A PhD THESIS ON

Dependable and Energy Efficient Design of Embedded Systems: A Cross-Layer Approach

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

Doctor of Technical Science

in

Engineering Sciences: Electrical Engineering (UE 786 710)

by

Saeed Seyedfaraji

11944048

Supervisor:

Associate Prof. Dr. -Ing. Privatdoz.in Semeen Rehman

Vienna, Austria

March 2025

Abstract

Embedded systems are at the heart of modern technology, powering diverse applications ranging from autonomous vehicles to medical devices and Internet of Things (IoT) solutions. These systems face unique constraints, such as limited energy budgets, real-time performance requirements, and stringent reliability demands. However, as processing capabilities continue to grow, the separation of memory and processing in traditional von Neumann architectures presents a significant challenge for embedded systems: the memory wall, which causes delays in data transfer between the processor and memory. This issue, characterized by the disparity between processor speeds and memory access latencies, exacerbates inefficiencies in embedded systems, where both energy and computational resources are scarce.

As Richard Sites observed in 1996: *"Today's chips are largely able to execute code faster than we can feed them with instructions and data. The real design action is in memory subsystems—caches, buses, bandwidth, and latency."* This statement highlights the persistent challenge in modern computing, particularly for embedded systems where the combination of constrained resources and real-time performance demands amplifies the inefficiencies introduced by the memory wall.

To address these challenges, novel approaches have redefined data processing within memory systems. Among these, Processing in Memory (PIM) architectures have emerged as a transformative solution, integrating computation directly within memory to minimize data movement and the associated energy per operation consumption. By reducing the reliance on traditional memory subsystems, PIM alleviates delays caused by the gap between processing speed and memory access times for data loads while enhancing energy efficiency, which is an essential requirement in embedded systems.

Despite its promise, existing PIM-based techniques face critical limitations in real-world deployments. For example, Spin Transfer Torque Random Access Memory (STT-RAM), a leading non-volatile memory technology for PIM, suffers from high write energy and latency due to stochastic switching behavior and process variations. Similarly, Static Random Access Memory (SRAM), widely employed in embedded accelerators, experiences inaccuracies in in-memory computations caused by nonlinearities in the Bit-Line (BL) discharge process. These challenges undermine the dependability and energy efficiency of PIM solutions, preventing them from realizing their full potential in memory-centric ap-

plications such as artificial intelligence, while limiting their ability to meet the real-time control and high-performance computing demands, which are crucial for embedded systems where speed and reliability are essential.

In order to address the above-mentioned challenges, this thesis proposes a cross-layer approach to achieving dependable and energy-efficient design for embedded systems, leveraging both volatile and non-volatile memory technologies. By addressing the unique constraints and performance demands of embedded systems, this thesis presents various innovative PIM-based techniques and system-level tools to overcome these limitations and advance the field of embedded computing. In particular, PIM-based systems face robustness challenges, and the lack of comprehensive modeling and benchmarking frameworks complicates the effective development and utilization of these PIM-based systems.

To address these research gaps, this thesis makes the following contributions, a) Developing advanced modeling and benchmarking frameworks for PIM-based systems; b) Designing robust and efficient PIM circuit architectures for improved reliability and performance. Through these key contributions, this thesis advances the field by offering both system-level tools for evaluating STT-RAM architectures and practical solutions for enhancing the efficiency, accuracy, and reliability of SRAM-based memory accelerators. These innovations collectively address long-standing challenges in memory-centric computing systems. To systematically address these challenges, the contributions of this thesis include the aspects of Volatile Memorys (VMs) and Non-Volatile Memorys (NVMs):

1. Volatile Memory

- **OPTIMA**: A modeling framework for rapid design-space exploration of SRAM-based accelerators, addressing circuit nonlinearities and power variations critical for embedded system design [1].
- **AID**: A circuit design technique that linearizes BL discharge in SRAM, significantly improving accuracy and reducing energy consumption in in-memory multiplication accelerators [2].
- **EMAC**: A method leveraging digital-to-time Word-Line (WL) modulation and logical weight encoding to enhance energy efficiency and accuracy in analog SRAM-based Multiplication and Accumulation (MAC) accelerators with minimal accuracy degradation in real-world embedded applications [3].

2. Non-Volatile Memory

- **An open-source, extendable STT-RAM memory controller** integrated into the gem5 simulator, enabling evaluations of power, latency, and throughput to guide optimization strategies [4].

- **A novel write optimization technique** combining stochastic switching and circuit-level approximations to reduce write energy and latency while enhancing robustness against soft errors and process variations [5].

Kurzfassung

Eingebettete Systeme stehen im Zentrum der modernen Technologie und treiben eine Vielzahl von Anwendungen an, die von autonomen Fahrzeugen über medizinische Geräte bis hin zu Internet of Things (IoT)-Lösungen reichen. Diese Systeme unterliegen einzigartigen Einschränkungen, wie begrenzten Energiebudgets, Echtzeitanforderungen und strengen Zuverlässigkeitsanforderungen. Mit zunehmender Verarbeitungskapazität stellt jedoch die Trennung von Speicher und Verarbeitung in traditionellen Von-Neumann-Architekturen eine erhebliche Herausforderung für eingebettete Systeme dar: die memory wall, die Verzögerungen beim Datentransfer zwischen Prozessor und Speicher verursacht. Dieses Problem, das durch die Diskrepanz zwischen Prozessorgeschwindigkeit und Speicherzugriffszeiten gekennzeichnet ist, verstärkt Ineffizienzen in eingebetteten Systemen, in denen sowohl Energie- als auch Rechenressourcen knapp sind.

Wie Richard Sites bereits 1996 feststellte: *"Today's chips are largely able to execute code faster than we can feed them with instructions and data. The real design action is in memory subsystems—caches, buses, bandwidth, and latency."* Diese Aussage unterstreicht die anhaltende Herausforderung in der modernen Informatik, insbesondere für eingebettete Systeme, bei denen die Kombination aus begrenzten Ressourcen und Echtzeitanforderungen die durch die memory wall verursachten Ineffizienzen verstärkt.

Um diese Herausforderungen zu bewältigen, wurden neue Ansätze zur Datenverarbeitung innerhalb von Speichersystemen entwickelt. Unter diesen haben sich PIM-Architekturen als eine transformative Lösung herausgestellt, die Rechenoperationen direkt in den Speicher integrieren, um die Datenbewegung und den damit verbundenen Energieverbrauch pro Operation zu minimieren. Durch die Reduzierung der Abhängigkeit von herkömmlichen Speichersubsystemen verringert PIM Verzögerungen, die durch die Lücke zwischen Prozessorgeschwindigkeit und Speicherzugriffszeiten bei Datenladungen entstehen, während es gleichzeitig die Energieeffizienz verbessert – eine wesentliche Anforderung in eingebetteten Systemen.

Trotz ihres Potenzials stoßen bestehende PIM-basierte Techniken in realen Anwendungen auf kritische Einschränkungen. Beispielsweise leidet STT-RAM, eine führende nichtflüchtige Speichertech-

nologie für PIM, unter hohem Schreibenergieverbrauch und hoher Latenz aufgrund stochastischer Schaltvorgänge und Prozessvariationen. Ebenso weist SRAM, das häufig in eingebetteten Beschleunigern verwendet wird, Ungenauigkeiten bei In-Memory-Berechnungen auf, die durch Nichtlinearitäten im BL-Entladeprozess verursacht werden. Diese Herausforderungen beeinträchtigen die Zuverlässigkeit und Energieeffizienz von PIM-Lösungen, wodurch sie ihr volles Potenzial in speicherzentrierten Anwendungen wie künstlicher Intelligenz nicht ausschöpfen können. Gleichzeitig schränken sie ihre Fähigkeit ein, den Echtzeitsteuerungs- und Hochleistungsrechenanforderungen gerecht zu werden, die für eingebettete Systeme essenziell sind.

Um diese Herausforderungen zu bewältigen, schlägt diese Dissertation einen schichtenübergreifenden Ansatz zur Entwicklung zuverlässiger und energieeffizienter eingebetteter Systeme vor, der sowohl flüchtige als auch nichtflüchtige Speichertechnologien nutzt. Durch die Berücksichtigung der spezifischen Einschränkungen und Leistungsanforderungen eingebetteter Systeme werden in dieser Arbeit verschiedene innovative PIM-basierte Techniken und systemorientierte Werkzeuge vorgestellt, um diese Einschränkungen zu überwinden und das Gebiet des eingebetteten Rechnens voranzutreiben. Insbesondere stehen PIM-basierte Systeme vor Herausforderungen hinsichtlich der Robustheit, und das Fehlen umfassender Modellierungs- und Benchmarking-Frameworks erschwert die effektive Entwicklung und Nutzung dieser PIM-basierten Systeme.

Um diese Forschungslücken zu schließen, trägt diese Dissertation zu folgenden Aspekten bei: a) Entwicklung fortschrittlicher Modellierungs- und Benchmarking-Frameworks für PIM-basierte Systeme; b) Entwurf robuster und effizienter PIM-Schaltungsarchitekturen zur Verbesserung der Zuverlässigkeit und Leistung.

Durch diese wesentlichen Beiträge treibt diese Arbeit das Fachgebiet voran, indem sowohl systemorientierte Werkzeuge zur Evaluierung von STT-RAM-Architekturen als auch praktische Lösungen zur Steigerung der Effizienz, Genauigkeit und Zuverlässigkeit von SRAM-basierten Speicherbeschleunigern bereitgestellt werden. Diese Innovationen adressieren gemeinsam langjährige Herausforderungen in speicherzentrierten Rechensystemen.

Um diese Herausforderungen systematisch zu adressieren, umfassen die Beiträge dieser Dissertation die Aspekte von VMs und NVMs:

1. Volatile Memory

- **OPTIMA**: Ein Modellierungsframework für die schnelle Entwurfsraumanalyse von SRAM-basierten Beschleunigern, das Schaltungsnichtlinearitäten und Leistungsvariationen berücksichtigt, die für das Design eingebetteter Systeme entscheidend sind [1].
- **AID**: Eine Schaltungstechnik zur Linearisierung der BL-Entladung in SRAM, die die Genauigkeit

erheblich verbessert und den Energieverbrauch in In-Memory-Multiplikationsbeschleunigern reduziert [2].

- **EMAC**: Eine Methode, die digitale-zu-Zeit-WL-Modulation und logisches Gewichtscodieren nutzt, um die Energieeffizienz und Genauigkeit in analogen SRAM-basierten MAC-Beschleunigern zu verbessern, mit minimalen Genauigkeitsverlusten in realen eingebetteten Anwendungen [3].

2. Non-Volatile Memory

- **Ein Open-Source, erweiterbarer STT-RAM-Speichercontroller**, der in den gem5-Simulator integriert ist und die Bewertung von Leistung, Latenz und Durchsatz ermöglicht, um Optimierungsstrategien zu entwickeln [4].
- **Eine neuartige Schreiboptimierungstechnik**, die stochastisches Schalten und schaltungstechnische Approximationen kombiniert, um den Schreibenergieverbrauch und die Latenz zu reduzieren, während gleichzeitig die Robustheit gegenüber Soft-Fehlern und Prozessvariationen verbessert wird [5].

Erklärung

Hiermit erkläre ich, dass die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt wurde. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder in ähnlicher Form in anderen Prüfungsverfahren vorgelegt.

Copyright Statement

I, Saeed Seyedfaraji, hereby declare that this thesis is my own original work and, to the best of my knowledge and belief, it does not:

- Breach copyright or other intellectual property rights of a third party.
- Contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
- Contain material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning.
- Contain substantial portions of third party copyright material, including but not limited to charts, diagrams, graphs, photographs or maps, or in instances where it does, I have obtained permission to use such material and allow it to be made accessible worldwide via the Internet.

Signature: ___________________________

Vienna, Austria, March 2025 Saeed Seyedfaraji

Acknowledgment

I am deeply grateful to my supervisor, Associate Prof. Dr.-Ing. Privatdoz. Semeen Rehman, whose unwavering support and guidance have been instrumental in shaping this Ph.D. thesis. Her mentorship has been invaluable, and without her encouragement, this work would not have been possible.

I extend my sincere thanks to Univ.Prof. Dipl.-Ing. Dr. techn. Axel Jantsch for his years of support and leadership as the head of the Institute of Computer Technology. His guidance has played a crucial role in my academic journey. I am also profoundly appreciative of Prof. Kuan-Hsun Chen and Univ.Prof. Dipl.-Ing. Dr. techn. Axel Jantsch for their insightful feedback and contributions as co-examiners.

My heartfelt gratitude goes to my colleagues at the Institute of Computer Technology, Microwave, and Circuit Engineering. Their camaraderie and collaboration have enriched my research experience. I am especially grateful to Mr. Baset Mesgari, whose guidance, expertise, and unwavering support have been instrumental in shaping my research. His collaboration has profoundly influenced my work, and I deeply appreciate his invaluable insights. I also extend my gratitude to Mr. Asad Aftab, whose support and discussions have contributed significantly to my research journey.

Finally, I reserve my deepest appreciation for my wife, Farzaneh Namdarpour. Her unwavering support, patience, and encouragement have been my pillar of strength throughout this journey, making every challenge more bearable and every success more meaningful.

Dedication

To my father and mother, whose presence in my life shines as brightly as stars in the night sky, illuminating my path with their wisdom, love, and unwavering support throughout every journey and endeavor.

To my beloved wife, who embodies the very essence and purpose of life itself. Her presence brings meaning, joy, and fulfillment to every moment, and her love is the guiding light that brightens my days.

Lastly, to the courageous and resilient men and women of my homeland, Iran, who tirelessly strive for freedom and equality. Their determination, perseverance, and sacrifices inspire hope and serve as a reminder of the enduring human spirit in the pursuit of justice and liberty for all.

List of publication

[1]. Seyedfaraji, S., Shakibhamedan, S., Seyedfaraji, A., Mesgari, B., TaheriNejad, N., Jantsch, A., & Rehman, S. (2024). E-MAC: Enhanced In-SRAM MAC Accuracy via Digital-to-Time Modulation. IEEE Journal on Exploratory Solid-State Computational Devices and Circuits (JXCDC) [3].

[2]. Seyedfaraji, S., Jager, S., Shakibhamedan, S., Aftab, A., & Rehman, S. (2024). OPTIMA: Design-Space Exploration of Discharge-Based In-SRAM Computing: Quantifying Energy-Accuracy Trade-offs. In Proceedings of the 61st ACM/IEEE Design Automation Conference (DAC) [1].

[3]. Seyedfaraji, S., Bichl, M., Aftab, A., & Rehman, S. (2024). HOPE: Holistic STT-RAM Architecture Exploration Framework for Future Cross-Platform Analysis. IEEE Access [4].

[4]. Seyedfaraji, S., Mesgari, B., & Rehman, S. (2022). AID: Accuracy improvement of analog discharge-based in-SRAM multiplication accelerator. In 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE) [2].

[5]. Seyedfaraji, S., Mesgari, B., & Rehman, S. (2022). SMART: Investigating the Impact of Threshold Voltage Suppression in an In-SRAM Multiplication/Accumulation Accelerator for Accuracy Improvement in 65 nm CMOS Technology. In 2022 25th Euromicro Conference on Digital System Design (DSD) [6].

[6]. Seyedfaraji, S., Daryani, J. T., Aly, M. M. S., & Rehman, S. (2022). EXTENT: Enabling approximation-oriented energy efficient STT-RAM write circuit. IEEE Access [5].

Contents

List of Tables

List of Figures

Acronyms

ADC	Analog-to-Digital Converter. xxii, 22, 34, 40, 48, 51, 56, 57, 58, 63, 70
AxC	Approximate Computing. 4
BER	Bit Error Rate. 33, 34, 37, 38, 44
BL	Bit-Line. iii, iv, vii, 5, 6, 15, 33, 34, 39, 47, 49, 50, 51, 57, 58, 60, 63, 66, 111
BLB	Bit-Line-Bar. xxi, xxii, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 47, 48, 49, 50, 51, 57, 58, 59, 63, 65, 67, 70
CIM	Computing In Memory. 21, 27
CMOS	Complementary Metal-Oxide Semiconductor. xvii, 3, 12, 14, 20, 21, 25, 32, 40, 41, 42, 44, 47, 58, 65, 66, 67, 78, 107, 109
CNN	Convolutional Neural Network. xviii, xxii, 21, 57, 68, 69, 70, 71, 72, 74, 112
CPU	Central Processing Unit. 15, 23, 26, 29
CVS	Clustered Voltage Scaling. 101
DAC	Digital-to-Analog Converter. 37, 38, 39, 44, 48, 51, 52, 56, 57, 112
DNN	Deep Neural Network. xix, 21, 22, 26, 28, 47, 54, 55, 56, 68, 69, 70
DPU	DRAM Processing Unit. 29
DRAM	Dynamic Random Access Memory. xviii, xix, xxi, xxiii, 2, 11, 12, 14, 15, 18, 20, 22, 23, 24, 25, 26, 27, 29, 77, 78, 79, 80, 83, 85, 86, 87, 88, 89, 90, 93, 94, 95, 96, 98
DVFS	Dynamic Voltage and Frequency Scaling. 101

DVS	Differential Voltage Sampling. 63, 65
DVS	Dynamic Voltage Scaling. 101
FeFET	Ferroelectric Field Effect Transistor. 21, 22
FeRAM	Ferroelectric Random Access Memory. xxi, 19, 20
FL	Fixed Layer. 80, 98
FOM	Figure of Merit. 52
FPGA	Field Programmable Gate Array. 4, 23, 28, 29
FPU	Floating-Point Unit. 24
GPU	Graphics Processing Unit. 4, 21
HBM	High Bandwidth Memory. xxi, 23, 24, 25
HDC	Hyperdimensional Computing. 21
HMC	Hybrid Memory Cube. 23, 25
HPC	High Performance Computing. xxi, 6, 83
HRS	High Resistance State. 18, 19
IDC	International Data Corporation. 1, 2
IMC	In-Memory Computation. 47, 51
IoT	Internet of Things. vi
IRDS	International Roadmap for Devices and Systems. 78
ISA	Instruction Set Architecture. 79
LLC	Last Level Cache. 104
LRS	Low Resistance State. 18
LSB	Least Significant Bit. 33, 39, 44, 51, 54, 62
MAC	Multiplication and Accumulation. iv, viii, xvii, xviii, xix, xxii, 22, 32, 40, 41, 42, 43, 44, 45, 56, 57, 58, 59, 60, 61, 64, 65, 67, 111, 112
MC	Monte Carlo. xxii, 40, 44, 54, 66, 67, 68, 109
MIM	Metal-Insulator-Metal. 18, 20
ML	Machine Learning. 4, 26
MLC	Multi-Level Cell. 16, 17, 18

MOSFET	Metal Oxide Semiconductor Field Effect Transistor. xvii, 35, 42, 48, 56
MPU	Memory Processing Unit. 27
MRAM	Magnetoresistive RAM. 21
MSB	Most Significant Bit. 33, 39, 44, 62
MTJ	Magnetic Tunnel Junction. xix, xxii, 16, 17, 80, 81, 83, 98, 99, 101, 102, 103, 104, 105, 106, 109
MVM	Matrix-Vector Multiplication. 25, 26
NMOS	N-channel Metal-Oxide-Semiconductor. 36, 42, 43, 58, 61, 101, 105
NN	Neural Network. 23, 24, 56
NoC	Network-on-Chip. 22
NVM	Non-Volatile Memory. iv, vii, 5, 8, 11, 12, 14, 17, 18, 19, 25, 78, 79, 82, 83, 87
NVMe	Non-Volatile Memory Express. 4
OS	Operating Systems. 23, 79, 88, 89
PCM	Phase Change Memory. xxi, 17, 18, 78
PE	Processing Element. 11
PIM	Processing in Memory. iii, iv, vi, vii, xvii, xxi, xxii, 3, 4, 5, 6, 7, 8, 9, 11, 12, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 32, 33, 46, 47, 56, 57, 61, 65, 88
PMOS	P-channel Metal-Oxide-Semiconductor. 100, 105
PuM	Processing using Memory. 23
PV	Process Variation. 100
PVT	Process, Voltage, and Temperature. xxii, 5, 8, 32, 46, 47, 48, 49, 52, 53
PW	Pulse Width. 33, 35, 36, 39, 40, 42, 44
PWM	Pulse Width Modulation. 60
PZT	Lead Zirconate Titanate. 19, 20
ReRAM	Resistive Random Access Memory. xxi, 21, 22
ResNet	Residual Network. 21

RL	Rotation Layer. 80, 98
RMS	Root Mean Square. 51
RNN	Recurrent Neural Network. 21
RRAM	Resistive RAM. xxi, 12, 14, 16, 18, 19, 20, 25, 27
SBT	Strontium Bismuth Tantalate. 19
SEU	Single-Event-Upset. 20
SIMD	Single Instruction Multiple Data. 24, 29
SIMT	Single Instruction Multiple Threads. 27
SLC	Single-Level Cell. 15, 17
SNR	Signal to Noise Ratio. xxi, 37, 38, 44, 48
SoC	System-on-Chip. 18
SRAM	Static Random Access Memory. iii, iv, vii, viii, xvii, xviii, xix, xxi, xxii, 5, 6, 7, 8, 9, 11, 12, 14, 15, 19, 20, 21, 25, 32, 33, 34, 35, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 51, 53, 54, 55, 56, 57, 58, 59, 61, 63, 65, 66, 67, 72, 74, 75, 78, 98, 111, 112, 113
STD	Standard Deviation. xxii, 67
STT-RAM	Spin Transfer Torque Random Access Memory. iii, iv, vi, vii, viii, xviii, xix, xxi, xxiii, 5, 6, 7, 8, 9, 11, 12, 14, 16, 17, 18, 20, 21, 25, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113
TCMA	Thermal-Controlled Magnetic Anisotropy. 97
TMR	Tunnel Magnetoresistance Ratio. 82, 99, 100, 106, 108
TPU	Tensor Processing Unit. 4
TSF	Thermal Stability Factor. 98
TSMC	Taiwan Semiconductor Manufacturing Company. xxii, 27, 49
TSV	Through Silicon Via. 23
VCMA	Voltage-Controlled Magnetic Anisotropy. 100
VM	Volatile Memory. iv, vii, 5, 6, 8, 12
WER	Write Error Rate. 98, 100, 104

WL	Word-Line. iv, viii, xvii, xxi, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 44, 47, 48, 49, 50, 51, 56, 58, 60, 62, 63, 112

Chapter 1

Introduction

We are currently navigating through an era overwhelmed by an unprecedented surge in data from diverse sources. Recent findings highlight that Google processes 3.5 billion searches daily, resulting in over 2.5 quintillion bytes of data. Concurrently, projections from Dell EMC suggest a doubling of global data every two years, with expectations of reaching 40,000 Exabytes (trend depicted in Fig. 1.1, a phenomenon widely recognized as Big Data within the scientific community [39].

From an economic perspective, International Data Corporation (IDC) forecasts underscore Big Data's growing significance as a revenue generator. Specifically, the global market value of Big Data and business analytics rose by over 50% from $122 billion in 2015 to $187 billion in 2019 [40]. However, understanding Big Data necessitates examining its technological foundations, historical development, and the challenges posed by its voluminous growth. The primary challenge lies in storing, managing, and computing the vast quantities of data efficiently, as depicted in Fig. 1.1.

Big Data encompasses massive, varied datasets that often exceed the capabilities of traditional computing systems. According to the Gartner dictionary, Big Data is characterized by its volume, velocity, and variety. However, current systems fall short in efficiently storing, analyzing, and visualizing this data, which significantly hampers the ability to support decision-making processes. As a result, the scientific community continues to face challenges in extracting meaningful insights from this wealth of data.

Big Data's influence extends across diverse domains, including business, management, and research, where it has revolutionized numerous sectors. Its applications span global economies, healthcare, retail, manufacturing, astronomy, environmental science, and large-scale e-commerce. This extensive usage highlights the pressing need for advanced computing systems capable of effectively managing and processing such vast datasets.

The rapid growth of data volumes, propelled by Big Data, has exposed critical limitations in tra-

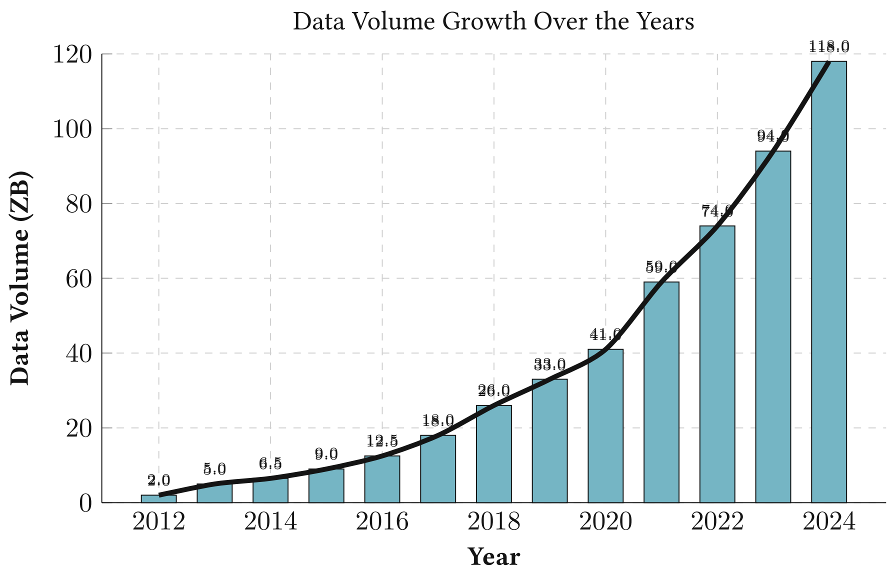


Figure 1.1: Histogram of data volume and future prediction

ditional computing architectures, particularly the von Neumann architecture. This processor-centric model, also known as the Princeton model, has served as the foundation for computer systems for decades [41]. However, its design, which necessitates significant data movement between memory and the CPU, proves inefficient for large-scale data processing. The von Neumann model consists of essential components, including a processing element with an arithmetic logic unit, processor registers, memory for storing data and instructions, external mass storage, and input/output mechanisms, all interconnected via a shared bus [42]. Fig. 1.2 illustrates this architecture.

Main memory plays a pivotal role in nearly all digital systems, from data centers and clouds to mobile devices and healthcare systems. Despite advancements, modern applications demand faster and more efficient data analysis. A study by IDC projects global data to surpass 100 ZB. However, memory bandwidth limitations remain a significant bottleneck [25, 43–48]. This bottleneck arises due to two primary factors: the processor-centric design and the memory wall. The former involves substantial data movement between memory and the CPU, while the latter highlights the challenges in developing high-capacity, low-latency, and cost-effective memory technologies [49–53].

Over the past decade, the data demands of applications such as Big Data, genome detection, and pattern recognition [49,54] have intensified, requiring extensive real-time or offline data transfer. To address these challenges, novel hardware accelerators tailored for data-centric applications have emerged. These accelerators leverage distributed memory systems and parallel processing architectures, emphasizing the importance of reliable memory modules [55, 56].

Scaling down charge-based memory technologies such as Dynamic Random Access Memory (DRAM) and flash has exacerbated these challenges. As these technologies approach smaller nodes, issues

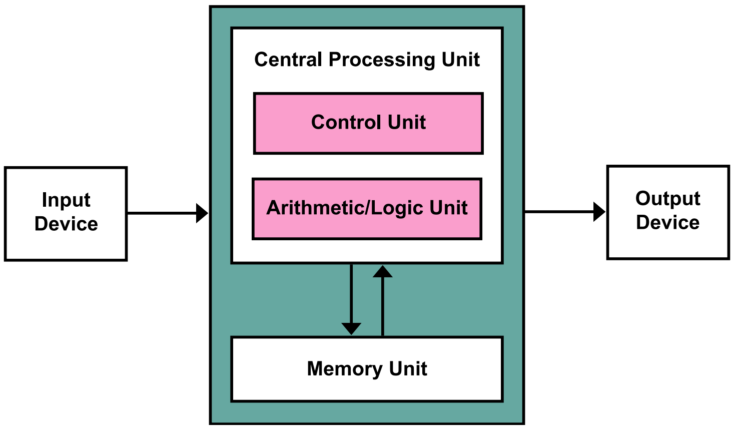


Figure 1.2: The von Neumann architecture and CPU-centric design approach [23]

like increased leakage power, reduced reliability, and higher costs become prominent [57, 58]. Similarly, Complementary Metal-Oxide Semiconductor (CMOS) downscaling faces limitations, further compounding the energy and reliability challenges [59]. These scaling constraints emphasize the urgent need for innovative memory architectures to overcome the memory wall.

Among the proposed solutions, Processing in Memory (PIM) has emerged as a particularly promising approach. By performing computations directly within memory, PIM minimizes data movement and significantly reduces energy consumption [60]. Studies reveal that PIM can reduce energy consumption by up to 49.1% and improve performance by 44.6% compared to traditional architectures [61]. This positions PIM as a viable solution for handling the growing demands of Big Data applications.

At the micro-architecture level, contemporary computing systems face three critical challenges [41]: the memory wall, parallel processing limitations, and the power wall. The memory wall arises from the latency and bandwidth gap between memory and processors, while the complexity of parallel programming and data management limits multi-core utilization. The power wall, stemming from thermal constraints and power dissipation, restricts further frequency scaling. Together, these challenges emphasize the need for innovative approaches to reduce data movement and enhance system efficiency.

Studies, such as [24], provide valuable insights into the energy and latency costs associated with data movement (see Fig. 1.4, 1.5,and 1.6) By using microarchitecture benchmarks with high memory access footprints, they quantitatively analyze energy consumption for various operations, revealing that data movement accounts for a significant portion of energy use. Similarly, research highlights PIM's potential to address these inefficiencies, particularly in embedded devices where data movement constitutes over 67.5% of energy consumption [25].

To address the memory wall and the associated data challenges, several forward-thinking strategies have been proposed:

1. **Process In Memory (PIM):** This approach eliminates data transfer by processing data directly within memory, making it ideal for real-time analytics and large datasets [22, 60, 62].
2. **Approximate Computing (AxC):** By accepting controlled inaccuracies, AxC reduces energy consumption and speeds up processing [50, 63–66].
3. **Non-Volatile Memory Express (NVMe) and Storage-Class Memory:** These technologies enhance performance by merging memory speed with storage permanence [67, 68].
4. **Hardware Accelerators (Graphics Processing Units (GPUs), Field Programmable Gate Arrays (FPGAs), Tensor Processing Units (TPUs)):** Specialized hardware accelerates tasks like machine learning and simulations [69].
5. **Distributed and Edge Computing:** This paradigme reduces latency by processing data closer to its source or across multiple machines [70].
6. **Quantum Computing:** Leveraging qubits, quantum systems promise unparalleled processing speeds [71, 71].
7. **3D Stacking and Advanced Packaging Technologies:** These technologies improve communication between components, mitigating the memory wall [72].

While techniques such as AxC and hardware accelerators have been explored, they come with limitations. AxC sacrifices accuracy for energy efficiency, which may not suit all applications [63]. Hardware accelerators, although efficient, require complex programming models. In contrast, PIM offers a scalable and generalized solution, reducing energy and latency by minimizing data movement between processor and memory [60]. Consequently, this thesis focuses on PIM-based solutions for overcoming traditional memory architecture limitations. By leveraging both volatile and non-volatile memory technologies, we aim to address critical challenges such as energy efficiency and latency, pushing the boundaries of data processing efficiency.

1.1 Problem Motivation

Modern data-centric applications, such as Machine Learning (ML) and real-time analytics, demand high-performance memory solutions that can overcome the limitations of traditional memory hierarchies, often referred to as the "memory wall." Conventional architectures struggle to support the large volumes of data required for these applications, leading to significant latency and energy inefficiencies due to frequent data movement between memory and processing units. This challenge drives the exploration of PIM techniques, where computations occur closer to the data, minimizing the bottlenecks associated with conventional memory systems. Consequently, selecting effective PIM-compatible memory technologies is crucial to achieving both performance and energy efficiency improvements.

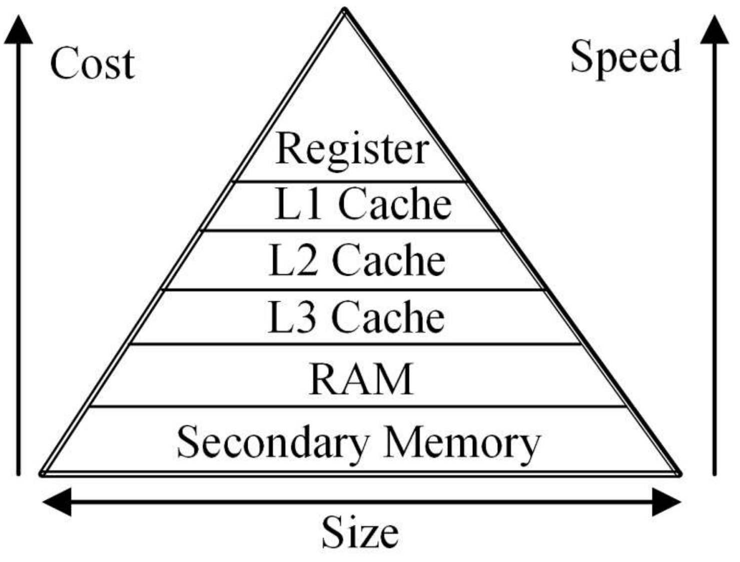


Figure 1.3: Memory hierarchy: data movement from main memory to register files

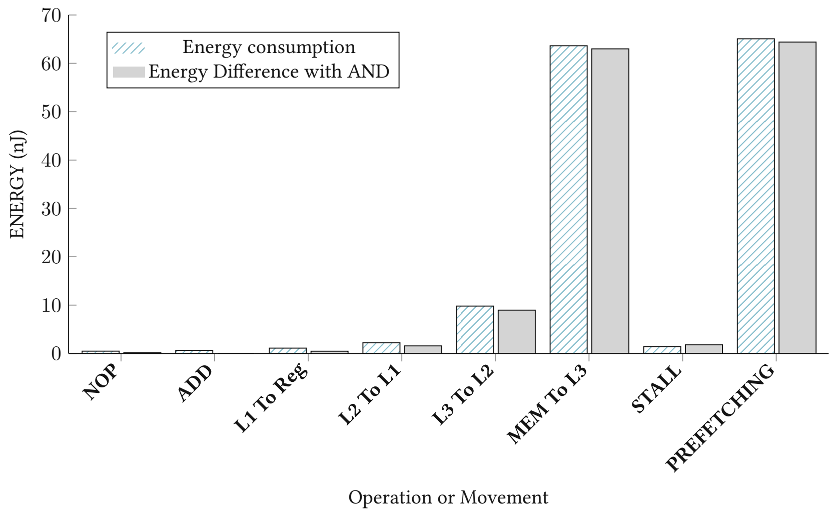


Figure 1.4: Energy consumption of data movement (nJ) and comparison with AND operation [24]

This thesis explores Static Random Access Memory (SRAM) as a promising candidate for volatile memory and Spin Transfer Torque Random Access Memory (STT-RAM) as a suitable choice for Non-Volatile Memory (NVM) in PIM architectures, addressing the unique challenges and benefits each offers.

SRAM-based systems present key advantages as Volatile Memory (VM) due to their high read/write speeds and reliability, which makes them ideal for applications demanding fast data retrieval and high processing throughput in real-time. However, conventional SRAM-based computing faces non-linearities and Process, Voltage, and Temperature (PVT) variations that can degrade performance in PIM applications. This thesis proposes solutions such as the OPTIMA framework, which models Bit-Line (BL) discharge behavior in 6T-SRAM, enhancing design-space exploration for in-memory multipliers. Further, techniques like AID [2] and EMAC [3] have been introduced to address accuracy limitations by

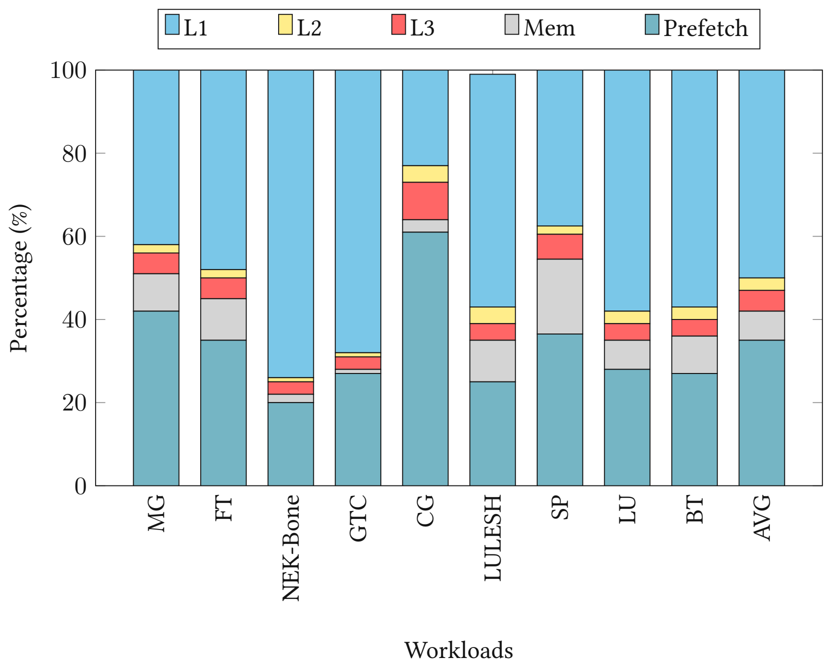


Figure 1.5: Energy consumption of data movement between memory hierarchy levels [25]

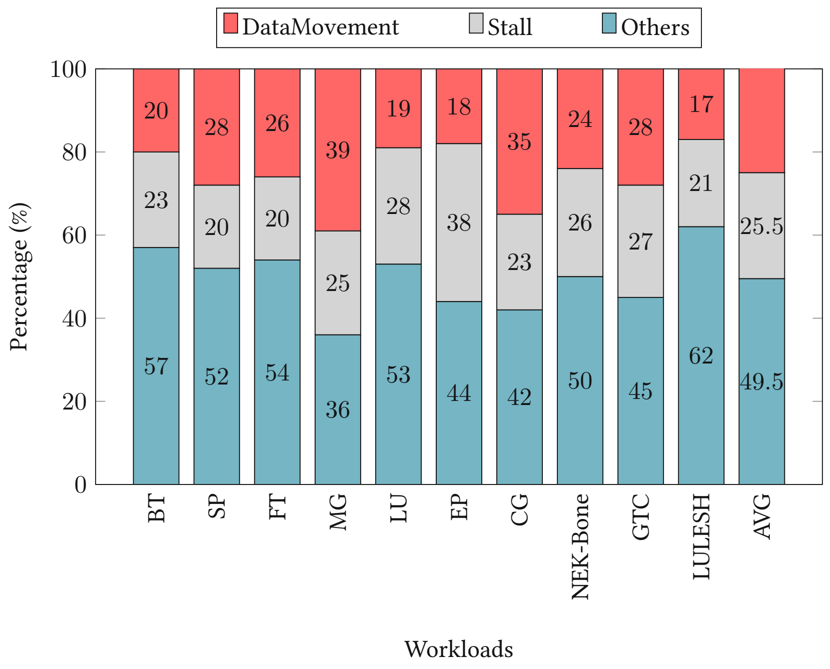


Figure 1.6: Relative energy of data movement for High Performance Computing (HPC) benchmark [25]

tackling BL non-linearities, thereby improving energy efficiency and computational precision. These contributions position SRAM as a viable PIM solution that aligns with the high-performance demands of VM applications in data-centric workloads.

STT-RAM, with its non-volatility, high density, and low leakage power, represents an advantageous alternative for non-volatile memory in PIM. Despite these benefits, challenges such as high write energy and latency have historically limited STT-RAM's industrial adoption. This thesis addresses these issues through a dual approach: first, by developing the HOPE framework, an open-source STT-RAM memory controller that enables accurate performance and energy evaluation within the gem5 simulator; and second, by proposing approximation techniques to reduce write energy and latency. These methods exploit the stochastic switching behavior of STT-RAM cells, achieving significant energy and latency reductions, and enabling reliable integration of STT-RAM in PIM applications. By improving write efficiency, this work establishes STT-RAM as a promising NVM candidate for PIM, balancing energy efficiency with the endurance required for data-intensive applications.

The complementary qualities of SRAM and STT-RAM in PIM applications provide a robust foundation for addressing the diverse memory requirements of modern computing systems. While SRAM delivers the speed and precision needed for high-frequency operations, STT-RAM offers energy-efficient, persistent storage, essential for applications that demand both immediate data access and non-volatility. By exploring these technologies in parallel, this thesis contributes to a more nuanced understanding of PIM's potential, aiming to overcome the memory wall through targeted enhancements in both volatile and non-volatile memory domains. Together, SRAM and STT-RAM offer a path towards scalable, efficient, and reliable PIM systems suited to the next generation of data-centric applications.

1.2 Research Questions and Objectives

Building upon the challenges posed by the rapid expansion of data, architectural bottlenecks, and inefficiencies in current memory systems, this thesis aims to explore innovative memory-centric solutions. The primary focus is on the application of PIM techniques, leveraging both STT-RAM and SRAM technologies to overcome the memory wall. The research questions guiding this work are:

1. **What are the key reliability challenges faced by STT-RAM and SRAM in PIM-based architectures, and how can these be mitigated?**
   Both volatile and non-volatile memory technologies present distinct reliability issues, such as write errors in STT-RAM and non-linearity in SRAM-based computations. The objective of this thesis is to investigate these problems and develop hardware-level solutions to enhance the reliability and overall robustness of PIM systems.
2. **How can STT-RAM-based PIM systems be analyzed for performance, power, and energy consumption before deployment?**
   Towards this the objective is to develop an open-source framework for evaluating power, latency, and energy characteristics of STT-RAM-based PIM architectures under real workloads. It will integrate analytical modeling, simulations, and empirical validation to assess system efficiency prior to deployment.

3. **How to optimize the energy consumption and latency of STT-RAM write operations without compromising the reliability?**
   Towards this, the objective is to investigate approximation-oriented techniques in STT-RAM that exploit its inherent stochastic switching behavior to enable energy-efficient write operations while ensuring system-level reliability.

4. **How to perform a fast design space exploration that can efficiently analyze energy consumption latency, and accuracy in a discharge-based in-SRAM computing circuits, while accounting for PVT variations?**
   Towards this, the objective is to develope a modeling framework designed to accelerate design-space exploration for discharge-based in-SRAM computing circuits that enables efficient trade-off analysis by providing fast and accurate evaluation of energy, latency, and accuracy while incorporating the impact of PVT variations.

1.3 Structure of the Thesis

This thesis addresses the research questions and challenges outlined in section 1.2, presenting original contributions to the fields of STT-RAM and SRAM technologies as candidates for PIM architectures. The thesis is organized into four main chapters:

Chapter 2 provides a review of the state-of-the-art approaches in memory technology, focusing on both NVM and VM options. This literature review establishes the foundational context for the thesis, examining the key characteristics, limitations, and opportunities within STT-RAM and SRAM technologies relevant to PIM applications.

Chapter 3 discusses SRAM as a promising VM candidate for PIM applications, highlighting the work conducted to mitigate non-linearity issues inherent in SRAM-based computations. This chapter presents methods such as the OPTIMA framework [1] for modeling SRAM discharge behavior, as well as the AID [2] and EMAC [3] techniques, which are developed to improve the accuracy, energy efficiency, and reliability of in-SRAM computation.

In Chapter 4, we focus on addressing STT-RAM write errors, energy, and latency. We propose a novel optimization technique that leverages stochastic switching and circuit-level approximations to reduce write energy and latency while enhancing robustness against soft errors, process variations, and write error reliability challenges [5]. In addition to this, we present a modeling and benchmarking framework for STT-RAM-based systems, integrating an extendable STT-RAM memory controller into the gem5 simulator. This framework enables the design and evaluation of STT-RAM architectures, analyzing power, latency, and throughput before deployment.

Finally, Chapter 5 draws together the findings from the previous chapters, summarizing the contributions of this research to the fields of memory technology and PIM. This chapter also explores future research directions, offering insights into potential improvements and further studies that could extend the work on STT-RAM and SRAM technologies within advanced computing architectures.

Chapter 2

Background and Related Work

To address data movement overhead in memory-centric applications, advanced memory technologies have enabled closer integration of memory and processing units, leading to the concept of PIM. PIM mitigates data movement overhead between memory and processing units, thereby reducing latency and energy consumption. This is achieved by integrating computational elements within or near memory, such as in-memory ICs, 3D-stacked layers, memory controllers, or processor caches [73].

PIM methodologies can be categorized into two types: (1) architectures where memory chips execute basic operations and (2) designs where data is loaded from memory into adjacent processing elements (Processing Elements (PEs)) for computation. Although PIM concepts are not new, past efforts faced obstacles such as the complexity of integrating processing and memory elements on a single chip and the limited benefits of reducing memory access overhead for workloads with relatively small working sets (e.g., early database queries or embedded system applications).

With the development of 3D-stacked technology, PIM architectures have gained renewed interest across various application domains. This thesis categorizes recent PIM approaches into three groups: (a) modifying traditional memory technologies (e.g., DRAM, SRAM, NVMs) to enable basic computations, (b) enhancing memory controllers with computational capabilities, and (c) 3D-stacked architectures that integrate multiple processing and memory components on a single chip. This work specifically focuses on advancing digital in-STT-RAM and analog-based in-SRAM PIM architectures (category a), aiming to improve energy efficiency and computation accuracy. These two memory types were selected due to their potential for non-volatility and high-density storage (STT-RAM) and their ability to support fast, in-memory analog computations (SRAM), making them promising candidates for energy-efficient and scalable PIM solutions.

2.1 Different Memory Technologies

Before discussing the state-of-the-art PIM techniques, it is critical to first analyze memory technologies concerning power, energy, and latency. Understanding these attributes helps system designers tailor their choices to application demands. Table 2.1 compares key factors such as endurance, read/write energy, latency, and CMOS compatibility, with insights from literature [7–21]. The selection of memory for PIM must balance speed, endurance, and power efficiency to ensure feasibility. While volatile memories offer high-speed access, they require continuous power, whereas non-volatile memories provide data retention but often suffer from higher latency and limited endurance.

Broadly, memory technologies fall into two categories:

1. **Volatile memories (VMs):** These require power to retain data and are primarily used for temporary storage in fast-access applications. SRAM and DRAM are the dominant types, offering low latency and high-speed operation. As shown in Table 2.1, SRAM has the lowest read/write latency (<5 ns) and the highest endurance (10^{16} cycles), making it ideal for cache memory and high-performance computing. However, its large cell size (>100 F^2) limits its scalability. DRAM provides a smaller cell size (4–12 F^2) and similar latency ($\approx$ 2–5 ns), making it more area-efficient. However, due to its need for periodic refresh operations, DRAM consumes significant power, reducing its suitability for energy-constrained PIM applications [74]. The high access speed of VM makes it well-suited for in-memory computing, but its volatility limits long-term data storage.
2. **Non-volatile memory (NVM):** NVM retains data without power and is commonly used for persistent storage. Table 2.1 highlights key NVM types such as flash, STT-RAM, PCMRAM, and Resistive RAM (RRAM), each with distinct trade-offs. Flash memory, widely used in SSDs and storage devices, offers high density but suffers from excessive write latency (500 µs) and limited endurance (10^6 cycles), making it unsuitable for frequent memory operations. Emerging NVM technologies, such as PCMRAM and RRAM, exhibit better endurance (10^{10}–10^{12} cycles) but still suffer from higher write latencies (>30 ns). In contrast, STT-RAM balances non-volatility, high endurance (10^{16} cycles), and moderate latency (5–20 ns), approaching DRAM-like speeds without requiring refresh power. Furthermore, STT-RAM has a lower write energy (0.1–2.5 pJ) compared to PCMRAM (18 pJ), making it a more energy-efficient alternative. These attributes make STT-RAM a strong candidate for digital PIM, whereas SRAM remains the preferred choice for analog in-memory computing due to its ultra-fast access speed and high endurance. Continued advancements in NVM aim to further optimize speed, energy efficiency, and endurance, making these technologies increasingly viable for PIM applications [8].

Table 2.1: Comprehensive comparison of different memory technologies [7–21]

Characteristic	**STT-RAM**	**PCMRAM**	**RRAM**	**Fe-FET**	**FLASH**	**SRAM**	**DRAM**
Non-volatility	+	+	+	+	+	-	-
Data Retention (years)	10	10	10	10	10	-	-
Cell Endurance (cycles)	10^{16}	10^{12}	10^{10}	10^{12}	10^{6}	10^{16}	10^{15}
Cell Size (F^2)	6-20	4-8	4	4-8	4-6	>100	4-12
Technology node (nm)	45	65	40	5	15	10	32
Read Latency (ns)	2-20	20-50	<10	10	25x10^3	<5	2
Write Latency (ns)	5-20	30	5	10	500x10^3	<5	5
Erase Latency (ns)	5-20	30	10	10	2 ms	<5	5
Write Energy (pJ)	0.1-2.5	18	0.1	1	0.1 - 1	<0.1	<0.1
Erase Energy (pJ)	1	18	0.1	1	10^3	<1	<1
Suppliers	Toshiba, Hitachi	Samsung, Intel, WD, IBM	Panasonic, Micron	Globalfoundries, FMC	Micron, Samsung	Qualcomm, Intel	Samsung, SK Hynix

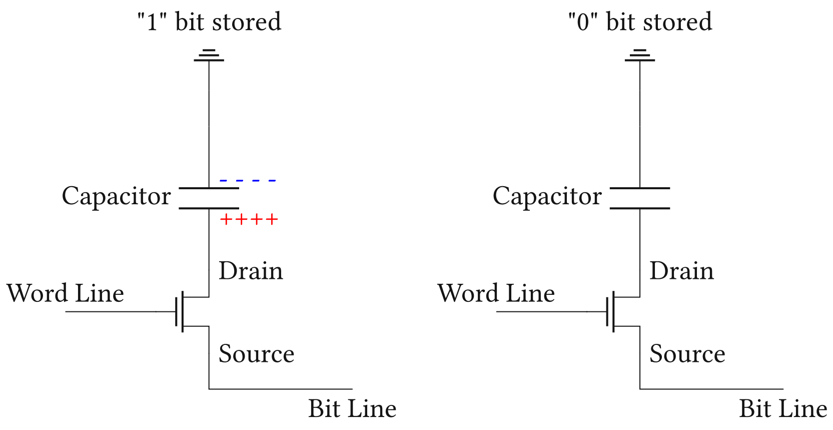


Figure 2.1: Schematic of a DRAM cell [26]

DRAM Memory Technology:

DRAM has been a cornerstone of computational systems for decades, originally introduced by Robert H. Dennard et al. [75]. The development of DRAM continues to advance, particularly with the advent of new CMOS fabrication technologies that push toward smaller feature sizes, now reaching $10nm$ and smaller, as highlighted in recent study [76]. At its core, a DRAM cell features a simple yet effective $1T1C$ (one transistor and one capacitor) structure (see Fig. 2.1, more details in [26]).

One of the DRAM's key advantages is its bit cell's architecture, which achieves high storage density due to minimal area requirements per cell. This efficiency results in compact DRAM cell sizes, recently achieving dimensions as small as $4F^2$ [77]. However, DRAM's design is not without its drawbacks. The primary issue lies in charge leakage from the tiny capacitors used to store bit states, necessitating regular refresh cycles to retain data. These refresh cycles, inherent to both maintenance and read operations (as reading a DRAM cell discharges the capacitor, requiring a restore operation afterward), are significant energy consumers. Studies indicate that refresh operations can account for over 20% of a DRAM device's total energy consumption [78, 79].

Moreover, DRAM's performance speed lags behind that of SRAM and various NVMs. Comparative analyses (see Table 2.1) reveal that SRAM, STT-RAM, and RRAM all significantly outpace DRAM, with speed improvements by a factor of up to 50 in read and write latencies [80].

SRAM Memory Technology:

SRAM, a staple in electronic devices since the 1960s, offers a distinct approach to data storage without relying on capacitors. As depicted in Fig. 2.2, a standard SRAM cell consists of six transistors. The data storage mechanism involves two cross-coupled inverters (formed by transistors M_1, M_2, M_3, M_4), which enable the cell to maintain its state as long as power is supplied. Transistors M_5 and M_6 serve

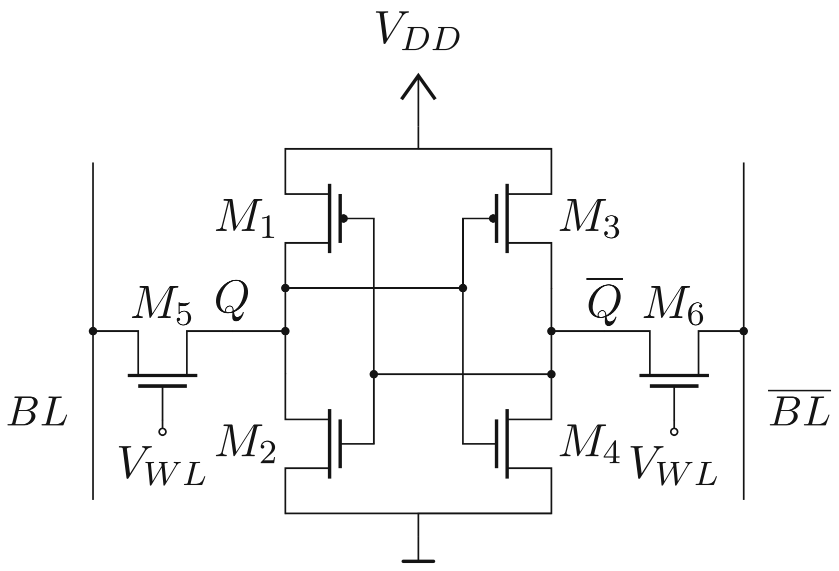


Figure 2.2: Schematic of an SRAM cell [2]

as gateways for reading and writing operations.

For read operations, activating at least one access transistor suffices, but leveraging both—alongside precharged BLs—can enhance the speed, as demonstrated in recent advancements [81]. Write operations, on the other hand, necessitate aligning the new data bit with both BLs: the direct bit value with the true BL and its complement with the inverted line. Pre-charging the BLs has also been shown to optimize write performance [82].

While SRAM cells outpace DRAM in terms of read and write speeds, they come with the trade-off of larger cell sizes. This results in lower memory density and higher production costs. Given these characteristics, SRAM's use is predominantly reserved for cache memory and register files within Central Processing Units (CPUs), where its rapid access times justify the limited memory capacity and higher expense.

FLASH Memory Technology:

Flash memory, invented by Masuoka in the late 1980s [83], is a durable, non-volatile storage medium widely used in electronic devices. It is composed of arrays containing numerous flash cells, each based on a floating gate transistor mechanism. These transistors utilize a control gate and a floating gate to manage the charge within the floating gate, as illustrated in the referenced Figure 2.3. Electrons are introduced to or removed from the floating gate through a process known as Fowler-Nordheim tunneling [84], which alters the transistor's threshold voltage (V_{th}). This alteration defines the cell states in Single-Level Cell (SLC) flash memory, distinguishing between the binary '1' and '0' through different V_{th} levels.

The enduring nature of flash memory stems from its ability to maintain the charge in the floating gate, ensuring data retention even without power. The memory is 'programmed' by charging the floating gate, and it can be 'erased' by applying an inverse voltage, clearing the stored data.

Flash memory comes in two main types: NOR and NAND. NAND flash is preferred for its higher storage capacity despite NOR flash's faster read capabilities. Erasing data in NAND flash affects entire blocks, necessitating a sequence of steps for rewriting data: identifying a free block, erasing it, transferring modified data from the old to the new block, and then marking the old block as free.

Moreover, flash memory supports Multi-Level Cells (MLCs), which store multiple bits per cell through various threshold voltage levels, enhancing storage density. Despite its benefits, flash memory's drawbacks include slower read/write speeds, limited cell lifespan due to wear from erasing cycles, and higher energy consumption for writing compared to technologies like STT-RAM or RRAM.

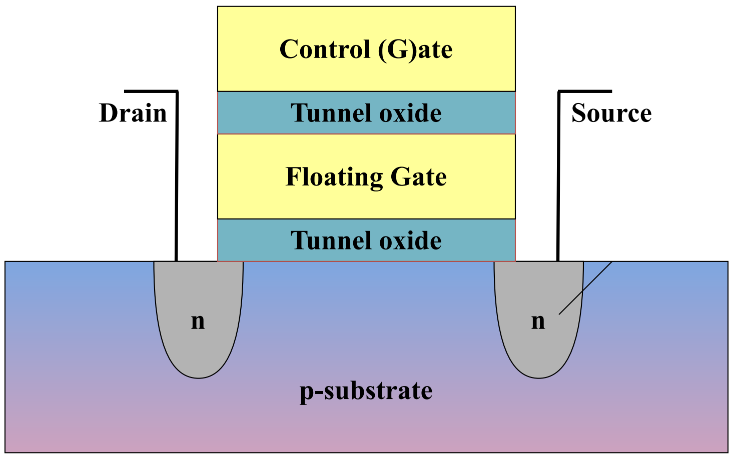
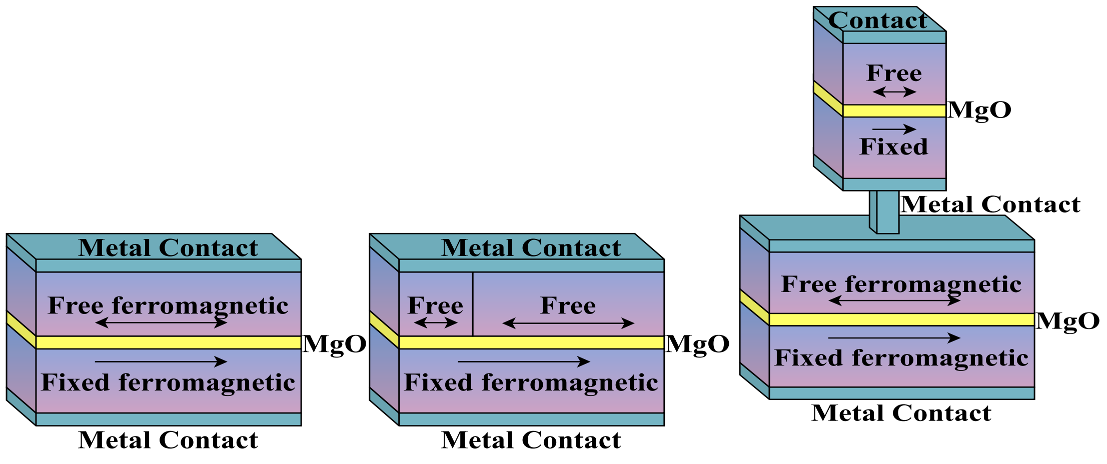


Figure 2.3: Schematic of an FLASH cell [27]

STT-RAM Memory Technology:

STT-RAM is a durable memory technology that relies on magnetic layers to store data [85]. Inside each STT-RAM cell is a component known as the Magnetic Tunnel Junction (MTJ), comprising a fixed magnetic layer, a non-magnetic barrier, and a free magnetic layer. The fixed layer's magnetic direction stays constant, while the direction of the free layer can be altered during operation. These layers are typically constructed using cobalt-iron-boron (CoFeB) for the magnetic parts and magnesium oxide (MgO) for the barrier, as noted in the referenced material. The MTJ's resistance varies depending on whether the magnetic layers align; it's lower when they match and higher when they oppose. This resistance variation encodes the bit's state in the cell [86–88].

A key benefit of STT-RAM is the compact size of its MTJ elements, which can be as small as six times the feature size squared ($6F^2$). This small footprint allows for densely packed memory cells. For even greater densities, MLC configurations are used, involving at least two STT-RAM cells arranged either in parallel or in series. A SLC setup is depicted in Fig. 2.4a, while parallel and serial MLC arrangements are illustrated in Figs. 2.4b and 2.4c, respectively. In MLC designs, the size variation of the free layers (in parallel arrangements) or the entire MTJ stack (in serial arrangements) is crucial. Larger magnetic layers require more current to alter their magnetic direction than smaller ones. This size-dependent change in resistance helps distinguish between the different levels in MLCs during readout processes [68, 88].



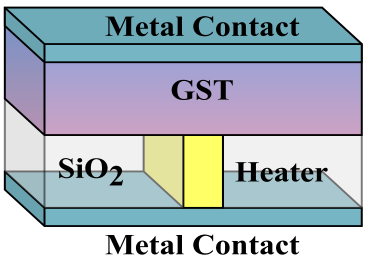
(a) STT-RAM:SLC (b) STT-RAM:MLC parallel (c) STT-RAM: MLC series

Figure 2.4: Schematic of an STT-RAM cell [28]

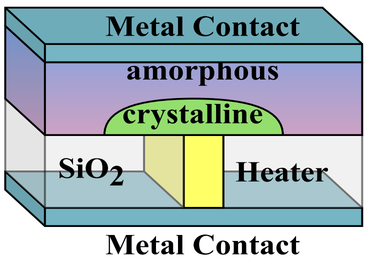
Phase Change Memory (PCM) Memory Technology:

PCM is a NVM technology that utilizes chalcogenide glass, typically composed of germanium, antimony, and tellurium (Ge2Sb2Te5 or GST), to store data [89, 90]. This material can switch between crystalline and amorphous states when heated, a process detailed in various studies. In its crystalline form, which represents a logical "1", GST has low electrical resistance, while its amorphous state, signifying a logical "0", has significantly higher resistance—the difference being roughly a thousandfold. Each PCM cell includes a GST element alongside a heater, often made from titanium nitride (TiN), which rapidly heats up to at least 600°C [91].

To erase data or set a cell to "0", the GST is heated to approximately 600°C, causing it to lose its crystalline structure. It then cools down quickly, within tens of nanoseconds, resulting in an amorphous state. To write data or set a cell to "1", the GST is heated just enough to reach the crystallization temperature but not so much that it melts, and then it is cooled down more gradually. This precise control is necessary to ensure that the GST transforms completely into the crystalline state without accidentally remelting due to excessive heat. These state transitions are depicted in Fig. 2.5a for erasing and Fig. 2.5b for writing the data bit.



(a) PCM cell structure



(b) PCM: state transition $0 \rightarrow 1$

Figure 2.5: Schematic of an PCM cell and the state transition [29]

For higher storage capacities, PCM uses MLC technology, which stores multiple bits in a single cell by partially crystallizing the GST to achieve intermediate resistance levels. This method involves precisely controlling the transition process to create semi-amorphous states with resistance values that fall between the fully amorphous and fully crystalline extremes.

The advantages of PCM lies in its small cell size and its swift read and write operations, making it a promising solution for future memory requirements [89].

RRAM Memory Technology:

RRAM is a type of NVM technology. It stores bits using the resistance levels in a Metal-Insulator-Metal (MIM) configuration, which forms the core structure of an RRAM cell (illustrated in figure 2.6a). The metal layers serve as electrodes connecting the cell to the memory circuit. A Low Resistance State (LRS) represents a logical "1," while a High Resistance State (HRS) signifies a logical "0." The state of the cell is changed by applying a voltage of a specific strength, direction, and duration [31].

RRAM operation can be influenced in two ways: either through the strength of the voltage applied (unipolar RRAM), or the direction of the voltage (bipolar RRAM). Similar to STT-RAM, PCM, and DRAM, RRAM can achieve high storage densities, with cells as small as $4F^2$, and boasts quick write and read times ranging from 5 to 50 ns. RRAM offers several advantages. First, cells do not require an access transistor and can be wired directly to the bit and word lines, simplifying cell access. This straightforward architecture allows for RRAM cells to be stacked in layers, creating 3D memory structures that significantly increase density. This is especially beneficial for memory packages or System-on-Chip (SoC) designs where space is at a premium [92].

The insulator layer, typically composed of slightly conductive materials like HfO_x or TaO_x, is critical in an RRAM cell. To set a cell to LRS, a high voltage is applied, forming nanoscale conductive paths, or filaments, between the metal electrodes through the insulator. This process is made possible by the displacement of oxygen ions, which ordinarily prevent the formation of these filaments. When these ions are removed, they leave behind vacancies that facilitate conductivity. Conversely, to revert to HRS, oxygen ions are reintegrated into the vacancies, which breaks the conductive path and restores high resistance. The ability to switch between states by controlling the presence of oxygen ions within the dielectric is a key feature of RRAM, as detailed in the literature on cell failure mechanisms [93].

Ferroelectric Random Access Memory (FeRAM) Technology:

FeRAM is an advanced NVM technology. A FeRAM cell is structured similarly to an RRAM cell but includes a layer of ferroelectric crystals instead of an insulator, as depicted in Fig. 2.6b. These ferroelectric layers are made from materials like Lead Zirconate Titanate (PZT)[1] or Strontium Bismuth Tantalate (SBT)[2], which exhibit spontaneous polarization and an electric field [94, 95]. The relationship between polarization and the electric field is characterized by a hysteresis loop, indicating that the polarization can be manipulated by an external electric field and remains in a residual state once the field is removed, thereby retaining data.

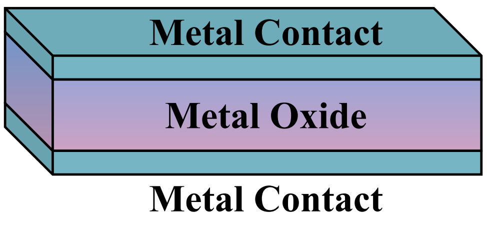
Ferroelectric materials, unlike ordinary crystals with symmetrically arranged atoms, contain crystals where an atom is displaced into a non-symmetrical position, creating a non-centrosymmetric structure. This displacement imparts a directional charge to the crystal, making it act like a dipole that determines the crystal's polarization direction.

An array of these ferroelectric crystals forms a FeRAM layer, where each crystal has its own dipole. Initially, these dipoles are unorganized, but with the application of a minimal external field, they can be oriented in the same direction. This alignment allows for the storage of a bit state in a FeRAM cell. The polarization—and thus the stored bit state—can be reversed by applying an external field in the opposite direction. The two polarization states correspond to logical "0" and "1". In a read operation, these states result in different charges that a sense amplifier can discern to determine the bit state. This read process is destructive, leading to a loss of charge, which makes a subsequent "Write after Read" procedure essential to restore the bit state.

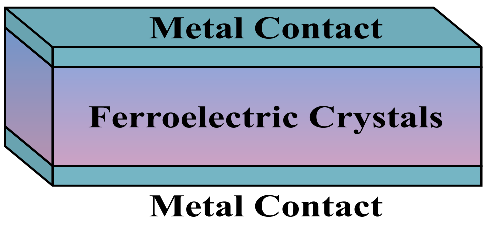
FeRAM boasts remarkable endurance, with up to 10^{14} write cycles, and offers a variety of cell designs that balance speed and density [96]. Configurations range from 6-transistor-2-capacitor (6T2C) structures, which resemble SRAM access times, to compact 1-transistor (1T) layouts, as shown in the literature [97]. However, the relatively large cell size, dictated by the charge needed for reliable sensing, and the necessity of a "Write after Read" process due to destructive reads, stand as the main drawbacks of FeRAM [96, 98, 99].

FeRAM cells are particularly susceptible to high temperatures, which can lead to two detrimental effects: Imprint and Thermal Depolarization. Imprint is an irreversible behavior where long-term ex-

[1] from the chemical formula $Pb[Zr_xTi_{1-x}]O_3$
[2] from the chemical formula $SrBi_2Ta_2]O_9$



(a) A single RRAM cell. The stack is also called MIM



(b) A single FeRAM cell. The ferroelectric crystals are called PZT which can change its capacitance by supplying a voltage

Figure 2.6: Basic structures of RRAM and FeRAM cells [30, 31]

posure to high temperatures reinforces the current polarization state at the expense of the cell's switchability. This effect can manifest at temperatures as low as $85°C$ over thousands of hours or within a few hours at $150°C$. Thermal Depolarization involves the loss of stored data states upon heating to temperatures approaching the material's Curie point. For instance, PZT cells lose their polarization at around $430°C$, but they are not damaged and can be rewritten once normal temperatures are resumed.

Thanks to the robust nature of ferroelectric data storage, FeRAM is significantly less affected by alpha particles and cosmic rays, leading to much lower Single-Event-Upsets (SEUs) rates compared to SRAM or DRAM [94]. Further research indicates that while the memory device is powered off, no SEUs occur in the CMOS access circuits of the FeRAM array.

Recent technological developments have allowed for FeRAM integration into smaller, $28nm$ process nodes, transitioning from older $130nm$ nodes and 1T1C structures. These advancements demonstrate FeRAM's compatibility with low-voltage operations (1.2V), expanding its applicability in the evolving landscape of memory technologies [30].

2.2 Process in Memory (PIM) Architectures and Classification

The field of PIM has experienced a significant increase in research activity, exploring various PIM models. This section provides a detailed background of recent advancements in this domain. We classified our analysis of PIM research into four distinct groups: the architectural design of PIM systems, their application domains, the memory technologies employed, and the methodologies used for evaluation.

2.2.1 Architectural Design of PIM

Processing at the Data Array: Numerous investigations have suggested the incorporation of computing functionalities at the level of the data array. These explorations primarily delve into harnessing the architecture at the cellular level within the data arrays to embed computational features.

Authors in [100] advocate for a co-processor utilizing STT-RAM aimed at enhancing the speed of

Convolutional Neural Network (CNN). They have implemented a cutting-edge $22nm$ technology across CMOS, SRAM, and STT-RAM. Their design for a CNN processing unit is capable of executing 3×3 convolution operations on a 2D image at $P \times P$ pixel locations, leveraging inputs from the buffer and filter coefficients stored in closely situated on-chip Magnetoresistive RAM (MRAM).

The authors in [101] capitalize on the analog properties inherent to traditional crossbar memory systems to activate crucial in-memory operations. This approach is distinct from previous methods that performed bit-wise calculations at the sense amplifier of each memory unit, as it enables bit-wise operations directly within the memory, eliminating the need to externalize the data. The team has implemented row-parallel computing, managing to achieve 1000 simultaneous additions/multiplications in a memory setup with 1000 rows.

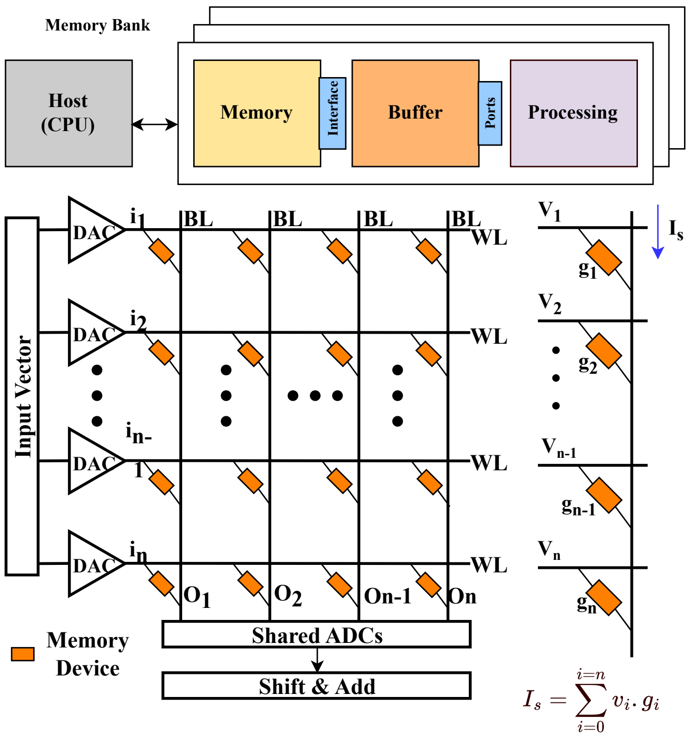
The authors in [102] suggest enhancing in-memory multiplication through innovative partition-based computing strategies. This includes data broadcasting/shifting across partitions, replacing the Wallace tree with a carry-save-add-shift (CSAS) multiplier, and introduction of a new full-adder configuration. This research takes advantage of a specific Resistive Random Access Memory (ReRAM) feature, the voltage-controlled variable resistance, to facilitate logic gates such as NOT, NOR, OR, and NAND.

The authors in [103] have developed a new mapping technique and dataflow strategy aimed at optimizing the reuse of weights and input data within an 8-bit ReRAM-based PIM setup. Their simulation, conducted using the NeuroSim simulator, indicates that their methodologies could reduce latency by 90% and energy consumption in interconnects and buffers by 68% on the Residual Network (ResNet)-34 benchmark.

The authors in [32] propose a ReRAM-based PIM design tailored for accelerating Recurrent Neural Networks (RNNs) (see Fig. 2.7). They introduce an PIM unit composed of three primary subarrays: crossbar subarrays for matrix-vector multiplication, special function units for nonlinear operations, and multiplier subarrays for element-wise calculations. Their techniques reportedly offer an average improvement of 79$\times$ over traditional GPU performance benchmarks.

The authors in [104] investigate reconfigurable design principles for a Computing In Memory (CIM)-based accelerator, which supports the operation of CNNs on already fabricated chips. The evaluation of this concept utilizes a modified Deep Neural Network (DNN) + NeuroSim framework to conduct a system-level performance assessment.

The authors in [105] present an approach for multibit in-memory Hyperdimensional Computing (HDC) that facilitates multibit operations using Ferroelectric Field Effect Transistor (FeFET) crossbar arrays for multiplication and addition, and FeFET multibit content addressable memories for associative searching.



$$I_s = \sum_{i=0}^{i=n} v_i . g_i$$

Figure 2.7: PIM architecture and ReRAM crossbar for matrix–vector multiplication proposed by [32]

Processing at the Row Buffer: A handful of research initiatives have explored circuit-level methods for instigating computational processes either within or close to row buffers and sense amplifiers. The authors in [106] have introduced a multiplication technique within the DRAM's subarray level that requires minimal alteration to the existing DRAM subarrays. This method simplifies multiplication into its constituent addition and logical AND operations, which enhances efficiency and cuts down on the computational burden for multiplication within DRAM. This innovation forms the foundation of a new PIM-DRAM bank design, outfitted with computational structures like adder trees and units for non-linear functions, all aimed at advancing machine learning acceleration. The design is also capable of supporting essential machine learning operations such as ReLU, batch-normalization, and pooling.

The authors in [107] have detailed a FeFET-based PIM architecture designed to expedite DNN inference. This design includes a digital in-memory engine for vector-matrix multiplication, leveraging the FeFET crossbar structure to facilitate bit-parallel calculation and sidestep the need for Analog-to-Digital Converter (ADC) seen in earlier mixed-signal PIM designs. Additionally, it introduces a specialized hierarchical Network-on-Chip (NoC) for more efficient data transmission and intermediate result processing.

The authors in [108] describe an experimental setup based on HBM2 that encompasses Multiplication and Accumulation (MAC) units and reducers, which together significantly boost the performance of matrix-vector multiplication tasks. The results from their study indicated substantial performance gains in both whole-bank and per-bank scheduling scenarios. Other researchers have also put forward various innovative PIM techniques with the aim of accelerating Neural Networks (NNs).

Processing in a unit near memory banks: The predominant body of work we have examined suggests computing units situated near memory, implementing streamlined cores that run a selection of CPU instructions to minimize the exchange of data between the memory and the processing unit. These studies encompass a range of approaches detailed in references [33, 35, 109, 110].

The authors in [109] contend that specific PIM mechanisms necessitate unique allocation and alignment of memory that aren't available in current memory allocation practices. These mechanisms, particularly in-DRAM copying, demand robust memory coherence management—a challenging task for analysis using proprietary systems or simulators. To address this, they have developed a prototype using an FPGA to fully integrate and evaluate these mechanisms with actual DRAM chips. This system includes a Processing using Memory (PuM) operations controller, which provides transparent control over PuM operations, and a custom memory controller that manages the refresh and timing for DDRx sequences initiating PuM operations. Accompanying software components, like the PuM operations library and a specialized Operating Systems (OS) layer, support memory management functions necessary for developers.

The authors in [111] delve into a high-performance NN accelerator architecture that utilizes logic dies similar to those in 3D High Bandwidth Memory (HBM) memory. The study discerns critical differences between HBM and Hybrid Memory Cube (HMC) memory in terms of their design for near-memory accelerators. They introduce data-fetching strategies like round-robin and group-wise broadcasting to effectively utilize the centralized Through Silicon Via (TSV) channels, optimizing data retrieval from DRAM dies to the logic die. They argue that this tailored approach for HBM differs from and outperforms conventional data-fetching strategies used in HMC for neural network accelerators.

The authors in [112] highlight the challenge of memory capacity bottlenecks as deep learning models and datasets grow. They propose a memory-centric deep learning system that transparently extends memory capacity for accelerators and facilitates fast inter-device communication essential for parallel training. This system, by pooling memory modules within the device-side interconnect and uncoupling them from the host interface, has demonstrated an average performance speedup of $2.8\times$ and expanded the system's memory capacity significantly.

The authors in [33] have devised a forward-thinking PIM architecture that integrates flawlessly with standard commercial processors and serves as a direct substitute for conventional DRAM. Their architecture showcases a PIM design utilizing HBM2 DRAM technology, detailed at $20nm$, and aligns this with an unmodified commercial processor while also crafting the essential software infrastructure. This PIM setup includes an HBM DRAM die designed for PIM, a bank connected to a PIM execution unit

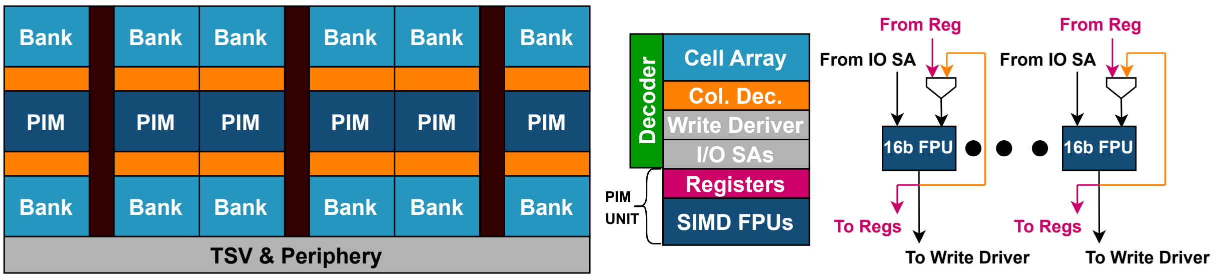


Figure 2.8: HBM DRAM die organization with PIM unit and its data path, presented in [33]

packed with a Single Instruction Multiple Data (SIMD) Floating-Point Unit (FPU) for parallel operations, as well as various register files for executing instructions (see figure 2.8).

During PIM operation, processing units distributed across memory banks execute common DRAM commands—such as load or store—initiated by the main processor. These units perform large-scale SIMD computations in parallel, ensuring seamless coordination with PIM instructions. The architecture's design divides the PIM execution unit into multiple pipeline stages, ensuring adherence to DRAM's internal timing for data operations. This approach involves initial instruction decoding, followed by data loading from the DRAM banks to the register files or the SIMD FPU, and then through multiple computational stages for executing multiply (MULT) or add (ADD) operations. Ultimately, the result is recorded in a general register file. This intricate design demonstrates an efficient, scalable approach to integrating PIM within existing computer systems, enhancing computational capabilities directly within memory components.

The authors in [113] scrutinize the practical implementation and broader adoption of PIM architectures across three domains. They focus on pinpointing real-world PIM opportunities and the programming complexities inherent to deploying PIM architectures. The authors identifies the challenges with respect to varying granularity in a variety of PIM kernels, the means to effectively allow PIM kernels to access vital virtual memory address translation mechanisms, different strategies for automatically identifying and assigning application segments to PIM.

Multiple studies have put forward techniques for near-memory processing to expedite Neural Network (NN) operations [111, 114, 115], and research by Huang et al. [116] is directed toward achieving energy-efficient graph processing through a cooperative hardware/software PIM design that leverages heterogeneity for better energy management.

2.2.2 PIM by Application Domain

Recent advancements in the realm of PIM have been directed towards improving computational efficiency in areas such as machine learning, AI, and NNs [32, 100, 103, 104, 106, 107, 111, 112, 117–120].

These advancements capitalize on the frequent use of Matrix-Vector Multiplication (MVM) in these domains by optimizing memory architecture for such operations, allowing for PIM and subsequent result consolidation. Alternatively, leveraging the logic layer within 3D-stacked memory chips enables the placement of computational units in proximity to DRAM cells, facilitating faster data processing. Notably, Imani,. et al [101] have shown how PIM can be applied to boost performance in graph and query processing, alongside machine learning enhancements. In a similar vein, authors in [116] have developed an innovative PIM architecture that merges memristor and CMOS technologies, catering to the varied needs of applications based on graph processing.

2.2.3 PIM by Memory Technology

Insights into the utilization of certain memory technologies within PIM frameworks reveal distinct preferences. Designs that integrate processing within the data array predominantly utilize advanced NVMs types like RRAM, and STT-RAM. The architecture of these memories aligns well with the requirements of such PIM implementations [100–102, 108, 116, 121, 122]. On the contrary, initiatives that involve integrating processing units adjacent to memory storage tend to opt for 3D-stacked memory solutions, such as HBM and HMC, to benefit from the extra logic layer they provide [33, 110, 111, 123–126].

Investigations into PIM enhancements employing conventional DRAM are less common, suggesting a focused area of research interest. A specialized segment of these studies directs attention towards UPMEM, a DRAM-based DIMM designed with built-in computing functions [35, 127, 128]. This attention underscores a trend towards marrying innovative memory technology with commercially viable products, offering a glimpse into the potential for PIM advancements to impact the broader technology market.

Analog-based processing using SRAM stands out as a promising direction in recent PIM studies, offering substantial promise for mitigating the challenges associated with the memory wall. This thesis proposes novel cross-layer PIM methodologies that incorporate STT-RAM from the NVM sector alongside SRAM, recognized for its conventional memory capabilities. Our initiatives aim to merge the unique advantages of these distinct memory technologies, presenting a unified strategy to navigate the complexities of the memory wall, underscoring the innovative integration of varied memory systems within PIM designs.

2.2.4 PIM by Evaluation Methodology

In the field of PIM, researchers utilize a range of methods for design and evaluation purposes, which can be categorized into analytical models, simulation models, and hardware implementations. Analytical models offer a theoretical basis and use mathematical calculations to estimate PIM system performance, eliminating the need for concrete prototypes. Simulation models, through software, emulate PIM architectures to facilitate comprehensive experimentation and refinement in a virtual setting, sidestepping the expenses and logistical demands of physical production. Conversely, hardware implementations entail the actual construction of PIM prototypes, enabling empirical testing and examination. This technique delivers critical data on the real-world applicability, efficiency, and obstacles associated with PIM technology.

1. **Analytical model:** A new analytical model [129], has been developed to design and analyze parallel algorithms that leverage PIM capabilities. This model combines the parallel processing strengths of CPU cores, which have efficient access to shared memory, with the advantages of PIM, where each processor core has local memory access. The model establishes universal metrics for assessing parallel complexity in both shared and distributed memory systems. The utility of this framework is demonstrated with a skip-list algorithm, tested across seven operations: retrieval, updating, deletion, finding predecessors and successors, upsertion, and range operations. This evaluation highlights the model's effectiveness in optimizing parallel algorithms for PIM environments, offering a structured method for examining complex processing tasks.
2. **Simulation model:** The absence of commercially viable PIM hardware has necessitated the use of simulators in advancing PIM systems research. Utilizing the GEM5 full-system simulator paired with DRAMSim2, Ghose et al. [113] evaluated an in-memory accelerator designed for efficient pointer chasing. This accelerator's unique feature is its in-DRAM address translation capability, which aims to enhance performance. DRAMSim2's accurate memory modeling and energy consumption analysis capabilities provide a comprehensive tool for researchers to investigate the potential benefits of PIM technology, enabling a closer look at the efficiency and energy optimizations possible with PIM solutions, even without access to physical hardware.

   In this work, Roy et al. [106] introduced a DRAM-based multiplication primitive for PIM to improve the efficiency of MVM operations essential for machine learning tasks. They evaluated its impact on DNN performance using HSPICE circuit simulations, demonstrating potential gains in DNN processing efficiency. This approach contributes valuable insights into the integration of DRAM-based PIM primitives within ML workflows, highlighting the promise of PIM in advancing ML applications. In their research, Zhou et al. [130] introduced an all-encompassing simulation platform designed to delve into the digital PIM design spectrum. The platform integrates a suite of software tools, a flexible compiler framework, and a simulation model for PIM architectures that is both quick and accurate. The authors report that their simulation tool significantly outperforms established benchmarks, providing simulations up to 10.3× faster with only a 6.3% variance in results. This efficiency and accuracy demonstrate the proposed simulator's effectiveness in facilitating the exploration and evaluation of digital PIM designs, underscoring its potential as a critical resource in the field.

In [131], the authors proposed a simulator called PIMSIM. It features a sophisticated front-end partitioner that effectively identifies and distributes instructions intended for PIM. A key feature of PIMSIM is its dynamic feedback system, which aids in determining the suitability of executing instructions in-memory. The simulator is adaptable, supporting three modes: fast simulation for quick checks, instrument-driven simulation for in-depth analysis, and full system simulation for exhaustive system-wide studies. These functionalities enable PIMSIM to stand out as a valuable tool in the development and testing of PIM-based computing solutions, facilitating a broad range of simulation needs.

The paper [132] propsed a simulator called NeuroSim. The NeuroSim simulator emerges as a specialized tool for simulating PIM hardware. It adeptly calculates crucial statistics such as area, latency, energy dynamics, and leakage power of the hardware it simulates. Validation of NeuroSim's performance comes from its comparison against actual data from a 16 kb CIM macro constructed using Taiwan Semiconductor Manufacturing Company (TSMC)'s 40 nm RRAM process at the 40 nm scale. Furthermore, the simulator's fidelity in replicating the performance of peripheral circuits, such as decoders, the switch matrix, multiplexers, and adders, has been verified through SPICE simulations. This level of validation highlights NeuroSim's effectiveness in accurately modeling the performance parameters of CIM hardware, making it a significant asset for evaluating CIM technology's practical applications.

The authors in [133] presenting a simulator called MPU-Sim, which serves as a simulation tool for evaluating general-purpose architectures designed for processing near memory banks. The simulator accurately represents a configuration where several Memory Processing Units (MPUs) are deployed inside a processor and linked through on-chip networks. Each of these MPUs features processing units located on the DRAM dies to facilitate proximity to memory storage. MPU-Sim's architecture is particularly designed to support the Single Instruction Multiple Threads (SIMT) programming model, aiming to maximize the utilization of bank-level parallelism while mitigating the impacts of control logic and communication bottlenecks. This capability underscores MPU-Sim's utility in advancing the development and understanding of near-bank processing solutions, addressing critical efficiency and parallelism challenges in contemporary computing systems.

MultiPim [134] and DAMOV [135] are developed atop the foundational simulators ZSim [136], a system simulator, and Ramulator [137], a memory simulator. These platforms are adept at facilitating the offloading of computational tasks to PIM units, exploring the potential efficiencies

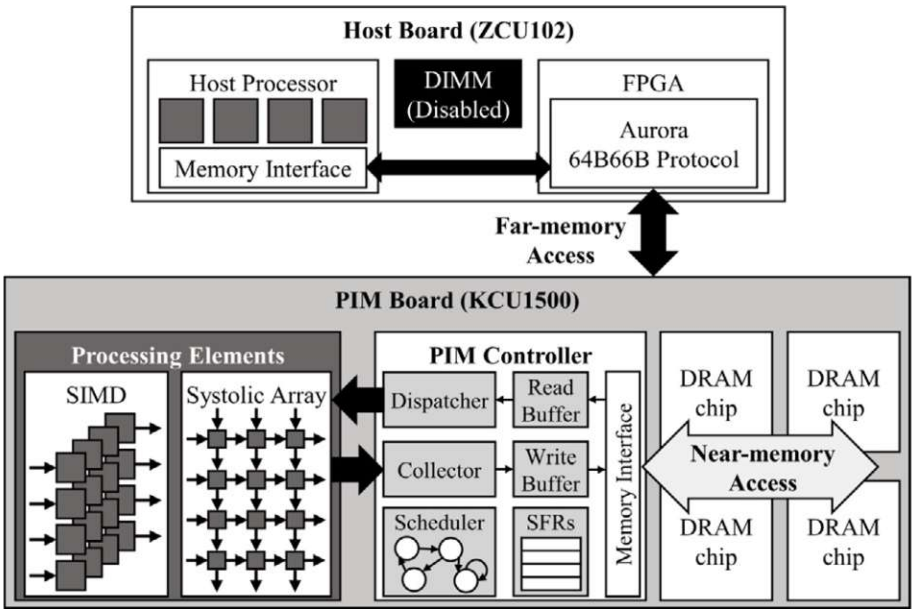


Figure 2.9: PIMCaffe system architecture by [34]

PIM can bring to computing. MultiPim sets itself apart by incorporating a multilayer interconnect system, employing crossbar switches, maintaining coherence among PIM cores, and integrating virtual memory systems. This configuration is aimed at optimizing the communication and data management across PIM units to boost system performance. In contrast, DAMOV specializes in conducting a thorough workload characterization to detect and analyze data-movement bottlenecks across a diverse set of applications. This analysis is crucial for understanding how PIM technologies can be strategically applied to alleviate such bottlenecks, potentially leading to more efficient computing solutions.

By focusing on different facets of PIM technology—MultiPim on the architectural and interconnect aspects, and DAMOV on workload and data movement analysis—both platforms make significant contributions to the research and development of PIM-enhanced computing environments.

3. **Hardware implementations:** In their research, Jaio et al. [120] introduced a chiplet-based design principle for PIM systems, tested through the deployment of a Tiny-Yolo DNN on a FPGA. The documentation lacks detailed information on the FPGA model and configuration used for testing. The core of their design strategy is a layer-wise partitioning method that disperses the tasks of a singular, large accelerator into a multi-chiplet pipeline. This methodology seeks to capitalize on the modular benefits of chiplet architectures, aiming for enhancements in system scalability and computational efficiency by distributing processing workloads across multiple units.

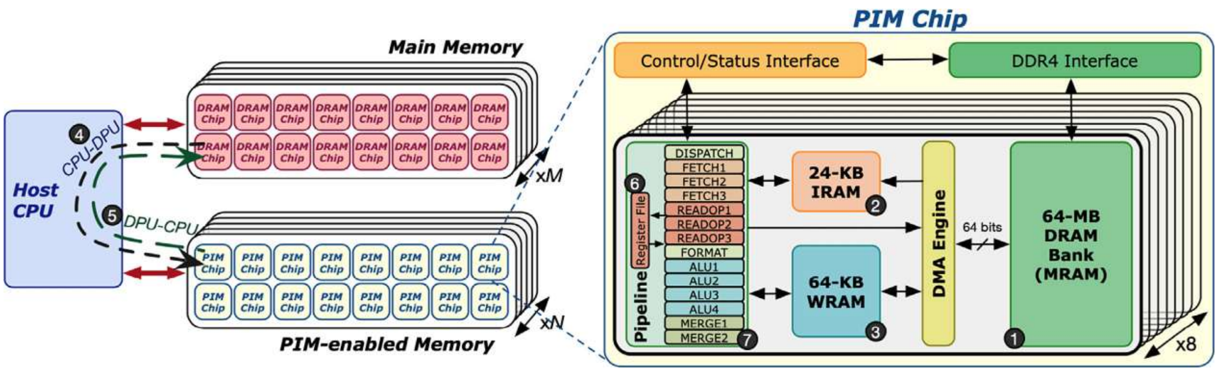


Figure 2.10: Architecture the proposed UPMEM-based PIM simulator [35]

PimCaffe [34] introduces an FPGA-based platform that simulates the functionality of PIM, equipped with SIMD and systolic array engines for computing. This setup enables effective vector and matrix operations within the PIM framework, demonstrating a significant leap towards efficient computational processing. The architecture and interconnectivity of these computing engines are detailed in a high-level block diagram, presented in Fig. 2.9. This visual representation is pivotal for grasping the innovative design and workflow of the PIM-emulation on the FPGA platform, emphasizing the role of SIMD and systolic arrays in optimizing computational efficiency. Samsung [33] has developed a PIM prototype leveraging a $20nm$ DRAM fabrication process, marking a significant step in industrial PIM technology. The core of this prototype is its execution unit, which boasts $16-bit$ SIMD lanes, each outfitted with both a floating-point adder and multiplier for advanced calculations. The architecture also includes a comprehensive set of registers: 32 entries for commands, 16 entries for general purposes, and 16 entries for scalar values. This elaborate setup demonstrates Samsung's commitment to enhancing computational efficiency through PIM, integrating powerful processing capabilities directly within memory components.

UPMEM has developed a commercially available PIM solution, as illustrated in Fig. 2.10. This solution integrates seamlessly with a system's host CPU and existing DRAM main memory, alongside PIM-augmented main memory. At the heart of this integration are the UPMEM modules, which are compatible with the DDR4 DIMM standard and include multiple PIM chips. Each PIM chip is equipped with 8 DRAM Processing Units (DPUs), with each DPU having its own $64MB$ DRAM bank, along with $24KB$ of instruction memory and $64KB$ of scratchpad memory for processing activities. Notably, these $64MB$ DRAM banks can be accessed by the host CPU, allowing for the straightforward transfer of input data and the retrieval of processed data, facilitating a streamlined computational workflow within the memory itself [35]. This approach is geared towards enhancing data processing speeds by reducing the latency associated with data communication between the CPU and memory.

Chapter 3

Design and Evaluation of Robust In-SRAM Computing Circuits

Publication Acknowledgment

Parts of this chapter were published in a similar form in the author's previous publications as follows:

- S. Seyedfaraji, B. Mesgari, and S. Rehman, "AID: Accuracy Improvement of Analog Discharge-Based in-SRAM Multiplication Accelerator," in Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE), 2022, pp. 873–878.
- S. Seyedfaraji, B. Mesgari, and S. Rehman, "SMART: Investigating the Impact of Threshold Voltage Suppression in an In-SRAM Multiplication/Accumulation Accelerator for Accuracy Improvement in 65 nm CMOS Technology," in Proceedings of the 25th Euromicro Conference on Digital System Design (DSD), 2022, pp. 821–826.
- S. Seyedfaraji, S. Jager, S. Shakibhamedan, A. Aftab, and S. Rehman, "OPTIMA: Design-Space Exploration of Discharge-Based In-SRAM Computing: Quantifying Energy-Accuracy Trade-offs," in Proceedings of the 61st ACM/IEEE Design Automation Conference (DAC), 2024, pp. 1–6.
- S. Seyedfaraji, S. Shakibhamedan, A. Seyedfaraji, B. Mesgari, N. TaheriNejad, A. Jantsch, and S. Rehman, "E-MAC: Enhanced In-SRAM MAC Accuracy via Digital-to-Time Modulation," in IEEE Journal on Exploratory Solid-State Computational Devices and Circuits, 2024.

Rapid advances in data-centric computing have spurred growing interest in processing-in-memory (PIM) architectures, particularly in-SRAM computing, for their potential to significantly reduce data movement costs and improve energy efficiency. However, designing robust in-SRAM circuits poses new challenges in accuracy, reliability, and scalability. This chapter addresses these challenges by first revisiting the fundamental operation principles of standard SRAM cells. We then demonstrate how conventional 6T-SRAM structures can be adapted to perform mixed-mode multiplication, exploring in detail the discharge behaviour of bit lines and the associated non-linearities.

Next, we present circuit-level techniques, such as threshold voltage suppression and novel charge-sharing approaches, to bolster robustness and reliability under process, voltage, and temperature (PVT) variations. We validate these methods using a 65nm CMOS technology node, comparing our proposed designs against state-of-the-art approaches in terms of energy efficiency, throughput, and MAC accuracy. Finally, we explore the design space of discharge-based in-SRAM computing to highlight optimal configurations for various performance-power trade-offs. By introducing comprehensive modelling and benchmarking frameworks alongside robust circuit design, this chapter lays a foundation for the reliable deployment of high-performance in-SRAM computing systems.

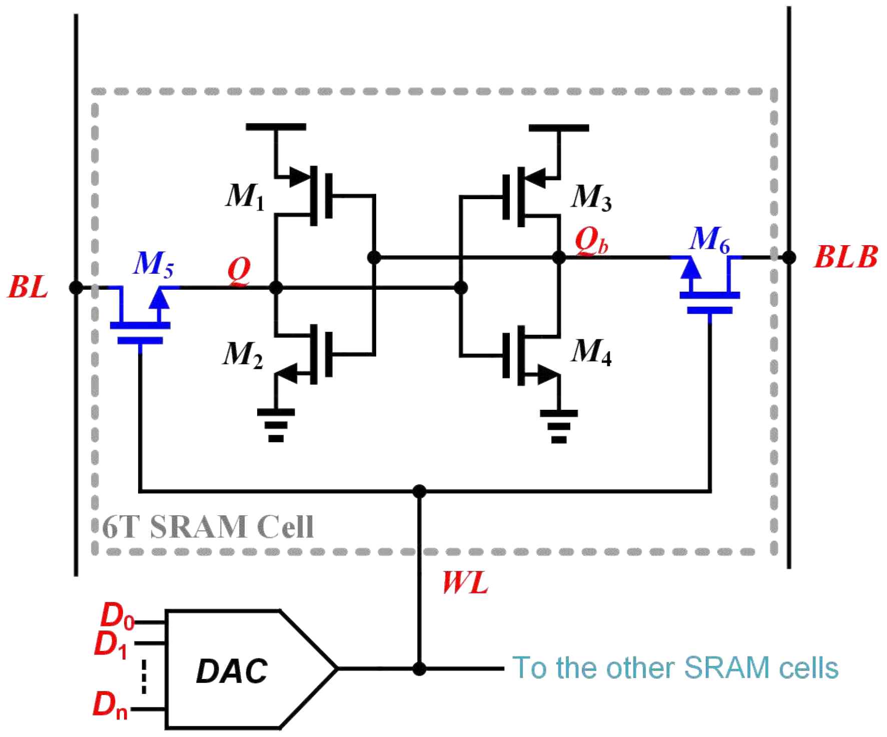


Figure 3.1: Standard single bit 6T-SRAM memory cell

3.1 SRAM Fundamental Operation Principles and Structures

Fig. 3.1 shows the standard 6T-SRAM, which is a basic unit of a static memory cell. To execute the typical read and write operations in a 6T-SRAM cell, certain circuit preparations are required. For a read operation, both bit-lines (BL, and Bit-Line-Bar (BLB)) are initially held high, meaning the gate voltage of the access transistors (M_5 and M_6) is discharged to the ground (with Word-Line (WL) nearly zero). Assuming Q is zero and Q_b is VDD, BL and BLB must be recharged to VDD, and WL is activated by applying VDD to the gates of M_5 and M_6. In this state, the gate-source voltage of M_6 is nearly zero (WL=VDD and Q_b=VDD), resulting in no significant current flow, keeping BLB and Q_b unchanged. In contrast, BL is pulled down through M_5 and M_6, reading the stored value (here zero, with Q=0) from the cell. Similarly, when VDD is stored, BL remains unchanged, while BLB is discharged to zero. Thus, for reading both 0 and VDD from the cell, BL and BLB are pre-charged to VDD initially. However, for a write operation, BL and BLB must be charged in different directions. Specifically, to write VDD into the cell, BL is pre-charged to VDD, BLB is pulled to zero, and WL is then raised.

In this chapter, we clarify using the read operation of a conventional SRAM cell to perform analog multiplication when Q=VDD and Q_b=0. We achieve the dot product by modulating the input digital data Din ($D_{n-1}, D_n, ..., D_1, D_0$) into the amplitude of the signal applied to the WL. We will discuss the details of in-SRAM analog multiplication next.

3.2 Quantitative Description of Mixed-Mode Multiplication Using a 6T-SRAM cell

To implement PIM within the 6T-SRAM structure, the result of the mathematical operation is encoded based on the discharge behavior (voltage drop) of the BLB of the SRAM cell. The voltage produced by each BLB must be weighted and combined with other BLBs according to the digital input weight (e.g., the Least Significant Bit (LSB) with a weight of one and the Most Significant Bit (MSB) with a weight of $2^{(n-1)}$) [22].

A common approach to modulate digital input into its analog counterpart involves using the amplitude and Pulse Width (PW) of the access transistor. The modulated data is applied to the SRAM cell via the gate of the access transistor, using varying amplitude and PW or a combination of both techniques [22, 36, 37]. Varying the amplitude at the gate of the access transistor can introduce non-linearities in the BLB discharge behavior, leading to an increase in Bit Error Rate (BER) of the sampled data. To address this issue, the authors in [22] apply the digital input to the access transistor using a semilinear function of the digital codes. This results in non-uniformly spaced analog voltages for BLB

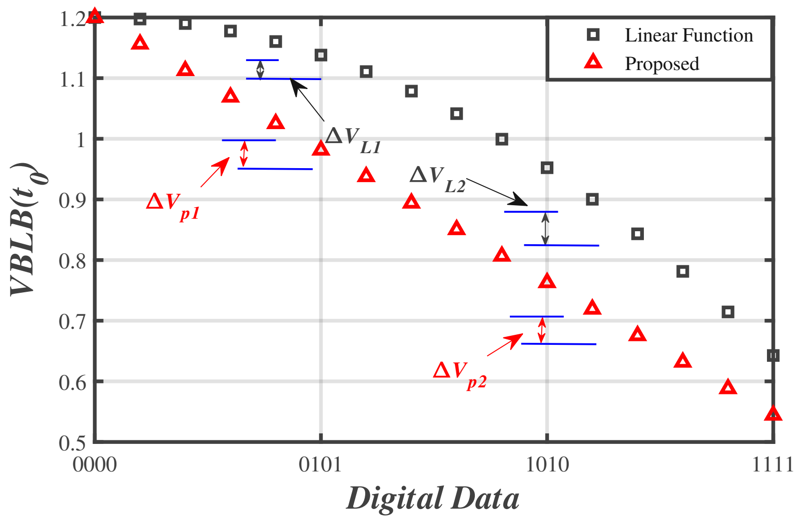


Figure 3.2: Non-linear behavior of BLB voltage and effect of proposed linear root function technique

discharge behavior, as illustrated in Fig. 3.2. Due to the nonlinear nature of the access transistor, the voltage drops with unequal differences, causing a high BER. Fig. 3.2 shows the digital number versus the interpolated analog counterpart at the BLB port, based on the linear function at the gate of the access transistor. For example, for a 4-bit number, interpolated analog values from 0000 to 0101 are not sufficiently spaced, meaning the ADC cannot distinguish between these six (zero to five) numbers. In Fig. 3.2, ΔV_{L1} is smaller than ΔV_{L2}; hence, the probability of accurately converting the expected multiplication value to the correct digital number is significantly lower for ΔV_{L1} compared to ΔV_{L2}. Although [22] employs a binary-weighted charge-sharing technique to enhance computation accuracy, the required bit margin is still insufficiently spaced, and a non-zero static current due to the pre-charge circuit is present. This static current becomes significant when the memory array size is relatively large.

In the read operation, instead of reading specific digital data, a product between two multi-bit words can be modulated in the discharge behavior of BLB to perform multiplication. To gain a deeper understanding of the BLB discharge behavior, we conducted an extensive analysis to see how the amplitude of WL affects the output BER. In the first phase of multiplication, WL is discharged to the ground (disable mode), and we write VDD into the SRAM cell, meaning $Q = VDD$ and $Q_b = 0$. In the second step, BL and BLB are pre-charged to VDD for a read operation. Fig. 3.3 shows the preparation phase for analog multiplication. When $Q = VDD$ and $Q_b = 0$, M_4 and M_1 are in the deep triode region (with low channel resistance), and M_3 and M_2 are in the cut-off region. Once WL is enabled, significant current can only flow through M_6 and M_5. Therefore, M_6 creates a path for BLB to discharge the initial voltage of C_{blb} from VDD to the ground ($0V$). Based on this discussion, the rightmost transistor represents an equivalent circuit to quantify the current flow I_{BLB}. By applying KCL at the drain node of this transistor, the difference equation for the BLB discharge behavior can be derived as shown in equation 3.1.

$$I_{BLB} + C_{blb}\frac{dV_{BLB}(t)}{dt} = 0 \tag{3.1}$$

For the Metal Oxide Semiconductor Field Effect Transistor (MOSFET) current equation, neglecting channel-length modulation, the drain-source current of M_6, denoted as I_0, is expressed in equation 3.2.

$$I_0 = \frac{1}{2}\mu_n C_{ox}\left(\frac{W}{L}\right)(V_{GS_{M6}} - V_{TH})^2 \tag{3.2}$$

In this equation, we assume the transistor stays in the saturation region during the BLB discharge. Here, μ_n is the electron mobility, C_{ox} is the gate-oxide capacitance, and W and L are the channel width and length of the MOSFET, respectively. V_{TH} is the threshold voltage, the minimum gate-to-source voltage required to create a nonzero current between the source and drain. To account for the channel-length modulation parameter λ, the current equation of M_6 is multiplied by $(1 + \lambda V_{BLB})$ in equation 3.2. Combining equations 3.1 and 3.2 and integrating both sides results in equation 3.3.

$$-\int_{t=0}^{t}\frac{I_0}{C_{blb}}d\tau = \int_{V_0=VDD}^{V_{BLB}(t)} dV_{BLB}(\tau) \tag{3.3}$$

Neglecting channel-length modulation and assuming V_{WL} equals $V_{GS_{M6}}$ (see Fig. 3.3), the BLB discharge behavior is given by equation 3.4. To consider the influence of λ, we solve the differential equation 3.1 by multiplying I_0 by $(1 + \lambda V_{BLB})$. Solving this and finding $V_{BLB}(t)$ reveals the effect of channel-length modulation, expressed in a nonlinear discharging manner in equation 3.5. Equations 3.4 and 3.5 show that applying an analog voltage with different amplitude and PW to the gate of M_6 can generate a variety of currents through the transistor, resulting in different voltage drops at the BLB terminal.

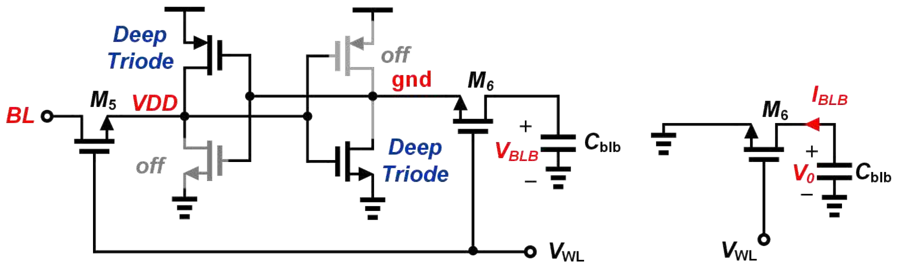
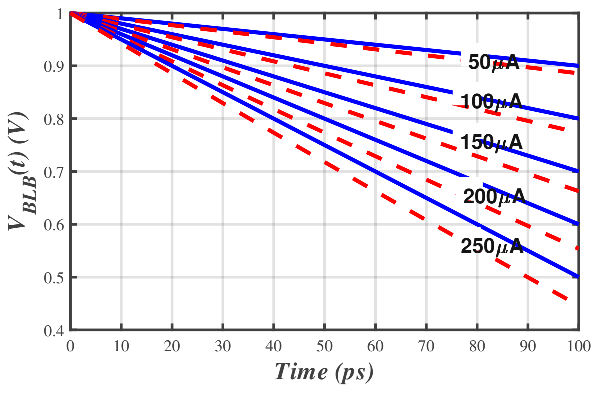
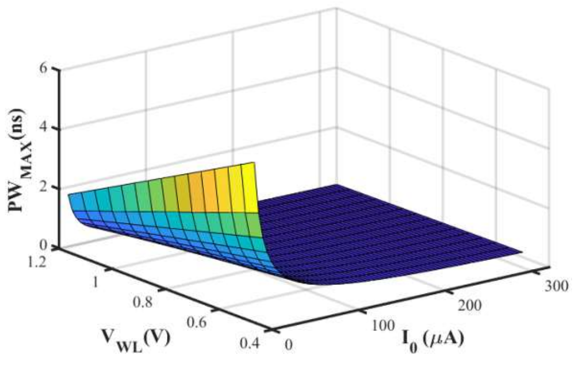


Figure 3.3: Equivalent circuit diagram of discharge behavior of BLB



(a) $V_{BLB}(t)$ versus time for different values of I_0 based on Eq. 3.4 and 3.5



(b) Maximum sampling time of V_{blb} while M_6 stays in active region versus I_0 and V_{WL}

Figure 3.4: discharge behavior of BLB (a), and 3D representation of PW_{max}, I_0, and V_{wl} (b)

$$V_{BLB}(t) = VDD - \left(\frac{\mu_n C_{ox}\left(\frac{W}{L}\right)(V_{WL} - V_{TH})^2 t}{2C_{blb}}\right) - \frac{I_0 t}{C_{blb}} = \frac{1}{\lambda}\ln\left(\frac{V_{BLB}(t) + \frac{1}{\lambda}}{VDD + \frac{1}{\lambda}}\right) \tag{3.4}$$

$$V_{BLB}(t) = \left(VDD + \frac{1}{\lambda}\right) e^{-\left(\frac{\lambda I_0}{C_{blb}}\right)t} - \frac{1}{\lambda} \tag{3.5}$$

Fig. 3.4a illustrates the discharge behavior of BLB using equations 3.4 and 3.5. For this simulation, we assume $V_{DD} = 1V$, $C_{blb} = 50\,\text{fF}$, and $\lambda = 0.15\,\text{V}^{-1}$. A multi-bit input can be mapped as a discharge voltage on the BLB within a specified time (PW) and a different value for I_0 (Amplitude). It's important to note that changing V_{WL} using a linear function cannot produce a linear relationship between the digital input data and the resultant BLB voltage, potentially causing significant errors in the multiplication result. As mentioned earlier, equation 3.2 is valid when M_6 remains in the saturation region. Therefore, during the discharge time of BLB, it is crucial to ensure that M_6 does not enter the triode region; otherwise, the BLB discharge rate will slow down. An accurate sampling of the BLB voltage just before M_6 enters the triode region can prevent systematic errors. This accurate sampling can be achieved by controlling the pulse width of V_{WL}. As a result, based on equation 3.4 and the saturation condition of an N-channel Metal-Oxide-Semiconductor (NMOS) transistor, the maximum pulse width PW_{Max} is calculated using equation 3.6.

$$V_{D_{M6}} \geq V_{D_{M6}} - V_{TH} \implies PW_{Max} \leq \frac{C_{BLB}}{I_0}(V_{DD} + V_{TH} - V_{WL}) \tag{3.6}$$

Using the design parameters mentioned earlier for Fig. 3.4a, a 3D representation of equation 3.6 is shown in Fig. 3.4b. This representation helps in accurately predicting the maximum PW_{Max} for the

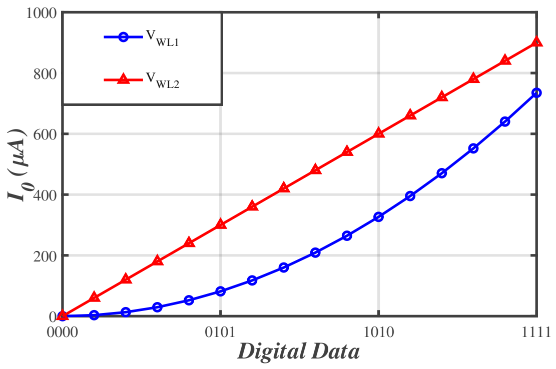


Figure 3.5: I_0 versus digital data using equations 3.7 and 3.8

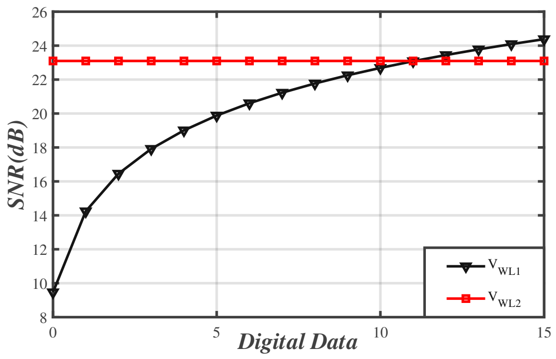


Figure 3.6: Signal to Noise Ratio (SNR) simulation based on Eq. 3.7 and, 3.8

voltage drop of the BLB for different currents and V_{WL} voltages while the access transistor remains in the active region. This curve indicates that a higher current through the access transistor requires less sampling time.

3.3 Mapping the Digital Data to the Analog Voltage of V_{WL} and Linearity Issue

For an $N \times N$ multiplication, where N is an arithmetic number ($N \in \mathbb{W}$), we need to generate a linearly spaced transistor current of $(0.5 \times N \times (N-1)+1)$ by controlling V_{WL}, thus controlling the discharge rate of BLB, while ignoring repetitive instances. For instance, in [22], a linear function is used to generate the WL voltage V_{WL1}, as formulated in equation 3.7. However, using 3.7 as V_{WL1} does not produce a linear current through M_6, due to the quadratic behavior of $V_{Digital-to-AnalogConverter(DAC)}$ affecting V_{BLB}. To address this, we propose using a root function for V_{WL2}, shown in equation 3.8. V_{DAC} is the controlling voltage for V_{WL}, typically managed by a DAC.

$$V_{WL1} = V_{TH} + V_{DAC} \times \frac{(V_{DD} - V_{TH})}{2^N - 1} \tag{3.7}$$

$$V_{WL2} = V_{TH} + \sqrt{V_{DAC} \times \frac{(V_{DD} - V_{TH})}{2^N - 1}} \tag{3.8}$$

Considering the equations 3.2, 3.7, and 3.8, I_0 is plotted for two different cases (V_{WL1} and V_{WL2}) against digital data in Fig. 3.5 This Fig. shows that by applying V_{WL2} to the gate of M_6, I_0 achieves linear proportionality with the digital input data, whereas V_{WL1} results in a non-linear relationship.

To evaluate the accuracy of the analog multiplier by calculating the SNR at the BLB port. As discussed earlier, using a linear function at the gate of the access transistor can lead to BER degradation.

To quantify this degradation, we use SNR, which can be expressed in terms of circuit parameters. There is an inverse relationship between SNR and BER, so an increase in BER results in a decrease in SNR and vice versa. We consider two different scenarios based on Fig. 3.4a For SNR calculation, we use equation 3.9, where P_{noise} is approximated by assuming the integrated noise variance of a parallel RC network, given by $\delta^2 = \frac{KT}{C}$. Here, K is the Boltzmann constant, and T is the absolute temperature in Kelvin.

$$SNR = 10 \log_{10}\left(\frac{P_{Signal}}{P_{noise}}\right) \tag{3.9}$$

P_{signal} is calculated based on two successive BLB voltages due to the change in the DAC digital input (V_{DACi} and V_{DACi+1}). By substituting V_{WL1} (equation 3.7) and V_{WL2} (equation 3.8) into equation 3.4 and simplifying, the voltage differences for both cases are obtained using equations 3.10 and 3.11 respectively. t_0 is the sampling time of the BLB voltage, which can be calculated using Fig. 3.4b and equation 3.6.

$$\Delta V_{BLB_{V_{WL1}}} = \frac{\beta t_0}{C_{blb}}\left(\frac{(V_{DD} - V_{TH})}{2^N - 1}\right)^2 (V_{DACi}^2 - V_{DACi+1}^2) \tag{3.10}$$

$$\Delta V_{BLB_{V_{WL2}}} = \frac{\beta t_0}{C_{blb}}\left(\frac{(V_{DD} - V_{TH})}{2^N - 1}\right) (V_{DACi} - V_{DACi+1}) \tag{3.11}$$

By dividing ΔV_{BLB}^2 by δ^2 and using equation 3.9, the SNR of the multiplication can be obtained as a function of the input digital data. To demonstrate the effectiveness of our proposed method of using a nonlinear function at the WL port, we conducted a simulation based on equations 3.9,3.10, and 3.11, assuming $V_{DD} = 1V$, $C_{BLB} = 50\,\text{fF}$, and $t_0 = 50\,\text{ps}$, as shown in Fig. 3.6. Using V_{WL1}, which is a linear function of the input digital data, results in a non-uniformly spaced analog voltage for BLB discharge, as illustrated in Fig. 3.4a. However, with our proposed method, forcing the BLB voltage to follow a linear trend by using a root function voltage for the gate of the access transistor, results in a relative accuracy improvement of up to $10.77dB$ average SNR.

3.4 Multi-Bit Calculation In 6T-SRAM

We utilize the standard 6T-SRAM cell as a single-bit multiplication unit, as shown in Fig. 3.1. Suppose we aim to perform a multiplication of a 4-bit D_{in} and 4-bit stored data in SRAM cells, denoted as J_s ($J_3 ... J_0$). If the stored data requires an 8-bit resolution, we must use eight single-bit SRAM cells arranged in an array. For D_{in}, the required resolution is encoded using the amplitude of V_{WL}. Fig. 8.a illustrates the architecture of the proposed in-memory 4-bit D_{in} multiplied by 4-bit J_s multiplier. It consists of four standard 6T-bit cells and read/write circuitry for two operating modes. In SRAM mode, the operation is a standard read/write of digital data, but in multiplication mode, a sampled product of analog multiplication is generated. Initially, as discussed earlier, J_s is stored in the 4-bit array during T_{WEN}, with BL and BLB reaching V_{DD} for the multiplication operation, immediately after the data storing period called T_{pre} (see Fig. 3.7.b). If "0" is stored, there will be no significant current flow, so BLB remains unchanged and stays near V_{DD}. The 4-bit J_s is stored in four unit cells of a 6T-SRAM column, as highlighted in Fig. 3.7.a, with the MSB on the left and the LSB on the right. To perform accurate analog multiplication considering bit significance for D_{in}, we use two techniques: a nonlinear DAC and a charge sharing circuit, both highlighted in green in Fig. 3.7.a. Our charge sharing method does not require static current, unlike other state-of-the-art methods, particularly [22], which uses a pre-charge circuit with PW control, incurring the cost of static current. As shown in Fig. 8, the PW of each path is controlled using two complementary switches; for example, when S_3 is connected, S_{3b} is open. This allows the PW of BLB to be adjusted based on bit significance. Consequently, the current flow time (T_0) is assigned to the MSB, while the LSB has a higher discharging time ($8 \times T_0$). It is worth noting that, based on equations 3.4 and 3.8, our weighted charge sharing method can lead to a proportional discharge behavior of BLB, essential for achieving high accuracy in multiplication results. In addition to the time coding of input data provided by the proposed charge-sharing circuit, to achieve a higher resolution of the BLB discharging rate, the amplitude of V_{WLs} is selected based on the bit significance of D_{in} as described in the previous section. Consequently, the amplitudes of the word-line

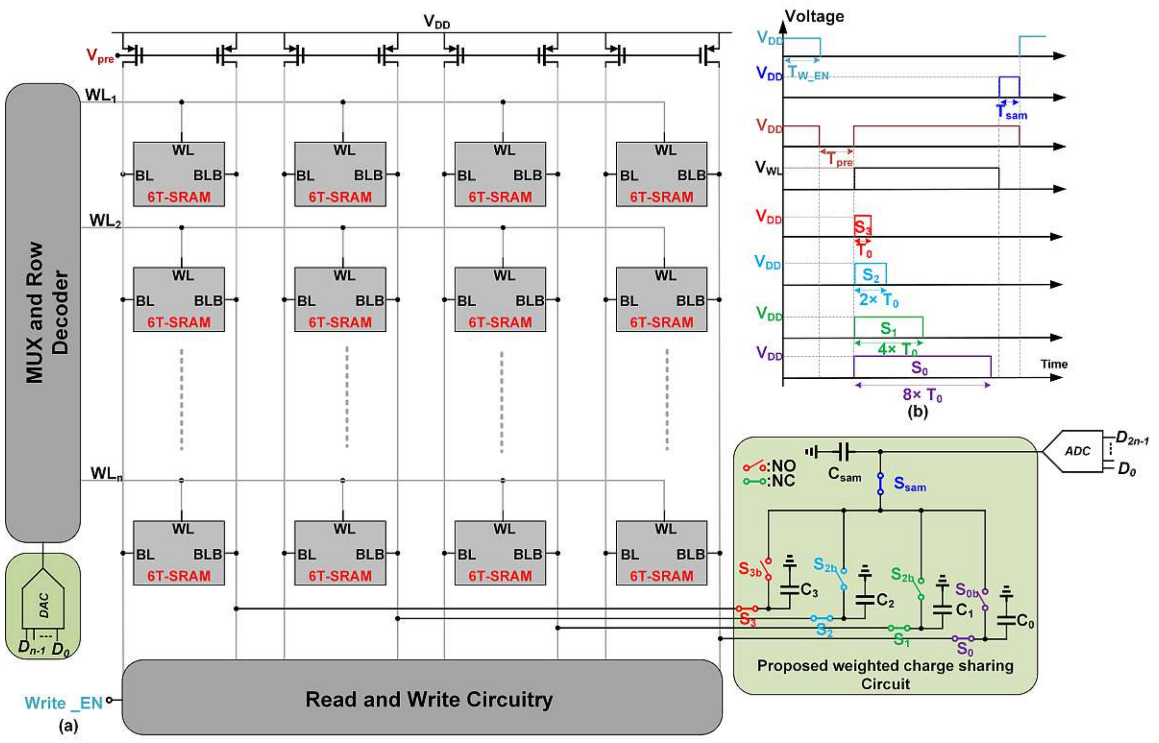


Figure 3.7: In-Memory multiplication operation 4-bit D_in multiply by 4-bit J_s digital data

Table 3.1: AID vs. state-of-the-art techniques

	[2]	[37]	[36]	[22]	[38]	[138]
Technology Size $[nm]$	65	65	180	65	65	65
Supply Voltage $[V]$	1.0	1.2	1.8	1.2	1.0	0.925
Bit-Width	4	5	5	4	8	4
MAC Energy $[pJ]$	0.523	3.5	1.167	0.9	1.3	0.32
Accuracy$[STD.V]$	0.086	N.A.	N.A.	0.6	N.A.	N.A.
Frequency$[MHz]$	200	2.5	N.A.	100	60-125	N.A.

terminals should be chosen such that $V_{WL0} > V_{WL1} > V_{WL2} > V_{WL3} > V_{TH}$.

To provide stable BLB data for the ADC at the end of the $(T_{WEN} + T_{pre} + 8 \times T_{on} + T_{sam}) = T_{MU}$ time frame, a sample and hold circuit is needed, as shown in Fig. 3.7.a. When S_{sam} is opened, the V_{BLBs} will be sampled and delivered to the ADC.

3.5 Experimental Setup in 65nm CMOS Technology

This circuit, depicted in Fig. 3.7.a, has been implemented at the circuit level using $65nm$ CMOS technology with a supply voltage of $1V$. The circuit parameters, such as the size of the transistors, WL voltages, and PW of the charge-sharing circuit, are optimized using SPECTRE transient simulation and equations 3.1- 3.11. This optimization aims to minimize T_{MU}, power consumption, and occupied area, thereby enhancing the accuracy of multiplication. Fig. 3.8 shows the discharge behavior of BLB versus time for different values of V_{WL}. The simulated result confirms that our mathematical analysis is quite precise, especially in the linear region. Consequently, sampling V_{BLB} at a specific time corresponds to the input digital value. Assuming a 4-bit D_{in}, $V_{WL} = 0.6$V can be interpreted as a binary value of "1111", while 1V corresponds to "0000".

Random variations in the employed transistors, such as threshold voltage, gate oxide thickness, and mobility variations, are significant issues that degrade the performance of analog multiplication circuits. Therefore, in this thesis, a 1000-point Monte Carlo (MC) simulation has been conducted, considering process variations and mismatches to characterize the accuracy of 4-bit D_{in} multiplied by 4-bit J_s. As mentioned, the 4-bit D_{in} is encoded into the amplitude of the WL while J_s is directly stored in each 6T-SRAM cell. The worst-case standard deviation is less than 0.086 for the case of (15×15), as illustrated in Fig. 3.9.

To compare the performance improvement of this work with other topologies, table 3.1 is prepared that includes the MAC energy and standard deviation of recent methods.

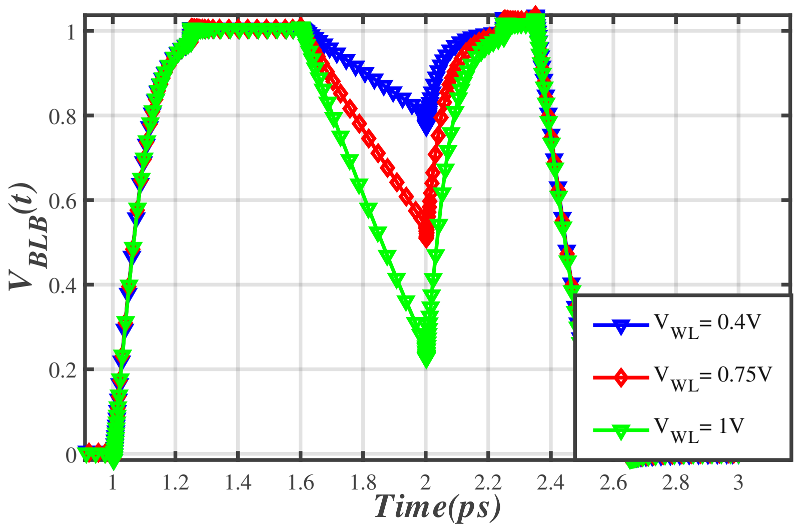


Figure 3.8: Simulated V_{BLB}(t) vs. time for different V_{WL}

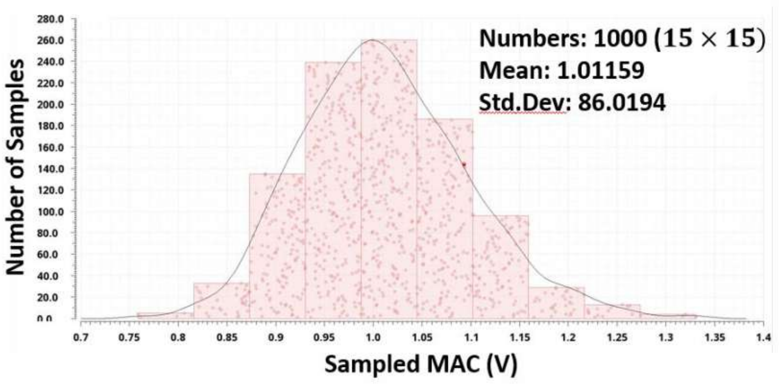


Figure 3.9: The worst-case standard deviation of our proposed analog multiplication

3.6 Investigating the Impact of Threshold Voltage Suppression in an In-SRAM MAC

In this section, we examined how lowering the threshold voltage of access transistors in the In-SRAM MAC accelerator affects performance. Specifically, we focused on improving the discharge rate of the BLB, which enhances the accuracy of the MAC operation.

As discussed in the previous section, the dynamic behavior of current through the access transistor(i.e., M_6 in Fig. 3.1) defines the accuracy of analog MAC operations. The available voltage range for the WL determines the output current accuracy and the maximum number of coded bits, typically limited by the transistor's threshold voltage (V_{th}). In [22] and [2], a 350 mV margin was chosen to prevent nonlinear behavior in the BLB voltage drop for the M_6 transistor implemented in 65nm CMOS technology. According to [22], if the power supply is equal to VDD and V_{th} is near zero, the accuracy of generating coded analog data can ideally be improved by $(V_{th}/(VDD - V_{th}) \times 100)$ percent. Additionally, controlling the voltage of the bulk pin of M_6 can suppress the transistor's body effect. This method requires using a deep-nwell transistor for M_6, which is feasible in a standard CMOS process.

Considering Eqs. 3.1 and 3.2, and the channel length modulation represented by the parameter λ in the current equation of M_6, Eq. 3.2 should be multiplied by a factor $(1 + \lambda V_{BLB})$. To calculate V_{BLB}, we combine Eqs. 3.1 and 3.2 and integrate both sides, resulting in Eq. 3.12 based on the BLB voltage drop:

$$V_{BLB} = V_{DD} - \frac{\mu_n C_{OX}(W/L)(V_{WL} - V_{TH})^2 t}{2C_{BLB}} \tag{3.12}$$

It is important to note that Eq. 3.12 is accurate while M_6 remains in its saturation region. However, in reality, current leaks through M_6 and M_4 (see Fig. 3.1). Therefore, sampling the discharge of the BLB should be performed in the same region (saturation). If the access transistor enters the triode region before the sampling time, it will cause a systematic fault in the output, making the data invalid. To ensure accurate sampling, a maximum WL PW($WLPW_{MAX}$) can be calculated based on the saturation condition of an NMOS transistor, leading to Eq. 3.13:

$$WL_{PW_{MAX}} = \frac{C_{BLB}}{I_0}(V_{DD} + V_{TH} - V_{WL}) \tag{3.13}$$

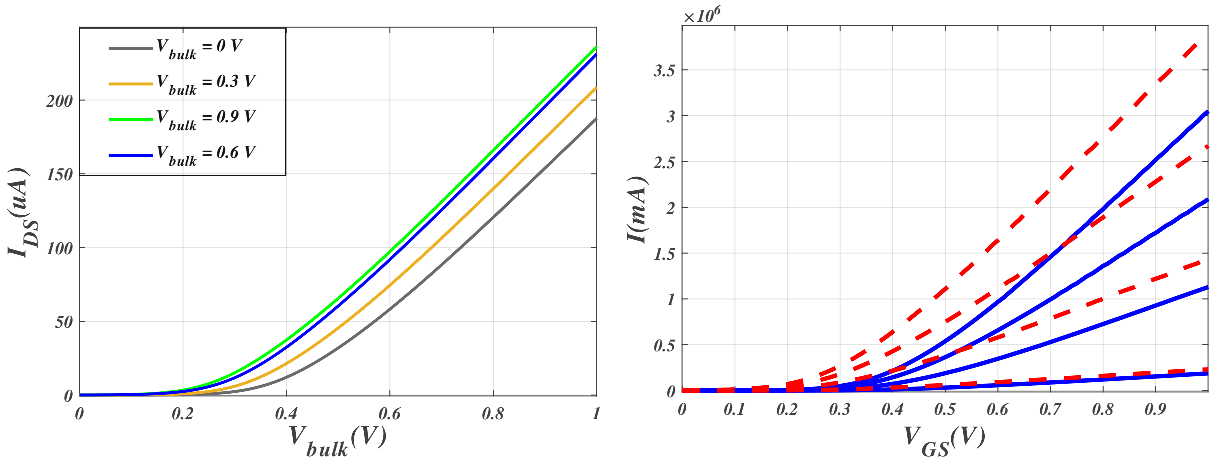
Before discussing the effect of V_{TH} on the output of the MAC operation, V_{WL} needs to be determined. For $N \times N$ multiplication (where N is the bit width of the input data and ranges from 0 to 2^{N-1}), the WL of the circuit requires 2^N identical voltage levels to represent each input data. By applying the KVL on the circuit, V_{WL} can be derived as follows:

$$V_{WL} = V_{TH} + \left(V_{DAC} + \frac{V_{DD} - V_{TH}}{2^N - 1}\right) \tag{3.14}$$

Considering both Eqs. 3.13 and 3.14, it is evident that increasing V_{TH} results in a longer PW for $WLPW_{MAX}$ and a greater margin for the WL itself. Therefore, V_{TH} is a crucial design parameter for improving the read margin. Providing a larger margin for the WL enhances the accuracy of the In-SRAM MAC accelerator, which will be discussed in detail in the following sections.

3.6.1 Body Effect of the MOSFET

In most designs, including the In-SRAM MAC accelerator shown in Fig. 3.1, it is typically assumed that the transistor's bulk and source are connected to the ground. This results in both $M1$ and $M4$ remaining in the triode region. However, when $M4$ is in the deep triode region, its drain-source voltage is not completely zero, leading to a higher V_{TH} for $M6$. As a result, according to Eq. 3.12, the discharge rate of the access transistor slows down. Therefore, this assumption is not entirely accurate because there



(a) Body biasing of the access transistor for different V_{bulk}

(b) Sweeping the transistors' width with V_{bulk} = 0 (blue) and V_{bulk} = 0.6 V (red dashed)

Figure 3.10: Observation of the V_{bulk} and the width of the transistor on V_{TH}

is a slight potential difference between the source and bulk of the transistors.

When the bulk voltage of an NMOS drops below the source voltage, as long as the source and drain connections remain in reverse-bias mode, the device continues to operate correctly, though its characteristics are modified. To understand these modifications, suppose $V_S = V_D = 0$, and V_G is slightly less than V_{TH}, forming a depletion region under the gate but no inversion layer. As V_B decreases towards a negative potential, more holes are attracted to the substrate connection, creating a larger negative bias under the depletion region. The threshold voltage depends on the total charge in the depletion region, and as the gate charge must balance Q_d before an inversion layer forms, a decrease in V_B and an increase in Q_d will raise V_{TH}. This effect is known as the "body effect" or the "back-gate effect." The threshold voltage V_{TH} can be determined using the following Eq. 3.15 extracted from [139], and is called the body-effect coefficient.

$$\gamma = \sqrt{(2q\epsilon_{si}N_{sub})/C_{OX}} \tag{3.15}$$

Therefore, to decrease V_{TH}, the only option is to make $\left(\sqrt{2\phi_F + V_{SB}} - \sqrt{|2\phi_F|}\right)$ negative. This means that $2\phi_F = -V_{SB}$. Fig. 3.10a shows that by reducing V_{TH}, we enable current to pass through the cell more quickly. By using body biasing, a 125(mV) reduction in V_{TH} (for a 0.6 V body voltage) can be achieved as intended (see Fig. 3.10a). In Fig. 3.10b, the plot shows the effect of varying the width of the access transistor. As depicted, for $V_{Bulk} = 0.6$ (i.e., low V_{TH}), the current passing through the cell increases (red dashed line) regardless of the transistor's width.

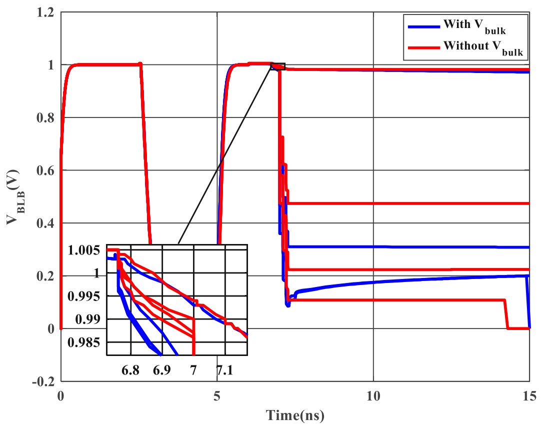
3.6.2 Experimental Setup and Comparison

To demonstrate the effect of V_{TH} modification on the result of the MAC operation, we first explain the operation process. One operand is stored in the SRAM cell, while the other operand is passed through a DAC to produce an analog representation of the digital data. State-of-the-art approaches [2, 22] use different DAC implementations; however, the available voltage for each input number can be calculated using Eq. 3.7 and Eq. 3.8 to operate in the saturation region.

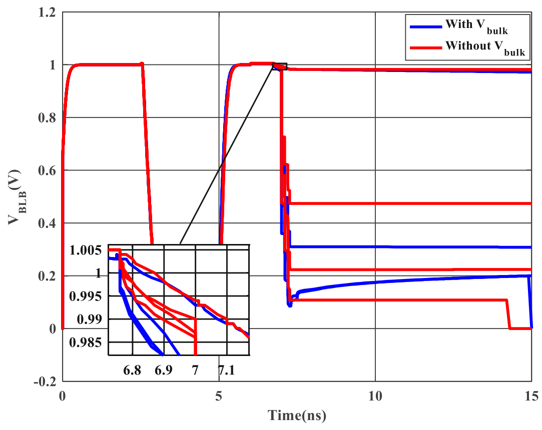
In both equations, N is the bit-width of the input data, so for a four-bit number, it would be equal to fifteen absolute consecutive linear cases. Fig. 3.11 show the simulation results of the latest state-of-the-art approaches [2, 22]. Both approaches demonstrate that body biasing accelerates the discharge of V_{BLB}, reducing the latency of the operation.

We based our architecture on the fundamental 6T-SRAM block, which can perform regular read/write operations in memory mode or one-by-one-bit multiplication in mathematical mode. For multiplication, the write circuit stores one operand in the memory cells. Then, the memory switches to mathematical mode, and the second operand is passed to the DAC to be encoded into an analog representation of V_{WL}. Our technique increases the available WL margin from $[300 - 700]$ mV in state-of-the-art approaches to $[175 - 700]$ mV in this part of our study. This additional margin can be used for an extra bit of width or to reduce BER. For comparison with [22] and [2], we maintained a four-by-four-bit multiplication and focused on BER reduction, which will be discussed shortly. Fig. 3.12 shows our architecture, with the green highlighted part indicating our novel contribution. The gates of the access transistors (i.e., M_5 and M_6 in Fig. 3.1) are connected to 0.6 V using a dual-VDD design technique. In this design, the MSBs are stored in the leftmost cell, and the LSBs are stored in the right cells. We used the circuitry design from [2], ensuring no static current is applied by the pre-charge circuit, resulting in no additional power overhead. Although the WL timing is the same as in [22] and [2] for a fair comparison, we believe there is potential for further optimization of the WL PW.

The architecture introduced in the previous section (see. Fig. 3.7) was implemented using Cadence Virtuoso 6.1.8 with 65nm CMOS technology. The power supply was set to 1 V, and the gate voltages were selected as 0.6 V. We optimized the circuit parameters, including transistor sizing and V_{WL}, via SPECTRE ADE-XL transient simulation considering Eqs. 3.12 - 3.15. The optimization aimed to improve MAC latency, circuit power, and cell area. Notably, there is no area overhead compared to state-of-the-art designs. Since physical design parameters (e.g., V_{TH}, oxide capacitance, and electron mobility) could affect the output, a 1000-point MC simulation (considering process and mismatch) was conducted, with results shown in Figs. 3.13a and 3.13b. Table 3.2 compares SMART and state-of-the-art methods regarding energy per computation, frequency, and accuracy, calculated based on the proposed SNR



(a)



(b)

Figure 3.11: Body biasing effect on discharge of VBLB for a: [2] and for b: [22]

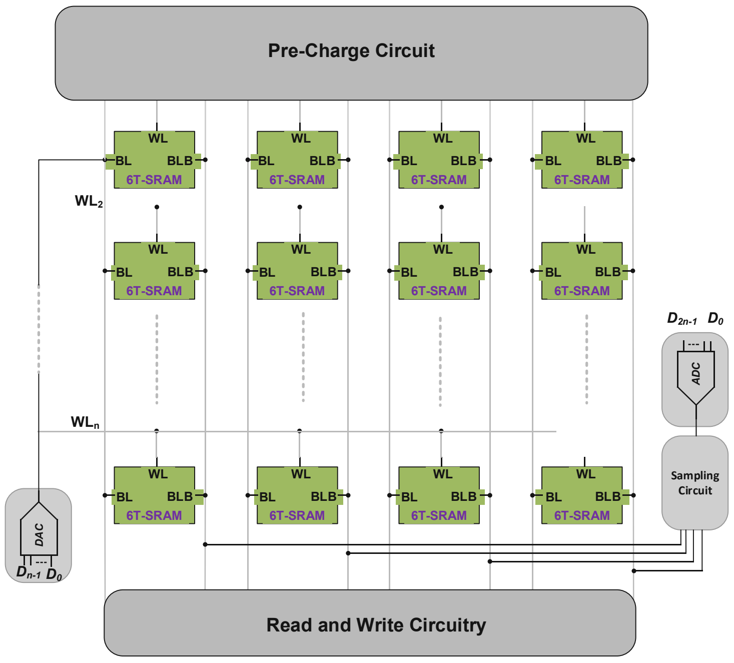
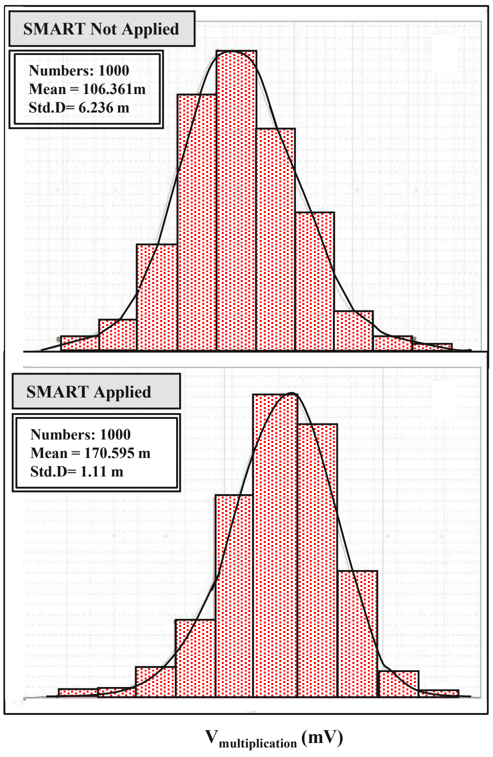


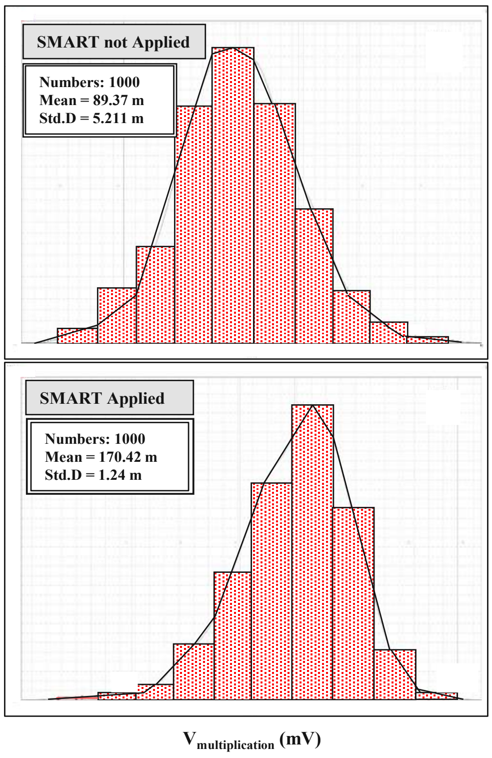
Figure 3.12: The architecture of SMART, the body biasing is applied in the access transistor

Table 3.2: SMART vs. state-of-the-art techniques

	[6]	[22]	[2]	[38]	[37]
Technology Size $[nm]$	65	65	65	65	65
Supply Voltage $[V]$	1.0	1.2	1	1	1.2
MAC Energy $[pJ]$	0.783	0.9	0.523	0.9	3.5
Accuracy$[STD.V]$	0.009	0.6	0.086	0.6	N.A.
Frequnecy$MHz]$	250	100	200	60-125	2.5

in [2]. As seen, both accuracy and frequency have been improved.





(a) Accuracy improvement for 1111 × 1111 in [2] exploiting SMART approach

(b) Accuracy improvement for 1111 × 1111 in [22] exploiting SMART approach

Figure 3.13: The effect of SMART on state-of-the-art techniques

3.7 Design-Space Exploration of Discharge-Based In-SRAM Computing

Discharge-based in-SRAM computing circuits are usually optimized for factors such as energy consumption, latency, accuracy, or area, resulting in a diverse design space. Fig. 3.14 compares various design points for PIM multipliers [2, 22, 37, 38]. It is evident that there are considerable trade-offs between these design metrics. For example, the circuit discussed in [38] features higher bit widths, leading to greater accuracy, while other circuits exhibit higher latency, making them significantly slower compared to the multiplier presented in [2].

These trade-offs must be thoroughly examined through design-space exploration to identify (Pareto-)optimal configurations. However, the analog nature of these circuits results in large configuration spaces that are typically evaluated using slow circuit simulations based on solving differential equations. Additionally, PVT variations need to be taken into account, which adds significant run-time overhead. Consequently, selecting an optimal analog-based PIM configuration is very time-consuming and inefficient, necessitating an effective approach to address this issue.

To address the aforementioned research challenge, we propose a modeling technique called OP-

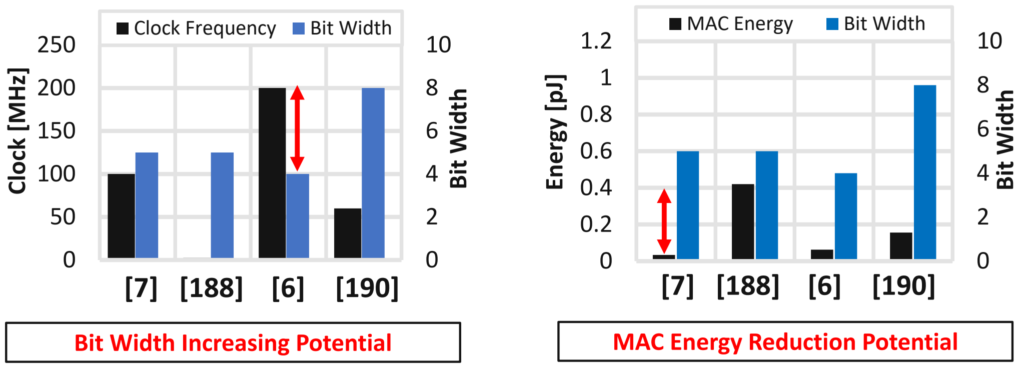


Figure 3.14: State-of-the-art in-SRAM multiplication design space

TIMA as follows:

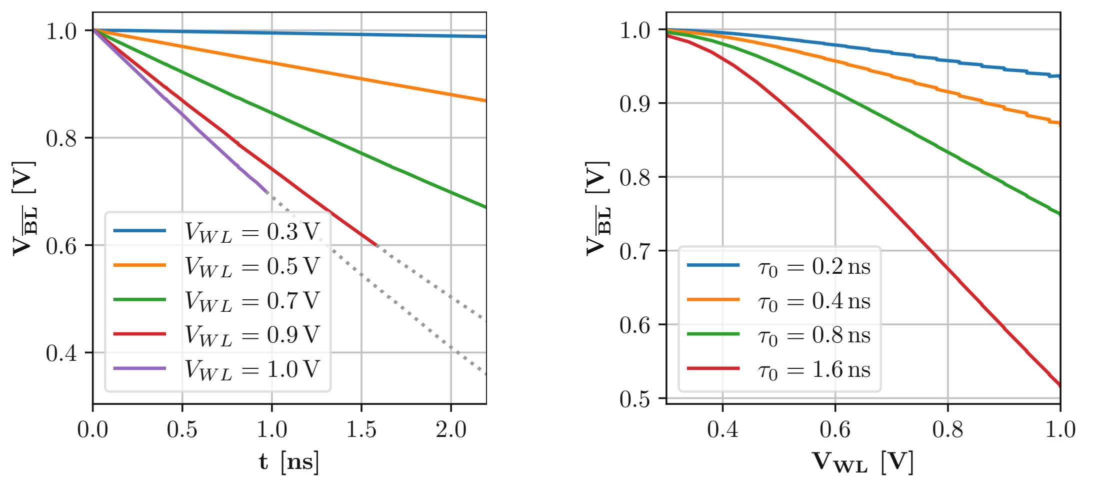
- First, we identify and analyze the error sources at the circuit level and evaluate their impact on SRAM PIM circuits in Section 3.7.1.
- Next, in Section 3.7.2, we develop an accurate model for SRAM BL discharge and power consumption, taking into account nonlinearities and PVT variations. We validate our model using circuit simulation data from a 65 nm technology transistor model.
- In Section 3.7.3, we demonstrate OPTIMA by conducting a rapid design-space exploration of a 4-bit multiplication circuit. We derive a circuit configuration with an optimized energy-accuracy tradeoff.
- Finally, we evaluate the performance of the optimized in-memory multiplier in DNNs. In Section 3.7.4, we assess the accuracy with four standard networks on two major datasets.

3.7.1 In-SRAM Computing Error Sources

To understand why practical implementations of in-SRAM multiplication circuits are limited to small bit widths, we examine the effects of non-idealities and variations in operating conditions at the circuit level. Although these error sources are discussed separately in this section, they all occur together in any discharge-based In-Memory Computation (IMC) circuit. The data presented here is based on circuit simulations using TSMC 65 nm CMOS technology. Additional error sources with lesser impact are analyzed in [140].

Circuit Nonlinearity

In-6T-SRAM multipliers function based on a data-dependent current through the pass transistor. When the stored data is zero, no discharge occurs, and the BLB voltage stays at V_{DD}. However, if the data is '1' and a voltage representing '0' is applied through the WL, a small discharge happens due to the non-



(a) BLB over time with dotted curves indicating saturation

(b) WL voltage dependency when sampled at $t = \tau_0$

Figure 3.15: BLB discharge non-idealities.

zero source-drain current of the MOSFET at V_{th} (see Fig. 3.15a). This asymmetry can lead to different outcomes for $a \times b$ and $b \times a$, reducing the multiplier's accuracy.

Additionally, the quadratic current-voltage relationship of MOSFETs creates a nonlinear discharge dependency on the applied WL voltage (see Fig. 3.15b). The quantization performed with a conventional DAC leads to nonlinear multiplication results. A potential solution is the use of a nonlinear DAC [2], though its practical implementation in circuits poses significant challenges.

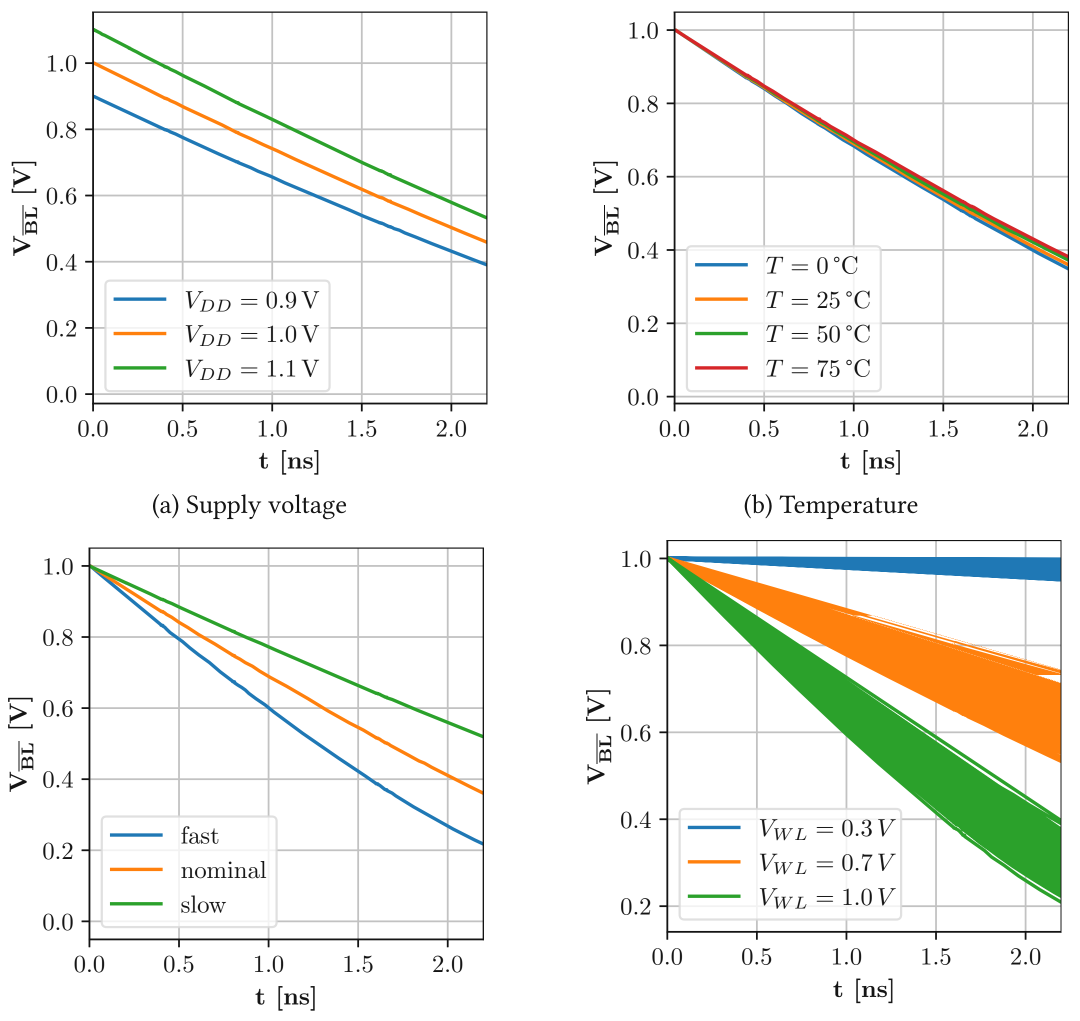
Another non-ideality arises from the transition of pass transistors from the saturation to the linear region with increasing BLB discharges. The saturation condition for M_6 is given by:

$$V_{BLB} \geq V_{WL} - V_{th} \tag{3.16}$$

If the BLB discharges below this threshold, the transistor enters the linear region, which leads to a reduced current and therefore slower discharge. Selecting an appropriate ADC sampling time τ_0 ensures that the pass transistors remain in saturation during the BLB discharge. However, smaller τ_0 values lead to reduced BLB voltage swings, degrading the SNR.

PVT Variations:

PVT variations are the second major limiting factor for accurate in-6T-SRAM computing. Their effect on the discharge voltage is shown in Fig. 3.16. While temperature fluctuations only slightly affect the discharge speed, supply voltage and process variations significantly alter the discharge voltage curves. It is important to note that changes in supply voltage impact not only the SRAM circuit but also the thresholds of ADCs and DACs. Additionally, the variations are data dependent. For example,



(a) Supply voltage

(b) Temperature

(c) Process corners

(d) Mismatch (1000 samples)

Figure 3.16: Influence of PVT variations on the BLB discharge in TSMC 65 nm technology

in Fig. 3.16d, the deviation caused by transistor mismatch increases with the applied WL voltage.

3.7.2 Proposed Modeling Framework: OPTIMA

Our modeling framework, OPTIMA, is available as open-source software[1]. It enables the simulation of analog BL voltage in an event-based manner, similar to digital simulation tools, promising significantly shorter runtime. We achieved this through the following steps:

- Conduct extensive multi-corner circuit simulations.
- Create behavioral models for key analog metrics, incorporating non-idealities and accounting for PVT variations using circuit equations and simulation data.
- Integrate these models into a flexible discrete-time simulation framework written in System Verilog. Our model is versatile enough to accommodate various discharge-based in-SRAM opera-

[1]The source code can be found at `https://github.com/sevjaeg/optima`.

tions.

The behavioral modeling within OPTIMA includes parameterized discharge and energy models:

Discharge Model:

To accurately model the BLB discharge, OPTIMA employs an iterative approach. Initially, the voltage is modeled as a function of time and WL voltage. Subsequently, sources of variation are introduced. All models are based on polynomial functions. In this thesis, $\mathrm{p}_n(X)$ represents a polynomial of degree n with $n+1$ coefficients for the variable X.

$$\delta V(t) \propto V_{WL} \cdot d \cdot t \tag{3.17}$$

As expressed in Eq. 3.17, the voltage on the BLB depends on time and the WL voltage. However, as shown in Fig. 3.15, this relationship is nonlinear. Consequently, the function

$$V_{BLB}\left(t, V_{WL}\right) = V_{DD} + \mathrm{p}_4(V_{\mathrm{od}}) \cdot \mathrm{p}_2(t) \tag{3.18}$$

with the overdrive voltage $V_{\mathrm{od}} = V_{WL} - V_{th}$ is used to model the BLB discharge. This model inherently incorporates the nonlinearities discussed in Section 3.7.1. This model is extended by a supply voltage function:

$$V_{BLB}\left(t, V_{WL}, V_{DD}\right) = V_{BLB}\left(t, V_{WL}\right) \cdot \mathrm{p}_2(\Delta V_{DD}) \tag{3.19}$$

where $\Delta V_{DD} = V_{DD} - V_{DD,\mathrm{nom}}$.

As depicted in Fig. 3.16b, temperature has only a minor effect. Therefore, it is modeled as an additive error term for deviations from the nominal temperature T_{nom} using the function:

$$\begin{aligned} V_{BL}\left(t, V_{WL}, V_{DD}, T\right) =& V_{BLB}\left(t, V_{WL}, V_{DD}\right) \\ &+ \left(t \cdot (T - T_{\mathrm{nom}}) \cdot \mathrm{p}_3(V_{WL})\right) \end{aligned} \tag{3.20}$$

Unlike the previously discussed parameters, process variations are inherently stochastic. Therefore, they are modeled using a statistical approach. As mismatches cause Gaussian variations in the BLB voltage, their standard deviations σ are modeled as:

$$\sigma\left(t, V_{WL}\right) = \mathrm{p}_3(t) \cdot \mathrm{p}_3(V_{WL}) \tag{3.21}$$

Energy Model:

To evaluate energy-accuracy trade-offs in SRAM IMC, the energy consumption of the cells must also be modeled. This is primarily associated with charging the BL capacitances during the pre-charge phases.

The energy consumed during writes is data-independent due to the symmetric cell layout and can be modeled as:

$$E_{\mathrm{wr}}\left(V_{DD}, T\right) = \mathrm{p}_2(V_{DD}) \cdot \mathrm{p}_1(T) \tag{3.22}$$

In contrast, the discharge energy depends on the BLB discharge, which is influenced by both operands (data d and WL voltage):

$$E_{\mathrm{dc}}\left(d, V_{DD}, V_{WL}, T\right) = \mathrm{p}_1(V_{DD}) \cdot \mathrm{p}_3(\Delta V_{BLB}) \cdot \mathrm{p}_1(T) \tag{3.23}$$

The BLB discharge ΔV_{BLB} depends on d, V_{DD}, V_{WL}, and T, and is calculated using the models presented in Eq. 3.18–3.20.

Model Evaluation:

Least-squares fitting is used to determine the coefficients for the models in Eq. 3.18–3.23 based on extensive simulation data. These parameters are then incorporated into the discrete-time simulation model. For transistor mismatch, the Gaussian distribution with σ from Eq. 3.21 is sampled for each discharge. The resulting voltages and energies are shown in Fig. 3.17. The Root Mean Square (RMS) modeling errors are 0.76 mV (basic discharge), 0.88 mV (VDD), 0.76 mV (temperature), 0.59 mV (mismatch σ), 0.15 fJ (write energy), and 0.74 fJ (discharge energy), respectively. These values are below typical ADC LSB voltages for in-SRAM operations, indicating sufficient accuracy for reliable analyses of in-SRAM computing circuits.

3.7.3 Case Study: In-SRAM Multiplier

We demonstrate the application of OPTIMA [1] using the 4-bit multiplication circuit presented in [22]. This circuit employs the discharge principle applied to a 4-bit per word array, as illustrated in Fig. 3.18. The WLs voltages are controlled using a 4-bit DAC, and the discharge occurs at intervals of τ_0, $2\tau_0$, $4\tau_0$, and $8\tau_0$ on different BLBs to implement bit weighing. The discharge voltages are then sampled using switches and capacitors, and the combined discharge voltage is captured using an ADC. For simplicity, we omit the analog accumulation step from the original publication and focus on the multiplication process, which can be modeled efficiently with OPTIMA. To demonstrate our methodology, we define a design space characterized by three circuit parameters:

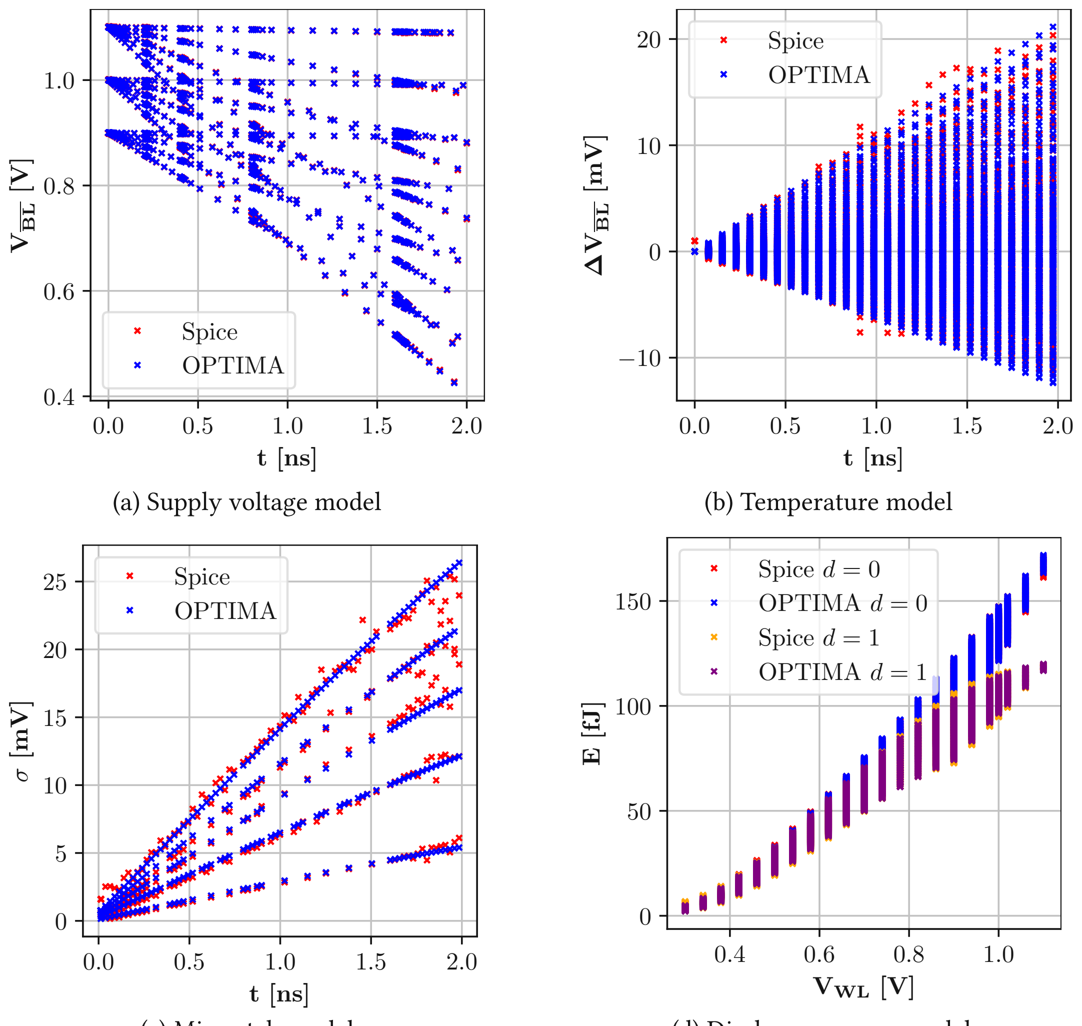
- τ_0: Discharge time of the least significant BLB

- $V_{\mathrm{DAC},0}$: Output voltage of the DAC for data word '0'
- $V_{\mathrm{DAC,FS}}$: Full-scale output voltage of the DAC

We select 48 design corners and simulate the circuit using the OPTIMA framework. The results of this design-space exploration are shown in Fig. 3.19. A higher value of $V_{\mathrm{DAC,FS}}$ leads to a linear increase in energy consumption but is associated with higher accuracy in most cases. Increasing $V_{\mathrm{DAC},0}$ or τ_0 also results in higher energy consumption. The former positively impacts multiplication errors, while the latter has minimal influence on accuracy.

We select three interesting configurations to perform PVT analyses with OPTIMA. The first corner, *fom*, is chosen based on maximizing a Figure of Merit (FOM) that combines the averages of quantization error ϵ_{mul} and energy per operation E_{mul}:

$$\mathrm{FOM} = \frac{1}{\overline{\epsilon_{\mathrm{mul}}} \cdot \overline{E_{\mathrm{mul}}}} \tag{3.24}$$



(a) Supply voltage model (b) Temperature model

(c) Mismatch model (d) Discharge energy model

Figure 3.17: OPTIMA discharge modeling evaluation

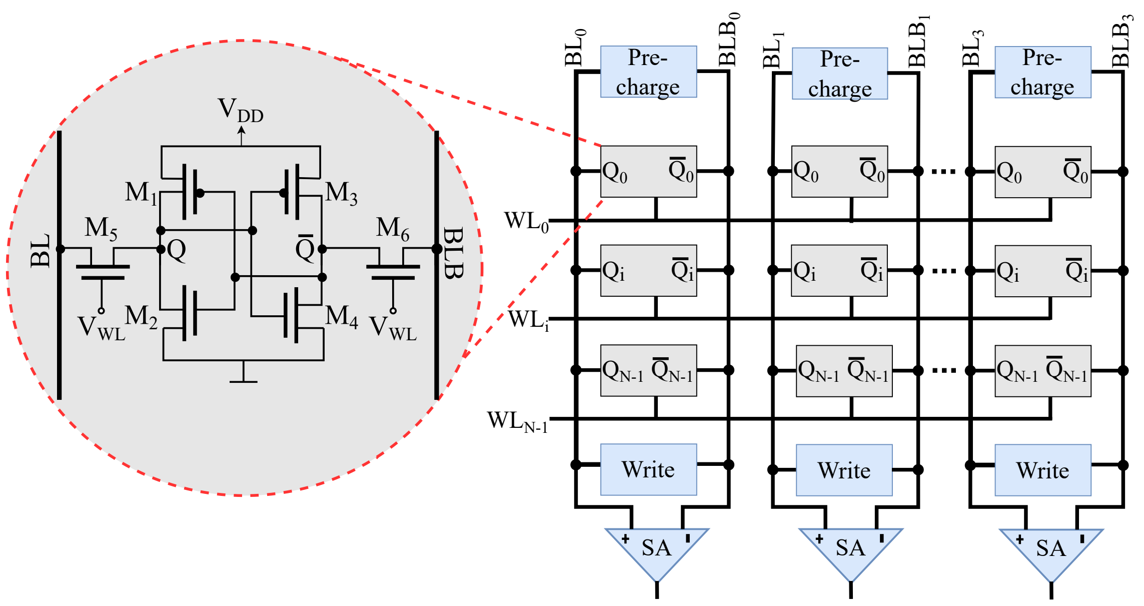


Figure 3.18: 6T SRAM cell and SRAM array

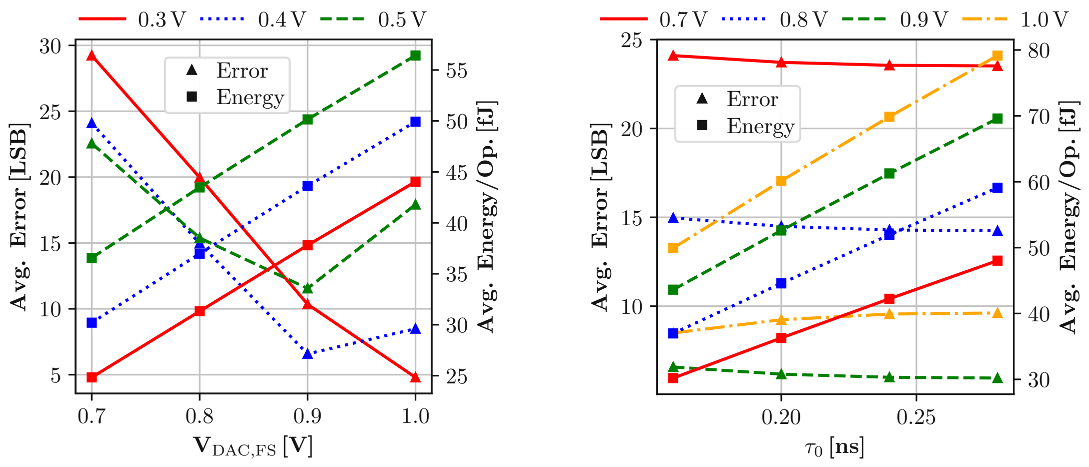


Figure 3.19: Design space corners simulations with OPTIMA for different values of $V_{\mathrm{DAC},0}$ with $\tau_0 = 1.6\,\mathrm{s}$ (left) and $V_{\mathrm{DAC,FS}}$ with $V_{\mathrm{DAC},0} = 0.4\,\mathrm{V}$ (right)

The second corner, *power*, is the one with the minimum energy per multiplication, while the third corner, *mismatch*, shows the smallest standard deviation at the maximum discharge (i.e., it is least affected by process variation). Table 3.3 summarizes the corresponding parameters.

The PVT analysis results, including sampling mismatch corners, are presented in Fig. 3.20. Deviations in average multiplication results indicate the impact of circuit nonlinearities, while high standard deviations suggest susceptibility to mismatch. In the *power* configuration, issues are observed in both cases, with the *variation* corner performing significantly worse than *fom* for small values. However, for large values, it shows robustness against process variation. Voltage and temperature fluctuations also significantly affect the error level, with the *fom* corner being the least susceptible to these variations.

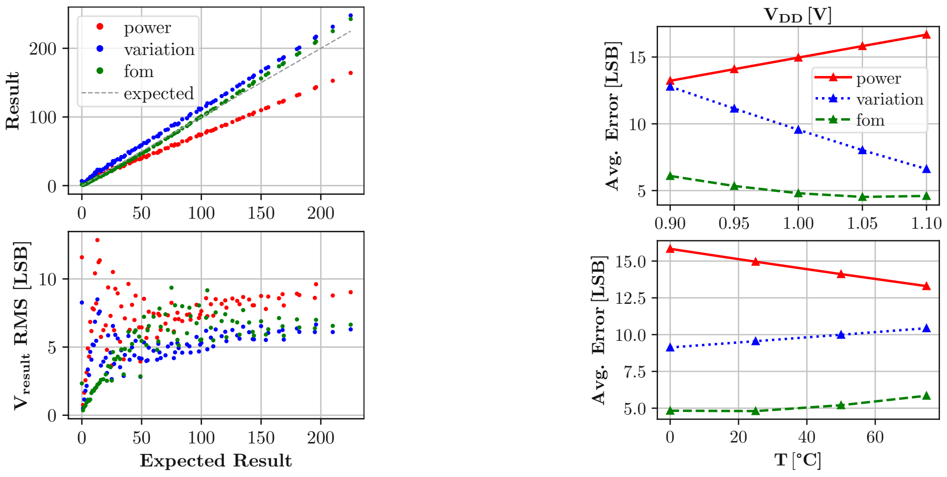


Figure 3.20: Average multiplication results and analog standard deviations (left) as well as influence of voltage and temperature variations on the error (right) for the selected corners

Table 3.3: Selected design corners

Corner	τ_0	$V_{\mathrm{DAC},0}$	$V_{\mathrm{DAC,FS}}$	$\overline{\epsilon_{\mathrm{mul}}}$	$\overline{E_{\mathrm{mul}}}$
fom	0.16 ns	0.3 V	1.0 V	4.78	44 fJ
power	0.16 ns	0.3 V	0.7 V	15	37 fJ
variation	0.24 ns	0.4 V	1.0 V	9.6	69.8 fJ

From our design-space exploration using OPTIMA, we conclude that the *fom* configuration yields the most favorable results. At an operating frequency of 167 MHz, it exhibits an average multiplication error of 4.8 LSBs. The worst-case analog standard deviation is 5.04 mV. For a single operation, including write and multiplication, the average energy consumption is 1.05 pJ. For the multiplication circuit, OPTIMA achieves a speedup of 101× for iteration over the input space and design corners and 28.1× for mismatch MC sampling compared to circuit simulation in Cadence Virtuoso.

3.7.4 Application Analysis

In this research, we evaluate the effectiveness of the proposed in-SRAM multiplier configurations for DNN inference. We use these multipliers in deep learning models for image classification, specifically VGG16, VGG19 [141], ResNet50, and ResNet101 [142], trained on the ImageNet dataset [143]. Additionally, we apply these models to the CIFAR-10 [144] dataset to assess their performance.

In the initial set of experiments, we employ pre-trained DNNs from the Keras model zoo, originally trained on the ImageNet dataset, to demonstrate the efficacy of the selected in-memory multiplier configurations. These pre-trained DNNs use a FLOAT32 number representation. Our experiments aim to use the proposed multiplier for all multiplication operations within these DNNs. To achieve this, we quantize the pre-trained DNNs to an INT4 number representation using post-training quantization.

This quantization process follows the specifications of TensorFlow Lite, with INT8 replaced by INT4

Table 3.4: DNN classification accuracies for CIFAR10

Model	**FLOAT32**	**INT4**	**In-Memory *fom* (INT4)**	**In-Memory *power* (INT4)**	**In-Memory *variation* (INT4)**
	Top-1 Accuracy [%]	**Top-1 Accuracy [%]**	**Top-1 Accuracy [%]**	**Top-1 Accuracy [%]**	**Top-1 Accuracy [%]**
VGG16	92.24	92.04	91.98	87.39	68.10
VGG19	92.71	92.42	92.29	89.79	66.85
ResNet50	93.10	92.86	92.83	90.81	73.83
ResNet101	93.35	93.06	93.04	90.42	69.77

Table 3.5: DNN classification accuracies for ImageNet

Model	**Number of Multiplications [$\times 10^9$]**	**Baseline FLOAT32**		**Baseline INT4**		**In-Memory *fom* (INT4)**		**In-Memory *power* (INT4)**		**In-Memory *variation* (INT4)**	
		Top-1 Accuracy [%]	**Top-5 Accuracy [%]**	**Top-1 Accuracy [%]**	**Top-5 Accuracy [%]**	**Top-1 Accuracy [%]**	**Top-5 Accuracy [%]**	**Top-1 Accuracy [%]**	**Top-5 Accuracy [%]**	**Top-1 Accuracy [%]**	**Top-5 Accuracy [%]**
VGG16	15.61	70.30	90.10	69.25	89.62	68.97	89.11	64.45	81.79	38.22	47.81
VGG19	19.77	71.30	90.00	70.09	89.78	69.91	89.24	63.34	79.61	36.66	48.37
ResNet50	4.14	74.90	92.10	73.48	91.75	73.39	91.65	61.56	80.88	48.07	56.71
ResNet101	7.87	76.40	92.80	75.12	91.91	74.95	91.63	59.77	78.49	48.45	53.19

and corresponding adjustments to the range, restrictions, and other specifications. We then implement retraining procedures to mitigate the impact of quantization on the relevant metrics. The proposed in-memory multiplier configurations are applied to execute all multiplication operations within the DNNs.

Given the focus on image classification, the evaluation emphasizes top-1 and top-5 accuracies as key metrics. The results, depicting the utilization of the in-memory multiplier in three configurations for the quantized ImageNet-trained DNNs, are presented in Table 3.5.

To further evaluate our findings, we conduct a second set of experiments on DNNs to classify the CIFAR-10 dataset. Again, we use INT4 quantization for the DNNs. Additionally, we replace the last layer with a fully-connected layer containing 10 neurons (reflecting the number of classes in CIFAR-10) and employ transfer learning for training. The same experimental conditions are applied to assess the performance of the multiplier configurations. The multiplication operations in each DNN closely align with the values in Table 3.5, as the modifications were confined to the last layer, resulting in minimal changes ($< 0.03\%$) in the number of multiplications for an individual inference.

Among the selected multiplier configurations, the *fom* corner is the most effective in our application. It shows a minimal decrease in accuracy compared to the quantized INT4 DNNs (considered as the baseline) and outperforms all other configurations across all DNNs. For the ImageNet dataset, the top-1 accuracies degrade by only 1.42% and 0.18%, and for the CIFAR-10 dataset, by 0.315% and 0.06% compared to FLOAT32 and INT4 representations, respectively. The *power* configuration trades off accuracy for reduced energy consumption and exhibits more significant accuracy reductions. Interestingly, the *variation* corner achieves top-1 accuracies of only 42.85% and 69.63% for ImageNet and CIFAR-10, respectively, on average. This is due to its high error level for multiplications with small operands (see Fig. 3.20), which predominate in DNN workloads.

To address the challenges of selecting a design for in-SRAM computing circuits, we introduced OPTIMA. This technique provides a fast and accurate design-space exploration, achieving approximately

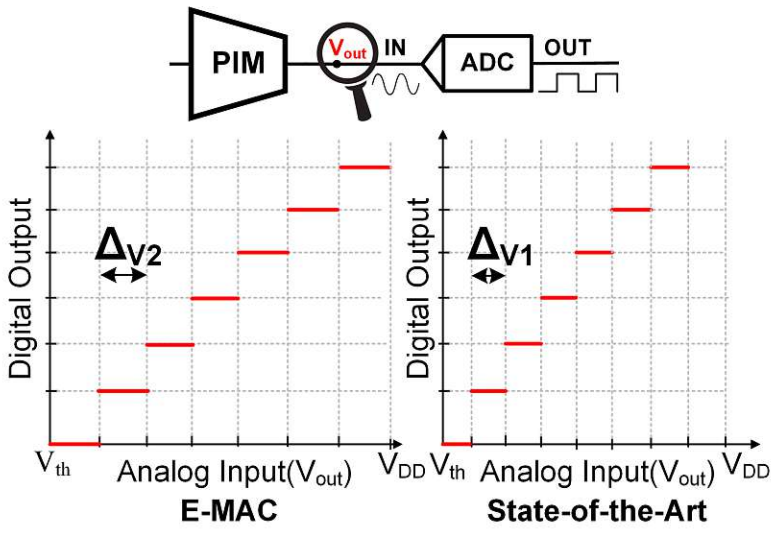


Figure 3.21: Conceptual ADC transfer function for E-MAC vs. state-of-the-art techniques [2,22,36]

100× simulation speed-up while maintaining an RMS modeling error of 0.88 mV for an in-SRAM multiplier. Our exploration techniques yield an optimized multiplier with an energy consumption of 1.05 pJ per 4-bit operation. When applied to quantized DNNs, the classification accuracies are 71.8% (top-1) and 90.4% (top-5) for ImageNet, and 92.5% for CIFAR-10 datasets.

3.8 Enhanced In-SRAM MAC Accuracy via Digital-to-Time Modulation

So far, state-of-the-art analog and digital MAC accelerator techniques suffer from three fundamental drawbacks:

(1) Current analog PIM techniques encounter significant errors due to the quadratic current-voltage relationship of a MOSFET. As noted in the literature [2,22], the discharge behavior does not follow a linear trend with respect to the WL voltage. Most proposed setups employ a DAC to produce a linear WL voltage, but this leads to non-linear outcomes during multiplication. Although these state-of-the-art techniques [2,22,36] have improved accuracy at the circuit level, their design complexity increases due to additional circuitry, resulting in up to a $2\times$ increase in power and energy consumption, particularly concerning for complex applications like NNs.

(2) When the multiplicands are generated via the WL [2,22,36–38], and the V_{GS} is slightly higher than V_{th}, a small discharge occurs due to the non-zero source-drain current of a MOSFET. This effect can significantly impact results, especially for the most significant bits, since their contribution to the total discharge grows exponentially with bit weight. As a result, this circuit non-ideality (refer to section 3.8.3) can cause different results for $a \times b$ compared to $b \times a$. Although the mathematical results of the two multiplications are identical, the voltage drops will differ, as shown in the color-coded representations in Table 3.7. This discrepancy is undesirable as it reduces the available voltage swing for multiple multiplications, ultimately decreasing computation accuracy (see Fig. 3.21). By leveraging the E-MAC technique, we can achieve a larger voltage margin between consecutive voltage drops ($\Delta_{V1} \leq \Delta_{V2}$), representing different multiplication results. This ultimately translates to higher output accuracy.

(3) At the application level, data-agnostic approaches [2, 22, 37, 38] reduce output accuracy and energy efficiency. Preprocessing data and identifying distinctive patterns at a higher abstraction level could significantly improve energy consumption and latency. In the following section, we explore this aspect and propose a data-aware approach for developing a MAC accelerator to address this issue. In order to address the aforementioned research challenges, we propose a novel E-MAC technique that includes the following key features:

- We introduce a new constant amplitude digital-to-time PIM-based MAC accelerator that does not require a DAC. This accelerator uses an analog time-based charge sampling method on the BLB of the SRAM to improve MAC operations for CNNs (see section 3.8.2 for more details). By eliminating the DAC, we remove the associated area and energy overhead present in current PIM-based approaches, thus enhancing the efficiency and performance of our technique compared to traditional methods.
- Our architecture leverages existing memory cells to produce logical bit values, unlike current techniques that depend on capacitance-based weight creation or multi-amplitude voltage. The former increases area and introduces non-ideal behavior into the system, while the latter results in a non-linear drop in BLB voltage, a topic further discussed in Section 3.8.1.
- We propose an analytical model to explain the analog in-SRAM MAC operation and the related non-linear effects (see section 3.8.1). This model helps in understanding the dynamic behavior of the BL and BLB capacitance discharge.
- To the best of our knowledge, our exploration and application of differential sampling have been successful in achieving consistent BLB discharge voltages for operations involving $a \times b$ or $b \times a$. This is a notable improvement over other analog state-of-the-art PIMs, where the discharge voltages for $a \times b$ and $b \times a$ often differ. Such discrepancies usually lead to reduced ADC accuracy and increased vulnerability to circuit and quantization noise. When the discharge product of $a \times b$ does not equal $b \times a$, higher voltage levels are necessary to avoid the loss of multiplication or summation results (see section 3.8.2).
- We have developed a data-aware architecture that incorporates data properties, particularly the probability distribution of CNN weights, into the design process. This approach ensures optimal mapping and operation on the E-MAC architecture. Further details on the design of our E-MAC

architecture are provided in Section 3.8.5.

3.8.1 Problem of Non-linear Voltage Drops During MAC Operation

Considering the SRAM cell shown in Fig. 3.22(a), during a MAC operation, transistors M_2 and M_3 are in the deep triode region, while M_1 and M_4 are cut off. Thus, Fig. 3.22(a) can be simplified to Fig. 3.22(b). Using Kirchhoff's Voltage Law and Kirchhoff's Current Law, V_{out} and I_0 can be determined as follows:

$$\frac{I_0}{C}t = V_{dd} - V_{out} \tag{3.25}$$

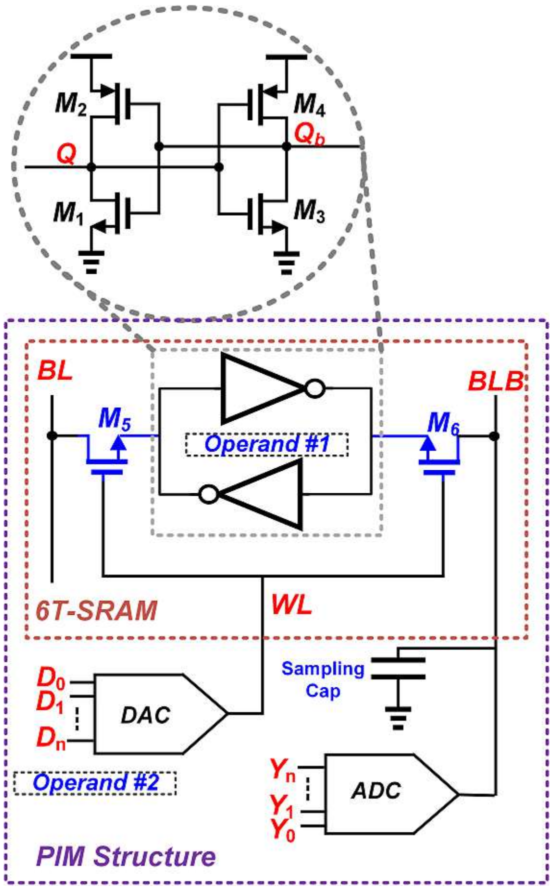
$$I_0 = \frac{\beta}{2}(V_{WL} - V_{th})^2 \tag{3.26}$$

$$\beta = \mu_n C_{ox}\left(\frac{W}{L}\right) \tag{3.27}$$

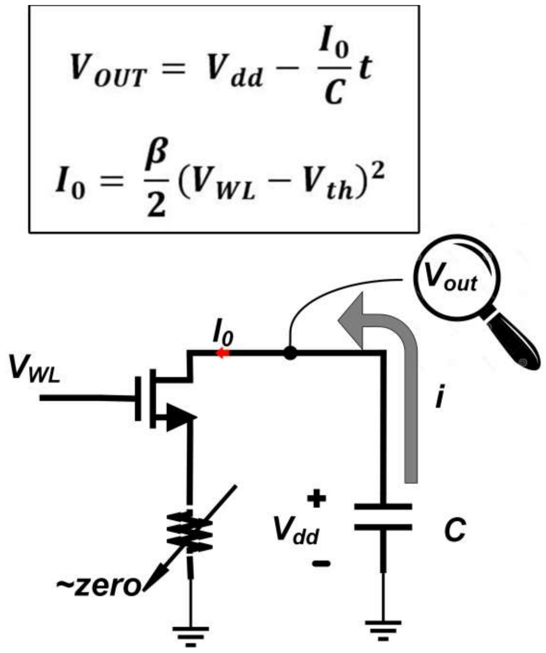
Here, V_{out} is the output voltage of the computation, V_{dd} is the supply voltage (1 V in 65nm CMOS technology), I_0 is the current passing through the transistors, C is the constant parasitic capacitance of the BL or BLB, and t denotes time. V_{WL} is the voltage of the WL, and V_{th} is the threshold voltage of the transistor. β is defined as in Eq. 3.27, where μ_n is the electron mobility of the NMOS, and C_{ox} is the oxide capacitance per unit area. For simplicity, we neglect the effect of channel length modulation in this equation.

Since the parameter C is constant and relates to the physical properties of the design, the only two adjustable parameters to control the output voltage are either t or I_0 (see Eq. 3.25). State-of-the-art approaches such as [2, 22, 36–38] used I_0 as the control signal in their proposed techniques. However, it is important to note that the presence of a quadratic factor in I_0 (see Eq. 3.26) introduces non-linear behavior in the output voltage (V_{out}), as illustrated in Fig. 3.23(a). This non-linearity poses challenges in digital output interpretation, leading to increased errors and reduced accuracy in the output of the ADC. Consequently, it becomes necessary to explore alternative methods that mitigate this non-linearity while maintaining accuracy and minimizing complexity.

Fig. 3.23 shows the multiplication output voltage for both linear and non-linear approaches. Fig. 3.23(a) demonstrates that non-linear techniques can lead to erroneous output interpretation, especially for smaller digital inputs. This occurs because the V_{out} values for different multiplications overlap, causing incorrect interpretation by the ADC (or necessitating a high-precision ADC, which increases system complexity and results in area and energy overhead). This issue has been discussed in previous studies [2, 22, 36–38]. Fig. 3.23(b) illustrates the multiplication output result of the technique presented in [43], which uses additional circuitry (i.e., a root function circuit) to force the voltage to follow a linear trend. *Although this approach improves accuracy, the additional circuitry increases design complexity and*

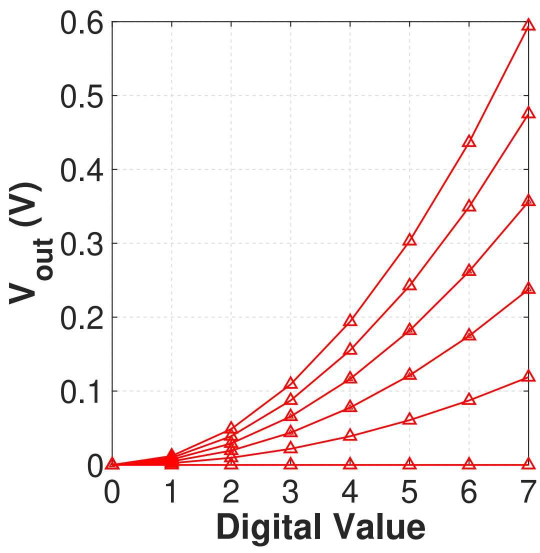


(a)

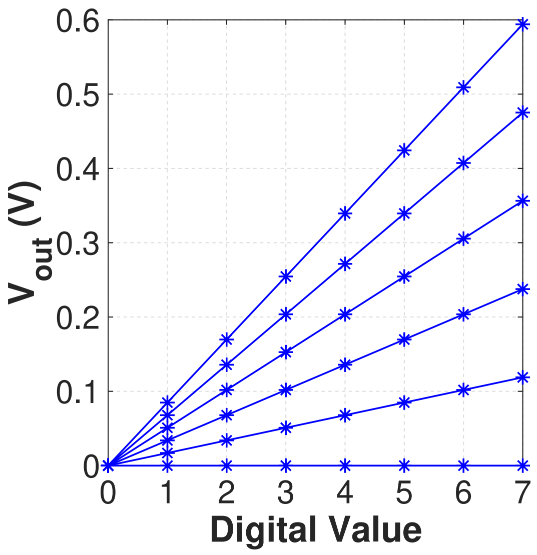

$V_{OUT} = V_{dd} - \frac{I_0}{C}t$
$I_0 = \frac{\beta}{2}(V_{WL} - V_{th})^2$
Vout
I0
VWL
i
+
Vdd
-
C
~zero

(b)

Figure 3.22: (a) 6T-SRAM memory structure, and (b) equal large signal model during MAC operation



(a)



(b)

Figure 3.23: State-of-the-art BLB discharge behavior (a) non-linear [22,36–38], and (b) linear technique [2]

associated energy overhead compared to E-MAC.

In our case, we use the active time duration of a single WL (i.e., the parameter t in Eq. 3.25) as a control signal while keeping the current source (I_0) constant. A replicated instance of the linear current will be used to generate distinct values. As a result, E-MAC not only improves the accuracy of the MAC operation compared to state-of-the-art techniques [2, 22], but also reduces energy consumption and circuit design complexity overhead.

3.8.2 Our Novel E-MAC Technique, and Concept Overview

We introduce and present an analytical model that examines the circuit behavior, enabling more efficient circuit design. Secondly, we propose a solution to address the challenge of achieving identical results with varying voltage drops. Finally, we present our data-aware architecture, which is designed to facilitate the efficient mapping of applications onto the circuit.

WL's Time Modulation Modeling and Constraints

In our approach to achieve meaningful interpolation between the digital input and the analog voltage, **first**, we need to create a coding scheme to map the three-bit operand into a seven-bit operand according to the $4-2-1$ logical weighting scheme stored in the memory column (First Operand $\rightarrow$ Coded Operand). By utilizing the existing memory cells without requiring additional circuitry, our approach avoids imposing any area overhead on the circuit. While encoding a three-bit variable into a seven-bit variable may result in additional power and latency overhead due to extra write operations, this issue is mitigated by the data-aware technique presented in 3.8.6.

Second, the other operand needs to be coded into a time interval using a programmable frequency divider [145] to keep the WL activated (see Table 3.6). For this coding, the WL will not be activated for an operand value of '0' (i.e., the result will always be zero), meaning no discharge will occur in this case, and the output result will remain zero for multiplication. Fig. 3.24 shows the design structure of the proposed methodology. It is important to note that the pulses, equal to Δt_i, can be generated using a frequency synthesizer along with the system's existing clock. Although the Pulse Width Modulation (PWM) technique used here may seem common and outdated, having been proposed in other computing systems, our approach takes a unique perspective. Our primary objective is to maintain the linearity of BL drops. As discussed earlier, any method, including those proposed in state-of-the-art references [2, 22, 36–38], that encodes data in the amplitude of the WL will inevitably result in a quadratic response in the output, leading to non-linearity. By leveraging this approach and considering that the majority of the dataset performs MAC operations on the value zero (more details in section 3.8.6), we can achieve improvements in both accuracy and energy consumption. The sign bit is concatenated to the output result directly, allowing the circuit to behave the same in both negative and positive ranges from -7 to +7.

To clarify the analog coding for the second non-zero operand (as indicated in Table 3.6), we refer to Fig. 3.22 and Fig. 3.24, Eq. 3.25, Eq. 3.26, and the condition for the NMOS transistor to remain in the active region [2]. By considering the provided supply power, we can calculate the maximum sampling time (for more details, see [2]).

$$V_{out} \leq \alpha V_{dd} - V_{th} \tag{3.28}$$

$$T_{max} = \frac{[(1-\alpha)V_{dd} - V_{th}] \times C}{\frac{\beta}{2} \times [(1-\alpha)V_{dd} - V_{th})]^2} \tag{3.29}$$

Table 3.6: Encoding scheme of operands in a MAC process

First Operand	Coded operand	Second Operand	$\Delta t_{2'd}$ (ns)
000	0000000	000	0
001	0000001	001	$(1/7)T_{max}$
010	0000110	010	$(2/7)T_{max}$
011	0000111	011	$(3/7)T_{max}$
100	1111000	100	$(4/7)T_{max}$
101	1111001	101	$(5/7)T_{max}$
110	1111110	110	$(6/7)T_{max}$
111	1111111	111	T_{max}

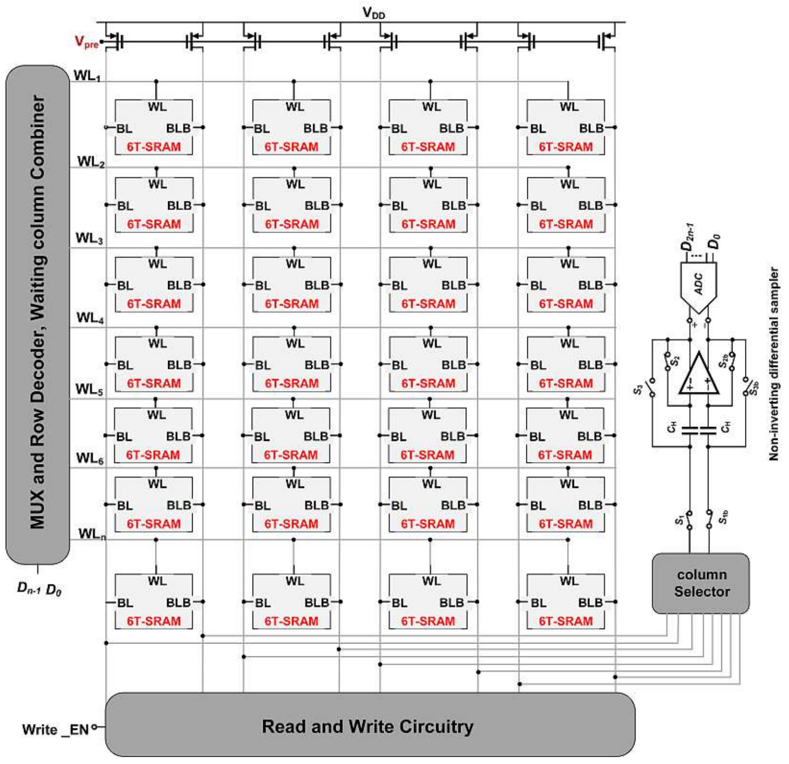


Figure 3.24: Structure of the proposed 6T-SRAM PIM

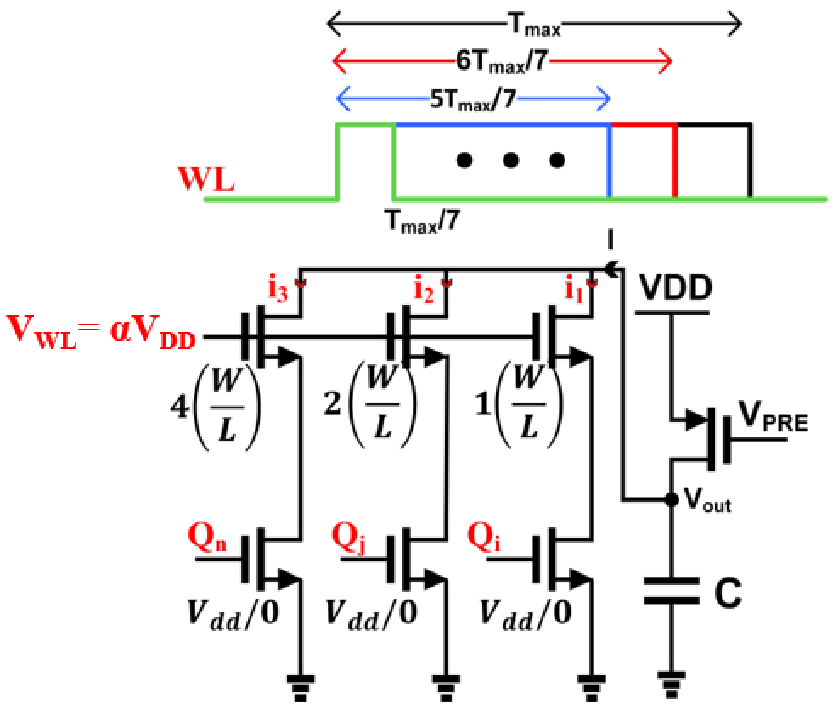


Figure 3.25: Large-signal model of circuits during MAC operations

$$\Delta t_i = (\frac{T_{max}}{2^N - 1}) \times A_{(10)} \tag{3.30}$$

Hence, to calculate each sampling time for other operands, the time coding will follow Eq. 3.29. Δt_i is the required time interval interpolated with the digital input (ranging from 0 to 7 as shown in Table 3.6). T_{max} is the maximum required time that transistor M_6 remains active, corresponding to a 7×7 multiplication. $A_{(10)}$ is the E-MAC equivalent of the input, and N is the bit-width of the input. The behavior of the proposed architecture is modeled in Fig. 3.25. The MSB, consisting of four memory cells, is modeled as a transistor with a size of $4\left(\frac{W}{L}\right)$, while the LSB fits into a $1\left(\frac{W}{L}\right)$ transistor. The second operand is passed to the gate of lower transistors and can be either '0' (V) or V_{dd}. Therefore, by applying KVL for V_{out} and KCL on the branches, we obtain Eq. 3.31. Note that α is a scaling factor, typically less than one, applied to the gates of the transistors associated with the WL. The selection of α depends on the desired rise and fall times of the WL voltages. Lowering the voltage at the gates results in faster rise and fall times. To better explain our methodology, let's consider an example scenario illustrated in Fig. 3.26. Suppose we need to multiply 5 by 4 in our application. To perform this operation, we first encode the first operand (5) using seven memory elements, as shown in Table 3.6. The resulting binary code "1111001" is then written into the seven memory cells indicated in blue. Next, we pass the second operand (4) through the mutual WL of the aforementioned cells, in the form of the time duration of the

WL signal. Throughout the process, the amplitude of the WL signal remains constant.

$$V_{out} = V_{dd} - \frac{\sum_{i=0}^{N-1} I_i}{C} \Delta t_i \tag{3.31}$$

As depicted in Fig. 3.27, after the precharge process, the WL signal remains active for $\frac{4}{7} \times T_{max}$, causing the capacitance C_L discharge via weighted (4-2-1) equal current sources. In this example, the middle two cells are passing zero ampere current. Finally, the output can be sampled once the WL signal is deactivated.

3.8.3 Problem of Identical Results with Diverse Voltage Drops

The use of analog in-SRAM multiplication presents an inherent challenge due to the nature of the multiplication process. As previously discussed, one operand is stored in the cell, while the other is provided as an analog input to the WL. This analog input can be represented by voltage amplitudes, as seen in previous research, or by time duration, as introduced in last section. When the WL is activated, there is a decrease in the charge stored in either the BL or BLB capacitance. This decrease is then recovered and interpreted as the multiplication result by using an ADC at the output. In this setup, one operand is in the digital domain while the other is an analog signal. This can lead to a situation where different voltage drops result in different outputs. For instance, when multiplying two numbers, A and B, where both range from 0 to 3 (i.e., up to 3-bit-width), there are only seven possible unique results.

Table 3.7 shows the multiplication output for a scenario of 4-bit number multiplication. The table reveals that the multiplication results have several similarities, with only a few distinct products out of all possible multiplications (highlighted through color coding). However, current analog in-SRAM multiplication methods used in state-of-the-art approaches [2,22,36–38] have not addressed this issue. Specifically, when multiplying 1×7 or 7×1, two different voltage drops occur over the BL/BLB, even though the result is the same and equal to 7. This discrepancy results in additional circuit costs. To address this problem, we utilized the Differential Voltage Sampling (DVS) technique. This adjustment allowed us to allocate the extra voltage margin to improve the accuracy of multiplication, increase the bit width, or save energy. Current state-of-the-art methods sample the voltage drops on BL or BLB to compute the result. These voltages can be represented by Eq. 3.32:

$$V_{BL/BLB} = V_{dd} - \frac{\sum_{i=0}^{N-1} I_i}{C} \Delta t_i \tag{3.32}$$

Eq. 3.32 is inherently non-linear due to the V_{dd} factor, which varies over time while sampling. By leveraging DVS, the non-linear term V_{dd} is eliminated (since both BL and BLB share one V_{dd} factor), leading to Eq. 3.33.

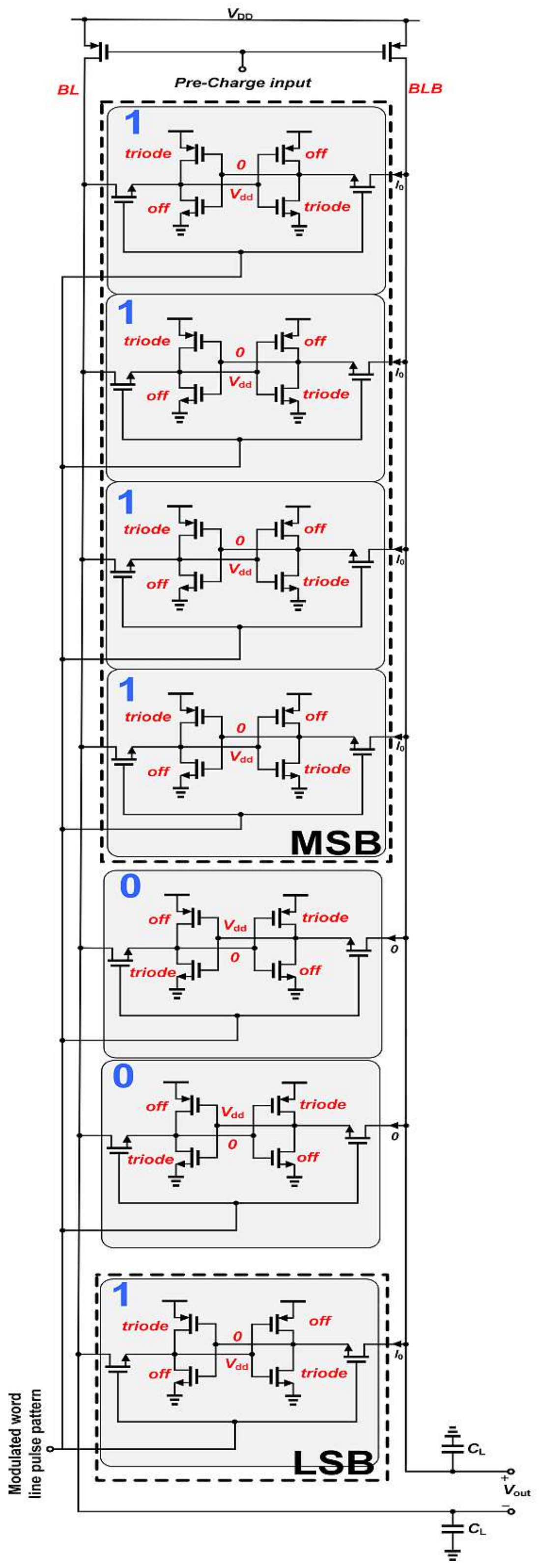


Figure 3.26: Operand encoding scheme for MAC operation (5×4)

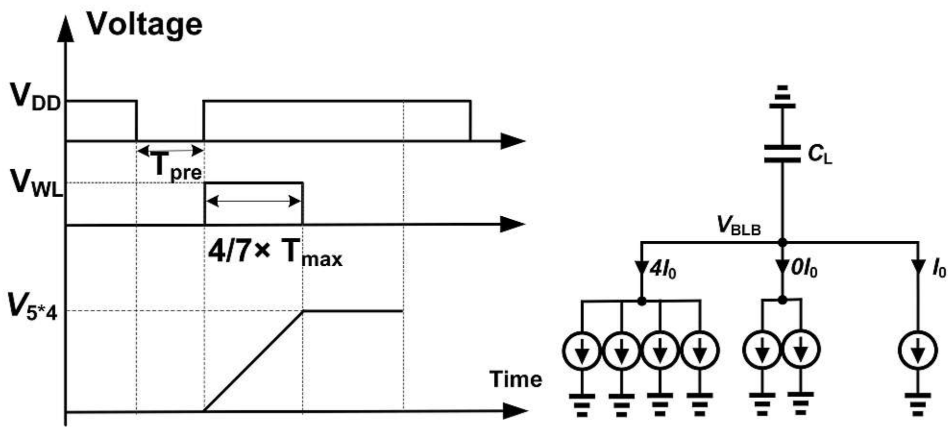


Figure 3.27: Discharge behavior during MAC operation (5×4)

Table 3.7: All possible 3-bit operands multiplication

$A \times B$	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	8	10	12	14
3	0	3	6	9	12	15	18	21
4	0	4	8	12	16	20	24	28
5	0	5	10	15	20	25	30	35
6	0	6	12	18	24	30	36	42
7	0	7	14	21	28	35	42	49

$$| V_{BL/BLB}(DVS) | = \frac{\sum_{i=0}^{N-1} I_i}{C} \Delta t_i \tag{3.33}$$

By using this differential technique, we have effectively resolved the non-linearity issue in the output voltage, a problem that has challenged state-of-the-art methods. It is important to highlight that achieving linearity is crucial for ensuring equal voltage drops and consistent results. For instance, in this approach, the output voltage for both (4×5) and (5×4) computations is identical. In the first case, the first operand (i.e., 4) is equal to $4.I_0$ multiplied by $\frac{5}{7}.T_{max}$, which equals $\frac{20}{7}.I_0.T_{max}$. This value is exactly the same for the second case of 5×4.

3.8.4 Analytical Model and E-MAC Circuit Implementation and Results

We developed our analytical model using MATLAB R2022a. The circuit architecture of our proposed 6T-SRAM PIM (Fig. 3.24) is implemented in a 65 nm CMOS technology with a supply voltage of 1 V. The circuit parameters, such as the size of transistors, BLB capacitance, and T_{max}, are optimized using Cadence SPECTRE transient simulation according to Eq. 3.29, Eq. 3.30, and Eq. 3.31 ($V_{th} = 250$ mV and $V_{DD} = 1$ V).

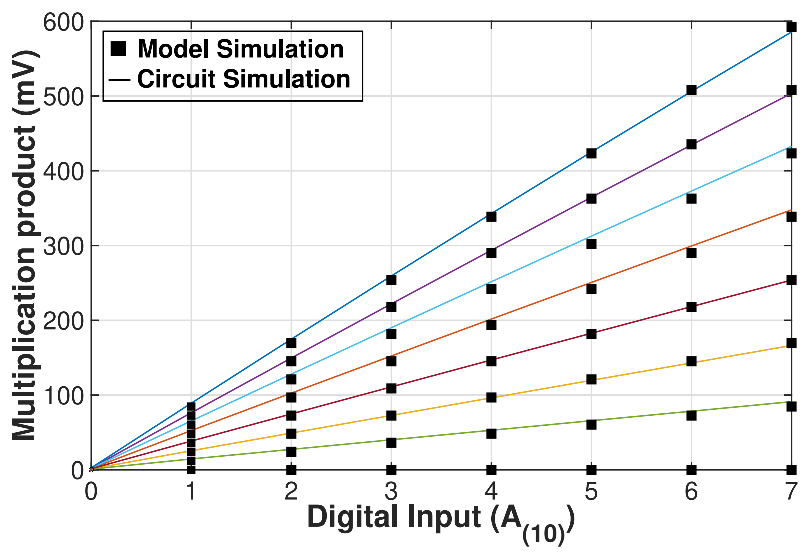
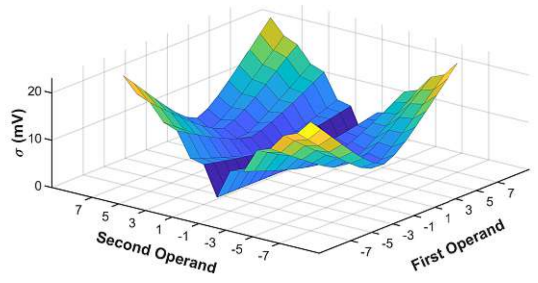
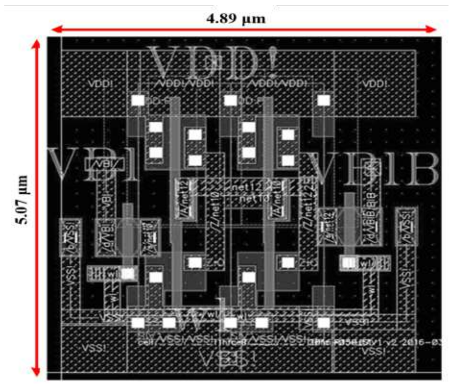


Figure 3.28: The circuit results verify the accuracy of the proposed analytical model

Our optimization objective is to minimize the power and area of our circuit. Fig. 3.28 shows the overlapping plot of our model and the circuit simulation results. As shown, our model accurately predicted the behavior of the final circuit. The slight deviation in specific numerical values is due to parasitic capacitance and the physical parameters of the model, which were also considered in the circuit simulation. However, this does not affect the model's reliability as it follows the same linear trend for the multiplication result. Additionally, when the multiplication result is the same, the discharge of the BL is also similar, as seen in Fig. 3.28. The incorporation of the mathematical model alongside the circuit simulation validates the effectiveness of our approach in maintaining the necessary linearity of the voltage drop. Considering two signed integer numbers in the range of $[-7 : +7]$ for multiplication, we performed a 1000-point MC simulation to depict the effect of threshold voltage, gate oxide thickness, and various mobility of transistors on the accuracy of the result in Fig. 3.29a. We achieved a 23 mV standard deviation from the accurate result for the worst-case multiplication (7×7), which outperforms the state-of-the-art accuracy by 73%. Fig. 3.30 depicts the result for both best (0×0) and worst cases (7×7). Fig. 3.29b illustrates the 6T-SRAM cell layout, which occupies $5.07\mu m \times 4.89\mu m$ in the 65 nm CMOS technology. Table 3.8 demonstrates the comprehensive analysis of the proposed architecture in circuit metrics compared with state-of-the-art approaches [2,22,36–38]. As can be seen, E-MAC outperforms the state-of-the-art and achieves 47.31% energy improvement compared with the best result, and in terms of accuracy, we improved by 73.25%. We should elaborate that there exist other techniques within the domain of in-SRAM, as detailed in [146]. These alternatives may operate either in the digital domain or as near-memory core accelerators. However, we have chosen the aforementioned techniques to compare due to their alignment with the proposed abstraction level. It's important to note that digital or near-memory approaches would not perform as effectively as pure analog in-SRAM tech-



(a) Standard Deviation of the multiplication results



(b) The area of the proposed in-SRAM multiplier

Figure 3.29: Standard Deviation (STD) and the cell area of the proposed circuit

Table 3.8: Comparison with state-of-the-art in-SRAM analog-based MAC accelerators

	[3]	[22]	[37]	[2]	[36]	[38]
Technology (nm)	65	65	65	65	180	65
Supply Voltage (V)	1	1.2	1.2	1	1.8	1
Bits-width	4	5	5	4	5	8
MAC Energy (pJ)	0.147	0.279	3.5	0.523	1.167	1.3
Accuracy $(std.dev)$	0.023	0.6	/	0.086	/	/
CLK Freq. (MHz)	103.1	100	2.5	200	/	60-125

niques [146, 147] in this context, rendering a comparison meaningless. In our study, we have rigorously addressed the impact of parasitic capacitance through three comprehensive approaches. Firstly, we defined parameter C to encapsulate the cumulative effect of all parasitic capacitors from the node to the ground, as detailed in Fig. 3.22.b and 3.25. This parameter includes contributions from the drain-bulk, drain-source, and drain-gate capacitances of all connected transistors, along with a column capacitor influencing the discharge behavior of each column. The significance of parameter C in circuit behavior is thoroughly analyzed in Equations 3.25 and 3.29.

Secondly, the 6T-SRAM cell layout, presented in Fig. 3.29b, accounts for the parasitic capacitances of the transistors and interconnects between them. The hierarchical layout approach ensures accurate post-layout simulations, including peripheral circuitry, with results depicted in Fig. 3.28, 3.29a, and 3.30. Lastly, our proposed structure underwent post-layout circuit simulation using the TSMC 65 nm CMOS design process, incorporating precise parasitic modeling via BSIM 4 and validated through MC simulations. While our current study focuses on the 65 nm technology node, we acknowledge the potential significance of parasitic capacitance in more advanced nodes such as 28 nm and 14 nm. The study [2] demonstrated that despite pronounced deep-submicron effects in smaller channel lengths, the discharge behavior of the BLB port remains unaffected, indicating robustness in our design. Furthermore, the shift from quadratic to linear current behavior in more advanced technologies is anticipated to enhance the accuracy of our design, as detailed in our previous findings.

3.8.5 Performance Evaluation

The proposed in-memory multiplier technique offers significant potential for energy-efficient DNN inference on resource-constrained devices. To assess the effectiveness of the proposed technique, we have focused on CNNs due to their substantial energy consumption and data intensity requirements. This paper utilizes a CNN inspired by Lenet5 [148] and the VGG16 [141] architecture to perform image classification tasks on the MNIST [149] and ImageNet [150] datasets, as used in state-of-the-art approaches.

The first CNN consists of eight layers, including five convolutional layers and three fully connected layers, with the final layer serving as the output layer indicating the class of the input image. The VGG16 model processes a 224x224 RGB image as input and outputs a vector of probabilities for each of the 1000 possible object categories in the ImageNet dataset. The first layer of the VGG16 model is a convolutional layer with 64 filters, followed by a max-pooling layer that reduces the spatial dimensionality by half. The subsequent layers consist of convolutional layers with 128, 256, and 512 filters, each followed by a max-pooling layer.

The final layers of the VGG16 model are fully connected layers that perform the final classification. Specifically, there are two fully connected layers with 4096 neurons each, followed by a final output layer with 1000 neurons corresponding to the number of classes in the ImageNet dataset. A notable feature of the VGG16 architecture is its use of small 3x3 convolutional filters, which enables the model

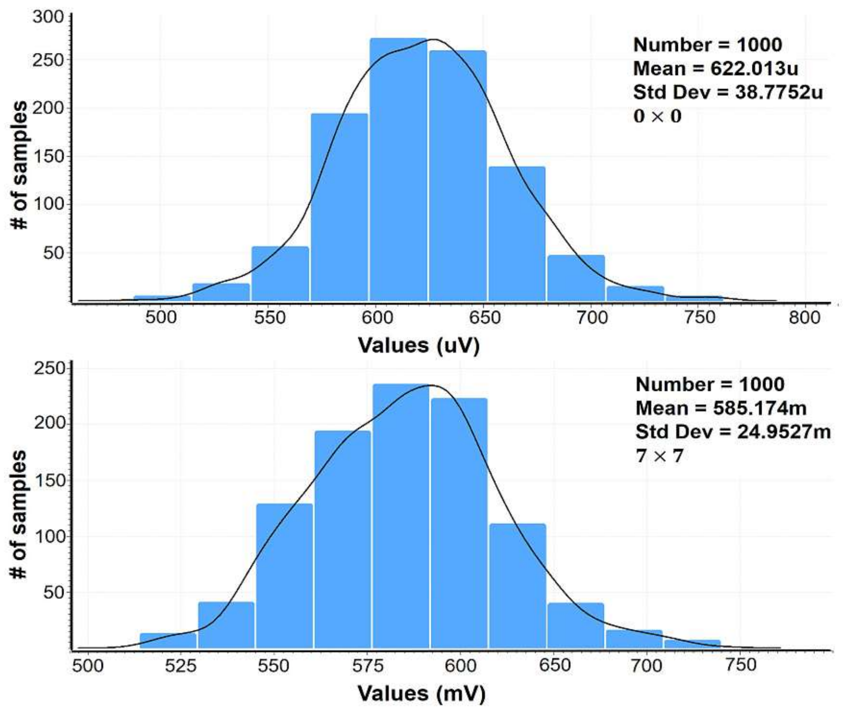
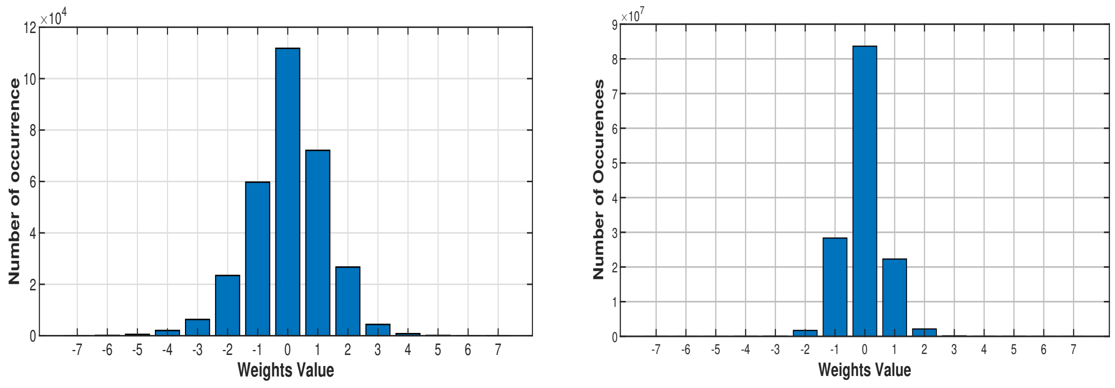


Figure 3.30: MC simulation for 1000 points for both best (0×0) and worst (7×7) multiplications



(a) (b)

Figure 3.31: (a) Histogram of the CNN weights, and (b) profiling of VGG16 weight distribution

to learn more local features of the input image. Additionally, the uniform architecture makes it easier to understand and replicate the model's performance.

Most operations in DNNs are multiplications, which have higher energy consumption compared to other arithmetic operations [151]. Therefore, techniques like the in-memory multiplier, which can reduce the energy consumption of multiplication operations, are crucial for improving the energy efficiency of DNNs.

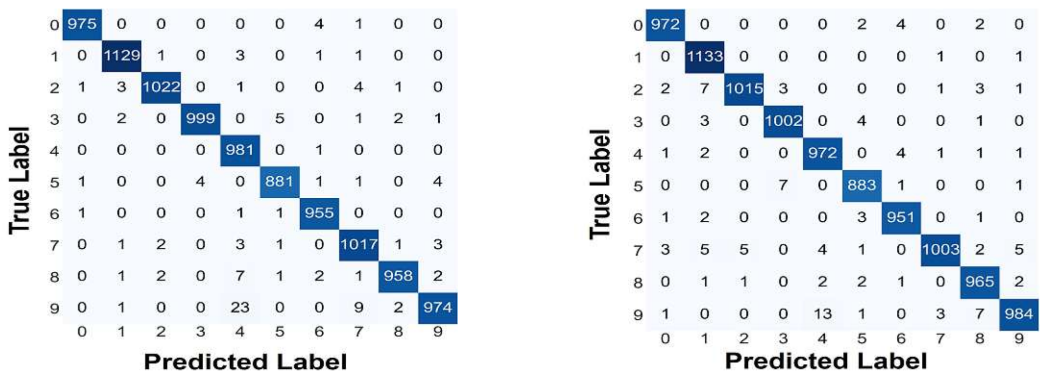
3.8.6 Data-Awareness

In our proposed method, we aimed to minimize computation errors by incorporating data-awareness into the design. To achieve this, we designed our multiplier to have the least computation error for the majority of multiplications. Since our case studies and experiments involve quantized pre-trained CNNs, the weights of the CNNs are known beforehand. We calculated the histogram of these weights to identify which values occur most frequently in our CNNs.

Fig. 3.31a and Fig. 3.31b show the histograms of the weight values for our case studies. As illustrated, most of the weight values in both cases fall within the range of [-2,2]. This insight was crucial in our design process. We ensured that our proposed multiplier exhibits the least error within the [-2,2] range, as shown in Fig. 3.29a. This design consideration significantly enhances the accuracy of the multiplier for the most common weight values in our CNNs, thereby improving the overall performance and efficiency of the system.

3.8.7 CNN Implementation, Evaluation, and Results

In addition to mitigating performance degradation, our proposed in-memory multiplier can also enhance energy-efficient DNN inference on resource-constrained devices.



True Label \ Predicted Label	0	1	2	3	4	5	6	7	8	9
0	975	0	0	0	0	0	4	1	0	0
1	0	1129	1	0	3	0	1	1	0	0
2	1	3	1022	0	1	0	0	4	1	0
3	0	2	0	999	0	5	0	1	2	1
4	0	0	0	0	981	0	1	0	0	0
5	1	0	0	4	0	881	1	1	0	4
6	1	0	0	0	1	1	955	0	0	0
7	0	1	2	0	3	1	0	1017	1	3
8	0	1	2	0	7	1	2	1	958	2
9	0	1	0	0	23	0	0	9	2	974

(a)

True Label \ Predicted Label	0	1	2	3	4	5	6	7	8	9
0	972	0	0	0	0	2	4	0	2	0
1	0	1133	0	0	0	0	0	1	0	1
2	2	7	1015	3	0	0	0	1	3	1
3	0	3	0	1002	0	4	0	0	1	0
4	1	2	0	0	972	0	4	1	1	1
5	0	0	0	7	0	883	1	0	0	1
6	1	2	0	0	0	3	951	0	1	0
7	3	5	5	0	4	1	0	1003	2	5
8	0	1	1	0	2	2	1	0	965	2
9	1	0	0	0	13	1	0	3	7	984

(b)

Figure 3.32: Confusion matrix for the accelerated trained quantized CNN in (a) 8-bit and (b) 4-bit number representations.

4-bit symmetric quantization [152] is a specific type of quantization used in DNN. In this technique, the range of values that weights and activations can take is divided into eight equal-sized intervals (absolute values), with each interval representing 1/8th of the total range. The values are then rounded to the nearest value within the interval.

Since 4-bit quantization uses only eight intervals (absolute values), it can represent only eight distinct values. This significantly reduces the memory and computational requirements of DNNs, making them more efficient and easier to deploy on hardware with limited resources. However, the reduction in precision can also lead to quantization errors and decreased model accuracy.

CNN-Based Image Classification (MNIST)

Our experiments use the MNIST dataset [149] for training and inference (testing). The MNIST dataset includes 70,000 images of handwritten digits (0-9) across ten classes. 60,000 images are used for training, while 10,000 images are used for the inference phase. First, the CNN is trained using the TensorFlow library [153] in Python. The training is performed using the ADAM optimizer [154], with a batch size of 2048, for 50 epochs, and categorical cross-entropy as the loss function.

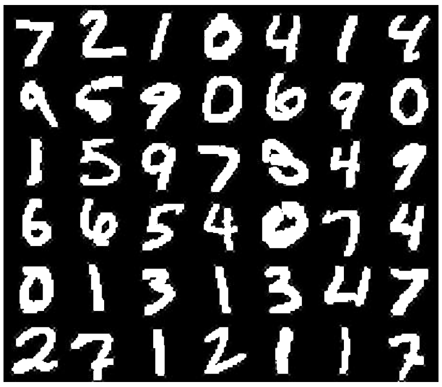
After training, the CNN is quantized to int-4 (-7 to 7) by considering the maximum absolute value of each layer in the trained network. Subsequently, all multiplication operations are replaced by the proposed in-memory multiplier. The quantized CNN (4-bit weights and activations) achieved 98.91% accuracy on the MNIST test data. The confusion matrix and performance criteria of the trained quantized (int-4) CNN are shown in Fig. 3.32 and Table 3.9, respectively.

As demonstrated, the accelerator's output is an 8-bit value interpreted from an analog voltage of the BLB. The voltage of the BLB ranges from 0.4 to 1 volt, discretized into 256 levels, with each level corresponding to a number in the range -128 to 127, using an ADC.

Table 3.9: Performance results for float quantized, and proposed model for LeNet5-Inspired CNN.

Model	Accuracy	Precision	Recall	F1-Score
Original (float32)	99.29	99.28	99.28	99.28
Quantized (int4)	99.01	99.01	99.02	99.01
In-Memory (int4)	98.91	98.91	98.90	98.90

As mentioned, the weights of the CNN are quantized to an int-4 representation (-7 to 7, symmetric), while the MNIST dataset is in uint-8 format, with values ranging from 0 to 255. To align with the int-4 representation, a mapping is done from uint-8 to a (0 to 7) interval. Fig. 3.33 depicts the first 42 images of the MNIST dataset in uint-8 and their mapped versions to illustrate the difference between these formats. As shown, the structures remain the same, with only marginal differences in minor details, which are negligible and intangible for human perception. Furthermore, the t-distributed stochastic neighbor embedding (T-SNE) plot [155] demonstrates that the quantization effect on the MNIST dataset is negligible. The distributions of the MNIST dataset in uint-8 and int-4 representations are shown in Fig. 3.34. As illustrated in Fig. 3.34, our quantization method has not significantly affected the MNIST


(a) MNIST dataset in 8-bit representation

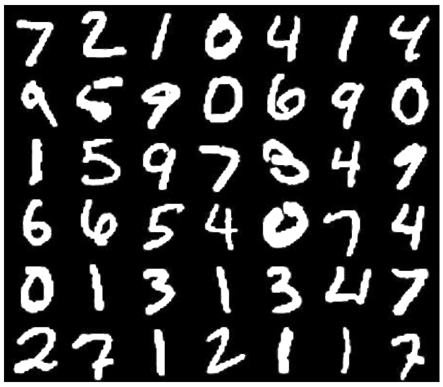
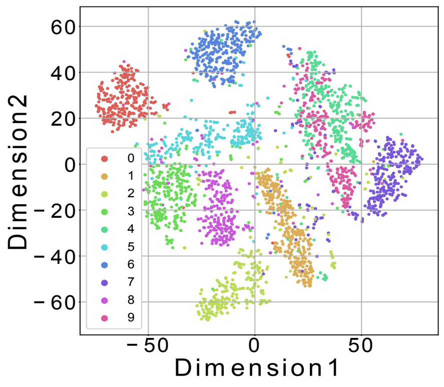
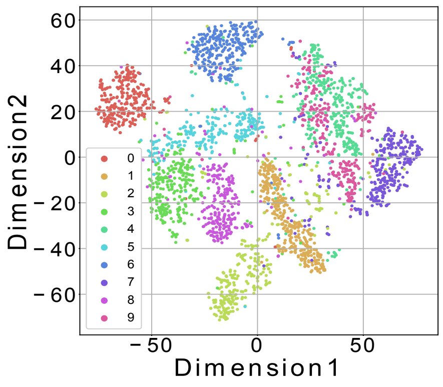

(b) MNIST dataset in 3-bit representation

Figure 3.33: MNIST dataset in two different number representation



(a)



(b)

Figure 3.34: T-SNE visualization of the MNIST dataset in (a) 8-bit and (b) 3-bit representation

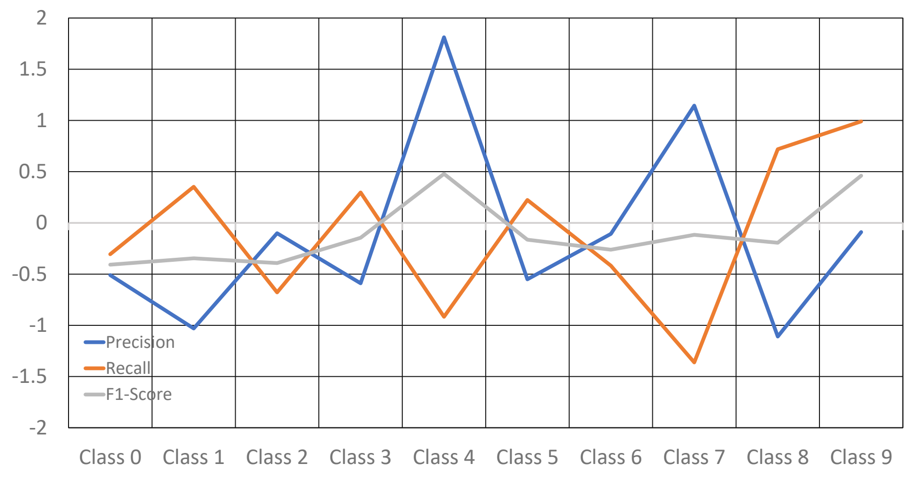


Figure 3.35: Performance change for each metric.

data distribution. During the inference phase, all multiplication operations are performed using the proposed in-SRAM multipliers to achieve energy efficiency with negligible performance degradation. The performance criteria (Table 3.9) indicate that the CNN's performance remains nearly the same with negligible accuracy degradation. Additionally, the performance metrics for each class are depicted in Fig. 3.35.

As seen in Fig. 3.35, although the overall accuracy has a negligible drop compared to the baseline architecture in 4-bit data representation, the classification accuracy for classes four, seven, and nine has improved. This improvement is due to the pattern, spatial structure, and numbers used to represent these classes. Next, we evaluated the performance of the proposed CNN by considering the energy aspect. We compared our proposed in-SRAM multiplier with a conventional von Neumann system [62] and state-of-the-art in-memory multiplier accelerators. For a fair comparison, we used the same experimental setups and circuit parameters as in [22] and [2]. We reported the energy gain and energy per inference.

For a comprehensive study, we also compared our model's performance with other in-SRAM multipliers with analog computation implemented for the MNIST dataset. As shown in Table 3.10, we achieved minimal accuracy degradation, indicating that the E-MAC in-SRAM multiplier minimally impacts the CNN's performance while providing up to 15.53x energy efficiency. Additionally, we experienced the least accuracy degradation compared to the state-of-the-art (see Table 3.10).

VGG16 (ImageNet)

In this case, we utilized the ImageNet dataset [150] for inference (testing) purposes in our experiments. The ImageNet dataset, with 1000 classes, is a highly diverse and challenging benchmark for image

Table 3.10: Comparison with the related works, tested with the MNIST Dataset.

Work	Input/Weight Precision	Algorithm	Baseline Accuracy	Accuracy Degradation
[22]	5/5	LeNet-5	**99.20**%	0.15%
[62]	6/4	LeNet-5	99.7%	1.30%
[156]	-/4	MLP	98.27%	0.11%
[157]	ternary	MLP	98.77%	0.12%
[158]	(16 or 32)/(8 or 16)	MLP	98.36%	0.14%
[159]	16/4	LeNet-5	98%	1%
[160]	14/(4 or 5)	SNN	90%	-
This Work	**3/4**	CNN	99.01%	**0.10**%

Table 3.11: Performance results for float, quantized, and proposed model for VGG16

Model	Top-1 Accuracy	Top-5 Accuracy	Geomean Of Top-1 & Top-5 Accuracies
Original (float32)	71.30%	90.10%	80.15%
Quantized (int4)	70.29%	89.24%	79.20%
In-Memory Multiplier	70.12%	89.18%	79.08%

recognition models. It contains over 1.28 million hand-annotated images of objects, scenes, and animals across 1000 distinct categories. To achieve high performance on the ImageNet 1K dataset, models must be able to recognize subtle differences between images and accurately classify objects in diverse settings. The dataset's large size, variety of images, and challenging nature have made it a valuable resource for developing and testing image recognition techniques. Consequently, it has become a standard benchmark for evaluating the performance of computer vision models [161].

In this study, we evaluated the VGG16 using the proposed in-memory multiplier to assess its effectiveness on a higher level. To evaluate our classifier, we used the following performance metrics: top-1 accuracy and top-5 accuracy of the original and quantized models.

To quantize the trained VGG16 weights from float32 to int4, we first normalized them based on the maximum absolute value of weights in each layer, then multiplied by seven and rounded to the nearest integer value (in the range [-7, 7]). We then replaced all multiplications in our model with the proposed in-memory multiplier. As shown in Table 3.11, due to quantization, the top-1 accuracy and top-5 accuracy have been degraded by only 1.01% and 0.86%, respectively. However, using the in-memory multiplier resulted in up to 0.17% and 0.06% degradation in top-1 accuracy and top-5 accuracy, respectively, compared to our quantized baseline model, due to the computational error of the aforementioned in-memory multiplier. Nonetheless, our proposed in-memory multiplier can be used effectively in VGG16 due to the negligible change in performance criteria.

Our results show that we can achieve a 7.40× energy gain by sacrificing 1.18% and 0.92% of top-1 accuracy and top-5 accuracy, respectively. These results demonstrate that in scenarios with power

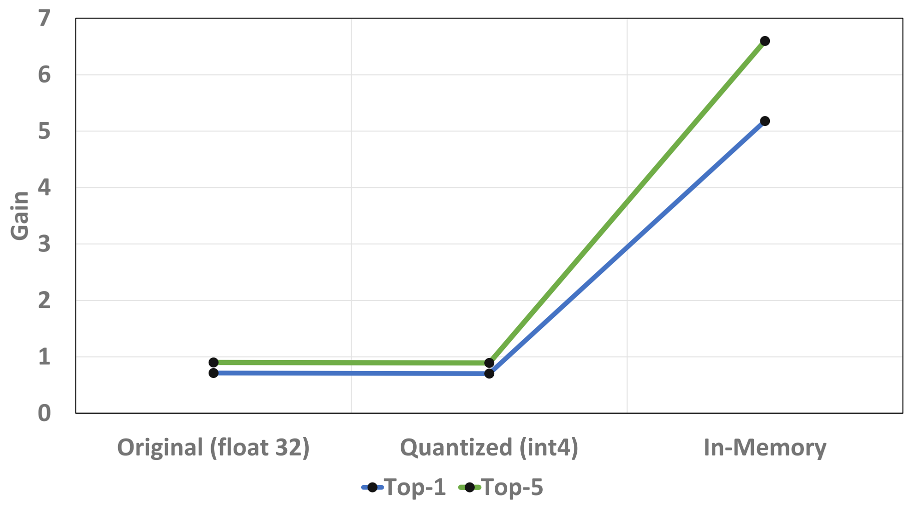


Figure 3.36: Combination of energy gain and performance metrics for VGG16 (Top1 and Top5)

Table 3.12: Energy usages and gains of VGG16 per inference.

	Energy (mJ/inference)	**Relatively Energy Gain**
von Neumann system	16.77	1
E-MAC	2.27	7.37

Table 3.13: Comparison with state-of-the-art techniques [22], [2]

	Energy (nJ/inference)	**Relatively Energy Gain**
von Neumann system	3866.78	1
[22]	994.85	1.89
[2]	1864.90	3.55
This Work	524.17	7.37

constraints (such as embedded systems and edge devices) or demands for real-time inference, we can prioritize energy efficiency by tolerating a negligible accuracy drop.

To provide a comprehensive evaluation that includes both resource requirements (energy) and performance (top-1 and top-5 accuracy), we can multiply the resource (energy) gains (Table 3.12) with the performance values (Table 3.11). This approach allows us to better evaluate and select suitable designs when both performance and resources have similar importance in the system. These combined gains are shown in fig. 3.36. The combined gains are determined by multiplying the energy gains from Table 3.13 with the corresponding performance measurements (top-1 and top-5 accuracies) presented in Table 3.11. Based on the results of our experiments, we observed that VGG16, trained on the ImageNet dataset and utilizing our proposed in-SRAM multiplier, provides high accuracy with energy efficiency.

In summary, we used the MNIST dataset for training and inference of our proposed LeNet-inspired CNN. Afterward, we quantized the trained CNN to the int-4 representation, achieving 99.01% accuracy on the quantized MNIST test data. The quantized CNN demonstrated a negligible impact on the distribution of the MNIST dataset, with minimal performance degradation. The proposed in-SRAM multipliers used for all multiplication operations during the inference phase achieved energy efficiency with negligible performance degradation.

Compared to a conventional von Neumann system and state-of-the-art in-memory multiplier accelerators, our proposed in-SRAM multiplier showed better energy efficiency and minimal accuracy degradation. The E-MAC achieved the minimum effect on the performance of the CNN while providing energy efficiency up to 7.37×.

Additionally, we evaluated the E-MAC on VGG16, which was trained using the challenging ImageNet dataset, containing over 1.2 million hand-annotated images and 1000 classes. Our proposed system and reported results lay a strong foundation for further research in the area of image classification systems.

Overall, using the E-MAC for implementing image classification algorithms provides an efficient and effective solution for image classification tasks, demonstrating high accuracy with energy efficiency. The results of our experiments open new avenues for future research in the field, with potential applications in healthcare, security, and robotics.

Chapter 4

Improving the Reliability and Energy Efficiency of STT-RAM Systems

Publication Acknowledgment

Parts of this chapter were published in a similar form in the author's previous publications as follows:

- S. Seyedfaraji, J. T. Daryani, M. M. S. Aly, and S. Rehman, "EXTENT: Enabling Approximation-Oriented Energy Efficient STT-RAM Write Circuit," in IEEE Access, vol. 10, pp. 82144–82155, 2022.
- S. Seyedfaraji, M. Bichl, A. Aftab, and S. Rehman, "HOPE: Holistic STT-RAM Architecture Exploration Framework for Future Cross-Platform Analysis," in IEEE Access, 2024.

In this chapter, we delve into the role of STT-RAM as a promising technology to address the memory wall, improving speed, scalability, and non-volatility. The chapter is divided into two main sections. First, we explore STT-RAM technology in depth, including the need for system evaluation frameworks, fundamental operation principles, and performance characteristics. This section also introduces a modeling framework, which serves as a crucial tool for evaluating STT-RAM's performance and efficiency, followed by a comparison with DRAM metrics to highlight STT-RAM's potential benefits and trade-offs.

In the second part, we focus on advancing energy efficiency within STT-RAM through approximation-oriented designs. We discuss the challenges of write reliability in STT-RAM and propose an approximation-based write driver aimeing at reducing write energy and latency. This section concludes with a cross-layer evaluation of the proposed design, assessing its effectiveness in improving performance while optimizing reliability concerns w.r.t. write error.

4.1 System-Level Exploration of STT-RAM Technology

STT-RAM offers multiple advantageous attributes such as non-volatility, high density, soft error resistance, compatibility with CMOS processes, high endurance, and scalability [9–12]. The International Roadmap for Devices and Systems (IRDS) [162] identifies STT-RAM as the leading candidate to supersede existing memory technologies.

In comparison (see table 2.1), the energy required for writing and reading in DRAM and SRAM per operation is typically lower than that for NVMs. However, as volatile memories, both DRAM and SRAM depend on continuous power to maintain data. DRAM, often used as the primary memory in computing systems, requires more energy due to the need for periodic refreshing to ensure data integrity. Similarly, SRAM needs constant power for data retention. In contrast, NVMs, such as Flash memory, retain data without power, which makes them more energy-efficient than DRAM and SRAM and a viable alternative for primary memory applications. PCM and STT-RAM [5] are among the prominent NVMs considered for replacing DRAM in main memory configurations. NVMs are also being investigated for their potential to expand on-chip cache capacities due to their high density. Among these, STT-RAM is particularly noted for its advances in on-chip computation capabilities and energy efficiency [163].

Nevertheless, the adoption of STT-RAM in broad industrial applications is constrained by several challenges, including delays in write operations and high energy consumption during write operations. To address these challenges, various strategies have been developed at different levels of abstraction, encompassing circuit-level techniques [85, 164–167], architectural solutions [13, 88, 168], and application-specific methods [169, 170].

However, while these strategies improve certain aspects of STT-RAM performance, they are not without limitations. Circuit-level techniques, for instance, often face trade-offs between energy efficiency and reliability, particularly under variable operating conditions, which can affect data retention and switching accuracy. Architectural solutions, though effective in managing energy at a system level, may not fully mitigate the inherent latency challenges associated with write operations, especially under high workload conditions. Furthermore, application-specific methods can offer targeted optimizations but are often limited in scope, potentially lacking generalizability across broader applications. As a result, there remain unresolved challenges in achieving a comprehensive solution to fully address the memory wall through STT-RAM technology alone.

4.1.1 Need for STT-RAM-based System Modeling

Significant strides have been made in refining comparison metrics at various abstraction levels, including application insights, architectural frameworks, and circuit design [7–10, 12–14, 16, 17, 166, 167].

Despite these advancements, current state-of-the-art techniques primarily rely on behavioral circuit models, which may not fully capture the intricate and precise behaviors of actual computing systems. This reliance on behavioral models highlights the limitations of existing evaluation methods and highlights the necessity for an extensive system evaluation framework.

A critical research question addressed in this thesis is: *how can we design a holistic system evaluation framework to evaluate the impact of incorporating STT-RAM memories in current systems, accurately modeling the scaling, energy consumption, and performance characteristics of these devices, and enabling comprehensive architectural design space exploration?*

Several simulation environments are utilized for the development and analysis of system-level exploration in computer architecture, notably gem5 and ZSIM [171]. The gem5 simulator is particularly favored for its comprehensive system emulation capabilities, aiding in the analysis of system-level metrics across various Instruction Set Architectures (ISAs), including Alpha, ARM, SPARC, MIPS, RISC-V, and x86, along with diverse timing and CPU modes [172,173]. Conversely, while ZSIM lacks full-system simulation capabilities, its non-event-driven execution model offers faster simulation speeds. Given the project's aim to integrate STT-RAM within a complete system and demonstrate the feasibility of running an OS on STT-RAM, gem5 has been chosen. It provides sufficiently rapid simulation speeds for benchmark applications running on an OS.

To fulfill the objective of creating a comprehensive system exploration framework, this thesis introduces, to the best of our knowledge, a novel memory interface incorporating STT-RAM, representing a significant advancement in the field. This interface has been meticulously developed and seamlessly integrated into the gem5 simulator, where it interacts with the existing memory controller to enable accurate and detailed system-level evaluations. This work unifies multiple contributions under a cohesive framework that collectively enhances our understanding of STT-RAM integration and its impact on system performance and efficiency.

- We introduce a modeling framework for STT-RAM modeling and simulation, seamlessly integrated into the gem5 full system simulator [4].
- We utilize the recently developed NVM interface in gem5 to merge HOPE with the current gem5 memory interfaces. This approach differs from previous methods that depended on external patches (like NVMain), which are less maintainable and hinder progressive development. Our framework adds a third memory interface dedicated to STT-RAM, providing highly detailed outputs akin to the current DRAM implementations in gem5. The integration of our framework into gem5 is straightforward, involving minimal modifications to existing gem5 files, as all new functionalities are contained in new files that integrate smoothly. This implementation operates similarly to existing memory interfaces and could potentially be adopted by the official gem5

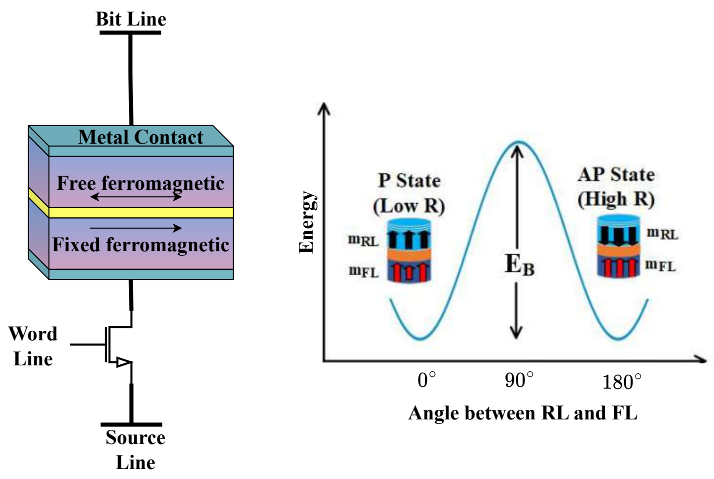


Figure 4.1: Schematic representation of magnetic orientation and energy barrier between two MTJ states [28]

repository's core maintainers, ensuring its future compatibility and relevance.

- Additionally, we have enhanced the gem5 power model, DRAM-Power, to accommodate our STT-RAM model.
- We assess the performance of HOPE using high-performance computing (HPC) applications from the SPEC CPU 2017 [174] benchmark suite on our event-driven gem5 simulator, effectively extracting evaluation metrics from both application and circuit perspectives.

4.1.2 STT-RAM Fundamental Operation Principles and Structure

To understand the integration of STT-RAM into system architectures, it is essential to first understand its fundamental structure and operation principles. Knowledge of STT-RAM structure provides insight into the physical mechanisms that drive its performance, reliability, and suitability for various applications.

The typical architecture of an STT-RAM cell consists of an MTJ unit (see 2.4a) paired with an access transistor, forming a $1T-1MTJ$ configuration. An MTJ cell is characterized by an oxide barrier flanked by two ferromagnetic layers known as the Rotation Layer (RL) (also referred to as the Free Layer) and the Fixed Layer (FL). The orientation of magnetization between these layers can manifest in two distinct states: parallel (P) and anti-parallel (AP), which correspond to logic states one and zero, respectively (see Fig. 4.1).

STT-RAM Write Operation Principle

A fundamental understanding of the STT-RAM write and read operation principles is essential for analyzing its energy consumption, latency, and overall performance in system applications. The write operation, in particular, directly impacts STT-RAM's suitability for various memory-intensive applications due to its energy requirements and reliability implications.

To encode data into the MTJ cell, a write current, denoted as I_{write}, must be applied across the memory cell. A successful write operation requires surpassing a minimum energy barrier, denoted as E_B, of the MTJ cell. If this energy threshold is met, the memory state can be altered according to the direction of the current passing through the cell (refer to Fig. 4.1). In a standard STT-RAM model, the necessary current to exceed this energy barrier can be described by the following equation:

$$I_c = I_{c_0}.(1 - \frac{1}{\Delta}\ln(f_0 t_p)) \tag{4.1}$$

where f_0 represents the attempt frequency, typically around $\sim$1 ns, and t_p is the operating pulse width, corresponding to an access frequency of $1GHz$. Δ represents the thermal stability factor, given by:

$$\Delta = \frac{M_s H_k t d}{2k_B T} \tag{4.2}$$

Here, $k_B T$ symbolizes the ambient thermal energy in the system due to thermal fluctuations [28]. Additionally, I_{c_0} is the critical write current for the STT-RAM model at 273° Kelvin, related to the intrinsic properties of the MTJ cell as:

$$I_{c_0} = \frac{8\alpha e M_s t}{\eta h \pi d^2} H_k \tag{4.3}$$

where M_s denotes the material's saturation magnetization, H_k the effective magnetic anisotropy field, α the damping factor, η the spin polarization, and t, d the physical dimensions of the MTJ cell.

These equations provide the foundation for understanding the energy and reliability challenges associated with STT-RAM write operations. Specifically, they highlight the dependencies of write current and stability on material properties and operating conditions. In this thesis, these principles are leveraged to evaluate the energy efficiency and performance trade-offs of STT-RAM within system-level architectures, serving as a basis for exploring optimization strategies. By analyzing these parameters, we can gain insights into how variations in material and structural attributes impact write energy and latency, which is crucial for developing more energy-efficient and reliable STT-RAM designs.

STT-RAM Read Operation Principle

The reading process in the MTJ cell involves the passage of a current, I_{read}, through the cell. This current must be lower than $I_{critical}$ to avoid altering the cell's state. The resistance of the MTJ depends on the magnetization orientation of its two layers, represented as either 0° or 180° (see Fig. 4.1). By sensing the resistance while passing I_{read}, the state of the cell can be determined. Furthermore, the Tunnel Magnetoresistance Ratio (TMR), defined as:

$$TMR = \frac{(R_{AP} - R_P)}{R_P} \tag{4.4}$$

where R_{AP} and R_P represent the resistance in anti-parallel and parallel states respectively, quantifies the difference between these resistance states. A higher TMR is indicative of quicker and more accurate read operations due to decreased read latency [28].

This thesis primarily focuses on aspects of STT-RAM technology that impact write energy efficiency, latency, and system performance under real-world applications. While the read operation is essential to STT-RAM's overall functionality, the main focus here is on optimizing write operations due to their significant role in determining the energy and latency characteristics crucial for system-level deployment. This brief discussion on read operations provides necessary context, but a detailed exploration of read-specific reliability issues is beyond the scope of this work, which is directed toward energy and latency improvements in write operations.

4.1.3 System level modeling and Evalution framework

Recent studies have identified NVM as a pivotal memory element in simulations using the gem5 simulator, which lacked a dedicated NVM interface until October 2020. Prior to this integration, NVM research was contingent on external tools such as NVSim [175], NVMain [176], and NVMain 2.0 [177].

For seamless integration and to maintain compatibility, these NVM simulators needed concurrent development with the gem5 simulator. Notably, NVMain provides a patch that integrates it with the gem5, modifying the simulator's source code. This patch must be regularly updated to align with new gem5 releases, creating a co-simulator environment where gem5 and NVMain interact continuously. Even minor updates in gem5 can necessitate adjustments in the NVM simulation tools and their patches. The latest official patch for integrating NVMain into gem5 was last updated in December 2016, rendering it incompatible with newer versions of gem5. Additionally, the lack of accessible open-source models for state-of-the-art STT-RAM gem5 tool flows has restricted research and experimentation, impacting the integration of STT-RAM in contemporary systems.

To address these limitations, this thesis introduces a comprehensive, open-source framework, HOPE, based on gem5, which directly supports the modeling of STT-RAM. The HOPE framework enables researchers to evaluate STT-RAM's energy efficiency, latency, and scalability under real workload conditions, directly contributing to the thesis's goals of optimizing STT-RAM for memory wall mitigation.

This framework not only provides a foundation for analyzing STT-RAM's power and performance trade-offs but also facilitates the exploration of approximation-oriented techniques aimed at reducing write energy, as outlined in our research questions. By integrating HOPE with gem5, this work equips researchers with a scalable tool to address current and future challenges in STT-RAM energy efficiency and reliability, promoting advancements in NVM technology within modern system architectures.

The study presented in [178] examines architectural modifications of STT-RAM structures to provide NVM-based row buffers, achieving a 67% energy reduction compared with existing techniques. Additionally, research in [170] investigates replacing DRAM with STT-RAM in main memory, using the SPEC CPU 2006 dataset for comparisons with DRAM-based systems. This evaluation was performed using a trace-based cycle-accurate simulator. In [179], a novel STT-RAM-based memory architecture employing a 9F2-cell at the circuit level was proposed, featuring an MTJ model that requires low switching current for state transitions. Application-level analyses were conducted using the HPC SPEC CPU 2017 benchmark [174] to assess latency improvements. Existing techniques, such as NVSim, generally perform time-intensive circuit-level simulations, while system-level application evaluations remain proprietary and are not widely available as open-source, thereby hindering broader adoption of the STT-RAM model. *Our innovative HOPE framework integrates a complete architectural model of STT-RAM into the gem5 simulator with a memory controller to exploit system-level characteristics, contrasting with previous methods that depend on external patches like NVMain. HOPE is an event-driven gem5 framework that supports system-level evaluations and facilitates the extraction of comparative metrics across all system hierarchy layers.*

The innovative STT-RAM interface introduced in this thesis serves as an additional memory interface within the gem5 simulator. To ensure compatibility with gem5's MemCtrl component, minor modifications to the MemCtrl are necessary. These adjustments are designed to be non-intrusive, preserving the existing functionality for connecting DRAM and NVM memories.

Historically, the MemCtrl component in the gem5 system configuration offered a single port named *DRAM*, which was utilized for all memory types. In this work, we have implemented the STT-RAM interface as an alternative to the existing DRAMInterface and NVMInterface, as depicted in Fig. 4.2. The implementation of the interface is done in C++ (Component functional description), complemented by a Python wrapper for parameter configuration (part of Component definitions). This Python wrapper is essential as it defines and inherits the parameters necessary for the component's operation. The core functionality of the STTDDR4Interface is encapsulated within its C++ class. The component STT_1333_4x16, used as the test device, is configured using parameters sourced from the EMD4E001G16G2 datasheet [180]. Our framework facilitates direct modifications to the memory settings through its integrated interface, allowing for the retrieval of output data directly within our component definition. This

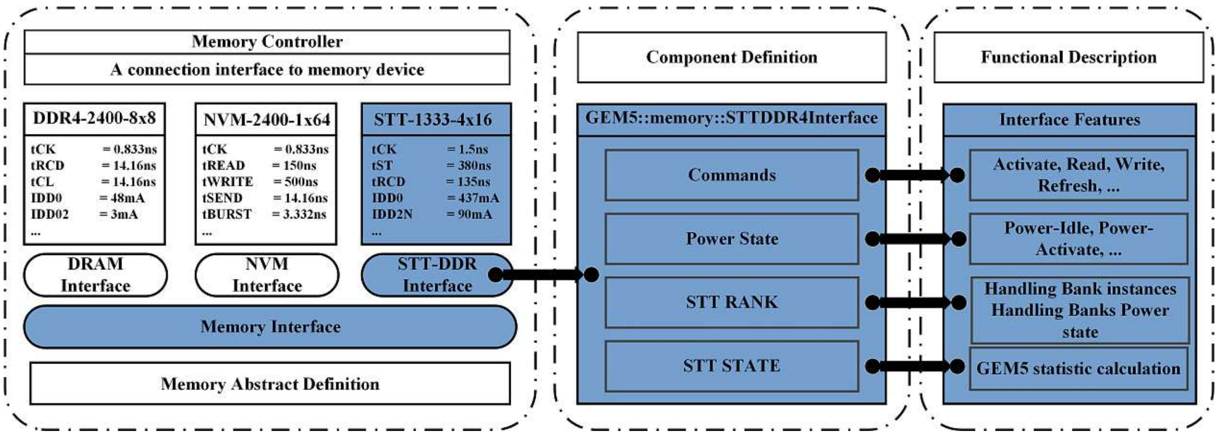


Figure 4.2: The STTDDR4 Interface integration into the gem5 standard components library. Components in blue are modified or new to gem5, and components in white are unmodified gem5 components.

capability enables researchers to perform accurate, system-level evaluations of STT-RAM performance, energy consumption, and reliability under realistic workloads, addressing core research questions of this thesis, such as the scalability and energy efficiency of STT-RAM in large-scale applications. By integrating all functionalities into the STT-RAM memory controller via the provided interface, our framework enables detailed exploration of approximation techniques and reliability optimizations at various levels of abstraction. This unified approach represents a significant advancement over previous methods that relied on co-simulation with external tools like NVSim and NVMain, as it allows for comprehensive, accessible, and iterative testing directly within gem5. Consequently, the HOPE framework provides essential insights into the design trade-offs necessary to optimize STT-RAM for mitigating the memory wall, aligning with the thesis's goal of advancing energy-efficient, high-performance memory solutions.

HOPE STT-RAM Power Model:

The implementation of the STTDDR4 memory interface is designed as a state machine, comprising states for idle, storing, activating, powering up, and powering down, as illustrated in Fig. 4.3. The MemCtrl instance in the simulated system facilitates the transitions within this state machine. Upon system initialization, the PWR_IDLE state is the default starting point. From this state, transitions to either the active state PWR_ACT or the active with store state PWR_ACT_ST are possible through the use of ACT or ACT_ST commands, respectively. The ACT_ST command, a novel addition to gem5, manages STT-RAM-specific data operations within the volatile page buffer.

Standard gem5 commands are utilized for manipulating main memory, such as ACT (activate), RD (read), WR (write), REF (refresh all banks), SREF (Self-Refresh), and PRE (explicit pre-charge of a single bank). This thesis introduces an additional command, ACT_ST (activate with store), known as ACT*

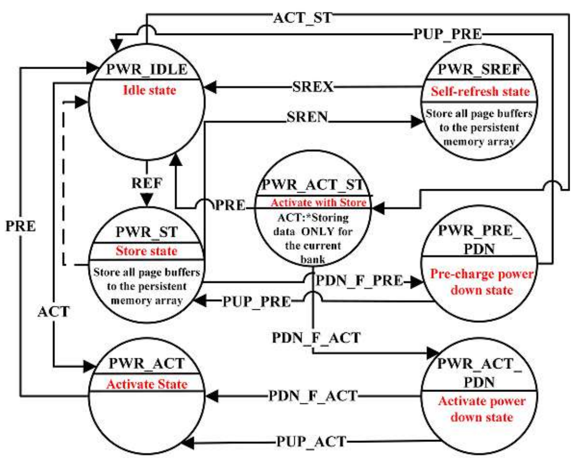


Figure 4.3: Power state machine of STTDDR4 integration to gem5

Algorithm 1 Select between ACT and ACT_ST command on rank:bank:row

```
 1: Initialize: bank.storingState ← PERSISTENT
 2: bank.lastRow ← 0
 3: procedure ActivateBank($rank, bank, row$)
 4:     $cmd \leftarrow ACT$
 5:     if $bank.lastRow \neq row$ then
 6:         if $bank.storingState = BUFFER$ then
 7:             $cmd \leftarrow ACT\_ST$
 8:             $cmdDelay \leftarrow cmdDelay + tST$
 9:         end if
10:     end if
11:     $cmdList.push\_back(cmd, bank, delay)$
12:     $bank.lastRow \leftarrow row$
13:     $bank.storingState \leftarrow BUFFER$
14:     process $cmd$ in drampower
15: end procedure
```

in the EMD4E001G16G2 datasheet [180]. The ACT* command facilitates a storage procedure for the accessed bank. The integration of the ACT_ST command into gem5 includes the creation of a new event, *actStoreEvent*, which supports the transition to the novel power state PWR_ACT_ST. Unlike in DRAM systems, the automatic execution of REF commands, necessary for data persistence, is omitted in the STTDDR4 model because STT-RAM devices, as per the specifications of the EMD4E001G16G2, do not require refreshes.

Method calls play a crucial role in facilitating MemCtrl interactions with the memory interface during simulations. For instance, when reading or writing data, the MemCtrl initiates a burst access to the memory device. During this process, the MemCtrl specifies the rank, bank, and row where the burst should be executed. This critical information is relayed to the bank activation method, as depicted in Algo. 1. The EMD4E001G16G2 device, specifically, includes a feature that automatically stores page memory data in the persistent memory array to prevent data loss. However, it is important to note that the MemCtrl lacks the functionality to differentiate between the storing states in STT-RAM, which can be a limitation in managing memory states more precisely.

The STTDDR4Interface has been enhanced to include functionalities that monitor the storing state of data in the page buffer of each bank. These states are classified as BUFFER and PERSISTENT. Initially, all banks are set to PERSISTENT at startup, indicating that the data in the page buffer is to be saved to the persistent memory array. Conversely, the BUFFER state indicates that the bank's data has not yet been saved to the persistent memory array. Additionally, the last row accessed in each bank is recorded and updated with each bank activation. To switch the storing state of a bank, or all banks, to PERSISTENT, a store operation must be initiated. This can be triggered by an ACT_ST command, a REF command, or an SREF command. During a REF or SREF operation, store operations are executed on all banks in the BUFFER state. If no banks are in the BUFFER state, the commands REF and SREF will not trigger any action.

The decision to use either the ACT or ACT_ST command is made within the activated bank method, as outlined in Algo. 1. The requested row is compared to the last accessed row of the specific bank. If the last accessed data is still active, indicating the last accessed row and the requested row are the same, no store operation is necessary, and a normal ACT command is simulated. If the requested row differs from the last accessed row and the bank is in the BUFFER state, an ACT_ST command is initiated to simulate the additional store operation, which incurs a higher delay known as the store time (tST). This store operation also affects energy consumption, calculated in the power library. A bank's storing state is updated to PERSISTENT following a successful store operation on that bank or across all banks.

As with DRAM memory in gem5, the ACT command is handled by listing it among pending commands, which are then processed by the modified DRAMPower tool [181]. This tool is part of gem5 and calculates energy based on gem5 inputs.

Extensions to DRAM Power Model:

The DRAM power tool has been extended to include the ACT_ST command and associated energy calculations. These enhancements are detailed in Algo. 2, incorporate functions to count the number of store procedures during runtime and calculate the resulting energy and power usage.

Additionally, energy calculations in gem5 are updated not with every command execution but during specific simulation phases: upon suspension, at the end of a refresh (REF) command, or triggered from the gem5 system configuration script or the internal command line of the full system simulation using the m5 utility (the utility used in full system disk images).

Furthermore, the gem5 statistical output has been modified to include metrics such as store energy per rank and power state time per rank in the simulation results. This section introduces our HOPE framework, an integrated STT-RAM solution within the gem5 simulator utilizing the memory controller, enabling the examination of system-level meta-heuristics like power consumption, memory utilization, and thermal patterns.

Algorithm 2 DRAM Power Extension

```
Initialize: nOfActsBanks[banks] ← zeros(banks)
nOfStoresBanks[banks] ← zeros(banks)
procedure EVALUATECOMMANDS($cmdList$)
    for all $cmd \in cmdList$ do
        if $cmd.type = ACT$ then
            $handleAct(cmd.bank)$
        else if $cmd.type = ACT\_ST$ then
            $handleActSt(cmd.bank)$
        else if <other command types> then
            <handle commands>
        end if
    end for
end procedure

procedure HANDLEACTST($bank$)
    if isPrecharged(bank) then
        $nOfActsBanks[bank] += 1$
        $nOfStoresBanks[bank] += 1$
    end if
end procedure

procedure POWER_CALC()
    $calc(sum(nOfStoresBanks) \times tST, idd0 - idd3n)$
    for all $bank \in banks$ do
        $calc(nOfStoresBanks[bank] \times tST, idd0 - ione)$
    end for
end procedure

procedure CALC($cycles, current$)
    return $(cycles \times clkPeriod \times current \times voltage)$
end procedure
```

HOPE Case Study Configurations:

The gem5 instruction-level simulator has played a crucial role in enabling the integration of diverse memory technologies through its memory controller, which has seen substantial advancements over recent years. A significant milestone in this evolution was the introduction of a revised memory controller (MemCtrl) in May 2020, accompanied by a new NVM interface class, NVMInterface, officially incorporated into gem5 version 20.1. Designed as a generic framework, the NVMInterface supports key timing parameters, including tREAD, tWRITE, and tSEND, facilitating basic NVM memory simulations.

While the existing DRAM interface (DRAMInterface) in gem5 provides a highly detailed logic for DRAM timing and power state simulations—accounting for parameters such as tRCD, tCL, tRAS, and tREFI, along with energy metrics like IDD0, IDD4R, and VDD—such detailed modeling is absent in the NVMInterface. This limitation restricts the simulation accuracy of STT-RAM and other NVM technologies, as it lacks essential power and energy calculation capabilities.

To bridge this gap, the HOPE framework introduces an advanced memory interface specifically tailored for STT-RAM. Built upon DDR4 technology, this interface—termed STTDDR4Interface—integrates intricate timing and energy parameters while incorporating logic for power state and energy calculations. By doing so, it provides a comprehensive platform for evaluating STT-RAM performance at a system level. Fig. 4.2 illustrates the architectural flow of this enhanced framework within the modified gem5 simulator.

The simulation setup is configured using the `fs.py` system configuration script, which initializes a *system* instance based on the HOPE framework specifications. An x86-based architecture is employed with a TimingCPU to ensure precise memory timing measurements. This system is equipped with L1 and L2 caches and operates using a Linux kernel with a disk image containing the Ubuntu OS and SPEC CPU 2017 workloads. Communication within gem5 follows real-system interactions via ports, where the system block connects to the Membus, utilizing the SystemXBar as the memory bus by default. All CPU-memory interactions occur through the Membus to the memory controller, which has been modified to support STT-RAM via its memory port.

Fig. 4.2 depicts the integration of STT-RAM within the memory hierarchy through a dedicated class, STT_1333_4x16, designed for seamless interaction with the memory controller. This class defines essential parameters such as tCK, tST, tRCD, IDD0, and IDD2N, which are derived from the device's specifications. The tCK parameter denotes the clock period, calculated based on the operating frequency (e.g., at 667 MHz, tCK is approximately $1.5ns$, obtained from $1/fCK$). A unique parameter introduced for STT-RAM in gem5 is tST, which represents the time required to store data from the row address buffer to the persistent memory array. This distinction allows researchers to optimize store operations and evaluate Processing-in-Memory (PIM) strategies aimed at minimizing tST to enhance performance. Additionally, conventional DDR4 parameters such as tRCD, IDD0, and IDD2N are maintained for comparative analysis.

The incorporation of this enhanced interface enables more accurate simulations of modern STT-RAM devices, facilitating in-depth analysis of power states and energy consumption—capabilities previously unattainable within a standalone gem5 environment. This integration follows the specifications of the EMD4E001G16G2 STT-RAM module from Everspin Technologies [180], with key device characteristics summarized in Table 4.1.

One of the key distinctions between STT-RAM and conventional DRAM within this framework is the elimination of the Refresh (REF) command. Unlike DRAM, which mandates periodic refresh operations at intervals defined by tREFI, STT-RAM does not require refresh cycles due to its non-volatile nature. Consequently, the tREFI parameter is removed from STT-RAM simulations, reducing

Table 4.1: Memory configuration for the simulated STT-RAM device

Parameters	Configuration
Memory Size	1Gb x16
Organization	8 banks (2 banks per Bank Group)
Operating Frequency	667 MHz (1333 MT/s)
Access Time	225 ps
Min Supply Voltage	1.14 V
Max Supply Voltage	1.26 V

overhead and simplifying control logic. However, STT-RAM introduces a store operation delay, tST, which accounts for the time required to transfer recently written data from the page buffer to the persistent memory array.

Further deviations from DDR4 specifications include variations in memory size and operating frequency. While DRAM devices under DDR4 standards typically offer capacities of 2, 4, 8, and 16Gb, the simulated STT-RAM device is constrained to 1Gb. Additionally, its operating clock frequency is limited to 667 MHz, whereas DDR4-compliant DRAM devices operate within the 800-1200 MHz range. These distinctions highlight the unique design constraints of STT-RAM and underscore the necessity of tailored architectural optimizations.

By integrating these modifications, the HOPE framework provides a more robust and accurate methodology for evaluating STT-RAM in real-world computing environments. This enhanced simulation capability enables researchers to explore design trade-offs, assess energy efficiency, and optimize system-level performance, ultimately contributing to the advancement of next-generation non-volatile memory solutions.

HOPE Setup Configuration

Fig. 4.4 provides an overview of the HOPE framework's setup, with its main features shown in blue. The setup involves using a disk image that includes an OS and a kernel compatible with the OS. This image is created using a modified Packer SPEC CPU 2017 [174] script from the gem5 resources, which installs the Ubuntu OS and the SPEC CPU 2017 benchmarks for x86 architecture on the disk image (①). After the benchmarks are installed, the disk image is mounted on the host system (②). Each benchmark is compiled, trained, and executed once to finalize the installation. The gem5 full system simulation, enhanced with HOPE-specific extensions and modifications, uses this disk image (③). Simulations run selected workloads from the disk, create checkpoints post-OS boot, and produce detailed statistics at the end (④). The simulation integrates the STT-RAM extension and modifies the DRAMPower for precise energy calculations. The modified McPAT template file supports the output extensions of gem5 (⑤), and the GEM5ToMcPAT tool [182] uses these files to create inputs for McPAT analysis (⑥).

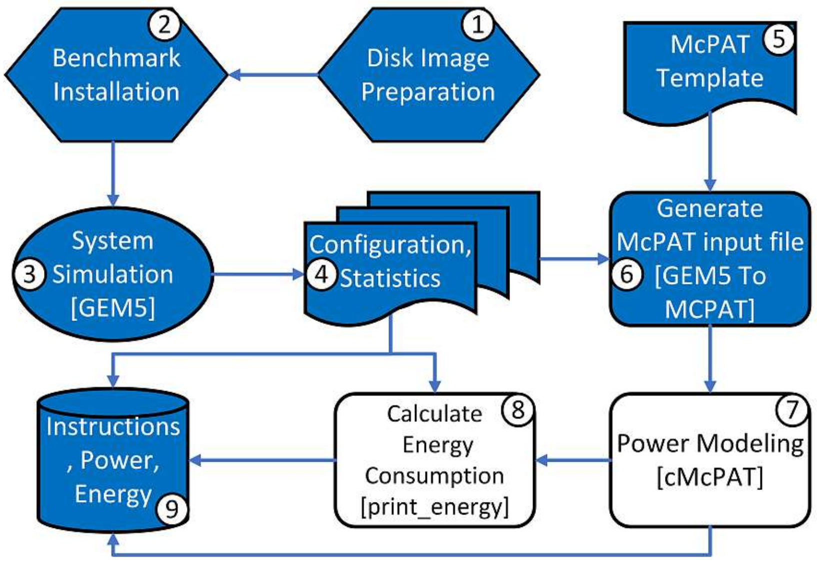

1 Disk Image Preparation
2 Benchmark Installation
3 System Simulation [GEM5]
4 Configuration, Statistics
5 McPAT Template
6 Generate McPAT input file [GEM5 To MCPAT]
7 Power Modeling [cMcPAT]
8 Calculate Energy Consumption [print_energy]
9 Instructions , Power, Energy

Figure 4.4: An overview of HOPE framework, along with the contributions in blue

Furthermore, HOPE includes the enhanced cMcPAT framework for power, area, and timing modeling, which processes the power parameters (⑦). Finally, the script ⑧ "print_energy" [182] computes the total energy usage based on gem5 outputs, which include instruction counts among other detailed statistics.

The simulated system is set up with the gem5 standard library (gem5 stdlib) for an x86 configuration. Detailed specifications for the processor, cache, and memory configurations for both experimental systems using STT-RAM and DRAM are presented in Table 4.2. We have chosen an existing STT-RAM device for our simulations, paired with a commonly used DRAM device to allow functional comparisons. It is crucial to mention that the selected STT-RAM device does not conform to the JEDEC JESD79-4A DDR4 standard, which restricts our choice of a DDR4-based DRAM device with similar parameters.

Additionally, Fig. 4.5 illustrates the system configuration for the benchmarks using STT-RAM. We utilized the general full system default configuration script provided by gem5 (fs.py) to accurately represent the architecture of our simulated system. The simulations were conducted in gem5's full system emulation mode to best mimic real-world system behaviors. The operating system selected

Table 4.2: Systems configuration for STT-RAM and DRAM

System elements	Processor	L1 Instr. cache	L1 Data cache	L2 cache	Main Memory	Clock Speed	Row Buffer Size	Device Size	Channel Cappacity	tRCDmin	tRCmin	tRASmin	tFAWmin	tRPmin
STT-RAM	64-bit x86 single core, timing 3GHz	Private, 32kB	Private, 32kB	Shared, 256kB	1 channel, 2 ranks per channel, 4 chips per rank, EMD4E001G16G2, 1Gb x16, 1333MHz	667MHz	256B	1Gb	8Gb	135ns	44.5ns	32ns	15ns	12.5ns
DRAM	— " —	— " —	— " —	— " —	1 channel, 2 ranks per channel, 8 chips per rank, MT40A1G8SA, 8Gb x8, 2400MHz	1200MHz	1kB	8Gb	128Gb	12.5ns	190ns	143ns	240ns	7.5ns

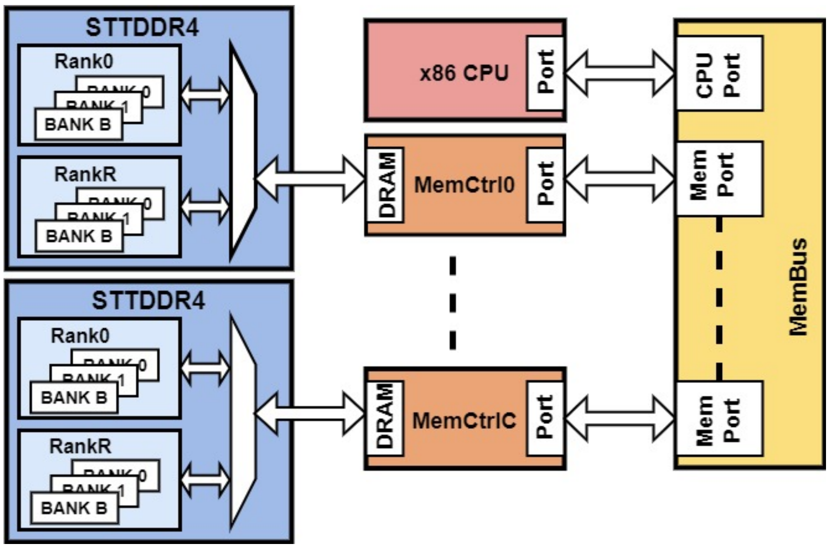


Figure 4.5: Architecture of the STT-RAM simulated system.

is Ubuntu 18.04, specifically configured for gem5 and equipped with the SPEC CPU 2017 benchmark suite [174]. We employ the Linux kernel version 4.19.83. The gem5 configuration script is responsible for setting up memory controllers and devices, with memory rank and bank counts specified in the memory device configuration. The parameters for the STT-RAM device configuration in gem5 are derived from the EMD4E001G16G2 datasheet [180].

We conducted SPEC CPU 2017 benchmarks on our simulated systems, creating two checkpoints per benchmark to enable detailed simulations with gem5's TimingCPU. The first checkpoint is captured after the OS boots completely, while the second is recorded after the first 4.5 billion instructions of the benchmark application to ensure the initialization phase is complete. This setup positions the simulation to directly enter the main algorithm of the benchmark. Both checkpoints are created using gem5's fast-forward method with AtomicSimpleCPU. From the second checkpoint, the main simulation run proceeds for a total of 2 billion instructions. This process is replicated for all benchmark applications included in the SPEC CPU 2017 suite.

Experimental Setup and Results:

In our research, we primarily focused on simulating real-world device parameters, particularly a selected STT-RAM device, which is DDR4-compatible despite some deviations from the official standard. These deviations significantly impacted our results, which are thoroughly documented. Our analysis using the HOPE framework highlighted the potential of STT-RAM for main memories in practical systems, offering insights into performance and energy efficiency with the SPEC CPU 2017 benchmarks. This evaluation not only highlights the potential of STT-RAM but also advances research in memory technology optimization.

The gem5 simulator facilitates versatile system configuration via its standard library, which includes CPUs, memory units, boards, and cache hierarchies. Users can customize components such as modifying CPU architectures by selecting specific boards (e.g., x86 Board or ArmBoard) and adjusting memory configurations. Additionally, systems can be configured without predefined boards by manually linking CPUs to memory devices via a memory bus and opting for either simple or complex cache setups. Our framework employs fs.py for system configuration, allowing quick modifications through command-line inputs to suit various architectures, with examples like $configs/example/riscv/fs\_linux.py$ for RISC-V or $configs/example/arm/fs\_bigLITTLE.py$ for ARM. We utilized both fs.py and custom scripts for a comprehensive analysis of our simulations.

STT-RAM State Time Distribution

As shown in Fig. 4.8(b), the IDLE state power times in STT-RAM vary depending on the workload. A key advantage of STT-RAM is its elimination of the need for periodic data refreshes, resulting in no time spent in the REF power state. In contrast, the bank activation (ACT) times, as depicted in the same figure, show only slight variations within the STT-RAM-based system. These variations are primarily due to extended delays during the bank activation process, particularly in the store state (ACT_ST) for STT-RAM. These observations provide critical insights into the dynamic behavior and efficiency of STT-RAM, offering valuable knowledge for enhancing real-world implementations and optimizations of memory technologies.

4.1.4 STT-RAM Performance Evaluation

Fig. 4.6(a) illustrates that STT-RAM devices typically experience higher read and write request volumes. This observation is further elaborated in Fig. 4.8(a), which highlights the row hit rate for read operations and suggests that these rates vary depending on application complexity. Additionally, Fig. 4.8(c) presents the average read and write bandwidth for STT-RAM. Moreover, as shown in Fig. 4.6(b), the average latency for each application is closely linked to the hit ratio. Our STTDDR4 power state machine description indicates that frequent changes in accessed rows can significantly affect both the energy consumption and latency of STT-RAM, impacting store operations as well. This comprehensive analysis highlights the performance dynamics and potential optimization areas for STT-RAM in memory technology applications.

4.1.5 STT-RAM Power and Energy Breakdown

Fig. 4.6(c and d) provides a detailed breakdown of power and energy for our simulated systems, revealing insights into their performance attributes. These figures display the accumulated energy for various commands issued to the memory device. Specifically, "Activation energy" accounts for the total energy consumed by all ACT commands, while "Store energy" aggregates the energy of all store operations. For ACT_ST operations, the energy contributions from activation and store are tallied separately. Additionally, "Refresh energy" corresponds to REF commands, and "RD/WR energy" encompasses the energy during read and write bursts.

A key observation is the high number of store operations coupled with a notable IDD0 current of 437mA, significantly impacting the store energy calculations. Notably, STT-RAM eliminates the need for refresh energy, contributing to its overall energy efficiency. However, the substantial energy demands of store operations raise the total energy expenditure. These insights highlight the distinct energy consumption patterns of STT-RAM compared to traditional memory technologies, indicating potential for more energy-efficient computing approaches. While the energy parameters shown in d represent only a subset of the total calculated values in gem5, they include critical metrics like interface energy, standby energy, self-refresh energy, and the energy consumed during power-down and power-up phases, providing a foundation for optimizing memory architectures towards energy-efficient computing systems.

4.1.6 DRAM Metrics

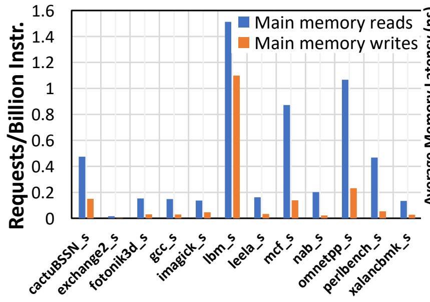
Our framework retains a versatile interface to DRAM, enhancing its adaptability across various memory technologies and computing scenarios. This section delves into an in-depth analysis of DRAM based on state time distribution, power consumption, energy usage, and latency for SPEC CPU 2017 applications, showcasing how DRAM functions as the primary memory in our system.

Fig. 4.9(b) delineates the state time distribution for these applications with DRAM. The memory request patterns, which provide insights into data locality and cache utilization, are displayed in Fig. 4.7(a). Bandwidth utilization, a key indicator of memory system efficiency and potential bottlenecks, is detailed in Fig. 4.9(c). Row hit rates are illustrated in Fig. 4.9(a), and latency metrics per application are presented in Fig. 4.7(b). Additionally, average power and energy usage while running the SPEC CPU 2017 applications are exhibited in Figs. 4.7(c and d). These comprehensive metrics not only underscore the operational efficiency of our framework with DRAM but also highlight its potential to adapt to future memory technologies and diverse computing environments.

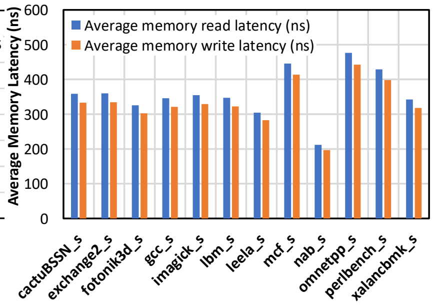
4.1.7 Comparing STT-RAM and DRAM Results

Our evaluation of SPEC CPU 2017 benchmarks indicates that while STT-RAM holds promise as a non-volatile alternative to DRAM, it currently faces significant challenges in replacing DRAM-based main memory for many applications. The primary bottlenecks stem from higher store latency and increased energy consumption, which hinder its adoption in high-performance computing environments. Unlike DRAM, where store operations are handled efficiently, STT-RAM introduces additional overhead due to its requirement to store data from the page buffer to the persistent memory array. This results in significant delays, particularly in the ACT-ST state, where each store operation incurs an additional 380 ns. Furthermore, STT-RAM operates at a lower frequency (1333 MHz compared to DRAM's 2400 MHz), further exacerbating its performance limitations in write-intensive workloads such as *Ibm_s* (see Fig. 4.6a).

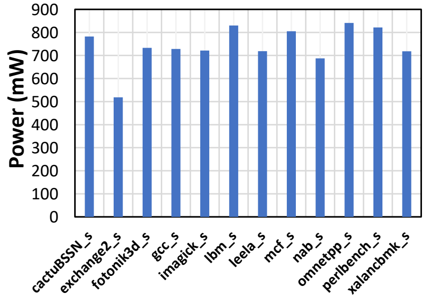
Addressing these challenges necessitates technological and architectural optimizations to enhance STT-RAM's efficiency and make it a viable competitor to DRAM. While advancements in device-level optimizations, write reduction techniques and circuit-level improvements can mitigate some of these bottlenecks, a holistic system-level evaluation remains critical. This is where the HOPE framework provides a strategic advantage, offering a systematic methodology to analyze, optimize, and refine STT-RAM architectures. By leveraging this tool, we can explore architectural configurations, identify potential performance trade-offs, and develop optimization strategies that improve latency, energy efficiency, and overall system performance. Such refinements are essential for positioning STT-RAM as a feasible alternative to DRAM in future memory hierarchies.



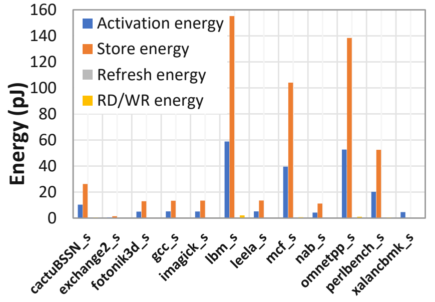
(a) R/W requests/billio. instr.



(b) Average memory latency (ns)

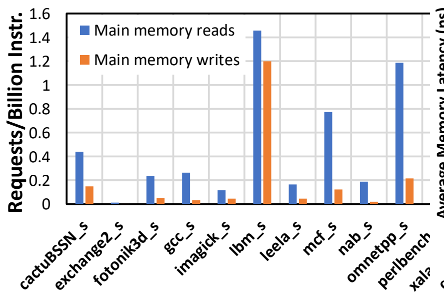


(c) Power consumption (mW)

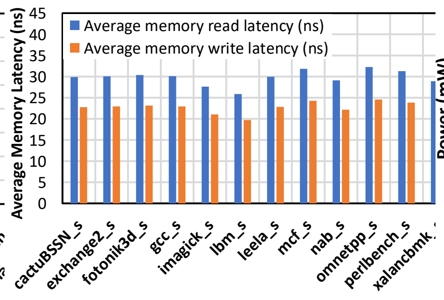


(d) Energy usage (pJ)

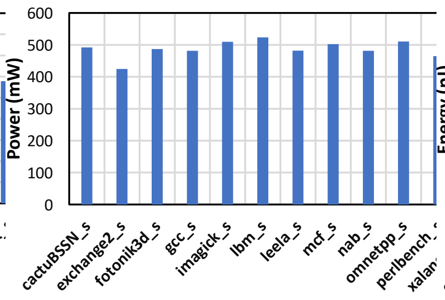
Figure 4.6: STT-RAM evaluation metrics for different application



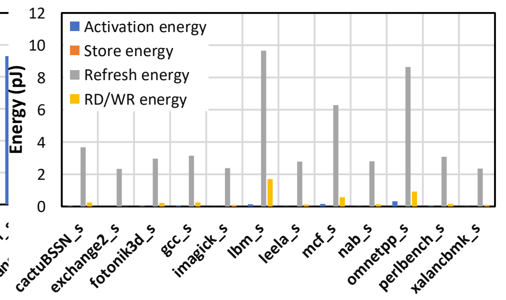
(a) R/W requests/billio. instr.



(b) Average memory latency (ns)

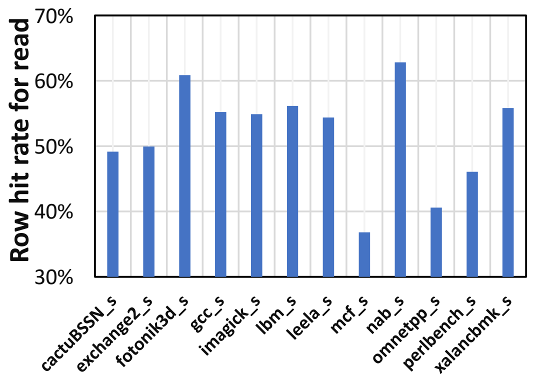


(c) Power consumption (mW)

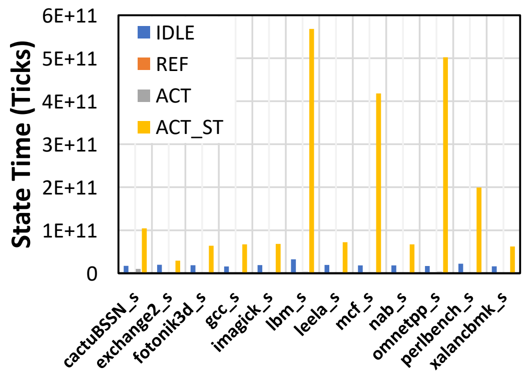


(d) Energy usage (pJ)

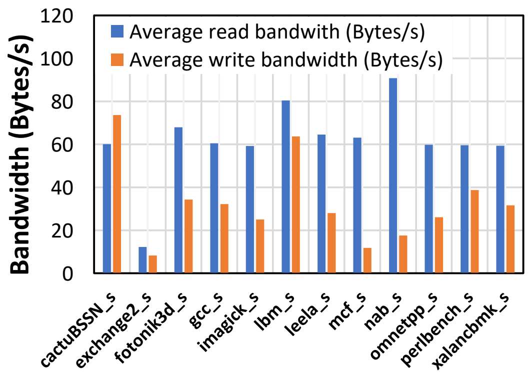
Figure 4.7: DRAM evaluation metrics for different application



(a) Read operation row hits (%)

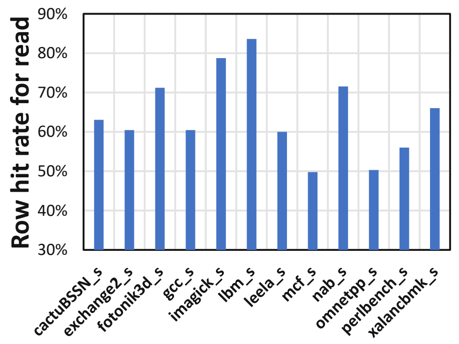


(b) State delays (# ticks)

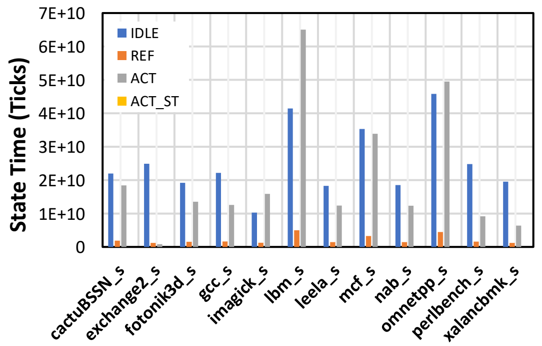


(c) Average R/W bandwidth (Bytes/sec)

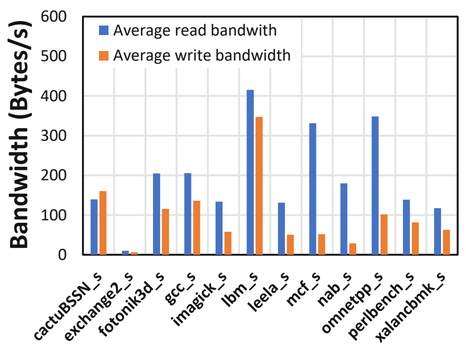
Figure 4.8: STT-RAM row hits, state time, and bandwidth usage



(a) Read operation row hits (%)



(b) State delays (# ticks)



(c) Average R/W bandwidth (Bytes/sec)

Figure 4.9: DRAM row hits, state time, and bandwidth usage

4.2 Enabling Approximation-Oriented Energy Efficient STT-RAM Write Circuit

STT-RAM has garnered interest due to its non-volatility, low leakage power, and high density. Its magnetic properties play a crucial role in STT-RAM switching operations through thermal effectiveness. However, a significant challenge for the industrial adoption of STT-RAM is the high write energy and latency. In this section, we address this challenge by leveraging the stochastic switching activity of STT-RAM cells along with circuit-level approximation, a method that introduces trade-offs between power efficiency and data integrity. To enhance the robustness of this approach, we analyze the vulnerability of write operations to radiation-induced soft errors and implement a low-cost improvement that mitigates potential data integrity issues.

STT-RAM is highly sensitive to temperature, and even slight changes in room temperature can significantly impact the magnetic rotation process, as studied in [183]. Thermally-assisted switching, or Thermal-Controlled Magnetic Anisotropy (TCMA), presents a trade-off between write time and write energy [184]. Addressing the thermal effects on the switching process can improve both write energy and write time. Therefore, this paper considers current-driven switching in conjunction with STT-RAM.

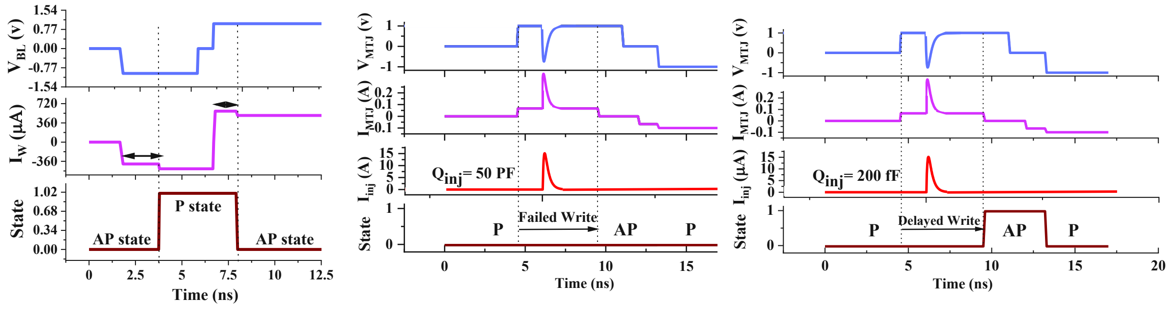
For the industrial adoption of STT-RAM, several critical challenges need to be addressed, particularly regarding write energy and latency [184–187]. To successfully complete a write operation, two potential solutions are: 1) applying a continuous current with a consistent amplitude for the required duration, or 2) increasing the current amplitude to accelerate the writing process [187]. However, both approaches result in excessive power overhead in the circuit. To address this, we propose the EXTENT technique, which enables power savings through circuit-level approximation while balancing write energy, retention time, and data reliability under variable conditions.

4.2.1 Reliability Challenges in STT-RAM

STT-RAM cells are vulnerable to radiation-induced soft errors, compromising data integrity in environments exposed to high radiation, such as aerospace and medical devices. Process variability during manufacturing can lead to inconsistencies in device performance, affecting reliability and predictability in STT-RAM-based systems. Additionally, retention failures pose a challenge, especially under varying environmental conditions, as does the endurance of STT-RAM cells, which can degrade over repeated write and erase cycles. Magnetic interference from external sources or adjacent cells can also disrupt STT-RAM operation, leading to errors. Addressing these challenges is crucial for the widespread adoption of STT-RAM in mission-critical applications such as autonomous vehicles, satellite communication systems, medical devices, and industrial control systems, where data integrity and durability are essential. The following subsections will delve into the issues of write energy and radiation-induced problems, and our proposed optimization techniques to address these issues.

- **Write error rate and thermal stability factor:**
  In order to replace SRAM or DRAM in future computing systems, read and write operations in STT-RAM must be reliable, energy-efficient, and have minimal delay. The current direction through the MTJ determines the stored data as “1” (logic-one) or “0” (logic-zero). The stochastic nature of the write operation in STT-RAM means different current densities and durations are needed for successful writes. The P to AP transition (writing logic-one) takes more time, as shown in Fig. 4.10, 4.10b and 4.10c. This is due to the current direction passing through the MTJ; when electrons flow from the FL to the RL, they quickly align the RL dipoles due to a strong magnetic field. Conversely, in the opposite direction, electrons encounter a weak magnetic field first, requiring more current density or time to reverse the RL dipoles and write the desired data into the MTJ. Consequently, the write failure probability for $P \rightarrow AP$ transitions is much lower than for $AP \rightarrow P$ transitions [87, 188–191, 191]. Increasing write energy compensates for the delay and vice versa. The stochasticity of the write operation is further complicated by stochastic thermal effects. The Thermal Stability Factor (TSF), denoted by Δ, is crucial as it affects data retention capability. A higher energy barrier in the MTJ stabilizes magnetization, resulting in a longer retention time. For an in-depth understanding and accurate prediction of low Write Error Rate (WER), micromagnetic effects must be considered. Previous WER calculations, considering these effects, have been conducted using 64 and 10^3 independent stochastic simulations [188, 192, 193]. The thermal dependence of WER is defined in Eq. 4.5 as follows:

$$WER_{\text{bit}}(t_w) = 1 - \exp\left(\frac{-\pi^2(I-1)\Delta}{4\,(I\exp(C(I-1)t_w)-1)}\right), \quad I = \frac{I_w}{I_c} \tag{4.5}$$



(a) MTJ Switching (b) Failed write operation (c) Delayed write operation

Figure 4.10: STT-RAM stochastic write operation

Where I is the write current, I_C is the critical current of the MTJ cell for writing, I_W is the write current, t_w is the switching time of the MTJ, and C is a technology-dependent parameter. The stochastic behavior originates from process variations affecting both the threshold voltage of access transistors and the resistance of the MTJ cell. These variations result in non-deterministic high and low resistance states in the circuit, increasing the WER due to incomplete writes. The probability of incomplete write operations can be calculated using Eq. 4.6 and Eq. 4.7. Here, γ is the gyromagnetic factor, α is the Landau-Lifshitz-Gilbert damping constant, H_k is the effective anisotropy field, t_{wr} is the writing pulse duration, and t_{sw} is the average switching delay of the MTJ cell. In the following, we present a write driver circuit that adjusts the write current density based on application needs to improve performance in STT-RAM.

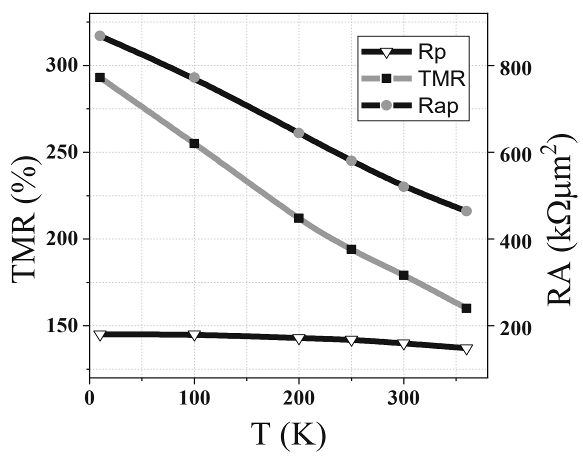
$$P_{\text{WER}} = 1 - \exp \frac{(\frac{-\pi^2}{4}(\frac{I}{I_c} - 1)}{(\frac{I}{I_c})\exp(\frac{2\alpha\gamma H_k t(\frac{I}{I_c}-1)}{1+\alpha^2})} - 1 \tag{4.6}$$

$$P_{\text{WER}} = e^{\left(-\frac{t_w}{t_{sw}}\right)} \tag{4.7}$$

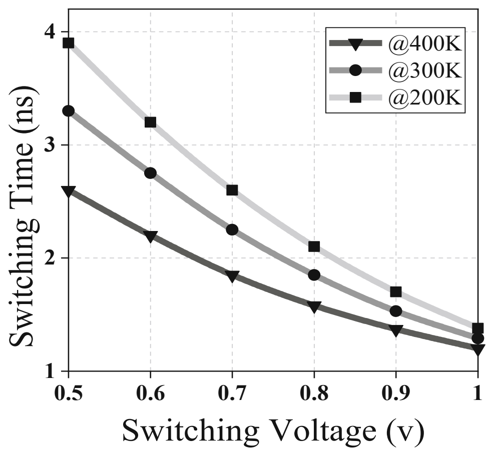
- **Random error intensifying write error rate** An important consideration for STT-RAM is implementing mechanisms to relax stress on the oxide barrier to ensure reliable, fast, and energy-efficient operations. One critical aspect is temperature management, as it influences several performance and reliability factors. Thermal stability is essential, as temperature affects the TMR, which describes the difference in resistance between parallel and anti-parallel magnetic orientations in MTJ. TMR tends to decrease as temperature increases, which can lead to reduced data retention stability.

  Fig. 4.11a demonstrates the effect of temperature on TMR, showing a decrease in TMR with rising temperature, which influences the stability of MTJ cells. Additionally, Fig. 4.11b illustrates the relationship between switching voltage and switching time for MTJ cells at various temperatures. It is observed that at a fixed switching voltage, the switching time decreases as temperature increases, indicating that MTJ switching becomes faster at higher temperatures. Conversely, for a given switching time, a lower switching voltage is required as temperature rises.

  Higher write currents can increase retention time but also pose a risk of oxide barrier breakdown, a type of hard error that compromises device reliability [86, 194]. To improve retention time, higher write energy is often necessary; however, applying such energy can also reduce endurance. Endurance is defined as the number of switching cycles a device can handle before failure, and it is influenced by the ratio of time-to-failure to the number of switch cycles. Thus, retention and endurance are inversely related. In this context, endurance is predominantly af-



(a) TMR relativity to temperature



(b) Switching time/voltage correlation to temperature

Figure 4.11: STT-RAM stochastic write operation

fected by the amount of power applied during switching events, which directly impacts thermal stress on the device.

STT-RAM typically has lower endurance and higher reliability compared to Voltage-Controlled Magnetic Anisotropy (VCMA) because it requires high write energy to alter the direction of free layer spins. However, STT-RAM does not depend on magnetic fields, which could affect adjacent cells in the submicron area. Consequently, STT-RAM offers high reliability in terms of radiation induced problems, temperature, and Process Variation (PV) despite limited scalability. Thus, it is essential to consider the retention/write-speed tradeoff in memory unit design. Write speed is often a super-exponential function of the applied voltage. Hence, while enhancing write speed or applying immense stress can improve retention time, it adversely affects endurance.

The state-of-the-art technique proposed in [195] aims to enhance the write performance of STT-RAM memories. The primary focus is on reducing the WER as much as possible. In older technologies, such as 32 nm, increasing the size of transistors improves reliability against high-energy particle strikes. However, in technologies below 32 nm, increasing the transistor size does not mitigate the effect of a high energy particle strike. Larger transistors reduce channel resistance and increase the switching speed of the memory element due to higher current flow rates. To study the impact of high energy particle strike on the write operation, a double exponential current source is utilized. Fig. 4.10a illustrates the write circuit operation without soft errors, while Figs. 4.10b and 4.10c depict the degradation of the writing process due to charged particle collisions. Simulation results show the effects of injected charges of $Q_{\text{inj}} = 50\,\text{Pf}$ and $Q_{\text{inj}} = 200\,\text{Ff}$, respectively.

In a conventional write circuit with both P-channel Metal-Oxide-Semiconductor (PMOS) and

NMOS transistors, the current source direction can either be out of the stroked node (negative glitch) or into the node (positive glitch). The effects vary depending on the particle's strength and type. The behavior of STT-RAM under thermal fluctuation is described by Eqs. 4.8, 4.9, and 4.10 [196]. In these equations, α is the Gilbert damping coefficient, γ is the gyromagnetic ratio, e is the electron charge, μ_B is the Bohr magneton, μ_0 is the permeability of free space, M_S is the saturation magnetization of the MTJ cell layers, H_K is the magnetic anisotropy field, V_{Sl} is the volume of the free layer, and E is the energy barrier. θ_0 is the initial angle of the magnetization direction of the free layer, which is thermally distributed and can be calculated by [197]. $\tau_0 \sim 1.0\,\mathrm{ns}$ is the relaxation time, λ is a coefficient ($\lambda = 0.2333$), and I is the writing current.

$$I_c = \alpha \left(\frac{\gamma \times e}{\mu_B \times g(T)} \right) \mu_0 M H_k V_k V_{si} = 2\alpha \left(\frac{\gamma \times e}{\mu_B \times g(T)} \right) E \tag{4.8}$$

$$t^{-1} = \frac{1}{\tau_0 \ln(\frac{\pi}{2\theta_0})} \times \left(\frac{1}{\lambda I_c} - 1 \right) \tag{4.9}$$

$$g(T) = \sqrt{\frac{TMR(T,V) \times (TMR(T,V) + 2)}{2(TMR(T,V) + 1)}} \tag{4.10}$$

4.2.2 Proposed Approximated Write Driver

Our proposed technique write circuit architecture is presented in Fig. 4.12. The write driver, which as part of a row decoder, includes a soft error-tolerant component and an approximate write current injector. For writing a "logic zero," the access transistors are connected to VDD_{Low} (VDDL). For writing a "logic one," there are three methods available to switch the state from parallel to antiparallel. It is important to note that implementing the $dual_V DD$ concept will add minimal overhead to the fabrication process. Various methods exist for managing voltage domains, such as level converters, Clustered Voltage Scaling (CVS), Dynamic Voltage Scaling (DVS), or Dynamic Voltage and Frequency Scaling (DVFS) [198, 199]. However, since our approach only requires two voltage levels, we can use a bandgap circuit to achieve this. A bandgap voltage reference is a temperature-independent circuit commonly used in integrated circuits. It provides a stable voltage that remains constant despite power supply variations, temperature changes, or circuit loading from a device [139].

We also make modifications to the threshold voltage of transistors connected to high voltage (VDDH) to manage the incremental heat dissipation caused by the leakage current path through these transistors (i.e., $\bar{T}_{22}$, $\bar{T}_3$, $\bar{T}_{33}$, T_3, T_{22}, and T_{33}). The write driver operates based on the quality decoder's decision, activating a specific part of the write driver in the memory circuit.

If the input data is "logic zero," then T_0 and the transistors connected to VDDL will start to inject current into the memory element. Conversely, if the input data is "logic one," there are three possible options for injecting the write current. The quality controller will determine the final choice among these three options, tagging them based on the importance of the data.

In the first category, where the data ("logic-one") has minor importance, the transistors T_1 and $\bar{T}_1$ are responsible for writing to the circuit. These transistors are connected to VDDL, resulting in the least amount of current passing through the cell, which may cause some write errors during limited time intervals.

In the second scenario, where the data's priority is medium, two pairs of transistors ($T_2, \bar{T}_2, T_{22}, \bar{T}_{22}$) are involved in writing the input data. The threshold current density in the MTJ cell can be expressed by Eqs. 4.11, 4.12, and 4.13. In these equations, t is the width of the free layer of the MTJ cell, $g(\theta)$ is the efficiency, and H_S, H_{ki}, and H_{dip} are the applied perpendicular anisotropy and dipole fields from the pinned layer acting on the free layer, respectively. At the architecture level, incoming data is tagged based on its priority, which is assigned by the programmer via our provided API from the application level. The data is categorized into four different priority levels from 00 to 11.

$$J_{CO}^{P \to AP} = \frac{\alpha \gamma e M_s t}{\mu_B \times g(0)} \times [(H_{ex} + H_{dip}) + (H_{ki} + H_d)] \tag{4.11}$$

$$J_{CO}^{P \to AP} = \frac{\alpha \gamma e M_s t}{\mu_B \times g(\pi)} \times [(H_{ex} + H_{dip}) + (H_{ki} + H_d)] \tag{4.12}$$

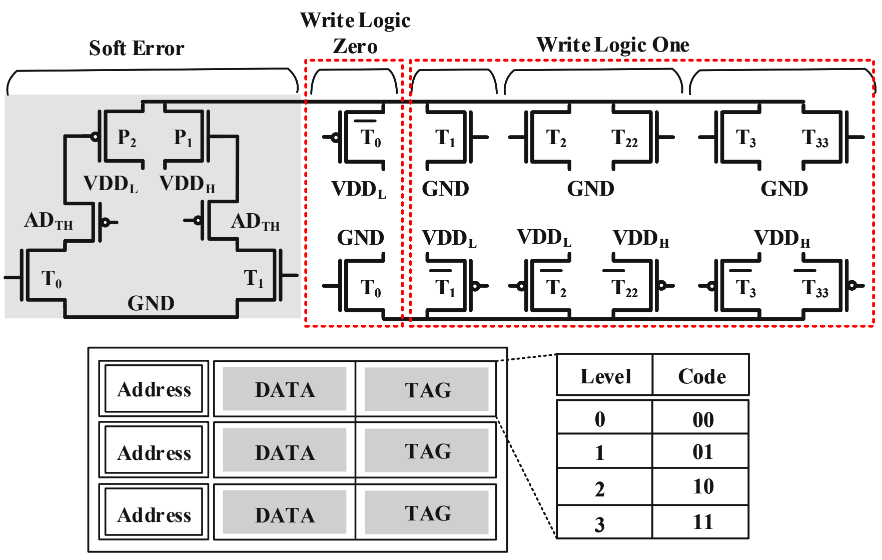


Figure 4.12: Proposed architecture of write circuit

$$g(\theta) = \frac{P}{2 \times (1 + P^2) \cos\theta} \tag{4.13}$$

The EXTENT module then begins to read/write data in the circuit. Initially, the quality control detects the data's priority and activates the related write enable signal. The address is fetched from the address decoder, and if the CMP module grants write permission (approved if the data is new and not repetitive), the write module injects current. During operation, the CMP module senses the cell's value and cuts off the write current immediately when the cell's value changes.

The CMP is a device that compares two voltages (or currents) and outputs a digital signal indicating which is higher. It has two analog input terminals, V^+ and V^-, and one binary digital output V_o. While this article does not emphasize using a specific type of comparator, critical parameters to consider are resolution and compatibility with the CMOS process. Our HSPICE simulation uses the circuit design from [64].

To find a correlation between body-biasing and the threshold voltage of the transistor, Eq. 4.14 can be applied. The sub-threshold current can be calculated by tuning this threshold parameter using Eq. 4.15. As the sub-threshold parameter increases, the mobility parameter of the transistors also increases, as described by Eq. 4.16.

$$V_{th} = V_{th_0} + \gamma \left(\sqrt{2\phi_F + V_{SB}} - \sqrt{2\phi_F} \right) \tag{4.14}$$

$$I_{DS} = I_S 10^{\frac{V_{GS}-V_{th}}{s}} \left(1 - 10^{\frac{V_{GS}-V_{th}}{S}} \right) \tag{4.15}$$

$$I_{DS} = \mu C_{ox} \left(\frac{W}{L} \right) \left(V_{GS} - V_{th} - \frac{V_{DS}}{2} \right) V_{DS} \tag{4.16}$$

The final result of an increase in mobility will be an increase in the transistor's temperature (according to Eq. 4.17). In these equations, V_{th0} is the threshold voltage for the initial state of V_{SB}, and Φ_F is the surface potential. W and L are the width and length of the transistor, respectively. C_{ox} is the gate oxide capacitance per unit area, and μ is the carrier mobility of the transistor, which depends on the temperature as indicated by Eqs. 4.15 to 4.19.

$$\mu(T) = \mu(T_r) \left(\frac{T}{T_r} \right)^{-K_u} \tag{4.17}$$

$$P_{SW} = 1 - \exp\left(\frac{-t_p}{\tau} \right) \tag{4.18}$$

$$\tau = \tau_0 e^{\Delta\left(1 - \frac{V}{V_{C0}}\right)} \tag{4.19}$$

In these equations, P_{sw} is the switching probability of the MTJ cell, t_p is the duration of the voltage pulse period, and τ_0 is the initial thermal state of the MTJ cell, which is $0°K$. According to Eqs. 4.18, and 4.19, we can derive the correlation between temperature and the probability of change in the free layer.

By increasing the size of the transistor, it is not possible to protect it against the collision of charged particles in technology nodes below $32nm$. Therefore, in the circuit design, we increased the number of parallel transistors without increasing the transistor size. This approach ensures that the designed circuit can be used for any technology and transistor size, functioning appropriately.

Depending on the application, two transistors with an adjustable threshold can be fabricated with varying V_{th}. Consider a scenario where a particle strikes over $N1$, and transistors T_1 and T_2 are in one state. The negative glitch caused by the particle will switch the adjustable transistor to the on-state, consequently switching P_2 to the on-state, thus compensating for the unexpected glitch.

4.2.3 Cross-Layer Evaluation of Proposed Approximated Write Driver

To comprehensively evaluate the proposed approximation-based write driver, we conduct a cross-layer analysis spanning circuit, architecture, system, and application levels. At the circuit level, we analyze electrical behaviour, including write latency, energy consumption, and process variations. The architectural-level evaluation integrates our design into the last-level cache (LLC) within the GEM5 full-system simulator to assess its impact on cache operations. At the system level, we measure write error rate (WER) and overall energy efficiency, considering system-wide effects. Finally, the application-level evaluation employs MiBench workloads to quantify real-world performance improvements and energy savings. The following sections present detailed evaluations at each of these layers.

- **Architectural Evaluation:** This section evaluates the architecture of the proposed design (see Fig. 4.12) against various industrial benchmarks. We modified and rectified the cache within the GEM5 full-system simulator to implement our proposed method for a homogeneous architecture [200]. Accordingly, we applied our architecture to the Last Level Cache (LLC).
  We modelled the quality decoder similar to a random fault injection scheme during the write operation for "logic-one" in both level-1 and level-2. The fault injection follows the WER probability function described in Eqs. 4.6 and 4.7 for each memory element, using a uniform distribution. Table 4.3 provides detailed information about the simulation platform used for our method.
  System-level accuracy either matches the circuit's WER or improves due to application-level masking effects. As all examined techniques are applied to the same workloads, the application-level masking effects are similar. Hence, we can assess system-level accuracy using WER. Our approach demonstrates either a similar or better WER compared to the state-of-the-art.

Table 4.3: Configuration of the memory cell

Component	**Details**
CPU	Quad-core, 1GHz, out-of-order
L1 Cache (Ins. and Data)	32KB, 4-way set associative, 64B B-size,
L2 Cache	1MB, 8-way set associative, 64B B-size, shared, EXTENT active

The normalized average energy usage, reported in Fig. 4.14, remains unaffected under this assumption.

- **Circuit Evaluation:**

  For calculating the exact values of V_{th} for NMOS and PMOS transistors, simulations were conducted with the circuit at $400°K$. According to the relationships described in Eqs. 4.15 - 4.19, these temperatures result in a decrease in the overall write latency of the circuit. The exact value for *VDDL* is calculated to be 0.86001V, considering *VDDH* of the circuit, total power consumption, and latency for writing "logic-zero" as optimization parameters [201]. The circuits were simulated under various process corners, showing that our circuit is immune to process variation. The simulation waveforms of our proposed write circuit shown in Fig. 4.13 demonstrate that this circuit can identify the input data.

  If the new input data is equal to the stored data in the cell, the system immediately cuts the current to prevent repetitive write operations, leading to energy savings. The parameter θ represents the angle between the free and fixed layers of MTJ cells. The minor disturbance observed in the θ waveform occurs due to the initial current passing through the MTJ cell before the system identifies it as a repetitive write attempt. In write operations where the input data differs from the stored data, θ ranges between $[0, 180]$, respectively, changing the magnetization orientation of the free and fixed layers. In a non-repetitive write-based operation, after the write operation concludes, the cell's resistance changes. It is worth mentioning that the write enable signal pulse width is set to $10ns$, consistent with state-of-the-art approaches [64, 85, 202, 203]. These resistance changes cause minor disturbances over either V_{BL} or V_{SL}. This disturbance is used in the comparator module to cut the write current right after the operation concludes. One of the write drivers will be activated to adjust the write error probability based on the importance of the input data (there are four levels of approximation in the circuit). Writing "logic-one" typically takes more time and energy than "logic-zero" $(2.5X)$. Table 4.5 compares the essential parameters of STT-RAM memories. The proposed circuit improves the main parameters of STT-RAM with negligible area overhead compared to state-of-the-art solutions. In all cases, the "Read-Energy" and "Read-Latency" parameters remain equal as there is no change in the read circuit of these designs. For comparison purposes, it is worth noting that the technique proposed in [66] discusses

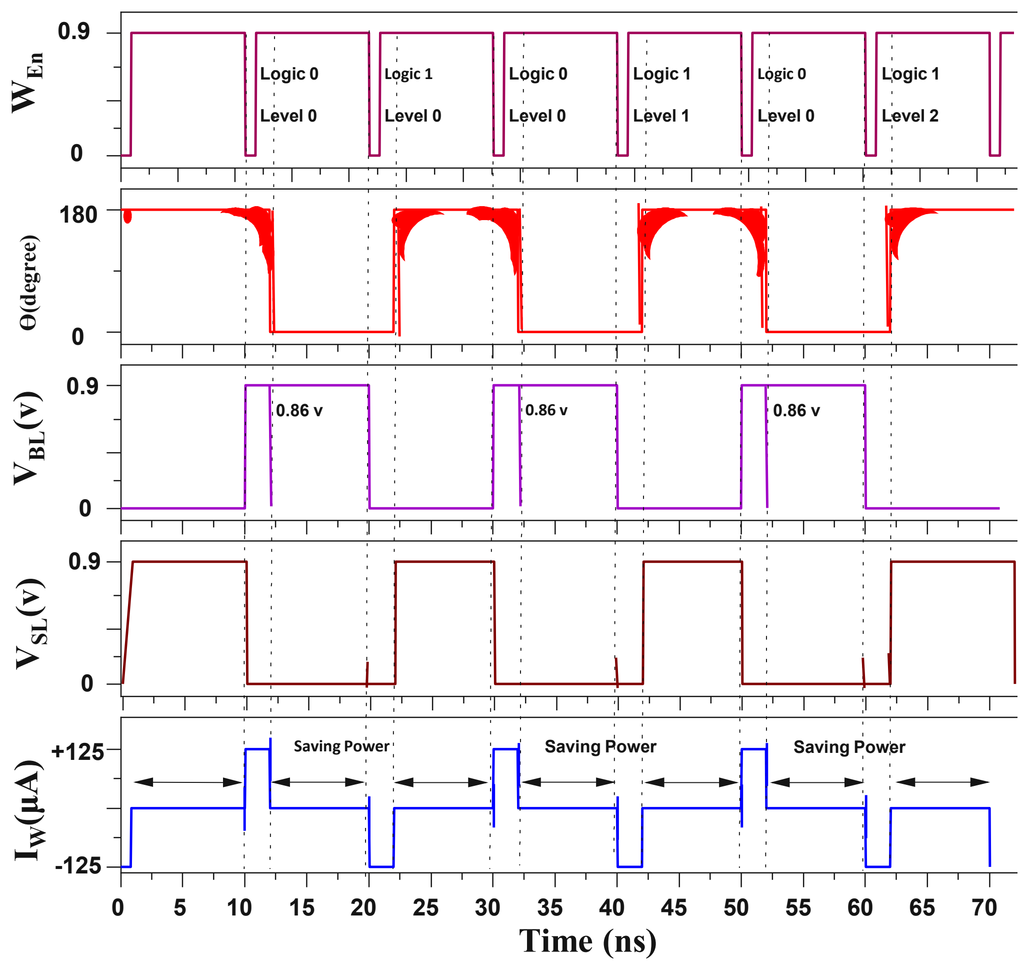


Figure 4.13: Simulation waveform of different levels

Table 4.4: MTJ cell physical parameters

Parameter	Description	Default Value
Area [mm^2]	MTJ Cell Surface	$16e^{-9}$
TMR	TMR with zero V_{bias}	200%
T_ox	Oxide Barrier Height	$8.5e^{-10}$
R.A [$\Omega.mm^2$]	$Resistance \times Area$	5
I_C [$\mu.A$]	Critical Current	200
T [$°K$]	Temperature	300
T_SL	Height of the free layer	$1.3e^{-9}$
R_P [$K\Omega$]	Low Resistance	4.2
R_AP [$K\Omega$]	High Resistance	6.6

the effect of the thermal stability factor on image output quality. To make a meaningful comparison, we first implemented the approach in [68] with a $32nm$ technology node and increased the thermal stability factor from 10 to 70 using SPICE simulation to measure write latency and write energy, and required circuit-level evaluation is reported in Table 4.5.

Table 4.5: EXTENT vs. state-of-the-art techniques

	Basic Cell	[65]	[66]	[204]	[This Work [68]]
Area [mm^2]	1.31	1.37	1.31	1.41	1.46
Write Latency [ns]	19.0	2.20	7.30	7.80	6.90
Write Energy [Pj]	1046.0	503.6	393.3	356.9	337.2
Write Termination	No	No	No	YES	YES
Monitoring	None	Continuous	None	Continuous	Continuous

Subsequently, we passed these results into our GEM5 full-system simulator to extract the output results for our applications. The results for [65] and [66] were averaged over their reported range. Circuit-level characteristics were extracted via SPICE simulation for all available approximation levels in our study, including fully approximate, fully accurate, and partially approximate configurations. System-level evaluation was conducted on the same GEM5 machine for result comparison. We performed circuit-level simulation using HSPICE for a 3-level write driver circuit. The STT-RAM model compatible with SPICE in all these simulations is presented in [205], and the $32nm$ PTM CMOS technology model is presented in [206]. Table 4.4 summarizes the basic configuration of the STT-RAM model. In all analyses, the supply voltage is set to $0.9V$, and the pulse width duration is $10ns$. This technique is generalized and can be applied to any specific model available.

- **Application Analysis:**
  We consider several applications from the MiBench benchmark suites [207]. The amount of energy used for each benchmark is shown in Fig. 4.14. As can be seen, greater energy savings are achieved for all benchmarks with a majority of “logic-zero” to “logic-one” transitions. It is worth mentioning that EXTENT is highly independent of the running application. The MiBench benchmark is used here to compare our methodology to state-of-the-art techniques. Although the aforementioned benchmark may not fully utilize the CACHE, it will experience a heavy workload when the running application is in a closed loop.
- **Variation Analysis:**
  Magnetization switching assisted by thermal agitation, where the current density can be significantly smaller than the critical density, is an effective energy reduction approach. Due to serious reliability challenges in nanometer-scale technology, simulations have been conducted to quantitatively evaluate reliability and energy consumption under the combined influence of process, supply voltage, and temperature variations. Fig. 4.15a illustrates the impact of process variation on the minimum write energy of EXTENT with different transistor sizes, considering whether the approximation is applied or not. Thermal activation switching occurs in approximately ten

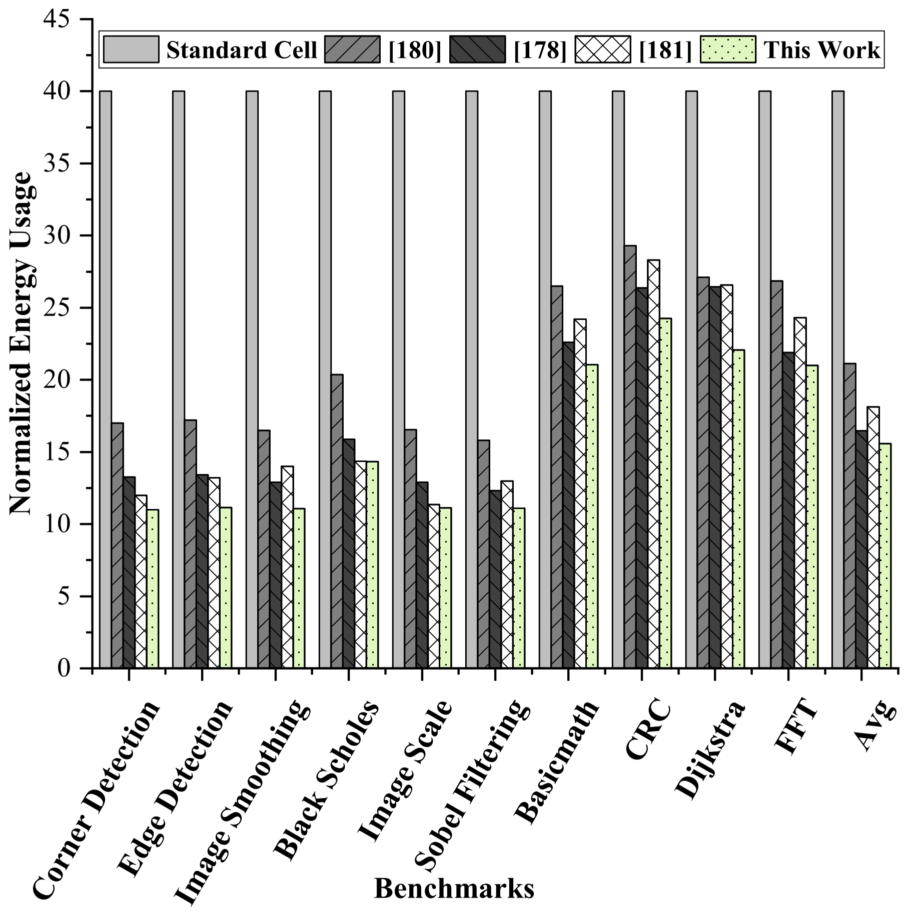
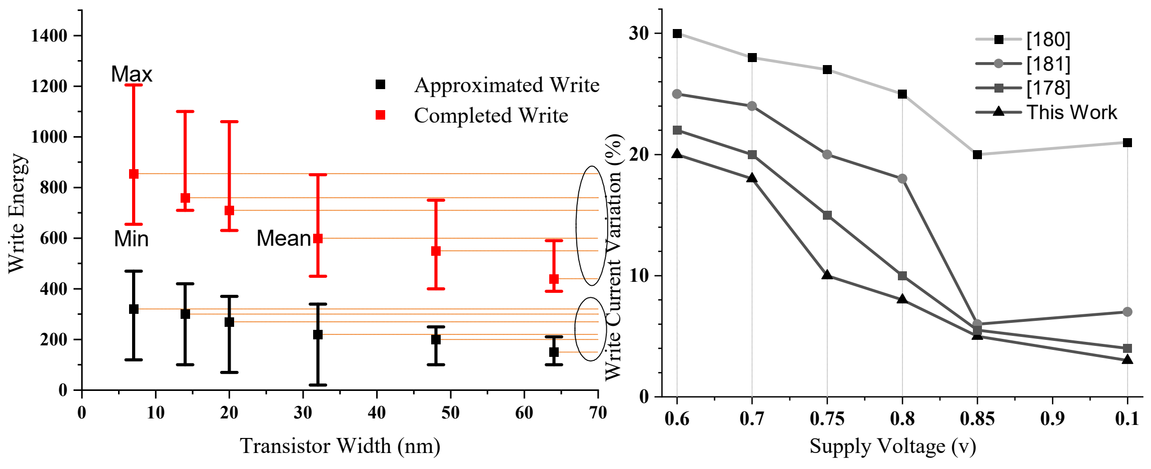


Figure 4.14: Normalized energy improvement compared to state of the art

ns pulse width duration. Variations in input supply voltage lead to changes in write current and write performance. Fig. 4.15b shows the effect of voltage variation on write performance.

Since an increase in the resistance of one element can compensate for a decrease in the resistance of another, our reliability simulations account for process variations affecting all relevant parameters simultaneously. Various process parameters, including standard deviation and mean, have been considered. Due to the low TMR of STT-RAM compared to other non-volatile memory technologies, process variation improvement could be possible by reducing the resistance of STT-RAMs in a parallel state and increasing the resistance in an antiparallel state. Parallelizing two low resistance STT-RAMs and serializing two high resistance STT-RAMs can be an efficient solution, along with physical level solutions mentioned in [206].

Considering the voltage spectrum, EXTENT has a broader range (3% to 20%), similar to the methods presented in [66, 204]. In Fig. 4.15b, the write energy of EXTENT across different technology nodes, both with software assistance (approximated) and uniform write (completed), is depicted. The effect of variation for approximated write is minimal (between minimum and maximum values), whereas the write energy for completed write ranges between 400 to 1200 pJ, and approximated write ranges from nearly 0 to 500 pJ. For a fair comparison with other methods, considering [208], a range of 1% to 10% variation has been considered. By integrating process variations (including oxide barrier thickness 10%, FM layer thickness 10%, and resistance 5%) and thermal fluctuations (10%) into a compact PMA STT-MTJ SPICE model [205], and employing a commercial CMOS 32-nm design kit (with process variations of 3σ for channel length, width, and threshold voltage), MC statistical simulations (1000 runs) were performed to evaluate the impacts of process variations and thermal fluctuations on the entire circuit. MC simulations, conducted 1000 times considering variations in L (channel length), W (channel width) of transistors, and oxide (barrier) thickness of both transistors and MTJs, obtained the variation of write energy in the presence of process variations in the proposed circuit. To evaluate the effects of process variations on the regular operation of the proposed method, we selected a value representing the percentage variation in the mentioned parameters, varying up to 10% from their original values using a Gaussian distribution with a standard deviation of 3%.



(a) Supply Voltage Variation (b) Process Variation

Figure 4.15: Process and voltage variation effect

Chapter 5

Conclusion and Future Works

5.1 Thesis Summary

As data-centric applications continue to grow, traditional von Neumann architectures face critical limitations due to the so-called “memory wall”—an inefficiency caused by frequent data transfers between the CPU and memory, leading to excessive power consumption and latency. Moreover, these memory-centric applications often run on hardware prone to various reliability threats, such as soft errors and process variations, which can degrade both reliability metrics and output accuracy.

This thesis addresses the research challenges associated with memory-centric systems and proposes various techniques to enhance the robustness of both STT-RAM- and SRAM-based architectures. In particular, it introduces novel approaches in approximate computing and processing in memory to further improve system reliability.

This thesis addresses three principal challenges in state-of-the-art in-memory computing systems, focusing on both STT-RAM- and SRAM-based designs. First, the high write energy consumption and reliability concerns in STT-RAM are mitigated by the development of EXTENT, a write optimization technique that exploits stochastic switching and circuit-level approximations to enhance robustness while improving energy efficiency and latency [5] (see Chapter 4.2). In parallel, we introduce HOPE, an open-source STT-RAM memory controller integrated into gem5 (see Chapter 4.1), which enables microarchitectural-level evaluations of power, latency, and throughput. By accurately modeling write operations and system-level trade-offs, the HOPE framework allows for informed optimizations under varying application constraints.

Second, the challenge of *energy-accuracy trade-offs* in in-SRAM MAC operations—where non-linear voltage behavior can significantly degrade precision—is tackled through multiple innovations. In Chapter 3, our proposed technique **AID** ensures linear BL discharge in 6T-SRAM, delivering higher accuracy for CNN inference without sacrificing energy efficiency [2]. Similarly, **SMART** improves read margins for wider bit-widths, reducing error rates in in-SRAM MAC accelerators (see Chapter 3.6). Additionally, **E-MAC** eliminates the need for expensive DACs, employing time-modulated WL voltage to boost energy efficiency with minimal accuracy loss, as demonstrated on diverse CNN benchmarks (see Chapter 3.8 for more detail).

Lastly, this thesis addresses the lack of robust design-space exploration (DSE) methodologies, which causes a limited range of design choices for in-SRAM computing circuits. To overcome this challenge, we introduce OPTIMA (in Chapter 3.7), a fast yet accurate DSE framework specifically tailored to in-SRAM circuits. OPTIMA offers multi-parameter optimization, drastically reducing simulation times while enabling precise trade-offs among power, accuracy, and reliability.

Overall, these contributions form a cohesive framework that spans the memory technology spectrum—covering reliability improvements in STT-RAM, advanced accuracy techniques in SRAM, and comprehensive modeling and exploration framework for in-SRAM circuits. The proposed methods not only push the boundaries of performance and energy efficiency but also lay a robust foundation for real-world applications in edge devices and IoT systems, where tight resource constraints demand increasingly sophisticated and reliable in-memory computing solutions.

5.2 Future Works

The result of the research presented in this thesis open several avenues for further research and development in the scope of in-memory computing. Future work should focus on enhancing the reliability, scalability, and adaptability of processing-in-memory (PIM) architectures to broaden their applicability across diverse fields.

1. **Reliability Enhancements in STT-RAM and SRAM:**
   While the EXTENT method addressed write error reliability in STT-RAM, additional research is necessary to enhance the overall reliability of both STT-RAM and SRAM architectures. Future work could explore approaches to mitigate soft errors, improve write endurance, and ensure data integrity under various operating conditions, such as temperature fluctuations and process variations. These enhancements are particularly critical for applications demanding high fault tolerance.
2. **Advanced Approximation Techniques for Energy and Performance Trade-offs:**
   As energy efficiency and speed continue to be crucial for in-memory computing, the development of dynamic approximation techniques that adapt to workload characteristics could provide substantial benefits. Research in this area could focus on creating adaptive algorithms that opti-

mize performance by adjusting approximation levels in real-time, balancing energy savings with the accuracy and reliability requirements of specific applications.

3. **Scalability to Complex Architectures and Emerging Memory Technologies:**
   To further validate the effectiveness of the proposed methods, it is essential to assess their scalability to larger and more complex neural network models, such as ResNet and Transformer architectures. Moreover, integration with emerging memory technologies like ReRAM and Phase-Change Memory (PCM) could offer new performance enhancements while addressing some limitations of existing technologies. This exploration could potentially yield hybrid memory architectures that combine the advantages of volatile and non-volatile memory types.
4. **Comprehensive Design Automation Tools:**
   Developing advanced design automation tools that incorporate our proposed methodologies could streamline the adoption of PIM architectures in various industries. Such tools would facilitate design-space exploration, optimization, and verification processes, enabling designers to evaluate trade-offs in energy, latency, and accuracy more efficiently. Integrating these tools into widely used simulation frameworks would also encourage adoption and standardization in the field.
5. **Security and Privacy in PIM Architectures:**
   As in-memory computing architectures are deployed in sensitive applications such as healthcare and finance, security and privacy concerns must be addressed. Future research could explore potential vulnerabilities introduced by approximation techniques and develop mitigation strategies to ensure data security. Techniques such as differential privacy and hardware-based security modules could be integrated into PIM architectures to provide robust protection against potential threats.
6. **Reliability-Centric In-Memory Architectures for Critical Applications:**
   Building on the reliability improvements introduced with EXTENT, future research could focus on developing in-memory computing architectures with reliability as a primary design criterion. This could involve creating redundancy schemes, error correction mechanisms, and fault tolerance strategies tailored for STT-RAM and SRAM architectures in applications where data integrity is critical, such as aerospace, healthcare, and autonomous systems.

In conclusion, this thesis lays a robust foundation for advancing in-memory computing, addressing power efficiency, accuracy, and—importantly—reliability. Despite the clear advantages of PIM-based systems, they still face significant robustness challenges, mainly due to the absence of comprehensive modeling and benchmarking frameworks that can accurately assess proposed architectures for reliability and energy efficiency prior to deployment. To bridge this gap, this thesis introduces novel modeling and benchmarking frameworks tailored for PIM architectures, enabling systematic evaluation and optimization. Furthermore, it advances robust PIM circuit design methodologies to ensure resilience against process variations and other reliability threats. Together, these contributions pave the way for broader adoption of PIM-based technologies in real-world applications, fostering innovation in data-centric fields that demand high-performance, energy-efficient, and reliable computing solutions.

Bibliography

[1] S. Seyedfaraji, S. Jager, S. Shakibhamedan, A. Aftab, and S. Rehman, "Optima: Design-space exploration of discharge-based in-sram computing: Quantifying energy-accuracy trade-offs," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*, 2024, pp. 1–6.

[2] S. Seyedfaraji, B. Mesgari, and S. Rehman, "Aid: Accuracy improvement of analog discharge-based in-sram multiplication accelerator," in *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2022, pp. 873–878.

[3] S. Seyedfaraji, S. Shakibhamedan, A. Seyedfaraji, B. Mesgari, N. TaheriNejad, A. Jantsch, and S. Rehman, "E-mac: Enhanced in-sram mac accuracy via digital-to-time modulation," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, 2024.

[4] S. Seyedfaraji, M. Bichl, A. Aftab, and S. Rehman, "Hope: Holistic stt-ram architecture exploration framework for future cross-platform analysis," *IEEE Access*, 2024.

[5] S. Seyedfaraji, J. T. Daryani, M. M. S. Aly, and S. Rehman, "Extent: Enabling approximation-oriented energy efficient stt-ram write circuit," *IEEE Access*, vol. 10, pp. 82 144–82 155, 2022.

[6] S. Seyedfaraji, B. Mesgari, and S. Rehman, "Smart: Investigating the impact of threshold voltage suppression in an in-sram multiplication/accumulation accelerator for accuracy improvement in 65 nm cmos technology," in *2022 25th Euromicro Conference on Digital System Design (DSD)*. IEEE, 2022, pp. 821–826.

[7] D. Jana, S. Roy, R. Panja, M. Dutta, S. Z. Rahaman, R. Mahapatra, and S. Maikap, "Conductive-bridging random access memory: challenges and opportunity for 3d architecture," *Nanoscale research letters*, vol. 10, pp. 1–23, 2015.

[8] J. S. Meena, S. M. Sze, U. Chand, and T.-Y. Tseng, "Overview of emerging nonvolatile memory technologies," *Nanoscale research letters*, vol. 9, pp. 1–33, 2014.

[9] Y. Wang, C. Zhang, H. Yu, and W. Zhang, “Design of low power 3d hybrid memory by non-volatile cbram-crossbar with block-level data-retention,” in *Proceedings of the 2012 ACM/IEEE international symposium on Low power electronics and design*, 2012, pp. 197–202.

[10] Y. Shin, “Non-volatile memory technologies for beyond 2010,” in *Digest of Technical Papers. 2005 Symposium on VLSI Circuits, 2005.* IEEE, 2005, pp. 156–159.

[11] B. Gervasi, “Will carbon nanotube memory replace dram?” *IEEE Micro*, vol. 39, no. 2, pp. 45–51, 2019.

[12] J. Lamb, S. Gibbons, R. Trichur, Y. Jiang, K. Mangelson, K. Kremer, and D. Janzen, “Advancements in microelectronics-grade carbon nanotube materials for nram® device manufacture and analysis of carbon nanotube mass in end user devices,” *Nanotech 2014*, pp. 194–197, 2014.

[13] M. Imani, S. Patil, and T. Rosing, “Low power data-aware stt-ram based hybrid cache architecture,” in *2016 17th international symposium on quality electronic design (isqed).* IEEE, 2016, pp. 88–94.

[14] S. Jeloka, Z. Wang, R. Xie, S. Khanna, S. Bartling, D. Sylvester, and D. Blaauw, “Energy efficient adiabatic fram with 0.99 pj/bit write for iot applications,” in *2018 IEEE symposium on VLSI circuits.* IEEE, 2018, pp. 85–86.

[15] “International roadmap for devices and systems,” Online, 2020, accessed: Jan 2024. [Online]. Available: https://irds.ieee.org/editions/2020

[16] I. Yoon, A. Anwar, T. Rakshit, and A. Raychowdhury, “Transfer and online reinforcement learning in stt-mram based embedded systems for autonomous drones,” in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE).* IEEE, 2019, pp. 1489–1494.

[17] B. Narasimham, V. Chaudhary, M. Smith, L. Tsau, D. Ball, and B. Bhuva, “Scaling trends in the soft error rate of srams from planar to 5-nm finfet,” in *2021 IEEE International Reliability Physics Symposium (IRPS).* IEEE, 2021, pp. 1–5.

[18] J. Wang, N. Xiu, J. Wu, Y. Chen, Y. Sun, H. Yang, V. Narayanan, S. George, and X. Li, “An 8t/cell fefet-based nonvolatile sram with improved density and sub-fj backup and restore energy,” in *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2022, pp. 3408–3412.

[19] J. Y. Kim, M.-J. Choi, and H. W. Jang, “Ferroelectric field effect transistors: Progress and perspective,” *APL Materials*, vol. 9, no. 2, p. 021102, 02 2021.

[20] S. Yu, Q. Wang, Y. Zhang, P. Yang, X. Luo, H. Liu, C. Chen, Q. Li, and S. Liu, “Multistate capability improvement of beol compatible fefet by introducing an al2o3 interlayer,” *IEEE Transactions on Electron Devices*, vol. 70, no. 11, pp. 5632–5637, 2023.

[21] J. Y. Park, D.-H. Choe, D. H. Lee, G. T. Yu, K. Yang, S. H. Kim, G. H. Park, S.-G. Nam, H. J. Lee, S. Jo, B. J. Kuh, D. Ha, Y. Kim, J. Heo, and M. H. Park, “Revival of ferroelectric memories based on emerging fluorite-structured ferroelectrics,” *Advanced Materials*, vol. 35, no. 43, p. 2204904, 2023.

[22] M. Ali, A. Jaiswal, S. Kodge, A. Agrawal, I. Chakraborty, and K. Roy, “Imac: In-memory multi-bit multiplication and accumulation in 6t sram array,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 8, pp. 2521–2531, 2020.

[23] Wikipedia contributors, “Von Neumann architecture — Wikipedia, The Free Encyclopedia,” 2024, [Online; accessed 28-October-2024]. [Online]. Available: https://en.wikipedia.org/wiki/Von_Neumann_architecture

[24] G. Kestor, R. Gioiosa, D. J. Kerbyson, and A. Hoisie, “Quantifying the energy cost of data movement in scientific applications,” in *2013 IEEE international symposium on workload characterization (IISWC)*. IEEE, 2013, pp. 56–65.

[25] A. Boroumand, S. Ghose, Y. Kim, R. Ausavarungnirun, E. Shiu, R. Thakur, D. Kim, A. Kuusela, A. Knies, P. Ranganathan *et al.*, “Google workloads for consumer devices: Mitigating data movement bottlenecks,” in *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems*, 2018, pp. 316–331.

[26] B. Jacob, S. Ng, and D. Wang, “Dram device organization: Basic circuits and architecture,” *Memory Systems: Cache, DRAM, Disk, Burlington, MA, USA: Morgan Kaufmann*, pp. 333–335, 2010.

[27] R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti, “Introduction to flash memory,” *Proceedings of the IEEE*, vol. 91, no. 4, pp. 489–502, 2003.

[28] A. Gebregiorgis, L. Wu, C. Münch, S. Rao, M. B. Tahoori, and S. Hamdioui, “Special session: Stt-mrams: Technology, design and test,” in *2022 IEEE 40th VLSI Test Symposium (VTS)*. IEEE, 2022, pp. 1–10.

[29] B. Hoffer, N. Wainstein, C. M. Neumann, E. Pop, E. Yalon, and S. Kvatinsky, “Stateful logic using phase change memory,” *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 8, no. 2, pp. 77–83, 2022.

[30] Y.-C. Luo, J. Hur, Z. Wang, W. Shim, A. I. Khan, and S. Yu, "A technology path for scaling embedded feram to 28 nm and beyond with 2t1c structure," *IEEE Transactions on Electron Devices*, vol. 69, no. 1, pp. 109–114, 2021.

[31] H. Philip, H.-Y. Lee, S. Yu, Y.-S. Chen, Y. Wu, P. Chen, B. Lee, T. F. Chen, and M. Tsai, "Metal–oxide rram," in *Proceedings of the IEEE*, vol. 100, no. 6, 2012, pp. 1951–1970.

[32] Y. Long, T. Na, and S. Mukhopadhyay, "Reram-based processing-in-memory architecture for recurrent neural network acceleration," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 12, pp. 2781–2794, 2018.

[33] S. Lee, S.-h. Kang, J. Lee, H. Kim, E. Lee, S. Seo, H. Yoon, S. Lee, K. Lim, H. Shin *et al.*, "Hardware architecture and software stack for pim based on commercial dram technology: Industrial product," in *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2021, pp. 43–56.

[34] W. Jeon, J. Lee, D. Kang, H. Kal, and W. W. Ro, "Pimcaffe: Functional evaluation of a machine learning framework for in-memory neural processing unit," *IEEE Access*, vol. 9, pp. 96 629–96 640, 2021.

[35] J. Gómez-Luna, I. E. Hajj, I. Fernandez, C. Giannoula, G. F. Oliveira, and O. Mutlu, "Benchmarking a new paradigm: An experimental analysis of a real processing-in-memory architecture," *arXiv preprint arXiv:2105.03814*, 2021.

[36] T. Chen, J. Botimer, T. Chou, and Z. Zhang, "An sram-based accelerator for solving partial differential equations," in *2019 IEEE Custom Integrated Circuits Conference (CICC)*. IEEE, 2019, pp. 1–4.

[37] K. Sanni, T. Figliolia, G. Tognetti, P. Pouliquen, and A. Andreou, "A charge-based architecture for energy-efficient vector-vector multiplication in 65nm cmos," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2018, pp. 1–5.

[38] M. Gong, N. Cao, M. Chang, and A. Raychowdhury, "A 65nm thermometer-encoded time/charge-based compute-in-memory neural network accelerator at 0.735 pj/mac and 0.41 pj/update," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 4, pp. 1408–1412, 2020.

[39] s. b. S. IDC, "Data age 2025: The digitization of the world from edge to core," 2018, accessed: 2024-10-28. [Online]. Available: https://www.seagate.com/files/www-content/our-story/trends/files/data-age-2025-idc-seagate.pdf

[40] O. Mutlu, S. Ghose, J. Gómez-Luna, and R. Ausavarungnirun, "Processing data where it makes sense: Enabling in-memory computation," *Microprocessors and Microsystems*, vol. 67, pp. 28–41, 2019.

[41] W. A. Wulf and S. A. McKee, "Hitting the memory wall: Implications of the obvious," *ACM SIGARCH computer architecture news*, vol. 23, no. 1, pp. 20–24, 1995.

[42] R. Buehrer and K. Ekanadham, "Incorporating data flow ideas into von neumann processors for parallel execution," *IEEE Transactions on Computers*, vol. 100, no. 12, pp. 1515–1522, 1987.

[43] O. Mutlu and L. Subramanian, "Research problems and opportunities in memory systems," *Supercomputing frontiers and innovations*, vol. 1, no. 3, pp. 19–55, 2014.

[44] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki, and B. Falsafi, "Clearing the clouds: a study of emerging scale-out workloads on modern hardware," *Acm sigplan notices*, vol. 47, no. 4, pp. 37–48, 2012.

[45] O. Mutlu, S. Ghose, J. Gómez-Luna, and R. Ausavarungnirun, "Enabling practical processing in and near memory for data-intensive computing," in *Proceedings of the 56th Annual Design Automation Conference 2019*, 2019, pp. 1–4.

[46] K. Kanellopoulos, N. Vijaykumar, C. Giannoula, R. Azizi, S. Koppula, N. M. Ghiasi, T. Shahroodi, J. G. Luna, and O. Mutlu, "Smash: Co-designing software compression and hardware-accelerated indexing for efficient sparse matrix operations," in *Proceedings of the 52nd annual IEEE/ACM international symposium on microarchitecture*, 2019, pp. 600–614.

[47] O. Mutlu, "Memory scaling: A systems architecture perspective," in *2013 5th IEEE International Memory Workshop*. IEEE, 2013, pp. 21–25.

[48] S. Kanev, J. P. Darago, K. Hazelwood, P. Ranganathan, T. Moseley, G.-Y. Wei, and D. Brooks, "Profiling a warehouse-scale computer," *IEEE Micro*, vol. 36, no. 3, pp. 54–59, 2016.

[49] M. Alser, Z. Bingöl, D. S. Cali, J. Kim, S. Ghose, C. Alkan, and O. Mutlu, "Accelerating genome analysis: A primer on an ongoing journey," *IEEE Micro*, vol. 40, no. 5, pp. 65–75, 2020.

[50] D. S. Cali, G. S. Kalsi, Z. Bingöl, C. Firtina, L. Subramanian, J. S. Kim, R. Ausavarungnirun, M. Alser, J. Gomez-Luna, A. Boroumand *et al.*, "Genasm: A high-performance, low-power approximate string matching acceleration framework for genome sequence analysis," in *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2020, pp. 951–966.

[51] M. K. Qureshi, A. Jaleel, Y. N. Patt, S. C. Steely, and J. Emer, "Adaptive insertion policies for high performance caching," *ACM SIGARCH Computer Architecture News*, vol. 35, no. 2, pp. 381–391, 2007.

[52] M. K. Qureshi, M. A. Suleman, and Y. N. Patt, "Line distillation: Increasing cache capacity by filtering unused words in cache lines," in *2007 IEEE 13th International Symposium on High Performance Computer Architecture.* IEEE, 2007, pp. 250–259.

[53] S. W. Keckler, W. J. Dally, B. Khailany, M. Garland, and D. Glasco, "Gpus and the future of parallel computing," *IEEE micro*, vol. 31, no. 5, pp. 7–17, 2011.

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

[55] M. A. Suleman, O. Mutlu, M. K. Qureshi, and Y. N. Patt, "Accelerating critical section execution with asymmetric multicore architectures," *IEEE micro*, vol. 30, no. 1, pp. 60–70, 2010.

[56] L. Subramanian, V. Seshadri, Y. Kim, B. Jaiyen, and O. Mutlu, "Mise: Providing performance predictability and improving fairness in shared main memory systems," in *2013 IEEE 19th International Symposium on High Performance Computer Architecture (HPCA).* IEEE, 2013, pp. 639–650.

[57] Y. Koh, "Nand flash scaling beyond 20nm," in *2009 IEEE International Memory Workshop.* IEEE, 2009, pp. 1–3.

[58] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Error patterns in mlc nand flash memory: Measurement, characterization, and analysis," in *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE).* IEEE, 2012, pp. 521–526.

[59] ——, "Threshold voltage distribution in mlc nand flash memory: Characterization, analysis, and modeling," in *2013 Design, Automation & Test in Europe Conference & Exhibition (DATE).* IEEE, 2013, pp. 1285–1290.

[60] V. Lee *et al.*, "Accelerating big data processing with hadoop, spark and memcached on in-memory cluster computing," *IEEE Transactions*, 2017.

[61] M. Halpern, Y. Zhu, and V. J. Reddi, "Mobile cpu's rise to power: Quantifying the impact of generational mobile cpu design trends on performance, energy, and user satisfaction," in *2016*

IEEE International Symposium on High Performance Computer Architecture (HPCA). IEEE, 2016, pp. 64–76.

[62] M. Kang, S. Lim, S. Gonugondla, and N. R. Shanbhag, "An in-memory vlsi architecture for convolutional neural networks," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 8, no. 3, pp. 494–505, 2018.

[63] A. Sampson *et al.*, "Enerj: Approximate data types for safe and general low-power computation," in *Proceedings of the 32nd ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2011.

[64] H. Zhao, L. Xue, P. Chi, and J. Zhao, "Approximate image storage with multi-level cell stt-mram main memory," in *2017 IEEE/ACM International Conference on Computer-Aided Design (ICCAD).* IEEE, 2017, pp. 268–275.

[65] A. Ranjan, S. Venkataramani, X. Fong, K. Roy, and A. Raghunathan, "Approximate storage for energy efficient spintronic memories," in *Proceedings of the 52nd Annual Design Automation Conference*, 2015, pp. 1–6.

[66] A. M. H. Monazzah, M. Shoushtari, S. G. Miremadi, A. M. Rahmani, and N. Dutt, "Quark: Quality-configurable approximate stt-mram cache by fine-grained tuning of reliability-energy knobs," in *2017 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED).* IEEE, 2017, pp. 1–6.

[67] J. Xu and S. Swanson, "Nova: A log-structured file system for hybrid volatile/non-volatile main memories," in *14th USENIX Conference on File and Storage Technologies (FAST)*, 2016.

[68] N. Sayed, F. Oboril, A. Shirvanian, R. Bishnoi, and M. B. Tahoori, "Exploiting stt-mram for approximate computing," in *2017 22nd IEEE European Test Symposium (ETS).* IEEE, 2017, pp. 1–6.

[69] J. Dean *et al.*, "Large scale distributed deep networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems*, 2012.

[70] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, 2017.

[71] J. Preskill, "Quantum computing in the nisq era and beyond," *Quantum*, vol. 2, 2018.

[72] K. Kim *et al.*, "3d stacked memory architectures for multi-core processors," in *Proceedings of the 35th Annual International Symposium on Computer Architecture*, 2008.

[73] M. M. S. Aly, T. F. Wu, A. Bartolo, Y. H. Malviya, W. Hwang, G. Hills, I. Markov, M. Wootters, M. M. Shulaker, H.-S. P. Wong *et al.*, “The n3xt approach to energy-efficient abundant-data computing,” *Proceedings of the IEEE*, vol. 107, no. 1, pp. 19–48, 2018.

[74] J. L. Hennessy and D. A. Patterson, *Computer Architecture: A Quantitative Approach.* Elsevier, 2011.

[75] R. H. Dennard, “Field-effect transistor memory,” U.S. Patent 3 387 286, 1968, https://lens.org/167-699-520-357-803.

[76] S. Shiratake, “Scaling and performance challenges of future dram,” in *2020 IEEE international memory workshop (IMW).* IEEE, 2020, pp. 1–3.

[77] K. K. Min, S. Hwang, J.-H. Lee, and B.-G. Park, “Vertical inner gate transistors for 4f 2 dram cell,” *IEEE Transactions on Electron Devices*, vol. 67, no. 3, pp. 944–948, 2020.

[78] I. Bhati, M.-T. Chang, Z. Chishti, S.-L. Lu, and B. Jacob, “Dram refresh mechanisms, penalties, and trade-offs,” *IEEE Transactions on Computers*, vol. 65, no. 1, pp. 108–121, 2015.

[79] Y.-H. Gong and S. W. Chung, “Exploiting refresh effect of dram read operations: A practical approach to low-power refresh,” *IEEE Transactions on Computers*, vol. 65, no. 5, pp. 1507–1517, 2015.

[80] S. Mittal and J. S. Vetter, “A survey of software techniques for using non-volatile memories for storage and main memory systems,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 5, pp. 1537–1550, 2015.

[81] T. Nirshl, B. Wicht, and D. Landsiedel, “High speed, low power design rules for sram pre-charge and self-tming under technology variation,” in *Proc. Intern. Workshop of Power and Time Modeling Optimization and Simulation.* Citeseer, 2001.

[82] H. M. D. Kabir and M. Chan, “Sram precharge system for reducing write power,” *HKIE transactions*, vol. 22, no. 1, pp. 1–8, 2015.

[83] F. Masuoka, M. Momodomi, Y. Iwata, and R. Shirota, “New ultra high density eprom and flash eeprom with nand structure cell,” in *1987 International Electron Devices Meeting.* IEEE, 1987, pp. 552–555.

[84] S. M. Sze and K. K. Ng, *Physics of Semiconductor Devices*, 3rd ed. John Wiley & Sons, 2006.

[85] R. Bishnoi, M. Ebrahimi, F. Oboril, and M. B. Tahoori, “Improving write performance for stt-mram,” *IEEE Transactions on Magnetics*, vol. 52, no. 8, pp. 1–11, 2016.

[86] H. Naeimi, C. Agustin, and A. Raychowdhury, “STTRAM scaling and retention failure,” *ResearchGate*, 2016. [Online]. Available: https://www.researchgate.net/publication/304496377_Sttram_scaling_and_retention_failure

[87] E. Aliagha, A. M. H. Monazzah, and H. Farbeh, “React: Read/write error rate aware coding technique for emerging stt-mram caches,” *IEEE Transactions on Magnetics*, vol. 55, no. 5, pp. 1–8, 2019.

[88] E. Cheshmikhani, H. Farbeh, and H. Asadi, “3rset: Read disturbance rate reduction in stt-mram caches by selective tag comparison,” *IEEE Transactions on Computers*, vol. 71, no. 6, pp. 1305–1319, 2021.

[89] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, “Architecting phase change memory as a scalable dram alternative,” in *Proceedings of the 36th annual international symposium on Computer architecture*, 2009, pp. 2–13.

[90] M. K. Qureshi, V. Srinivasan, and J. A. Rivers, “Scalable high performance main memory system using phase-change memory technology,” in *Proceedings of the 36th annual international symposium on Computer architecture*, 2009, pp. 24–33.

[91] M. Wuttig and N. Yamada, “Phase-change materials for rewriteable data storage,” *Nature Materials*, vol. 6, no. 11, pp. 824–832, 2007.

[92] D. Ielmini and H. Wong, “In-memory computing with resistive switching devices. nat. electron. 1, 333–343 (2018).”

[93] L. Xie, H. A. Du Nguyen, J. Yu, A. Kaichouhi, M. Taouil, M. AlFailakawi, and S. Hamdioui, “Scouting logic: A novel memristor-based logic design for resistive computing,” in *2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. IEEE, 2017, pp. 176–181.

[94] C. Sansoè, M. Tranchero *et al.*, “Use of fram memories in spacecrafts,” *Ferroelectrics-Applications*, pp. 213–230, 2011.

[95] Fujitsu Semiconductor Limited, *FRAM Guide Book*, 6th ed. Fujitsu Semiconductor Limited, 2010, 6th ed.

[96] J. Rodriguez, K. Remack, J. Gertas, L. Wang, C. Zhou, K. Boku, J. Rodriguez-Latorre, K. Udayakumar, S. Summerfelt, T. Moise *et al.*, “Reliability of ferroelectric random access memory embedded

within 130nm cmos," in *2010 IEEE International Reliability Physics Symposium.* IEEE, 2010, pp. 750–758.

[97] T. Eshita, T. Tamura, and Y. Arimoto, "Ferroelectric random access memory (fram) devices," in *Advances in non-volatile memory and storage technology.* Elsevier, 2014, pp. 434–454.

[98] A. Sheikholeslami and P. G. Gulak, "A survey of circuit innovations in ferroelectric random-access memories," *Proceedings of the IEEE*, vol. 88, no. 5, pp. 667–689, 2000.

[99] J. Müller, P. Polakowski, S. Mueller, and T. Mikolajick, "Ferroelectric hafnium oxide based materials and devices: Assessment of current status and future prospects," *ECS Journal of Solid State Science and Technology*, vol. 4, no. 5, p. N30, 2015.

[100] B. Sun, D. Liu, L. Yu, J. Li, H. Liu, W. Zhang, and T. Torng, "Mram co-designed processing-in-memory cnn accelerator for mobile and iot applications," *arXiv preprint arXiv:1811.12179*, 2018.

[101] M. Imani, S. Gupta, Y. Kim, M. Zhou, and T. Rosing, "Digitalpim: Digital-based processing in-memory for big data acceleration," in *Proceedings of the 2019 on Great Lakes Symposium on VLSI*, 2019, pp. 429–434.

[102] O. Leitersdorf, R. Ronen, and S. Kvatinsky, "Multpim: Fast stateful multiplication for processing-in-memory," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1647–1651, 2021.

[103] X. Peng, R. Liu, and S. Yu, "Optimizing weight mapping and data flow for convolutional neural networks on rram based processing-in-memory architecture," in *2019 IEEE International Symposium on Circuits and Systems (ISCAS).* IEEE, 2019, pp. 1–5.

[104] A. Lu, X. Peng, Y. Luo, S. Huang, and S. Yu, "A runtime reconfigurable design of compute-in-memory–based hardware accelerator for deep learning inference," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 26, no. 6, pp. 1–18, 2021.

[105] A. Kazemi, M. M. Sharifi, Z. Zou, M. Niemier, X. S. Hu, and M. Imani, "Mimhd: Accurate and efficient hyperdimensional inference using multi-bit in-memory computing," in *2021 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED).* IEEE, 2021, pp. 1–6.

[106] S. Roy, M. Ali, and A. Raghunathan, "Pim-dram: Accelerating machine learning workloads using processing in commodity dram," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 11, no. 4, pp. 701–710, 2021.

[107] Y. Long, D. Kim, E. Lee, P. Saha, B. A. Mudassar, X. She, A. I. Khan, and S. Mukhopadhyay, "A ferroelectric fet-based processing-in-memory architecture for dnn acceleration," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 5, no. 2, pp. 113–122, 2019.

[108] W. J. Lee, C. H. Kim, Y. Paik, J. Park, I. Park, and S. W. Kim, "Design of processing-"inside"-memory optimized for dram behaviors," *IEEE Access*, vol. 7, pp. 82 633–82 648, 2019.

[109] A. Olgun, J. G. Luna, K. Kanellopoulos, B. Salami, H. Hassan, O. Ergin, and O. Mutlu, "Pidram: A holistic end-to-end fpga-based framework for processing-in-dram," *ACM Transactions on Architecture and Code Optimization*, vol. 20, no. 1, pp. 1–31, 2022.

[110] M. Drumond, A. Daglis, N. Mirzadeh, D. Ustiugov, J. Picorel, B. Falsafi, B. Grot, and D. Pnevmatikatos, "Algorithm/architecture co-design for near-memory processing," *ACM SIGOPS Operating Systems Review*, vol. 52, no. 1, pp. 109–122, 2018.

[111] N. Park, S. Ryu, J. Kung, and J.-J. Kim, "High-throughput near-memory processing on cnns with 3d hbm-like memory," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 26, no. 6, pp. 1–20, 2021.

[112] Y. Kwon and M. Rhu, "Beyond the memory wall: A case for memory-centric hpc system for deep learning," in *2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2018, pp. 148–161.

[113] S. Ghose, A. Boroumand, J. S. Kim, J. Gómez-Luna, and O. Mutlu, "Processing-in-memory: A workload-driven perspective," *IBM Journal of Research and Development*, vol. 63, no. 6, pp. 3–1, 2019.

[114] J. Liu, H. Zhao, M. A. Ogleari, D. Li, and J. Zhao, "Processing-in-memory for energy-efficient neural network training: A heterogeneous approach," in *2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2018, pp. 655–668.

[115] Y. Wang, W. Chen, J. Yang, and T. Li, "Towards memory-efficient allocation of cnns on processing-in-memory architecture," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 6, pp. 1428–1441, 2018.

[116] Y. Huang, L. Zheng, P. Yao, J. Zhao, X. Liao, H. Jin, and J. Xue, "A heterogeneous pim hardware-software co-design for energy-efficient graph processing," in *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2020, pp. 684–695.

[117] S. Gupta, M. Imani, H. Kaur, and T. S. Rosing, "Nnpim: A processing in-memory architecture for neural network acceleration," *IEEE Transactions on Computers*, vol. 68, no. 9, pp. 1325–1337, 2019.

[118] A. Roohi, S. Angizi, D. Fan, and R. F. DeMara, "Processing-in-memory acceleration of convolutional neural networks for energy-effciency, and power-intermittency resilience," in *20th International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2019, pp. 8–13.

[119] C.-C. Lin, C.-L. Lee, J.-K. Lee, H. Wang, and M.-Y. Hung, "Accelerate binarized neural networks with processing-in-memory enabled by risc-v custom instructions," in *50th International Conference on Parallel Processing Workshop*, 2021, pp. 1–8.

[120] B. Jiao, H. Zhu, J. Zhang, S. Wang, X. Kang, L. Zhang, M. Wang, and C. Chen, "Computing utilization enhancement for chiplet-based homogeneous processing-in-memory deep learning processors," in *Proceedings of the 2021 on Great Lakes Symposium on VLSI*, 2021, pp. 241–246.

[121] S. Jung, H. Lee, S. Myung, H. Kim, S. K. Yoon, S.-W. Kwon, Y. Ju, M. Kim, W. Yi, S. Han *et al.*, "A crossbar array of magnetoresistive memory devices for in-memory computing," *Nature*, vol. 601, no. 7892, pp. 211–216, 2022.

[122] M. S. Hosseini, M. Ebrahimi, P. Yaghini, and N. Bagherzadeh, "Near volatile and non-volatile memory processing in 3d systems," *IEEE Transactions on Emerging Topics in Computing*, vol. 10, no. 3, pp. 1657–1664, 2021.

[123] J. Ahn, S. Yoo, O. Mutlu, and K. Choi, "Pim-enabled instructions: A low-overhead, locality-aware processing-in-memory architecture," *ACM SIGARCH Computer Architecture News*, vol. 43, no. 3S, pp. 336–348, 2015.

[124] A. Pattnaik, X. Tang, A. Jog, O. Kayiran, A. K. Mishra, M. T. Kandemir, O. Mutlu, and C. R. Das, "Scheduling techniques for gpu architectures with processing-in-memory capabilities," in *Proceedings of the 2016 International Conference on Parallel Architectures and Compilation*, 2016, pp. 31–44.

[125] J. Zhang, Y. Zha, N. Beckwith, B. Liu, and J. Li, "Meg: A riscv-based system emulation infrastructure for near-data processing using fpgas and high-bandwidth memory," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 13, no. 4, pp. 1–24, 2020.

[126] P. Gu, X. Xie, Y. Ding, G. Chen, W. Zhang, D. Niu, and Y. Xie, "ipim: Programmable in-memory image processing accelerator using near-bank architecture," in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2020, pp. 804–817.

[127] J. Nider, C. Mustard, A. Zoltan, J. Ramsden, L. Liu, J. Grossbard, M. Dashti, R. Jodin, A. Ghiti, J. Chauzi *et al.*, "A case study of {Processing-in-Memory} in {off-the-Shelf} systems," in *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, 2021, pp. 117–130.

[128] C. Giannoula, I. Fernandez, J. Gómez-Luna, N. Koziris, G. Goumas, and O. Mutlu, "Towards efficient sparse matrix vector multiplication on real processing-in-memory architectures," *ACM SIGMETRICS Performance Evaluation Review*, vol. 50, no. 1, pp. 33–34, 2022.

[129] H. Kang, P. B. Gibbons, G. E. Blelloch, L. Dhulipala, Y. Gu, and C. McGuffey, "The processing-in-memory model," in *Proceedings of the 33rd ACM Symposium on Parallelism in Algorithms and Architectures*, 2021, pp. 295–306.

[130] M. Zhou, M. Imani, Y. Kim, S. Gupta, and T. Rosing, "Dp-sim: A full-stack simulation infrastructure for digital processing in-memory architectures," in *Proceedings of the 26th Asia and South Pacific Design Automation Conference*, 2021, pp. 639–644.

[131] S. Xu, X. Chen, Y. Wang, Y. Han, X. Qian, and X. Li, "Pimsim: A flexible and detailed processing-in-memory simulator," *IEEE Computer Architecture Letters*, vol. 18, no. 1, pp. 6–9, 2018.

[132] A. Lu, X. Peng, W. Li, H. Jiang, and S. Yu, "Neurosim simulator for compute-in-memory hardware accelerator: Validation and benchmark," *Frontiers in artificial intelligence*, vol. 4, p. 659060, 2021.

[133] X. Xie, P. Gu, J. Huang, Y. Ding, and Y. Xie, "Mpu-sim: A simulator for in-dram near-bank processing architectures," *IEEE Computer Architecture Letters*, vol. 21, no. 1, pp. 1–4, 2021.

[134] C. Yu, S. Liu, and S. Khan, "Multipim: A detailed and configurable multi-stack processing-in-memory simulator," *IEEE Computer Architecture Letters*, vol. 20, no. 1, pp. 54–57, 2021.

[135] G. F. Oliveira, J. Gómez-Luna, L. Orosa, S. Ghose, N. Vijaykumar, I. Fernandez, M. Sadrosadati, and O. Mutlu, "Damov: A new methodology and benchmark suite for evaluating data movement bottlenecks," *IEEE Access*, vol. 9, pp. 134 457–134 502, 2021.

[136] D. Sanchez and C. Kozyrakis, "Zsim: Fast and accurate microarchitectural simulation of thousand-core systems," *ACM SIGARCH Computer architecture news*, vol. 41, no. 3, pp. 475–486, 2013.

[137] Y. Kim, W. Yang, and O. Mutlu, "Ramulator: A fast and extensible dram simulator," *IEEE Computer architecture letters*, vol. 15, no. 1, pp. 45–49, 2015.

[138] S. K. Gonugondla, M. Kang, and N. R. Shanbhag, “A variation-tolerant in-memory machine learning classifier via on-chip training,” *IEEE Journal of Solid-State Circuits*, vol. 53, no. 11, pp. 3163–3173, 2018.

[139] B. Razavi, *Design of Analog CMOS Integrated Circuits.* New York: McGraw-Hill, 2001.

[140] A. Kneip and D. Bol, “Impact of analog non-idealities on the design space of 6t-sram current-domain dot-product operators for in-memory computing,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 5, pp. 1931–1944, 2021.

[141] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[142] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[143] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[144] A. Krizhevsky, “Learning multiple layers of features from tiny images,” *University of Toronto*, 2012.

[145] C. S. Vaucher, I. Ferencic, M. Locher, S. Sedvallson, U. Voegeli, and Z. Wang, “A family of low-power truly modular programmable dividers in standard 0.35-/spl mu/m cmos technology,” *IEEE Journal of Solid-State Circuits*, vol. 35, no. 7, pp. 1039–1045, 2000.

[146] C.-J. Jhang, C.-X. Xue, J.-M. Hung, F.-C. Chang, and M.-F. Chang, “Challenges and trends of sram-based computing-in-memory for ai edge devices,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 5, pp. 1773–1786, 2021.

[147] J.-s. Seo, J. Saikia, J. Meng, W. He, H.-s. Suh, Y. Liao, A. Hasssan, I. Yeo *et al.*, “Digital versus analog artificial intelligence accelerators: Advances, trends, and emerging designs,” *IEEE Solid-State Circuits Magazine*, vol. 14, no. 3, pp. 65–79, 2022.

[148] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[149] L. Deng, “The mnist database of handwritten digit images for machine learning research,” *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[150] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition.* Ieee, 2009, pp. 248–255.

[151] Y. Chen, Y. Xie, L. Song, F. Chen, and T. Tang, "A survey of accelerator architectures for deep neural networks," *Engineering*, vol. 6, no. 3, pp. 264–274, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2095809919306356

[152] X. Zhao, Y. Wang, X. Cai, C. Liu, and L. Zhang, "Linear symmetric quantization of neural networks for low-precision integer hardware," in *International Conference on Learning Representations*, 2020.

[153] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[154] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[155] L. van der Maaten and G. Hinton, "Viualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 11 2008.

[156] A. Jaiswal, I. Chakraborty, A. Agrawal, and K. Roy, "8t sram cell as a multibit dot-product engine for beyond von neumann computing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, pp. 2556–2567, 2019.

[157] S. Yin, Z. Jiang, J.-S. Seo, and M. Seok, "Xnor-sram: In-memory computing sram macro for binary/ternary deep neural networks," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 6, pp. 1733–1743, 2020.

[158] P. N. Whatmough, S. K. Lee, H. Lee, S. Rama, D. Brooks, and G.-Y. Wei, "14.3 a 28nm soc with a 1.2ghz 568nj/prediction sparse deep-neural-network engine with >0.1 timing error rate tolerance for iot applications," in *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, 2017, pp. 242–243.

[159] B. Moons and M. Verhelst, “A 0.3–2.6 tops/w precision-scalable processor for real-time large-scale convnets,” in *2016 IEEE Symposium on VLSI Circuits (VLSI-Circuits)*, 2016, pp. 1–2.

[160] J. K. Kim, P. Knag, T. Chen, and Z. Zhang, “A 640m pixel/s 3.65mw sparse event-driven neuromorphic object recognition processor with on-chip learning,” in *2015 Symposium on VLSI Circuits (VLSI Circuits)*, 2015, pp. C50–C51.

[161] D. Tsipras, S. Santurkar, L. Engstrom, A. Ilyas, and A. Madry, “From imagenet to image classification: Contextualizing progress on benchmarks,” in *International Conference on Machine Learning*. PMLR, 2020, pp. 9625–9635.

[162] “Irds 2022 beyond cmos and emerging materials integration,” Online, 2022, accessed on: November 7, 2023. [Online]. Available: https://irds.ieee.org/editions/2022

[163] S. M. Nair, R. Bishnoi, A. Vijayan, and M. B. Tahoori, “Dynamic faults based hardware trojan design in stt-mram,” in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2020, pp. 933–938.

[164] S. Swami and K. Mohanram, “Reliable nonvolatile memories: Techniques and measures,” *IEEE Design & Test*, vol. 34, no. 3, pp. 31–41, 2017.

[165] S. Seyedfaraji, A. M. Hajisadeghi, J. Talafy, and H. R. Zarandi, “Dysco: Dynamic stepper current injector to improve write performance in stt-ram memories,” *Microprocessors and Microsystems*, vol. 73, p. 102963, 2020.

[166] E. Garzon, R. De Rose, F. Crupi, L. Trojman, G. Finocchio, M. Carpentieri, and M. Lanuzza, “Assessment of stt-mrams based on double-barrier mtjs for cache applications by means of a device-to-system level simulation framework,” *Integration*, vol. 71, pp. 56–69, 2020.

[167] R. Saha, Y. P. Pundir, and P. K. Pal, “Design of an area and energy-efficient last-level cache memory using stt-mram,” *Journal of Magnetism and Magnetic Materials*, vol. 529, p. 167882, 2021.

[168] E. Cheshmikhani, H. Farbeh, and H. Asadi, “Robin: Incremental oblique interleaved ecc for reliability improvement in stt-mram caches,” in *Proceedings of the 24th Asia and South Pacific Design Automation Conference*, 2019, pp. 173–178.

[169] N. Mahdavi, F. Razaghian, and H. Farbeh, “Data block manipulation for error rate reduction in stt-mram based main memory,” *The Journal of Supercomputing*, vol. 78, no. 11, pp. 13 342–13 372, 2022.

[170] E. Kültürsay, M. Kandemir, A. Sivasubramaniam, and O. Mutlu, "Evaluating stt-ram as an energy-efficient main memory alternative," in *2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2013, pp. 256–267.

[171] D. Sanchez and C. Kozyrakis, "Zsim: Fast and accurate microarchitectural simulation of thousand-core systems," *ACM SIGARCH Computer architecture news*, vol. 41, no. 3, pp. 475–486, 2013.

[172] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti *et al.*, "The gem5 simulator," *ACM SIGARCH computer architecture news*, vol. 39, no. 2, pp. 1–7, 2011.

[173] A. Hansson, N. Agarwal, A. Kolli, T. Wenisch, and A. N. Udipi, "Simulating dram controllers for future system architecture exploration," in *2014 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*. IEEE, 2014, pp. 201–210.

[174] "The spec cpu 2017 benchmark package," accessed: Jan 2024. [Online]. Available: https://www.SPEC.org/cpu2017

[175] X. Dong, C. Xu, Y. Xie, and N. P. Jouppi, "Nvsim: A circuit-level performance, energy, and area model for emerging nonvolatile memory," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 7, pp. 994–1007, 2012.

[176] M. Poremba and Y. Xie, "Nvmain: An architectural-level main memory simulator for emerging non-volatile memories," in *2012 IEEE Computer Society Annual Symposium on VLSI*. IEEE, 2012, pp. 392–397.

[177] M. Poremba, T. Zhang, and Y. Xie, "Nvmain 2.0: A user-friendly memory simulator to model (non-) volatile memory systems," *IEEE Computer Architecture Letters*, vol. 14, no. 2, pp. 140–143, 2015.

[178] J. Meza, J. Li, and O. Mutlu, "Evaluating row buffer locality in future non-volatile main memories," *arXiv preprint arXiv:1812.06377*, 2018.

[179] S. Chung, K.-M. Rho, S.-D. Kim, H.-J. Suh, D.-J. Kim, H.-J. Kim, S.-H. Lee, J.-H. Park, H.-M. Hwang, S.-M. Hwang, J.-Y. Lee, Y.-B. An, J.-U. Yi, Y.-H. Seo, D.-H. Jung, M.-S. Lee, S.-H. Cho, J.-N. Kim, G.-J. Park, G. Jin, A. Driskill-Smith, V. Nikitin, A. Ong, X. Tang, Y. Kim, J.-S. Rho, S.-K. Park, S.-W. Chung, J.-G. Jeong, and S.-J. Hong, "Fully integrated 54nm stt-ram with the smallest bit cell dimension for high density memory application," in *2010 International Electron Devices Meeting*, 2010, pp. 12.7.1–12.7.4.

[180] "1 gb non-volatile st-ddr4 spin-transfer torque mram," accessed: Jan 2024. [Online]. Available: https://www.everspin.com/family/emd4e001g

[181] K. Chandrasekar, C. Weis, Y. Li, B. Akesson, N. Wehn, and K. Goossens, "Drampower: Open-source dram power & energy estimation tool," *URL:http://www. drampower. info*, vol. 22, 2012.

[182] A. Brokalakis, N. Tampouratzis, A. Nikitakis, I. Papaefstathiou, S. Andrianakis, D. Pau, E. Plebani, M. Paracchini, M. Marcon, I. Sourdis, P. R. Geethakumari, M. C. Palacios, M. A. Anton, and A. Szasz, "Cossim: An open-source integrated solution to address the simulator gap for systems of systems," in *2018 21st Euromicro Conference on Digital System Design (DSD)*, 2018, pp. 115–120.

[183] L. Zhang, Y. Cheng, W. Kang, L. Torres, Y. Zhang, A. Todri-Sanial, and W. Zhao, "Addressing the thermal issues of stt-mram from compact modeling to design techniques," *IEEE Transactions on Nanotechnology*, vol. 17, no. 2, pp. 345–352, 2018.

[184] S. S. Faraji, J. Talafy, A. M. Hajisadeghi, and H. R. Zarandi, "Duster: Dual source write termination method for stt-ram memories," in *2018 21st Euromicro Conference on Digital System Design (DSD)*. IEEE, 2018, pp. 182–189.

[185] H. Li, X. Wang, Z.-L. Ong, W.-F. Wong, Y. Zhang, P. Wang, and Y. Chen, "Performance, power, and reliability tradeoffs of stt-ram cell subject to architecture-level requirement," *IEEE Transactions on Magnetics*, vol. 47, no. 10, pp. 2356–2359, 2011.

[186] E. Cheshmikhani, H. Farbeh, S. G. Miremadi, and H. Asadi, "Ta-lrw: A replacement policy for error rate reduction in stt-mram caches," *IEEE Transactions on Computers*, vol. 68, no. 3, pp. 455–470, 2018.

[187] H. Farkhani, M. Tohidi, A. Peiravi, J. K. Madsen, and F. Moradi, "Stt-ram energy reduction using self-referenced differential write termination technique," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 2, pp. 476–487, 2016.

[188] U. Roy, T. Pramanik, L. F. Register, and S. K. Banerjee, "Write error rate of spin-transfer-torque random access memory including micromagnetic effects using rare event enhancement," *IEEE Transactions on Magnetics*, vol. 52, no. 10, pp. 1–6, 2016.

[189] T. Zheng, J. Park, M. Orshansky, and M. Erez, "Variable-energy write stt-ram architecture with bit-wise write-completion monitoring," in *International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE, 2013, pp. 229–234.

[190] J. Song, H. Dixit, B. Behin-Aein, C. H. Kim, and W. Taylor, “Impact of process variability on write error rate and read disturbance in stt-mram devices,” *IEEE Transactions on Magnetics*, vol. 56, no. 12, pp. 1–11, 2020.

[191] J. J. Nowak, R. P. Robertazzi, J. Z. Sun, G. Hu, J.-H. Park, J. Lee, A. J. Annunziata, G. P. Lauer, R. Kothandaraman, E. J. O’Sullivan *et al.*, “Dependence of voltage and size on write error rates in spin-transfer torque magnetic random-access memory,” *IEEE Magnetics Letters*, vol. 7, pp. 1–4, 2016.

[192] T. Kawahara, K. Ito, R. Takemura, and H. Ohno, “Spin-transfer torque ram technology: Review and prospect,” *Microelectronics Reliability*, vol. 52, no. 4, pp. 613–627, 2012.

[193] E. Cheshmikhani, H. Farbeh, and H. Asadi, “A system-level framework for analytical and empirical reliability exploration of stt-mram caches,” *IEEE Transactions on Reliability*, vol. 69, no. 2, pp. 594–610, 2019.

[194] C. Xu, D. Niu, X. Zhu, S. H. Kang, M. Nowak, and Y. Xie, “Device-architecture co-optimization of stt-ram based memory for low power embedded systems,” in *2011 IEEE/ACM International Conference on Computer-Aided Design (ICCAD).* IEEE, 2011, pp. 463–470.

[195] R. Saha, Y. P. Pundir, and P. K. Pal, “Comparative analysis of stt and sot based mrams for last level caches,” *Journal of Magnetism and Magnetic Materials*, vol. 551, p. 169161, 2022.

[196] R. Bishnoi, M. Ebrahimi, F. Oboril, and M. B. Tahoori, “Improving write performance for stt-mram,” *IEEE Transactions on Magnetics*, vol. 52, no. 8, pp. 1–11, 2016.

[197] D. Apalkov, A. Khvalkovskiy, S. Watts, V. Nikitin, X. Tang, D. Lottis, K. Moon, X. Luo, E. Chen, A. Ong *et al.*, “Spin-transfer torque magnetic random access memory (stt-mram),” *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 9, no. 2, pp. 1–35, 2013.

[198] T. D. Burd, T. A. Pering, A. J. Stratakos, and R. W. Brodersen, “A dynamic voltage scaled microprocessor system,” *IEEE Journal of solid-state circuits*, vol. 35, no. 11, pp. 1571–1580, 2000.

[199] S. H. Kulkarni and D. Sylvester, “High performance level conversion for dual v/sub dd/ design,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, no. 9, pp. 926–936, Sep 2004.

[200] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti *et al.*, “The gem5 simulator,” *ACM SIGARCH computer architecture news*, vol. 39, no. 2, pp. 1–7, 2011.

[201] S. S. Faraji, A. M. Hajisadeghi, and H. Zarandi, "Tamper: Thermal assistant method to improve write performance in stt-ram memories," in *2019 27th Iranian Conference on Electrical Engineering (ICEE).* IEEE, 2019, pp. 2039–2044.

[202] H. Farbeh, A. M. H. Monazzah, E. Aliagha, and E. Cheshmikhani, "A-cache: Alternating cache allocation to conduct higher endurance in nvm-based caches," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 66, no. 7, pp. 1237–1241, 2018.

[203] Z. Azad, H. Farbeh, A. M. H. Monazzah, and S. G. Miremadi, "Aware: Adaptive way allocation for reconfigurable eccs to protect write errors in stt-ram caches," *IEEE Transactions on Emerging Topics in Computing*, vol. 7, no. 3, pp. 481–492, 2017.

[204] A. M. H. Monazzah, A. M. Rahmani, A. Miele, and N. Dutt, "Cast: Content-aware stt-mram cache write management for different levels of approximation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 12, pp. 4385–4398, 2020.

[205] Y. Zhang, W. Zhao, Y. Lakys, J.-O. Klein, J.-V. Kim, D. Ravelosona, and C. Chappert, "Compact modeling of perpendicular-anisotropy cofeb/mgo magnetic tunnel junctions," *IEEE transactions on Electron devices*, vol. 59, no. 3, pp. 819–826, 2012.

[206] M. Kang, S. Gonugondla, A. Patil, and N. Shanbhag, "A 481pj/decision 3.4m decision/s multifunctional deep in-memory inference processor using standard 6t sram array," *arXiv*, Oct 2016, accessed: Sep. 17, 2021. [Online]. Available: http://arxiv.org/abs/1610.07501

[207] M. Carbin, S. Misailovic, and M. C. Rinard, "Verifying quantitative reliability for programs that execute on unreliable hardware," *ACM SIGPLAN Notices*, vol. 48, no. 10, pp. 33–52, 2013.

[208] W. Kang, L. Zhang, J.-O. Klein, Y. Zhang, D. Ravelosona, and W. Zhao, "Reconfigurable codesign of stt-mram under process variations in deeply scaled technology," *IEEE Transactions on Electron Devices*, vol. 62, no. 6, pp. 1769–1777, 2015.