** with $I_B = 92 \mu\text{A}$, and $F_{\text{clk}} = 8.3 \text{ MHz}$.
2. **Low bias, low frequency (LB-LF):** with $I_B = 67 \mu\text{A}$, and $F_{\text{clk}} = 8.3 \text{ MHz}$.
3. **High bias, high frequency (HB-HF):** with $I_B = 92 \mu\text{A}$, and $F_{\text{clk}} = 16.6 \text{ MHz}$.
4. **Low bias, high frequency (LB-HF):** with $I_B = 67 \mu\text{A}$, and $F_{\text{clk}} = 16.6 \text{ MHz}$.

In all cases, the ADC LSB step size and offset are tuned to match the signal current range to the input range of the 6 b SAR ADC. Furthermore, we obtain the IMC output for each of the above operating conditions at $B_{\text{ADC}} \in \{1, 2, \dots, 6\}$ by choosing the number of SAR conversion cycles.

Figure 5.7 shows the change in the ADC column transfer function when switching from HB-LF to LB-LF operating condition. In the HB case with higher signal currents (Fig. 5.7(a)), the non-linearity in the transfer function arising from parasitic conductances (PC) is dominant and determines the SNDR_d . Reducing the bias current leads to a drop in SNDR_d at low signal currents (Figure 5.7(b)). We suspect this SNDR_d drop is linked to the increased noise of both the CS-loop and ADC at low-bias operating conditions.

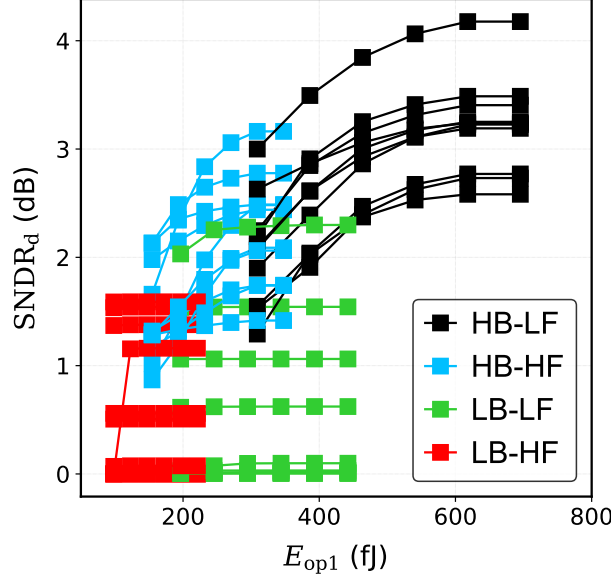


Figure 5.8: Energy-accuracy trade-off curves at different operating conditions (HB-LF, LB-LF, HB-HF, and LB-HF) obtained by sweeping $B_{\text{ADC}} \in \{1, 2, \dots, 6\}$ for ten ADC columns.

Figure 5.8 illustrates the SNDR_d vs. E_{op1} trade-off across various operating conditions. The highest SNDR_d (black curve) is achieved for the HB-LF condition at the highest E_{op1} when $B_{\text{ADC}} = 6$. The maximum SNDR_d incurs a drop (blue curve) at an increased frequency (HB-HF) due to a shorter sampling duration compared to the settling time in the auto-zeroing phase (see Fig. 4.3(b)). Additionally, in both HB cases, we observe a decrease in SNDR_d with E_{op1} , achieved by reducing B_{ADC} as quantization noise increases. On the other hand, in the LB operating conditions with lower signal current, i.e., LB-LF (green curve) and LB-HF (red curve), we obtain a lower SNDR_d and E_{op1} compared to the HB case. As compared to the HB case, the SNDR_d remains constant as B_{ADC} changes. This trend remains consistent across ADC columns with differences in maximum SNDR_d levels arising from the spatial mismatch.

Under each of the above operating conditions, we obtain the input-output query pairs $(\mathbf{x}, \hat{\mathbf{y}})$ from the MRAM-based IMC prototype. For the basis vector attack, we follow the same setup described in Section 5.3.1, where the number of input vectors $M = N \times N_{\text{rep}}$. However, for the LS and SGD attacks, we obtain $(\mathbf{x}, \hat{\mathbf{y}})$ for multiple (10) datasets, each comprising $M = 2000$ input vectors with i.i.d. $x \in \{\pm 1\} \sim \text{Be}(0.5)$. The retrieved

vectors $\hat{\mathbf{w}}$ are averaged across the ten datasets and then quantized to ± 1 .

5.5.2 Bank-level Efficacy

Figure 5.9 shows the BER vs. number of input vectors M for BV, LS, and SGD attack at HB-LF and LB-LF operating conditions. Similar to the previous BER trends (Fig. 5.4), the LS and SGD attacks achieve a lower BER compared to the BV attack, as they incorporate both input and output statistics, whereas BV relies solely on outputs. The final BER for the BV attack increases significantly from $\approx 20\%$ (Fig. 5.9(a,b)) to 70-80% (Fig. 5.9(c,d)) as we switch from the HB to the LB regime, indicating an increased number of mispredictions. Although the LS and SGD attacks still perform better than the BV attack, their efficacy degrades in the LB operating condition, resulting in a higher final BER than in the HB condition. This increase in BER with a drop in SNDR_d indicates enhanced security against MEAs at lower compute accuracy.

Additionally, in the case of the ADC column with lowest SNDR_d (worst-case) under HB operating condition (Fig. 5.9(b)), the non-linear SGD attack achieves a minimum BER of 1.58% in 821 measurements, which is $2\times$ lower in terms of both the BER (3.16%) and the number of measurements (1605) the linear LS attack. However, this situation is reversed in the LB-LF case for the worst ADC column (Fig. 5.9(d)), where the linear LS attack achieves a final BER of 21.85% which is $2\times$ lower compared to the BER achieved by the non-linear SGD attack, in 2000 measurements. We conjecture that this reversal is because of the SGD attack converging to a local minimum due to its use of a non-linear model (5.10).

5.5.3 Network-level Efficacy

Figure 5.10 illustrates the energy-accuracy-security trade-offs at the network-level by evaluating the maximum CIFAR-10 accuracy obtained with the retrieved weights $\hat{\mathbf{w}}$ of the last layer of ResNet-20 under different operating conditions. The accuracy trends across attacks observed in Fig. 5.6 are observed in Fig. 5.10 at the highest (avg. SNDR_d , E_{op1}) in the HB-LF condition. The BV attack performs the worst, achieving a maximum accuracy of

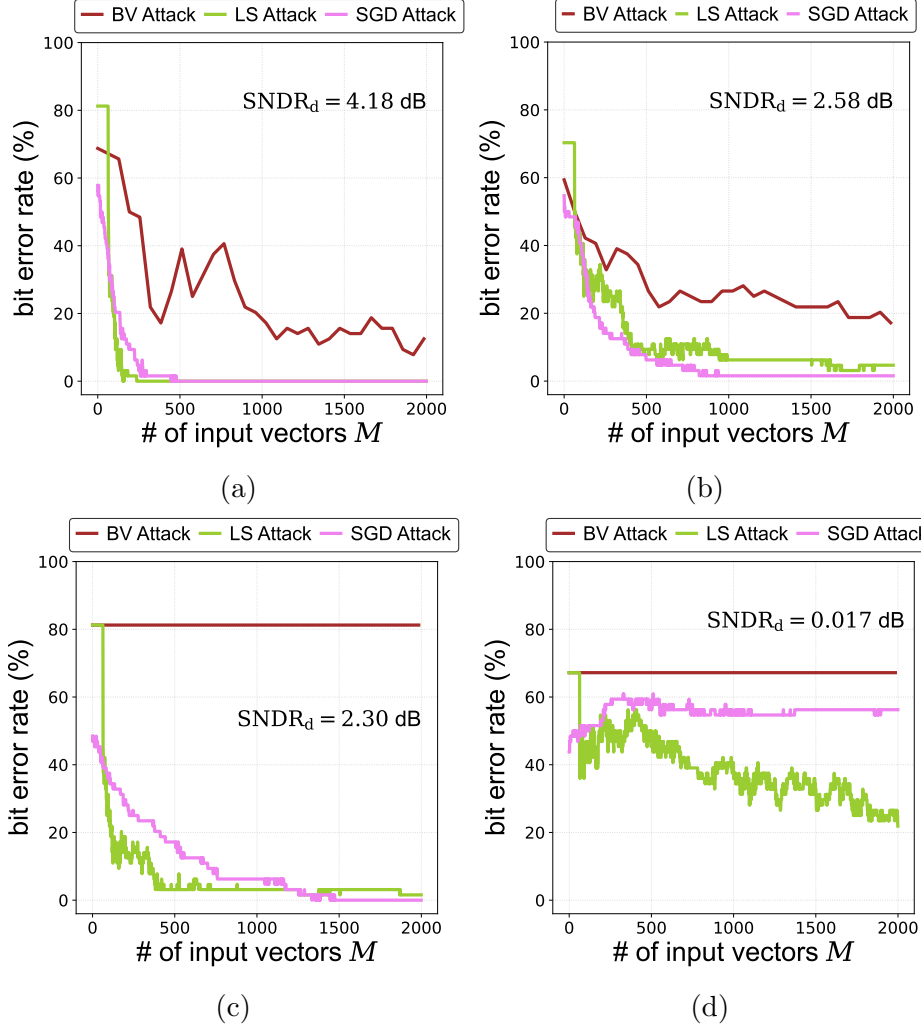


Figure 5.9: Comparing the bank-level efficacy of proposed attacks at $B_{\text{ADC}} = 6$ for the ADC column with the highest SNDR_d under: (a) HB-LF, and (c) LB-LF scenarios; for ADC column with the lowest SNDR_d under: (b) HB-LF, and (d) LB-LF scenarios.

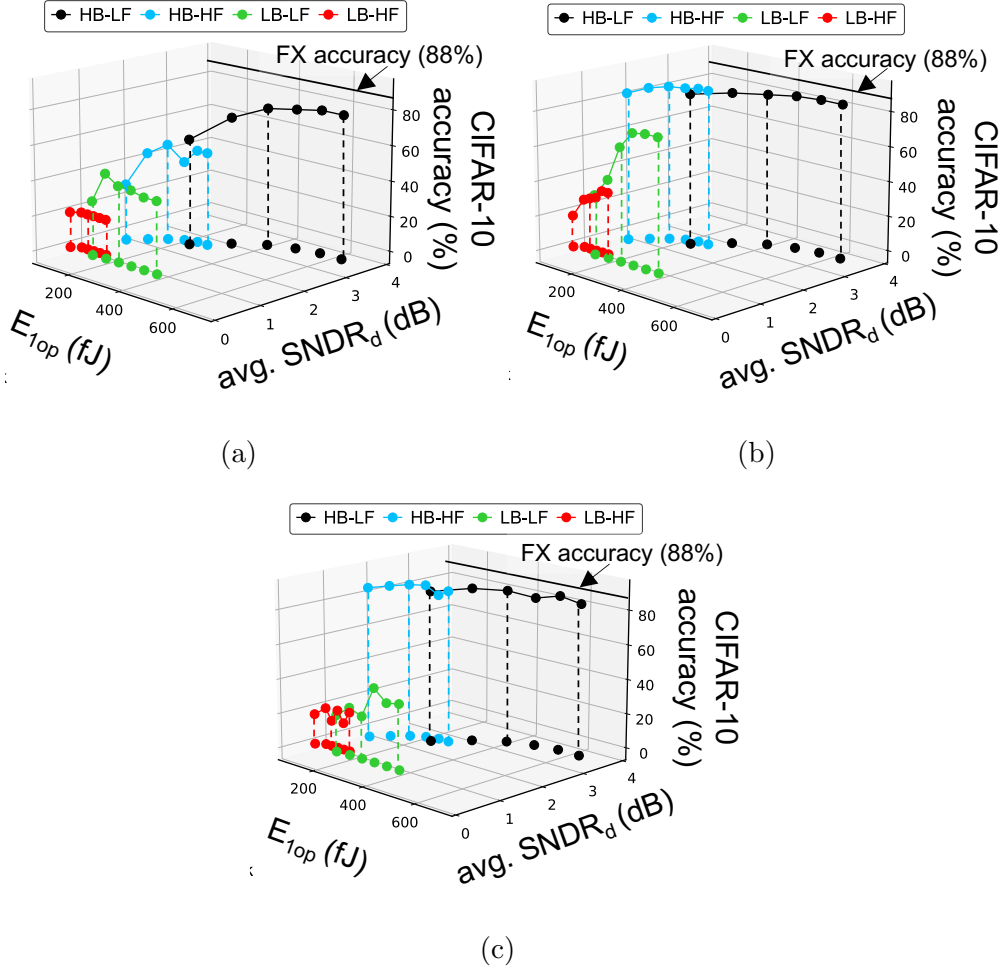


Figure 5.10: Energy-accuracy-security trade-offs obtained by sweeping $B_{ADC} \in \{1, \dots, 6\}$ and using the retrieved weights of the last layer of ResNet-20 on CIFAR-10 for: (a) basis vector, (b) least square, and (c) SGD attack.

80.5%, followed by the LS attack at 87.34%, while the SGD attack achieves the highest accuracy of 87.88%, coming within 0.1% of the FX baseline.

The SGD attack achieves the highest inference accuracy in the HB regime at both frequencies as it achieves the lowest BER (Fig. 5.9) for all ADC columns. In the LB regime, the linear LS model performs best at both frequencies, attaining the highest accuracy of 75.82% at LB-LF and 35.49% at LB-HF. This is consistent with the BER trends observed in Fig. 5.9, where the SGD attack is the strongest in the high-SNDR HB regime, while the LS attack is the strongest in the low-SNDR LB regime. However, all attacks fail to achieve a high inference accuracy at low bias, signifying the security against MEAs.

5.6 Conclusion

In this work, we proposed a statistical framework to investigate security vulnerabilities of eNVM-based IMCs and constructed three distinct MEAs under a strong attack (*Active 2* in Table 5.1) scenario. The results in Figs. 5.4, 5.5, and 5.6 obtained by applying the MEAs to attack an MRAM-based IMC prototype IC show that they are highly effective in compromising security, achieving a network prediction accuracy of within 0.1% of a fixed-point baseline in the high-SNDR operating conditions. However, reducing the bias current—and thereby the compute SNDR and E_{op1} —enhances the security of MRAM-based IMCs against MEAs (Fig. 5.10). This indicates that resistive IMCs operating at low compute SNDR and high energy efficiency have the potential of achieving increased security. Furthermore, employing noise-aware training (NAT) [23, 28, 24, 19] enables resistive IMCs with low bank-level SNDR to achieve high network-level accuracy, at increased security and high energy efficiency. Therefore, we conclude that it is possible for resistive IMCs to jointly achieve energy efficiency, inference accuracy, and enhanced security against MEAs, by operating at low compute SNDR (for energy efficiency and enhanced security) while employing NAT (for improved inference accuracy).

CHAPTER 6

CONCLUSION AND FUTURE WORK

Resistive IMCs have gained popularity for their non-volatility and high areal density, offering a promising opportunity to design DNN accelerators with low data movement costs and support duty-cycled operations. However, achieving high accuracy with resistive IMCs while maintaining high energy and area efficiency is challenging. This difficulty arises from the need for sensitive read-out circuitry to detect small changes in resistances, leading to increased energy and area overhead. The presence of analog non-idealities exacerbates this challenge, further degrading accuracy. This dissertation aims to develop a comprehensive understanding of the limits on the computational accuracy of resistive IMCs, which is essential for understanding the operation of a single-bank design and enabling scalability from single-bank macro-level designs to multi-bank system architectures.

6.1 Summary of Contributions

First, we propose a new benchmarking methodology for IMCs that encompasses a compositional view of IMCs, a comprehensive set of benchmarking metrics, and a systematic strategy for analyzing data and metrics reported in publications. When applied to a database of over 70 IC designs, this methodology reveals that resistive IMCs fall behind SRAM-based IMCs and digital accelerators in both energy efficiency and compute density. This finding is particularly surprising, given that resistive IMCs are often praised for their high memory density.

Next, to address the performance gap in resistive IMCs, we propose a behavioral model to analyze their energy-accuracy trade-off, an area not well understood in the current literature. Our model is used to examine the dependence of bank-level compute accuracy, measured in terms of signal-

to-noise-plus-distortion ratio (SNDR), on device, circuit, and architecture parameters for MRAM-, ReRAM-, and FeFET-based resistive IMCs. The analysis reveals that the maximum compute SNDR is limited by pre-ADC analog non-idealities, and increasing the conductance contrast does not improve SNDR beyond a range of 12-15.

We employ this behavioral model to propose two accuracy-boosting techniques: one at the circuit level, which reduces mismatch across ADC columns, and another at the algorithmic level, which mitigates non-linearity in the dot-product output. These compensation techniques demonstrate a 2.3-to-6 dB SNDR boost for an MRAM-based IMC macro in 22 nm FD-SOI technology. This improvement can be traded off to achieve a $5\times$ reduction in energy per 1b operation, with an SEC energy overhead of only 0.8%. The SEC-enabled neural network (NN) accuracy for CIFAR-10 over ResNet-20 improves significantly, from 74.8% to 82.0%, without the need for on-chip training.

Lastly, we investigate the security vulnerabilities of resistive IMCs and find that contrary to expectations, the intrinsic analog noise does not provide robustness against model extraction attacks. Statistical attacks aimed at retrieving the weights stored in memory from the input-output responses of the IMCs achieve nearly perfect retrieval. These attacks result in network-level accuracy within 0.1% of the digital fixed-point baseline using the retrieved weights, demonstrating a significant security risk.

6.2 Future Prospects

Despite much progress in the domain of IMCs, including both eNVM- and SRAM-based IMCs, the field remains rich with potential and numerous opportunities for innovation. Exploiting these opportunities will require collaborations between researchers in diverse areas. Future opportunities for IMC research include [collaboration topics are indicated in square brackets]:

1. Fully understanding and quantifying the fundamental efficiency vs. accuracy trade-offs in various IMCs (see [146, 185, 149]). This includes minimizing ADC energy by tailoring its SNDR to that of the column computation (see [140, 144, 138]). [circuits, statistical analysis]
2. Developing algorithmic approaches such as statistical error compensa-

tion (SEC) [98, 186, 58] to enhance the accuracy of IMCs beyond what is possible via purely circuit optimization. [circuits, communications]

3. Designing eNVM-based IMCs on emerging devices such as ReRAM, 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 [187]) 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 (> 12 -b) computation is required, e.g., signal processing, communications, scientific computing, and others. [applications, algorithms]
9. Explore IMCs for secure hardware design, i.e., *defense through design*, e.g., using homomorphic encryption [188] and differential privacy [189] methods. [security, circuits]
10. Exploring the energy-accuracy-security trade-off in IMCs, particularly under various attack scenarios, and incorporating different eNVM devices such as ReRAM, FeFETs, as well as SRAM-based IMCs. [security, circuits, statistical analysis]

This dissertation provides an in-depth exploration of the challenges in achieving high bank-level accuracy with resistive IMCs, paving the way for

developing accuracy-boosting techniques at both the circuit and algorithm levels. Additionally, it broadens the potential applications of resistive IMCs in security contexts. Our work highlights the full-stack nature of IMCs, presenting numerous opportunities for collaboration among device researchers, analog/mixed-signal designers, architects, and system and algorithm designers to develop energy-efficient DNN accelerators and beyond.

REFERENCES

- [1] F. Hamzaoglu, “Basics of Nonvolatile Memories: MRAM, RRAM and PRAM,” *ISSCC Tutorial*, 2020.
- [2] C. Calligaro and U. Gatti, *Rad-hard semiconductor memories*. River Publishers, 2022.
- [3] “ReThink Data Storage with 3D ReRAM.” [Online]. Available: <https://www.crossbar-inc.com/products/high-density-memory/>
- [4] “ReRAM IP for Your Embedded Design.” [Online]. Available: <https://www.weebit-nano.com/technology/embedded-reram-ip/>
- [5] O. Mutlu, S. Ghose, J. Gómez-Luna, and R. Ausavarungnirun, “A modern primer on processing in memory,” in *Emerging Computing: From Devices to Systems: Looking Beyond Moore and Von Neumann*. Springer, 2022, pp. 171–243.
- [6] H. Lee, P. Chen, T. Wu, Y. Chen, C. Wang, P. Tzeng, C. Lin, F. Chen, C. Lien, and M.-J. Tsai, “Low power and high speed bipolar switching with a thin reactive Ti buffer layer in robust HfO₂ based RRAM,” in *IEEE International Electron Devices Meeting*, 2008, pp. 1–4.
- [7] Y. Hayakawa, A. Himeno, R. Yasuhara, W. Boullart, E. Vecchio, T. Vandeweyer, T. Witters, D. Crotti, M. Jurczak, S. Fujii et al., “Highly reliable TaO x ReRAM with centralized filament for 28-nm embedded application,” in *2015 Symposium on VLSI Technology (VLSI Technology)*. IEEE, 2015, pp. T14–T15.
- [8] M. Kang, M.-S. Keel, N. R. Shanbhag, S. Eilert, and K. Curewitz, “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)*, 2014, pp. 8326–8330.
- [9] Y. Zha and J. Li, “Reconfigurable in-memory computing with resistive memory crossbar,” in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2016, pp. 1–8.

- [10] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, “ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars,” *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 14–26, 2016.
- [11] Y. Chen, T. Luo, S. Liu, S. Zhang, L. He, J. Wang, L. Li, T. Chen, Z. Xu, N. Sun et al., “Dadiannao: A machine-learning supercomputer,” in *47th Annual IEEE/ACM International Symposium on Microarchitecture*, 2014, pp. 609–622.
- [12] A. Voulodimos, N. Doulamis, A. Doulamis, E. Protopapadakis et al., “Deep learning for computer vision: A brief review,” *Computational intelligence and neuroscience*, vol. 2018, 2018.
- [13] D. W. Otter, J. R. Medina, and J. K. Kalita, “A survey of the usages of deep learning for natural language processing,” *IEEE transactions on neural networks and learning systems*, vol. 32, no. 2, pp. 604–624, 2020.
- [14] H. Sak, A. Senior, K. Rao, O. Irsoy, A. Graves, F. Beaufays, and J. Schalkwyk, “Learning acoustic frame labeling for speech recognition with recurrent neural networks,” in *IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 2015, pp. 4280–4284.
- [15] R. Mochida, K. Kouno, Y. Hayata, M. Nakayama, T. Ono, H. Suwa, R. Yasuhara, K. Katayama, T. Mikawa, and Y. Gohou, “A 4M synapses integrated analog ReRAM based 66.5 TOPS/W neural-network processor with cell current controlled writing and flexible network architecture,” in *2018 IEEE Symposium on VLSI Technology*. IEEE, 2018, pp. 175–176.
- [16] B. Chen, F. Cai, J. Zhou, W. Ma, P. Sheridan, and W. D. Lu, “Efficient in-memory computing architecture based on crossbar arrays,” in *IEEE International Electron Devices Meeting (IEDM)*, 2015, pp. 17–5.
- [17] B. Zimmer, R. Venkatesan, Y. S. Shao, J. Clemons, M. Fojtik, N. Jiang, B. Keller, A. Klinefelter, N. Pinckney, P. Raina 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,” in *IEEE Symposium on VLSI Circuits*, 2019, pp. C300–C301.
- [18] H. Valavi, P. J. Ramadge, E. Nestler, and N. Verma, “A mixed-signal binarized convolutional-neural-network accelerator integrating dense weight storage and multiplication for reduced data movement,” in *IEEE Symposium on VLSI Circuits*, 2018, pp. 141–142.

- [19] R. Khaddam-Aljameh, M. Stanisavljevic, J. F. Mas, G. Karunaratne, M. Braendli, F. Liu, A. Singh, S. M. Müller, U. Egger, A. Petropoulos 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,” in *IEEE Symposium on VLSI Circuits*, 2021, pp. 1–2.
- [20] J. M. Correll, L. Jie, S. Song, S. Lee, J. Zhu, W. Tang, L. Wormald, J. Erhardt, N. Breil, R. Quon et al., “An 8-bit 20.7 TOPS/W multi-level cell ReRAM-based compute engine,” in *IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, 2022, pp. 264–265.
- [21] J.-S. Park, C. Park, S. Kwon, H.-S. Kim, T. Jeon, Y. Kang, H. Lee, D. Lee, J. Kim, Y. Lee et al., “A multi-mode 8K-MAC HW-utilization-aware neural processing unit with a unified multi-precision datapath in 4nm flagship mobile SoC,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 246–248.
- [22] I. A. Papistas, S. Cosemans, B. Rooseleer, J. Doevenspeck, M.-H. Na, A. Mallik, P. Debacker, and D. Verkest, “A 22 nm, 1540 TOP/s/W, 12.1 TOP/s/mm² in-memory analog matrix-vector-multiplier for DNN acceleration,” in *IEEE Custom Integrated Circuits Conference (CICC)*, 2021, pp. 1–2.
- [23] P. Deaville, B. Zhang, L.-Y. Chen, and N. Verma, “A Maximally Row-Parallel MRAM In-Memory-Computing Macro Addressing Readout Circuit Sensitivity and Area,” in *IEEE 47th European Solid State Circuits Conference (ESSCIRC)*, 2021, pp. 75–78.
- [24] P. Deaville, B. Zhang, and N. Verma, “A 22nm 128-kb MRAM Row/column-parallel in-memory computing macro with memory-resistance boosting and multi-column ADC readout,” in *IEEE Symposium on VLSI Circuits*, 2022.
- [25] P. Deaville, B. Zhang, and N. Verma, “A 256-kb Fully Row/Column-parallel 22nm MRAM In-Memory-Computing Macro with Differential Readout for Robust Parallelization and Scale-up,” in *IEEE 49th European Solid State Circuits Conference (ESSCIRC)*, 2023, pp. 21–24.
- [26] S. Yin, X. Sun, S. Yu, and J.-s. Seo, “High-throughput in-memory computing for binary deep neural networks with monolithically integrated RRAM and 90-nm CMOS,” *IEEE Transactions on Electron Devices*, vol. 67, no. 10, pp. 4185–4192, 2020.

- [27] H.-H. Hsu, T.-H. Wen, W.-H. Huang, W.-S. Khwa, Y.-C. Lo, C.-J. Jhang, Y.-H. Chin, Y.-C. Chen, C.-C. Lo, R.-S. Liu et al., “A Non-volatile AI-Edge Processor With SLC–MLC Hybrid ReRAM Compute-in-Memory Macro Using Current–Voltage-Hybrid Readout Scheme,” *IEEE Journal of Solid-State Circuits*, 2023.
- [28] P. Deaville, B. Zhang, and N. Verma, “A Fully Row/Column-Parallel In-Memory Computing Macro in Foundry MRAM With Differential Readout for Noise Rejection,” *IEEE Journal of Solid-State Circuits*, 2024.
- [29] V. Joshi, M. Le Gallo, S. Haefeli, I. Boybat, S. R. Nandakumar, C. Piveteau, M. Dazzi, B. Rajendran, A. Sebastian, and E. Eleftheriou, “Accurate deep neural network inference using computational phase-change memory,” *Nature communications*, vol. 11, no. 1, pp. 1–13, 2020.
- [30] C. Matsui, K. Toprasertpong, S. Takagi, and K. Takeuchi, “Energy-efficient reliable HZO FeFET computation-in-memory with local multiply & global accumulate array for source-follower & charge-sharing voltage sensing,” in *IEEE Symposium on VLSI Technology*, 2021, pp. 1–2.
- [31] C.-X. Xue, W.-H. Chen, J.-S. Liu, J.-F. Li, W.-Y. Lin, W.-E. Lin, J.-H. Wang, W.-C. Wei, T.-W. Chang, T.-C. Chang et al., “A 1Mb multibit ReRAM computing-in-memory macro with 14.6 ns parallel MAC computing time for CNN based AI edge processors,” in *2019 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 2019, pp. 388–390.
- [32] T. Na, S. H. Kang, and S.-O. Jung, “STT-MRAM sensing: a review,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2020.
- [33] W. Wan, R. Kubendran, C. Schaefer, S. B. Eryilmaz, W. Zhang, D. Wu, S. Deiss, P. Raina, H. Qian, B. Gao et al., “A compute-in-memory chip based on resistive random-access memory,” *Nature*, vol. 608, no. 7923, pp. 504–512, 2022.
- [34] T. Soliman, F. Müller, T. Kirchner, T. Hoffmann, H. Ganem, E. Karimov, T. Ali, M. Lederer, C. Sudarshan, T. Kämpfe et al., “Ultra-low power flexible precision FeFET based analog in-memory computing,” in *IEEE International Electron Devices Meeting (IEDM)*, 2020, pp. 29–2.
- [35] K. Stern, N. Wainstein, Y. Keller, C. M. Neumann, E. Pop, S. Kvatinsky, and E. Yalon, “Uncovering phase change memory energy limits by sub-nanosecond probing of power dissipation dynamics,” *Advanced Electronic Materials*, vol. 7, no. 8, p. 2100217, 2021.

- [36] C.-X. Xue, W.-H. Chen, J.-S. Liu, J.-F. Li, W.-Y. Lin, W.-E. Lin, J.-H. Wang, W.-C. Wei, T.-Y. Huang, T.-W. Chang et al., “Embedded 1-Mb ReRAM-based computing-in-memory macro with multibit input and weight for CNN-based AI edge processors,” *IEEE Journal of Solid-State Circuits*, vol. 55, no. 1, pp. 203–215, 2019.
- [37] C.-X. Xue, T.-Y. Huang, J.-S. Liu, T.-W. Chang, H.-Y. Kao, J.-H. Wang, T.-W. Liu, S.-Y. Wei, S.-P. Huang, W.-C. Wei et al., “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)*, 2020, pp. 244–246.
- [38] C.-X. Xue, J.-M. Hung, H.-Y. Kao, Y.-H. Huang, S.-P. Huang, F.-C. Chang, P. Chen, T.-W. Liu, C.-J. Jhang, C.-I. Su et al., “A 22nm 4Mb 8b-precision ReRAM computing-in-memory macro with 11.91 to 195.7 TOPS/W for tiny AI edge devices,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 245–247.
- [39] P. Deaville, B. Zhang, and N. Verma, “A 256-kb Fully Row/Column-parallel 22nm MRAM In-Memory-Computing Macro with Differential Readout for Robust Parallelization and Scale-up,” in *IEEE European Solid State Circuits Conference (ESSCIRC)*, 2023, pp. 75–78.
- [40] J.-H. Yoon, M. Chang, W.-S. Khwa, Y.-D. Chih, M.-F. Chang, and A. Raychowdhury, “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,” *IEEE Journal of Solid-State Circuits*, vol. 57, no. 1, pp. 68–79, 2021.
- [41] F. Cai, J. M. Correll, S. H. Lee, Y. Lim, V. Bothra, Z. Zhang, M. P. Flynn, and W. D. Lu, “A fully integrated reprogrammable memristor–CMOS system for efficient multiply–accumulate operations,” *Nature Electronics*, vol. 2, no. 7, pp. 290–299, 2019.
- [42] Q. Liu, B. Gao, P. Yao, D. Wu, J. Chen, Y. Pang, W. Zhang, Y. Liao, C.-X. Xue, W.-H. Chen et al., “A fully integrated analog ReRAM based 78.4 TOPS/W compute-in-memory chip with fully parallel MAC computing,” in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*. IEEE, 2020, pp. 500–502.
- [43] J.-M. Hung, Y.-H. Huang, S.-P. Huang, F.-C. Chang, T.-H. Wen, C.-I. Su, W.-S. Khwa, C.-C. Lo, R.-S. Liu, C.-C. Hsieh et al., “An 8-Mb DC-current-free binary-to-8b precision ReRAM nonvolatile computing-in-memory macro using time-space-readout with 1286.4-21.6 TOPS/W for edge-AI devices,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 1–3.

- [44] P. Chi, S. Li, C. Xu, T. Zhang, J. Zhao, Y. Liu, Y. Wang, and Y. Xie, “PRIME: A novel processing-in-memory architecture for neural network computation in ReRAM-based main memory,” *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 27–39, 2016.
- [45] S. Jain and A. Raghunathan, “Cxdnn: Hardware-software compensation methods for deep neural networks on resistive crossbar systems,” *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 6, pp. 1–23, 2019.
- [46] Z. He, J. Lin, R. Ewetz, J.-S. Yuan, and D. Fan, “Noise injection adaption: End-to-end ReRAM crossbar non-ideal effect adaption for neural network mapping,” in *Proceedings of the 56th Annual Design Automation Conference 2019*, 2019, pp. 1–6.
- [47] Y. Long, X. She, and S. Mukhopadhyay, “Design of reliable DNN accelerator with un-reliable ReRAM,” in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2019, pp. 1769–1774.
- [48] S. Jain, A. Sengupta, K. Roy, and A. Raghunathan, “RxNN: A framework for evaluating deep neural networks on resistive crossbars,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 2, pp. 326–338, 2020.
- [49] I. Chakraborty, M. Ali, A. Ankit, S. Jain, S. Roy, S. Sridharan, A. Agrawal, A. Raghunathan, and K. Roy, “Resistive crossbars as approximate hardware building blocks for machine learning: Opportunities and challenges,” *Proceedings of the IEEE*, vol. 108, no. 12, pp. 2276–2310, 2020.
- [50] I. Chakraborty, M. F. Ali, D. E. Kim, A. Ankit, and K. Roy, “Geniex: A generalized approach to emulating non-ideality in memristive xbars using neural networks,” in *57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.
- [51] S. Roy, S. Sridharan, S. Jain, and A. Raghunathan, “TxSim: Modeling training of deep neural networks on resistive crossbar systems,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 4, pp. 730–738, 2021.
- [52] F. M. Bayat, M. Prezioso, B. Chakrabarti, H. Nili, I. Kataeva, and D. Strukov, “Implementation of multilayer perceptron network with highly uniform passive memristive crossbar circuits,” *Nature communications*, vol. 9, no. 1, pp. 1–7, 2018.
- [53] M. Le Gallo, A. Sebastian, R. Mathis, M. Manica, H. Giefers, T. Tuma, C. Bekas, A. Curioni, and E. Eleftheriou, “Mixed-precision in-memory computing,” *Nature Electronics*, vol. 1, no. 4, pp. 246–253, 2018.

- [54] W. Wan, R. Kubendran, B. Gao, S. Joshi, P. Raina, H. Wu, G. Cauwenberghs, and H.-S. P. Wong, “A voltage-mode sensing scheme with differential-row weight mapping for energy-efficient RRAM-based in-memory computing,” in *2020 IEEE Symposium on VLSI Technology*. IEEE, 2020, pp. 1–2.
- [55] I. Boybat, M. Le Gallo, S. Nandakumar, T. Moraitis, T. Parnell, T. Tuma, B. Rajendran, Y. Leblebici, A. Sebastian, and E. Eleftheriou, “Neuromorphic computing with multi-memristive synapses,” *Nature communications*, vol. 9, no. 1, pp. 1–12, 2018.
- [56] M. Le Gallo, R. Khaddam-Aljameh, M. Stanisavljevic, A. Vasilopoulos, B. Kersting, M. Dazzi, G. Karunaratne, M. Brändli, A. Singh, S. M. Mueller et al., “A 64-core mixed-signal in-memory compute chip based on phase-change memory for deep neural network inference,” *Nature Electronics*, vol. 6, no. 9, pp. 680–693, 2023.
- [57] W.-H. Chen, K.-X. Li, W.-Y. Lin, K.-H. Hsu, P.-Y. Li, C.-H. Yang, C.-X. Xue, E.-Y. Yang, Y.-K. Chen, Y.-S. Chang et al., “A 65nm 1Mb nonvolatile computing-in-memory ReRAM macro with sub-16ns multiply-and-accumulate for binary DNN AI edge processors,” in *2018 IEEE International Solid-State Circuits Conference (ISSCC)*. IEEE, 2018, pp. 494–496.
- [58] S. Roy, H.-M. Ou, M. Gamal, P. Deaville, B. Zhang, N. Verma, P. Hanumolu, and N. Shanbhag, “SNR-boosted 22nm MRAM-based in-memory computing macro using statistical error compensation,” in *IEEE European Solid State Circuits Conference (ESSCIRC)*, 2023, pp. 75–78.
- [59] W. Li, S. Huang, X. Sun, H. Jiang, and S. Yu, “Secure-RRAM: A 40nm 16kb compute-in-memory macro with reconfigurability, sparsity control, and embedded security,” in *IEEE Custom Integrated Circuits Conference (CICC)*, 2021, pp. 1–2.
- [60] W.-S. Khwa, Y.-C. Chiu, C.-J. Jhang, S.-P. Huang, C.-Y. Lee, T.-H. Wen, F.-C. Chang, S.-M. Yu, T.-Y. Lee, and M.-F. Chang, “A 40-nm, 2M-Cell, 8b-Precision, hybrid SLC-MLC PCM computing-in-Memory macro with 20.5-65.0 TOPS/W for tiny-AI edge devices,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 1–3.
- [61] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell et al., “Language models are few-shot learners,” *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

- [62] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in neural information processing systems*, vol. 25, 2012.
- [63] J. Von Neumann, “First draft of a report on the EDVAC,” *Tech. Rep., Univ. Pennsylvania, Philadelphia, PA, USA*, Jun. 1945.
- [64] M. Horowitz, “1.1 computing’s energy problem (and what we can do about it),” in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*. IEEE, 2014, pp. 10–14.
- [65] D. Elliott, W. Snelgrove, and M. Stumm, “Computational RAM: A Memory-simd Hybrid And Its Application To DSP,” in *1992 Proceedings of the IEEE Custom Integrated Circuits Conference*, 1992, pp. 30.6.1–30.6.4.
- [66] C. Kozyrakis, S. Perissakis, D. Patterson, T. Anderson, K. Asanovic, N. Cardwell, R. Fromm, J. Golbus, B. Gribstad, K. Keeton, R. Thomas, N. Treuhaft, and K. Yelick, “Scalable processors in the billion-transistor era: Iram,” *Computer*, vol. 30, no. 9, pp. 75–78, 1997.
- [67] K. Mai, T. Paaske, N. Jayasena, R. Ho, W. Dally, and M. Horowitz, “Smart memories: a modular reconfigurable architecture,” in *Proceedings of 27th International Symposium on Computer Architecture (IEEE Cat. No.RS00201)*, 2000, pp. 161–171.
- [68] R. Genov and G. Cauwenberghs, “Kerneltron: support vector ”machine” in silicon,” *IEEE Transactions on Neural Networks*, vol. 14, no. 5, pp. 1426–1434, 2003.
- [69] M. Kang, S. K. Gonugondla, A. Patil, and N. R. Shanbhag, “A multi-functional in-memory inference processor using a standard 6T SRAM array,” *IEEE Journal of Solid-State Circuits*, vol. 53, no. 2, pp. 642–655, 2018.
- [70] M. Kang, S. K. Gonugondla, S. Lim, and N. R. Shanbhag, “A 19.4-nJ/decision, 364-K decisions/s, in-memory random forest multi-class inference accelerator,” *IEEE Journal of Solid-State Circuits*, vol. 53, no. 7, pp. 2126–2135, July 2018.
- [71] S. K. Gonugondla, M. Kang, and N. R. Shanbhag, “A variation-tolerant in-memory machine learning classifier via on-chip training,” *IEEE Journal of Solid-State Circuits*, vol. 53, no. 11, pp. 3163–3173, November 2018.
- [72] H. Dbouk, S. K. Gonugondla, C. Sakr, and N. R. Shanbhag, “A 0.44- μ J/dec, 39.9- μ s/dec, recurrent attention In-Memory Processor for Keyword Spotting,” *IEEE Journal of Solid-State Circuits*, vol. 56, no. 7, pp. 2234–2244, 2021.

- [73] G. W. Burr, S. Lim, B. Murmann, R. Venkatesan, and M. Verhelst, “Fair and comprehensive benchmarking of machine learning processing chips,” *IEEE Design & Test*, 2021.
- [74] N. R. Shanbhag and S. K. Roy, “Benchmarking in-memory computing architectures,” *IEEE Open Journal of the Solid-State Circuits Society*, vol. 2, pp. 288–300, 2022.
- [75] J.-W. Su, Y.-C. Chou, R. Liu, T.-W. Liu, P.-J. Lu, P.-C. Wu, Y.-L. Chung, L.-Y. Hung, J.-S. Ren, T. Pan et al., “16.3 a 28nm 384kb 6T-SRAM computation-in-memory macro with 8b precision for AI edge chips,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 250–252.
- [76] J. Yue, X. Feng, Y. He, Y. Huang, Y. Wang, Z. Yuan, M. Zhan, J. Liu, J.-W. Su, Y.-L. Chung et al., “A 2.75-to-75.9 TOPS/W computing-in-memory NN processor supporting set-associate block-wise zero skipping and ping-pong CIM with simultaneous computation and weight updating,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 238–240.
- [77] H. Jia, M. Ozatay, Y. Tang, H. Valavi, R. Pathak, J. Lee, and N. Verma, “A programmable neural-network inference accelerator based on scalable in-memory computing,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 236–238.
- [78] R. Guo, Z. Yue, X. Si, T. Hu, H. Li, L. Tang, Y. Wang, L. Liu, M.-F. Chang, Q. Li et al., “A 5.99-to-691.1 TOPS/W tensor-train in-memory-computing processor using bit-level-sparsity-based optimization and variable-precision quantization,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 242–244.
- [79] Y.-D. Chih, P.-H. Lee, H. Fujiwara, Y.-C. Shih, C.-F. Lee, R. Naous, Y.-L. Chen, C.-P. Lo, C.-H. Lu, H. Mori et al., “An 89 TOPS/W and 16.3 TOPS/mm² all-digital SRAM-based full-precision compute-in memory macro in 22nm for machine-learning edge applications,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 252–254.
- [80] R. A. Rasul and M. S.-W. Chen, “A 128x128 SRAM macro with embedded matrix-vector multiplication exploiting passive gain via MOS capacitor for machine learning application,” in *IEEE Custom Integrated Circuits Conference (CICC)*, 2021, pp. 1–2.
- [81] J. Lee, H. Valavi, Y. Tang, and N. Verma, “Fully row/column-parallel in-memory computing SRAM macro employing capacitor-based mixed-signal computation with 5-b inputs,” in *IEEE Symposium on VLSI Circuits*, 2021, pp. 1–2.

- [82] S. Yin, B. Zhang, M. Kim, J. Saikia, S. Kwon, S. Myung, H. Kim, S. J. Kim, M. Seok, and J.-s. Seo, "PIMCA: A 3.4-Mb programmable in-memory computing accelerator in 28nm for on-chip DNN inference," in *IEEE Symposium on VLSI Technology*, 2021, pp. 1–2.
- [83] R. Guo, H. Li, R. Liu, Z. Zhang, L. Tang, H. Sun, L. Liu, M.-F. Chang, S. Wei, and S. Yin, "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," in *IEEE Symposium on VLSI Technology*, 2021, pp. 1–2.
- [84] J.-W. Su, X. Si, Y.-C. Chou, T.-W. Chang, W.-H. Huang, Y.-N. Tu, R. Liu, P.-J. Lu, T.-W. Liu, J.-H. Wang et al., "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)*, 2020, pp. 240–242.
- [85] Q. Dong, M. E. Sinangil, B. Erbagci, D. Sun, W.-S. Khwa, H.-J. Liao, Y. Wang, and J. Chang, "A 351TOPS/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)*, 2020, pp. 242–244.
- [86] J. Yue, Z. Yuan, X. Feng, Y. He, Z. Zhang, X. Si, R. Liu, M.-F. Chang, X. Li, H. Yang et al., "A 65nm computing-in-memory-based CNN processor with 2.9-to-35.8 TOPS/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)*, 2020, pp. 234–236.
- [87] X. Si, Y.-N. Tu, W.-H. Huang, J.-W. Su, P.-J. Lu, J.-H. Wang, T.-W. Liu, S.-Y. Wu, R. Liu, Y.-C. Chou et al., "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)*, 2020, pp. 246–248.
- [88] C. Yu, T. Yoo, T. T.-H. Kim, K. C. T. Chuan, and B. Kim, "A 16K current-based 8T SRAM compute-in-memory macro with decoupled read/write and 1-5bit column ADC," in *IEEE Custom Integrated Circuits Conference (CICC)*, 2020, pp. 1–4.
- [89] H. Jia, H. Valavi, Y. Tang, J. Zhang, and N. Verma, "A programmable heterogeneous microprocessor based on bit-scalable in-memory computing," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 9, pp. 2609–2621, 2020.

- [90] Z. Jiang, S. Yin, J.-s. Seo, and M. Seok, "C3SRAM: An in-memory-computing SRAM macro based on robust capacitive coupling computing mechanism," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 7, pp. 1888–1897, 2020.
- [91] X. Si, J.-J. Chen, Y.-N. Tu, W.-H. Huang, J.-H. Wang, Y.-C. Chiu, W.-C. Wei, S.-Y. Wu, X. Sun, R. Liu et al., "24.5 A twin-8T SRAM computation-in-memory macro for multiple-bit CNN-based machine learning," in *IEEE International Solid-State Circuits Conference-(ISSCC)*, 2019, pp. 396–398.
- [92] J. Yang, Y. Kong, Z. Wang, Y. Liu, B. Wang, S. Yin, and L. Shi, "Sandwich-RAM: An energy-efficient in-memory BWN architecture with pulse-width modulation," in *IEEE International Solid-State Circuits Conference-(ISSCC)*, 2019, pp. 394–396.
- [93] T. Chen, J. Botimer, T. Chou, and Z. Zhang, "An SRAM-based accelerator for solving partial differential equations," in *IEEE Custom Integrated Circuits Conference (CICC)*, 2019, pp. 1–4.
- [94] R. Guo, Y. Liu, S. Zheng, S.-Y. Wu, P. Ouyang, W.-S. Khwa, X. Chen, J.-J. Chen, X. Li, L. Liu et al., "A 5.1 pJ/neuron 127.3 μ s/inference RNN-based speech recognition processor using 16 computing-in-memory SRAM macros in 65nm CMOS," in *IEEE Symposium on VLSI Circuits*, 2019, pp. C120–C121.
- [95] J. Kim, J. Koo, T. Kim, Y. Kim, H. Kim, S. Yoo, and J.-J. Kim, "Area-efficient and variation-tolerant in-memory BNN computing using 6T SRAM array," in *IEEE Symposium on VLSI Circuits*, 2019, pp. C118–C119.
- [96] S. Okumura, M. Yabuuchi, K. Hijioka, and K. Nose, "A Ternary Based Bit Scalable, 8.80 TOPS/W CNN accelerator with Many-core Processing-in-memory Architecture with 896K synapses/mm²," in *IEEE Symposium on VLSI Circuits*, 2019, pp. C248–C249.
- [97] W.-S. Khwa, J.-J. Chen, J.-F. Li, X. Si, E.-Y. Yang, X. Sun, R. Liu, P.-Y. Chen, Q. Li, S. Yu et al., "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)*, 2018, pp. 496–498.
- [98] S. K. Gonugondla, M. Kang, and N. Shanbhag, "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)*, 2018, pp. 490–492.

- [99] A. Biswas and A. P. Chandrakasan, “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)*, 2018, pp. 488–490.
- [100] Z. Jiang, S. Yin, M. Seok, and J.-s. Seo, “XNOR-SRAM: in-memory computing SRAM macro for binary/ternary deep neural networks,” in *IEEE Symposium on VLSI Technology*, 2018, pp. 173–174.
- [101] P.-C. Wu, J.-W. Su, Y.-L. Chung, L.-Y. Hong, J.-S. Ren, F.-C. Chang, Y. Wu, H.-Y. Chen, C.-H. Lin, H.-M. Hsiao et al., “A 28nm 1Mb time-domain computing-in-memory 6T-SRAM macro with a 6.6 ns latency, 1241GOPS and 37.01 TOPS/W for 8b-MAC operations for edge-AI devices,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 1–3.
- [102] H. Fujiwara, H. Mori, W.-C. Zhao, M.-C. Chuang, R. Naous, C.-K. Chuang, T. Hashizume, D. Sun, C.-F. Lee, K. Akarvardar et al., “A 5-nm 254-TOPS/W 221-TOPS/mm² fully-digital computing-in-memory macro supporting wide-range dynamic-voltage-frequency scaling and simultaneous MAC and write operations,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 1–3.
- [103] B. Yan, J.-L. Hsu, P.-C. Yu, C.-C. Lee, Y. Zhang, W. Yue, G. Mei, Y. Yang, Y. Yang, H. Li et al., “A 1.041-Mb/mm² 27.38-TOPS/W signed-INT8 dynamic-logic-based ADC-less SRAM compute-in-memory macro in 28nm with reconfigurable bitwise operation for AI and embedded applications,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 188–190.
- [104] F. Tu, Y. Wang, Z. Wu, L. Liang, Y. Ding, B. Kim, L. Liu, S. Wei, Y. Xie, and S. Yin, “A 28nm 29.2 TFLOPS/W BF16 and 36.5 TOPS/W INT8 reconfigurable digital CIM processor with unified FP/INT pipeline and bitwise in-memory booth multiplication for cloud deep learning acceleration,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 1–3.
- [105] K. Ueyoshi, I. A. Papistas, P. Houshmand, G. M. Sarda, V. Jain, M. Shi, Q. Zheng, S. Giraldo, P. Vrancx, J. Doevenspeck et al., “DI-ANA: an end-to-end energy-efficient Digital and ANALog hybrid neural network SoC,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 1–3.

- [106] D. Wang, C.-T. Lin, G. K. Chen, P. Knag, R. K. Krishnamurthy, and M. Seok, "DIMC: 2219TOPS/W 2569F2/b digital in-memory computing macro in 28nm based on approximate arithmetic hardware," in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 266–268.
- [107] A. Biswas, H. Sanghvi, M. Mehendale, and G. Preet, "An area-efficient 6T-SRAM based Compute-In-Memory architecture with reconfigurable SAR ADCs for energy-efficient deep neural networks in edge ML applications," in *IEEE Custom Integrated Circuits Conference (CICC)*, 2022, pp. 1–2.
- [108] B. Zhang, J. Saikia, J. Meng, D. Wang, S. Kwon, S. Myung, H. Kim, S. J. Kim, J.-s. Seo, and M. Seok, "A 177 TOPS/W, capacitor-based in-memory computing SRAM macro with stepwise-charging/discharging DACs and sparsity-optimized bitcells for 4-bit deep convolutional neural networks," in *IEEE Custom Integrated Circuits Conference (CICC)*, 2022, pp. 1–2.
- [109] Z. Chen, Q. Jin, Z. Yu, Y. Wang, and K. Yang, "DCT-RAM: A driver-free process-in-memory 8T SRAM macro with multi-bit charge-domain computation and time-domain quantization," in *IEEE Custom Integrated Circuits Conference (CICC)*, 2022, pp. 1–2.
- [110] E. Choi, I. Choi, C. Jeon, G. Yun, D. Yi, S. Ha, I.-J. Chang, and M. Je, "A 133.6 TOPS/W compute-in-memory SRAM macro with fully parallel one-step multi-bit computation," in *IEEE Custom Integrated Circuits Conference (CICC)*, 2022, pp. 1–2.
- [111] J. Heo, J. Kim, W. Han, S. Lim, and J.-Y. Kim, "T-PIM: A 2.21-to-161.08 TOPS/W processing-in-memory accelerator for end-to-end on-device training," in *IEEE Custom Integrated Circuits Conference (CICC)*, 2022, pp. 01–02.
- [112] H. Wang, R. Liu, R. Dorrance, D. Dasalukunte, X. Liu, D. Lake, B. Carlton, and M. Wu, "A 32.2 TOPS/W SRAM compute-in-memory macro employing a linear 8-bit C-2C ladder for charge domain computation in 22nm for edge inference," in *IEEE Symposium on VLSI Circuits*, 2022.
- [113] D. Saito, T. Kobayashi, H. Koga, N. Ronchi, K. Banerjee, Y. Shuto, J. Okuno, K. Konishi, L. Di Piazza, A. Mallik et al., "Analog in-memory computing in FeFET-based 1T1R array for edge AI applications," in *IEEE Symposium on VLSI Technology*, 2021, pp. 1–2.

- [114] L. Fick, S. Skrzyniarz, M. Parikh, M. B. Henry, and D. Fick, “Analog matrix processor for edge AI real-time video analytics,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 260–262.
- [115] S. D. Spetalnick, M. Chang, B. Crafton, W.-S. Khwa, Y.-D. Chih, M.-F. Chang, and A. Raychowdhury, “A 40nm 64kb 26.56 TOPS/W 2.37 Mb/mm² RRAM binary/compute-in-memory macro with 4.23 x improvement in density and > 75% use of sensing dynamic range,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 1–3.
- [116] J. M. Correll, L. Jie, S. Song, S. Lee, J. Zhu, W. Tang, L. Wormald, J. Erhardt, N. Breil, R. Quon, D. Kamalanathan, S. Krishnan, M. Chudzik, M. Zhang, W. D. Lu, and M. P. Flynn, “An 8-bit 20.7 TOPS/W multi-level cell ReRAM-based compute engine,” in *IEEE Symposium on VLSI Circuits*, 2022.
- [117] S. Xie, C. Ni, P. Jain, F. Hamzaoglu, and J. P. Kulkarni, “Gain-Cell CIM: leakage and bitline swing aware 2T1C Gain-Cell eDRAM compute in memory design with bitline precharge DACs and compact schmitt trigger ADCs,” in *IEEE Symposium on VLSI Circuits*, 2022.
- [118] S. Xie, C. Ni, A. Sayal, P. Jain, F. Hamzaoglu, and J. P. Kulkarni, “eDRAM-CIM: compute-in-memory design with reconfigurable embedded-dynamic-memory array realizing adaptive data converters and charge-domain computing,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 248–250.
- [119] Z. Chen, X. Chen, and J. Gu, “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,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 240–242.
- [120] A. Agrawal, S. K. Lee, J. Silberman, M. Ziegler, M. Kang, S. Venkataramani, N. Cao, B. Fleischer, M. Guillorn, M. Cohen et al., “A 7nm 4-core AI chip with 25.6 TFLOPS hybrid FP8 training, 102.4 TOPS INT4 inference and workload-aware throttling,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 144–146.
- [121] J.-S. Park, J.-W. Jang, H. Lee, D. Lee, S. Lee, H. Jung, S. Lee, S. Kwon, K. Jeong, J.-H. Song et al., “A 6k-MAC feature-map-sparsity-aware neural processing unit in 5nm flagship mobile SoC,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 152–154.

- [122] H. Mo, W. Zhu, W. Hu, G. Wang, Q. Li, A. Li, S. Yin, S. Wei, and L. Liu, “A 28nm 12.1 TOPS/W dual-mode CNN processor using effective-weight-based convolution and error-compensation-based prediction,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 146–148.
- [123] J. Park, S. Lee, and D. Jeon, “A 40nm 4.81 TFLOPS/W 8b floating-point training processor for non-sparse neural networks using shared exponent bias and 24-way fused multiply-add tree,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 1–3.
- [124] Z. Tan, S.-H. Tan, J.-H. Lambrechts, Y. Zhang, Y. Wu, and K. Ma, “A 400MHz NPU with 7.8 TOPS 2/W high-performance guaranteed efficiency in 55nm for multi-mode pruning and diverse quantization using pattern-kernel encoding and reconfigurable MAC units,” in *IEEE Custom Integrated Circuits Conference (CICC)*, 2021, pp. 1–2.
- [125] J. Lee, J. Kim, W. Jo, S. Kim, S. Kim, J. Lee, and H.-J. Yoo, “A 13.7 TFLOPS/W floating-point DNN processor using heterogeneous computing architecture with exponent-computing-in-memory,” in *IEEE Symposium on VLSI Circuits*, 2021, pp. 1–2.
- [126] S. Kang, D. Han, J. Lee, D. Im, S. Kim, S. Kim, and H.-J. Yoo, “GANPU: A 135TFLOPS/W multi-DNN training processor for GANs with speculative dual-sparsity exploitation,” in *IEEE International Solid-State Circuits Conference-(ISSCC)*, 2020, pp. 140–142.
- [127] Y. Jiao, L. Han, R. Jin, Y.-J. Su, C. Ho, L. Yin, Y. Li, L. Chen, Z. Chen, L. Liu et al., “A 12nm programmable convolution-efficient neural-processing-unit chip achieving 825TOPS,” in *IEEE International Solid-State Circuits Conference-(ISSCC)*, 2020, pp. 136–140.
- [128] C.-H. Lin, C.-C. Cheng, Y.-M. Tsai, S.-J. Hung, Y.-T. Kuo, P. H. Wang, P.-K. Tsung, J.-Y. Hsu, W.-C. Lai, C.-H. Liu et al., “A 3.4-to-13.3 TOPS/W 3.6 TOPS dual-core deep-learning accelerator for versatile AI applications in 7nm 5G smartphone SoC,” in *IEEE International Solid-State Circuits Conference-(ISSCC)*, 2020, pp. 134–136.
- [129] J. Oh, S. K. Lee, M. Kang, M. Ziegler, J. Silberman, A. Agrawal, S. Venkataramani, B. Fleischer, M. Guillorn, J. Choi et al., “A 3.0 TFLOPS 0.62 V scalable processor core for high compute utilization AI training and inference,” in *IEEE Symposium on VLSI Circuits*, 2020, pp. 1–2.

- [130] J.-H. Kim, J. Lee, J. Lee, H.-J. Yoo, and J.-Y. Kim, “Z-PIM: An energy-efficient sparsity aware processing-in-memory architecture with fully-variable weight precision,” in *IEEE Symposium on VLSI Circuits*, 2020, pp. 1–2.
- [131] J. Wang, X. Wang, C. Eckert, A. Subramaniyan, R. Das, D. Blaauw, and D. Sylvester, “A compute SRAM with bit-serial integer/floating-point operations for programmable in-memory vector acceleration,” in *IEEE International Solid-State Circuits Conference-(ISSCC)*, 2019, pp. 224–226.
- [132] Y. Ju and J. Gu, “A 65nm systolic neural CPU processor for combined deep learning and general-purpose computing with 95% PE utilization, high data locality and enhanced end-to-end performance,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 1–3.
- [133] H. Zhu, B. Jiao, J. Zhang, X. Jia, Y. Wang, T. Guan, S. Wang, D. Niu, H. Zheng, C. Chen et al., “COMB-MCM: computing-on-memory-boundary NN processor with bipolar bitwise sparsity optimization for scalable multi-chiplet-module edge machine learning,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 1–3.
- [134] K. Hirose, J. Yu, K. Ando, Y. Okoshi, Á. L. García-Arias, J. Suzuki, T. Van Chu, K. Kawamura, and M. Motomura, “Hiddenite: 4K-PE hidden network inference 4D-tensor engine exploiting on-chip model construction achieving 34.8-to-16.0 TOPS/W for CIFAR-100 and ImageNet,” in *IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 65, 2022, pp. 1–3.
- [135] A. Gupta, V. Konandur, T. Salam, S. Jain, O. Aiello, P. Croveti, and M. Alioto, “DDPMnet: all-digital pulse density-based DNN architecture with 228 gate equivalents/MAC Unit, 28-TOPS/W and 1.5-TOPS/mm² in 40nm,” in *IEEE Custom Integrated Circuits Conference (CICC)*, 2022, pp. 1–2.
- [136] B. Keller, R. Venkatesan, S. Dai, S. G. Tell, B. Zimmer, W. J. Dally, C. T. Gray, and B. Khailany, “A 17–95.6 TOPS/W deep learning inference accelerator with per-vector scaled 4-bit quantization for transformers in 5nm,” in *IEEE Symposium on VLSI Circuits*, 2022.
- [137] C.-F. Lee, C.-H. Lu, C.-E. Lee, H. Mori, H. Fujiwara, Y.-C. Shih, T.-L. Chou, Y.-D. Chih, T. Yung, and J. Chang, “A 12nm 121-TOPS/W 41.6-TOPS/mm² all digital full precision sram-based compute-in-memory with configurable bit-width for AI edge applications,” in *IEEE Symposium on VLSI Circuits*, 2022.

- [138] B. Murmann, “Mixed-signal computing for deep neural network inference,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 1, pp. 3–13, 2020.
- [139] S. K. Roy, A. Patil, and N. R. Shanbhag, “Fundamental limits on the computational accuracy of resistive crossbar-based in-memory architectures,” in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2022.
- [140] S. K. Gonugondla, C. Sakr, H. Dbouk, and N. R. Shanbhag, “Fundamental limits on the precision of in-memory architectures,” in *Proceedings of the 39th International Conference on Computer-Aided Design*, 2020, pp. 1–9.
- [141] C. Sakr, Y. Kim, and N. Shanbhag, “Analytical guarantees on numerical precision of deep neural networks,” in *International Conference on Machine Learning*. PMLR, 2017, pp. 3007–3016.
- [142] C. Sakr and N. Shanbhag, “An analytical method to determine minimum per-layer precision of deep neural networks,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 1090–1094.
- [143] D. Zhang, J. Yang, D. Ye, and G. Hua, “LQ-Nets: Learned quantization for highly accurate and compact deep neural networks,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 365–382.
- [144] C. Sakr and N. R. Shanbhag, “Signal processing methods to enhance the energy efficiency of in-memory computing architectures,” *IEEE Transactions on Signal Processing*, vol. 69, pp. 6462–6472, 2021.
- [145] S. Lloyd, “Least squares quantization in PCM,” *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.
- [146] S. K. Gonugondla, C. Sakr, H. Dbouk, and N. R. Shanbhag, “Fundamental limits on energy-delay-accuracy of in-memory architectures in inference applications,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2021.
- [147] R. Sehgal and J. P. Kulkarni, “Trends in analog and digital intensive compute-in-SRAM designs,” in *IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2021, pp. 1–4.
- [148] S. K. Cherupally, J. Meng, A. S. Rakin, S. Yin, M. Seok, D. Fan, and J.-s. Seo, “Improving DNN hardware accuracy by in-memory computing noise injection,” *IEEE Design & Test*, vol. 39, no. 4, pp. 71–80, 2021.

- [149] S. K. Roy and N. R. Shanbhag, “Energy-Accuracy Trade-offs for Resistive In-Memory Computing Architectures,” *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, 2024.
- [150] J.-H. Lee, D.-H. Lim, H. Jeong, H. Ma, and L. Shi, “Exploring cycle-to-cycle and device-to-device variation tolerance in mlc storage-based neural network training,” *IEEE Transactions on Electron Devices*, vol. 66, no. 5, pp. 2172–2178, 2019.
- [151] W. P. Zhang and X. Tong, “Noise modeling and analysis of SAR ADCs,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 12, pp. 2922–2930, 2014.
- [152] S. K. Gonugondla, C. Sakr, H. Dbouk, and N. R. Shanbhag, “Fundamental limits on energy-delay-accuracy of in-memory architectures in inference applications,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 10, pp. 3188–3201, 2021.
- [153] M. Spear, J. E. Kim, C. H. Bennett, S. Agarwal, M. J. Marinella, and T. P. Xiao, “The Impact of Analog-to-Digital Converter Architecture and Variability on Analog Neural Network Accuracy,” *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, 2023.
- [154] B. Murmann, “Mixed-signal computing for deep neural network inference,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 1, pp. 3–13, 2020.
- [155] B. Murmann, “ADC Performance Survey 1997-2023,” [Online]. Available: <https://github.com/bmurmann/ADC-survey>.
- [156] B. Murmann, “A/D converter trends: Power dissipation, scaling and digitally assisted architectures,” in *IEEE Custom Integrated Circuits Conference*, 2008, pp. 105–112.
- [157] S. K. Roy, A. Patil, and N. R. Shanbhag, “Fundamental limits on the computational accuracy of resistive crossbar-based in-memory architectures,” in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2022.
- [158] W. Wu, H. Wu, B. Gao, P. Yao, X. Zhang, X. Peng, S. Yu, and H. Qian, “A methodology to improve linearity of analog RRAM for neuromorphic computing,” in *2018 IEEE Symposium on VLSI Technology*. IEEE, 2018, pp. 103–104.
- [159] M. Cheng, L. Xia, Z. Zhu, Y. Cai, Y. Xie, Y. Wang, and H. Yang, “TIME: A training-in-memory architecture for RRAM-based deep neural networks,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 5, pp. 834–847, 2018.

- [160] I. Chakraborty, S. Roy, S. Sridharan, M. Ali, A. Ankit, S. Jain, and A. Raghunathan, “Design Tools for Resistive Crossbar based Machine Learning Accelerators,” in *IEEE 3rd International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2021, pp. 1–4.
- [161] H.-M. Ou and N. R. Shanbhag, “Enhancing the accuracy of resistive in-memory architectures using adaptive signal processing,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.
- [162] X. Zhang, A. A. Ding, and Y. Fei, “Deep-learning model extraction through software-based power side-channel,” in *IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2023.
- [163] C. Gongye, Y. Fei, and T. Wahl, “Reverse-engineering deep neural networks using floating-point timing side-channels,” in *ACM/IEEE Design Automation Conference (DAC)*, 2020.
- [164] S. Potluri and A. Aysu, “Stealing neural network models through the scan chain: A new threat for ML hardware,” in *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2021, pp. 1–8.
- [165] C. Gongye, Y. Luo, X. Xu, and Y. Fei, “Side-Channel-Assisted Reverse-Engineering of Encrypted DNN Hardware Accelerator IP and Attack Surface Exploration,” in *IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2023, pp. 1–1.
- [166] K. Yoshida, T. Kubota, S. Okura, M. Shiozaki, and T. Fujino, “Model reverse-engineering attack using correlation power analysis against systolic array based neural network accelerator,” in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020.
- [167] M. Jagielski, N. Carlini, D. Berthelot, A. Kurakin, and N. Papernot, “High accuracy and high fidelity extraction of neural networks,” in *29th USENIX security symposium*, 2020.
- [168] N. Carlini, F. Tramèr, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson et al., “Extracting training data from large language models,” in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 2633–2650.
- [169] F. Tramèr, F. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, “Stealing machine learning models via prediction {APIs},” in *25th USENIX security symposium (USENIX Security 16)*, 2016, pp. 601–618.

- [170] X. Hu, L. Liang, S. Li, L. Deng, P. Zuo, Y. Ji, X. Xie, Y. Ding, C. Liu, T. Sherwood et al., “Deepsniffer: A DNN model extraction framework based on learning architectural hints,” in *ASPLOS*, 2020.
- [171] L. Batina, S. Bhasin, D. Jap, and S. Picek, “CSI NN: Reverse engineering of neural network architectures through electromagnetic side channel,” *USENIX*, 2019.
- [172] D. Ryu, Y. Kim, and J. Hur, “*gamma*-Knife: Extracting Neural Network Architecture Through Software-Based Power Side-Channel,” *IEEE Transactions on Dependable and Secure Computing*, 2023.
- [173] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, “Membership inference attacks against machine learning models,” in *IEEE symposium on security and privacy (SP)*, 2017.
- [174] A. Luqman, A. Chattopadhyay, and K.-Y. Lam, “Membership Inference Vulnerabilities in Peer-to-Peer Federated Learning,” in *Proceedings of the 2023 Secure and Trustworthy Deep Learning Systems Workshop*, 2023, pp. 1–5.
- [175] S. Shukla, M. Alam, S. Bhattacharya, P. Mitra, and D. Mukhopadhyay, ““Whispering MLaaS”: Exploiting Timing Channels to Compromise User Privacy in Deep Neural Networks,” *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 587–613, 2023.
- [176] B. Balle, G. Cherubin, and J. Hayes, “Reconstructing training data with informed adversaries,” in *IEEE Symposium on Security and Privacy (SP)*, 2022.
- [177] A. Salem, A. Bhattacharya, M. Backes, M. Fritz, and Y. Zhang, “Updates-Leak: Data set inference and reconstruction attacks in on-line learning,” in *29th USENIX security symposium (USENIX Security 20)*, 2020, pp. 1291–1308.
- [178] Z. Wang, F.-h. Meng, Y. Park, J. K. Eshraghian, and W. D. Lu, “Side-channel attack analysis on in-memory computing architectures,” *IEEE Transactions on Emerging Topics in Computing*, 2023.
- [179] B. Sapui and M. B. Tahoori, “Power Side-Channel Analysis and Mitigation for Neural Network Accelerators based on Memristive Crossbars,” in *2024 29th Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2024, pp. 612–617.
- [180] S. K. Roy and N. R. Shanbhag, “On the Security Vulnerabilities of MRAM-based In-Memory Computing Architectures against Model Extraction Attacks,” *IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2024 (to appear).

- [181] U. Banerjee, L. Ho, and S. Koppula, “Power-based side-channel attack for AES key extraction on the ATmega328 microcontroller,” *arXiv preprint arXiv:2203.08220*, 2022.
- [182] D. P. Sahoo, P. H. Nguyen, D. B. Roy, D. Mukhopadhyay, and R. S. Chakraborty, “Side channel evaluation of PUF-based pseudorandom permutation,” in *IEEE Euromicro Conference on Digital System Design (DSD)*, 2017, pp. 237–243.
- [183] Y. Wang et al., “Hertzbleed: Turning power {Side-Channel} attacks into remote timing attacks on x86,” in *31st USENIX Security Symposium*, 2022, pp. 679–697.
- [184] E. Wallace, M. Stern, and D. Song, “Imitation attacks and defenses for black-box machine translation systems,” *arXiv preprint arXiv:2004.15015*, 2020.
- [185] M. Kang, Y. Kim, A. D. Patil, and N. R. Shanbhag, “Deep in-memory architectures for machine learning—accuracy versus efficiency trade-offs,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 5, pp. 1627–1639, 2020.
- [186] N. R. Shanbhag, N. Verma, Y. Kim, A. D. Patil, and L. R. Varshney, “Shannon-inspired statistical computing for the nanoscale era,” *Proceedings of the IEEE*, vol. 107, no. 1, pp. 90–107, 2018.
- [187] M. Kang, P. Srivastava, V. Adve, N. S. Kim, and N. R. Shanbhag, “An energy-efficient programmable mixed-signal accelerator for machine learning algorithms,” *IEEE micro*, vol. 39, no. 5, pp. 64–72, 2019.
- [188] M. Nabeel, H. Gamil, D. Soni, M. Ashraf, M. A. Gebremichael, E. Chielle, R. Karri, M. Sanduleanu, and M. Maniatakos, “Silicon-proven ASIC design for the polynomial operations of Fully Homomorphic Encryption,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2024.
- [189] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, “Deep learning with differential privacy,” in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.