ANALYSIS AND IMPLEMENTATION OF DENSE ON-DIE MEMORIES USING TRADITIONAL AND COMPUTE-IN-MEMORY APPROACHES

A Dissertation
Presented to
The Academic Faculty

By

Samuel Spetalnick

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
Georgia Institute of Technology
School of Electrical and Computer Engineering

Georgia Institute of Technology

December 2023

ANALYSIS AND IMPLEMENTATION OF DENSE ON-DIE MEMORIES USING TRADITIONAL AND COMPUTE-IN-MEMORY APPROACHES

Thesis committee:

Dr. Arijit Raychowdhury (Advisor)
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Visvesh Sathe
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Shimeng Yu
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Alexey Tumanov
School of Computer Science
Georgia Institute of Technology

Dr. Suman Datta
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Win-San (Vince) Khwa
Corporate Research
Taiwan Semiconductor Manufacturing Company (TSMC)

Date approved: November 13, 2023

...always take your job seriously, never yourself.

Dwight David "Ike" Eisenhower

For my friends and family who have supported me for so many years.

ACKNOWLEDGMENTS

I would like to thank my PhD advisor, Dr. Arijit Raychowdhury, for providing guidance, encouragement, mentorship and support throughout my four-and-something years as a graduate student. His positive approach to mentorship has helped me to improve my ability to work through challenging design problems, and for that I am grateful. It is also a testament to him that he has assembled a lab full of very capable people who have similarly served as an invaluable resource during my graduate work.

I would also like to thank the members of my thesis committee, Dr. Shimeng Yu, Dr. Suman Datta, Dr. Visvesh Sathe, Dr. Alexey Tumanov, and Dr. Win-San (Vince) Khwa. I have benefited from their suggestions, comments, encouragement, and conversation through these years. Where we have crossed paths or collaborated, thank you for providing these valuable opportunities to learn about research, career, and life.

My friends and colleagues, old and new, at the Integrated Circuits and Systems Research Lab (ICSRL) have been instrumental to my professional success and confidence as a researcher. Brian and Muya have been key parts of my time as a PhD student and I want to thank you both for all of your help, conversations, and collaboration. Ashwin, I am glad we finally got to spend some time on projects together this last year.

Finally, I would like to thank my family. Mom and Dad, thank you for your unwavering confidence. Ralph, Heather, Chloe, Brian, and Kate, thank you for always being there. And to my friends from high school and Hopkins, thank you for the good memories and fun moments through the years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

1T1R 1-Transistor, 1-Resistor

ADC Analog-to-Digital Converter

BCD Bipolar, CMOS, and DMOS

BEOL Back-End of Line

BER Bit-Error Rate

BL Bit-Line

CIM Compute-in-Memory

CNN Convolutional Neural Network

CPU Central Processing Unit

DAC Digital-to-Analog Converter

DRAM Dynamic Random-Access Memory

eDRAM Embedded Dynamic Random-Access Memory

eNVM Embedded Non-Volatile Memory

FDSOI Fully-Depleted Silicon-on-Insulator

FeFET Ferroelectric Field-Effect Transistor

GC Gain Cell

HBM High Bandwidth Memory

HRS High-Resistance State

LLC Last-Level Cache

LRS Low-Resistance State

MAC Multiply Accumulate

MB Megabyte

MIM Metal-Insulator-Metal

ML Machine Learning

MLC Multi-Level Cell

MOM Metal-Oxide-Metal

MOSCAP MOS Capacitor

MRAM Magneto-Resistive Random-Access Memory

MUX Multiplexor

NoC Network-on-Chip

NVM Non-Volatile Memory

PCM Phase Change Memory

PE Processing Element

RRAM Resistive Random-Access Memory

SA Sense Amplifier

SAR Successive Approximation Register

SL Source-Line

SRAM Static Random-Access Memory

TIA Transimpedance Amplifier

VLIW Very Long Instruction Word

WL Word-Line

SUMMARY

This dissertation discusses the challenge of efficiently computing with large on-chip memories using two proposed innovations: compute-in-memory (CIM) to optimize the process of accessing and computing with data, and emerging embedded nonvolatile memories (eNVM) as improved storage technologies for large on-die memory banks. The first innovation, CIM, proposes to offer improved performance, through reduced per-bit access costs, improved bandwidth, and reduced digital computing costs. However, the gains from CIM are not guaranteed: peripheral or bitcell area overheads can reduce bit density of the memory, and non-idealities can create an unfavorable accuracy-efficiency trade-off. The second innovation, eNVM, offers the possibility of limiting off-chip data accesses in the context of energy-constrained accelerators. Still, these potential benefits can be minimized by design patterns that lead to increased access energy or lower bandwidth for eNVM, bringing performance closer to that of off-chip memory.

After introducing and motivating machine learning (ML) as an important type of data-intensive task and describing the field of proposed eNVM technologies, this dissertation introduces CIM and all-on-chip computing as proposed ways to make use of these eNVM technologies. The second chapter discusses CIM in the context of traditional CMOS, to remove the variable of eNVM technology and draw a few general conclusions. The next two chapters focus on resistive random-access memory (RRAM), a specific flavor of eNVM, and discuss in total three implemented and tested designs that make use of RRAM in a foundry process. These designs aim to resolve some of the challenges of, first, CIM with RRAM and, second, all-on-chip computing with RRAM (*without* CIM). They were successfully fabricated and tested in silicon to validate the results.

The first two designs, discussed in the third chapter, implement current-summing CIM with RRAM and attempt to overcome challenges: the first macro improves storage density through circuit techniques and an optimized physical design, while the second macro ad-

dresses non-idealities. Specifically, the second macro introduces two kinds of augmented offset cancelling, in both the current-sensing transimpedance amplifier (TIA) front-end and the analog to digital converter (ADC), to reduce the impact of channel-to-channel variations on CIM accuracy while also managing IR drop and off-state current. By introducing techniques to address these challenges, these works have shown how some of the penalties associated with current-summing CIM with eNVM can be mitigated.

The final design is an end-to-end inference accelerator implementation with RRAM that moves away from the accuracy-efficiency trade-off space imposed by current summing CIM. This design instead follows a density-oriented all-on-chip computing approach with fully digital MACs replacing CIM. A custom RRAM macro is introduced to improve area and energy efficiency, and it is integrated tightly into a modular very long instruction word (VLIW)-based matrix module. Ten of these matrix modules are implemented on-chip to provide 5MB of on-chip NVM to support the all-on-chip edge inference application. The hierarchical design improves memory access characteristics, including energy and area-normalized NVM bandwidth while maintaining an advantage in either compute density or NVM storage density over prior published all-on-chip computing with eNVM work.

CHAPTER 1
MOTIVATION, BACKGROUND, AND INTRODUCTION

1.1 Motivation

Machine learning with deep neural networks (DNNs) has emerged as a driving technology across wide and varied domains within signal, image, and speech processing [1]. For computer systems or application accelerators executing any of a variety of modern data-intensive computing tasks, including ML inference [2], moving data from large memories into processing elements can be a limiting factor for energy efficiency and/or throughput (performance) [3].

The high cost of off-chip memory accesses has led to the extensive use of large, dense on-chip memories. These memories can store part (or all) of the data required by the application, such as the weights used to compute forward inference based on an input signal in the case of ML. Large caches allow the hardware to work continuously as data is loaded from off-chip memory (hiding memory access latency, [4]), while memories that are large enough to provide the entire application storage requirement on-chip can eliminate the need for off-chip accesses entirely (*e.g.* [5]). Subsequently, the potentially large leakage energy and limited density of on-chip volatile memory (classically static random-access memory, SRAM) has encouraged the development of newer embedded memory technologies including emerging embedded nonvolatile memories technologies (eNVMs, [6]), such as resistive random-access memory (RRAM), magneto-resistive random-access memory (MRAM) or phase-change memory (PCM); and on-die, back-end-of-line (BEOL) or near-die dynamic random-access memory (eDRAM/DRAM, [7]). The cost of accessing data stored in these large on-chip arrays has also motivated the re-emergence of the analog compute-in-memory (CIM) paradigm, where the stored states in multiple memory cells are selectively added to-

gether inside the memory array so that the read-out value represents a multiply-accumulate (MAC) operation result.

These new devices and the CIM technique both present design and implementation challenges at the intermediary circuit and the system architecture levels. Relative to traditional SRAM, eNVMs and some eDRAM technologies can require high voltages to program states, can have limited dynamic range (on/off ratio, such as R_{OFF}/R_{ON} for RRAM), and may present higher bit-error and fault rates along with limited endurance. CIM requires more precise readout circuitry which can include a relatively complex, large, and high-power analog-to-digital converter (ADC). The effect that any of these factors have on application suitability depends on design decisions at the circuit level when constructing fully-integrated memory sub-blocks and at the system level when deciding how to organize the memory blocks in an accelerator. Therefore, between these new low-level primitives and application compatibility are the necessary steps of circuit topology choice, design implementation, system design, and related design-space exploration.

1.2 Background

The "memory bottleneck" or "von Neumann" bottleneck (named as early as [8], referring to the Von Neumann architecture, see [9]) describes the bandwidth-limited connection between a processor and its memory as the limiter of overall computing system performance. The bottleneck applies similarly to system energy efficiency, with cache leakage and off-chip memory accesses contributing significantly to overall system energy use ([3], section 5). Abstractly, a processor or processing engine (PE) will require some application- and architecture-dependent amount of new data for each functional operation.

The ratio of functional operations to data fetches (arithmetic intensity) may be much greater than one, relating to the concept of data reuse. For a PE to achieve greater arithmetic performance, closer to the theoretical maximum, a cache [10, 11] or scratchpad memory [12] physically local to a PE stores data that is known or predicted to be re-used in the near

future so that expensive accesses to a larger physical memory (that might be further away or off-chip) can be avoided. Data fetched from this local storage can be involved in multiple simultaneous functional operations, so that, in net, there can be many functional operations for each local storage access and up to orders-of-magnitude more for each remote (*e.g.* DRAM) memory access. Maintaining large ratios is important due to the energy and time overhead of accessing larger memories and eventually going off-chip. The cheapest off-chip memory accesses can cost a few picojoules *per-bit* before considering the internal DRAM array energy at the off-chip memory [13].

Following this intuition, large on-chip memories have long played a fundamental role in translating application-derived data reuse into a reduction in expensive[1] off-chip DRAM accesses during run-time [14, 15]. It has been shown that the average energy and latency of memory accesses improve with increased cache size up until design penalties for larger SRAM array sizes overcome the incremental benefits from reduced miss-rates [16]. These fundamental density vs. access latency and energy trade-offs occur due to increased bit-line (BL), source-line (SL) and data routing capacitances and characteristic delays in larger, denser arrays along with a large leakage power component that scales with SRAM cell count [17, 18]. The first challenge can be mitigated using multi-level on-chip SRAM storage hierarchies, where the lowest level storage (local scratchpad or L1) which interfaces to the PE or processor is kept small for high speed and low power, while the higher-level storage (such as last-level cache, LLC) can be greatly expanded since its per-access time and energy are partially hidden by the hit-rate of the lower-level storage. Performance-oriented modern central processing units (CPUs) [19, 20, 21, 22, 23, 24], graphics processing units (CPUs) [25, 26, 27], and tensor processors [28, 29] almost universally implement on-die cache hierarchies organized as one or two levels of memory near individual cores or processing elements (PEs) in addition to a massive fully or partially shared cache of at least a few megabytes (MBs).

[1] In terms of energy per pit (pJ/bit) and/or latency.

However, while architectural choices, including cache hierarchy design, can limit the performance impact of access time and energy penalties associated with huge SRAM-based caches they cannot reduce the technology-driven limitations: leakage power and limited density. SRAM memory macro designs [30, 31, 32, 33, 34, 35], may leverage innovations such as low multiplexer (MUX) ratios and assist techniques to reduce dynamic power and aggressive V_{MIN} scaling and deep-sleep modes to reduce leakage but traditional static random-access memories (SRAMs) built with logic transistors cannot be fully power-switched while retaining stored data. Likewise, array-centered techniques such as the use of cells with lower current ranges (low-leakage cells) to reduce leakage currents trade directly against speed and/or density. SRAM density has scaled aggressively with logic as far as the 16nm technology generation; however, SRAM scaling slowed, relative to logic, in reported data for the 7nm and 5nm nodes [36, 37, 38, 39]. As an example of the slowing trend, Taiwan Semiconductor Manufacturing Company's (TSMC's) 16nm FinFET technology, according to published data, scales logic 2x relative to the planar 28nm node, with SRAM scaling similarly by 1.81x, a 92% alignment. From 16nm to 7nm, TSMC published a claimed 3.3x gate density improvement, with SRAM scaling 2.6x, a lower 79% alignment. Subsequently, the 5nm technology claims a 1.84x improvement in logic density relative to 7nm, while only showing a 1.28x improvement in SRAM bitcell density, an even lower 70% alignment. In the latest 3nm generation [40], logic density reportedly improved 1.18x to 1.56x while *no* density improvement was reported for the high-density SRAM cell.

These two persistent challenges, leakage power and density, have motivated the exploration and development of new, alternative embedded memory technologies including eNVMs and eDRAMs. Both of these technology classes attempt to improve beyond the density of SRAM by using simplified 1 - 3 transistor structures and a special switching material or device. Some prospective eDRAM technologies have introduced the possibility of 3D integration for further density improvements, conceptually similar to existing

and upcoming high-bandwidth memory (HBM) DRAM technologies [41, 42]. eNVMs, in contrast to SRAM, offer true zero-leakage modes while retaining data. This means that architectures using fine-grained power gating when computing with suitable application profiles can potentially decouple on-die memory size from total memory leakage power.

The next section of background material will contextualize the discussion by showing a pair of ML benchmark data-sets and discussing the weight storage footprint required to achieve strong benchmark accuracy. This will illustrate the need for larger on-die memory for edge applications where designers aim to reduce or eliminate off-chip accesses. After this, recent work on eNVM and eDRAM technologies will be introduced, focusing on specific or quantifiable advantages and disadvantages. Finally, CIM will be introduced. CIM is a technique that may improve the efficiency or throughput of reading data from memory, while accelerating MAC arithmetic. Recent analyses of prospects for CIM will be reviewed.

1.2.1 Application Discussion

This section, along with this dissertation, will focus on ML inference as the default application under consideration due to its recent importance. As just described, ML inference is a representative modern application with structured requirements for large input data [2]. A variety of dataflow options are available for accelerating ML inference. While inference accelerator design is discussed later in this background chapter, it is useful for the current discussion to introduce some basic examples: for traditional designs, an accelerator might use an output-stationary dataflow so that weights for neural network layers are sequentially streamed into a processing block with some number of parallel data-paths [43, 44]. In a CIM design, the dataflow could, by default, be weight-stationary with the weights stored in CIM-enabled memory arrays. In either case, input data (*e.g.* images, text, or signal data) are streamed in from off-chip, and activations are output and input into an on-chip buffer. If activations overflow the on-chip buffer, they are streamed to and from off-chip DRAM.

In the baseline case, this pattern would be re-used for weights as well. As noted previously, expensive off-chip accesses can be reduced if the weights and activations for the network can fit in on-die memory.

Since the total size of weights in a network can greatly exceed the peak (transient) activation storage requirement [45], fitting most or all of the application network's weights on chip can be challenging. An important design consideration for edge inference accelerators, then, is how to fit more weights on chip and further reduce off-chip accesses during computation. This motivates an analysis of the memory footprint of networks for solving various tasks, and how this footprint relates to inference accuracy. Two example tasks, which could plausibly apply to energy-constrained mobile platforms, are presented below.

Microsoft's Common Objects in COntext (COCO) dataset forms the first example. This dataset has served as a widely-used benchmark for training and evaluating object recognition networks [46]. When the dataset was reported, a baseline network (DPMv5-C, [47]) was also reported with an average precision (AP) for the bounding-box detection problem of 19.1. State-of-the-art object detectors leveraging transformers [48] have now achieved AP scores above 64 [49]. Figure 1.1 shows state-of-the-art performers on the COCO dataset across a range of network sizes. A strong log-dependence (exponential trend) of AP accuracy on network size is observed: every 12 AP requires a factor of ten increase in the number of parameters.

As a second example, LibriSpeech is a dataset for training and evaluating speech recognition models developed using public-domain audio books [55]. A similar trend to that for COCO is shown in Figure 1.2: there is an inverse log-dependence of word-error rate (WER) on network size. The trend is close to 0.50% per decade.

From these examples, three important trends can be observed: first, the dependence of application accuracy on network size is obvious once the field is limited to state-of-the-art models along the parameters-performance frontier; second, the dependence has a logarithmic trend in number of parameters (number of parameters scales exponentially for a linear

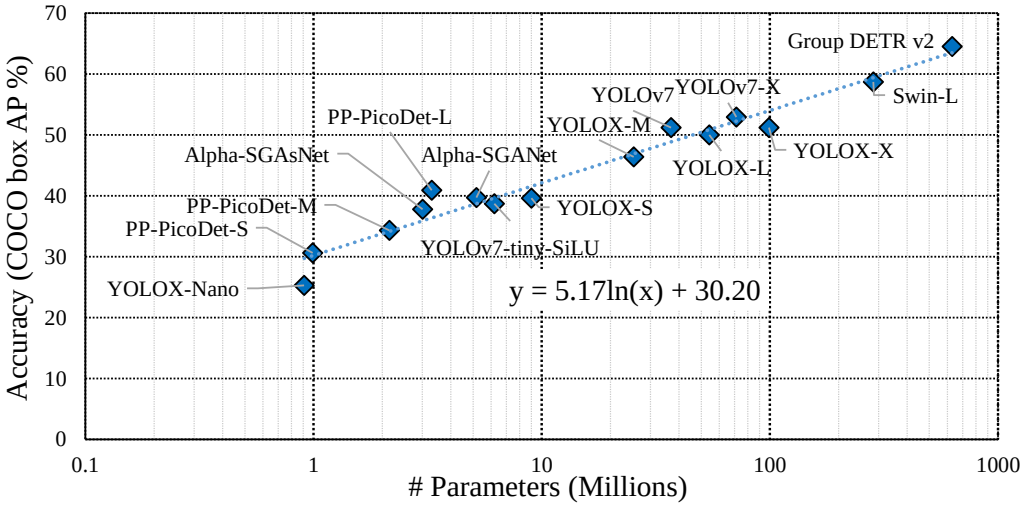


Figure 1.1: State-of-the-art object detectors sorted by performance on the MS COCO dataset. The trend is ≈12 AP per decade. Included are YOLOX [50], Group DETR v2 [49], PP-PicoDet [51], Alpha-SGANet [52], YOLOv7 [53], and Swin v2 [54].

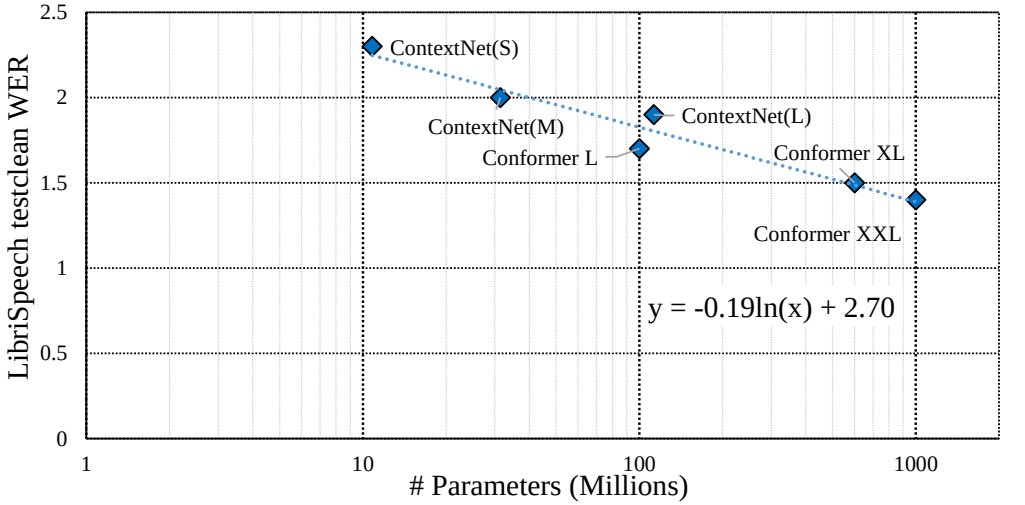


Figure 1.2: Speech recognition models sorted by performance on the LibriSpeech dataset. The downward error-rate trend is ≈0.46 WER per decade. Included are ContextNet [56] and Conformer [57].

improvement in accuracy accuracy), meaning significant improvements in accuracy require order-of-magnitude increases in network size; and, finally, the best-performing networks are relatively massive, in the 500 million to one billion parameter range. This all suggests that the need for large on-die weight storage is a typical characteristic of ML applications and is likely to continue to grow as ML network sophistication further increases.

1.2.2 Emerging Embedded Memory Technologies

Given the increasing demand for large on-die memories driven by ML and other memory-intensive workloads, technologies with better density and/or leakage trends than SRAM have recently received increased attention. A few such technologies will be briefly reviewed here with a focus on RRAM which, due to foundry support and compatibility at the mature 40nm planar logic node, is the technology leveraged in the presented research for this thesis. Beyond discussing RRAM, this sub-section will review other eNVMs including PCM, ferro-electric field-effect transistors (FeFETs), spin-transfer torque magneto-resistive RAM (STT-MRAM), and classic embedded flash (eFlash). This sub-section will also briefly discuss embedded DRAM.

RRAM

RRAM (sometimes ReRAM) is a logic-compatible nonvolatile memory technology based on the metal-insulator-metal (MIM, here *not* used as a capacitor) structure built using carefully chosen contact and dielectric materials [58]. The principle of operation involves the formation and rupturing of a conductive filament which allows at least two stable resistance states. This resistive switching of the insulator between a low-resistance state (LRS) and a high-resistance state (HRS) in response to "SET" and "RESET" pulses allows this structure to function as a memory cell. RRAM has been integrated with foundry CMOS and reliably manufactured down to the 40nm and 22nm nodes, including a FinFET node [59, 60, 61, 62]. These arrays, and all typical RRAM embedded memory arrays, use a

1-transistor, 1-resistor (1T1R) memory cell structure, where a selector transistor in series with the resistive device is used to allow or block current flow in a given row of memory cells based on the voltage on the word-line (WL).

During a digital (non-CIM) read operation, one word of data is accessed by raising one WL to logic-'1', which is typically VDD. Selected columns are each connected to a read circuit. A sense-amplifier (SA) in each readout column is connected via a multiplexer (column MUX) to a single BL out of a group. The SA typically uses a current-sensing topology, where the front-end of the SA clamps the BL voltage, flowing enough current through the selected RRAM cell to maintain the voltage of the selected BL near a target clamping voltage. The required current to perform this clamping is compared to a reference, with the comparison result stored in a final voltage-driven latching stage. The logical flow is from (cell) resistance, to current, to voltage, and finally to digital bits. For digital readout, these output bits, one per SA, represent the data stored in the selected word in memory. Writing RRAM cells is performed by applying a timed voltage pulse either forward (SET operation, toward lower resistance, $V_{BL} > V_{SL}$) or in reverse (RESET operation, toward higher resistance, $V_{SL} > V_{BL}$). The WL voltage is also pulsed, with WL voltage partially controlling write current allowance. All of these voltage pulses can easily exceed the core voltage range for the process's logic transistors.

While RRAM cell area could theoretically scale as $4F^2$ [58], the reported array implementations that disclose bitcell area in 40nm and 22nm CMOS claim a $53F^2$ cell footprint. This is still approximately 3x denser at the cell level than SRAM for these nodes ($162F^2$ at 28nm, [63]). Major challenges for RRAM include endurance, with these foundry cells surviving approximately 10^3 - 10^5 endurance cycles, and write-voltage, which can be 3.6V or higher, especially during the forming stage when cells are written for the first time. More manageable challenges include the low R_{OFF}/R_{ON} ratio. This can make LRS and HRS cells harder to distinguish during read, although in principal readout circuit design techniques can overcome this issue (potentially at the cost of area, power, or delay). Tight

memory windows, which correspond to low R_{OFF}/R_{ON} ratios, place a greater importance on small offsets in the readout SA during current sensing. These offsets can occur transiently due to noise or permanently due to random variation or even asymmetry in the SA circuit or physical layout; the typical solutions, if needed, are to increase the size (*e.g.* increase effective length of mirror pair) of transistors closer to the input (before small-signal gain) and/or employ offset cancelling. Memory window can also be degraded by temperature sensitivity and systematic process drift in either the HRS and LRS. While in the worst case a reduced memory window results in readout errors in weak channels, in the milder case it reduces readout speed by requiring more signal development time before latching the readout decision. Timing is typically set by tail HRS cells or tail LRS cells, depending on the reset condition of the current-sensing pre-amplifier in the SA. While the current-sensing SA design is sensitive to memory window in absolute terms, memory window can also be stated in normalized terms, for example, as a multiple of device conductance standard deviation. This allows quick estimation of bitcell-variation-induced bit-error rate (BER).

STT-MRAM

Aside from RRAM, there is a wide array of proposed eNVM technologies. STT-MRAM is promising and has been demonstrated in various scaled nodes down to at least 16nm FinFET [64, 65, 66, 67, 68, 69]. STT-MRAM achieves similar density to RRAM, but has shown excellent write endurance of at least 10^6 cycles. Still, STT-MRAM requires large write currents and can exhibit a very small R_{OFF}/R_{ON} ratio; works showing the switching range often report under 2x resistance switching (magneto-resistance ratio $< 200\%$). In 22nm or lower nodes, STT-MRAM bitcell areas have ranged from $94F^2$ to $129F^2$. At the larger 28nm and 40nm nodes, comparable to RRAM-compatible nodes just described, reported foundry STT-MRAM bit-cells have ranged from $41F^2$ to $46F^2$.

PCM

PCM has only recently reached a comparable level of maturity as RRAM or STT-MRAM in terms of major foundry support [70, 71, 72, 73, 74]. PCM arrays have been demonstrated in a planar 40nm logic node; a 90nm bipolar, CMOS, and DMOS (BCD) node for power applications; and fully-depleted silicon-on-insulator (FDSOI) nodes down to 18nm. Reported PCM implementations are as dense or denser than RRAM at $24F^2$ in 28nm FDSOI ($59F^2$ in 18nm FDSOI). As with other technologies, R_{OFF}/R_{ON} can be small especially for tail HRS bits, at higher ambient temperatures, and/or after longer retention times. However, in the best case, two to three orders-of-magnitude are possible which compares favorably to RRAM and STT-MRAM. Endurance for PCM can exceed 10^6 cycles. Scaling of PCM into newer (vs. 40nm) traditional planar nodes or into FinFET nodes has not yet been reported.

FeFETs

FeFETs have been demonstrated in a foundry 28nm FDSOI node and projected beyond the 22nm FDSOI process. FeFETs promise even higher density than RRAM and STT-MRAM with a 1T structure theoretically possible, large I_{ON}/I_{OFF} greater than two orders-of-magnitude, and endurance at least in the range of 10^5 and up to 10^{11}, but require high gate voltages up to $\approx 4V$ for write [75, 76, 77, 78, 79]. Despite this potential performance, highly-reliable (6σ or greater state separation) FeFET-based embedded memories with competitive cell density in a modern scaled logic process, like the array macros reported for RRAM, STT-MRAM, and PCM, have yet to be reported.

eFlash

Finally, the most mature embedded nonvolatile technology is almost certainly embedded flash (eFlash) based on the floating gate or charge trapping approaches. Compared to the emerging technologies, eFlash has been widely implemented across an assortment of processes, generally trailing logic process generations by a few years [80, 81, 82, 83, 84].

Memory macros, primarily targeting automotive applications, have been reported down to the 28nm planar node with scaling to 22nm proposed. The dual-gate approach preferred by designers in modern planar nodes is more like a 2T structure than a 1T floating-gate cell, leading to comparable or worse density when compared to the emerging 1T1R technologies. Cell areas in scaled planar nodes have ranged between $33F^2$ and $74F^2$, with the latter occurring for eFlash scaled to the 22nm node. The prospects for Flash scaling with integration into more advanced nodes, including FinFET nodes, are not well understood with a clear scaling path yet to emerge. This scaling concern is due to factors including voltage compatibility, cost, logic process compatibility, and access time.

This has led to more interest in emerging technologies, including the above-discussed RRAM, STT-MRAM, PCM, and FeFETs even just to replace eFlash for firmware storage in embedded processors. In addition to service as drop-in eFlash replacements for, *e.g.*, automotive use cases, these technologies may also be useful as last-level cache replacements in low-power processors or bulk on-chip storage to support area- and energy-efficient all-on-chip ML inference. Replacing SRAM, as is implied if an eNVM bank is to serve as a last-level cache for a processor, is a more challenging use case since net cycling endurance, access bandwidth, and potentially access latency and energy must be similar to those of high-performance logic-transistor-based SRAM.

eDRAM

Another option for bulk on-chip storage and caching is eDRAM. Beyond the mentioned eNVMs, eDRAM technologies [7], which are volatile memory technologies offering better area density than 6T SRAM and/or back-end-of-line (BEOL, [85]) compatibility for 3D integration, have been consistently reported. Both traditional 1T1C technologies, which promise aggressive bitcell scaling at the cost of added process steps and the need for frequent refresh cycles, and gain-cell (GC)-based eDRAM have been proposed. GC bitcell structures consist of more than one active device (transistor), allowing the bit-cells to am-

plify the voltage-domain signal due to the stored charge on the storage node. This detaches the fragile stored charge state from the read process, allowing non-destructive read while potentially improving read margins and allowing better voltage scaling. GCs can include either an explicit or implicit storage capacitor, where an explicit capacitor is often a MOS capacitor (MOSCAP) rather than a BEOL structure [86].

The three potential advantages of eDRAM over traditional DRAM are a reduction in component count and potentially overall application cost by eliminating off-chip dual data-rate (DDR) memory modules, a reduction in memory access cost (in pJ/bit) trading off-chip bit accesses for on-chip accesses, and an increase in memory access performance (reduced latency, higher bandwidth). GC-eDRAM has been demonstrated in a foundry 16nm FinFET node, showing limited (linear) density improvements relative to SRAM [87]. Due to the limited density advantages of CMOS GC-eDRAMs over SRAM and the issue of leakage leading to the need for periodic refresh operations, eDRAMs which use non-CMOS layers in the back-end-of-line stack to enable lower leakage and the possibility of stacking several memory layers have been proposed at the research level [88]. As a point of comparison with the previously mentioned technologies, reported foundry 1T1C eDRAM technologies in 40nm, 28nm, and 22nm FinFET have achieved bitcell areas in the range of $36F^2$ to $60F^2$ with this normalized size increasing for smaller nodes [89, 90, 91]. The 1C is a MiM structure, and the resulting area is competitive with eNVMs. Importantly, unlike with eNVM, eDRAM macros cannot retain data at zero power during retentive power-down.

1.2.3 Compute-in-Memory Approaches

Having introduced a variety of on-chip memory technologies, the next two sub-sections will describe proposed ways to use these technologies to accelerate ML applications using innovative techniques. This sub-section will start by discussing compute-in-memory with eNVMs, while the next sub-section will describe how designs merge a traditional digital-

MAC-driven methodology with these emerging memory technologies.

CIM Overview

For inference accelerator designs that do not use compute-in-memory (*e.g.* see [43]), even if the weight memory can be fully stored on-chip, the weights need to be streamed out of the memory and transferred to the storage local to the processing data path. Weights are stored in registers in/near the digital MAC logic, used once or, for better performance, reused many times, then replaced by new weights from the larger memory. This imposes theoretical overheads: redundant storage is needed to buffer the weights and avoid repeated costly memory accesses, which adds area and leakage power; the MAC units may need to stall while waiting for weight data to be read from the memory, especially if memory bandwidth is significantly lower than MAC bandwidth and weight data re-use is low (low arithmetic intensity); and there is an energy cost associated with every bit of data accessed from the memory, which will increase for the large memories which are typically needed to manage access latency due to even larger, further-away data stores.

CIM has been proposed as a way to reduce or eliminate the memory-access overheads [6]. CIM proposes to improve the effective bandwidth from the memory and reduce the per-bit energy cost by increasing the effective number of weight bits read at once. CIM can be seen as implementing an ideal version of the weight-stationary architecture [2]. Under the CIM paradigm, input activation data is streamed to the memory array(s), where the weights are stored statically during the computation under consideration. In some cases, the weights fit entirely on chip and no weight writing cost is incurred during computation. Instead of reading out the weight data and using digital logic to perform the MAC with the activation data, the activation data is provided as an input to the array. In general, each IO channel of the CIM macro combines part or all of the input activation vector data, at full or reduced precision, with multiple weight data bits stored in the array to produce an output value that represents a vector dot product *instead of* a single bit of weight data. Clearly, if

the overheads can be well-managed, this provides an increased throughput.

Basic CIM Example

CIM flavors involve a range of input, output, and weight data formats, ranging from fully binary to fully analog [92]. To better introduce the idea, consider a powerful (but expensive and difficult to implement) near-fully-analog variant of CIM. A resistive technology, like RRAM or PCM, is used to store weight data. Each memory column (BL/SL) stores a full vector of weight data, with each element stored in the cell conductance domain at full precision. This is achieved by trimming the conductance of storage elements to align with the corresponding floating-point weight value of the vector element. For this example, the WL drivers do not drive a binary value onto the WL; instead, they each contain a high-precision digital-to-analog converter (DAC) that allows them to bias the resistive storage elements in each row to a specific voltage using the access device. Each resistive storage cell now pulls a current, from BL to SL (VSS), proportional to the product of the input value and the stored weight. In total, on a BL for input vector X and weight vector W with length L:

$$I_{BL} = \sum_{i=0}^{L} V_{Unit} X_i \times G_{Unit} W_i = I_{Unit} \sum_{i=0}^{L} X_i \times W_i = I_{Unit} X \cdot W \tag{1.1}$$

This example illustrates the idea of multiplying an input vector against a stored weight vector and producing a result in the current domain, which can then by measured by an ADC and output as the dot-product result. This near-fully-analog variant requires significant assumptions and overheads: complex DACs at the WL drivers, the ability to voltage-bias storage resistors accurately using the access device or some other means, the ability to store many bits of data in each storage resistor, the ability to convert the output current back to the digital domain, and the ability to enable the full vector width in parallel. In addition to improving bandwidth by accessing multiple bits in parallel, the MAC workload is also handled in analog under this approach. Implemented variants of CIM seek to find a

useful compromise between this variant and the traditional approach of reading the weight memory line-by-line.

CIM Variants

The potential for significant access bandwidth and efficiency gains, under ideal conditions, from switching to a CIM-based design has encouraged the exploration of a wide variety of CIM formats. This sub-section will now discuss two adjacent technologies, charge-domain CIM and digital CIM, before elaborating on design patterns within the current-summing CIM framework that was just described.

These other CIM flavors address a few challenges that result from performing accumulation *directly* in the current domain using the scaled data storage devices. Memory devices suffer from process, voltage and temperature (PVT) variation: the HRS and LRS conductivities (unit current) can drift spatially and die-to-die, can suffer from local random variation, can be sensitive to changes in WL voltage, and can be sensitive to chip temperature changes. The local variation challenge follows from basic memory design goals. Bit-cells are scaled aggressively to improve overall storage density, since 50% or more of memory macro area can be due to storage cells, while random parametric variation is typically inversely proportional to device area (this trend is widely known for transistor threshold voltage matching [93]). Current-domain CIM presents other design challenges which charge-domain computing addresses, including the need for a transimpedance amplifier (TIA) consisting of costly linear gain stages and large currents when many LRS cells are enabled in parallel.

The first response to these challenges maintains analog-domain accumulation but changes the unit of accumulation away from cell current and toward a more consistent quantity that is nonetheless easily available in modern CMOS processes and straightforward to measure. Charge-domain CIM proposes to change the matched component to a unit capacitance by accumulating in the charge domain [94, 95, 96, 97, 98]. Metal-oxide-metal (MOM) capac-

itors offer good matching performance in scaled CMOS [99]. An extra step is required, since the cell data must be read out before being used to control a switch that selectively contributes a scaled unit of charge due to each selected memory cell. If the operation is truly performed in-array, the bitcell must accommodate at least one added active device to control a local switch. Otherwise, additional peripheral area is needed to accommodate the extra step. In any case, it is often possible to overlap the BEOL capacitor and the memory storage cells. A capacitive successive approximation register (SAR) ADC is then a good fit for the measurement and conversion back to the digital domain. Charge-based CIM offers improved robustness and promises excellent energy efficiency and compute density at the cost of area overheads, since the basis of the idea is to avoid using fully-scaled memory cells to perform accumulation. They can therefore present lower storage density (even if compute density remains high). Furthermore, while variation and noise-induced errors can be minimized, quantization noise remains a factor since a column ADC is required. Charge-based CIM implementations do not typically recover full precision to save on mixed-signal circuit expenses (*e.g.*, an ADC would ideally need to recover a full $\approx$22b in the case of 8b inputs, 8b weights, and 64-wide MACs).

The second response to analog-domain challenges is to switch the MAC computation back to digital. This parallels the intuition that drives charge-domain CIM: instead of accumulating using cell current, convert the cell current to digital and then compute using that value. For digital CIM, the analog-domain simplification is even stronger; the accumulation uses digital gates in place of charge summing [100, 101, 102, 103, 104, 105]. Digital CIM arrays typically consist of a bank of memory cells (such as 6T SRAM cells), a readout mechanism that produces bits from the selected weight-storage cells, an AND-like logical block that performs one-bit multiplications, and an adder tree digital block that performs accumulation. In the latest logic nodes, such as TSMC's 5nm FinFET node, these custom adder trees can be extremely efficient allowing this nominally simple format to produce energy efficiencies and compute densities comparable to or greater than those of

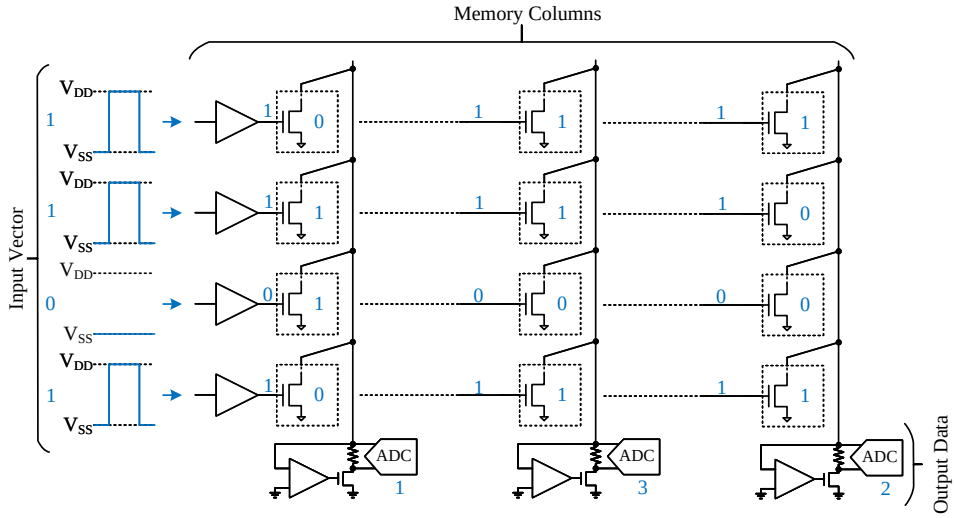


Figure 1.3: Basics of CIM with binary inputs on the WL, binary values stored in cells, and multi-bit analog accumulations on the BLs in the memory columns.

current-summing without associated accuracy degradation. Compared to standard digital logic-based data flows, digital CIM macros embed the MAC gates directly in the macro and implement them using hand-drawn custom logic.

Current-Summing CIM with RRAM

The non-current-based CIM mechanisms typically require adding special-purpose summing circuitry adjacent to relatively small blocks of cells. Furthermore, they are not natively compatible with eNVMs like RRAM, which require a sensitive front end to distinguish stored memory states. For example, a recent digital CIM implementation, [100], uses 12T bit-cells that include an output driver to intrinsically provide a solid digital-domain signal for the subsequent MAC logic to immediately use. This 12T cell is 3.57x the area of the reported 6T cell for the process [30]. While compute density and efficiency are very strong, storage density (4.93Mb/mm^2) is about 20% of a high-density array assuming 50% array efficiency for pure memory macros using the densest reported 6T cell. Another recent foundry digital CIM [105] shows lower storage density (3.215Mb/mm^2). Digital or charge-summing CIM arrays, therefore, might not replace the highest-density on-chip

weight buffers but could replace lower-level caches in high-reuse data-intensive MAC scenarios.

An advantage of current-summing RRAM arrays is that they can maintain the format of having a dense bitcell array with changes made only to peripheral circuitry on the sub-array boundary. CIM sensing circuitry then replaces binary readout circuitry. This format thus remains valuable as it proposes to maintain the eNVM benefits of cell density and low leakage while adding some of the efficiency and bandwidth benefits of CIM. The first step toward relaxed current-summing CIM design constraints is to "digitize" the design, shown in Figure 1.3. Instead of viewing the resistive memory cell as fully reflecting a floating-point weight value, the resistive memory cell is programmed to one of two or more conductance targets so that, when selected, it contributes some fixed small multiple of the unit conductance, G_{UNIT}. Thus, if the weights in the network are quantized to an integer format like INT8, each memory cell can store a subset of the weights. In the simplest case, eNVM cells are used in a binary mode where, ideally, they contribute either no conductance or exactly G_{UNIT} to the accumulation. This binary scheme maximizes the use of the cell dynamic range to reduce state variation. More generally, digitized cell states that represent only part of the weight precision opens a trade-off space (states per cell vs. accuracy/endurance loss) for design optimization.

An analogous simplification on the input side is similarly impactful not only because of the sparing use of dynamic range but also because it aligns CIM with the typical topology of foundry eNVM arrays. Typical RRAM arrays, for example [59, 60], use vertical BLs, vertical SLs which can be shared across multiple columns (ratio of BL:SL can be >1), and a horizontal WL. Thus, connecting the readout circuit to a certain BL/SL selects the column under consideration and driving a certain WL selects the row(s) which contribute(s) current. This row selection occurs through the 1T selector device in the 1T1R structure. The 1T selector can be modeled as a resistor in one of two states (plus variation) in series with the cell resistance; the foundry design goal is to minimize on-state resistance of the

selector when the WL is driven to core-VDD to improve cell dynamic range. To achieve improved resistive memory cell performance, the series resistance of the 1T1R cell should be dominated by the resistance due to the switching 1R material. The translation from WL voltage to cell current is thus expected to be nonlinear. While, nonetheless, there must be some magic-number WL voltages (along the 1T1R's nominal I_{BL}/V_{WL} transfer function) which cause a memory cell to pull half-current, quarter-current, and so-on, attempting to use scaled WL voltage to represent multiple input data states on an individual WL will use up cell dynamic range while in most cases providing a non-ideal result.

Both of these design changes imply that the entire vector-vector MAC is not computed in a single step but instead involves multiple mixed-signal operations in the memory array. The results are then appropriately bit-shifted (left-shifted once, that is, doubled, for each bit position above the LSB in the input and weight) and accumulated in a partial-summing digital circuit. For example, with 1-bit cells and 1-bit WLs, an 8b-input by 8b-weight MAC involves a 64x overall multiplier on the number of CIM reads. These can be time- or area-multiplexed. Having committed to using multiple read steps to produce the result, the final degree of freedom is to avoid applying the full vector-width on the WLs in a single step. Instead, a kernel with fewer active WLs, or allowed parallel summed cell currents, than the total number of elements in the MAC vectors can be repeated multiple times for each MAC operation until the total vector width is accumulated.

In the final picture: each cell is programmed to one of P_w stored states, P_x states are represented per-WL, and up to P_{WL} such WLs are activated in parallel. For an N-element-wide MAC operation with B_x input bits and B_w weight bits, the number of steps required M is then:

$$M = \lceil \frac{N}{P_{WL}} \rceil \times \lceil \frac{B_w}{\log_2(P_w)} \rceil \times \lceil \frac{B_x}{\log_2(P_x)} \rceil \tag{1.2}$$

Clearly, diverging from fully-analog all-at-once MAC computation to a kernel within this space of feasible implementations can apply a significant dividing factor to bandwidth and energy efficiency. Still, the expensive first layer of the MAC adding process is largely

moved down into analog in this approach. This situation is the same as in (fully) digital CIM introduced above, where P_{WL} is replaced by the width of the adder tree in the digital CIM macro. A generalized discussion of trade-offs covering this factor as well as other challenges in the context of CMOS technology, focusing on SRAM, is introduced in [106] and discussed in the first chapter of the body of dissertation work (chapter 2).

As an important aside, this CIM with RRAM background section has discussed the typical case where the memory array is structured with WLs running perpendicular to BLs/SLs, and where the CIM operation works by activating several parallel WLs. In this case, the current summing occurs on the BL metal and summed current runs North/South in the array. It is possible to construct analogous RRAM CIM macros from arrays with different metal wiring formats with appropriate adjustments made. It is also possible to use this typical format differently; for example, [107] activates a single WL and applies a multi-bit input using simplified DACs on the BL and SL. Cross-column summing is handled extrinsically. The general principles for such examples remain comparable to those of the typical case.

RRAM CIM Implementations

Having set the stage by covering types of eNVM and design variants of CIM, including the different types of designs *within* current-summing CIM, this section will now discuss specific implementations of current-summing CIM with RRAM. This background section will introduce the state of the art of published macros, while the second section of the body of the document will further discuss challenges and introduce two published works showing solutions to some of these challenges [108, 109]. Since this background will be expanded later in the dissertation, this summary will serve as a short introduction.

A wide range of RRAM-based current-summing macros have been published in leading conferences in the past few years [110, 111, 107, 112, 113, 114, 115, 108, 116, 109, 117]. Designs have been presented in 65nm, 40nm, and down to 22nm in foundry CMOS logic

processes. CIM modes (the kernel width, or P_{WL}) have ranged from allowing nine parallel cells to allowing full parallelism, that is, allowing all 256 parallel WLs to be enabled for a 256WL bitcell array. Designs using the typical approach of providing inputs on the WL have typically time-serialized the application of input bits and summed externally. As described above [107] is an outlier and instead activates a single WL and applies multiple input bits simultaneously on each active BL using a DAC.

Some outlier works [113, 114, 115] use voltage-based sensing, which involves either passing a constant current through the memory cells and measuring the resulting BL voltage or pre-charging the BL to a constant value and allowing the selected cells to discharge the BL for a brief period where the WLs are strobed high, as in SRAM. The majority sense current at the front-end by clamping the BL or SL to an approximately constant voltage and measuring the total current flowing. The advantage of the clamping approach is that it prevents the BL from vastly exceeding a well-defined safe value to avoid data degradation due to read-disturb while allowing a wide voltage-domain dynamic range at the input to the SA's latching stage. Clamping also provides read-channel PVT immunity due to the use of feedback in the clamping circuit. The voltage-based approaches lack these advantages but offer improved efficiency and smoother integration by avoiding large D.C. current transients when many on-state cells are selected in parallel. These designs then need a system to recover more tightly-spaced output BL states, since the maximum allowed BL voltage could be 300mV or lower and voltage-based schemes do not in general lead to linearly-spaced output states.

The majority of works also use binary cells. This potentially improves accuracy by reducing the total number of states represented on the BL and by maximizing the use of each cell's individual dynamic range. The works partially or exclusively using multi-level cells (MLC mode), [112, 116, 117], range from 1b weights to 4b weights with fully-analog weights also proposed. One of these works, [117], observes that MLC significantly affects accuracy for the CIM operation and reserves MLC for less-significant bits in the computa-

tion. MLC mode is attractive in the all-rows-active mode, which implies degraded accuracy, because with a limited array Y-dimension or application kernel length it allows yet more states to be represented on the BL to boost efficiency; it is attractive in general because it improves overall storage density by allowing more bits to be stored in the same number of cells.

This concludes the background discussion of CIM. After the previous subsection discussed on-chip memory flavors, this subsection has provided an overview of CIM and introduced the basic idea with an optimistic example. More readily implemented design variants were then introduced, followed by a focused discussion on current-summing CIM with RRAM. Finally, recent RRAM CIM implementations were briefly introduced to set the stage for the discussion in the second part of the body of this dissertation.

1.2.4 All-on-Chip Computing with Emerging Memory Technologies

The end of the previous subsection focused on the current-summing concept made possible by eNVM technology. Importantly, however, eNVMs offer benefits beyond the future-looking, trade-off-heavy CIM design space. The key advantages are non-volatility and density. Taken together, these characteristics allow for large, very dense static on-chip weight memories that can be power-switched to avoid the large leakage penalty that might come along with large SRAM-based banks. This subsection will provide background for the all-on-chip inference accelerator concept by first introducing typical digital inference accelerators (the mainstream competition to analog CIM) that do not use emerging technologies and then surveying prior digital (non-CIM) all-on-chip inference accelerator, that do make use of eNVM, in the literature.

Digital Inference Accelerators

An in-depth architectural description of modern application-specific accelerator designs targeting ML, including ML inference, is out of scope for this dissertation which places an

increased emphasis on circuits with emerging devices. This brief review will cover a few important recent works in this area and discuss the key features and structural choices. The goal of this review is to sketch how dedicated inference acceleration is done in a general sense.

Many earlier inference accelerator architecture proposals, including [12], focused on convolutional neural network (CNN) acceleration. This work proposes the row-stationary approach in the context of a spatial array of PEs to achieve improved energy efficiency for CNN computation. Row-stationary describes a way of breaking down large convolutions into primitives that can be more efficiently mapped onto hardware. A two-step mapping procedure, analogous to the strip-mining approach used to compute with long vectors using vector processors with a fixed maximum length [4], is introduced to optimize the energy efficiency improvement. While also working with a spatial array, the tensor processing unit (TPU) design introduced in [28] is sensitive to the observation that network types beyond CNNs are important in the data-center. This TPU design features a large 256x256 spatial array of 8b MACs. Data moves systolically through the array from buffers on the periphery, ideally allowing many MACs for each piece of input data that is input into the MAC array. While the systolic structure of the array can be hidden from software from a correctness perspective, one important characteristic of such a large systolic array is that efficient software must nonetheless optimize for execution on the systolic array. For certain applications, such optimization may not be fully possible and only a fraction of the full 64kMACs throughput of the array may be utilized.

The more general problem of scalability is addressed in [118]. Compared to the previous two works, the PEs here become more capable multi-lane data processors; they each contain 8 lanes, where each lane computes 8 MACs on 8b data. Within the PE, for the typical case, weights are held constant for several cycles while input activations change for each computing cycle but are broadcast across the 8 lanes to amortize the cost. Output accumulations are, as a result, constantly written at each cycle. To achieve more general

applicability across application requirements, this work implements 16 PEs on each chip along with a global buffer and network-on-chip (NoC). Several chips can then be combined on a package in a scalable fashion to meet performance requirements, with the demonstrated package containing 36 chips. More recently, [44] implements a similar PE structure as [118], but adds support for INT4 computation. The key feature add is the support of per-vector scaled quantization. The authors observe that reduced computational precision can lead to greatly improved efficiency at the cost of accuracy. The proposed method to restore accuracy is to reduce the granularity of the quantization scale factor to the vector-level from the many-thousand-element tensor-level.

The task of optimizing neural network processing for a mobile SoC product is explored in [119]. The largest level of granularity introduced in the work is the neural processing unit (NPU) core. Each NPU contains a large 1MB scratchpad and a tensor engine for computation. The tensor engine contains four groups of four MAC arrays, each computing 8 lanes of 32-wide dot products at INT8 for a total of 4kMACs per NPU core. An external fetch and dispatch unit brings data to the MACs and orchestrates flow, including special features to minimize utilization drop when working with sparse vectors; in the typical case, input vectors are shared within each group of four MAC arrays. The authors observe that the special fetching modes addressing sparsity require the fetchers to have *more* input data bandwidth than is otherwise required by the MAC units since vectors are made more dense before MAC computing. Finally, besides these special modes, the authors describe an adder-tree-based MAC (similar to that seen in the digital CIM works) that maintains flexibility to data modes: the 256b input can be treated as 64x INT4 elements, 32x INT8, or 16x FP16. This work maintains the pattern from [118, 44] of preferring a relatively fine granularity and good flexibility for the structured MAC units while implementing powerful features in nearby controllers to map application workflows onto them.

Digital All-On-Chip Inference with eNVM

One typical feature of these digital inference accelerators is a large (MB-scale) on-chip global buffer/scratchpad and an off-chip interface to much more storage in DRAM. Off-chip data accesses can cost at least 1pJ/bit, with recently reported high-performance off-chip links costing in the range of 1.2 - 1.6pJ/bit [120, 121, 122]. This is before considering additional energy costs from DRAM itself. Further, scaling-up off chip bandwidth using a given physical design for the link, requires allocating more input-output (IO) pads for this purpose. Finally, dedicated off-chip dies for bulk weight storage can add to overall design complexity and expense.

A response to this challenge that leverages eNVM technology is to use a large amount of eNVM to store ML network weights on the accelerator die. The accelerator can then sit idle, with low leakage power, until it is externally commanded to compute an inference task. Then, the input can be passed into the accelerator and computation can occur entirely on chip. Input and intermediate activations are stored in on-chip SRAM while weights are streamed out of the eNVM. This approach requires sufficient eNVM to store all of the network weights and sufficient on-die SRAM to hold the *peak* activation data requirement. This is the all-on-chip inference approach, and it has been explored by several published works. In principle, any type of computing (including traditional digital or CIM) or eNVM (RRAM, MRAM, etc.) could be used. The focus in this background section will be on digital systems using foundry-supported eNVM, including RRAM. This subsection will introduce these works, with some added details provided in chapter 4.

Published research demonstrating all-on-chip digital inference accelerators have included 2 - 4MB of eNVM, either MRAM [123, 124] or RRAM [125, 126]. The typical implementation pattern is to use a single large bank of eNVM in one area of the chip and a large MAC unit with SRAM, that makes use of the stored weights, next to it. An outlier is [123], which associates 0.75MB of RRAM with each of four processing elements. In these works, access energy for bits of weight data in the eNVM has remained above 1pJ/bit

with eNVM bandwidth remaining well below 10GB/s even for works in more scaled nodes including 22nm planar and FDSOI CMOS. Efficient ML inference computation thus implies the use of lower-access-energy, higher-bandwidth buffers near the MAC units that amortize these costs; the buffer-based design pattern is reminiscent of the DRAM-based designs. This has left open the possibility of an inference accelerator designed to optimize eNVM accesses and avoid extra buffering while maximizing density, an idea that will be discussed in detail in the third (last) body section of this dissertation.

1.3 Thesis Introduction & Organization

In summary, this background has motivated the importance of large on-chip memory for efficiently accelerating ML inference, introduced emerging eNVM technologies along with eFlash and eDRAM with a focus on RRAM, described CIM approaches with a focus on current-summing CIM, and reviewed traditional digital accelerator architecture before summarizing all-on-chip inference works. The rest of this dissertation will follow from the structure of this background section. In chapter 2, a design space exploration covering current-summing CIM in CMOS with SRAM, from earlier in the PhD work, will be introduced. Following that, in chapter 3, two works overcoming the density and computing non-ideality challenges for CIM with RRAM will be covered. Then, chapter 4 will introduce an end-to-end project to reduce the access costs for on-chip RRAM in the context of an all-on-chip inference accelerator. This last project, which will be discussed in more detail covering the design and testing procedure, leaned on more mature, end-to-end skills that were developed throughout the earlier work. After the body chapters a brief conclusion, chapter 5, will summarize the works that were presented in this dissertation.

CHAPTER 2
ANALYSIS OF CIM WITH SRAM

This chapter is the first of three body sections in the dissertation. In this chapter, we discuss the design space of CIM with standard SRAM in logic CMOS processes. This analysis work will then be followed by design and implementation work in chapter 3 and chapter 4. In this chapter, first, we will introduce a brief taxonomy using a set of published works in order to identify discriminating characteristics. We then focus on a subset, the *direct* (current-summing) CIM with SRAM systems and develop circuit analysis while discussing potential issues. Finally, we compare CIM to an all-digital approach under strong assumptions to draw a few conclusions.

This chapter is based on analysis work and ultimately a transactions article that was submitted for publication early in the PhD work [106]. The work consists of a brief survey, circuit analysis, and description and results of a simplified modeling approach comparing a system using SRAM with CIM to a system using standard SRAM. A comment about this work as-published is that the importance of charge-domain and digital CIM, especially with SRAM, is understated as this work places a focus on current-summing CIM. Recent publications have shown that these two techniques will likely play a key role in future developments around SRAM CIM. Some commentary and adjustments are also made here to adjust to the changing CIM landscape. Also, the full analysis [106] will not be fully reviewed here. Only sections that are important to the core flow of working from assumptions toward conclusions about current-sensing CIM with SRAM will be included.

2.1 Introduction

Reported CIM with SRAM implementations have shown good energy efficiency. These systems have also focused on lower-precision math and in particular have targeted applica-

Table 2.1: CIM-specific abbreviations used in this chapter.

Symbol	Definition
B_x	MAC input vector precision (in bits).
B_w	MAC weight vector precision (in bits).
P_x	# of input vector bits applied on a single WL per read cycle.
P_{WL}	# of parallel WLs allowed per read cycle.
P	# of states possible on each BL minus 1; $P_{WL} \times (2^{P_x}$ - 1).
M_s	# of equivalent SRAM single-cell reads required to emulate the memory accesses of the CIM system.

tions with loosened compute-level accuracy requirements. From this, it is not immediately clear how exactly the efficiency and throughput advantages of CIM emerge, which means the generalizability of these advantages, beyond the specific demonstrated examples, is similarly unclear. Since not all MAC-heavy workloads can tolerate imperfect accuracy and reduced precision (such as INT4 or INT8), we will use high-level models along with circuit modeling and simulation to examine the efficiency gains and penalties associated with CIM in SRAM arrays. Models which are needed to make informative statements about CIM will be developed, and the issues of accuracy loss and area increase will be contextualized in the insights and outlooks subsections. A broader overall discussion of the circuit models and accuracy loss issues may be found in [106].

Some CIM-specific abbreviations are introduced in this chapter, a subset of which were previously introduced in chapter 1. A core set of these is shown in Table 2.1. Unlike in chapter 1, P_x is now binary-weighted in that it refers to the number of input bits applied via the WL, not the number of possible states applied via the WL.

2.2 Brief Taxonomy

This first section will develop an abbreviated taxonomy using a subset of published works in the SRAM CIM space. The purpose of this taxonomy is to make it easier to describe specific kinds of works, rather than to be truly exhaustive. For this chapter, an (analog) CIM system is *a system which enables the values of multiple memory cells to be combined in the*

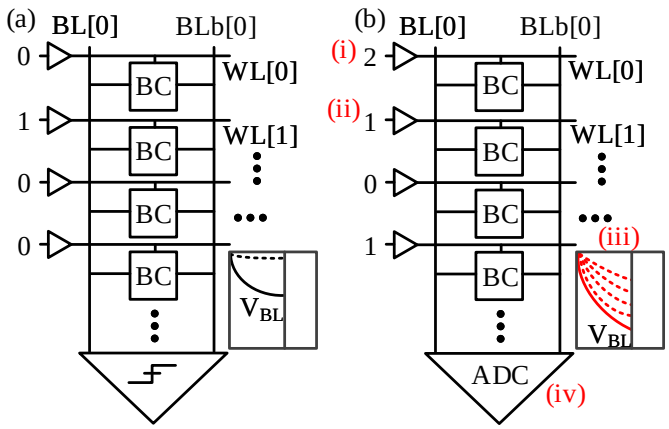


Figure 2.1: (a) Traditional and (b) CIM memory systems shown with 1:1 column multiplexing (unlikely) for simplicity. Four important degrees of freedom are introduced in the CIM system: (i) there may be more than two allowed input (WL) states, (ii) multiple WLs may be enabled at once, (iii) there may be more than two allowed output (BL) states, and (iv) these multiple states may require a more complex ADC than a 1-bit sense amplifier. Feature (ii) is, in the general case, the hallmark feature of CIM. In this figure, bitcell is abbreviated as BC.

analog domain then read out, as a single value, into the digital domain. As described in the background section, the number of output states may be further increased by allowing more than two WL states. In outlier designs, it is possible for only one cell to be active per column (with summing occuring *between* BLs). Still, multiple BL states are always present in CIM due to multiple states being represented at the input and/or in the cell. In CIM with SRAM, multiple states per-cell (multi-level cell, MLC) are less likely since the classic SRAM cell's inverter-loop storage mechanism is natively compatible with a single stored bit. To introduce SRAM CIM, four important differences between CIM and traditional memory systems are outlined in Figure 2.1.

To start off the taxonomy, a distinction can be drawn between systems which perform readout by *directly* measuring bitline (BL) voltage due to a pull-down (PD) operating step (during which a current proportional to the number on-state selected SRAM cells is pulled on the BL) and systems which measure the BL state *indirectly* by, briefly, using the BL to control a separate system which performs an analog computing step. This second category can include charge-domain or digital CIM as described in the background section. These

two methods are described below in the context of some published works.

2.2.1 The Direct Systems

The 'direct' systems [127, 128, 129, 130] operate similarly to traditional binary-BL SRAM: a single read-step produces the analog value on the BL which is then quantized. The read PD circuit formed in an SRAM cell when its WL is selected can be viewed as a variable current source, and the read step can therefore be viewed as a current to voltage conversion. This conversion may take one of two forms, which further classifies the direct systems (Figure 2.2). In resistive-capacitive-pull-down (RC-PD) macros, which operate almost identically to traditional SRAM, the BL is first pre-charged (*e.g.* to a rail such as VDD) after which a set of WLs is enabled based on the input vector values. The on-state selected cells create a PD current proportional to the MAC output value which causes the BL to slew downward. After a period of time, the slewing is terminated, for example, by disabling the access devices, and the value of the BL is quantized. In resistive-dividing or current-mode macros, a pull-up (PU) device flows an active current on the BL while the selected cells are activated. In the simplest case, the relationship of the large-signal resistance of the PU device to that of the PD circuit formed by the on-state selected cells creates a steady-state output voltage on the BL. In the case of current sensing, the voltage across the cells is pinned to a clamping voltage and the resulting current is measured and converted into the output value.

2.2.2 The Indirect Systems

These *direct* systems present a set of challenges including (1) reduced and variable state separation leading to restricted sensing margins and nontrivial reference generation, (2) inconsistent (noisy) state placement due to poorly matched SRAM PD circuits, and (3) limited and non-ideal mechanisms for multi-bit inputs and no native compatibility with analog-domain multi-bit weights. Note that (1) can be mitigated with careful readout mech-

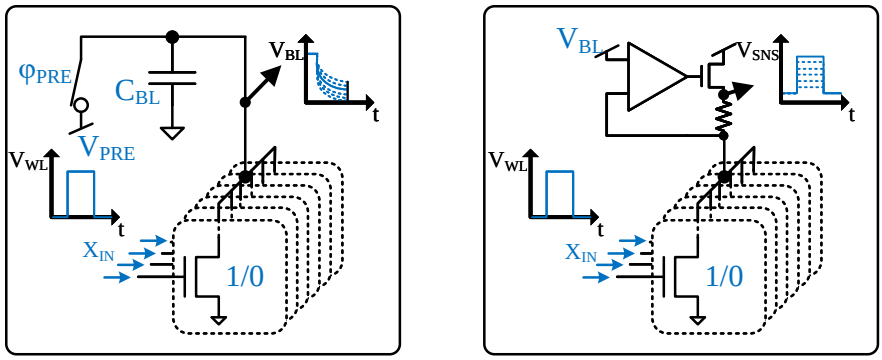


Figure 2.2: Symbolic representations of the (a) resistive-capacitive pull-down (RC-PD) and (b) resistive-dividing (or current-sensing) flavors of *direct* CIM systems.

anism design, especially in the case of current-sensing approaches.

These challenges may be avoided by shifting the analog computing operation away from the SRAM storage cells and onto a special-purpose summing circuit block. These special-purpose analog add-ons are characteristic of what are called *indirect* systems in this chapter and are typified by well-matched metal-oxide-metal (MoM) capacitors or other types of capacitors [131, 132, 133], shared transistors with improved matching characteristics [134, 135, 136, 137], or time-domain computing [138]. The goal is to decouple the matching and linearity behavior of the analog compute from the less-ideal density-oriented storage cells. Compared to the direct systems, the readout process for these indirect systems is less like a single SRAM readout step. The read operation instead typically involves a first step in which the memory cell values are extracted, then a follow-up step when these values are used to control the summing circuitry. The timing cost of these steps can be traded for an area overhead by more devices to the memory cells. This characteristic makes these systems less analogous to traditional memories. For this reason, and to maintain a clearer focus, in this chapter we analyze *direct* (current-summing) CIM with SRAM systems.

2.3 Comparative Approach

The energy cost of modern MAC workloads consists of digital logic and memory access components. Memory access costs have become a subject of focus in recent years [2, 3]. CIM has the potential to address energy costs, as it proposes to offer:

1. *Higher effective memory bandwidth* for improved memory access energy, and
2. *Reduced digital logic workload* due to analog-domain computing.

From this observation, in this chapter, we develop separate comparisons for (a) energy cost per effective bit accessed (b) overall digital workload under traditional vs. CIM regimes. In this section, we will first frame memory-access energy before estimating the digital workload savings due to CIM.

Another observation is that the magnitude of CIM's potential advantages in both areas is proportional to the allowed parallelism (*P*, *i.e.* the number of output states allowed to be accumulated in analog). Simply put, if there were no associated penalties (such as increased ADC complexity) and no application restrictions, turning on more WLs in parallel would always provide better performance. For this reason, separate analysis in this work will relate energy penalties or the severity of area, robustness, and accuracy concerns to *P*.

2.3.1 Effective Memory Bandwidth

The memory-bandwidth-improvement proposition for analog CIM holds that there is an advantage to reading out the sum of the values of a subset of memory cells rather than area- or time-serially reading the weight data and performing the MACs entirely in digital. Using the symbols from Table 2.1, a single read operation is used to simultaneously read P_{WL} weight bits (on a single BL). The sum across element-wise products in a B_x- by B_w-bit precision MAC with P_{WL} elements can be seen in some sense as a lossy compression operation in the weights; a group of P_{WL} same-binary-position weight bits need only be represented by at most $\log_2 P$ bits once added together.

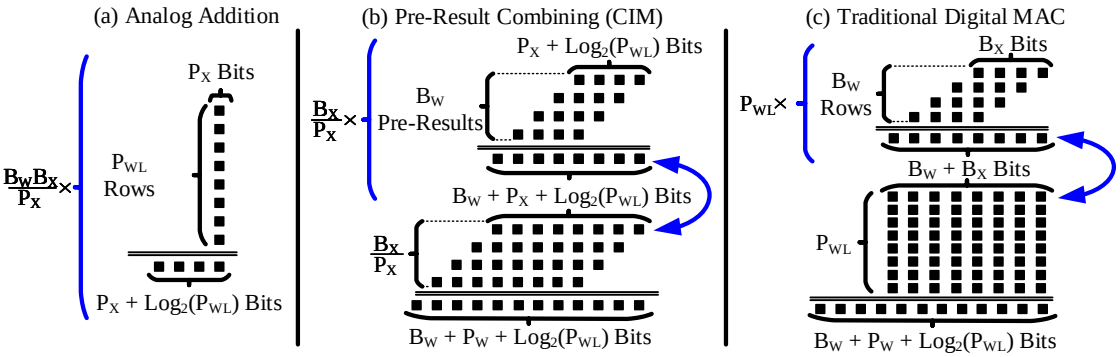


Figure 2.3: Each bit shown as a square for (a) the analog addition performed on the BL in a CIM system, (b) the pre-result combining workload needed by a CIM system, and (c) the same MAC performed completely in digital. The shown dots correspond to an 8-wide MAC with 4-bit input and weight precisions.

This precision-reduction operation applies a penalty in the form of input bit striping: multiplying the same weight against several inputs or input bits requires additional read operations, since the memory output is now specific to the input vector. To rehash from the background section, if analog multi-bit inputs are used, then the input-bit-striping penalty is divided by the precision available on each WL. Here, we will phrase the 'striping cost' in terms of the overall advantage of CIM. In net, the number (M_s) of equivalent SRAM single-cell reads required to emulate the CIM system is shown in Equation 2.1, where P_x is the analog-hardware-implemented input precision. In the published paper, P_x was the number-of-bits. It has been changed to number-of-states to match the background section.

$$M_s = \frac{P_{WL} \times \log_2(P_x)}{B_x} \tag{2.1}$$

The number of weight bits is not included here since, even in the case of a traditional digital system, each weight bit has to be read out separately. The point is that for CIM, unless multi-bit WLs are used, the same weight bits have to be repeatedly read out for each bit of precision in the input vector.

2.3.2 Digital Workload

The most obvious benefit of performing addition in analog is to reduce the digital MAC workload, namely multiplying, bit-shifting, and summing. This advantage is softened by two factors: first, implementation realities (*e.g.* pertaining to the ADC) may add back in some of the saved compute. For example, typical Flash ADCs require thermometer-to-binary conversion, whose implementation may approximately be viewed as a P-wide one-bit adder (the Flash ADC may also require bubble error correction [139]). If the overall ADC implementation is not actually less expensive than the adder it replaces, then clearly the digital-workload side of the benefit of CIM is lost. Second, the limited (analog-domain) input and stored weight precision of CIM systems may necessitate striped inputs and weights, which implies an added pre-result combining step (Figure 2.3 (b)) to produce the MAC output. The first issue is not explicitly accounted for in this work; instead, this added digital energy is included in the literature-derived constant ADC energy assumption used in the next section. The second issue, the saved digital compute cost relative to the cost for the remaining partial-sum arithmetic, is the focus of the remainder of this section.

We are interested in the *fraction of compute that is saved* by doing the first layer of the accumulation in the analog domain. To clarify the thought experiment, this will be the relative amount of computing work that no longer has to be done if the computing done in analog were free. Figure 2.3 breaks down the components of the CIM (a, b) and traditional (c) adding workloads along feasible lines. For example, (b) assumes the input is applied across all weight bits and that these pre-results are immediately combined, then barrel shifted and accumulated onto the sum corresponding to the bit-striped input. To translate this dot-diagram into something useful for analysis, two *ad hoc* methods will be used to compare the energy of the traditional system's digital workload (Figure 2.3(b)) to that of the CIM system's digital workload (Figure 2.3(c)). The simplest method is to consider the total digital MAC workload as consisting of Figure 2.3(a) followed by Figure 2.3(b). The fraction of saved compute can then be computed by dividing (a) by (a) + (b). To

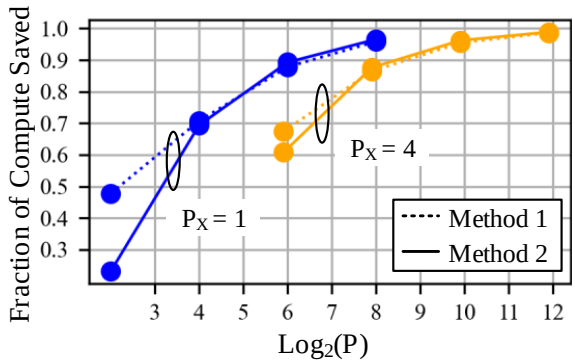


Figure 2.4: The fraction of MAC computing workload saved by using CIM in the case of $P_x = 1$ (one-bit WLs, with up to P wordlines active at once) and $P_x = 4$ (four-bit WLs). The first method (Method 1) counts total bits to be shift-added while the second method (Method 2) counts approximate gates needed to do this summing. For this example, the precision is set to $B_x = B_w = 8$. These curves are not very sensitive to precision.

approximate the computing cost of these sums, we observe that both (a) and (b) correspond *in shape* to the partial product reduction of a standard multiplier; typically, an algorithmic method (following from [140, 141]) will be employed to construct adder trees to reduce these sums. Counting the bits to be combined therefore provides a compact, although approximate, representation of complexity. Omitting algebra, the ratio (ν) of this bit count for the saved CIM compute to that of the total compute is given by Equation 2.2:

$$\nu = \frac{P_{WL}}{P_{WL} + 1 + \frac{\log P_{WL}}{P_x} + \frac{1}{P_x} + \frac{1}{B_w} + \frac{\log P_{WL}}{P_x \times B_w}} \tag{2.2}$$

More detailed estimation is possible by modeling gate-level implementations of (b) and (c) and counting gates *each time they are used* to estimate energy. This leans on further assumptions: we assume that the adders are formatted as in a fast multiplier with some specific kind of partial product reduction step followed by a ripple-carry adder. Of course, additional carry look-ahead logic may still be important in a practical implementation. Registers are not modeled, and a simple greedy algorithm for forming adder trees is used where appropriate. The value of interest is ((c) - (b)) / (c).

The two methods for approximating the digital compute savings due to CIM are plotted

Table 2.2: Parameters for the CIM SRAM systems and the reference SRAM that are modeled for the comparison in this chapter.

System	SRAM	(a)	(b)	(c)	(d)
Scheme	SE	SE Direct BL-PD			
Technology	28nm CMOS				
SRAM Cell	6T 0.32um^2 Non-Push Minimum 0.9V				
V_{WL}	0.5V				
Array Dim.	512x128 (Physically Square)				
BL Mux. Ratio	8:1			1:1	8:1
B_x	Same as CIM	8	4	4	1
P_x	1			4	1
P_{WL}	1	P_{WL}	P_{WL}	P_{WL}	P_{WL}
ADC States	2	$P+1$	64	16	2
ADC Margin	6σ	3σ	1σ	0.075σ	1σ

in Figure 2.4 for $B_x = B_w = 8$ (example case) across increasing CIM parallelism, P. The two methods generally agree, and the models show that the analog-domain compute is a dominant portion of overall compute for large parallelism (wide MAC kernels, large P).

This also confirms an important result, namely that supporting more states on the WL is less favorable (than enabling more WLs, that is, using a wider MAC kernel) for reducing digital adding workload due to the linear benefit to the striping penalty at exponential cost in terms of states represented on the BL. Using a wider kernel linearly increases states represented on the BL while providing a linear benefit to throughput/efficiency. This concludes the digital workload analysis. For this work, it was assumed that a synthesized area and energy analysis of this digital workload is not needed since it suffices to estimate the *relative fraction* of digital workload that is saved. The preference is toward maintaining generality.

2.4 Comparison to Traditional Systems

This section will construct a comparison between current-summing CIM with SRAM systems and a hypothetical SRAM that replicates the memory accesses. Importantly, the com-

parison is not holistic: it compares memory-access behavior only. The CIM system, as discussed above, also moves some (and potentially most of) the digital MAC workload into the analog domain. This section is not intended as an analysis of analog computing vs. digital computing; instead, the focus is on whether accessing more than one row in parallel in net reduces the cost of getting bits out of a memory array.

Two principles enable strong conclusions based on the energy comparison examples that will be shown in this section. First, all analysis is stated in terms of the *ratio* of energy used by a CIM system to that used by the corresponding SRAM system. This enables generalized analysis of trends from circuit fundamentals while ensuring that any comparison is intrinsically apples-to-apples: both systems (CIM and baseline) use the same array with the same capacitances, have access to the same SA with the same scaling trend, and so on. As discussed above, the basis of this ratio-driven analysis is the choice to model total MAC energy as consisting of a digital-logic component added to a memory-access component. The ratio of the former to the latter in a traditional implementation is a function of technology generation, application, and design choices.

A second constraint made on the analysis in this work, and perhaps one that is more important, is that only *direct*, RC-PD CIM systems are analyzed due to the practical advantages in area, energy, and state separation that such systems offer as described above. This leads to a cleaner comparison: the mechanism of action for RC-PD CIM systems and classic SRAM is the most similar out of all CIM flavors.

Here, CIM designs are compared to an SRAM design with the same dimensions which uses SE read. To this end, a 6T logic-rule $0.320\mathrm{um}^2$ SRAM cell layout was created to generate extracted SRAM array parasitics. A 128x128 array is used to extract capacitances while avoiding boundary effects, then scaled numerically. Transistors without push rules are used to align with the models simulated throughout this work. 512x128 is chosen for the presented results because it is a physically-square array, meaning the BL and WL capacitance are weighted equally by construction. The ADC energy model has two components: first,

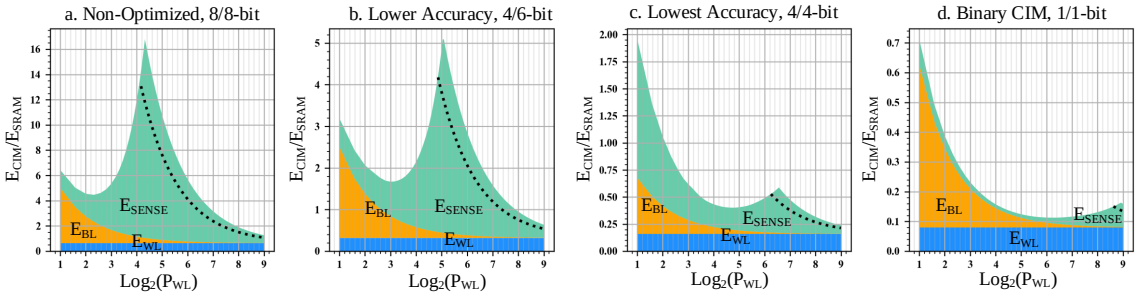


Figure 2.5: Predicted CIM performance results using the extracted models. Parameters for (a) - (d) are defined in Table 2.2. Array capacitances and SA energy are extracted from layout and schematic simulations, respectively.

SA (and Flash ADC) energy figures are modeled using schematic-level voltage comparator simulations following a standard design [142] across a range of input-pair sizes; as the number of output states is increased, Flash ADC energy eventually exceeds the constant-energy ADC models, and ADC energy is clipped at 440fJ (300fJ model shown in dotted line). The parameters for the comparison are given in Table 2.2.

The relationship between BL capacitance and SA energy has an effect on the active-energy-optimal BL PD time in a standard SRAM array. That is, more PD time creates a larger sensing margin and allows for a smaller, lower-energy SA. Therefore, to strike a fair comparison between SRAM and CIM SRAM, the SRAM SA sizing is set so that the chosen SA model achieves minimum total energy while satisfying 6σ of design margin based on the nominal threshold variation of the SA input transistor pair. Parameters for these comparative models are given in Table II.

A few more assumptions simplify the modeling process. Bits and memory cells are assumed to be binomial distributed with a 50% probability of being either one or zero, rather than following any specific application distribution. Also, auxiliary energy costs (*e.g.* MUX reconfiguration) are not modeled and, importantly, data movement costs are not modeled (as if the MAC logic were directly abutted to the SRAM).

2.4.1 Results

Modeling results are presented in Figure 2.5 as ratios of memory access energy across increasing WL parallelism P_{WL}. As mentioned, these examples have omitted the effects of digital workload savings which were shown to be plausible in subsection 2.3.2. All the shown examples exhibit an energy peak at which the constant-energy ADC model outperforms the scaled Flash ADC. The pattern is that the total energy used for the Flash ADC gets worse and worse as sensing margin decreases until the alternate ADC style is preferred, leading to an abrupt decreasing trend in per-bit sensing cost. Figure 2.5 (a) illustrates the most pessimistic case for CIM: the MAC is computed in full 8-bit precision, and the ADC is designed to recover the full precision ($1 + P$ bits) on the BL while satisfying 3σ of design margin. Loosening these three requirements yields improvements and the potential for energy-efficient designs. Figure 2.5 (b) demonstrates 4-bit inputs, which halves the energy ratio relative to (a), and only requires the ADC to achieve 6 bit precision with 1σ of design margin. This last change, reducing the design margin requirement, improves Flash ADC energy scaling and pushes the extrema toward higher P_{WL} although CIM memory accesses remain more expensive than SRAM out to $P_{WL} \approx 2^8$.

One recent work ([143]) showed high efficiency (> 351 TOPS/W) using RC-PD. It is challenging to contextualize the performance of a 7nm 4-bit MAC engine, but this work provides the framework for a higher-efficiency example. Three changes allow CIM to consistently outperform SRAM (Figure 2.5 (c)): the ADC precision is reduced to the input precision ($P_{OUT} = 15$), 4 bits are applied on each WL, and most importantly the design margin is aggressively reduced to 0.075σ [1]. This allows $> 50\%$ reduction in memory access energy across a range of P_{WL}. As in [143], the 4-bit WLs have been modeled as successive (thermometer-coded) pulses applied on the WL, and reducing the BL MUX ratio for this example is critical to dilute the increased WL energy. Energy efficiency achieves its logical

[1] Extracted from [143] for the case of 2^{10} states on the BL and 700mV dynamic range. [143] actually applies many more than 2^{10} states to the BLs, as binary weighted versions of 4 columns are combined before sensing.

maximum with a binary CIM example (Figure 2.5 (d)): even if the ADC design margin is tightened to 1σ, and the BL MUX ratio is reduced back to 8:1, the lack of input striping penalty and quadratic (vs. cubic) Flash scaling allow for very efficient memory accesses.

2.4.2 Model Limitations & Design Insights

The models in this work support certain guidelines about CIM system design. State separation in RC-PD systems can easily be below 10mV when $P \geq 2^6$; this sets a stringent requirement for noise coupling onto the BL (not simulated in this work). Accurate readout is further limited by cell current variation, WL timing errors, and ADC limitations. These present complex interactions: since Flash ADC energy is inversely quadratic in state separation, and noise-induced accuracy issues are reduced with larger state separation, wide voltage swings on the BL are desired. SRAM cells are not natively robust to full-swing reads, and sizing is largely ineffective at correcting read margin to the extent needed. Reducing V_{WL} improves read margin, reduces the effect of absolute timing errors by increasing the BL pulldown time constant, and improves state separation. However, decreased V_{WL} linearly to super-linearly increases cell current offset. If V_{WL} is reduced as far as allowed by design accuracy requirements and read margin, channel resistance, and/or state separation requirements are not met, the logical next step is to increase the length of the read pass-device which improves read margin, channel resistance, and cell mismatch at the cost of write margin.

Generally, single-ended implementations are preferred since they offer better read margin and reduced BL energy. That said, for CIM, BL energy is less relevant for large P as per-state BL swing is necessarily reduced. To clarify, BL energy is capped at $C_{BL}V_{DD}^2$ while other energy consumers, such as the ADC, can increase aggressively as more states are represented on the BL. This motivates taller arrays with more C_{BL} which improves density, increases the PD time constant (reduced sensitivity to timing offsets), potentially reduces coupling noise, and under some specific design conditions could even offer a lower-

noise sampling capacitor to benefit a high-precision ADC. This also may allow current sensing systems to perform well, since they trade added BL energy for reduced sensitivity to absolute BL timing errors. At a higher abstraction level, Equation 2.1 shows that CIM performance is proportional to P_{WL} and P_x and inversely proportional to B_x due to bit striping. P_{WL} is preferred to P_X since P_x exponentially increases the number of states on the BL. Similarly, P_x shows inferior trends to P_{WL} in terms of saved compute vs. states on BL. P_x is valuable if application or array parameters restrict P_{WL}; without specific reason otherwise, it is better to scale P_{WL}.

2.5 Conclusion

From Figure 2.5 and the motivating analysis, a few statements can be made about how CIM systems can outperform SRAM under the right circumstances. First, if a traditional system is heavily biased toward compute energy, especially at lower precisions, then CIM (analog compute) may improve efficiency by reducing digital computing workload at similar memory access costs. Second, at higher (input) precisions, RC-PD systems struggle to overcome the linear penalty of striped input bits (Figure 2.5(a)) leading to degraded energy efficiency. Third, if a CIM system is to directly achieve improved memory access costs, the following prescription defines how the energy savings emerge: (1) the input-striping penalty applies a multiplier to energy that must be overcome; (2) increasing P through P_{WL} and potentially P_x eventually reduces per-memory-cell E_{BL}, trading away sensing margin; (3) the sensing system (ADC) accommodates the reduced sensing margin without imposing an unmanageable energy cost or at reduced per-state memory cost. Since, per-state, Flash ADCs are characterized by super-linear costs and CIM ADC costs are only linearly amortized, achieving (3) *implies* reduced ADC design criteria (accuracy, precision) *or* a design which places enough states on the BL (defined here by parameter P) to amortize a high-precision high-efficiency ADC.

Accuracy loss in these systems relates directly (and potentially super-linearly) with en-

ergy efficiency. The accuracy characteristics of the ADC, and overall system, are core to understanding the merit of a CIM implementation - accuracy notwithstanding, it is possible to place arbitrary numbers of states on the BL for arbitrary efficiency. Works like [144, 145], which relate analog computing errors to viability for larger networks (or other such real-world tasks) are therefore important for designing and assessing this kind of CIM system. Representing fewer states on the BL typically corresponds to better accuracy due to less accumulating error and less sensitivity to noise, and potentially enables better generality across applications (which might have lower-dimension vectors). Moving forward, we expect designers to continue to focus on CIM systems with ADC designs that are energy efficient across a lower precision range (*e.g.* 2-7 bits), like the medium-precision successive-approximation register (SAR) designs recently shown in some CIM SRAM works [137, 134, 135, 133, 134]. To avoid poor array efficiency and a resulting potential increase in expensive off-chip data movement, successful designs will likely demonstrate novel ADC architectures with smaller footprints.

CHAPTER 3
CIM WITH RRAM IMPLEMENTATIONS

This chapter will discuss the design and implementation of two RRAM CIM macros [108, 109] and their context in the existing literature. This builds on the previous chapter, chapter 2, in two ways: the discussion now includes eNVM (namely, RRAM), and the subject is RRAM CIM macros that were built and tested in silicon rather than analysis. RRAM CIM will be introduced, followed by background discussion contextualized where appropriate by review of prior work. Then, the two published designs and measurements will be detailed, including their context in a larger system-level design. Finally, the overall results will be summarized in a conclusion.

3.1 Introduction

Both of the works that will be discussed implement current-summing CIM with RRAM. They use a 1T1R RRAM memory array in a 40nm foundry process, as in the foundry-reported non-CIM RRAM array for this technology [59]. To improve array density while providing enough metal to reduce resistivity on the BL and SL, one SL is shared for every two BLs. In typical memory macro designs, part of the addressing bits control a column MUX that connects each readout sensing circuit or each write circuit to one of C columns, where C might be 8, 16, 32, or some other value so that the MUX ratio is greater than one. For eNVMs like RRAM, the column pitch can be narrow to allow a small cell size, making higher MUX ratios more practical. This means the typical use case would not activate cells on adjacent BLs for read or write simultaneously. So, by sharing the SL, the cross-sectional amount of SL metal per-BL can remain constant, or increase due to less empty space, while the current density in the SL, and therefore the negative affects due to SL resistance, are reduced. The cell size, as previously reported, is $53F^2$.

This bare RRAM array was provided by the foundry, including only the cells and metallization. All of the peripheral circuitry including complete read and write sub-circuits were implemented as part of this CIM macro design effort. Therefore, while some of the design work was focused on CIM-specific features, by necessity a large part of the work was dedicated to building all of the peripheral components needed to read and write RRAM cells. This was especially true for the first macro: once the first macro design was implemented and tested, pieces such as the write circuit could be pasted into subsequent designs and then altered, from that starting point, to provide specific improvements. As an example, pitch-matched circuits like thick-oxide column write MUXes created difficult layout tasks due to widened design rules for metal lines and devices at HV. Having these finished layouts in-hand made followup work substantially easier.

The two designs follow the typical current-summing CIM with RRAM design pattern. Both designs allow multiple WLs to be activated in parallel for each memory macro. The WLs are each driven with a single bit, to VSS or VDD, and therefore each represent one bit from an element in the input vector. The readout circuit is the primary peripheral component that is tuned specifically to support CIM. The readout circuit in both macros is responsible for clamping the voltage across the cells in the column so that a current flows that is proportional to the number of LRS selected cells. This current is converted to a voltage, linearly, using a reference process resistor. This voltage is then input to an ADC, which converts the sensed current into the digital output value for the column. A digital wrapper provides bit inputs to the WL drivers, manages the write and read modes and timing, and collects the raw ADC results.

The two macros have separate design goals. The first macro, [108], was designed to reduce the area overheads of CIM while providing a linearized readout of the conductance of selected cells in each column. This macro was a follow-up to prior work, [113]. The second macro, [109], benefited from more knowledge and experience during the design process and was designed to address specific challenges of CIM with RRAM, including IR

drop, off-state current, and column mismatch. Both macros were implemented as unit parts of largeer systems [146, 147] which made design more difficult by requiring the designs to be rectangular sub-module blocks with well-defined mostly-digital interfacing. This also made testing more difficult by making it hard to isolate single macros for measurement and, for example, provide clean voltage rails.

3.2 Background and Challenges

These projects faced two kinds of challenges: design challenges due to attempting to implement current-summing CIM with improvements relative to previously published work, and implementation challenges arising from attempting to build a modular macro before a strict tape-out deadline without a prior peer work for a truly modular macro. Many aspects of the design process, particularly the sheer number and variety of peripheral elements needed to make the first macro work, pushed the intellectual envelope early in the PhD experience and led to continuous learning throughout the process of making key design decisions with the best information available at the time. These lessons led to a sharper design in the second CIM iteration and additionally aided in the design of a robust, non-CIM third macro as part of the final end-to-end digital system design which is discussed in chapter 4.

This section will focus on design challenges for RRAM CIM with limited discussion of other published work where appropriate to understand and contextualize these challenges. The challenges addressed by these two macro designs are: peripheral overheads that reduce memory density, the need to support various operations including measuring RRAM cells for research purposes (flexibility), compatibility with integration into a larger digital system (modularity), channel-to-channel variation, IR drop along the BL and SL, and cell resistance variation.

3.2.1 Area Overheads

Area overheads for RRAM CIM arise from two places. First, using an eNVM technology RRAM as compared to SRAM requires high voltages for write and therefore requires level shifters (LSs) and isolation devices between the array and the read circuitry. Isolation devices protect the read circuits from the high voltages that can be present on the array metal (BL, WL, SL) during write operations. Second, using CIM requires a trans-impedance amplifier (TIA) or other type of front-end followed by an ADC. Compared to a binary current-sensing front-end with local reference [62], this approach will typically require more area. At the time of the design of the first macro, CIM macro area was not typically reported. As an example of how this has been a less significant design priority in prior work, the direct predecessor work [113], reported an area of 0.437mm^2 for the 64kb macro, or about 0.143Mb/mm^2 while using a technology with a raw cell density of approximately 8Mb/mm^2.

3.2.2 Flexibility

The flexibility goal is both for application applicability as well as practicality. Practically, the design involves using an emerging eNVM technology with limited characterization data available from the foundry. This made it important to be able to tweak aspects like gain and the number of parallel cells being read during testing, especially for the first macro, to accommodate the actual cell behavior during macro characterization. It was also important to support tangential work that would involve memory cell characterization and experimentation, which would require features like reading a single cell at higher gain. These details were not typically reported for the available published work, making them parts of the high-level design process for the macro.

3.2.3 Modularity

Modularity here refers to the ability for the macro to act as a module as part of a larger design. This is in contrast to prior work, which were typically reported as large chip areas including both the macro and test-keys without detailed area breakdown. Unlike these typical macro-centered test-chips designed specifically to allow the macro to be fully characterized, these macros would need a limited interface: very few power rails, few (or, preferably, zero) analog signals to the off-chip test-bench, and a defined interface that would be compatible with the interface RTL written by a lab colleague.

3.2.4 Channel Variation

The first three challenges are more qualitative (excepting area overheads) or general and were very relevant for the design of the first macro. For the second macro, these remained valuable but a set of more quantitative challenges relating to the CIM readout process became the focus. These challenges are summarized in Figure 3.1.

The first of these is channel variation, which comes in two forms. First, and typically most challenging for CIM is channel-to-channel gain variation. This applies a constant scalar multiplier for each channel in relating to the amount of gain error; that is, the readout result is scaled by $1 + \delta_g$ for some additive gain error δ_g. This can arise from variations in the clamping voltage, variations in the ADC for some ADC topologies, and variations in the sensing resistor in the TIA.

The second form is channel-to-channel offset error, where each channel's sensed voltage is effectively shifted by a signal-independent offset δ_o. This can arise in the voltage domain, due to the ADC. Offsets could also occur in some TIA topologies, especially if the sensed output voltage is referenced to a virtual ground node that could vary or if the baseline (0-cell-current) value is set by a bias current that could vary. Several RRAM CIM works have described offset cancellation in the SA [110, 111, 107, 115] to improve sensing margin due to the tight current-domain spacing of output states for current-sensing CIM.

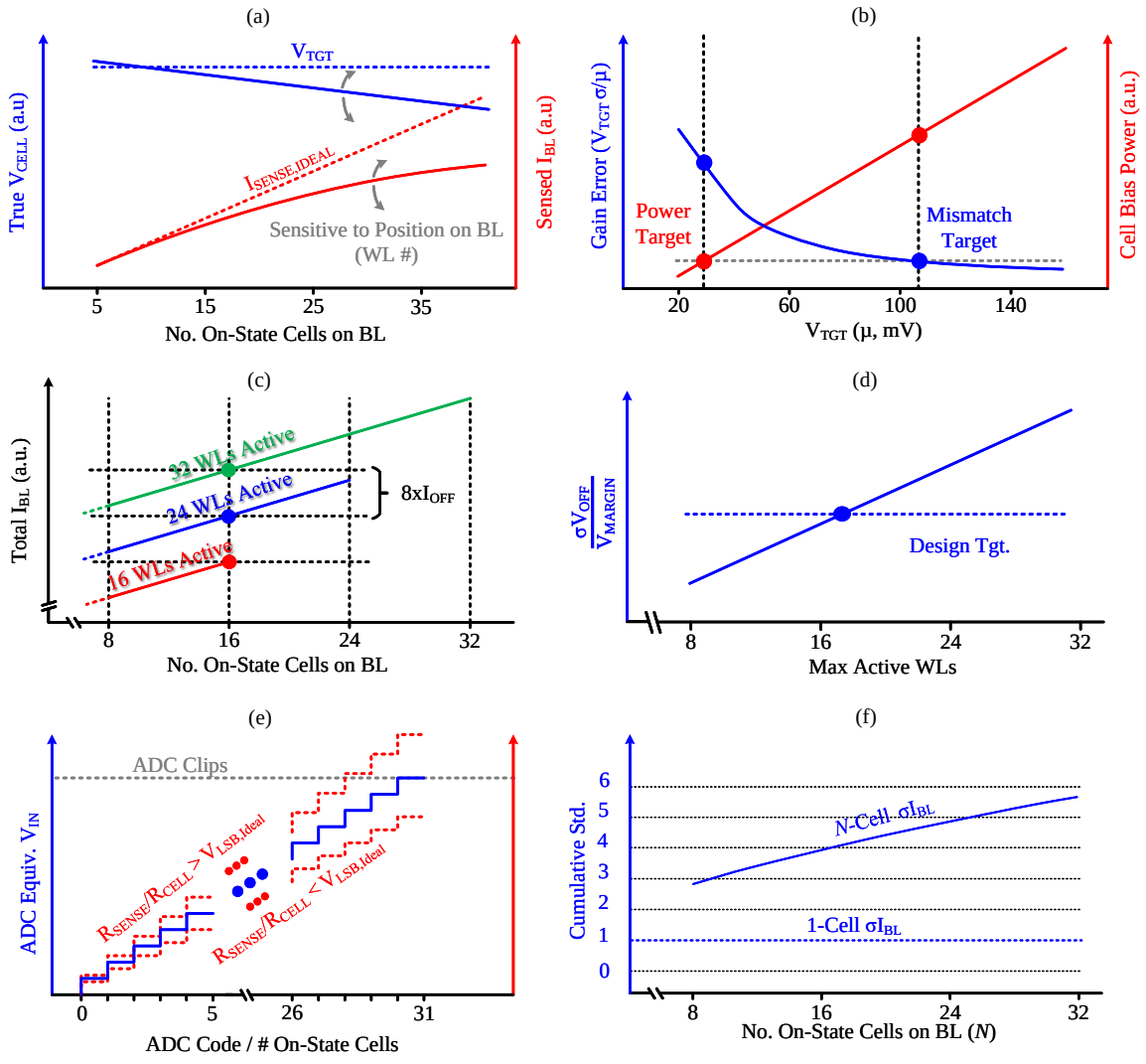


Figure 3.1: Six challenges for current-summing CIM with foundry RRAM: (a) IR drop along the BL, SL, and in peripherals including MUXes, (b) channel gain mismatch, (c) off-state current, (d) channel ADC offset, (e) global cell current variation, and (f) local (random) cell current variation.

3.2.5 IR Drop

IR drop applies to works that turn on many cells at arbitrary positions on the BL/SL. Consider the case of IR drop along the BL. Working in the resistance domain, consider that nominal LRS resistance could be as low as 2.5kΩ. If four such cells are activated in parallel, the equivalent parallel resistance will be approximately 600Ω. If the BL resistance is 60Ω, which could be reasonable for a scaled array in a modern process, the total load seen by the current sensing circuit is 660Ω, a full 10% higher than the nominal load due to selected LRS cells. The name 'IR drop' results from the observation that the cell current has *dropped* $I_{CELL} \times 60\Omega$ volts across the parasitic BL resistance before reaching the cells. Ideally, all cells would be biased exactly at the clamping voltage. Instead, the parasitic voltage drop means the cells are biased at some signal-dependent voltage below the nominal clamping voltage. IR drop occurs in any resistive component in series with the memory cells between the read circuit and VSS, including the MUXes, BL, SL, and SL VSS tie.

Clearly, with many more parallel LRS possible even in 9-WL CIM modes, IR drop can lead to significant errors for the higher-current states. IR drop limits the linearity of the readout due to the resistor-divider effect; the series sum of the parasitic resistances and the net cell resistances saturates toward just the parasitic resistances as the cell resistances are reduced by having with more on-state cells in parallel. IR drop can be viewed as a type of gain error. IR drop is made more challenging by the fact that the amount of IR drop seen by any cell depends on its position along the BL/SL, leading to a changing sensed current as a constant test pattern is shifted down the BL/SL. Works that only enable a single cell per column [107] are partially immune from IR drop issues since less current passes through the scaled BL/SL lines of the foundry RRAM array.

3.2.6 Off-State Current

This challenge could be rephrased as *limited memory window due to low on/off ratio*. This can be simplified by viewing the on-state current as an incremental current on top of the

off-state current. That is, *every* selected cell contributes I_{OFF} to the BL current, but *LRS* selected cells also contribute an extra current, I'_{ON} to the BL current, where $I'_{ON} = I_{ON} - I_{OFF}$. There is therefore a current-domain signal on the BL proportional to the accumulated I'_{ON}, plus an offset due to the accumulated I_{OFF}. The problem is that the total number of selected cells for CIM is proportional (for binary WLs) to the number of ones at the current bit position in the input vector, which is variable. If the on/off ratio is limited, then this variable offset can be significant compared to the signal.

3.2.7 Cell Variation

Cell current variation is a fundamental challenge for current-sensing CIM due to the tension between having a very dense memory cell with a reduced cross-sectional area and having an array of well-matched resistors capable of being programmed to precise values. The most important type of cell current variation is local (random) variation, although global variation can lead to challenges as well. Cell global current variation could result from intrinsic cell shift at the physical material level across the die or more subtly due, for example, to changes in how cells are programmed across the die due to variations in the power-delivery network (PDN) impedance.

Local variation leads to accumulating error for wider CIM kernels with potentially more parallel LRS cells enabled during readout. For L parallel LRS cells enabled, the standard deviation of read current due to LRS cells scales with $\sqrt{L}$. While this is sub-linear, the absolute sensing margin is still getting worse for higher-value readout states and therefore the probability of quantizing the measured state incorrectly is worse for these states. If all other effects including readout circuit noise and non-idealities are accounted for, cell variation (including both LRS and HRS contributions) fundamentally limits the number of cells that can be enabled in parallel for a given accuracy target. However, since reducing the number of cells measured in parallel reduces the net cell current variation, a successive readout approach can improve the trade-off in some cases [148]. Otherwise, the only way

the read circuit interfaces with the local variation issue is by not degrading the sensing margin through, for example, excessive offset or noise.

3.3 A Dense CIM with RRAM Macro

This section will now describe the first of the RRAM macro implementations. This first RRAM work followed a predecessor work and aimed to integrate better into a system-level work with many macros on-chip. From this background, the design originated with four aims: improve density by reducing peripheral area, achieve good modularity with limited need for analog voltage distribution around the chip, maintain or improve on the linearity of the readout system to make CIM operation more robust, and maintain flexibility for testing. This first design served as a learning experience about analog peripheral circuit design including current sensing for CIM. It was also a rehearsal for understanding the process of how to build a modular memory macro. This design was therefore an important stepping stone toward more sophisticated work.

3.3.1 Design

Figure 3.2 shows the topology of the implemented RRAM macro. The design consists of a write section at the top of the array, spatially separated WL drivers for low-voltage (LV) read and high-voltage (HV) write, and a read section at the bottom of the array. Shading is used to indicate which components operate at LV and which operate at HV. LV components that interact with the cell array all include some kind of isolation device or scheme to prevent any LV devices from experiencing HV stress when the array metal lines (BL, SL, WL) are driven to HV during write. This subsection will walk through the design covering these four topics and the challenges faced in their design.

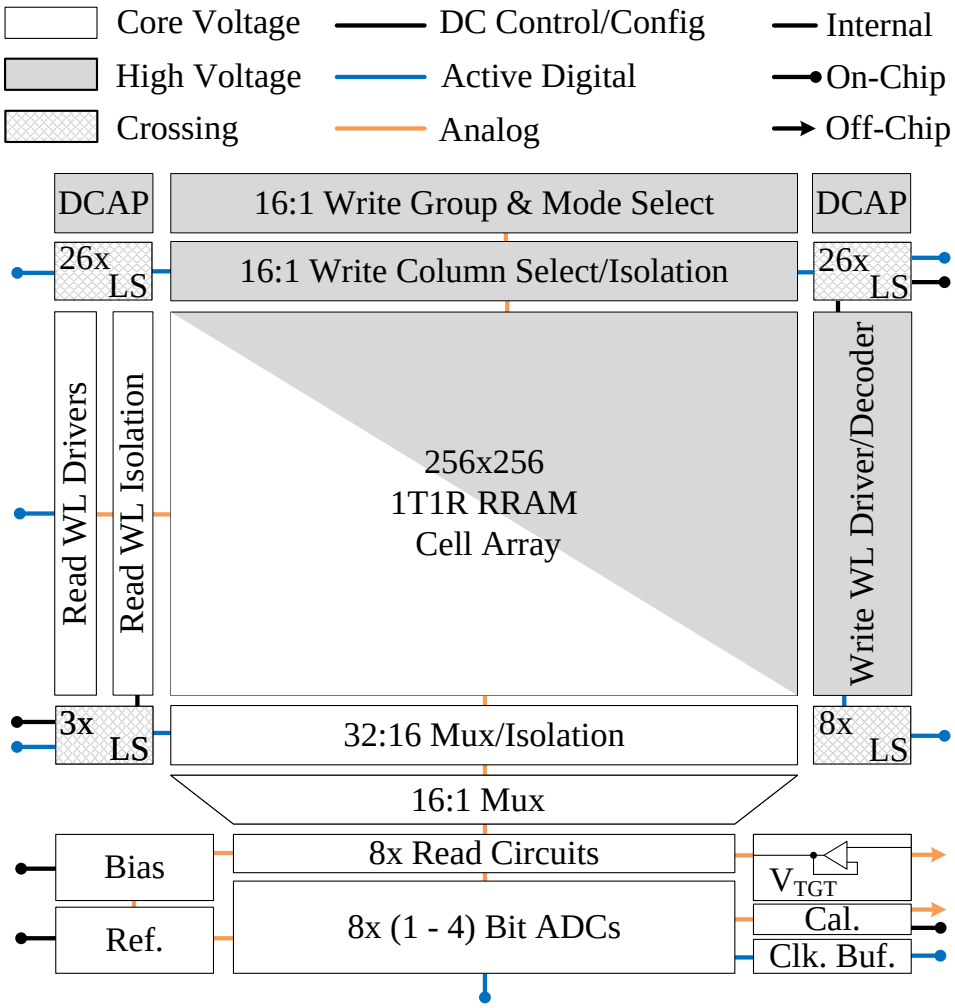


Figure 3.2: 256x256 RRAM CIM macro functional overview and rough floorplan.

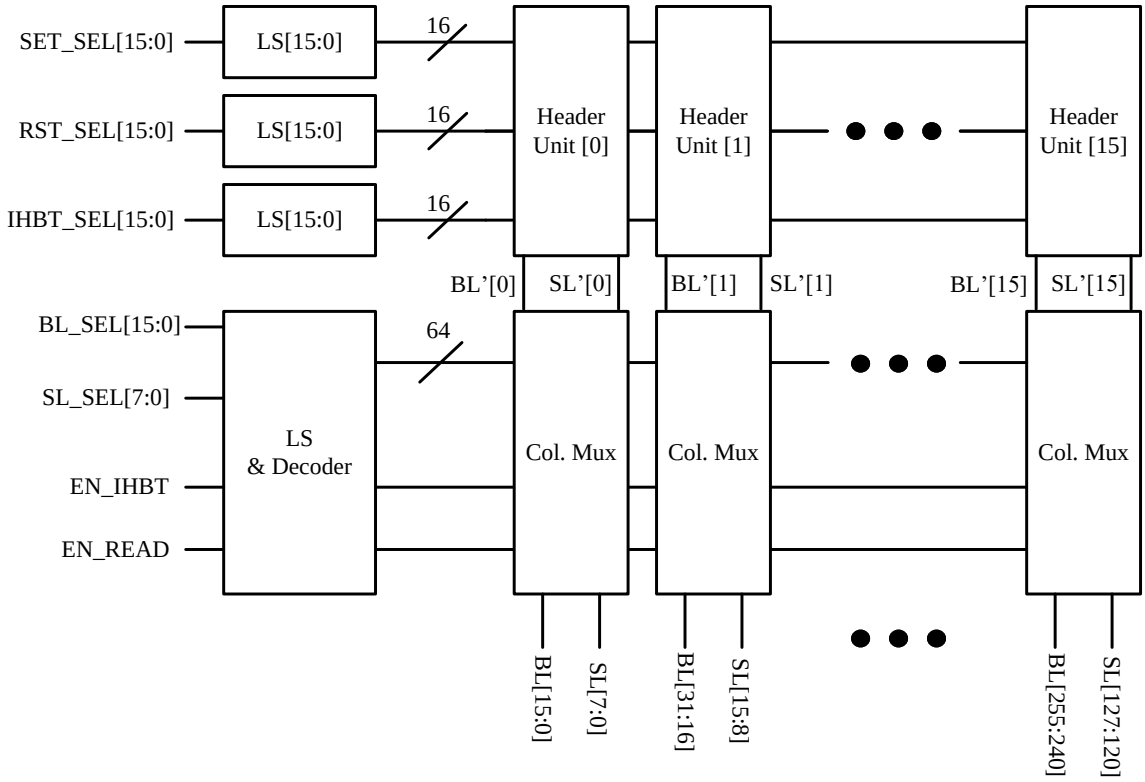


Figure 3.3: Voltage-based write driver for 256-column RRAM array.

Write Circuits

A voltage-based write scheme is chosen for these macros since the goal is not to extend cell endurance as in a digital NVM array targeting reliability for products with long lifetimes, but instead just to toggle cells repeatable between two stable states. This requires selectively switching each BL and SL between a write voltage VDD_{WR} and VSS as well as to $VIHBT$, the write inhibition voltage. The WL is switched between VSS and a HV VDD_{WL} rail to allow sufficient cell current to flow during programming and to allow the access device to turn on even in the 'upside-down' bias condition during RESET pulses. The WL driver for write will be discussed later, in the more general WL driver design discussion. The point of the inhibition voltage just mentioned is to reduce the stress on access devices in un-selected columns when the WL is driven to HV by raising both the BL and SL to an intermediate voltage (here, VDD).

The write BL/SL driver (Figure 3.3) is effectively a 16:1 MUX followed by 16 write configuration headers that drive the *selected wires* to the appropriate rails to send the correct kind of write pulse, as commanded by external logic. The lower-level MUX connects one of the 16 BLs in the group to this single per-group internal net, BL'_i, local to the group, while doing the same for the eight SLs and SL'_i. When a BL or SL is *not* selected for write, then it is either connected directly to $VIHBT$ if the array is in Write mode (WR) or left floating if the array is in Read mode (RD). In net, each BL and SL in the group can connect to one of two places. This then requires at least $(16 \times 2) + (8 \times 2) = 48$ HV switches implemented with thick-oxide devices per group. In actuality, some decoding for $VIHBT$ is performed locally and pass-gates are used for write pulses increasing the device count further.

In addition, a direct VSS connection using an HV MUX made of wide HV NMOS devices is included to provide a low resistance path from the SL to VSS during RD. The ones-hot signals to control the lower MUXes total 64, with two additional wires added to enable the $VIHBT$ connection for the floating BLs/SLs during WR and to enable the SL VSS tie during RD. Pre-decoding occurs in RTL, with some additional decoding including inversion to drive the pass gates occurring adjacent to the LSs. This custom pre-decoding is done situationally to limit the number of digital control pins on the macro. The HV control wires run horizontally over the large block of switches from LSs on the periphery, with the lower MUX system consuming about 73% of the pitch-matched write circuit area. Since the array format is not a butterfly style, the corner areas above the WL drivers and adjacent to the write driver are available for circuit placement and are used for LSs and decoupling capacitance (DCAP) on the write rails.

The write configuration header units are more straightforward switching blocks that, if a write is occurring, connect one of the BL or SL to VDD_{WR} and the other to VSS. If the array is in WR mode and the lower MUX is therefore driving BL'_i and SL'_i, but a write pulse is not needed for the group associated with the header unit, the header ties both of

these internal nodes to $VIHBT$. This could be the case, for example, if a word is being written but the cell is already at the desired final state due to a previous write event. As with the MUX, pre-decoding is in RTL and the ones-hot switch-control signals are level shifted on the periphery into three groups of 16, with each of the 16 wires in each group going to one of the 16 headers. In principle, this scheme allows 16-wide write pulses. In practice, there are two limitations: the VDD_{WR} and VDD_{WL} voltages are different for SET and RESET, so these different polarities cannot occur simultaneously, and the external wiring cannot necessarily support the full required current for very-parallel writes. For this chip, to keep the design risk low, the RTL allows one column (one cell) to receive a write pulse at a time.

WL Drivers

The WL drivers perform two tasks: sending a high-voltage pulse over a single WL during WR and sending a low-voltage pulse over several WLs in parallel during RD. The WR pulse can be slow with significant setup time, since WR time is mostly technology-limited, but the RD pulse should be fast. Since these tasks are fundamentally different (HV vs. LV, slow vs. fast, many-WL vs. single-WL) the two WL drivers are split for this design. While the LV RD path is simplified to a buffer chain followed by isolation devices (input is a vector from RTL), the HV WR path is more challenging since a similar RTL-driven decoder approach would require a LS for each WL at significant area overhead. As mentioned above, the array is not in a butterfly format so the overhead set by the WL driver x-dimension directly reduces area efficiency.

The design of the HV WR WL driver is shown in Figure 3.4. Once the WR WL driver is split from the RD driver, the design goal becomes to minimize overall area by encoding as much as possible. Encoding the control signals then decoding using simple gates reduces the number of large LSs needed in the design. For the 256-WL array, the fully-encoded approach requires eight LSs for signaling and two more for control. The decoder tree is

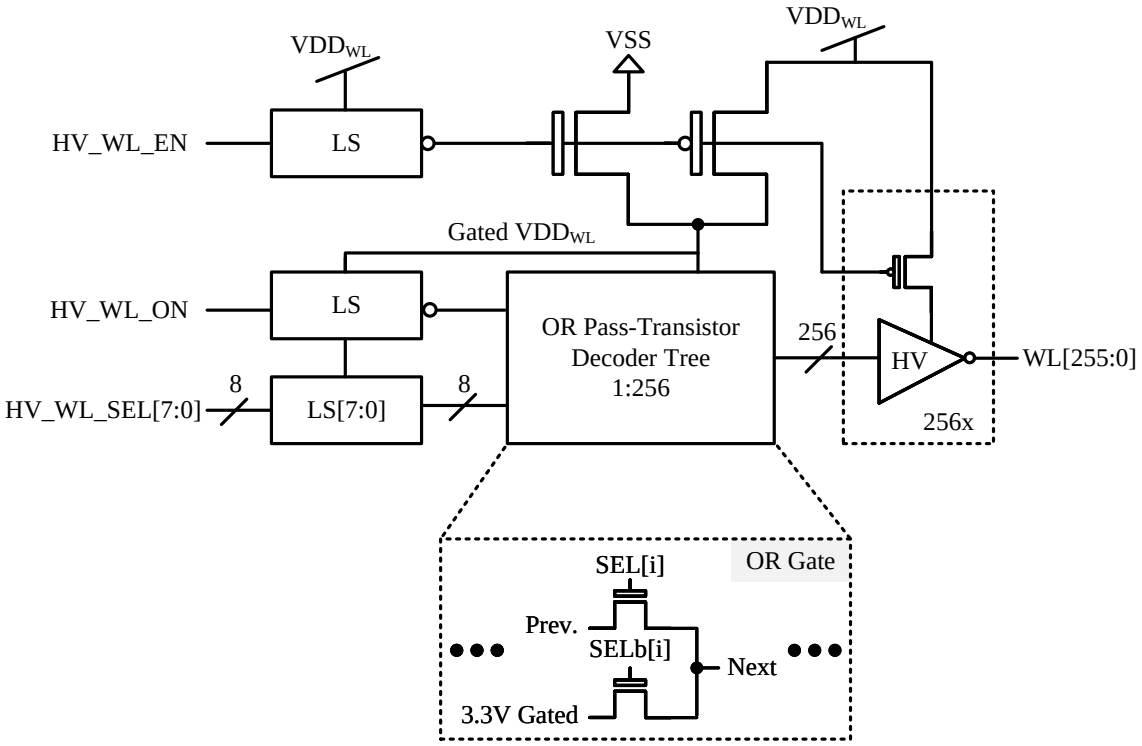


Figure 3.4: High-voltage WL driver for write with pass-transistor OR decoder. The OR tree has eight 2:1 layers.

built from eight layers of OR gates and is entirely designed using pass-transistor logic (PTL). In this way, each OR gate requires only two high-voltage NMOS devices. Signal swing is restored before driving the WL using a WL driving inverter made of HV devices with a PMOS header. The PMOS header makes it possible to fully disconnect the HV WL drivers from the WLs, as needed during RD, by disabling the header and driving the input of the buffer to VSS (to disable the NMOS pull-down device). The input of this final buffer can be driven to VSS natively by power gating the whole system with an added explicit VSS connection.

In net, ten LSs are required plus approximately seven additional thick-oxide devices per-WL to form the post-decoder and final driver (four in the tree, three in the driver). Since an LS-plus-decoder has a somewhat larger footprint compared to seven devices, the design was found to save area in net. The tree structure leads to a simpler layout as well.

Figure 3.5 shows the design of the LV WL driver. The driver at each WL is a buffer

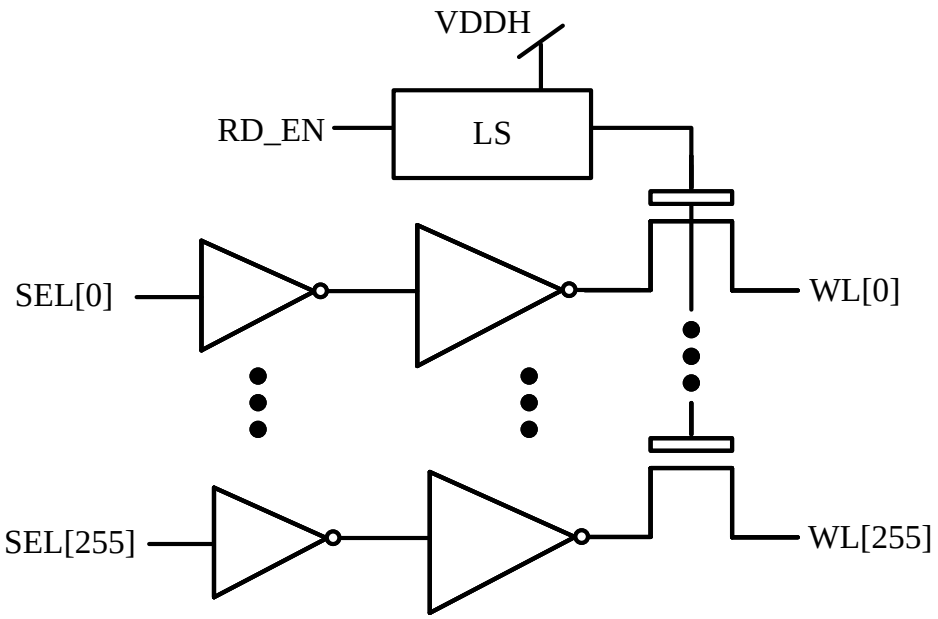


Figure 3.5: Low-voltage WL driver for read with pass-transistor isolation device.

with a series thick-oxide device to allow the output LV inverter to be isolated from the WL during WR. One LS is needed to drive these final isolation NMOS devices. There are several options for this final-drive device, including HV PMOS, HV NMOS, HV pass-gate, and LS with HV inverter. When normalized by total isolation-device gate area, the NMOS out-performs the PMOS and pass-gate approaches. This follows intuition since the LV WL signal is near VSS relative to the HV VDD (which could be 3.3V). The LS with HV inverter offers similar performance to the NMOS when normalized by output device area, but the LS before the inverter is much larger than the NMOS and would result in about five times larger total area for this output device according to design-time analysis.

Read Circuit and Biasing

Besides allowing the read controller RTL to drive multiple WLs high at once, CIM macros are distinguished from digital macros by the current-sensing read circuit with ADC. The goal for the front-end readout circuit for this design was to improve on the previous design [113] by linear-izing the readout to produce a ladder of evenly-spaced output states. The circuit introduced to accomplish this is shown in Figure 3.6. This circuit concept has been

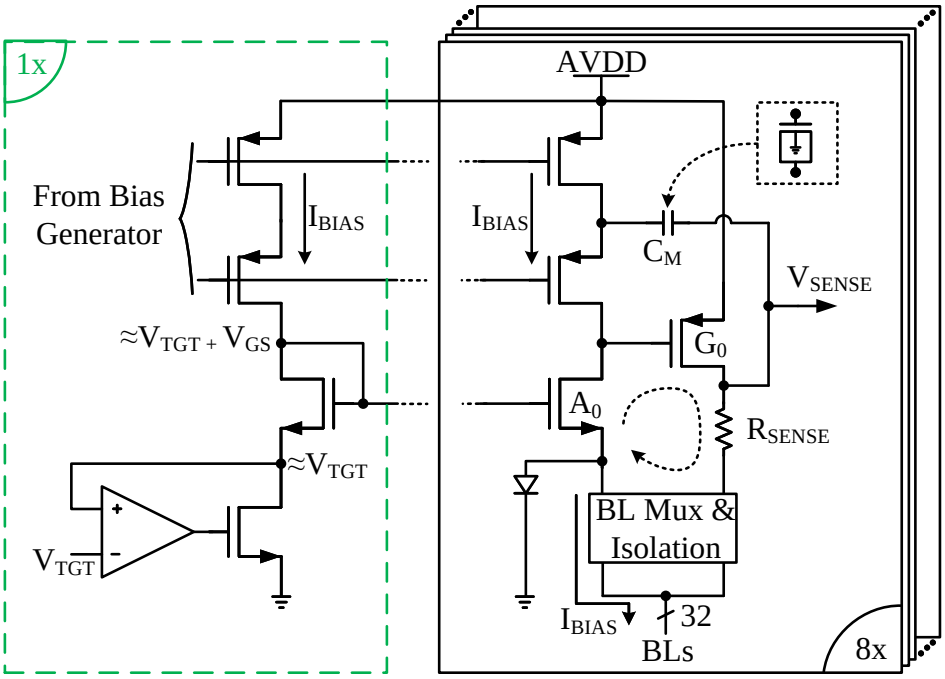


Figure 3.6: BL-clamping readout circuit with linearized output (TIA).

used before, as recently as [149]. This previous work was a high-speed SRAM design and thus omits the explicit process resistor that linear-izes the I/V conversion and the Miller capacitance.

The original publication called this topology "voltage-regulating current sense" although "current-sensing" probably would suffice. This is because current-sensing somewhat implies the use of a common-gate stage to clamp the BL to a target voltage while subsequently measuring the current that flows. The circuit in Figure 3.6 works like a typical TIA, with the op-amp having been replaced by the single-device trans-conductance stage A_0 biased by a cascoded current source and controlling an output trans-conductance stage G_0 that drives the current into the BL. The A_0 transistor in each channel compares the sensed BL voltage in that channel to the target clamping voltage, V_{TGT}, and produces an error signal that controls G_0, closing the feedback loop.

The named transistors A_0 and G_0 are sized to provide enough total trans-conductance gain to linear-ize the output across the expected load range. Resistor R_{SENSE} is a process polysilicon resistor, since RRAM-material-based resistors are not available in the process

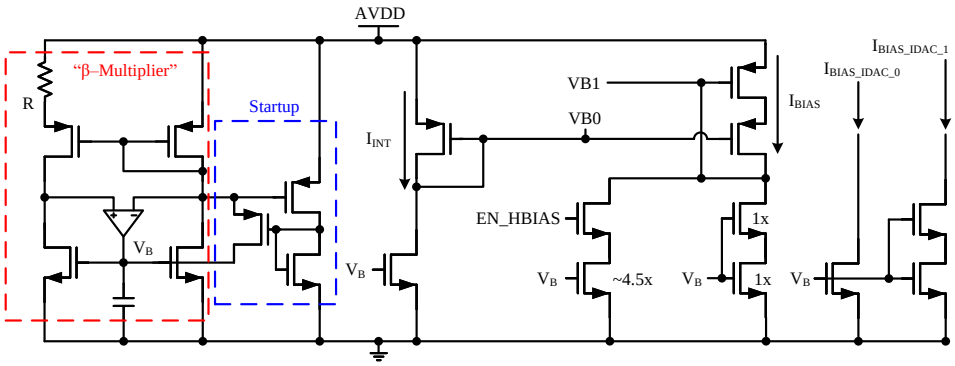


Figure 3.7: Bias generator with option for lower or higher bias current modes.

design kit (PDK). R_{SENSE} is placed between the regulated BL voltage and the output device since the resistance of the output device is already relatively large (drain-side) and because it then sits on top of a regulated node, providing supply noise rejection. The gate bias voltage for A_0 is produced once and shared across the eight channels in the array. The target BL value, V_{TGT}, is routed externally through the system-level design and is one of only two analog signals distributed widely across the chip (the other being a calibration port to audit the ADC references). It is buffered at each macro and used to drive the source of a replica device for A_0 in the macro bias circuit, to produce a gate voltage one unit V_{GS} above V_{TGT}. Since the devices and current-densities nominally match, V_{GS} will nominally match between the channels and bias circuit.

Separate paths are used for BL sensing and BL current drive in the BL MUX and isolation block, as in the previous work [113]. With this higher-gain design, this greatly aids linearity since it allows the gain of the clamping circuit to reduce the resistance seen due to the MUX. This block includes a 2:1 first-stage constructed using HV devices that can be fully disabled in RD mode. The second stage is a normal 16:1 LV decoder. A small keeper device, a diode-connected NMOS shown as a small diode, keeps the source of A_0 close enough to the target voltage when no cells are selected to avoid a long slewing time with lighter cell loads, a significant overshoot for heavier loads, and the risk of read-disturb if this node were allowed to drift toward VDD.

To avoid passing more sensitive voltage or current bias signals around the chip, each

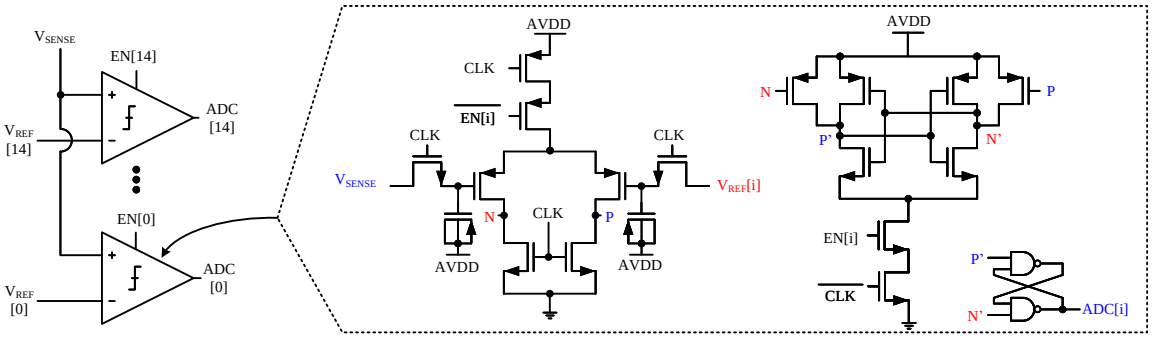


Figure 3.8: 15-Level flexible Flash ADC with input-sampling double-tail comparator.

macro has a self-biasing circuit, shown in Figure 3.7, based on the standard β-multiplier concept [150]. The bias circuit produces an internal bias voltage V_B then builds cascode current source bias voltages and sends reference currents to the current DACs (IDACs) that generate the Flash reference voltages. The readout circuit design assumes the cell load pulls at least the few-μA bias current when at the target clamping voltage. For very light loads, such as un-formed cells, it may be necessary to read with lower current. A switch, controlled by EN_HBIAS, provides this functionality.

Flash ADC and References

The last part of the readout chain is the Flash ADC, detailed in Figure 3.8. The comparator design is based on the double-tail comparator, a design which allows good performance across a wide input common-mode [151]. Since eight ADCs will share the same set of references in the array, kickback is an issue since a slight timing offset between the ADCs will mean one ADC's comparator can experience noise in its references due to the kick-back from another ADC in the macro. This is avoided by adding input-sampling at the comparator inputs (driven by the clock). However, this does not eliminate kickback charge but only blocks it from traveling up-stream to cause reference issues. Without parasitic loading on the comparator inputs, the gate voltages of the input transistors will increase dramatically due to the kickback (since kickback charge *is still generated*). The addition of explicit MOSCAPS on the now-isolated gate nodes mitigates the issue. In the physical

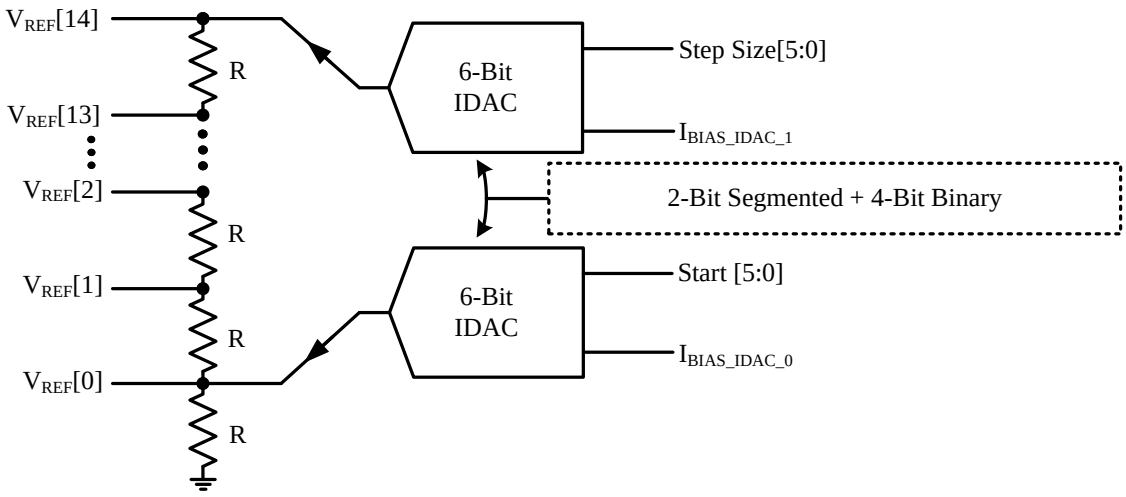


Figure 3.9: 15-Level ladder reference generator using two IDACs.

layout, shielding of the reference inputs to the ADC over long wire runs along the ADC bank provides further kickback immunity through parasitic MoM DCAP.

The ADC is made flexible via enable signals that allow some comparators to be disabled during read modes that require less precision to save power. Finally, as mentioned above, switch is included so that the ADC can measure either the sensed CIM voltage, V_{SENSE}, or an off-chip voltage. This allows the ADC references, described next, to be calibrated and understood against real-world voltages.

The Flash ADC requires a reference generator, shown in Figure 3.9. Two 6b IDACs are biased from the bias generator with scaled dynamic ranges to match their individual roles. The upper IDAC pushes current through *all* of the 15 reference-generating resistors, while the lower IDAC pushes current just through the first (lowest in the ladder) resistor. In this way, the top IDAC controls *step-size* while the bottom IDAC controls *offset*. This aligns with the readout circuit behavior. The readout circuit, ideally, places a sequence of linearly-spaced readout states on top of a BL clamping voltage that can be varied to match the observed cell resistance range for the chip under test.

HV Isolation and RD/WR Modes

The last design consideration for the macro is safety; that is, avoiding RD and WR collisions that expose core-voltage devices to HV stresses. Details about this have been provided throughout this design subsection, but the RD and WR modes will be summarized here to clarify. In WR mode, the LV WL drivers are isolated from the WLs by disabling the thick-oxide output NMOS device, while the LV BL MUXes are isolated from the BL by disabling the first stage of the column MUX which was built using thick-oxide devices. There is no connection of core-voltage devices to the SL. During RD mode, the HV WL drivers are disconnected from the WLs by disconnecting the pull-up path in the drivers using a pass PMOS and driving the input of these drivers to VSS to disable the pull-down path. The HV BL/SL MUXes are set to all-VSS to disable the MUXes. The SL tie to VSS at the top of the array is also enabled.

3.3.2 Context

This first macro design was integrated into a larger test chip and taped-out in 40nm CMOS as part of a system with 288 macros [146]. The die micro-photograph is shown in Figure 3.10. The macro computes up to 9-wide 1b CIM MAC operations with higher precision constructed externally using shift-adds.

3.3.3 Results

The overall design was successfully tested in silicon. The success of the project relied on good communication about the RTL needed at the macro boundary to read and write memory cells and use the CIM mode. It was possible to use the larger system to test the behavior of individual macros to gain some insight into how well the macro design performs in isolation. A portion of the measured results will be presented here to show how the macro met the design goals that were laid out at the outset of the project.

The color-coded area breakdown of the mixed-signal portion of the macro is shown in

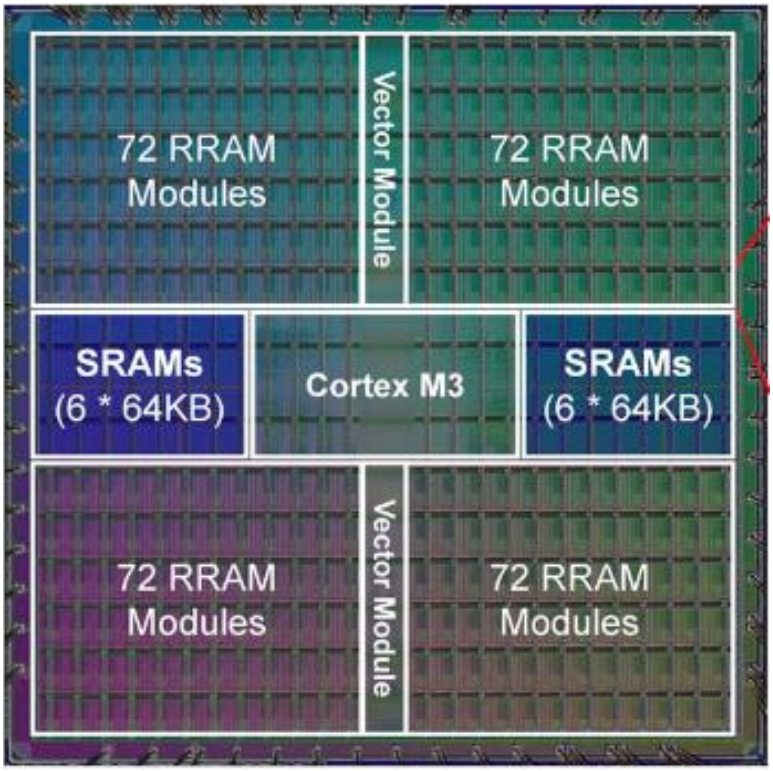


Figure 3.10: Die micro-photograph of system with 288 of the first of the two macros [146].

Table 3.1: Pulse and threshold definitions for Figure 3.12. These are approximate and use the calibration port to report voltages and nominal numbers from the design to report these voltages in terms of resistance in Ohms.

Write Op.	SET	RESET	FORM
Pulse Length	125ns	10μs	100μs
VBL	V_{WRITE}	0	V_{WRITE}
VSL	0	V_{WRITE}	0
VWL	V_{WRITE}	V_{WRITE}	1.7V
V_{SENSE} Threshold	> 385mV	< 270mV	> 350mV
Resistance Threshold	< 3.2kΩ	> 8.6kΩ	< 4kΩ

Figure 3.11. Considering the mixed-signal components, roughly 29% of overall macro area is RRAM memory cells. This results in a considerably (16.2x) improved density compared to the predecessor work. Area is dominated by the ADCs and the write driver. The Flash ADCs, even with 15 states, take considerable area. The write driver is large due to the need for many high-voltage devices.

The functionality of the RRAM memory cells and the readout chain was first confirmed

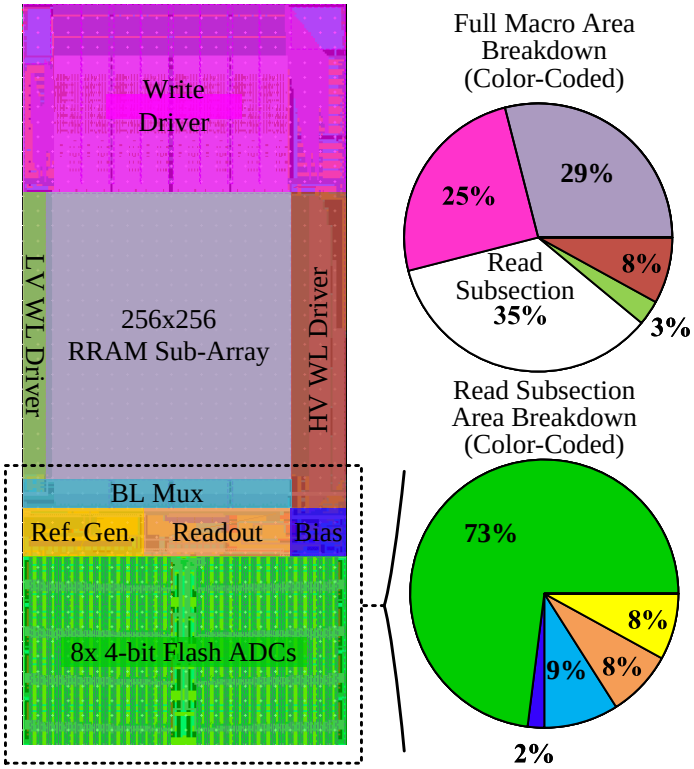


Figure 3.11: Color-coded area breakdown of the mixed-signal part of the macro (excluding RTL-based parts).

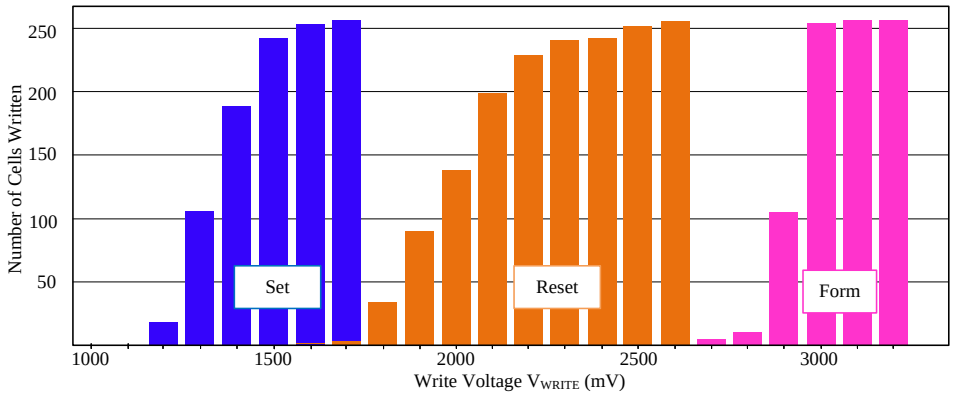


Figure 3.12: Number of cells, out of 256, written in each write mode using different pulse settings.

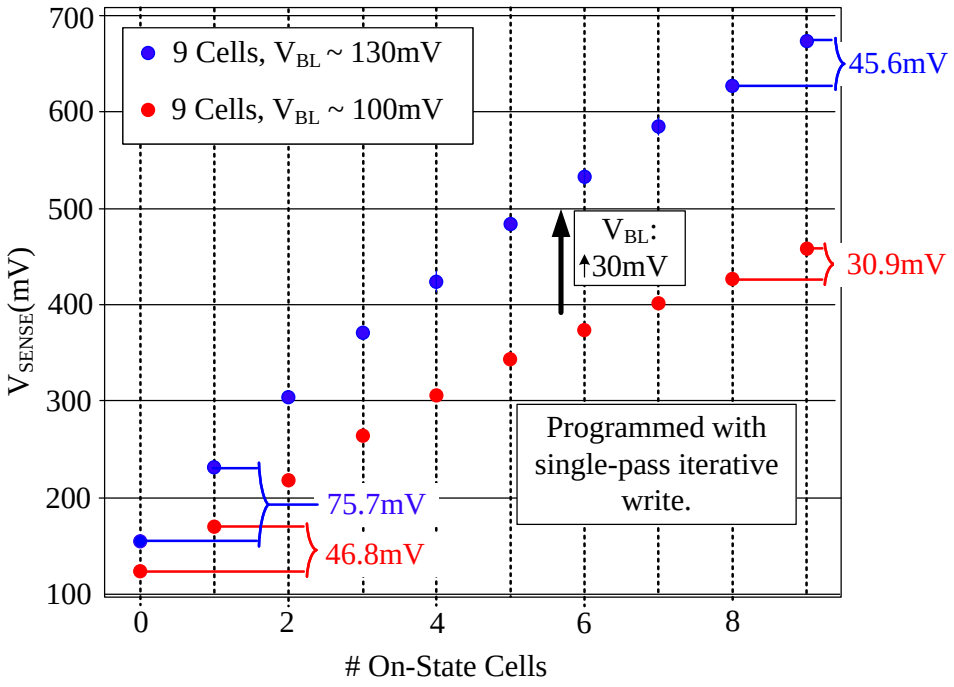


Figure 3.13: Mean state placement for 9-WL CIM mode with two different target clamping voltages at 1.1V VDD. Ideally, the linear state placement extends from the clamping voltage at the low-end to about V_{GS} below VDD.

by FORMing, SETing, and RESETing the cells in the array. Figure 3.12 shows the results of attempting one-shot write operations in the macro using sample populations of 256 cells (one column). The pulse parameters and approximate threshold conditions used to determine success for these operations are given in Table 3.1.

The next characterization item is the transfer function of the CIM operation, shown in Figure 3.13. This measured result is produced by sweeping the ADC through known values that were previously calibrated using the off-chip port. The IDACs are adjusted to create many measurements for the same readout condition. These many binary values are then combined in software to estimate the real voltage produced by the front-end for the given set of memory cells and input vector. The cleanest curve occurs in one-time-programmable (OTP) mode, where HRS cells are never formed. This improves the on/off ratio. The original publication included error bars: while these are accurate to the measured data, they are subject to the limitations of this test and could be confusing (the sweeping protocol

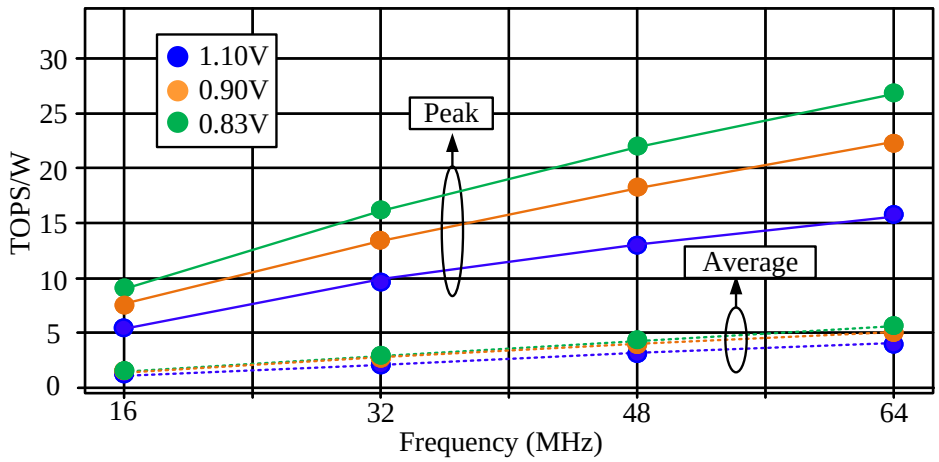


Figure 3.14: Estimated energy efficiency in TOPS/W of one macro based on measurement in the 9-cell CIM mode within the 288-macro system implementation.

likely significantly reduces the importance of noise and offset). They are omitted here. The plot is produced by programming a column and then enabling cells to produce each LRS-count output state, and measuring as described to extract the mean transfer function. The linearity is greatly improved, aligning with the design goal.

The last characterization point for this work that will be presented is energy efficiency (in TOPS/W), shown in Figure 3.14. This is an efficiency estimate using the measured power results from the chip and modeling for a single macro. The average case is 50% 1/0 for the input and stored weights. In addition to meeting the design goals and having competitive energy efficiency, this macro was the densest reported CIM with RRAM macro at the time of publication even before normalizing by technology node.

3.4 Circuit Solutions Addressing RRAM Compute-in-Memory Non-Idealities

While the first macro design met the design goals that were set out for that project, it did not include many explicit design elements aimed at improving CIM robustness. The linearized readout, shown in Figure 3.13, is the main measured item for robustness in the first macro and only shows how that macro produces an output signal proportional to the number of selected, LRS cells. Improving robustness further means confronting the non-idealities

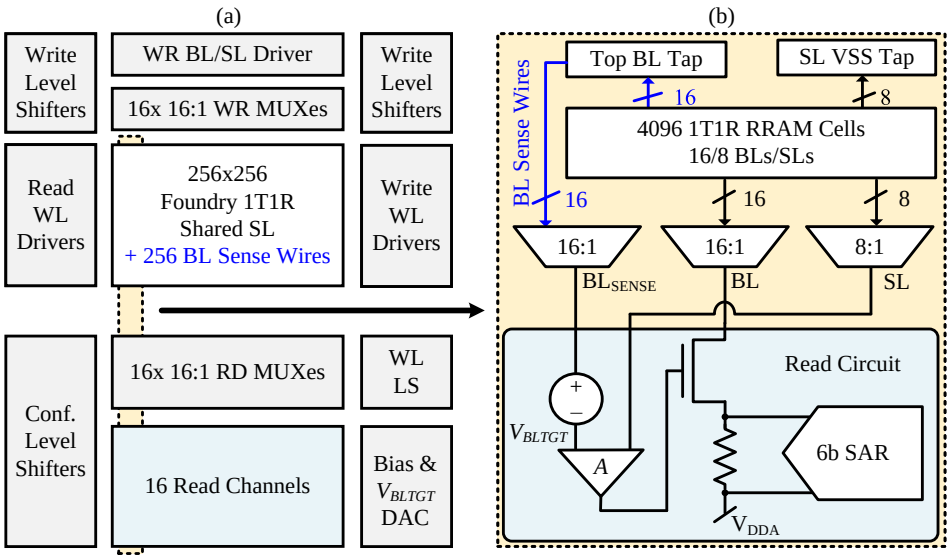


Figure 3.15: (a) Overview of the second macro with write subsection and WL drivers mostly borrowed from the first macro. (b) Highlight of a single pitch-matched read channel in the second macro.

introduced in the challenges section above: gain and offset errors due to channel mismatch, IR drop, and off-state current.

The follow-up macro design (the third-generation macro in this line of work and the second and last RRAM CIM macro that forms a part of the PhD work) introduced circuit ideas to help reduce the impact of these non-idealities on CIM performance. The design featured a more sophisticated TIA design, a 6b SAR ADC designed by a pair of collaborators from the lab, and a 7b DAC to generate the target clamping voltage internally at each macro. The main goals of this macro design are to show, through measurement in working-in-silicon macros with competitive density and efficiency, significant improvement compared to measured or simulated baseline in key areas: readout linearity, channel mismatch, IR drop, and error due to off-state current.

3.4.1 Design

Since the non-idealities all relate to readout, this macro design almost exactly copies the previous design for WL drivers and write circuits. The voltage scheme for write was ad-

Table 3.2: Summary of categorizing non-idealities and assigning them to circuit blocks for mitigation.

Non-Ideality	Type	Circuit Block	Resolution
IR Drop	Gain Error	TIA	Use approximate 4-terminal (Kelvin) current sensing.
Channel Gain Mismatch	Gain Error	TIA	Use an offset-cancelling TIA.
Cell Global Variation	Gain Error	TIA	Trim the clamping voltage at each macro.
Variable I_{OFF}	Offset Error	ADC	Dynamically adjust ADC offset.
ADC Offset	Offset Error	ADC	Statically trim ADC offset.
Cell Random Variation	Random Error	External	Apply ECC [148] or Tolerate Errors.

justed slightly to simplify the overall chip design, although the details are not significant and will not be discussed here. An overall picture of the new macro is shown in Figure 3.15. This macro features 16 read channels (contrasting with 8 in the first macro) and a new read MUXing scheme that will be explained while discussing the TIA design. A 6b SAR ADC replaces the 4b Flash design from the previous macro. This design discussion will introduce the approach for how the macro manages non-idealities before describing the functional blocks and finally providing overall implementation comments.

Approach

The management of non-idealities in this design relies on the observation that the challenges can be cleanly grouped into two categories. The reference point is the voltage-domain signal that feeds into the SAR ADC. The first category is gain error, and there are three core contributors: channel mismatch in the TIA front-end or sensing resistor, IR drop due to array parasitic resistance, and global variation in the ratio of the RRAM LRS resistance to the sensing resistance. The second category is offset error, with two contributors: ADC intrinsic input-referred voltage offset, and variable off-state current due to different counts of nonzero bits in the input vector.

Once the non-idealities are categorized, assigning them to circuit blocks for mitigation is more straightforward. Gain errors are naturally handled by the TIA. Adjusting the clamping voltage that the TIA applies to the memory cells scales the current that flows and thus controls the gain of the signal that eventually feeds the ADC. Adjusting the ADC offset, tautologically, adjusts the voltage-domain offset at the ADC's input; this is less ideal, however, since offset due to off-state current occurs pre-TIA and is therefore affected by TIA non-linearity. This work does not offer a solution beyond linear-izing the front-end; a more ideal solution could actually inject current at the front-end, although this requires an expensive current-DAC or reserved set of RRAM cells. A summary of the proposed approach is shown in Table 3.2.

Impedance Sensing: TIA

Recapping the above, both input-referred offset error in the TIA's operational trans-conductance amplifier (OTA) and IR drop between the TIA input and the RRAM cell load scale the clamped voltage seen by RRAM cells during readout. This scales the current flowing through the TIA's sensing resistor, leading to gain error at the ADC input. Channel variation in the process resistor used for current/voltage (I/V) conversion in the TIA also manifests as gain error, but a narrow macro (reduced worst distance between sensing resistors) and a line-array layout with dummies reduces this to a secondary concern.

Offset cancelling in an OTA is a well-understood problem [152], and alignment of RRAM resistance with sensing resistance to achieve the desired dynamic range can be accomplished by adding a medium-precision (7b) DAC in the macro to set a trimmed V_{TGT} clamping value locally. The most interesting design question due to these design challenges is how to mitigate the impact of IR drop. The proposed solution will be described in detail here. Figure 3.16 (a) explicitly shows parasitic resistance in a CIM column. Total parasitic resistance changes with the position of the load on the BL/SL. The model in Figure 3.16 (b) omits parasitic resistors *between* selected current-pulling cells leaving two

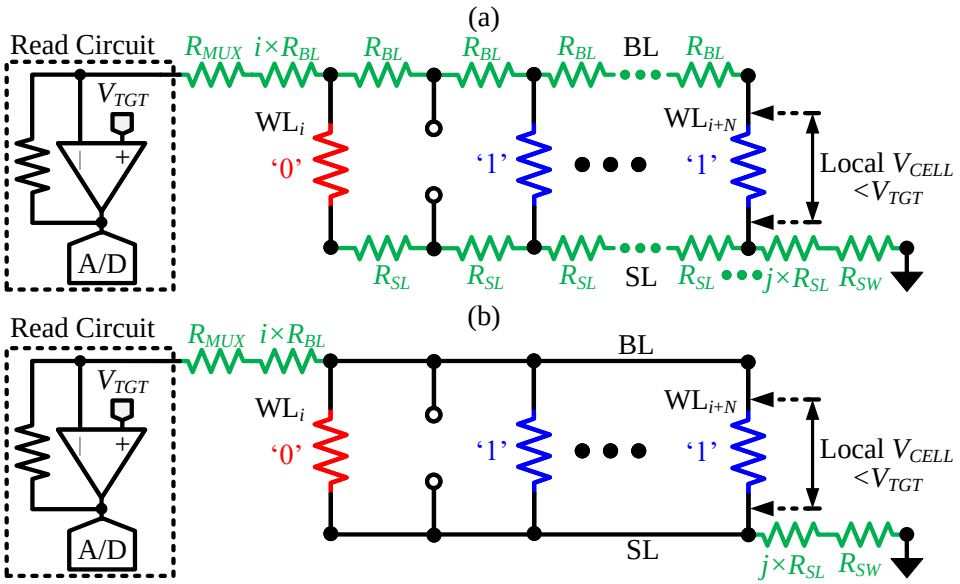


Figure 3.16: Current-sensing CIM for a single column showing (a) a complete picture of BL, SL, and MUX/switch parasitic resistances and (b) a lumped parasitic resistor model. Variables i and j relate to position along the BL: $i + N + j$ is equal to the number of WLs in the array.

lumped resistors: above the group of cells on the BL and below the cells on the SL.

An observation is that the *intra-cell* parasitic resistances are invisible to outside measurement, where the measurement is presumably made by connecting a resistance sensor to the BL and SL at the periphery of the array, without knowledge of cell values. There is only limited observability into the details of the resistance network using the desired one-shot-read approach. From this, the lumped model, Figure 3.16 (b), represents a *solvable variant* of the IR drop problem which ignores the effect of these less-observable resistances. Pathological cases, with large gaps between cells, are possible and would expose the weakness of the model. These cases are unlikely during application of fixed-length vectors at fixed intervals along the WLs during CIM operation and are not considered here.

Shown in Figure 3.17, the positional dependence can be reduced through complementary routing of the BL and SL (drive current from the TIA into the BL at array South, tie the SL to V_{SS} at array North) if per-cell resistance of the BL and SL are similar. This approach was used in the first macro and is very easy to implement since it requires adding just

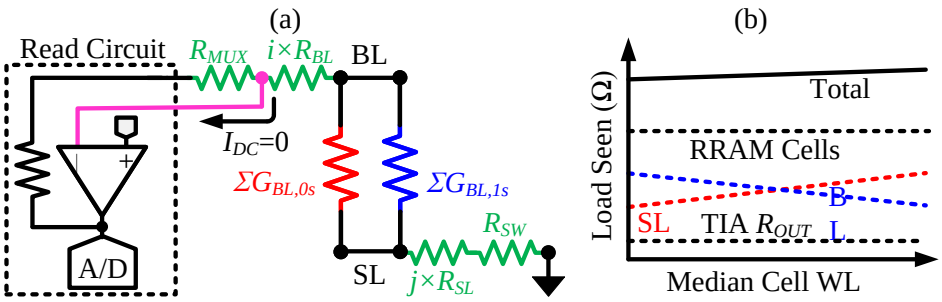


Figure 3.17: Sensing around the column MUX and thick-oxide read isolation devices while flowing current in/out of the BL and SL at opposite ends of the array, (a), reduces total parasitic resistance and positional dependence, (b).

another set of switches, with a relatively simple control scheme, in the large write-switch area. Further, MUX resistance can be reduced using the TIA's feedback gain by including a sensing path in the column MUX. Still, input resistance of the TIA is bottle-necked by array parasitics, limiting CIM read linearity.

The proposed solution is inspired by Kelvin sensing and is to add sensing paths *around* the lumped resistors in Figure 3.16 (b). Strictly under the lumped model, four-terminal resistance sensing, Figure 3.18 (a), can be achieved by sensing V_{SL} at array South, clamping the SL to V_{SS} at array North, measuring V_{BL} at array North, and driving current into the BL at array South. This separates the resistive segments of the BL and SL that carry the clamping current from the resistive segments of the BL and SL that provide feedback to the OTA and set the clamping accuracy. Feedback to the OTA must now be differential, and it should be designed to drive $V_{BL} - V_{SL}$ to be equal to the target clamping voltage V_{TGT}.

There are two new design tasks: add these new sensing paths and design an offset-cancelling TIA that accepts the differential input. To sense V_{BL} at array North we add vertical metal-6 wires, each via'd to the BL at array North and to an added column MUX at array South (Figure 3.18 (b)). To clamp the *differential* sensed voltage to V_{TGT} (set by 7b per-macro digital-to-analog converter, DAC), we propose using the standard [152] offset-cancelling folded OTA topology in a *modified context* by driving the input offset to $-V_{TGT}$, rather than zero, during offset-cancellation cycles. The overall scheme from DAC to the

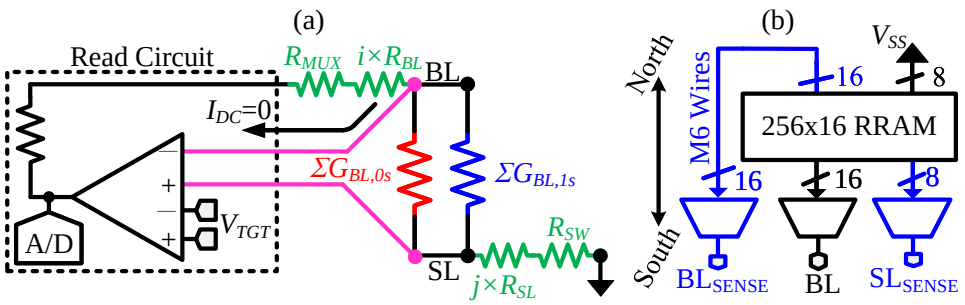


Figure 3.18: Four-terminal (a) impedance sensing drives the differential voltage across the RRAM cells to a differential target voltage set by a local DAC. Shown in (b), this approach requires separate MUXes for sensing wires on the BL and SL and added wires for BL sensing.

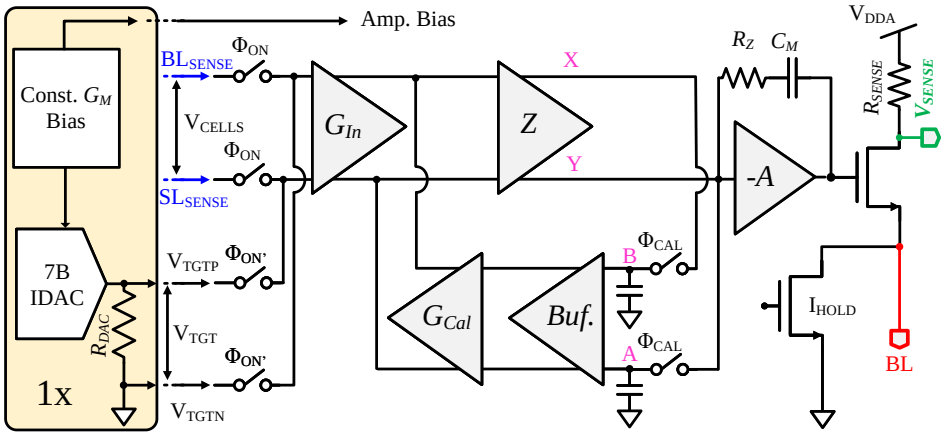


Figure 3.19: Symbolic topology of the current-sensing front end (TIA). During offset cancelling cycles, the input of G_{In} is shorted together and the path through G_{Cal} is configured by feedback gain so that the storage nodes A and B are configured to (1) cancel G_{In}'s offset and (2) set the new zero-point of the amplifier to the point where $V_{BL} - V_{SL} = V_{TGT}$.

voltage output provided to the ADC is shown in Figure 3.19.

The transistor-level implementation and switching scheme are shown in Figure 3.20 (a) and (b). To improve availability, retention is extended by storing the analog calibration nodes, A and B, on thick-oxide PMOSCAPs (choice driven by simulation) and buffering through an NMOS follower (M1/M2). The follower also partially offsets V_{GS} of M3/M4 to make A and B easier to drive for the trans-resistance stage, whose output common mode is set by the gate connection of M5/M6. The DAC and bias circuit are derived from the designs learned earlier during the development of the first macro with some details, such as device sizing and DAC precision, improved for this design.

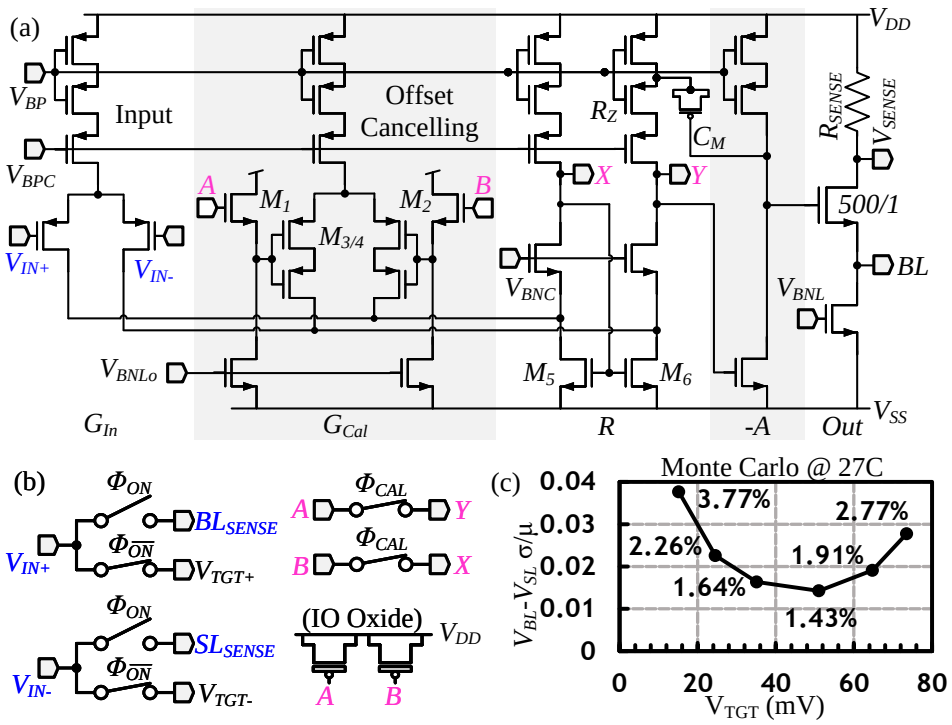


Figure 3.20: Implementation of the four-terminal TIA for RRAM readout: (a) transistor-level details, showing (left-to-right) input and offset-cancelling trans-conductance stages G_{In} and G_{Cal}, trans-resistance stage R, gain stage $-A$, and output stage Out; (b) the switches; and (c) 6 separate 1k-sample Monte Carlo simulations at 900mV, typical, 27C with 16-LRS-cell load.

Monte Carlo simulation in Figure 3.20 (c) shows the normalized clamping voltage standard deviation (σ/μ) of the implemented TIA across a range of V_{TGT} DAC values, demonstrating two-sided behavior (a valley). Typical offset-cancelling OTAs show a one-sided behavior; working with larger signals (μ) leads to lower normalized σ/μ. The implemented TIA exhibits new behavior since cancelling large offsets (applying large $-V_{TGT}$ during cancellation) challenges the lower-trans-conductance offset-cancelling arm. Careful adjustment of the OTA provides a wide usable cell bias region, at low V_{TGT}, with variation in the $1-2.5\%$ range. Since accumulated variation for higher-value states (more LRS in parallel) will greatly exceed 2.5% with current RRAM technology, the matching is sufficient for the task.

Data Conversion: ADC

The ADC design was completed by lab co-workers then merged into the final macro design as a sub-block physically beneath (South-of) the TIA block. The 6b SAR ADC design is notable for its density ($278\mu m^2$ per lane) and built-in offset-cancelling CDAC. Continuing from the previous discussion, the non-idealities that the ADC is responsible for addressing are its own intrinsic offset and the voltage-domain shift due to the variable off-state current. A quick analysis will show that off-state current indeed appears as an input-dependent current offset.

Neglecting parasitic resistances, variation, and cell non-linearity, we can assert that selected (WL='1') RRAM cells contribute G_{ON} in the low-resistance state (LRS) or G_{OFF} in the high-resistance state (HRS) toward the total BL conductance, G_{BL}. If we define an incremental LRS conductance in place of the absolute LRS conductance:

$$G'_{ON} = G_{ON} - G_{OFF} \tag{3.1}$$

then for N selected cells and M LRS selected cells:

$$G_{BL} = MG_{ON} + (N - M)G_{OFF} = MG'_{ON} + NG_{OFF} \tag{3.2}$$

If we define the input-vector dependent offset G_{Leak} as NG_{OFF}, then the total BL conductance is composed of a signal component MG'_{ON} and a leakage component G_{Leak}. As a quick aside, in OTP mode, where $G_{OFF} \approx 0$ relative to G_{ON}, note that $G'_{ON} \approx G_{ON}$. Continuing, using sense resistor R_{SENSE}, the measured signal is then:

$$|V_{SENSE}| = \frac{V_{TGT}}{R_{SENSE}} \left(MG'_{ON} + G_{Leak}\right) \tag{3.3}$$

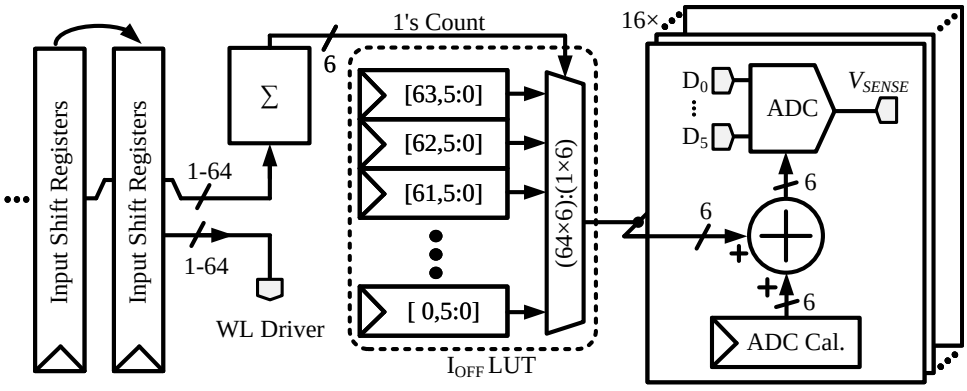


Figure 3.21: System for adjusting ADC offset to cancel off-state current.

The voltage error introduced by off-state current is:

$$|V_{OFF}| = N\frac{V_{TGT}G_{OFF}}{R_{SENSE}} \tag{3.4}$$

Concluding, off-state current applies a voltage offset at the ADC input in series with the ADC's intrinsic offset, proportional to V_{TGT}, R_{SENSE}, and N (ones-count). The consistency of this offset across output states depends on off-state current variation and the linearity of the front-end TIA.

Due to the improved linearity of the implemented TIA, enabled by four-terminal sensing as described, we made the decision for this work to address off-state current in the voltage domain at the ADC input. Once committed to this approach, the implementation involves a two-step calibration to first trim macro-wide for the offset due to I_{OFF} and then trim locally at each channel for the ADC's mismatch.

The details are summarized in Figure 3.21. The proposed technique is to maintain two structures: a per-channel 6b register storing each ADC's self-measured intrinsic offset, and a per-macro 64-entry 6b lookup table (LUT) storing a calibrated offset for each ones-count value from the input vector. The 6b SAR ADCs are equipped with built-in offset cancelling DACs. During readout, the number of ones in the input vector is summed and the resulting value is used to index into the LUT with the expense amortized over 16 columns. Per-column (per-ADC), the intrinsic offset and this calibrated offset are combined and used to

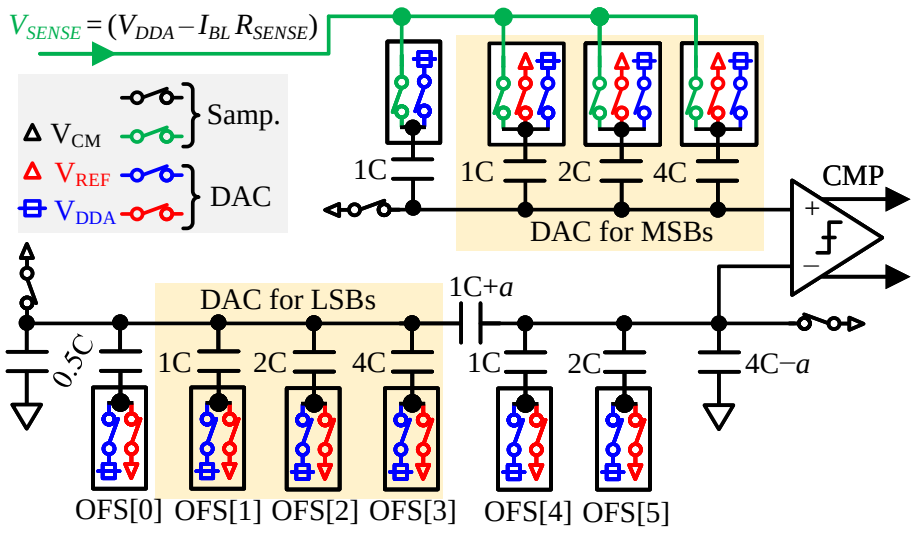


Figure 3.22: Implementation of the offset-sensing/offset-adjusted 6-Bit Split-DAC SAR ADC. The unit capacitance is 1.7fF.

set each ADC's voltage-domain input offset, dynamically, for the next conversion.

Most of this approach is implemented in the RTL wrapper for the macro. The mixed-signal component, the SAR ADC, is shown in Figure 3.22. To achieve a very small footprint for the SAR ADC while offering acceptable linearity and performance, a split-DAC approach is used with a custom, hand-drawn 1.7fF unit MoM capacitor. Separate capacitive DACs sample the TIA output and the offset input simultaneously, avoiding timing overhead due to a separate offset-cancellation step.

While we committed to this approach for cancelling off-state current, others were available. Compared to scaling the transimpedance gain (sensing resistor or V_{TGT}) based on ones-count, or injecting current into the TIA to emulate a constant G_{Leak}, this approach requires no extra analog area and reduced power overhead in the front-end but is sensitive to TIA linearity. The gain scaling approach was used in the work that preceded the first macro design [113]. Compared to digitally adjusting *post hoc*, the approach we used allows finer correction, since the ADC offset-correcting DAC's LSB is $0.5\times$ the conversion LSB. This also extends the usable ADC's input dynamic range.

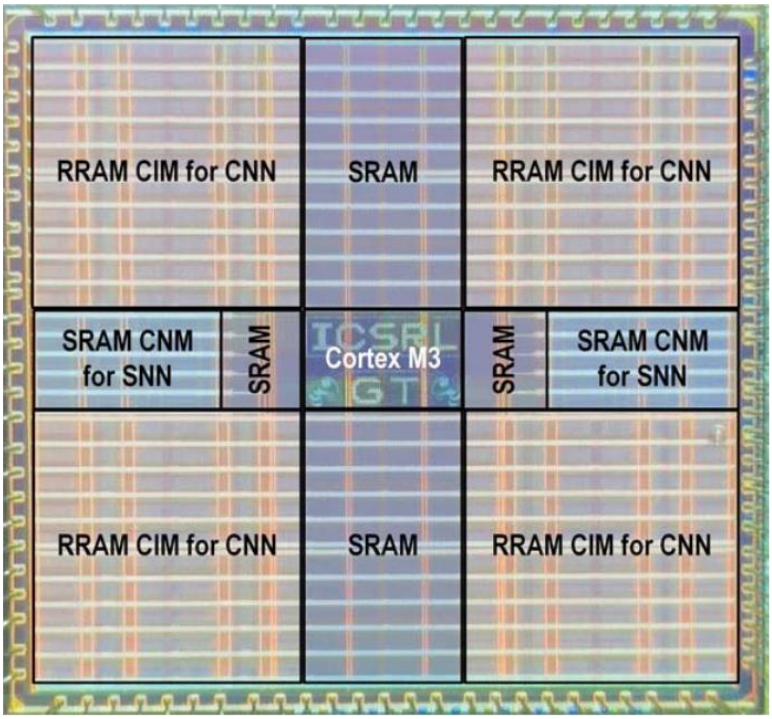


Figure 3.23: Die micro-photograph of system with 160 of the second of the two macros [147].

3.4.2 Context

Like the first macro, this second macro design was integrated into a larger test chip and taped-out in 40nm CMOS with foundry RRAM. This chip featured 160 macros organized as five groups of 32 [147]. The die micro-photograph is shown in Figure 3.23. The 256×256 CIM macro has a post-shrink area of 0.0263mm^2 and achieves an array efficiency of 24%. The 16-read-channel macro natively computes 16× 0-to-256-wide 1b-in, 1b-weight, 6b-out MACs. Full precision MACs are constructed externally using shift-adding.

3.4.3 Results

This design was tested in silicon. To show the improvement in robustness beyond the first macro, the characterization for this second macro focuses on measuring the extent to which the improvements just described resolve the stated challenges.

Since only the readout subsection differs significantly from the first macro, the area

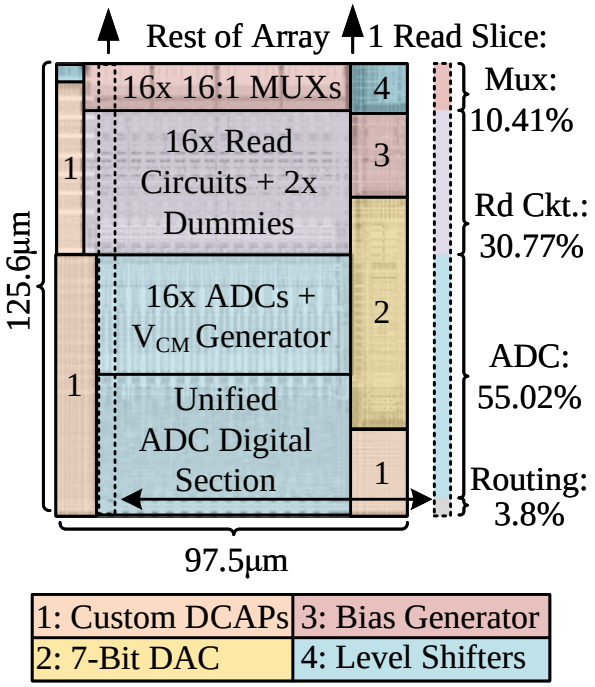


Figure 3.24: Breakdown of the read circuit section for the second macro.

breakdown of just the new readout subsection is shown in Figure 3.24. As mentioned for the first macro, when a butterfly format is not used the corner areas, including the bottom-left and bottom-right of the macro, can be available for placing circuit blocks. Here, the seven bit V_{TGT} DAC and the bias generator are built to fill this area; the rest of the open area is filled with custom DCAPs for the analog rails. Since the read channels are pitch-matched, the critical dimension that determines how important they are to total area is the height (y-dim). Even with added offset-cancelling, the read circuit is only about 31% of the read section height, while the ADC is a more important 55%. This validates the choice of adding sophistication to the front-end and focusing on density as a priority for the ADC.

For column-level MAC characterization, a checkerboard pattern was programmed using write-readback and pseudorandom test vectors for each mode (8WL, 16WL, ...) were generated in Python. To provide some statistical confidence, 1,000 vectors per MAC output state were used. For example, in the 32-cell mode, the characterization involves 33,000 total raw ADC measurements, since 32 active cells imply 33 possible states including the

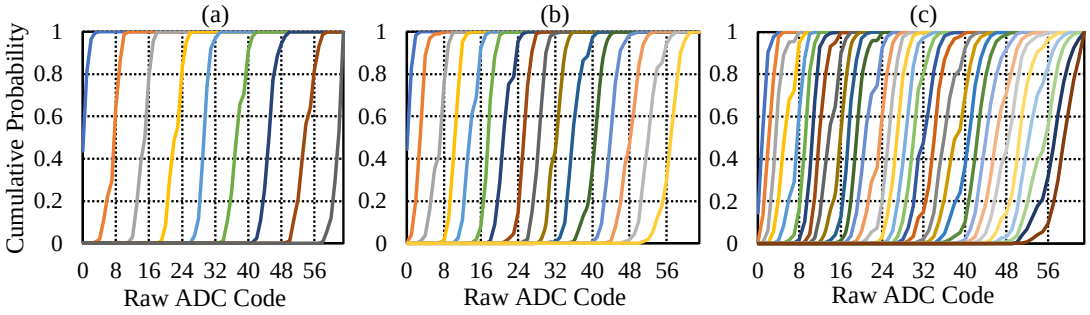


Figure 3.25: Characterization is performed using 1000 random samples for each output state. (a), (b), and (c) respectively show cumulative distribution function (CDF) waterfalls for the allowed output states for 8, 16, and 32-cell CIM modes in terms of the measured raw ADC code.

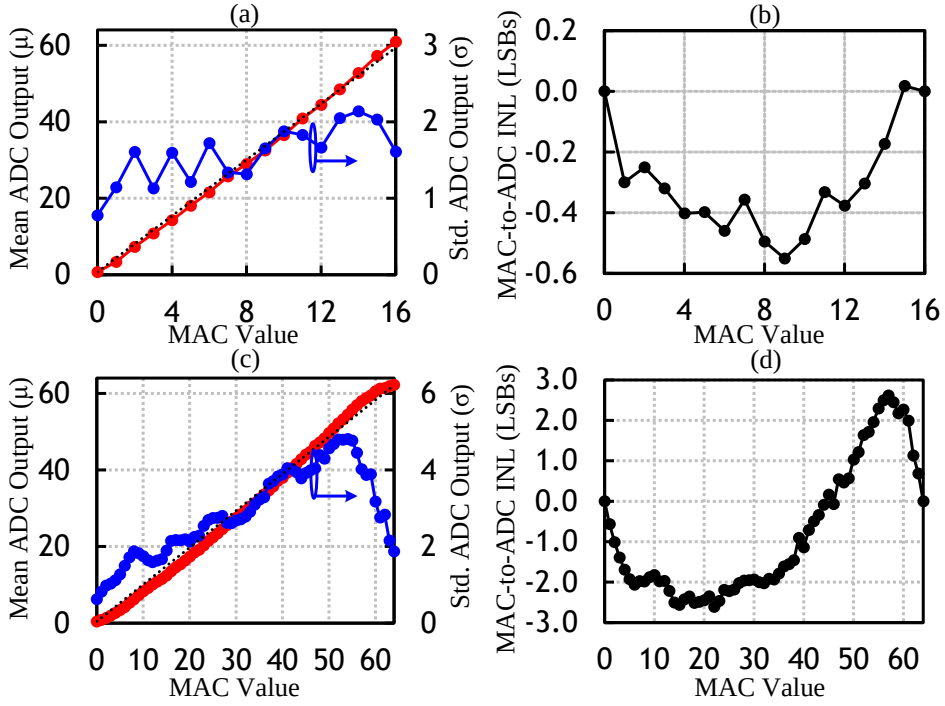


Figure 3.26: (a) 16-cell-mode measured column CIM output (ADC code) linearity across true MAC output values showing standard deviation from 1k samples, 1.1V. (b) Integral non-linearity (INL) of this transfer function from true MAC value to ADC output code. (c) and (d) are for 64-cell-mode.

all-zero condition. This characterization uses OTP so that it is not affected by off-state current to give a clearer picture of the raw front-end performance, and I_{OFF} cancellation is shown separately. Data for the 8, 16, and 32-cell modes are shown in Figure 3.25.

Compared to the first two macros, the TIA using higher feedback gain and having the

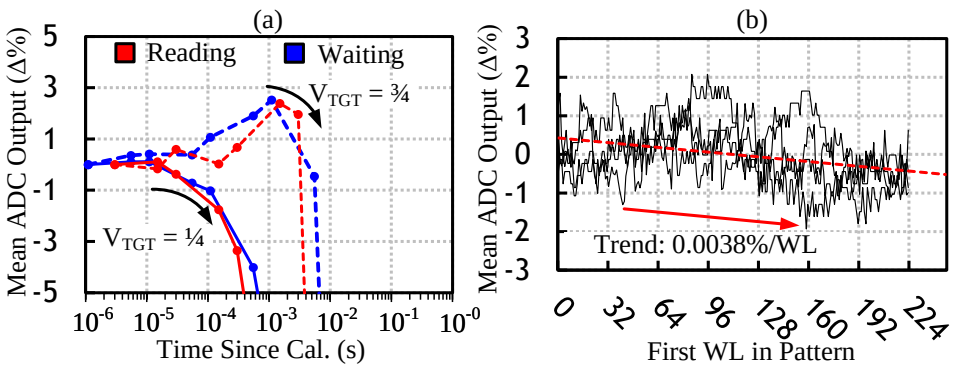


Figure 3.27: Key measurements for the TIA. (a) Measured calibration retention time for two V_{TGT} DAC settings while constantly reading or reading just for this measurement. (b) Measurement of sensitivity to IR drop by shifting a 32-cell pattern vertically down the BL (across the WLs). All at 1.1V. Here and elsewhere, for non-RMSE-related measurements, ADC read-outs are repeated 1k times each and averaged to remove local measurement noise.

ability to sense around array parasitics promises improved linearity. Figure 3.26 shows linearity and standard deviation in the ADC output for the 16-cell- and 64-cell-modes. Mode here refers to the width of the input vector, which is the maximum number of allowed active WLs. INL is calculated in terms of the step size, in ADC LSBs, due to one on-state selected cell. For example, the nominal/ideal step-size for 16-cell mode must be 64 ADC LSB steps divided into 16 output state steps, or 4LSBs per state. The real number is calculated from data and will be slightly below 4; the reported INL plot thus measures deviation from this perfect step at each output state mean. This is slightly different from, for example, ADC characterization but it captures the deviation of the measured readout from a linear mapping of MAC value onto ADC code.

Focusing on the current-sensing characteristics first, Figure 3.27 shows key measurements for the offset-cancelling TIA: retention time allows approx. 10kHz refresh, and total mean shift for 32 selected LRS cells due to IR drop is just under 1% when shifting the test pattern from the top of the BL to the bottom. This test shows how the Kelvin approach combined with the deliberate placement of the SL VSS tie can mitigate the issue of location-dependent IR drop. The effectiveness of this approach in eliminating the output-resistance bottleneck to improve linearity was shown by the linearity test (Figure 3.26).

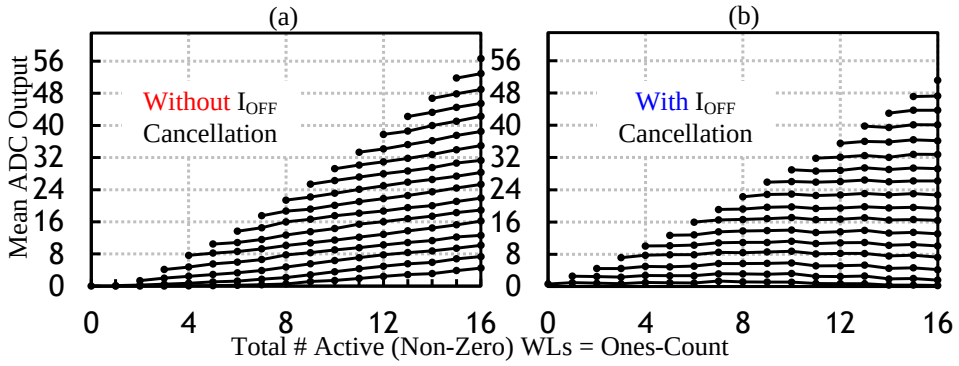


Figure 3.28: I_{OFF} cancellation characterization. Measured 16-cell CIM (a) without and (b) with I_{OFF} cancellation. All at 1.1V.

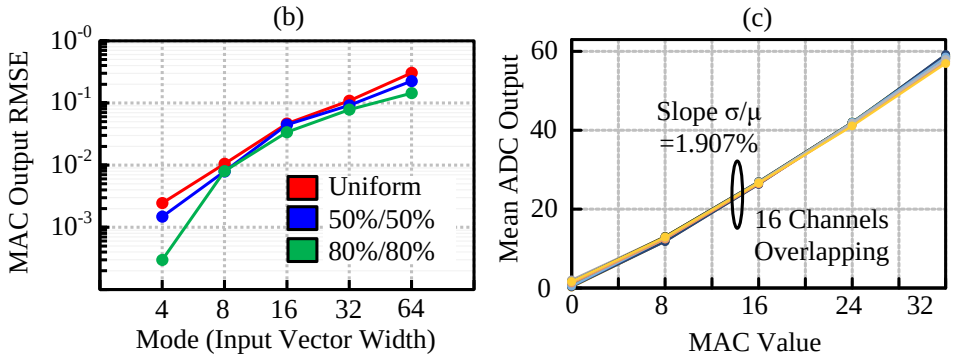


Figure 3.29: Accuracy analysis. (a) Measured column RMSE weighted for three different input/weight data statistics settings. 50% and 80% refer to bit sparsity. 1k samples per output mode per output state. (b) Measured 32-cell CIM transfer function across 16 read channels in macro. All at 1.1V.

The next area for characterization is I_{OFF} cancellation. Figure 3.28 (a) and (b) show the behavior with and without cancellation. The goal is to achieve flat lines left-to-right. Since linearity degrades more significantly for higher current loads (Figure 3.26), the technique is most effective for lighter loads (left side).

Concluding the characterization of non-ideality mitigation, Figure 3.29 (a) and (b) show end-to-end channel root-mean-square error (RMSE) and channel variation post-calibration. RMSE is calculated by binning the ADC outputs using hard thresholds, for example, into 17 bins for 16-cell-mode. RMSE improves for low-parallelism CIM modes and higher sparsity (see left-to-right trend from Figure 3.26 (a) and (c), secondary axis). Finally, channel variation, shown in Figure 3.29 (b) aligns with the simulation shown in Figure 3.20 (c)

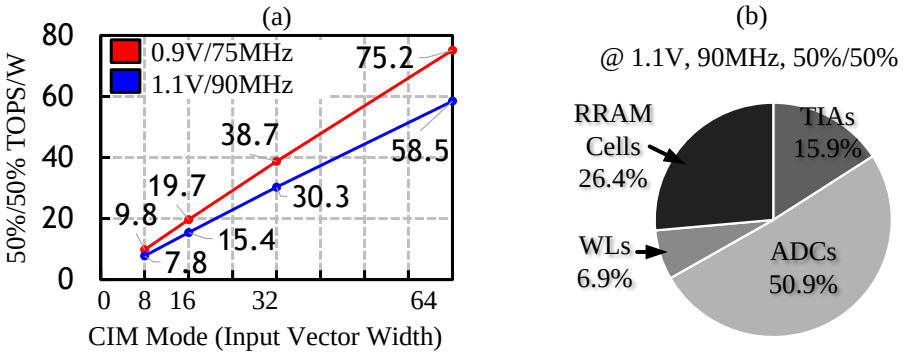


Figure 3.30: (a) Efficiency modeled using static power from measurement with added dynamic power from simulation, to better estimate single-macro efficiency from within the large integrated system. (b) 64-Cell power breakdown from modeling and simulation. Sparsity of input and weight bits are 50% and 50% in (a) and (b).

indicating little added variation due to sensing resistor, cell, or write-circuit drift. The 16 overlaid channels overlap near-perfectly after trimming ADC offset, with measured slope standard deviation just 1.91% between channels. This is well within the designed envelope and indicates successful operation of the offset-cancelling system.

The last characterization point, as with the first macro, is energy efficiency. Figure 3.30 (a) shows estimated efficiency using measurement and simulation, and (b) shows read power breakdown from modeling and simulation. In 1b/1b terms, where each active WL contributes 2 OPs × 16 channels, efficiency ranges from 7.8 to 58.5 TOPS/W at 1.1V (9.8 - 75.2 @ 0.9V). Sparsity is 50%-in and 50%-weight. Higher CIM modes greatly improve efficiency. The input buffer supports up to 256 active WLs, allowing higher efficiency at the cost of RMSE. Efficiency is sensitive to overall configuration of the macro, and simultaneously achieving good accuracy and strong efficiency is challenging.

3.5 Conclusion

This section provided background on CIM with RRAM and introduced a set of important challenges both in terms of the big-picture design goals and specific readout non-idealities. Two macro designs, both of which were implemented and characterized in silicon, were then presented. A first macro design was introduced that improved on the predecessor

work by increasing density and improving readout linearity. The second macro maintained density but directly addressed key non-idealities, including IR drop, off-state current, and channel mismatch. These solutions were then characterized using test-chip measurements, confirming their functionality in silicon.

CHAPTER 4

AN 8-BIT DIGITAL ACCELERATOR WITH RRAM

The dissertation work presented up to this point has focused on analysis of CIM and subsequently on the design of current-summing CIM macros using a foundry RRAM technology. However, the analog current-summing CIM approach is limiting in that it works within an accuracy/efficiency trade-off space which was partially shown by the work just presented. Current-summing also increases peripheral area overheads. This last work approaches RRAM from a different angle by focusing on what RRAM enables *aside from* current-summing: density due to the 1T1R structure and low leakage power due to the non-volatility. In light of this, this chapter describes a hierarchical module-based design with ten 8b digital matrix units each with 0.5MB of dense RRAM weight storage connected by a ring network. First, an introduction section will motivate all-on-chip inference with RRAM. The next section will describe the state-of-the-art and present the goals for the design. Next, the design will be introduced from the macro level up to the system level. Finally, a measurement setup and result measurements will be presented before a conclusion to summarize and end the chapter. The terminology 'digital' will be used to quickly distinguish the macro and system work here from the previous work using analog CIM.

4.1 Introduction

This work forms the concluding project for the PhD work. It resulted from another opportunity to use the foundry 40nm logic process [59] with embedded RRAM that was described in the introduction of chapter 2. While the previous work focused on building CIM macros that were then incorporated into larger system-level projects with help from a lab colleague, this work was an end-to-end effort from the macro/mixed-signal level up to the system level. This involved successively building layers of abstraction starting with

a new set of peripherals to implement a custom RRAM macro (traditional binary read-out, without current-summing CIM) around the foundry-provided memory cell array, through a module layer implementing a tile-able matrix unit with a ring interface containing a large block of RRAM, and finally through the system layer with ten such modules implemented and attached to a customized serial-peripheral interface (SPI) for testing and measurement.

The design work for this project focused on newly formulated goals that diverge slightly from the goals for CIM. The first goal was to maximize memory subsystem performance following traditional figures of merit: access energy in pJ per bit; access bandwidth in gigabytes (GB) per second; density in mega-bits (Mb) per square millimeter (mm^2) at the macro, module and system levels; and bit-error rate (BER) for low-cycling cells at the module level. This design goal is motivated by the concept of all-on-chip inference. Density is key to aid in fitting as large a model as possible onto the chip to improve the application performance, while BER is key to avoid degrading application accuracy or requiring large ECC overheads that degrade density. Performance (pJ/bit, GB/s) are key to avoid bottleneck-ing overall inference performance and to avoid requiring more layers of buffering and thus degrading density.

The second goal was to design a configurable overall system with good end-to-end infrastructure that can be readily programmed to test application performance while also remaining flexible for research-related testing. This goal was met at design-time by using a programmable VLIW core, designed mostly by a lab colleague and adjusted to integrate into the design, and by using a custom SPI protocol to broadcast testing instructions across the chip. This goal has been partially met at testing-time through the design of a programmer-friendly compact PCB that can be plugged into a USB port without needing external equipment. This concept builds on the work done for previous RRAM test-chip testing PCBs by a lab colleague but is fundamentally distinct in terms of parts used and in important areas of the design philosophy.

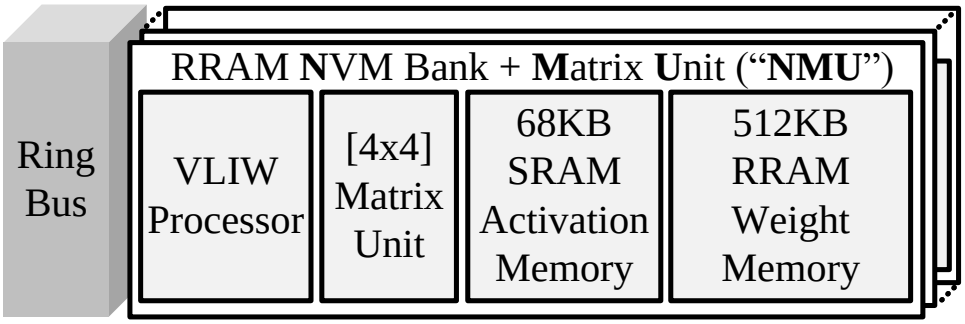


Figure 4.1: Summary view of the digital test chip. The test-chip consists of 10 matrix units with 0.5MB RRAM, 68KB SRAM, a 4x4 8b matrix multiplier, and a VLIW processor to orchestrate computation. They are linked together using a data ring.

4.1.1 Digital Accelerator with RRAM in 40nm CMOS

In terms of the immediate research goals of this project, this design was successfully taped-out and accepted for publication. The RRAM-based on-chip inference accelerator part of the test-chip, summarized in Figure 4.1, has been measured with all key sub-components working as expected. At V_{MIN}, when it is most efficient, the inference subsection of the chip achieves an efficiency of 0.84 TOPS/W end-to-end. 5MB of RRAM are implemented at an *overall* density of 2.07Mb/mm^2, including the full chip area. The memory subsystem performance is a major highlight: memory access energy improves 3.79$\times$ over prior work to 0.256pJ/bit, and peak NVM bandwidth improves 8.42$\times$ over prior work to 12.8GB/s before any normalization. In summary, NVM density, NVM access energy, and area-normalized NVM bandwidth were each improved at least 3$\times$ compared to prior all-on-chip inference accelerators. This strengthens the advantage of all-on-chip computing vs. going off-chip to DRAM or Flash by creating a more significant margin in terms of bandwidth and energy. In a retentive standby mode, the large ratio of RRAM to SRAM leads to just $110\mu W$ of power.

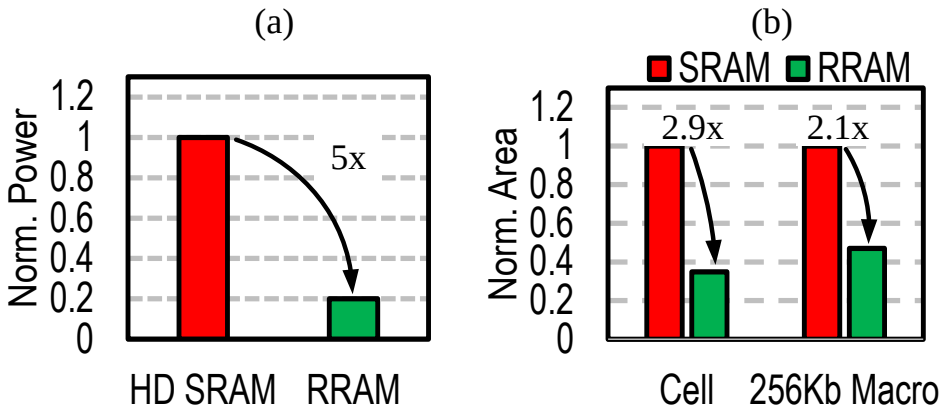


Figure 4.2: Comparison of large SRAM and large RRAM in 40nm using simulation for a 256Kb macro at 0.9V. (a) shows power-down power, where RRAM can show a 5× improvement. (b) shows density, where the implemented RRAM macro in this work gives a 2.1× net improvement. The density improvement at the cell level is 2.9×.

4.2 Background and Challenges

From observations of prior work, two challenges can be defined. These two challenges are used to steer the design of the test chip. First, application accuracy requirements can preclude the use of CIM and can require large edge networks, placing an emphasis on on-chip NVM density. Second, there is a trade-off between faster and more efficient memory accesses, when NVM banks are distributed among logic, and overall NVM density (density is typically maximized by using large contiguous arrays to amortize peripheral area). These challenges will be described in the second and third subsections. First, the next subsection will introduce the potential density and leakage advantages of RRAM that make it an attractive technology for this design, irrespective of design decisions.

Figure 4.2 shows the leakage and density benefits of RRAM in 40nm CMOS for the case of 256Kb memory macros. RRAM is compared to SRAM. The comparison is not exact, since the RRAM macro built for this work has some elements, such as write logic and pre-decoders, implemented outside the macro (off-chip and in RTL). Still, there is clear advantage to RRAM with leakage power improving $5\times$ and overall macro density improving $2.1\times$ for this example. To show the point further, by calculating using the full

area of the 18-NVM-macro module (shown later, in the design section) including other macros and logic, the NVM density per-256kb remains almost $1.5\times$ that of just the SRAM macro. This comparison clearly includes all of the RTL-based pre-decoders for all of the macros, but also other components, so it lower-bounds the density advantage of RRAM. Furthermore, the leakage power of the RRAM macros is not fully optimized; it could be closer to zero, but was deemed sufficiently low at design time to preclude further power gating. Due to these advantages, RRAM is a promising technology at 40nm for all-on-chip inference.

4.2.1 Challenge: Application Accuracy

This detail was covered in the previous sections. To summarize, application accuracy influences the design choices for this work in two ways: digital MACs are used in place of CIM to avoid accuracy trade-offs and improve macro-level density, and NVM density is prioritized across layers in the design to increase the number of parameters that the application network can use.

4.2.2 Challenge: Memory Structure Trade-Offs

The second challenge is illustrated in Figure 4.3. The idea is that prioritizing density and prioritizing performance (bandwidth/efficiency) lead to different designs. A purely density-oriented design would use the largest possible bitcell array to minimize peripheral area overheads. It might also group all of the sub-banks into one singular large section of the chip to avoid power routing overheads and other density losses from abutting NVM with logic. Under this paradigm, a relatively low-bit-width bus could be used to connect the memories to logic in place of wide, local logic. This can lead to lower bandwidth and higher power, since bits are traveling further to get from the NVM to the logic. Since the performance of NVM accesses is reduced, designers can compensate by adding higher-performance buffers. However, this requires a compatible dataflow to create the reuse

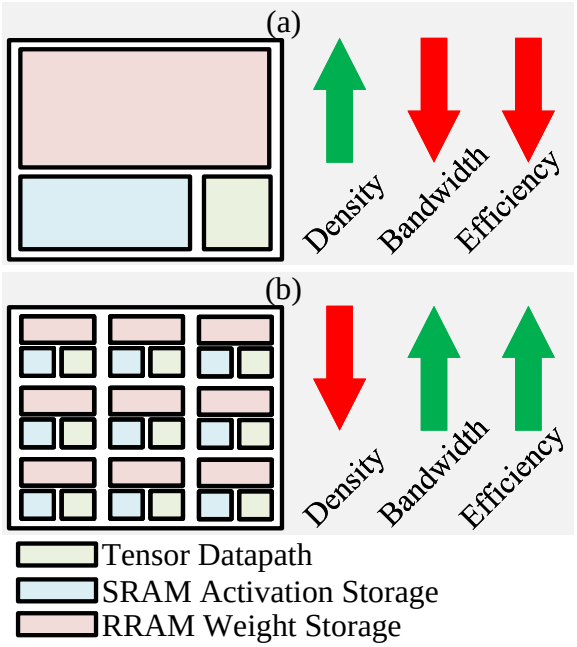


Figure 4.3: Illustration of (a) using fewer, larger NVM banks or (b) using many smaller NVM banks distributed among processors and MAC units.

needed for these buffers to effectively reduce access costs. Further, large buffers can add back the leakage and area overheads that using NVM avoids. In the extreme, this architecture resembles using off-chip memory.

The design goal is to distribute the RRAM sub-banks among logic to reduce access costs and maximize bandwidth, while minimizing density overheads. We address this challenge using an end-to-end design including an area-conscious custom macro and an optimized module-based digital design and implementation. The resulting design allows streaming weights into MAC units at full speed from eNVM without requiring buffers.

4.3 Design

This end-to-end design will now be detailed. The discussion will start with the custom digital macro design, then move on to the VLIW matrix processor. Finally, the top-level implementation will be described.

Table 4.1: Comparison of recently published RRAM and STT-MRAM digital macros. 'N/R' for not reported, 'CSA' for current sense amplifier, and 'OC' for offset-cancelling.

Work	Chou et al. [59]	Jain et al. [62]	Chih et al. [68]	Wei et al. [67]
Memory	RRAM	RRAM	STT-MRAM	STT-MRAM
Process	40nm	22nm FinFET	22nm Planar	22nm FinFET
Read Time	9ns	<10ns/<5ns	10ns/6ns	8ns/4ns
Voltage	N/R	0.5V/0.7V	0.68V/0.8V	0.9V/0.6V
Density	N/R	10.1Mb/mm^2	8.9Mb/mm^2	10.6Mb/mm^2
SA Type	CSA	OC CSA	Digital OC CSA	OC CSA
SA Ref.	Voltage	Resistance	Resistance	Resistance
Read Energy	N/R	<1pJ	0.8pJ	N/R

4.3.1 RRAM Macro

The design goals for the macro are for it to have high density, use low access energy, work at moderately high speed, and, most importantly, work reliably. Quantitatively, the goals for the design are better than three Mb/mm^2, much less than 1 pJ/bit, operation in the 100MHz range (half of logic), and for all of the macros in the design to read data without any peripheral-induced errors for fresh, low-cycling RRAM cells.

Prior Work

For digital RRAM macros, there have been several recently-published product-like foundry macros [62, 59]. Published STT-MRAM macros are also useful reference material because STT-MRAM has a small memory window, like RRAM, and requires a low-offset current-sensing SA [67, 68]. A summary of these four works is shown in Table 4.1. These works all use a 'butterfly' floorplan to improve density and show total read sensing times in the range of 4 - 10ns. They typically construct a pitch-matched column design with column MUXs, SAs, and write drivers in the shared middle area between two sub-arrays. Some circuitry is typically added on the outside (top and bottom) edges as well. All of these works read data out using a current-sensing SA with a clamping (common gate) stage connected to the BL. To overcome reduced sensing margin at the SA input due to the reduced memory window of the memory technology, [67] and [62] use an analog offset-cancelling phase

before the data-sensing phase as part of a three or four phase overall readout procedure. While not using offset cancelling, [68] uses digital trimming by scaling the effective width of the clamping device at each channel. The reference can be a local resistance, such as a ladder-based RDAC [62, 67] or a voltage bias on the gate of a current source [59]. In any case, the current-sensing first SA stage (pre-amplifier) is followed by a latching SA to produce read-out bits.

Macro Design Overview

The new digital macro needs a high-voltage write driver, an optimized HV and LV WL driver, and a readout signal chain. The details are summarized in Table 4.2. Some design ideas and sub-blocks are re-used from the CIM designs in chapter 3, but almost all blocks are at least re-formatted to fit the new macro. The write drivers are almost identical, with re-worked wiring and voltage schemes, while the WL drivers are fully re-worked to be more optimized for the digital (single-WL) use case. The readout scheme, including the SA and timing generator, is new.

The macro floorplan with a single pitch-matched column highlighted is shown in Figure 4.4. This subsection will now walk through the design of the read components, WL drivers, write driver, and finally the RTL wrapper.

Readout: SA and Timing

For eNVM technologies like RRAM with limited memory windows, the most distinguishing part of the readout circuitry is the current-sensing front-end. Current sensing was preferred by all of recent designs as discussed above. A symmetrical current sensing front-end employs feedback and therefore offers improved robustness compared to a voltage-sensing approach. Furthermore, since there is a risk of read instability if the BL voltage exceeds $\approx$300mV during sensing, the use of a clamping device during read is attractive.

Within the framework of symmetrical current-sensing, two questions emerge: what

Table 4.2: Summary of custom digital macro design.

Characteristic	Implementation	Description
Memory	RRAM	Foundry 1T1R Shared SL Array [59].
Process	40nm	ULP (0.9V) Planar CMOS.
Read Time	10ns/5ns	Target read cycle time at 0.8V and 1.1V.
Voltage	0.8V - 1.1V	Shared VDD for logic and all read peripherals.
Density	4.58Mb/mm^2	Some components are in external RTL or off-chip. Described below.
Timing	In-Macro	Custom-digital timing generator in each macro. 2 logic clock cycles per read.
Write	Driver in Macro	Voltage-based. MUXing and polarity controlled in-macro. Voltage generation off-chip. Sequencing is in RTL. Write control is off-chip (SPI).
Level Shift	In-Macro	All level shifting is internal to macro using custom schemes described in text.
SA Type	CSA	Carefully sized CSA to avoid needing OC.
SA Ref.	Resistance	Per-channel 4b ladder RDAC using process resistor (not RRAM).
Read Energy	0.342/0.256 pJ/bit	Measured in silicon at 1.1V/0.8V. Measurement setup described in text.
Power Down	$\approx 1\mu W$	Power down (PD) disables internal bias generation.
Best BER	0 Errors	Pre-ECC. Measured for 4.7M freshly written, low-cycling cells at 27C.
Width	16 R/W	16 physical read channels and 32 physical write channels. Only 16 write channels used at once.
Addresses	16,384	512 physical rows by 512 physical columns split into 16b words.
Format	4x 256x256	Four 64k-cell physical arrays in a butterfly floorplan.

kind of reference to use and whether to use offset cancelling. For the first question, voltage-based (locally, current-based) or resistance-based schemes are available. Voltage-based schemes generate a reference voltage once inside the macro using shared circuitry then pass that to each channel's SA for comparison. This voltage can then, for example, bias a current-source transistor to provide a current-domain reference local to each SA. Resistance-based schemes use a resistor local to each SA to provide a local reference.

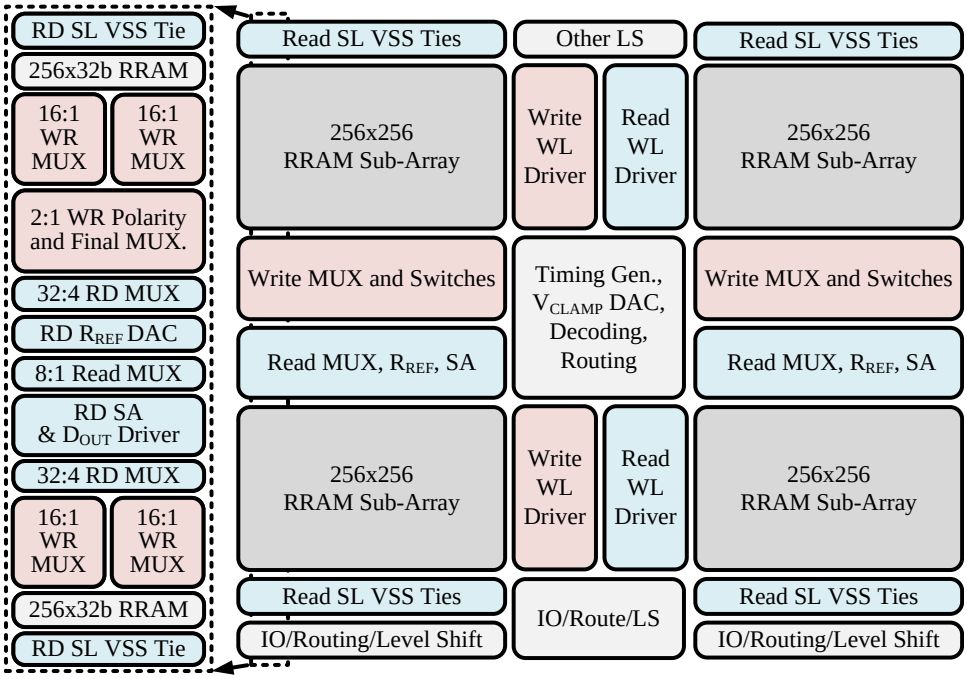


Figure 4.4: Overall floorplan and single column floorplan for 16,384-line 16-IO RRAM macro.

Voltage-based schemes can offer smaller channel area or finer tune-ability of the reference since a shared, higher-precision DAC can be used to generate the voltage before buffering it for use across the macro.

However, voltage-based schemes can be less robust since, broadly speaking, they operate in the voltage domain while actual memory cells that are clamped to a specific bias point operate natively in the current domain. Transient or permanent net currents along the supply metals can lead to changing V_{GS} for the current source transistor, which can lead to noisy or systematically skewed reference currents along the array. Random variation in the current source transistor can also have this effect. Another issue is that these schemes break the symmetry of the current sensor: if there is noise in the clamping voltage, for example, then this is likely to affect the current from a resistive device (RRAM) differently from a saturated device (transistor) leading to higher supply and bias noise sensitivity. Since robustness is the fundamental goal, this design uses a local resistive DAC (RDAC) reference.

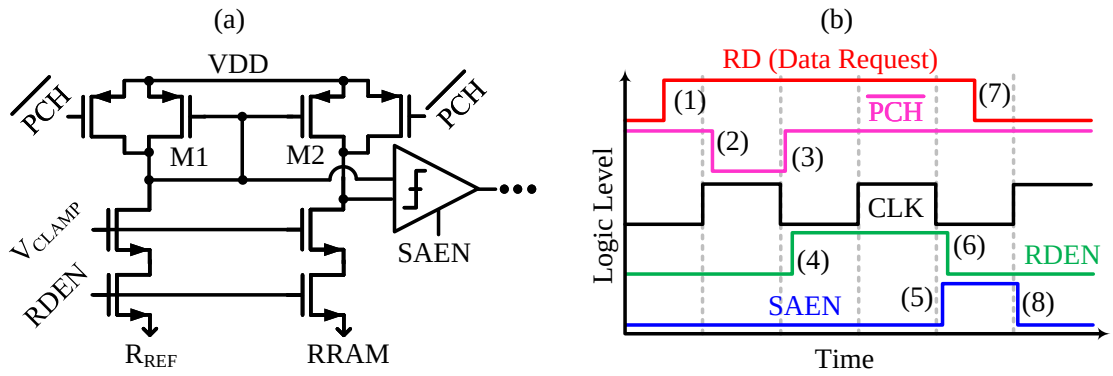


Figure 4.5: (a) Implementation of resistor-referenced current-sensing SA with clamping (common-gate) input loaded by a current mirror, followed by latching second stage. (b) Timing diagram for the sensing procedure for the case of the VLIW core requesting a single read. The two-core-clock cycle has the following phases: 1) Core requests read, (2)-(3) SA pre-charges, (4) read current flows, (5) SA latches the decision, (6) read current stops, (7) core drops the read request, (8) read is done and the SA resets.

The second question, whether to use offset cancelling, was decided during design by using Monte Carlo simulation and adjusting transistor V_{Th} flavor choices and sizing until an acceptable variation was achieved. For this, $\pm 3\sigma$ 'corner' RRAM devices were modeled using model access devices built from re-sized logic transistors in the PDK and ideal resistors. These fixed corner estimates for RRAM were used with the Monte Carlo variation models for the transistors provided in the PDK to estimate the robustness of the readout circuitry at design time. The RRAM 1T1R models were calibrated to measurements made using the previously implemented macros. Since adequate SA matching performance was achieved just using re-sizing, offset cancelling was omitted from this design. This analysis depends on the worst-case (variation-affected) memory window for the NVM technology.

The SA implementation and timing diagram is shown in Figure 4.5. Pre-charge ($\overline{PCH}$) is included to improve the transient response. The mirror pair, M1/M2, load the common-gate clamping devices and perform the current comparison. Since M1 (in the reference arm) is diode connected, the input to the latch will be at least $V_{GS,M1}$ below VDD. The mirror pair should match in the current domain, and therefore a longer device is preferred. By giving the input devices in the latch a W/L greater than that of the mirror pair, which

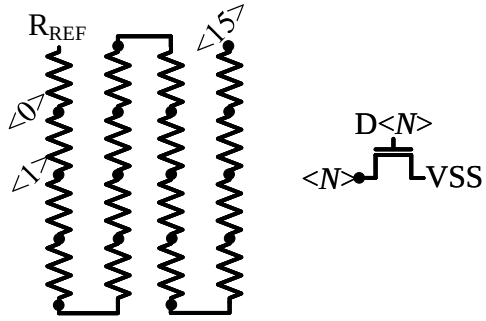


Figure 4.6: Programmable reference resistor (RDAC) implementation.

thus occurs naturally, both a V_{Th} shift and reduced effective current density lead to a significantly reduced V_{GS} for the latch's input pair relative to M1/M2. This means it is safe to use a P-type input for the latch as there will generally be sufficient overdrive. The latch is a 'StrongARM' design [153]. The RDAC, Figure 4.6, is a ladder of process resistors. The reference node of the SA connects to the top of the ladder, and the ladder is tied to VSS at one of 16 possible steps, labeled zero through 15, providing four bits of control.

The timing generator is included in the center control block of the macro and is responsible for generating the pattern in Figure 4.5(b). It splits two cycles of the input clock into four timing phases. Using two cycles not only provides more time for the read system to operate but also natively provides these phases since the timing generator is designed to trigger on both positive and negative edges of the clock. Further, the most critical timing component, the sensing development time, is easily set by the *period* of the clock, rather than the pulse width. The clock generator takes two input signals, the clock and a read enable flag, and uses two opposite-polarity back-to-back flip-flops to generate four distinct phases. Reset occurs whenever the read enable flag is dropped. A set of logical and delay elements driven from these four phases time the three output signals: SAEN, RDEN, and $\overline{PCH}$. RDEN and $\overline{PCH}$ do not overlap, but RDEN and SAEN do.

The read MUXing scheme consists of a first-stage thick-oxide NMOS switch dedicated to protecting the following core-voltage MUX stages during write, followed by a 32:4 stage. There is one each per-channel for the top and bottom array, giving 64:8 total. 64:4

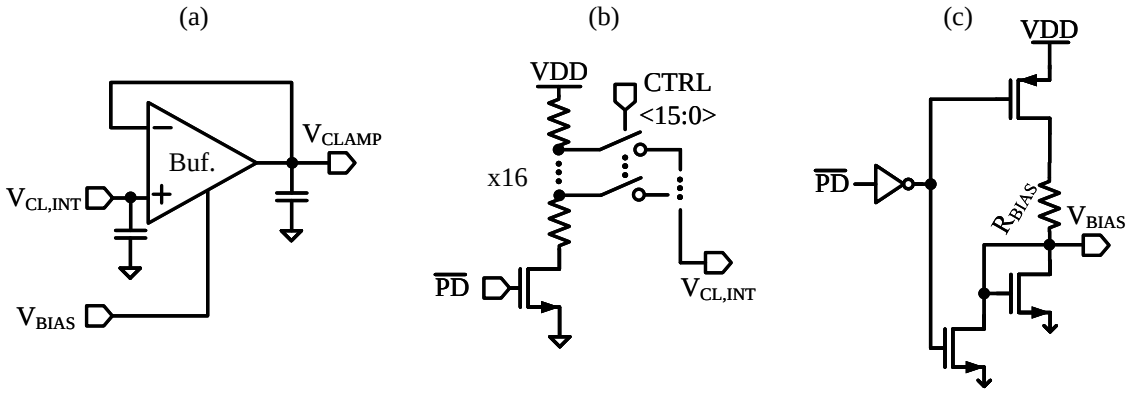


Figure 4.7: (a) Overall V_{CLAMP} generation scheme using a 4b DAC for $V_{CL,INT}$ and a classic 9T OpAmp [150] to produce the buffered clamping voltage output. (b) 4b resistor string voltage DAC with power-down switch. (c) Low-cost bias generator with power-down switch.

is conceivable, since only one WL will be active across both the top and bottom RRAM arrays; however, this doubles the capacitive loading. The last stage is then an 8:1 MUX local to the SA. The SLs are terminated to VSS during read at the opposite ends of the arrays as in the previous two designs to decrease positional dependence of the BL/SL resistance.

V_{CLAMP} and Power Down

Besides the timing generator, DCAP, and various decoders and LS blocks, the middle-circuit area of the array contains a voltage DAC and buffer for locally generating the clamping voltage, V_{CLAMP}, that is needed for the current sensing pre-amplifier in Figure 4.5(a). A DAC is preferred to avoid routing an analog signal across the test chip while retaining adjust-ability. The small challenge introduced is that the DAC will require a trickle current to bias the resistive ladder and a buffer to drive the 16 IO channels. The scheme is shown in Figure 4.7. The buffer is a 9T OpAmp [150] to provide a wide output range using a single bias voltage, which is produced using a low-cost resistor-based bias generator. The voltage DAC is a resistor ladder, adjusted to provide an output range between 20% and 95% of VDD.

As described above, when the read request signal is low the timing generator will re-

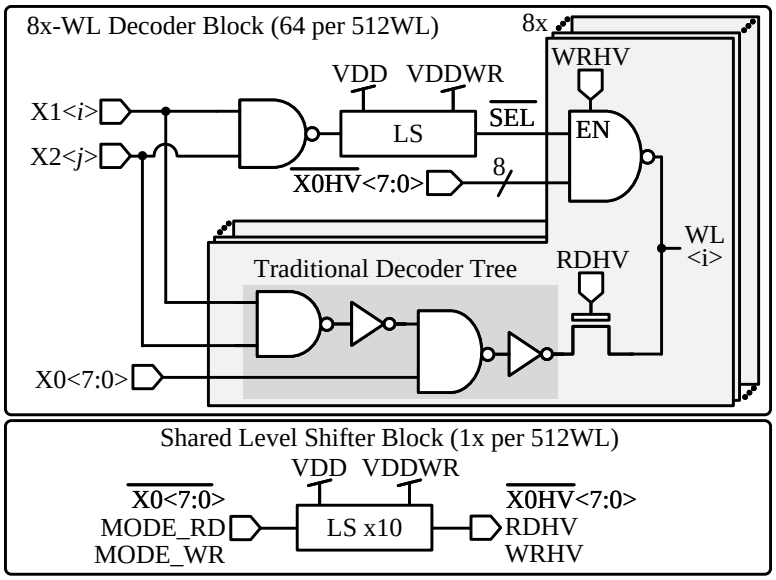


Figure 4.8: Level shifter scheme with the first two decoder layers (X1, X2) shared between HV and LV (write and read).

main in the reset state and the macro is in a standby mode. The power-down (PD) mode provided by the V_{CLAMP} generator in Figure 4.7 removes the analog bias power overheads: in PD, the ladder, Figure 4.7(b), is shut off while the bias voltage, Figure 4.7(c), is pulled to the negative rail to shut down the NMOS tail bias of the buffer in Figure 4.7(a). The single-stage buffer is output-stabilized and provides adequate stability and performance across a wide range of tail bias currents.

WL Drivers

The requirements for the WL drivers for this macro are different from those for the WL drivers for the previous (current-summing CIM) macros in that they only ever need to drive a single WL in any mode, read or write. Also, the butterfly format means the WL drivers are shared across twice as many columns for this macro as in the previous, single-array CIM macros. This does not affect functionality but improves area efficiency and restricts metal usage (a horizontal metal now needs to run left-to-right across the full driver).

The new scheme, shown in Figure 4.8, is designed to provide a merged HV and LV WL driver system with a reduced number of vertically-routed pre-decoded control wires

while not requiring a significant LS overhead. HV here refers to the VDD_{WL} rail, which is outside of the core voltage range to allow RRAM cells to be written. The key idea is to place one LS in the tight WL driver layout area for every eight rows. Then, the final 8:1 decoding is performed in the HV domain using HV signals. The LV driver is implemented as a three layer decoder, with each layer driven by eight ones-hot pre-decoded signals from RTL. Note that $8 \times 8 \times 8 = 512$, as needed for the 512 physical WLs.

These ones-hot buses and are labeled X0, X1 and X2, where X0 is the lowest level (*i.e.* changing the value of X2 changes which of the eight 64-wide blocks are selected). Considering just the LV WL signaling, these wires run vertically over the WL driver area in the middle of the macro requiring 20 global vertical routing channels in the top and bottom half-arrays. In the implemented design, X1 and X2 are fully shared among HV and LV. Considering a group of 8 adjacent WLs, X1 and X2 provide enough information for the local decoder to determine if the group is selected. In the LV case, a logical AND operation between X0, X1, and X2 then determines if a WL should be driven high. In the HV case, for write, a logical AND operation of X1 and X2 produces a local select ($\overline{SEL}$) that is then level-shifted using the eight-way-shared, in-WL-driver LS. A logical AND between this select and a separate X0 signal ($\overline{X0HV}$) then determines if the WL is driven high for write.

The scheme which covers LV and HV WL signaling thus requires approximately 28 vertical routing channels vs. 20 in the LV-only design, eight being HV. A total of ten LS are needed in the middle-circuit area of the macro with 64 needed inside the WL drivers, each pitch-matched to eight WLs as described. The extra two HV signals control (1) a thick-oxide NMOS pass device to protect the LV WL driver and (2) a series pull-up PMOS in the thick-oxide NAND gate to disconnect the HV path during read. The overall scheme is compatible with dense physical design in that it requires only a limited set of pre-decoded ones-hot signals in addition to an LS that is amortized across eight rows.

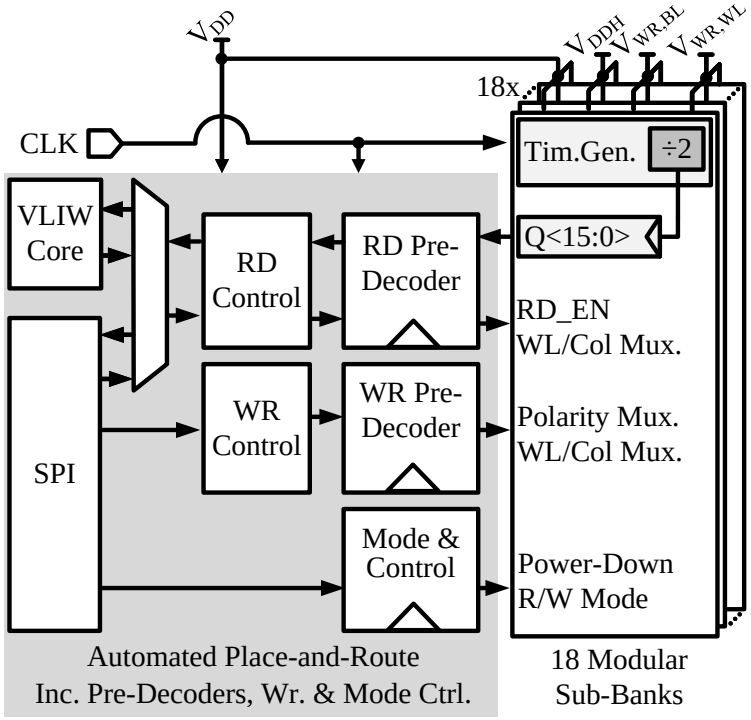


Figure 4.9: Scheme with tight integration of 18 macros and RTL-based wrappers, all sharing clocking and core-voltage VDD.

Write Driver

The write driver scheme for this work is the same as for the previous works except for largely physical changes. The write drivers for the top and bottom arrays are moved next to the SAs. The SL VSS ties are split off, as mentioned, and kept at the opposite side of the arrays from the SAs. The HV MUX control signals are shared for the top and bottom MUXes, with the control over which array receives the write pulse achieved using the WL drivers. That is, for write, the top and bottom arrays appear as a single 512-WL array. Two 16:1 first-stage column MUXes feed two write header units per 32 cells, as before (see chapter 3), with the RTL controller allowing one of these two headers (16 of 32 per macro) to send a write pulse at a time.

RTL Wrapper

To reduce area and design-time overheads by simplifying the design, the macros are tightly integrated with an RTL wrapper. This concept is shown in Figure 4.9. While the macro receives HV supplies for write, the core-voltage read supply, along with the clock for the timing generator, are shared with the RTL-based wrapper which is eventually implemented using automated place-and-route (APR). Most complex logical operations, outside of the local post-decoders for the WLs and the timing generator, are written and tested in RTL to allow fast design iterations for mode-control state machines, write sequencing, and pre-decoders. As a simplified example, during read, a 14-bit address is input from outside logic and decoded into two eight bit ones-hot column-decode signals (64:1, six bits of the address) and three eight bit ones-hot row-decode signals (256:1, remaining eight bits of the address). Write is controlled off-chip using SPI, while read can be controlled by the VLIW cores, described next, or by the SPI port for direct RRAM testing. A carefully tuned constraints file and thoroughly verified cycle-accurate model for the custom RRAM memory macro were used to increase the probability of achieving a working design in silicon.

4.3.2 VLIW Matrix Processor

The broader RTL-based chip design is implemented in two APR steps. The first of these implements the NVM matrix unit (NMU) tile with a VLIW processor, 18 macros with RTL wrappers to form a single 0.5MB bank after ECC, and four other SRAM macros implementing three functional memory blocks. This first layer is described in this subsection, with the top implementation layer (a second APR step) described in the next subsection. The RTL for the NMU, including the overall architecture and critical test-benching infrastructure, was written in large part by a lab colleague before being modified and merged into the design for this project.

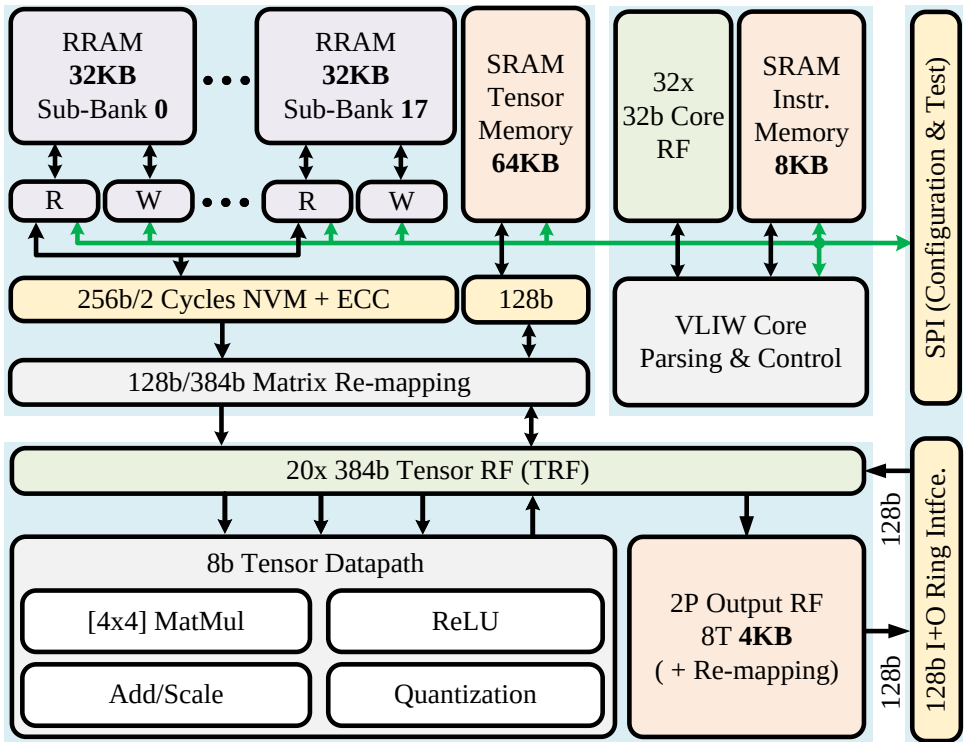


Figure 4.10: Overview of the NVM matrix unit (NMU) with 0.5MB RRAM, VLIW processor, 4x4 8b matrix datapath, 64KB tensor memory, 8KB instruction memory, and 4KB IO ring buffer.

NMU Overview and Resources

The NMU architecture is shown in Figure 4.10. The NMU prioritizes memory density, both NVM and SRAM: 75% of NMU area is non-redundant storage. The NMU is built around a 4x4 8b matrix datapath that computes one matrix multiply (4x4, 64MACs) or one processing operation (ReLU, add/multiply, bit-shift) per cycle on 16 matrix elements. MAC accumulation occurs into 24b. The matrix datapath interfaces with a 20-entry tensor register file (TRF) that stores one 4x4 matrix per entry, where entry zero is the static all-zero entry. The TRF manages the interface between 0.5MB RRAM weight storage, 64KB SRAM activation storage, the 4x4x8b ring interface connecting to neighbor NMUs, and the 8b matrix datapath. Any block that accepts matrix inputs can read from any entry in the TRF during any cycle, but each block, such as the RRAM bank, is assigned a subset of TRF entries that it can write to. This is a straightforward solution to avoiding contention

during TRF writes. Another subtlety about the TRF is that the large data storage blocks (RRAM and the 64KB 6T SRAM) work with 8b integers for a total of 128b per matrix word, while the tensor datapath accumulator works with 24b integers for a total of 384b per matrix word. Thus, writes to the TRF from different sources use the appropriate re-mapping scheme to store 384b correctly.

VLIW Processor

The NMU processor follows the load-store paradigm, as implied by the above discussion of the TRF, and is a loose extension of RISC-V [154]. VLIW is used to allow all of the resources to be coordinated in parallel, such as scheduling RRAM and SRAM reads into the TRF while using the TRF to feed matrix multiplies. Communication between NMUs occurs through the ring interface: each cycle, the NMU can read one matrix word from one neighbor and can copy another matrix word of data into its output buffer (a 4KB 8T 2-port register file, RF) from the TRF for its other neighbor to access. Ring accesses are requested by providing an address in the neighbor's 8T output RF. The VLIW core uses a separate 32b 32-entry core RF for program flow control and address computations and an 8KB (512-entry by 128b instruction) instruction memory to store local programs. The RRAM, 6T tensor memory, and instruction memory can all be configured for test execution over SPI. Program execution is initiated over SPI, and program status can be audited over SPI as well.

Figure 4.11 shows the breakdown of the 128b VLIW instruction and shows a simplified example program. The description here will work from left to right in Figure 4.11(a). SRAM load/stores refer to the 64KB 6T tensor memory, and require two 5b addresses plus an 8b immediate value. The source 5b address provides an entry in the core RF whose value is added to the immediate to index into the tensor memory. The address in the TRF to load/store to/from is provided by the other 5b address. The situation is identical for RRAM loads (but stores are not allowed during core operation, as the RRAM is read-only

(a)

Type	SRAM LD/ST	RRAM LD	Control	Tensor Math	Ring
Width	21b	24b	21b	24b	19b + 19b
Opcode	3b	1b	3b	4b	1b + 1b
&RF0	5b	5b	5b		5b + 5b
&RF1			5b		
&TRF0	5b	5b		5b	5b + 5b
&TRF1		5b		5b	
&TRF2				5b	
&TRF3				5b	
Imm.	8b	8b	8b		8b + 8b

(b)

(Header code sets up registers a, b, c, T1, T5, T6, T7)

SRAM LD/ST	RRAM LD	Control	Tensor Math
[a+3]→T0	[b+1]→T3,T4	a+4→a	(T1@T5)+T7→T7
[a+0]→T1	**NOOP (Tock)**	b+2→b	(T0@T6)+T7→T7
[a+1]→T0	[b+0]→T5,T6	**NOOP**	(T1@T3)+T7→T7
[a+2]→T1	**NOOP (Tock)**	**BNE**(a,c,-3)	(T0@T4)+T7→T7

Footer Code:

T7→[a+4]	**NOOP**	**STOP**	**NOOP**

Figure 4.11: (a) The breakdown of the 128b VLIW instructions. (b) Example of locally computing (no ring transfers) a local-output-stationary dataflow.

during inference) except that the RRAM requires two destination TRF addresses since it reads, post-ECC, two matrices every two clocks to match the SRAM bandwidth. Control instructions include branching, arithmetic, and program halt. Tensor instructions include matrix multiplication and the other operations described earlier. Four 5b TRF addresses are needed, since the full matrix MAC computes $(A@B) + C \rightarrow D$. Finally, ring operations have two components: 'send' and 'receive.' These can operate in parallel each are effectively separate sub-words of the VLIW instruction. They work like the previously discussed SRAM loads/stores, except that a 'send' is write-only and moves data into the local output RF, while 'receive' is read-only and pulls data from one neighbor's output RF.

The program shown in Figure 4.11(b) could be viewed as a point-wise (1×1) convolution on a 4-pixel patch across 4 output channels. In this example, some core RF entries have been named a, b, and c while TRF entries are simply numbered. The program runs a loop, with a and b increasing by 4 and 2 respectively in each iteration. It is assumed that a

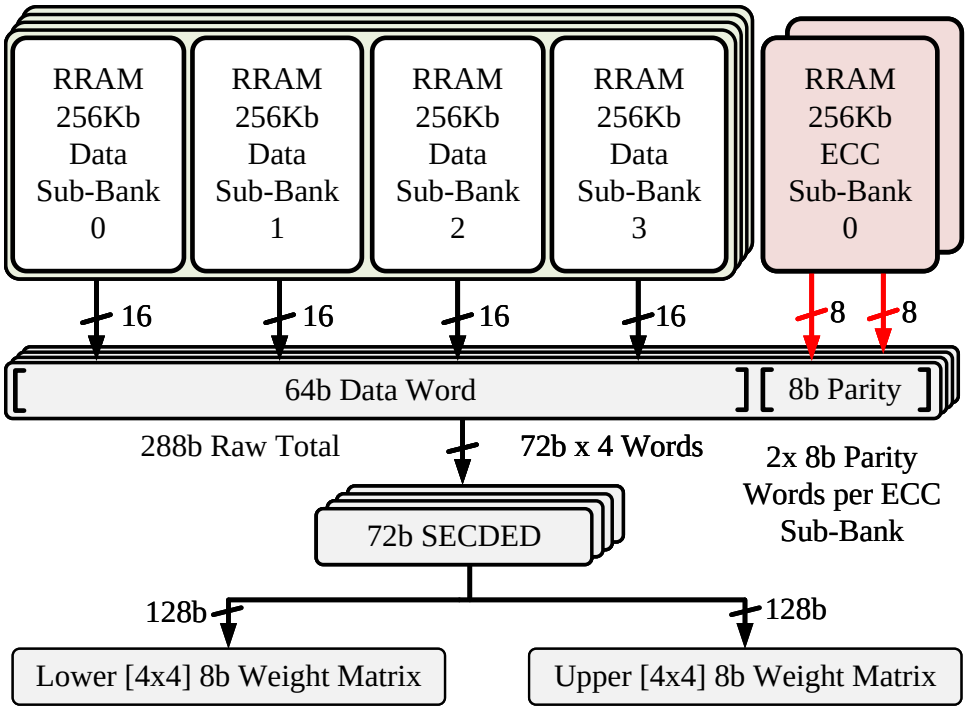


Figure 4.12: Organization and SECDED ECC scheme for 18 RRAM sub-banks forming a 0.5MB NVM bank in each NMU.

program header has set up a and b as zero and c as an endpoint. The program loops until a reaches c before running footer code, here just an exit instruction. During the loop, a is used to continuously pull new data entries from SRAM into the TRF while b is used to pull data from RRAM into the TRF. These entries are then matrix-multiplied sequentially while accumulating continuously into the 4-channel, 4-pixel 4x4 output matrix at 24b (TRF entry T7). This example does not necessarily track the actual pipeline delays of the VLIW processor since the point is to give an understandable program snippet.

RRAM Bank

As described above, 18 total RRAM macros (called sub-banks here) are organized to provide 0.5MB of storage with 256 usable (data) bits output, every two cycles, as two 128b matrices (=two 8b 4x4 matrices). The organizational scheme including ECC is shown in Figure 4.12. The 18 sub-banks are organized as four sets of four data sub-banks with the remaining two sub-banks providing ECC bits. Each set of four data sub-banks contains

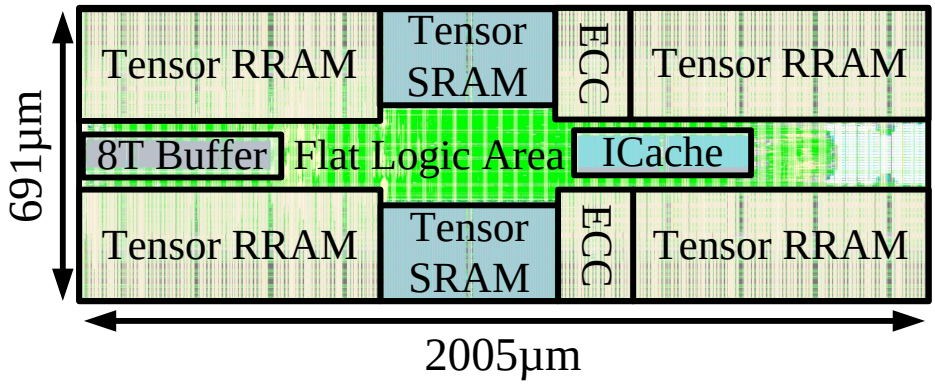


Figure 4.13: CAD screenshot of NMU physical design.

64b, and thus uses an 8b SECDED parity word to provide single-bit correction. The four 8b parity words are stored in the two ECC sub-banks. This adds up to the expected total of 288 bits ($18 \times 16 = 288$). During readout, ECC correction is computed on the four 72b words to produce 256 corrected bits, which are then split into two 4x4 8b matrices for storage in two TRF entries. The total post-ECC capacity is therefore 16,384 address lines $\times$ 256 bits per line which gives the expected 0.5MB. The bandwidth is likewise 256 bits per two clock cycles, or one 4x4 8b matrix per cycle on average.

Under this SECDED scheme with 64b data words, the computed threshold for achieving part-per-billion BER is about 18 measured bit-errors per bank (≈4.7 million cells). This is achievable for low-cycling cells. Implementations targeting greater cell endurance would need to tolerate more bit errors to extend array lifetime, and would likely benefit from a single more complex ECC scheme tolerating double or triple errors in the full 256b data word.

4.3.3 System Implementation

The NMUs are implemented as a rectangular tile, as shown in Figure 4.13. They are organized as five tiles on the left-hand and five tiles on the right-hand sides of the chip. They are mirrored and flipped so that the digital IOs of the tiles all face toward the center and so that the send and receive ring ports for adjacent NMUs align. The narrow, vacant core

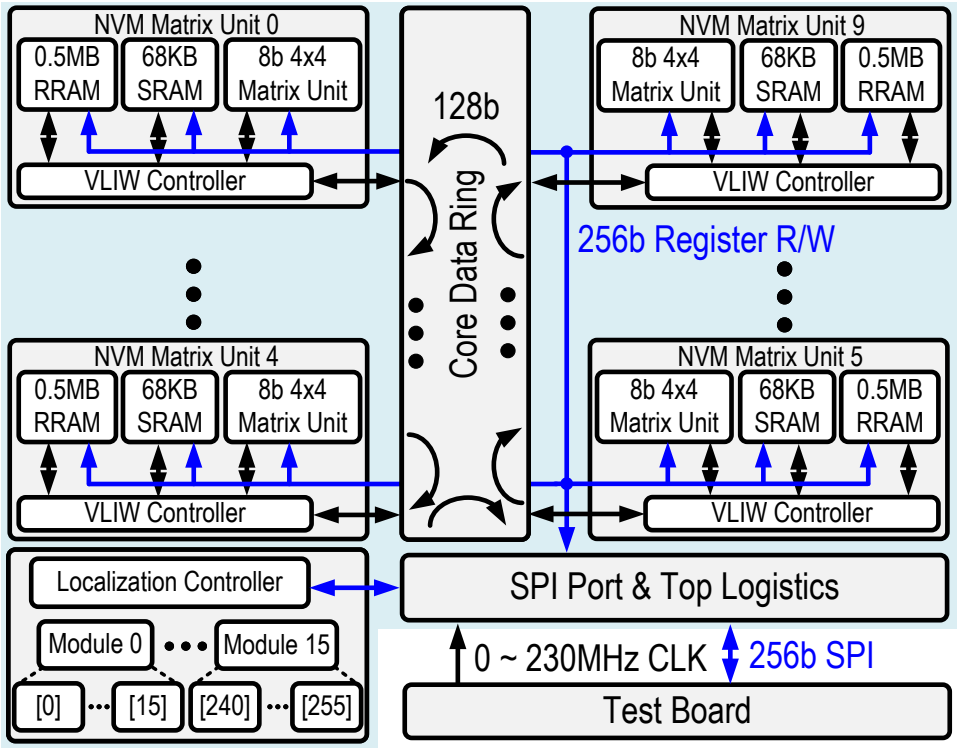


Figure 4.14: Top-level chip design 10 NMUs and localization block. The localization block, bottom-left, is a colleague's work and is not discussed here.

region in the middle then forms a data 'spine' for communication to the NMUs over SPI and between them over the ring.

For test and measurement, the system uses a single custom SPI port with 256b words to write wide VLIW instructions and matrix data with reduced overheads. The SPI is synchronized into the core clock domain after data word transmission completes and is broadcast to everything at once as 256 bits. A header dis-ambiguates the destination. At any given time, one block is privelaged to send data over SPI and subsequent SPI transactions will send out data from this block (if the block is actually configured to send data). Clocking is provided externally. A shared VDD is used for RRAM and core logic, as described, in addition to three HV voltage rails for RRAM write.

The top-level chip design is shown in Figure 4.14. This shows a block with a 'Localization Controller' in addition to the 10 NMUs and associated data ring. This block is not described here since it is a colleague's work and is not required to discuss the core

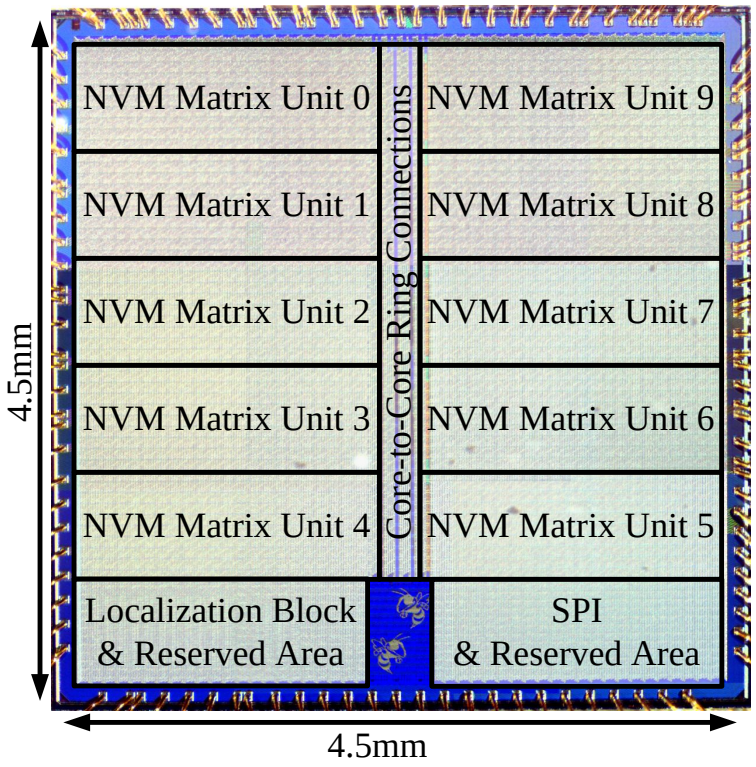


Figure 4.15: Die micro-photograph of the system with 10 NMUs.

RRAM-related work described here.

4.4 Context

Like the first two CIM macro designs, this concluding project was taped-out in 40nm CMOS with foundry RRAM. It includes the 10 NMUs with 180 total custom RRAM macros for 5MB of post-ECC storage. Also included on-die is the localization block mentioned above which is not described here. The die micro-photograph is shown in Figure 4.15.

4.5 Results

In addition to the test-chip design, an important part of this project has been the development of test infrastructure including a test PCB, microcontroller code, Python interface to the microcontroller, and test scripts run in Python on the host PC. This approach was

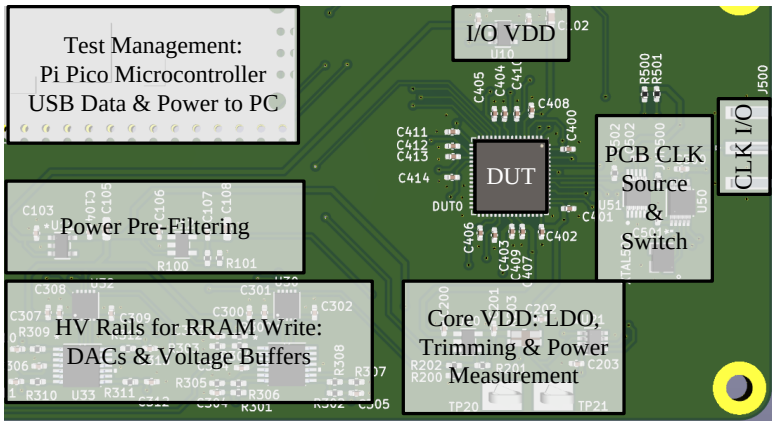


Figure 4.16: Zoomed CAD rendering of the test PCB for digital RRAM system with labels.

inspired by testing work completed by a lab colleague for the previous RRAM tapeouts, but the details were substantially different. The detailed measured results are important as well, and they have provided quantitative verification that the goals of the design were met.

4.5.1 Testing Infrastructure

A rendering of the test PCB is shown in Figure 4.16. The test-chip is packaged in a QFN64 package and soldered to the test PCB. A raspberry Pi Pico microcontroller board plugs into female headers on the PCB and provides digital control of the board, including high-speed communication with the chip over SPI. The USB+5V rail from the USB connection of the Pi Pico to the host PCB is used as the base power supply. Optionally, +5V and GND can be provided using a lab supply. Texas Instruments TPS2110PWR provides automatic switching between these sources. Two low-dropout regulators (LDOs, Analog Devices ADP122-3.3) provide 3.3V filtered rails for the test-chip and PCB. An Analog Devices ADP123 LDO provides a 2.05V reference rail on the PCB as well. This reference and the filtered 3.3V PCB rail are used with a 4-channel Analog Devices AD5304 DAC chip and then buffered by an Analog Devices AD8534 4-channel OpAmp chip to provide the programmable high-voltage rails for RRAM. Another ADP123 is controlled using a DAC channel to provide a trim-able 0.5V - 1.25V VDD for the test chip. This rail is monitored using a 50mΩ sensing resistor that can be read digitally using a Texas Instruments INA236

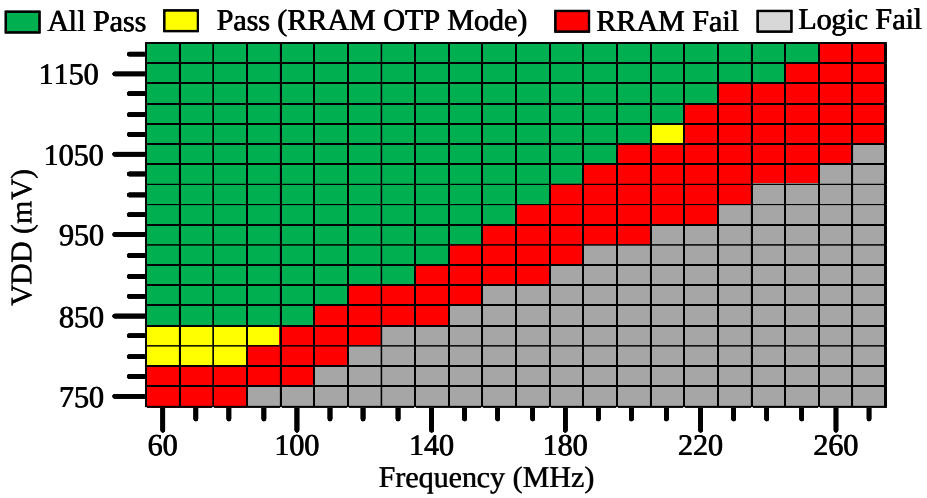


Figure 4.17: Measured voltage/frequency shmoo for end-to-end data path from RRAM read through MAC computation and write-back.

power monitor chip. This arrangement can be bypassed using a lab supply for measurement validation or when the power limit of the LDO is exceeded (≈300mA). Finally, clocking is provided either on-PCB via a Skyworks Si5351 clock generator or off-chip using a lab source. Switching between the clock sources uses a Texas Instruments TMUX1072.

4.5.2 Measurements

This PCB allows straightforward analysis of power and functionality across a wide range of voltages and frequencies. These measurements for the implemented test-chip will now be presented. An end-to-end voltage/frequency shmoo is shown in Figure 4.17. This test uses a test program running on the VLIW to perform matrix multiplication from RRAM and write-back to SRAM, to verify end-to-end core functionality. This shows two extrema: a V_{MIN} for maximum efficiency at 800mV and 80MHz, using OTP mode where RRAM cells storing zeros are never formed, and a F_{MAX} point for maximum throughput at 1.1V and 210MHz. Higher frequencies at higher voltages are possible but they exceed the nominal 1.1V process V_{MAX} and are omitted from further measurement.

Figure 4.18 shows measured chip power in the most efficient considered OTP mode, the most efficient considered non-OTP mode, and at the highest considered F_{MAX} point.

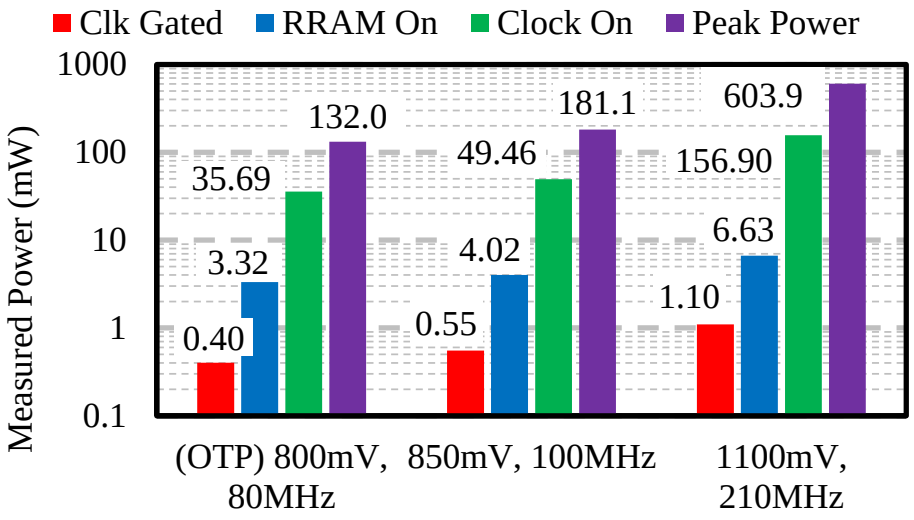
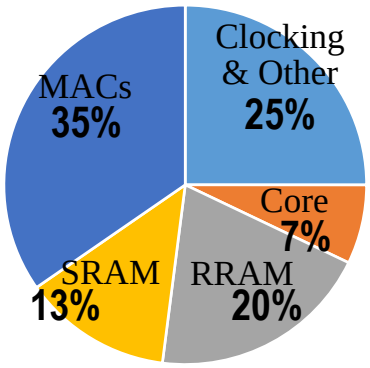


Figure 4.18: Measured power on core VDD rail across three voltage/frequency conditions and four test cases.

Due to the large ratio of RRAM to SRAM, power reduces to just 0.4mW at 800mV when the clock is frozen and the RRAM is in PD mode. This is a retentive state, since the SRAM retains data at 800mV, although it is also a functional state since logic and memory reads function at 800mV/80MHz. The most efficient non-functional retention voltage that was measured, not shown visually, is 500mV requiring just $0.11\mu W$ of power. The peak power test case uses a VLIW program that constantly reads from SRAM and RRAM while performing distinct matrix multiplications every cycle.

The VLIW system allows the marginal power due to specific sub-systems to be measured. For example, test program loops can be run that do not make RRAM read accesses, to establish a baseline power, and then loops that do make back-to-back distinct RRAM accesses can be compared to establish the marginal power added by making RRAM accesses. By measuring the marginal power in this way, a component-level breakdown can be produced from chip measurements without relying on isolated rails. Figure 4.19(a) shows the breakdown for the peak power test case at V_{MIN}. Figure 4.19(b) goes further and uses the marginal power to compute per-bit energy use, end-to-end, for accessing data from the 0.5MB RRAM banks or the 64KB SRAM banks in the NMUs. The 0.256pJ/bit figure due

(a)



(b)

	RRAM	SRAM
800mV 80MHz	0.256	0.171
850mV 100MHz	0.272	0.193
1100mV 210MHz	0.342	0.272

(pJ/bit)

Figure 4.19: (a) Power breakdown at peak power at 800mV/80MHz. (b) Measured local energy access cost from marginal power increase for the 64KB SRAM and 0.5MB RRAM banks in the NMUs.

to the marginal-power measurement at 80MHz is good evidence that the modular design with NVM close to matrix math data-paths has yielded improved per-bit RRAM access costs.

Aside from the peak-power test case, an energy efficiency test procedure was developed using 8b weights and activations that are normally distributed around zero, with activation matrices clipped to be 50% sparse and positive. The resulting efficiency measurement is shown in Figure 4.20. At the OTP-mode V_{MIN}, efficiency is 0.84 TOPS/W when operating realistically, reading from RRAM and SRAM with 3x weight reuse. This level of reuse is achieved using straightforward data-flows with a temporarily static weight matrix and three distinct input matrices applied serially, with the result accumulated into one of three distinct output 16x24b locations in the TRF. During less-typical data-flows, efficiency improves to 1.14 TOPS/W (transiently) while working from the TRF between memory accesses. This is the 'peak' efficiency for these stated data statistics for this design, when memory access energy is minimized.

Since a custom RRAM digital macro was developed and used for this design, it is important to characterize the resulting BER at points of interest. Figure 4.21 shows an error-rate characterization using freshly written cells at 27C, for a single bank contain-

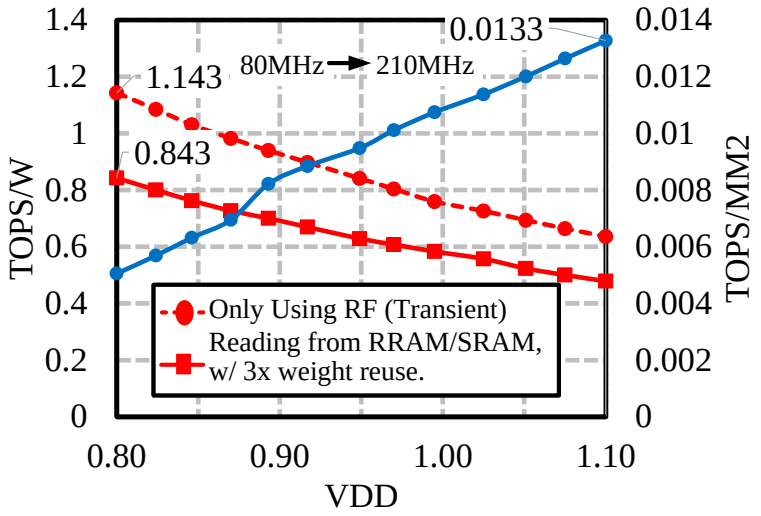


Figure 4.20: Energy efficiency of the implemented design for a continuous matrix multiplication test workload with distinct weight and activation matrices that are normally distributed. The activation matrices are 50% sparse and positive since they are modeled as rectified.

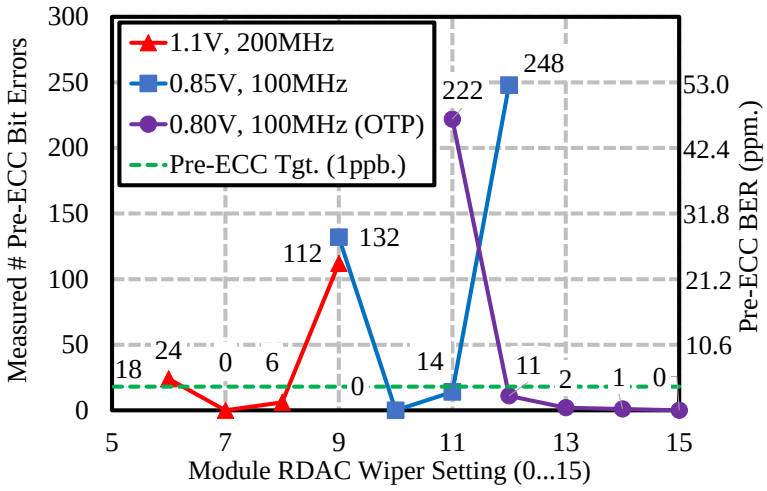


Figure 4.21: RRAM macro error rate characterization for a single bank with 4,718,592 freshly written cells at 27C.

ing 4,718,592 freshly-written cells. The X-axis shows the reference resistor setting, with a higher value indicating a higher reference resistance. In non-OTP modes, a U-shaped curve is observed. This is because too-low of a reference setting causes errors on LRS bits, and too-high of a reference setting causes errors for HRS bits. This curve therefore represents memory window in terms of the reference RDAC LSBs. The voltage scaling is eventually limited, even in OTP mode, by an inability to use a high-enough reference resistance to contrast LRS cells from HRS cells. This likely results from decreased WL voltage causing access device resistance to increase. This access device behavior is likely not well-replicated by the logic transistor used to tie the RDAC wiper to VSS, leading to a greatly increased sensitivity near and below 800mV of net cell resistance to VDD (and thus WL voltage) compared to the reference path. Still, it is possible to measure zero errors out of almost five million cells under these ideal conditions (low cell cycling, freshly programmed, room temperature). This suggests that the macro design works as intended.

4.6 Conclusion

The key contribution of this work has been the modular design that reduces memory access costs without imposing a significant density penalty. The design goals for all-on-chip inference are to maintain memory density to avoid off-chip accesses while reducing NVM access energy and delay to strengthen the advantage over off-chip memory. Compared to prior on-chip inference works [124, 126, 123, 34], this work shows $> 3.64\times$ improved NVM bandwidth per mm^2 and $> 3.79\times$ improved NVM pJ/bit, both at V_{MIN}. At the 40nm node, this work shows 3.61 - 4$\times$ improved NVM density. Both density and efficiency are sensitive to tech. node. Despite the modular design and 3× improved access characteristics, none of the reference works simultaneously demonstrate better chip-level NVM density and compute density. The NMU block does not rely on local buffers and extensive weight reuse to stream weights into the MAC unit at full speed, greatly improving flexibility and aiding density. Finally, due to the large ratio of RRAM to SRAM, this work

Table 4.3: Summary of RRAM all-on-chip digital inference accelerator test-chip.

Technology	40nm ULP with Foundry RRAM
Chip Size	4.5mm × 4.5mm
Package	QFN 64
Voltages	0.8 – 1.1V VDD, 3.3V IO, 1.5 – 4.0V Write
Interface	256b High-Speed SPI (up to 26.25Mbps)
Clock Source	Si5351, Lab, or MCU (Pi Pico) Pin
Modules	10x NVM Matrix Units, 1x Localization
Retentive Sleep	110μW @ 500mV for Full SRAM Retention
On-Chip RRAM	5MB (5.625MB inc. ECC bits)
Total Raw RRAM Bitcell Area	4.645mm^2 (23%)
RRAM ECC	72bit-Word SECDED
On-Chip SRAM	760KB
RRAM Access Energy @ V_{MIN}	256fJ/bit
Efficiency @ V_{MIN}	0.843 TOPS/W
Compute Density @ F_{MAX}	0.0133 TOPS/mm^2

is able to demonstrate a 110μW retentive sleep mode for maintaining state at low standby power. A summary table for this chip is shown in Table 4.3.

CHAPTER 5
CONCLUSION

Efficiently computing with data stored in large memories is challenging. The high cost of off-chip accesses and the emergence of ML inference as an important application for energy-constrained accelerators has only increased the design significance of large on-chip memories. This has led to the exploration of alternative memory technologies including eNVM and eDRAM that offer improved density and/or lower leakage compared to large SRAMs implemented in standard CMOS. This has also led to the exploration of CIM as an alternative approach to traditional memory access patterns.

The first chapter of this dissertation framed the on-chip memory challenge in the context of ML and summarized an important set of emerging memory technologies. A special focus was placed on RRAM since it plays an important role in the research presented in the second and third chapters of this dissertation. CIM was also introduced, starting with a basic example of the straightforward fully-analog variant, before moving on to more feasible variants including the partially-digital current-summing approaches that are implemented in the works presented in the second chapter. Finally, all-on-chip inference with eNVM was introduced as an alternative to CIM, with an accompanying discussion of digital inference accelerators and prior all-on-chip with eNVM works.

The second chapter described analytical work that investigated the feasibility and potential advantages of current-summing CIM with CMOS SRAM arrays. This work looked at the penalties and potential benefits offered by CIM. Topics included the need for striped inputs for multi-bit activations and identifying the most beneficial ways to increase the readout parallelism. The conclusion of this analysis was that CIM may outperform traditional systems for limited cases, especially where accuracy and numerical precision can safely be traded away.

The third chapter described the implementation of two CIM with RRAM macros in 40nm CMOS using the current-summing approach. The first macro followed a predecessor work and introduced design components to reduce area overheads and improve modularity. The tested and working macro reduced mixed-signal active area 16× compared to the active area for the predecessor work. The second macro targeted specific CIM non-idealities, namely channel mismatch, IR drop, and off-state current. An offset-cancelling TIA with approximate four-terminal sensing and a dynamically offset-adjust 6b SAR ADC were described. These specific solutions were measured in the tested macro, showing the effectiveness of the solutions.

The fourth chapter described an end-to-end all-on-chip inference with RRAM implementation. This design was informed by the design process and characteristics of the CIM-based works and used digital MAC computation using large on-chip RRAM. The test chip includes a dense, custom memory macro with RRAM integrated into a modular matrix unit design. This matrix unit was tiled 10 times to form an all-on-chip inference accelerator that demonstrates improved eNVM memory access characteristics while also featuring strong eNVM storage density relative to prior work all-on-chip inference with eNVM works. Like the CIM works, this work was taped-out and measured in silicon. The measured eNVM access characteristics, energy per bit and area-normalized bandwidth, were improved 3× compared to prior work while eNVM density improved 3× when comparing at the same 40nm node. These benefits resulted from the bottom-up design of energy- and area-efficient macros integrated into area-efficient digital modules. By keeping computing resources close the RRAM banks, an improvement in memory access characteristics that strengthens the advantage over going off-chip to access data was demonstrated.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[3] 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.

[4] J. L. Hennessy and D. A. Patterson, *Computer architecture: a quantitative approach*. Elsevier, 2011.

[5] Z. Li *et al.*, "Rram-dnn: An rram and model-compression empowered all-weights-on-chip dnn accelerator," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 4, pp. 1105–1115, 2020.

[6] S. Yu, X. Sun, X. Peng, and S. Huang, "Compute-in-memory with emerging nonvolatile-memories: Challenges and prospects," in *2020 ieee custom integrated circuits conference (cicc)*, IEEE, 2020, pp. 1–4.

[7] S. S. Iyer *et al.*, "Embedded dram: Technology platform for the blue gene/l chip," *IBM Journal of Research and Development*, vol. 49, no. 2.3, pp. 333–350, 2005.

[8] J. Backus, "Can programming be liberated from the von neumann style? a functional style and its algebra of programs," *Communications of the ACM*, vol. 21, no. 8, pp. 613–641, 1978.

[9] J. Von Neumann, "First draft of a report on the edvac," *IEEE Annals of the History of Computing*, vol. 15, no. 4, pp. 27–75, 1993.

[10] A. J. Smith, *Design of CPU cache memories*. Computer Science Division, University of California, 1987.

[11] C. Nugteren, G.-J. Van den Braak, H. Corporaal, and H. Bal, "A detailed gpu cache model based on reuse distance theory," in *2014 IEEE 20th International Symposium on High Performance Computer Architecture (HPCA)*, IEEE, 2014, pp. 37–48.

[12] Y.-H. Chen, J. Emer, and V. Sze, "Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks," *ACM SIGARCH computer architecture news*, vol. 44, no. 3, pp. 367–379, 2016.

[13] H. Park *et al.*, "30-gb/s 1.11-pj/bit single-ended pam-3 transceiver for high-speed memory links," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 2, pp. 581–590, 2020.

[14] S. Przybylski, M. Horowitz, and J. Hennessy, "Performance tradeoffs in cache design," in *[1988] The 15th Annual International Symposium on Computer Architecture. Conference Proceedings*, IEEE, 1988, pp. 290–298.

[15] S. Przybylski, M. Horowitz, and J. Hennessy, "Characteristics of performance-optimal multi-level cache hierarchies," *ACM SIGARCH Computer Architecture News*, vol. 17, no. 3, pp. 114–121, 1989.

[16] C.-L. Su and A. M. Despain, "Cache design trade-offs for power and performance optimization: A case study," in *Proceedings of the 1995 international symposium on Low power design*, 1995, pp. 63–68.

[17] V. Agarwal, M. Hrishikesh, S. W. Keckler, and D. Burger, "Clock rate versus ipc: The end of the road for conventional microarchitectures," in *Proceedings of the 27th annual international symposium on Computer architecture*, 2000, pp. 248–259.

[18] N. S. Kim, K. Flautner, D. Blaauw, and T. Mudge, "Circuit and microarchitectural techniques for reducing cache leakage power," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, no. 2, pp. 167–184, 2004.

[19] P. Gepner, D. L. Fraser, and V. Gamayunov, "Evaluation of the 3rd generation intel core processor focusing on hpc applications," in *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA)*, The Steering Committee of The World Congress in Computer Science, Computer …, 2012, p. 1.

[20] E. Rotem *et al.*, "Intel alder lake cpu architectures," *IEEE Micro*, vol. 42, no. 3, pp. 13–19, 2022.

[21] J. Doweck *et al.*, "Inside 6th-generation intel core: New microarchitecture code-named skylake," *IEEE Micro*, vol. 37, no. 2, pp. 52–62, 2017.

[22] P. Conway, N. Kalyanasundharam, G. Donley, K. Lepak, and B. Hughes, "Cache hierarchy and memory subsystem of the amd opteron processor," *IEEE micro*, vol. 30, no. 2, pp. 16–29, 2010.

[23] T. Singh *et al.*, "3.2 zen: A next-generation high-performance× 86 core," in *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, 2017, pp. 52–53.

[24] D. Suggs, M. Subramony, and D. Bouvier, "The amd "zen 2" processor," *IEEE Micro*, vol. 40, no. 2, pp. 45–52, 2020.

[25] D. Foley and J. Danskin, "Ultra-performance pascal gpu and nvlink interconnect," *IEEE Micro*, vol. 37, no. 2, pp. 7–17, 2017.

[26] J. Choquette, E. Lee, R. Krashinsky, V. Balan, and B. Khailany, "3.2 the a100 datacenter gpu and ampere architecture," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 64, 2021, pp. 48–50.

[27] J. Macri, "Amd's next generation gpu and high bandwidth memory architecture: Fury," in *2015 IEEE Hot Chips 27 Symposium (HCS)*, IEEE, 2015, pp. 1–26.

[28] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th annual international symposium on computer architecture*, 2017, pp. 1–12.

[29] S. Markidis, S. W. Der Chien, E. Laure, I. B. Peng, and J. S. Vetter, "Nvidia tensor core programmability, performance & precision," in *2018 IEEE international parallel and distributed processing symposium workshops (IPDPSW)*, IEEE, 2018, pp. 522–531.

[30] T.-Y. J. Chang *et al.*, "A 5-nm 135-mb sram in euv and high-mobility channel finfet technology with metal coupling and charge-sharing write-assist circuitry schemes for high-density and low-v min applications," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 1, pp. 179–187, 2020.

[31] T. Song *et al.*, "A 3-nm gate-all-around sram featuring an adaptive dual-bitline and an adaptive cell-power assist circuit," *IEEE Journal of Solid-State Circuits*, vol. 57, no. 1, pp. 236–244, 2021.

[32] Y. Kim *et al.*, "Energy-efficient high bandwidth 6t sram design on intel 4 cmos technology," *IEEE Journal of Solid-State Circuits*, vol. 58, no. 4, pp. 1087–1093, 2022.

[33] Y. Aoyagi *et al.*, "A 3-nm 27.6-mbit/mm2 self-timed sram enabling 0.48-1.2 v wide operating range with far-end pre-charge and weak-bit tracking," in *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, IEEE, 2023, pp. 1–2.

[34] J. Chang *et al.*, "A 3nm 256mb sram in finfet technology with new array banking architecture and write-assist circuitry scheme for high-density and low-v min applications," in *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, IEEE, 2023, pp. 1–2.

[35] Y. Osada *et al.*, "3.7-ghz multi-bank high-current single-port cache sram with 0.5 v-1.4 v wide voltage range operation in 3nm finfet for hpc applications," in *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, IEEE, 2023, pp. 1–2.

[36] S.-Y. Wu *et al.*, "A 16nm finfet cmos technology for mobile soc and computing applications," in *2013 IEEE International Electron Devices Meeting*, IEEE, 2013, pp. 9–1.

[37] S.-Y. Wu *et al.*, "A 7nm cmos platform technology featuring 4 th generation finfet transistors with a 0.027 um 2 high density 6-t sram cell for mobile soc applications," in *2016 IEEE International Electron Devices Meeting (IEDM)*, IEEE, 2016, pp. 2–6.

[38] G. Yeap *et al.*, "5nm cmos production technology platform featuring full-fledged euv, and high mobility channel finfets with densest 0.021 μm 2 sram cells for mobile soc and high performance computing applications," in *2019 IEEE International Electron Devices Meeting (IEDM)*, IEEE, 2019, pp. 36–7.

[39] B. Sell *et al.*, "Intel 4 cmos technology featuring advanced finfet transistors optimized for high density and high-performance computing," in *2022 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, IEEE, 2022, pp. 282–283.

[40] S.-Y. Wu *et al.*, "A 3nm cmos finflex™ platform technology with enhanced power efficiency and performance for mobile soc and high performance computing applications," in *2022 International Electron Devices Meeting (IEDM)*, IEEE, 2022, pp. 27–5.

[41] D. U. Lee *et al.*, "25.2 a 1.2 v 8gb 8-channel 128gb/s high-bandwidth memory (hbm) stacked dram with effective microbump i/o test methods using 29nm process and tsv," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, IEEE, 2014, pp. 432–433.

[42] H. Jun *et al.*, "Hbm (high bandwidth memory) dram technology and architecture," in *2017 IEEE International Memory Workshop (IMW)*, IEEE, 2017, pp. 1–4.

[43] R. Venkatesan *et al.*, "Magnet: A modular accelerator generator for neural networks," in *2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, IEEE, 2019, pp. 1–8.

[44] B. Keller *et al.*, "A 95.6-tops/w deep learning inference accelerator with per-vector scaled 4-bit quantization in 5 nm," *IEEE Journal of Solid-State Circuits*, vol. 58, no. 4, pp. 1129–1141, 2023.

[45] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[46] T.-Y. Lin *et al.*, "Microsoft coco: Common objects in context," in *European conference on computer vision*, Springer, 2014, pp. 740–755.

[47] R. B. Girshick, P. F. Felzenszwalb, and D. McAllester, "Discriminatively trained deformable part models, release 5," 2012.

[48] A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[49] Q. Chen *et al.*, "Group detr v2: Strong object detector with encoder-decoder pretraining," *arXiv preprint arXiv:2211.03594*, 2022.

[50] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "Yolox: Exceeding yolo series in 2021," *arXiv preprint arXiv:2107.08430*, 2021.

[51] G. Yu *et al.*, "Pp-picodet: A better real-time object detector on mobile devices," *arXiv preprint arXiv:2111.00902*, 2021.

[52] H. Li, Q. Zhou, Y. Mao, B. Zhang, and C. Liu, "Alpha-sganet: A multi-attention-scale feature pyramid network combined with lightweight network based on alpha-iou loss," *Plos one*, vol. 17, no. 10, e0276581, 2022.

[53] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv preprint arXiv:2207.02696*, 2022.

[54] Z. Liu *et al.*, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 012–10 022.

[55] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, IEEE, 2015, pp. 5206–5210.

[56] W. Han *et al.*, "Contextnet: Improving convolutional neural networks for automatic speech recognition with global context," *arXiv preprint arXiv:2005.03191*, 2020.

[57] A. Gulati *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.

[58] H.-S. P. Wong *et al.*, "Metal–oxide rram," *Proceedings of the IEEE*, vol. 100, no. 6, pp. 1951–1970, 2012.

[59] C.-C. Chou *et al.*, "An n40 256k× 44 embedded rram macro with sl-precharge sa and low-voltage current limiter to improve read and write performance," in *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2018, pp. 478–480.

[60] C.-C. Chou *et al.*, "A 22nm 96kx144 rram macro with a self-tracking reference and a low ripple charge pump to achieve a configurable read window and a wide operating voltage range," in *2020 IEEE Symposium on VLSI Circuits*, IEEE, 2020, pp. 1–2.

[61] O. Golonzka *et al.*, "Non-volatile rram embedded into 22ffl finfet technology," in *2019 Symposium on VLSI Technology*, IEEE, 2019, T230–T231.

[62] P. Jain *et al.*, "13.2 a 3.6 mb 10.1 mb/mm 2 embedded non-volatile reram macro in 22nm finfet technology with adaptive forming/set/reset schemes yielding down to 0.5 v with sensing time of 5ns at 0.7 v," in *2019 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2019, pp. 212–214.

[63] S.-Y. Wu *et al.*, "A highly manufacturable 28nm cmos low power platform technology with fully functional 64mb sram using dual/tripe gate oxide process," in *2009 Symposium on VLSI Technology*, IEEE, 2009, pp. 210–211.

[64] Y.-J. Song *et al.*, "Highly functional and reliable 8mb stt-mram embedded in 28nm logic," in *2016 IEEE International Electron Devices Meeting (IEDM)*, IEEE, 2016, pp. 27–2.

[65] Y.-C. Shih *et al.*, "Logic process compatible 40-nm 16-mb, embedded perpendicular-mram with hybrid-resistance reference, sub-ua sensing resolution, and 17.5-ns read access time," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 4, pp. 1029–1038, 2019.

[66] X. Fong, Y. Kim, R. Venkatesan, S. H. Choday, A. Raghunathan, and K. Roy, "Spin-transfer torque memories: Devices, circuits, and systems," *Proceedings of the IEEE*, vol. 104, no. 7, pp. 1449–1488, 2016.

[67] L. Wei *et al.*, "13.3 a 7mb stt-mram in 22ffl finfet technology with 4ns read sensing time at 0.9 v using write-verify-write scheme and offset-cancellation sensing technique," in *2019 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2019, pp. 214–216.

[68] Y.-D. Chih *et al.*, "13.3 a 22nm 32mb embedded stt-mram with 10ns read speed, 1m cycle write endurance, 10 years retention at 150 c and high immunity to magnetic

field interference," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2020, pp. 222–224.

[69] Y.-C. Shih *et al.*, "A reflow-capable, embedded 8mb stt-mram macro with 9ns read access time in 16nm finfet logic cmos process," in *2020 IEEE International Electron Devices Meeting (IEDM)*, IEEE, 2020, pp. 11–4.

[70] J. Wu *et al.*, "A 40nm low-power logic compatible phase change memory technology," in *2018 IEEE International Electron Devices Meeting (IEDM)*, IEEE, 2018, pp. 27–6.

[71] N. Grossier *et al.*, "Asil-d automotive-grade microcontroller in 28nm fd-soi with full-ota capable 21mb embedded pcm memory and highly scalable power management," in *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, IEEE, 2023, pp. 1–2.

[72] M. Carissimi *et al.*, "An extended temperature range epcm memory in 90-nm bcd for smart power applications," in *ESSCIRC 2022-IEEE 48th European Solid State Circuits Conference (ESSCIRC)*, IEEE, 2022, pp. 373–376.

[73] D. Min *et al.*, "18nm fdsoi technology platform embedding pcm & innovative continuous-active construct enhancing performance for leading-edge mcu applications," in *2021 IEEE International Electron Devices Meeting (IEDM)*, IEEE, 2021, pp. 13–1.

[74] F. Arnaud *et al.*, "High density embedded pcm cell in 28nm fdsoi technology for automotive micro-controller applications," in *2020 IEEE International Electron Devices Meeting (IEDM)*, IEEE, 2020, pp. 24–2.

[75] S. Beyer *et al.*, "Fefet: A versatile cmos compatible device with game-changing potential," in *2020 IEEE International Memory Workshop (IMW)*, IEEE, 2020, pp. 1–4.

[76] M. Trentzsch *et al.*, "A 28nm hkmg super low power embedded nvm technology based on ferroelectric fets," in *2016 IEEE International Electron Devices Meeting (IEDM)*, IEEE, 2016, pp. 11–5.

[77] S. Dünkel *et al.*, "A fefet based super-low-power ultra-fast embedded nvm technology for 22nm fdsoi and beyond," in *2017 IEEE International Electron Devices Meeting (IEDM)*, IEEE, 2017, pp. 19–7.

[78] N. Zagni, F. M. Puglisi, P. Pavan, and M. A. Alam, "Reliability of hfo2-based ferroelectric fets: A critical review of current and future challenges," *Proceedings of the IEEE*, 2023.

[79] A. I. Khan, A. Keshavarzi, and S. Datta, "The future of ferroelectric field-effect transistor technology," *Nature Electronics*, vol. 3, no. 10, pp. 588–597, 2020.

[80] Y. Taito *et al.*, "A 28 nm embedded split-gate monos (sg-monos) flash macro for automotive achieving 6.4 gb/s read throughput by 200 mhz no-wait read operation and 2.0 mb/s write throughput at tj of 170°c," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 1, pp. 213–221, 2015.

[81] C. Y.-S. Cho, J. Wang, Y. Lin, Y. Chih, and S. Natarajan, "A 55-nm, 0.86-volt operation, 125mhz high speed, 75ua/mhz low power, wide voltage supply range 2m-bit split-gate embedded flash," in *2013 5th IEEE International Memory Workshop*, IEEE, 2013, pp. 124–127.

[82] D. Shum *et al.*, "Highly reliable flash memory with self-aligned split-gate cell embedded into high performance 65nm cmos for automotive & smartcard applications," in *2012 4th IEEE International Memory Workshop*, IEEE, 2012, pp. 1–4.

[83] J. Pak *et al.*, "40nm & 22nm embedded charge trap flash for automotive applications," in *2018 IEEE International Memory Workshop (IMW)*, IEEE, 2018, pp. 1–4.

[84] H.-C. Yu, K.-F. Lin, Y.-D. Chih, and J. Chang, "A 40nm split gate embedded flash macro with flexible 2-in-1 architecture, code memory with 140mhz read speed and data memory with 1m cycles endurance," in *2017 Symposium on VLSI Circuits*, IEEE, 2017, pp. C198–C199.

[85] S. Datta, S. Dutta, B. Grisafe, J. Smith, S. Srinivasa, and H. Ye, "Back-end-of-line compatible transistors for monolithic 3-d integration," *IEEE micro*, vol. 39, no. 6, pp. 8–15, 2019.

[86] A. Teman, P. Meinerzhagen, A. Burg, and A. Fish, "Review and classification of gain cell edram implementations," in *2012 IEEE 27th Convention of Electrical and Electronics Engineers in Israel*, IEEE, 2012, pp. 1–5.

[87] R. Giterman, A. Shalom, A. Burg, A. Fish, and A. Teman, "A 1-mbit fully logic-compatible 3t gain-cell embedded dram in 16-nm finfet," *IEEE Solid-State Circuits Letters*, vol. 3, pp. 110–113, 2020.

[88] H. Ye *et al.*, "Double-gate w-doped amorphous indium oxide transistors for monolithic 3d capacitorless gain cell edram," in *2020 IEEE International Electron Devices Meeting (IEDM)*, IEEE, 2020, pp. 28–3.

[89] C. Y. Chen *et al.*, "A high-performance low-power highly manufacturable embedded dram technology using backend hi-k mim capacitor at 40nm node and beyond,"

in *Proceedings of 2011 International Symposium on VLSI Technology, Systems and Applications*, IEEE, 2011, pp. 1–2.

[90] K. Huang *et al.*, "A high-performance, high-density 28nm edram technology with high-k/metal-gate," in *2011 International Electron Devices Meeting*, IEEE, 2011, pp. 24–7.

[91] F. Hamzaoglu *et al.*, "13.1 a 1gb 2ghz embedded dram in 22nm tri-gate cmos technology," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, IEEE, 2014, pp. 230–231.

[92] A. Shafiee *et al.*, "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.

[93] M. J. Pelgrom, H. P. Tuinhout, and M. Vertregt, "Transistor matching in analog cmos applications," in *International electron devices meeting 1998. technical digest (Cat. No. 98CH36217)*, IEEE, 1998, pp. 915–918.

[94] Z. Chen *et al.*, "Cap-ram: A charge-domain in-memory computing 6t-sram for accurate and precision-programmable cnn inference," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 6, pp. 1924–1935, 2021.

[95] H. Jia *et al.*, "Scalable and programmable neural network inference accelerator based on in-memory computing," *IEEE Journal of Solid-State Circuits*, vol. 57, no. 1, pp. 198–211, 2021.

[96] S. Xie, C. Ni, A. Sayal, P. Jain, F. Hamzaoglu, and J. P. Kulkarni, "16.2 edram-cim: Compute-in-memory design with reconfigurable embedded-dynamic-memory array realizing adaptive data converters and charge-domain computing," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 64, 2021, pp. 248–250.

[97] H. Wang, R. Liu, R. Dorrance, D. Dasalukunte, D. Lake, and B. Carlton, "A charge domain sram compute-in-memory macro with c-2c ladder-based 8-bit mac unit in 22-nm finfet process for edge inference," *IEEE Journal of Solid-State Circuits*, vol. 58, no. 4, pp. 1037–1050, 2023.

[98] P. Chen *et al.*, "7.8 a 22nm delta-sigma computing-in-memory (Δ cim) sram macro with near-zero-mean outputs and lsb-first adcs achieving 21.38 tops/w for 8b-mac edge ai processing," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, 2023, pp. 140–142.

[99] H. Omran, H. Alahmadi, and K. N. Salama, "Matching properties of femtofarad and sub-femtofarad mom capacitors," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 63, no. 6, pp. 763–772, 2016.

[100] H. Fujiwara *et al.*, "A 5-nm 254-tops/w 221-tops/mm 2 fully-digital computing-in-memory macro supporting wide-range dynamic-voltage-frequency scaling and simultaneous mac and write operations," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 65, 2022, pp. 1–3.

[101] Y.-D. Chih *et al.*, "16.4 an 89tops/w and 16.3 tops/mm 2 all-digital sram-based full-precision compute-in memory macro in 22nm for machine-learning edge applications," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 64, 2021, pp. 252–254.

[102] J. Oh, C.-T. Lin, and M. Seok, "D6cim: 60.4-tops/w, 1.46-tops/mm 2, 1005-kb/mm 2 digital 6t-sram-based compute-in-memory macro supporting 1-to-8b fixed-point arithmetic in 28-nm cmos," in *ESSCIRC 2023-IEEE 49th European Solid State Circuits Conference (ESSCIRC)*, IEEE, 2023, pp. 413–416.

[103] H. Kim, Q. Chen, T. Yoo, T. T.-H. Kim, and B. Kim, "A 1-16b precision reconfigurable digital in-memory computing macro featuring column-mac architecture and bit-serial computation," in *ESSCIRC 2019-IEEE 45th European Solid State Circuits Conference (ESSCIRC)*, IEEE, 2019, pp. 345–348.

[104] M. Gupta, S. Cosemans, P. Debacker, and W. Dehaene, "A 2mbit digital in-memory computing matrix-vector multiplier for dnn inference supporting flexible bit precision and matrix size achieving 612 binary tops/w," in *ESSCIRC 2023-IEEE 49th European Solid State Circuits Conference (ESSCIRC)*, IEEE, 2023, pp. 417–420.

[105] H. Mori *et al.*, "A 4nm 6163-tops/w/b 4790-tops/mm2 sram based digital-computing-in-memory macro supporting bit-width flexibility and simultaneous mac and weight update," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, 2023, pp. 132–134.

[106] S. Spetalnick and A. Raychowdhury, "A practical design-space analysis of compute-in-memory with sram," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 4, pp. 1466–1479, 2022.

[107] C.-X. Xue *et al.*, "15.4 a 22nm 2mb reram compute-in-memory macro with 121-28tops/w for multibit mac computing for tiny ai edge devices," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2020, pp. 244–246.

[108] S. D. Spetalnick *et al.*, "A 40nm 64kb 26.56 tops/w 2.37 mb/mm 2 rram binary/compute-in-memory macro with 4.23 x improvement in density and¿ 75% use of sensing dy-

namic range," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 65, 2022, pp. 1–3.

[109] S. D. Spetalnick *et al.*, "A 2.38 mcells/mm 2 9.81-350 tops/w rram compute-in-memory macro in 40nm cmos with hybrid offset/i off cancellation and i cell r blsl drop mitigation," in *2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, IEEE, 2023, pp. 1–2.

[110] W.-H. Chen *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.

[111] C.-X. Xue *et al.*, "24.1 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.

[112] Q. Liu *et al.*, "33.2 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.

[113] J.-H. Yoon, M. Chang, W.-S. Khwa, Y.-D. Chih, M.-F. Chang, and A. Raychowdhury, "29.1 a 40nm 64kb 56.67 tops/w read-disturb-tolerant compute-in-memory/digital rram macro with active-feedback-based read and in-situ write verification," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 64, 2021, pp. 404–406.

[114] C.-X. Xue *et al.*, "16.1 a 22nm 4mb 8b-precision reram computing-in-memory macro with 11.91 to 195.7 tops/w for tiny ai edge devices," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 64, 2021, pp. 245–247.

[115] J.-M. Hung *et al.*, "8-b precision 8-mb reram compute-in-memory macro using direct-current-free time-domain readout scheme for ai edge devices," *IEEE Journal of Solid-State Circuits*, vol. 58, no. 1, pp. 303–315, 2022.

[116] J. M. Correll *et al.*, "An 8-bit 20.7 tops/w multi-level cell reram-based compute engine," in *2022 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, IEEE, 2022, pp. 264–265.

[117] W.-H. Huang *et al.*, "A nonvolatile al-edge processor with 4mb slc-mlc hybrid-mode reram compute-in-memory macro and 51.4-251tops/w," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, 2023, pp. 15–17.

[118] B. Zimmer *et al.*, "A 0.32–128 tops, scalable multi-chip-module-based deep neural network inference accelerator with ground-referenced signaling in 16 nm," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 4, pp. 920–932, 2020.

[119] J.-S. Park *et al.*, "A multi-mode 8k-mac hw-utilization-aware neural processing unit with a unified multi-precision datapath in 4-nm flagship mobile soc," *IEEE Journal of Solid-State Circuits*, vol. 58, no. 1, pp. 189–202, 2022.

[120] J. W. Poulton *et al.*, "A 1.17-pj/b, 25-gb/s/pin ground-referenced single-ended serial link for off-and on-package communication using a process-and temperature-adaptive voltage regulator," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 1, pp. 43–54, 2018.

[121] G. Gangasani *et al.*, "A 1.6 tb/s chiplet over xsr-mcm channels using 113gb/s pam-4 transceiver with dynamic receiver-driven adaptation of tx-ffe and programmable roaming taps in 5nm cmos," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 65, 2022, pp. 122–124.

[122] Y. Wei *et al.*, "9.3 nvlink-c2c: A coherent off package chip-to-chip interconnect with 40gbps/pin single-ended signaling," in *2023 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, 2023, pp. 160–162.

[123] Q. Zhang *et al.*, "A 22nm 3.5 tops/w flexible micro-robotic vision soc with 2mb emram for fully-on-chip intelligence," in *2022 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits)*, IEEE, 2022, pp. 72–73.

[124] D. Rossi *et al.*, "4.4 a 1.3 tops/w@ 32gops fully integrated 10-core soc for iot end-nodes with 1.7 μw cognitive wake-up from mram-based state-retentive sleep mode," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 64, 2021, pp. 60–62.

[125] Z. Wang *et al.*, "An all-weights-on-chip dnn accelerator in 22nm ull featuring 24$\times$ 1 mb erram," in *2020 IEEE Symposium on VLSI Circuits*, IEEE, 2020, pp. 1–2.

[126] K. Prabhu *et al.*, "Chimera: A 0.92-tops, 2.2-tops/w edge ai accelerator with 2-mbyte on-chip foundry resistive ram for efficient training and inference," *IEEE Journal of Solid-State Circuits*, vol. 57, no. 4, pp. 1013–1026, 2022.

[127] 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 *2020 IEEE Custom Integrated Circuits Conference (CICC)*, IEEE, 2020, pp. 1–4.

[128] 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 *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2018, pp. 490–492.

[129] J. Zhang, Z. Wang, and N. Verma, "In-memory computation of a machine-learning classifier in a standard 6t sram array," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 4, pp. 915–924, 2017.

[130] W.-S. Khwa *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 *2018 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2018, pp. 496–498.

[131] A. Biswas and A. P. Chandrakasan, "Conv-sram: An energy-efficient sram with in-memory dot-product computation for low-power convolutional neural networks," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 1, pp. 217–230, 2018.

[132] H. Valavi, P. J. Ramadge, E. Nestler, and N. Verma, "A 64-tile 2.4-mb in-memory-computing cnn accelerator employing charge-domain compute," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 6, pp. 1789–1799, 2019.

[133] 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.

[134] X. Si *et al.*, "15.5 a 28nm 64kb 6t sram computing-in-memory macro with 8b mac operation for ai edge chips," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2020, pp. 246–248.

[135] J. Yue *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 *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 64, 2021, pp. 238–240.

[136] J.-W. Su *et al.*, "15.2 a 28nm 64kb inference-training two-way transpose multibit 6t sram compute-in-memory macro for ai edge chips," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2020, pp. 240–242.

[137] J.-W. Su *et al.*, "16.3 a 28nm 384kb 6t-sram computation-in-memory macro with 8b precision for ai edge chips," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 64, 2021, pp. 250–252.

[138] J. Yang *et al.*, "24.4 sandwich-ram: An energy-efficient in-memory bwn architecture with pulse-width modulation," in *2019 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2019, pp. 394–396.

[139] S. Tsukamoto, W. G. Schofield, and T. Endo, "A cmos 6-b, 400-msample/s adc with error correction," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 12, pp. 1939–1947, 1998.

[140] L. Dadda, "Some schemes for parallel multipliers," *Alta Frequenza*, vol. 45, pp. 345–356, 1965.

[141] K. Bickerstaff, M. Schulte, and E. Swartzlander, "Reduced area multipliers," in *Proceedings of International Conference on Application Specific Array Processors (ASAP'93)*, IEEE, 1993, pp. 478–489.

[142] B. Wicht, T. Nirschl, and D. Schmitt-Landsiedel, "A yield-optimized latch-type sram sense amplifier," in *ESSCIRC 2004-29th European Solid-State Circuits Conference (IEEE Cat. No. 03EX705)*, IEEE, 2003, pp. 409–412.

[143] Q. Dong *et al.*, "15.3 a 351tops/w and 372.4 gops compute-in-memory sram macro in 7nm finfet cmos for machine-learning applications," in *2020 IEEE International Solid-State Circuits Conference-(ISSCC)*, IEEE, 2020, pp. 242–244.

[144] A. S. Rekhi *et al.*, "Analog/mixed-signal hardware error modeling for deep learning inference," in *Proceedings of the 56th Annual Design Automation Conference 2019*, 2019, pp. 1–6.

[145] 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.

[146] M. Chang *et al.*, "A 40nm 60.64 tops/w ecc-capable compute-in-memory/digital 2.25 mb/768kb rram/sram system with embedded cortex m3 microprocessor for edge recommendation systems," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 65, 2022, pp. 1–3.

[147] A. S. Lele *et al.*, "A heterogeneous rram in-memory and sram near-memory soc for fused frame and event-based target identification and tracking," *IEEE Journal of Solid-State Circuits*, 2023.

[148] B. Crafton *et al.*, "Improving compute in-memory ecc reliability with successive correction," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, 2022, pp. 745–750.

[149] A. Fritsch *et al.*, "24.1 a 6.2 ghz single ended current sense amplifier (csa) based compileable 8t sram in 7nm finfet technology," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, IEEE, vol. 64, 2021, pp. 334–336.

[150] R. J. Baker, *CMOS: circuit design, layout, and simulation*. John Wiley & Sons, 2019.

[151] S. Babayan-Mashhadi and R. Lotfi, "Analysis and design of a low-voltage low-power double-tail comparator," *IEEE transactions on very large scale integration (vlsi) systems*, vol. 22, no. 2, pp. 343–352, 2013.

[152] B. Razavi, *Design of Analog CMOS Integrated Circuits, Second Edition*. McGraw Hill, New York, 2017.

[153] B. Razavi, "The strongarm latch [a circuit for all seasons]," *IEEE Solid-State Circuits Magazine*, vol. 7, no. 2, pp. 12–17, 2015.

[154] A. Waterman, Y. Lee, D. A. Patterson, and K. Asanovic, "The risc-v instruction set manual, volume i: User-level isa, version 2.0," *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2014-54*, 2014.

VITA

Samuel Spetalnick received a Bachelor's degree in Electrical Engineering and Computer Engineering, with Honors, from the Johns Hopkins University in Baltimore, Maryland in 2018. As an undergraduate, he received departmental awards: the William H. Huggins Award for a Junior in Electrical Engineering, the John Boswell Whitehead Award in Electrical and Computer Engineering, and the IEEE Eta Kappa Nu Excellence Award. He was a grateful recipient of the JHU Bloomberg Scholarship and was on the Dean's list for every enrolled semester. While pursuing the PhD degree at the Georgia Institute of Technology, he has been a grateful recipient of the President's Fellowship. His research interests include memory circuit and system design for low power and emerging applications.

List of Publications

- Spetalnick, Lele, Crafton, Chang, Ryu, Yoon, Hao, Ansari, Khwa, Chih, Chang, Raychowdhury. A 40nm VLIW Edge Accelerator with 5MB of 0.256 pJ/bit RRAM and a Localization Solver for Bristle Robot Surveillance. *To be presented:* ISSCC 2024.

- Lele, Chang, Spetalnick, Crafton, Konno, Wan, Bhat, Khwa, Chih, Chang, Raychowdhury. A Heterogeneous RRAM In-Memory and SRAM Near-Memory SoC for Fused Frame and Event-Based Target Identification and Tracking. JSSC 2023.

- Lele, Chang, Spetalnick, Crafton, Raychowhdury, Fang. Neuromorphic Swarm on RRAM Compute-in-Memory Processor for Solving QUBO Problem. DAC 2023.

- Spetalnick, Chang, Konno, Crafton, Lele, Khwa, Chih, Chang, Raychowdhury. A 2.38 MCells/mm2 9.81 - 350 TOPS/W RRAM Compute-in-Memory Macro in 40nm CMOS with Hybrid Offset/IOFF Cancellation and IR Drop Mitigation. VLSI 2023.

- Chang, Lele, Spetalnick, Crafton, Konno, Wan, Bhat, Khwa, Chih, Chang, Raychowdhury. A 73.53 TOPS/W 14.74 TOPS Heterogeneous RRAM In-Memory and SRAM Near-Memory SoC for Hybrid Frame and Event-Based Target Tracking. ISSCC 2023.

- Spetalnick, Raychowdhury. A Practical Design-Space Analysis of Compute-in-Memory With SRAM. TCAS 1, 2022.

- Shrestha, Campbell, Chakraborty, Gupta, Saligram, Spetalnick, Raychowdhury, Datta. Multi-bit per-cell 1T SiGe Floating Body RAM for Cache Memory in Cryogenic Computing. VLSI 2022.

- Crafton, Talley, Spetalnick, Yoon, Raychowdhury. Characterization and Mitigation of IR-Drop in RRAM-based Compute In-Memory. ISCAS 2022.

- Spetalnick, Chang, Crafton, Khwa, Chih, Chang, Raychowdhury. A 40nm 64kb 26.56TOPS/W 2.37Mb/mm2RRAM Binary/Compute-in-Memory Macro with 4.23x Improvement in Density and >75% Use of Sensing Dynamic Range. ISSCC 2022.

- Chang, Spetalnick, Crafton, Khwa, Chih, Chang, Raychowdhury. A 40nm 60.64TOPS/W ECC-Capable Compute-in-Memory/Digital 2.25MB/768KB RRAM/SRAM System with Embedded Cortex M3 Microprocessor for Edge Recommendation Systems. ISSCC 2022,

- Crafton, Spetalnick , Yoon, Wu, Tokunaga, De, Raychowdhury . CIM SECDED: A 40nm 64Kb Compute In Memory RRAM Macro with ECC. ASSCC 2021.

- Crafton, Spetalnick , Yoon, Raychowdhury. Statistical optimization of compute in memory performance under device variation. ISLPED 2021.

- Crafton, Spetalnick , Raychowdhury. Merged Logic and Memory Fabrics for AI Workloads. ASPDAC 2021.

- Crafton, Spetalnick , Fang, Raychowdhury. Merged logic and memory fabrics for accelerating machine learning workloads. Design & Test 2020.
- Ye, Gomez, Chakraborty, Spetalnick, Dutta, Ni, Raychowdhury, Datta. Double-gate W-doped amorphous indium oxide transistors for monolithic 3D capacitorless gain cell eDRAM. IEDM 2020.
- Crafton, Spetalnick , Murali, Krishna, Lim, Raychowdhury. Breaking barriers: Maximizing array utilization for compute in memory fabrics. VLSI SoC 2020.