POST-CMOS MEMORY TECHNOLOGIES AND THEIR APPLICATIONS IN EMERGING COMPUTING MODELS

A Dissertation
Presented to
The Academic Faculty

By

Insik Yoon

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

Georgia Institute of Technology

August 2019

POST-CMOS MEMORY TECHNOLOGIES AND THEIR APPLICATIONS IN EMERGING COMPUTING MODELS

Approved by:

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

Dr. Asif Islam Khan
School of Electrical and Computer Engineering
Georgia Institute of Technology

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

Dr. Suman Datta
Department of Electrical Engineering
University of Notre Dame

Dr. Titash Rakshit
Advanced Logic Lab
Samsung Semiconductor

Date Approved: May 20, 2019

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my research advisor Dr. Arijit Raychowdhury for his support and guidance throughout my doctoral research. Without his encouragement, I would not be able to overcome the difficulties that I faced in the course of research.

I would also like to thank my committee members – Dr. Madhavan Swaminathan, Dr. Asif Khan, Dr. Shimeng Yu, Dr. Suman Datta, Dr. Titash Rakshit and Dr. Suman Datta for providing me guidance and insight to my research.

I would like to thank my research group members, Dr. Samantak Gangophadyay, Dr. Saad Bin Nasir, Dr. Anvesh Amaravati, Abhinav Parihar, Ningyuan Cao, Muya Chang, Aqeel Anwar, Bitan Bhar, Brian Crafton, Anupam Golder , Foroozan Karimzade and Rakshith Saligram, Dr. Kaushik Bhattacharyya, Dr. Yan Fang and Dr, Jong-Hyeok Yoon. Your help, support, input and friendship was a crucial part of my experience as a graduate student.

I would like to thank my family and friends. I am extremely grateful to my parents for their support. Lastly, I would like to thank my wife, Dr. Hyo-Jin Nam for unlimited support and patience.

SUMMARY

The objective of this proposed research is to take a holistic approach to the post-CMOS in/near-memory processing system design for machine learning and optimizations. We first address the current issues of Spin-Transfer Torque Magnetic Random Access Memory(STT-MRAM) and multi-bit ferroelectric FET in the device level. At the circuit level, the research shows how these issues shape the peripheral circuit of STT-MRAM and ferroelectric FET memory arrays. Lastly, at the system level, the research leads to the efficient memory architecture and system design that maximizes the benefits of STT-MRAM and ferroelectric FET while mitigating the current limitations of these devices. In the proposed research, we apply the in/near memory processing system design with STT-MRAM and ferroelectric FETs to various applications such as reinforcement learning with a drone, image classification with Deep Neural Network and least square minimization for image reconstruction. For the remaining part of this research, we will focus on near-memory processing system with STT-MRAM for reinforcement learning of a drone and evaluate the system to quantify how much benefits are expected in terms of latency, power and energy. From this project, we would like to show that near-memory processing system with non-volatile devices is a key enabler for real-time learning systems with stringent power and energy constraints.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 Motivation

In recent years, Post-CMOS memory technologies are extensively explored as the importance of deep neural network based machine learning and distributed optimization accelerators increases. Among other post-CMOS memory technologies, spin torque transfer magnetic Random Access Memory (STT-MRAM) and ferroelectric Field Effect Transistor(FerroFET) are viable candidates for deep learning and distributed optimization accelerators. Spin Transfer Torque Magnetic Random Access Memory (STT-MRAM) is an emerging memory technology which exhibits non-volatility, high density, high endurance and nano-second read and write times with no refresh operations. These attributes of STT-MRAM make it suitable as a DRAM replacement in near-memory architecture of deep learning based accelerator. Since STT-MRAM shows short read latency and no refresh power, application such as deep learning based Unmanned Aerial Vehicles (UAVs) with small power-constraints is a perfect application for STT-MRAM.

Ferroelectric FET (FerroFET) have recently received great interest for its application in non-volatile memory. It is CMOS compatible and retains ferroelectricity for thin films with thickness around 10 nm. By tuning the portion of switched ferroelectric domain, a ferroFET can exhibit multiple intermediate resistance states. Due to this feature, a single FerroFET device can be used as a analog multiplier by measuring current across FerroFET after applying voltage between the source and drain of the device. By embedding FerroFET, a processing element, in the memory subarray itself in so called processing-in-memory architecture, the system can solve memory bottleneck, a problem due to a large amount of data traffic between logic and memory blocks, of deep learning and optimization

accelerators.

In the dissertation, we propose in/near-memory processing system design with post-CMOS devices such as STT-MRAM and FerroFET for machine learning and optimization accelerators. First, we introduce the properties and characteristics of STT-MRAM and FerroFET and show why using these technologies can improve systems for deep learning and optimization. Then we identify the challenges of STT-MRAM and FerroFET in device & circuit level and provide potential solutions to the challenges. Lastly, we demonstrate the implementation of;

1. Near-memory system with STT-MRAM for Reinforcement Learning algorithm for a drone
2. In-memory system with FerroFET for distributed convex optimization via least squares method

to compare the system performance in latency, power and energy with state-of-the-art conventional system. In the conclusion, we show whether post-CMOS based in/near-memory system exhibits better performance compared to conventional system even with the limitations in post-CMOS devices.

In the next section, we present prior works on post-CMOS devices(mainly STT-MRAM and FerroFET) and challenges of STT-MRAM & FerroFET based system with in/near memory architectures.

1.2 Prior works

1.2.1 Post CMOS memory: Spin Transfer Torque Magnetic RAM (STT-MRAM)

It is well understood that next-generation memory-intensive ultra low power learning-based systems require a memory technology which shows;

1. high-density
2. low-standby power (hence eNVM)
3. acceptable R/W speeds

4. compatibility with a logic process both in terms of process thermal budget and voltage domains

This is required to ensure that the design, along with an eNVM, can take advantage of the numerous scaled high performance, low power digital logic blocks that are essential for any area and power constrained design like the one we have described in this paper. Compared to other NVMs such as Phase-change memory or resistive RAM, STT-MRAM exhibits better read/write latency [1][2] and is more mature than Ferroelectric FET based RAMs. Recent publications from leading foundries [3][4][5] have demonstrated MBs of STT-MRAM arrays with necessary peripheral circuits. Compared to STT-MRAMs, RRAMs show larger device-to-device and cycle-to-cycle variations making it hard to commercialize [6].

Table 1.1: Comparison between STT-MRAM [7][5] and competing technologies (EFlash [8][9][10], RRAM[11][12], PCRAM[13][14]

	SRAM	Eflash	STT-MRAM	RRAM	PCRAM
Cell size	80~100F^2	~6F^2	>6F^2	>4F^2	>4F^2
Non-volatility	No	Yes	Yes	Yes	Yes
Program voltage	< 1V	<~10V	< 1.5V	< 3V	< 1V
Write speed	~1ns	660 µs	~30ns	~ 1 µs	~80ns
Read speed	~1ns	45 µs	~10ns	~1 µs	~10ns
Endurance	10^{16}	$10^4 \sim 10^6$	10^{15}	10^{10}	10^{12}
Retention	N/A	10 yrs	10 yrs	10 yrs	10 yrs

Although our study investigates STT-MRAM based stacks, all eNVM suffer from high write latency and energy; and hence the algorithm-hardware co-design that we propose is applicable to similar other platforms. The STT-MRAM model parameters are summarized in Table 1.1.

The STT-MRAM bitcell consists of one access transistor and one Magnetic Tunnel Junction (MTJ) where a single bit of information is stored[15]. Typical MTJ stacks comprise of an insulator (MgO) which is sandwiched between a ”fixed” ferromagnetic layer (typically CoFeB based) whose magnetic moment is pinned to one direction and a ”free” ferromagnetic layer whose moment changes direction based on applied external current or magnetic field. Since MTJ exhibits TMR (Tunneling magnetoresistance)[16], the resis-

tance of the stack changes depending on the orientation of the ”free” layer, which in turn stored the data of the bit-cell. When the direction of the magnetic moment inside the free layer of an MTJ is anti-parallel to the fixed layer, the MTJ has high resistance and its state is defined as bit ”1” [15]. Likewise, when the direction of the magnetic moment in an MTJ is parallel to the magnetic moment of the fixed layer, the MTJ exhibits low resistance and it is defined as bit ”0”.

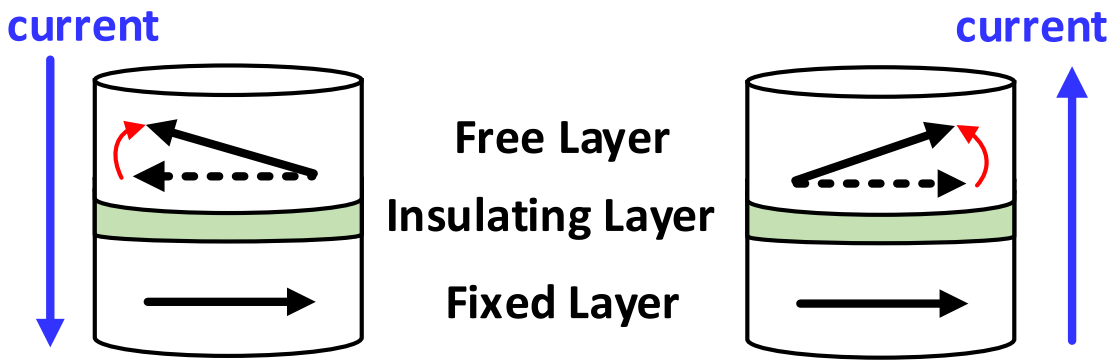


Figure 1.1: The direction of magnetic moment in free layer changes from (a)anti-parallel to parallel (b) parallel to anti-parallel to the direction of magnetic moment of fixed layer. The arrow in the free/fixed layer indicates the direction of magnetic moment.

Fig. 1.1 describes how the direction of magnetic moment in the free layer changes based on the current across the MTJ. Fig. 1.1 shows how the direction of magnetic moment in the free layer changes from (a) anti-parallel to parallel and (b) parallel to anti-parallel direction compared to the direction of magnetic moment in fixed layer. Since the fixed layer acts as a spin polarizer, the spin polarized electrons that pass the fixed layer exerts the torque on the magnetic moment in the free layer and causes a flip in the direction of the magnetic moment in fixed layer as shown in Fig. 1.1(a). When the current flows from the fixed layer to the free layer as shown in Fig. 1.1(b), the electrons with opposite spin are reflected back from the fixed layer and exerts a torque that changes the direction of the magnetic moment of the free layer to an anti-parallel direction with respect to the magnetic moment in the fixed layer. The alignment of the magnetic moment in the fixed and free layers determine the resistance across the MTJ. When the magnetic moments in the two

layers are anti-parallel to each other, the resistance across MTJ is high.

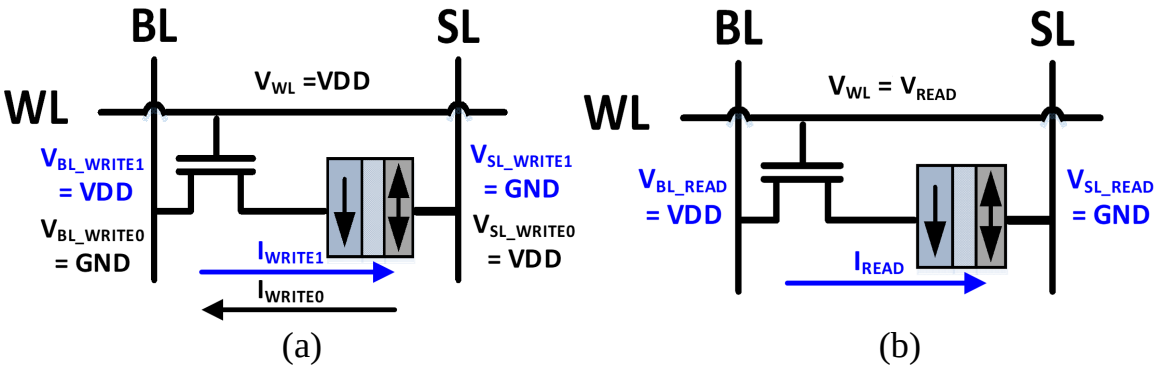


Figure 1.2: The STT-MRAM cell schematic of (a) write (b) read operation

Table 1.2: STT-MRAM array parameters as compiled from [5][4][17]

Technology	22nm FFL FinFET
TMR	180%
RA	9 $\Omega\mu m^2$
Density	8Mb
Cell architecture	1T 1MTJ
unit cell size	$9F^2$
Power supply(core)	1.0V
MTJ size	60~80nm
MTJ type	Perpendicular MTJ

A low resistance is achieved when both the magnetic moments are parallel to each other. The high/low resistance is mapped to 1/0. The bias conditions applied for the write and read operations are shown in Fig. 1.2. As shown in Fig. 1.2(a), the write operation is bi-directional. In case of writing a 1, the bit-line and the source line are set to VDD and GND and the write current flows from the fixed layer to the free layer of the MTJ. The biasing condition for writing a 0 is the opposite and is shown in Fig. 1.2(a). In case of read operations, the word-line is asserted to VREAD and the bitline and the source line are set to VDD and GND. This causes a weak current to flow across the MTJ and the resistance

state is sensed using either a constant current scheme or a BL discharge scheme [18]. Table 1.2 shows STT-MRAM array parameters from the silicon implementation of STT-MRAM.

1.2.2 Post CMOS memory: Ferroelectric FET

We explore FerroFETs as the technology of choice for implementing resistive cross-bar architectures that can accelerate linear algebraic operations. In particular, HfO_2 based Ferroelectric FETs (FerroFETs) have recently received great interest for its application in non-volatile memory (NVM) [19]. It is CMOS compatible and retains ferroelectricity for thin films with thickness around 10nm. By tuning the portion of switched ferroelectric domain, a FerroFET can exhibit multiple intermediate states, which has been used in neuromorphic computing [20, 21].

The operation of FerroFET as an multi-valued eNVM storage is different from a traditional binary memory [19] in that a series of weak pulses are applied to set the device in a desired state [20, 21]. Various pulse schemes are proposed to tune the state, including identical pulse schemes[22], pulse-width modulation schemes[23], and pulse-amplitude modulation schemes [21][24]. For illustration, Fig. 1.3 illustrates the operation with pulse-amplitude modulation scheme, which is used in this paper. Fig. 1.3(e) shows the applied pulse waveform. After each pulse, the percentage of switched ferroelectric domains is modified. The device states are shown in Fig. 1.3 (a)-(d). The device I_{DS}-V_{GS} corresponding to different states are shown in Fig. 1.3 (f), which shows the intermediate states. The different states could be sensed by applying a read pulse, V_R, the corresponding drain-to-source conductance, G_{DS}, can be sensed. Fig. 1.3 (g) shows the ideal G_{DS} as a function of applied pulse numbers. G_{DS} increases/decreases linearly with pulse number during potentiation/depression, respectively. A symmetrical potentiation/depression is necessary for high accuracy computation. The FerroFET model includes atomistic simulation of domain dynamics with a drift-diffusion based FET model. The simulation results closely match the experimental data and is shown in Fig. 1.4 where the different conductance levels are

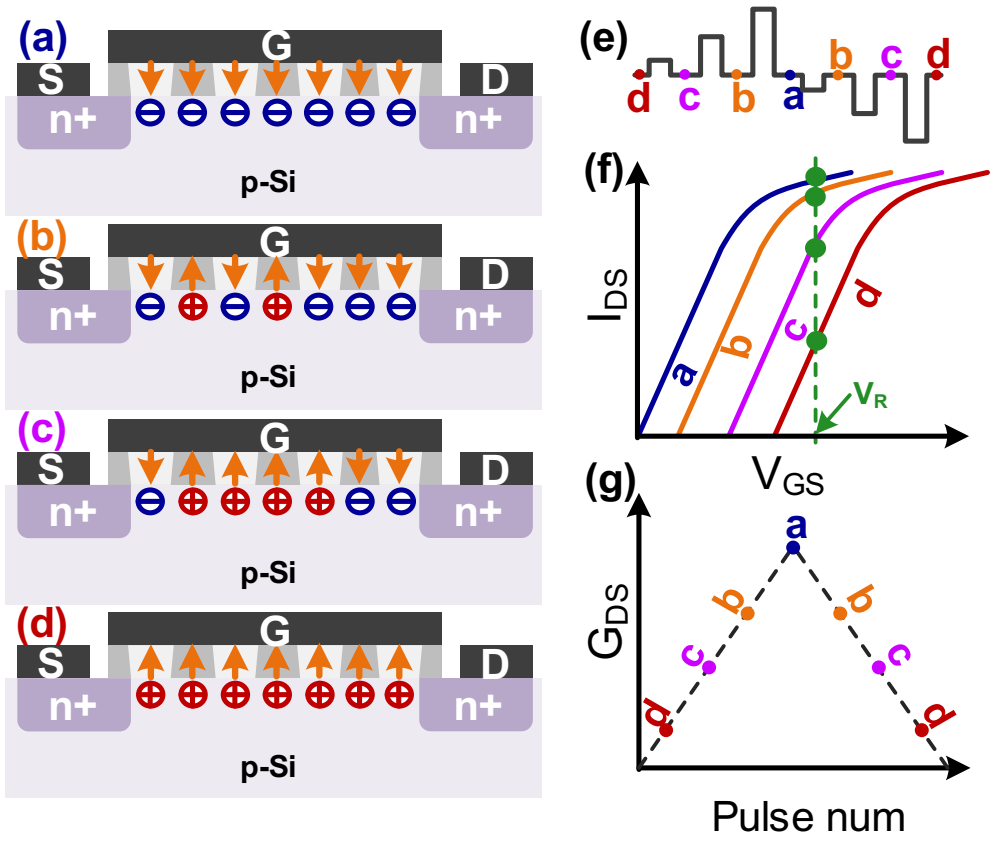


Figure 1.3: (a) – (d) show different FerroFET states, corresponding to different portions of ferroelectric domain switching. The yellow arrows indicate the polarization direction. The blue/red circles represent electron/hole, respectively. (e) shows the applied pulse amplitude modulation scheme. The states after each pulse are also illustrated. The initial state is assumed to be all polarizations are pointing toward the gate. (f) shows the I_{DS}-V_{GS} characteristics after each pulse. (g) shows the measured drain to source conductance as a function of applied pulse number. Here ideal case is presented, which shows linear and symmetrical potentiation and depression.

shown as a function of the number of programming pulses.

1.2.3 Challenges with STT-MRAM and FerroFET based system

Magnetic Coupling Across Bit-Cells in STT-MRAM

As STT-MRAM arrays become dense and the cell dimensions become smaller, the magnetic field coupling from ferromagnetic layers of one MTJs affect write and read operation of its neighboring bits. As shown in [25], scaling MTJ in a densely packed array causes program errors due to large stray field coupling. When MTJ scales down and they are

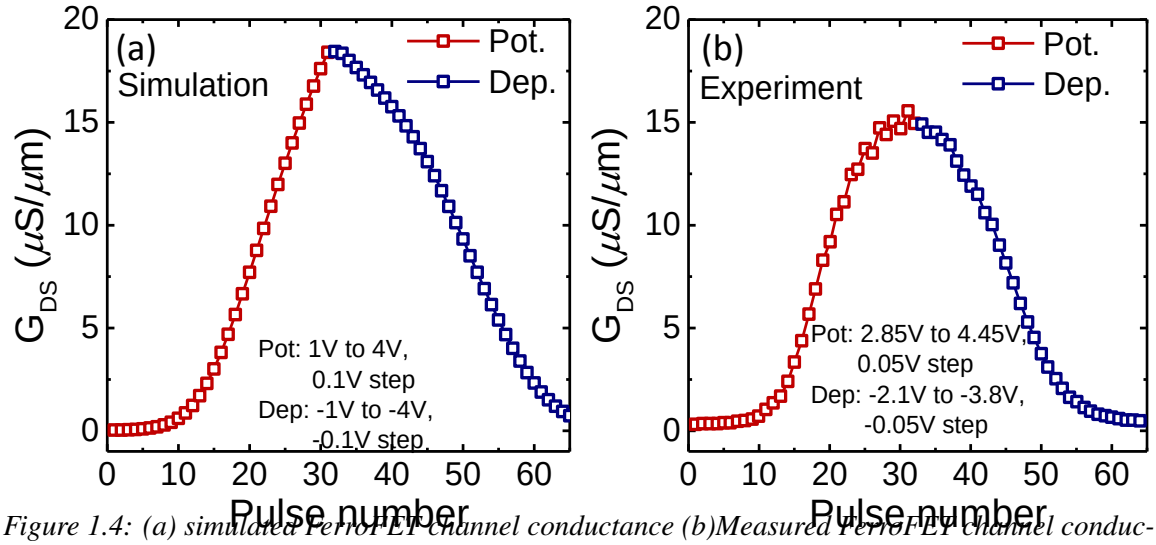


Figure 1.4: (a) simulated FerroFET channel conductance (b)Measured FerroFET channel conductance (G_{DS}) as a function of pulse number.

densely packed in an array, magnetic coupling of MTJs could become a significant problem since the distance of the ferromagnets, free and fixed layer of MTJs, reduces to cause even stronger magnetic coupling. Therefore, there is an urgent need to identify how magnetic coupling affects properties of STT-MRAM and analyze whether magnetic coupling will pose as a scaling challenge in further scaling of STT-MRAM dimensions.

There is limited prior work on the analysis of magnetic coupling on STT-MRAM arrays. Observation of H_{stray} in victim MTJ with four neighboring MTJs in technology scaling was presented by one of the authors in [26]. However, detailed models of magnetic coupling, the role of technology scaling on stray field and their effect on the electrical characteristics has not been discussed. On the other hand, there is ample research that analyzes how static and dynamic properties of MTJ are affected by technology scaling. [27][28] presents a scaling roadmap of MTJ that contains trends for thermal stability, switching current density (J_{c0}), critical switching current (I_c), Resistance-Area product (RA), etc. The effect of technology scaling on the dynamic properties of MTJs is also well explored; [29],[30] present how write current(I_c) and critical current density(J_{c0}) change with technology nodes. While scaling MTJ dimension, the authors calibrate H_k to maintain a target thermal stability of the MTJ. [31] presents changes in write current density (J_{c0}) across different MTJ types.

Further, [32][33] proposes a scaling trend of anisotropy energy ($K_u t$) for single and dual interface MTJs. The authors also present models of thermal stability as a functions of MTJ dimensions. [34] examined how P_{fail} of a chip, which relates to thermal stability, changes with technology node. We also explore the case where the fixed layer is an anti-ferromagnet and the magnetic fields are closed. In this case, the free layer nanomagnets create the magnetic field which affects the performance and stability of the victim cell.

Retention test Challenges of STT-MRAM Arrays

STT-MRAM arrays are expected to suffer from read and write failures which are induced by electrical defects and process variations. The role of variations in read and write have been extensively studied, including prior work by the authors[35]. However, the role of resistive and capacitive defects and coupling faults is relatively unexplored (except for preliminary work in [36]). Apart from read and write faults, STT-MRAMs can also suffer from retention failures. The non-volatility (or retention characteristics) of the bit can be measured by the thermal stability factor. [37][38] describe retention failure as a bit-flip in a cell caused by thermal noise. The thermal activation model of STTMRAM in [37] suggests that a bit flip has a poisson distribution with time constant of $\tau.e^{\Delta}$ where $\tau \approx 1ns$. Due to this fact, conventional test methods for retention have very large number of test times and there is a strong need for implementing a retention test scheme of STT-MRAM arrays that has low testing latency.

Challenges in FerroFET based In-memory processing architecture

Modern computing systems based on the Von-Neumann architecture rely on a clear distinction between logic and memory, and processes information by executing a sequence of precise atomic instructions with periodic uploads to the memory. Such systems are the foundation of the digital revolution which began with the demonstration of the self-aligned planar-gate silicon MOSFET in the sixties and was accelerated by rapid advances in transis-

tor technology. However, in the last one decade, the volume of data collected by distributed sensors and networks has grown exponentially. Ingesting, processing and extracting actionable intelligence out of this abundant data requires large amount of data traffic between logic and memory blocks leading to the problem of memory bottleneck. This requires novel ways of architecting the compute platform. For example, by embedding processing elements in the memory sub-array itself in so called Processing-In-Memory (PIM) architectures [39, 40, 41, 42, 43], the traditional Von-Neumann bottleneck can be addressed and significant acceleration and improved power-efficiency can be achieved.

HfO_2 based Ferroelectric FETs (FerroFETs) have recently received great interest for its non von-neumann application in nonvolatile memory (NVM) [19]. Among all of post CMOS memories, the developments in FerroFET technology is a rather recent occurrence; thanks to the breakthrough discovery of the underlying physical phenomenon: ferroelectricity in CMOS compatible Hf based binary oxides in 2011 [44][45] [46] a flurry of research activities on FerroFETs has ensued worldwide [47] [48][49]. FerroFET are also the most energy efficient among all eNVM technologies. This is due to the fact that, in contrast to the other non-volatile memories which are all current driven, the FerroFETs relies on electric field-effect for memory state switching. While non-Von-Neumann architectures based on other emerging eNVM technologies are being explored in depth [50] [51][52][53][54], the FerroFET technology provides unique features for adoption in such emerging architectures and applications.

However, Using FerroFET as a computation device in subarray for in-memory computing architecture has major challenge. Since the increase in conductance level with respect to the amplitude of write pulse to the gate of FerroFET is non-linear, the output of analog multiplication(current across FerroFET, the product of conductance and voltage across FerroFET) contains error. Also, when analog multiplication output is transferred to digital domain by using Analog to Digital Converter(ADC), quantization errors from ADC will exacerbate the error. Therefore, in order to implement FerroFET based in-memory com-

puting architecture, we must quantify the effect of these limitations of FerroFET on system performance and present the device requirements that enables in-memory computing architecture.

Challenges of STT-MRAM based system for reinforcement learning on a drone

Over the past decade, there has been considerable success in using Unmanned Aerial Vehicles (UAVs) or drones in varied applications such as reconnaissance, surveying, rescuing and mapping. Irrespective of the application, navigating autonomously, particularly with camera based inputs, is one of the key desirable features for small drones, both indoors and outdoors. In recent years, reinforcement learning (RL) has been extensively explored for different type of robotic tasks, including drone navigation and collision avoidance. RL, in spite of its biomimetic approach, is computationally challenging [55][56]. The agent (drone) needs to collect visual data and train a neural network based model in real-time [56][57]. For a given velocity of the drone, the corresponding distance traveled between two frames (d_{frame}), and the minimum distance between obstacles (a measure of clutter in the environment), we can calculate the minimum number of frames/second (fps) required for collision avoidance. Since the drone needs to train on acquired data at least at the same rate as the fps, the amount of computation that needs to be performed is prohibitively large for embedded systems that can be mounted on small drones. Further, the emergence of STT-MRAM [58][59][60] technologies that exhibit high-density and low-standby-power aims to disrupt the design of embedded systems. In spite of their advantages, STT-MRAM technologies shows high write latency and energy. This makes them unsuitable for storing model weights in real-time RL systems such as drones, both in terms of meeting an fps (or, velocity) requirement and energy target.

1.3 Key contributions of the research

In the research, we classify the challenges listed above into device, circuit, memory array and system level.

At the device level, we analyze the magnetic coupling across bit-cells of STT-MRAM and show whether this challenge prohibits the STT-MRAM memory scaling. First, we present a model of magnetic field induced coupling between adjacent bits in an STT-MRAM array. A comprehensive analysis, across four technology nodes and different MTJ technologies, has been presented and we have analyzed the role of the magnetic coupling on electrical performance, both static and dynamic. We conclude that for MTJ technologies with dense memory bits and lower stored energy, the coupling field can cause significant change in the average retention time. Data patterns that activate the worst and best case scenarios have also been explored. Dynamic analysis reveals that critical current densities are weakly disturbed by the coupling field. It should be noted that the research explores ultra-dense memory bit cells with cell sizes which are $15F^2$ and $6F^2$. The state-of-the art bit-cells are significantly larger (3X larger) and effects such as magnetic coupling will be reduced. However, key observations such as the data pattern dependence of retention, will remain unchanged and as the technology matures and denser bit-cells are enabled, magnetic field induced coupling will play a key role in both design and test.

At the circuit level, we present a comprehensive test methodology that solves the retention test challenges of STT-MRAM arrays. We identify electrical defects and magnetic coupling induced data pattern dependence on tests for read, write and retention and propose an MBIST architecture (EMACS) capable of collecting statistical data in an STT-MRAM subarray to estimate the thermal stability and retention. The proposed MBIST shows 93.75% improvement in test-time compared to a brute-force approach [37] with less that 5% estimation error.

At the memory array level, we analyze the challenges of FerroFET based in-memory

computing architecture and present a systolic processing-in-memory(PIM) architecture based on analog FerroFet pseudo-crosspoint arrays with in-situ computation to enable distributed convex optimization(non-uniform sampling) via least square minimization. The system demonstrated $21\times$, $3\times$ improvement in energy efficiency and compute time compared to an SRAM based Processing- In-Memory (PIM) architecture.

At the system level, we present a hardware-algorithm frame-work for STT-MRAM based embedded systems for application to small drones. we present a hardware-algorithm frame-work for STT-MRAM based embedded systems for application to small drones. We show that TL followed by RL on the last few layers of a deep CNN provides comparable performance compared to an E2E RL system, while reducing latency and energy by 79.4% and 83.45% respectively.

1.4 Dissertation overview

In the next chapters, we first address the current issues of Spin-Transfer Torque Magnetic Random Access Memory(STT-MRAM) and multi-bit ferroelectric FET in the device level. At the circuit level, the research shows how these issues shape the peripheral circuit of STT-MRAM and ferroelectric FET memory arrays. Lastly, at the system level, the research leads to the efficient memory architecture and system design that maximizes the benefits of STT-MRAM and ferroelectric FET while mitigating the current limitations of these devices. Lastly, we applies the in/near memory processing system design with STT-MRAM and ferroelectric FETs to various applications such as reinforcement learning with a drone, image classification with Deep Neural Network and least square minimization for image reconstruction. we focus on near-memory processing system with STT-MRAM for reinforcement learning of a drone and evaluate the system to quantify how much benefits are expected in terms of latency, power and energy. we would like to show that near-memory processing system with non-volatile devices is a key enabler for a real-time learning systems with stringent power and energy constraints.

CHAPTER 2

MAGNETIC COUPLING ACROSS BIT-CELLS OF STT-MRAM

In this chapter, we analyze how magnetic coupling affects both static and dynamic properties of MTJs with in-plane anisotropy,[61] Bulk perpendicular anisotropy[62] and interface induced perpendicular anisotropy [63] across different technology nodes. In modeling section, we present a compact model of MTJs and show the effect of magnetic field coupling as a function of MTJ dimensions and spacings. Then the data pattern dependence of magnetic coupling is analyzed in a 3×3 array and the worst case data pattern for each of the MTJ stacks is discussed. In the analysis section, we present how static properties (Δ, τ) are affected by different scenarios of magnetic field induced coupling.

2.1 Modeling of STT-MRAM and external magnetic field

2.1.1 MTJ physical dimension modeling

From [62][64], dimensions of in-plane, bulk and interface-induced perpendicular MTJ are retrieved. For more details on the three types on MTJs and their relative merits/demerits and role in the technology development, interested readers are pointed to [65][26][66][67]. in-plane MTJ (IMTJ) is modeled as an elliptical pillar and perpendicular MTJs (PMTJ) are modeled as cylinders.

Fig. 2.1 illustrates the physical dimensions of in-plane and perpendicular MTJ cells. In Fig.2.1 t_f, t_{sp} and t_{fix} represent thickness of free layer, insulating layer and fixed layer respectively. Length of in-plane MTJ is determined by the product of aspect ratio (AR) and the width of the in-plane MTJ. Since aspect ratio is one of the factors that determines H_k and thermal stability, its value changes with target thermal stability. In order to observe how magnetic coupling of MTJ cell array change with respect to technology node, we scale

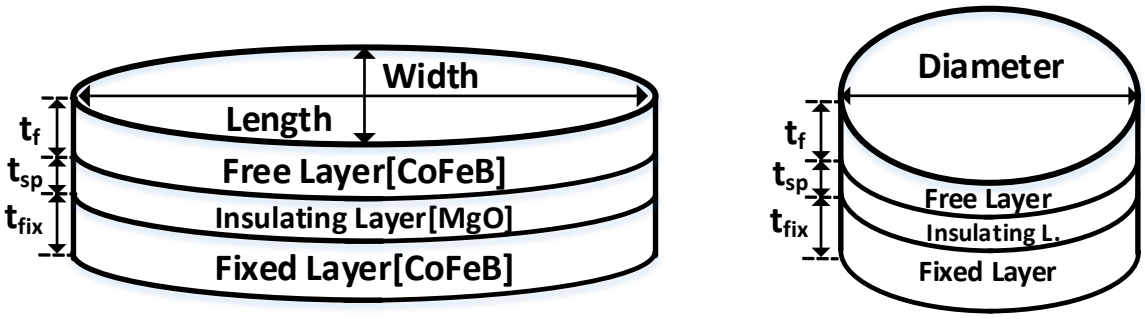


Figure 2.1: (a) In-plane MTJ (b) Perpendicular MTJ
Physical dimensions of MTJ cell types. The perpendicular MTJ can be Bulk or Interface perpendicular MTJ

physical dimensions of MTJs. Table2.1 shows physical dimensions at different technology node. Saturation magnetization remains constant in all the technology nodes. Similar to aspect ratio in IMTJ, free layer thickness (t_f) of Interface PMTJ is also an important design variable that determines thermal stability. Therefore, AR and t_f are scaled appropriately to maintain a constant thermal stability in all the technology nodes. From [62][29], device parameters of Table 2.1 were chosen.

Table 2.1: Physical dimensions of MTJ in STT-MRAM bit-cells across technology generations

Cell type	Dimension Parameter(nm)	Technology node(nm)			
		22nm	16nm	10nm	7nm
IMTJ	width	50	35	24.5	17.2
	length	AR*width			
	t_f	3			
	t_{fix}	5			
	t_{sp}	1.2			
Bulk PMTJ	diameter	40	28	19.6	13.7
	t_f	3			
	t_{fix}	5			
	t_{sp}	1.2			
Interface PMTJ	diameter	40	28	19.6	13.7
	t_f	variable dependent on delta			
	t_{fix}	3			
	t_{sp}	0.9			

2.1.2 Modeling the H field

The magnetic field of a single MTJ is first modeled to observe the net magnetic field coupling between adjacent cells. In an STT-MRAM array we consider a cell in the center of a 3×3 lattice as the victim cell and the eight neighbors as aggressors. Under the assumption of uniform magnetization of the the MTJ material, the magnetic dipoles inside MTJs cancel out and finally the magnetic dipoles on the edges of the MTJ are unpaired.

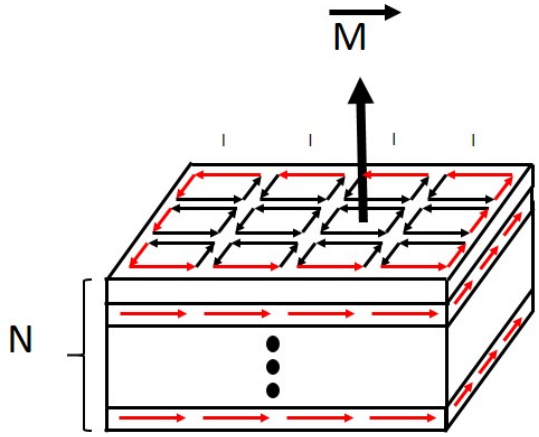
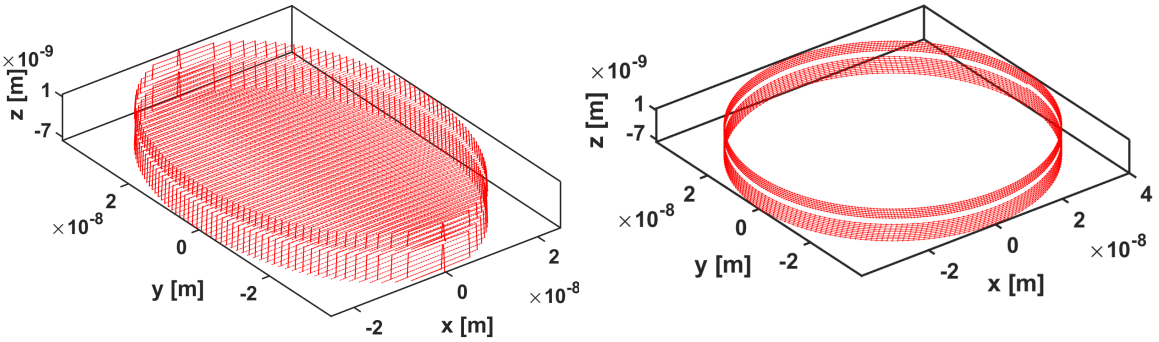


Figure 2.2: Schematic representation of current loops in the nanomagnet

Magnetic dipoles can be in turn modeled as current loops following[68]. Fig. 2.2shows how magnetic dipoles inside an MTJ cancels each other's internal current loops [68] . Hence we model an MTJ as a solenoid which has bound current paths wrapped around itself to produce the saturation magnetization (M_s) of an MTJ as described in [62].

Since magnetic moment is derived from the volume and M_s of an MTJ ($M_s = \frac{Magnetic\,moment}{Volume\,of\,MTJ}$) and it is the product of the bound current, the cross sectional area of the MTJ and the number of coils, the amount of current needed to produce the magnetic field can be calculated. The current is expressed as $\frac{M_s t}{no.of coils}$, t is the thickness of an MTJ layer.

Fig. 2.3 shows the IMTJ and PMTJ with the corresponding solenoid model for evaluating the resultant magnetic field. The current loop around an MTJ is wrapped around in a direction that generates the net M_s. Finally, we can calculate the magnetic field at any specific point in space by applying the Biot-Savart law [68], as:



(a) Solenoid modeling of free and fixed layer of IMTJ

(b) Solenoid modeling of free and fixed layer of PMTJ

Figure 2.3: Solenoid representation of current loops in IMTJ and PMTJ to model magnetic field around MTJs

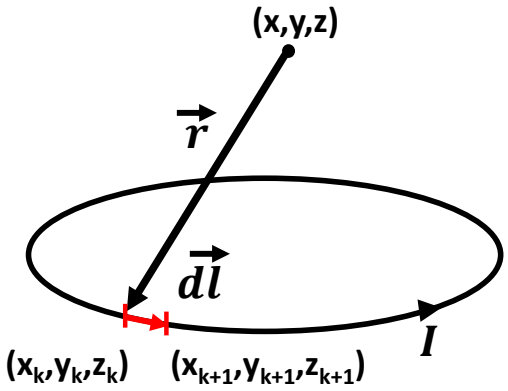


Figure 2.4: Finite element representation of biot savart law

$$\vec{H}(x,y,z) = \frac{I}{4\pi}\int_C \frac{d\vec{l}\times\vec{r}}{|\vec{r}|^3} \tag{2.1}$$

where, d$\vec{l}$ is defined by $\{d\vec{x}, d\vec{y}, d\vec{z}\}$, which is equal to $\{\vec{x}_{\mathrm{k+1}} - \vec{x}_{\mathrm{k}}, \vec{y}_{\mathrm{k+1}} - \vec{y}_{\mathrm{k}}, \vec{z}_{\mathrm{k+1}} - \vec{z}_{\mathrm{k}}\}$ from Fig.2.4.

Algorithm. 2 shows the pseudo-code for a discrete finite element representation of Biot-savart law, which is used to calculate magnetic field at coordinate (x,y,z). For each segment in the model, Algorithm. 2 computes dHx,dHy and dHz, x,y,z components of d$\vec{l}\times\vec{r}$, and stores it in an array. After computing d$\vec{l}\times\vec{r}$ for all segments, we can find the magnetic field in x,y,z direction at point (x,y,z) by summing up dHx, dHy and dHz and multiplying by the

coefficient $\frac{I}{4\pi}$

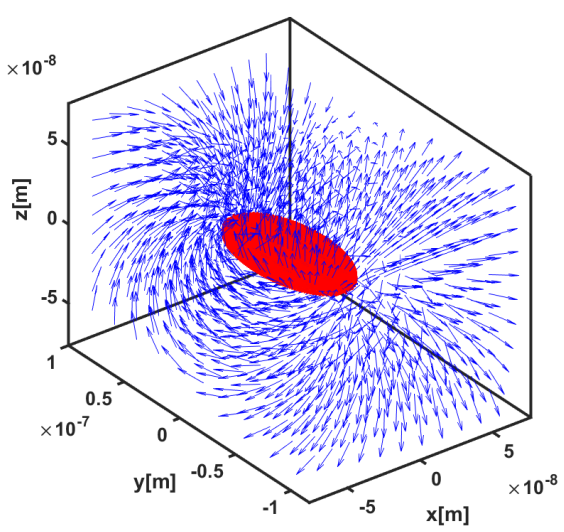
Result: Calculate H_{stray} from MTJs at coordinate (x,y,z)
N = number of points in MTJ model;
x_p[N] = array of x-coordinates of MTJ model;
y_p[N] = array of y-coordinates of MTJ model;
z_p[N] = array of z-coordinates of MTJ model;
for *k = 1; k <N-1; k++* **do**
    r_mag = sqrt((x-x_p[k])^2 + (y-y_p[k])^2+(z-z_p[k])^2);
    dx[k] = x_p[k+1]-x_p[k];
    dy[k] = y_p[k+1]-y_p[k];
    dz[k] = z_p[k+1]-z_p[k];
    dHx[k] = (dy[k]*(z-z_p[k])-dz[k]*(y-y_p[k]))/(r_mag)^3;
    dHy[k] = (dz[k]*(x-x_p[k])-dx[k]*(z-z_p[k]))/(r_mag)^3;
    dHz[k] = (dx[k]*(y-y_p[k])-dy[k]*(x-x_p[k]))/(r_mag)^3;
end
Hx = (I/(4*pi))*sum(dHx);
Hy = (I/(4*pi))*sum(dHy);
Hz = (I/(4*pi))*sum(dHz);

Algorithm 1: Biot Savart law for finding magnetic field H at (x,y,z) coordinate

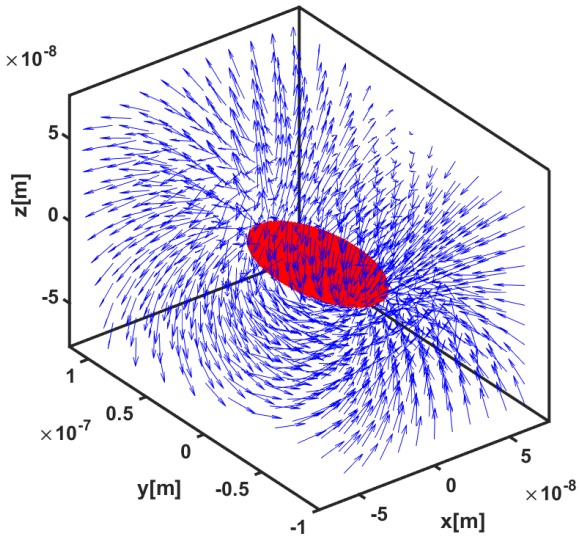
By using Biot-Savart law and finite element method as above, we find magnetic field at a set of coordinates in 3D space and Fig. 2.5 shows the complete magnetic field modeling for free layer of IMTJ and PMTJ in space. In Fig. 2.5, The magnetic field direction between Fig. 2.5(a) and Fig. 2.5(b), Fig. 2.5(c) and Fig. 2.5(d) are opposite to each other because direction of bound current into the coil is opposite.

We expect that the magnetic coupling between aggressor cells and a victim cell would be affected by the distance between the cells 2.1. In order to observe the difference in $\vec{H}$ with respect to distance between cells, we consider two types of MTJ cells: (1) a nominal cell of size $5F \times 3F$ and (2) a compact cell size as $3F \times 2F$. Here F is the half-pitch of the poly-silicon layer for a given technology node.

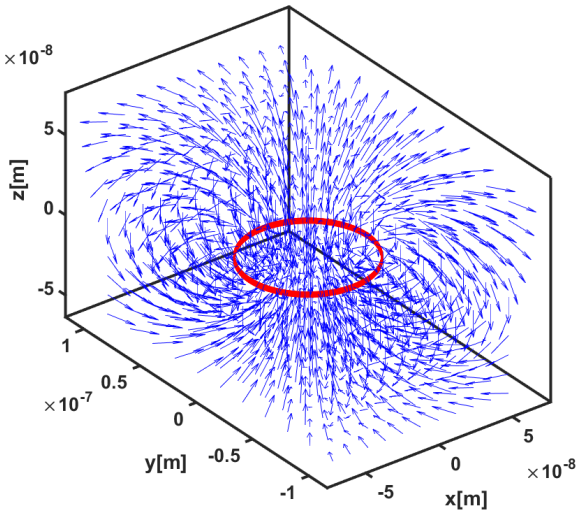
Fig. 2.6 shows the nominal and compact cells in array configurations. The MTJ at the center of an array in Fig. 2.6 is the victim MTJ and distance labeled in Fig. 2.6 is the center to center distance between the victim cell and its aggressor neighboring cells. For each cell, we model the net magnetic field generated by both the free layer and the fixed layers. Then



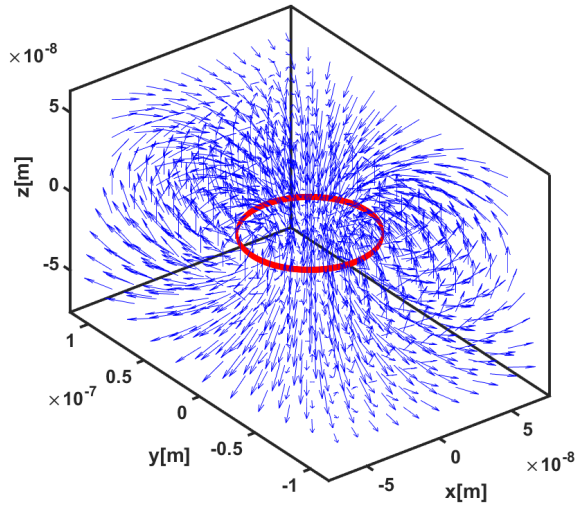
(a) Magnetic field around IMTJ diverging to +y direction



(b) Magnetic field around IMTJ diverging to -y direction



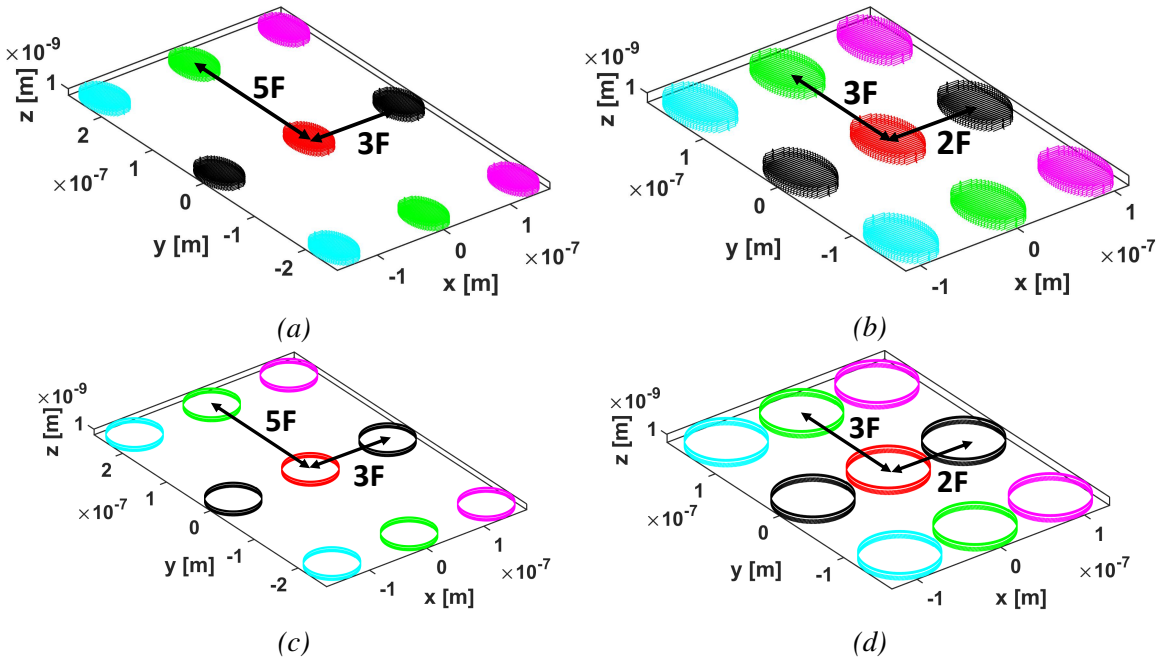
(c) Magnetic field around PMTJ diverging to +z direction



(d) Magnetic field around PMTJ diverging to -z direction

Figure 2.5: Magnetic field around IMTJ and PMTJ when current is applied to current loops.(a) current flowing from +y to -y direction (b) current flowing from -y to +y direction (c) current flowing from -z to +z direction (d) current flowing from +z to -z direction

we calculate the net magnetic field from each MTJ and compute the total magnetic field at the victim node. Although the fixed layer has its magnetic moment pointing in a specific direction, the direction of the magnetic moment in the free layer is data dependent. Hence, the net field generated by the neighboring cells on the victim, depends on the over-all data pattern of the 3×3 array. In the next section, we explore the effect of data pattern on the



spacing(nm)	cell size	Technology node(nm)			
		22nm	16nm	10nm	7nm
x-spacing	normal	135	96	60	42
	compact	90	64	40	28
y-spacing	normal	225	160	100	70
	compact	135	96	60	42

Figure 2.6: (a) IMTJ default cell array (b) IMTJ compact cell array (c) PMTJ normal default array (d) PMTJ compact cell array

coupling field on the victim node and determine the worst and best data patterns that can reduce magnetic coupling. It should be noted that our discussion in this paper is limited to the nanomagnet. The access transistor in the bit-cell plays an important role in the cell dynamics[69][70], especially the write properties. The retention properties of the cell are not disturbed by the access transistor, at least to the first order. However, the aim of this paper is to explore the performance and retention behavior of bit-cells with and without magnetic coupling from the neighboring cells. Hence, we have not considered the role of the access transistor in our discussions.

2.2 Role of Magnetic coupling in dense arrays

.

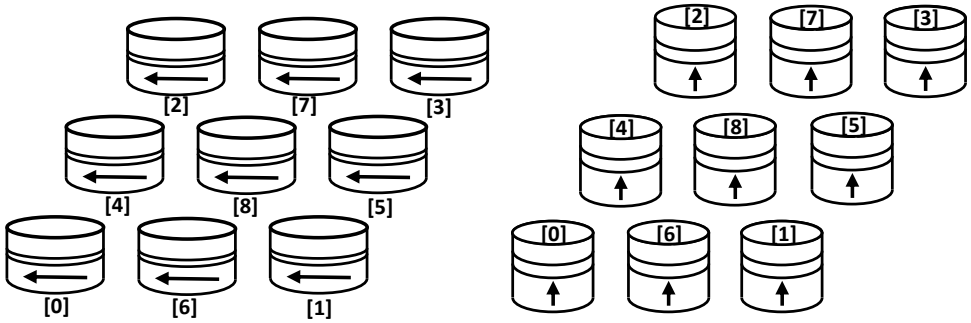
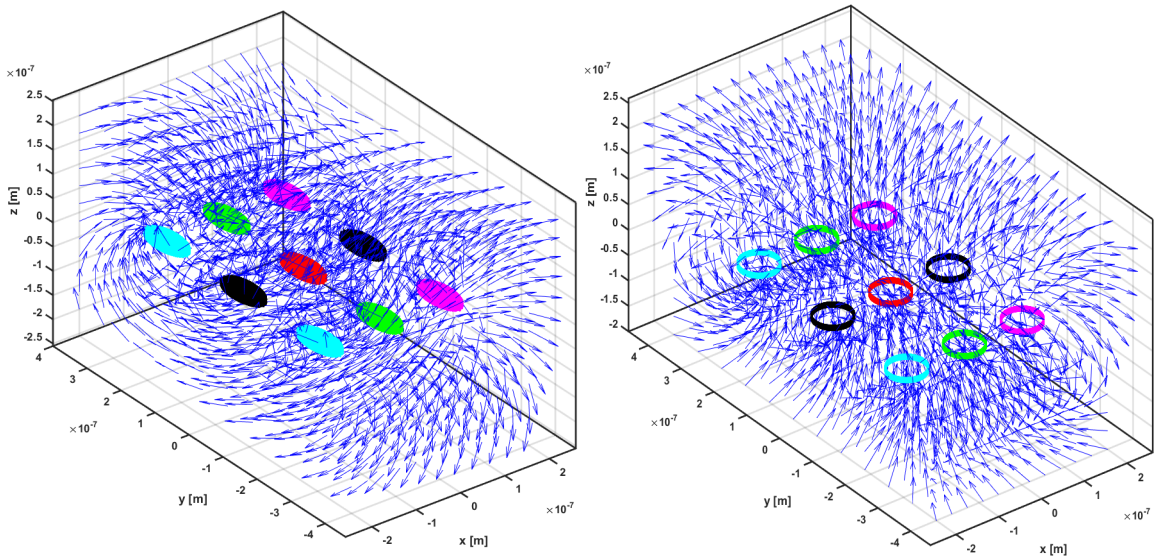


Figure 2.7: Arrangement of MTJs in a 3×3 *array[71]*



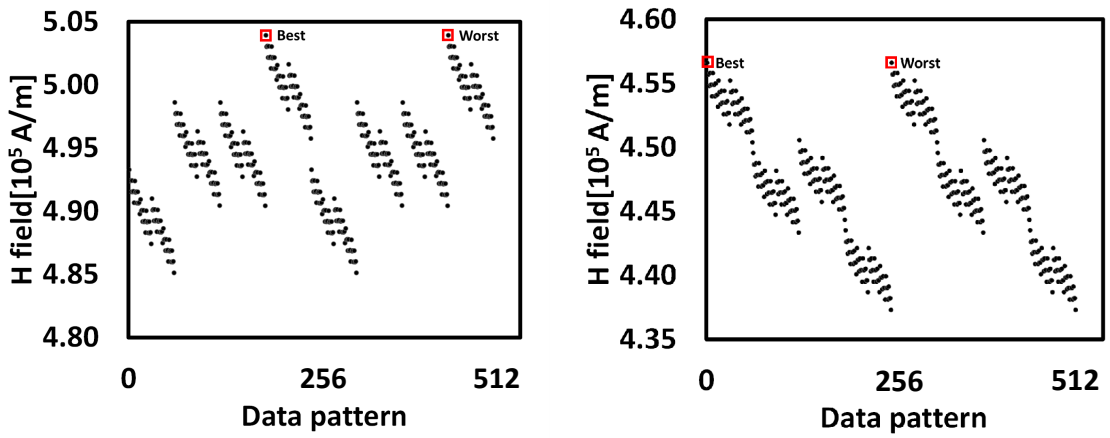
(a) IMTJ *(b) PMTJ*

Figure 2.8: Magnetic field visualization of IMTJ and PMTJ 3×3 *arrays for the worst data pattern[71]*

Similar to electrical coupling between DRAM cells[72], The data pattern on neighboring STT-MRAM cells can cause magnetic coupling with a cell since data inside a nanomagnet determines magnetic field direction. we need to analyze the magnetic field coupling and its magnitude in STT-MRAM bitcells. In this section we present modeling of magnetic coupling effects on a 3×3 array and analyze the best case and worst case data patterns which yields minimum and maximum magnetic coupling on a cell at the center of the array. In order to capture the complete magnetic coupling effect from adjacent cells, doing

analysis with more number of adjacent cells can increase the accuracy of the model. However, since magnetic field decreases quadratically with distance, the cells that are farther away would assert a very weak field on the victim. Hence, we invoke the near neighbor interactions only, which is staple in the modeling and simulation of most interacting magnetic structures. We increase the accuracy of our model by including the diagonal elements as opposed to only the four nearest neighbors. Therefore, we use a 9 cell lattice, to explore how magnetic coupling affects the victim cell's characteristics.

Initial results and observations on the data pattern dependence of magnetic coupling have been briefly discussed by the authors in [71] The magnetic coupling is measured by adding magnetic field vectors from neighboring nanomagnets on the victim cell. Fig. 2.7 shows the arrangements of the 3×3 array of magnets and the figure denotes that victim bit is located at position [8]. Fig. 2.8 shows the magnetic field from IMTJ and PMTJ arrays which saturation magnetization is set to 1.257e6 A/m.

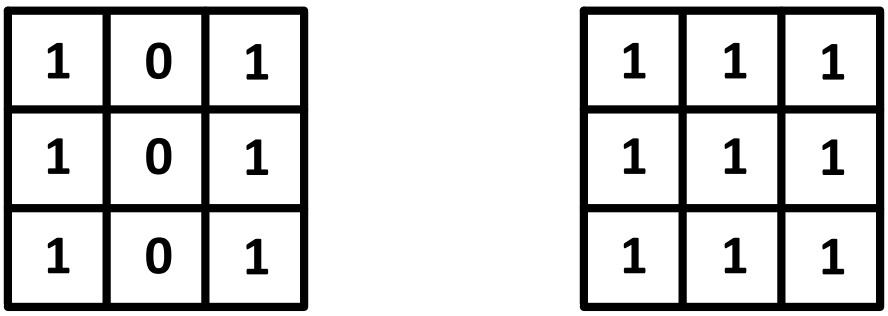


(a) IMTJ *(b) PMTJ*

Figure 2.9: Residual H field vs. data pattern in IMTJ and PMTJ[71]

2.2.1 Impact of Magnetic Coupling on Write and Retention

To visualize the best and worst case data patterns, we represent the information stored in the 3×3 array as a 9-bit number where each bit represents the data stored (0 for anti-parallel and 1 for parallel) in the i^{th} bit as shown in Fig. 2.7. Because of this encoding, data patterns



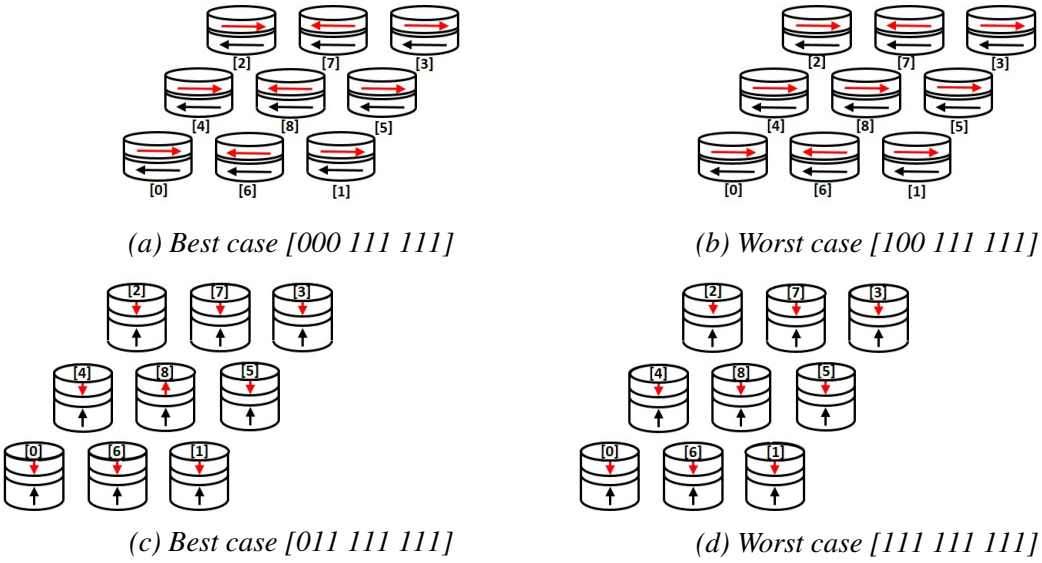
(a) IMTJ block data pattern *(b) PMTJ block data pattern*

Figure 2.10: Magnetic coupling induced worst-case data pattern for thermal stability[71]

0 to 255 represent the victim storing a 0 and 256 to 511 represents the victim storing a 1. Fig. 2.9 show residual magnetic field strength from all the aggressors for all possible data arrangements. Residual field in the direction of the free layer's magnetization enhances stability and improves retention (thereby degrading writability) while residual fields in the opposite direction would tend to destabilize the magnet. We note that data pattern [111 000 000] and [011 000 000] are the best and worst case data patterns for thermal stability (or retention) for IMTJ. For both varieties of PMTJ, best and worst data patterns are [100 000 000] and [000 000 000]. The residual field is taken at the center of free layer of a victim cell.

Due to the uni-axial anisotropy in two MTJ types, best and worst case data pattern are different between in-plane and perpendicular MTJs, i.e. due to their physical structure and anisotropy. While the magnetization of IMTJ is aligned to the y-axis and magnetization in PMTJs is aligned to the z-axis. Therefore, the best and worst case data pattern for IMTJ and PMTJ are different as the vector field on the victim magnet and its effect on the victim need to be evaluated.

The worst case patterns for the 3X3 block is shown in Fig. 2.10.



(a) Best case [000 111 111] *(b) Worst case [100 111 111]*

(c) Best case [011 111 111] *(d) Worst case [111 111 111]*

Figure 2.11: MTJ best/worst data pattern[71]

2.3 Effect of Magnetic Coupling on Static Characteristics of the Victim Cell

In static analysis, we analyze the effect of magnetic coupling on thermal stability and retention of a victim cell i.e., the cell at the center of the 3x3 STT-MRAM array. Analysis is conducted on in-plane (IMTJ), Bulk and interface-induced perpendicular MTJs (CPMTJ, IPMTJ). By varying the technology nodes (22/16/10/7nm), we observe how change in physical dimension of an MTJ and the distance between MTJs in 3×3 array impact the magnetic coupling and its effect on Δ and retention. Also, effect of cell size (nominalcompact) and bestworst data patterns on magnetic coupling in each MTJ types is studied. In order to gauge how magnetic coupling causes variation with respect to Δ, we set nominal Δ of MTJs to be 20,40 and 60. These three types of MTJs represent trade-offs between non-volatility and lower write power [73]. In short, we analyze;

1) Which type of MTJ is affected the most from magnetic coupling in terms of of thermal stability and retention.

2) The effect of magnetic couping on thermal stability and retention with respect to

changes in;

a) target thermal stability ($\Delta = 20, 40, 60$)

b) technology scaling (22 /16 /10 /7 nm)

c) nominal and compact cell sizes (15 F^2 vs. 6 F^2)

d) best/worst data pattern

Based on M_s (1.257e6 A/m) and physical dimension of MTJs discussed in Table. 2.1, we modify other parameters of MTJs to set nominal Δ of MTJ to be 20,40 and 60. Since Δ is defined as [74]

$$\Delta = \frac{K_u V}{k_B T} = \frac{H_k M_s V}{2 k_B T} \tag{2.2}$$

we vary H_k to achieve nominal Δ. However, since H_k is a property which is related to AR, K_u and t_f in IMTJ, CPTMJ and IPTMJ according to Eqn.s 2.3, 2.4 and 2.5

$$\text{IMTJ } H_k = 2(\frac{4\pi M_s t(AR-1)}{wAR}) \tag{2.3}$$

[64]

$$\text{Bulk PMTJ } H_k = \frac{2K_u}{M_s} - 4\pi M_s \tag{2.4}$$

[74]

$$\text{Interfacial PMTJ } H_k = \frac{4\pi {M_s}^2 t_c}{M_s t_f} - 4\pi N_{DZ} M_s \tag{2.5}$$

[29]

t_c from equation(5) is critical thickness of CoFeB layer. N_{DZ} is z-axis dependent demagnetizing factor. Since AR and t_f both affect H_k and volume of MTJ, they are determined through iterations between H_k and Δ Eqn.s 2.3, 2.5 and 2.2. AR, K_u and t_f parameters for different nominal Δ in technology nodes are defined in Table. 2.2

Table 2.2: Design parameters for maintaining a target Δ

Cell type	Variable properties	Δ @85C	Technology node(nm)			
			22	16	10	7
IMTJ	Aspect Ratio	20	1.147	1.21	1.3	1.425
		40	1.293	1.421	1.6	1.853
		60	1.44	1.628	1.895	2.28
Bulk PMTJ	K_u (10^6J/m^3)	20	0.909	0.936	0.992	1.106
		40	0.935	0.989	1.101	1.329
		60	0.961	1.043	1.21	1.553
Interface PMTJ	t_f (nm)	20	1.485	1.471	1.441	1.379
		40	1.471	1.442	1.382	1.258
		60	1.457	1.413	1.323	1.137

2.3.1 Effect of Magnetic Coupling on Thermal stability

The effect of magnetic field on the stored magnetic energy in an MTJ can be modeled as [64]

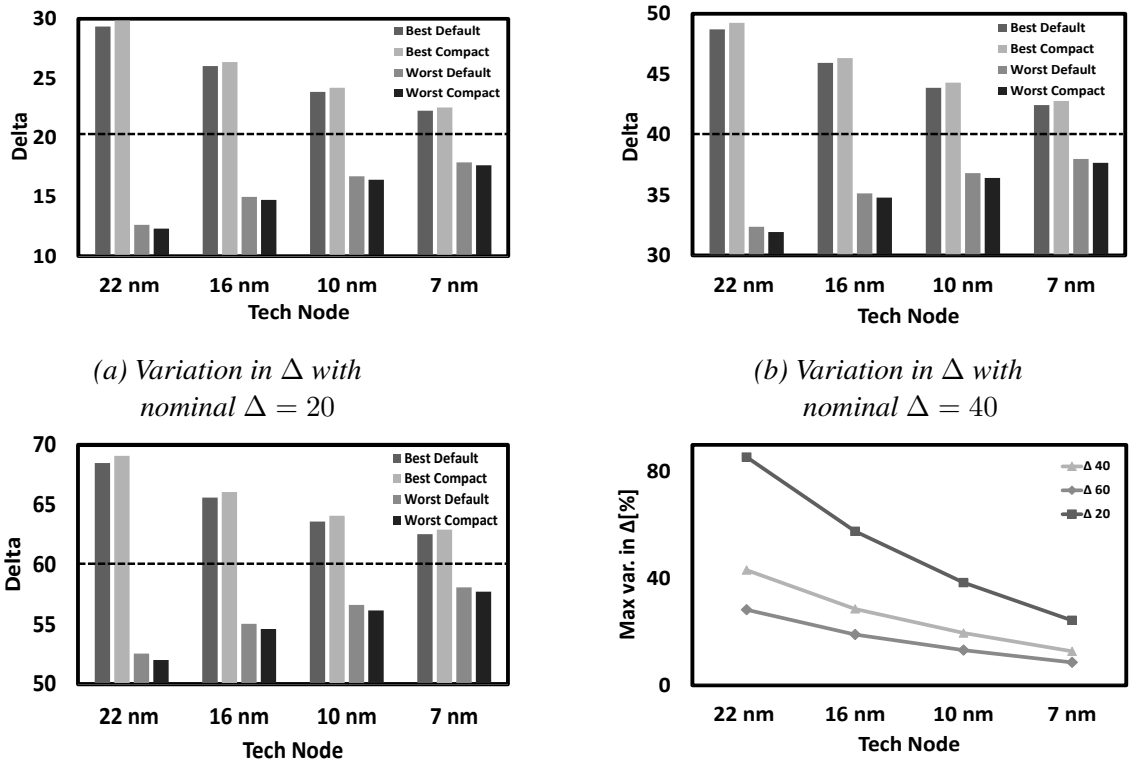
$$\Delta(H) = \Delta(H = 0)(1 \pm \frac{H_{\text{stray}}}{H_k})^2 \tag{2.6}$$

This shows that an external magnetic field (normalized by H_k) at the free layer of victim cell can cause variation in Δ of MTJ. We model the magnetic field from neighboring cells and the vector field, (H_{stray}) is calculated and it is applied to the victim cell as shown in Eqn. (2.6).

in-plane MTJ

Fig.2.12 represents the variation of Δ in an IMTJ due to H_{stray} with respect to technology node, cell size and data pattern for target Δ = 20/40/60. Dotted line in the figure represents target Δ. The data patterns in MTJ array that yield the H parallel and anti-parallel to magnetization of victim cell are labeled as best and worst in the figure. The labels nominal and compact indicate the nominal and compact cell sizes as defined in earlier sections.

The common trend in Δ variation in Fig. 2.12 is that the variation is decreasing as we decrease technology node. This phenomenon is expected because as we decrease technol-



(a) Variation in Δ with nominal $\Delta = 20$

(b) Variation in Δ with nominal $\Delta = 40$

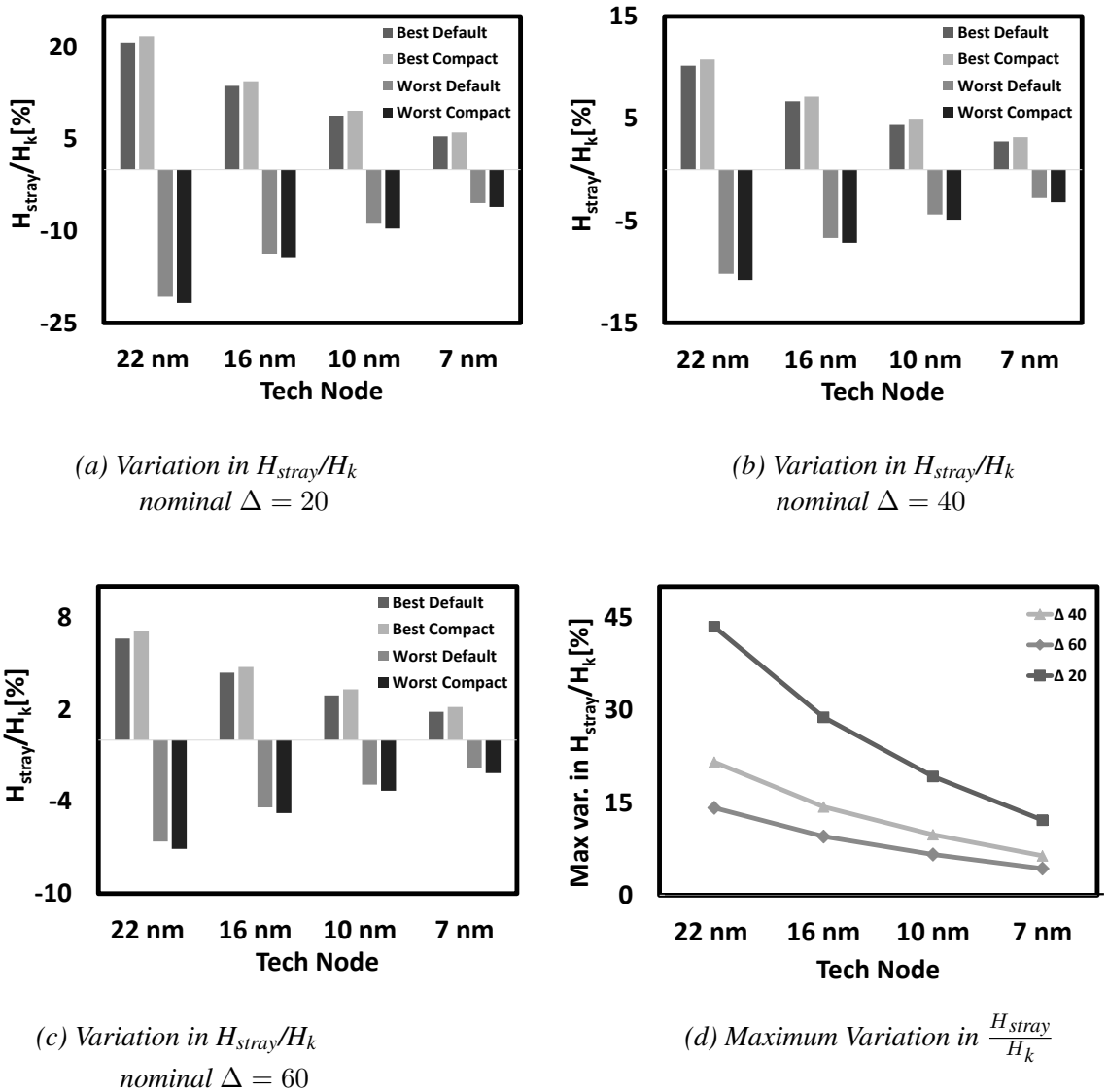
(c) Variation in Δ with nominal $\Delta = 60$

(d) Maximum Variation in Δ

Figure 2.12: Variation of Δ in IMTJ with respect to technology nodes, data patterns and cell array configuration (a) Variation with nominal Δ = 20 (b) Variation with nominal Δ = 40 (c) Variation with nominal Δ = 60 (d) Maximum variation of Δ across combinations of data pattern and cell array configuration in technology node.

ogy node, the volume of MTJ is decreasing and it causes Δ to decrease (2.2). In order to maintain target Δ across all technology nodes, we can either adjust M_s or H_k. In this analysis, we fixed M_s to be constant for all technology nodes, as it is a material property. We tune geometric parameters of the bit cell to achieve a target H_k. Hence, with technology scaling, the magnetic coupling does increase, but surprisingly we note that a stronger cell anisotropy (owing to increased H_k), results in an effective decrease of H_{stray}/H_k. Fig. 2.13 shows the variation of $\frac{H_{\text{stray}}}{H_k}$ across technology nodes. $\frac{H_{\text{stray}}}{H_k}$ is positive when H_{stray} is aligned with M_s and negative when it is anti-parallel to M_s. As we can see, $\frac{H_{\text{stray}}}{H_k}$ is decreasing as technology scales. The second trend that we observe from Fig. 2.12(d) is that the maximum variation(%) across data pattern and cell size decreases as target Δ changes from 20 to 60.

This is due to increasing in H_k as target Δ increases. With same M_s, the only variable to tune Δ of a MTJ to target Δ is H_k. Therefore, H_k increases as target Δ increases. As we



(a) Variation in H_{stray}/H_k nominal $\Delta = 20$

(b) Variation in H_{stray}/H_k nominal $\Delta = 40$

(c) Variation in H_{stray}/H_k nominal $\Delta = 60$

(d) Maximum Variation in $\frac{H_{stray}}{H_k}$

Figure 2.13: Variation of H_{stray}/H_k in IMTJ with respect to technology nodes, data patterns and cell array configuration (a) Variation with nominal Δ = 20 (b) Variation with nominal Δ = 40 (c) Variation with nominal Δ = 60 (d) Maximum variation of H_{stray}/H_k across combinations of data pattern and cell array configuration in technology node.

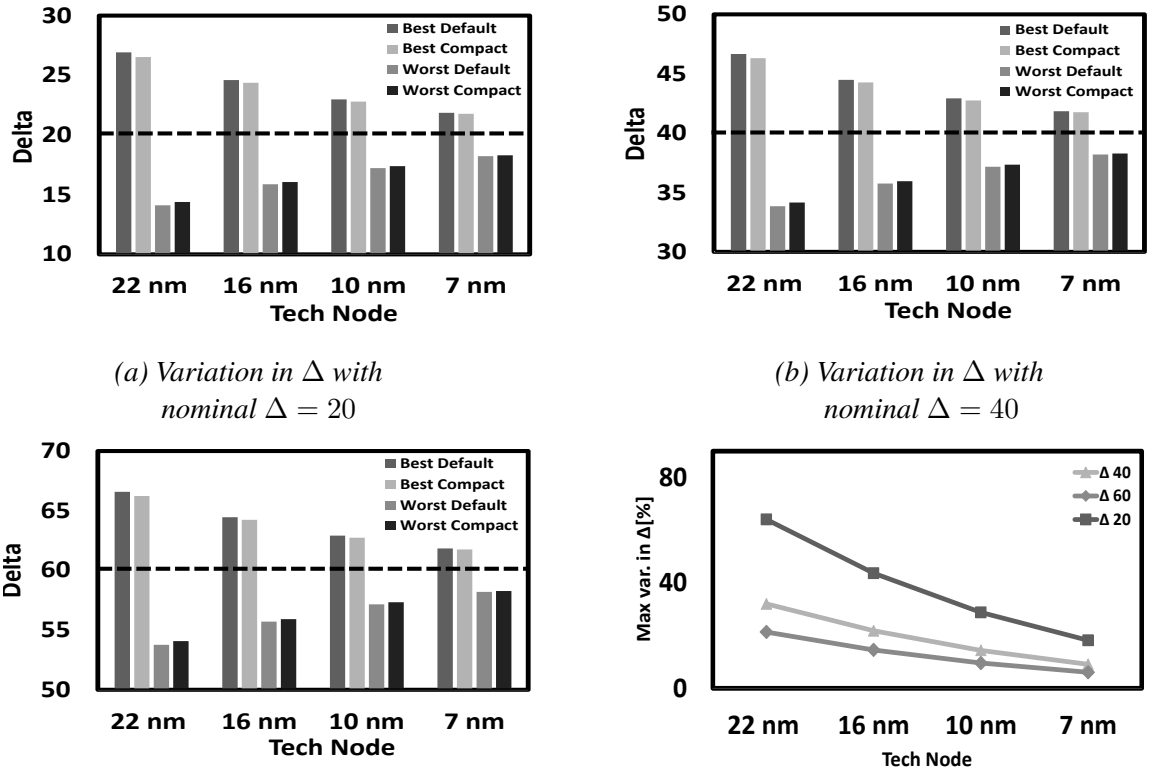
discussed in section III, best data pattern, which causes maximum H_{stray} in direction of M_s, boosts Δ and the worst data pattern that causes maximum H_{stray} in the opposite direction of M_s degrades Δ. When it couples with cell size, it yields maximum variation of 85%

between a compact cell with best data pattern and a compact cell with worst data pattern when target $\Delta = 20$ in 22nm based on 2.12(d). On the other hand, at target $\Delta = 60$ in 7nm, the maximum variation between compact cell with best and worst data pattern is 8.6%. By comparing results from nominal and compact cell sizes with same data pattern, we observe a 3% variation at the 22nm node.

Bulk perpendicular MTJ

Fig.2.14 exhibits Δ variation in CPMTJ. From the figure, CPMTJ also presents a decreasing trend of Δ variation as technology scales down and target Δ increases. However, the maximum variation in CPMTJ is less than maximum variation of IMTJ. From Fig. 2.9, we observe that magnitude of H_{stray} from perpendicular MTJ is less than in-plane MTJ due to the geometry of MTJ types and the direction of M_s for the same magnitude of M_s. This result explains why $\frac{H_{\text{stray}}}{H_{\text{k}}}$ across all technology node in PMTJ is less than that that of IMTJ as shown in Fig. 2.15 . As a result, we conclude that the Δ variation in PMTJ is less than that of IMTJ.

However, for CPMTJ the Δ variation in nominal and compact cell sizes is different from the Δ variation in IMTJ for different cell sizes. Between different cell sizes, maximum Δ variation is 2% in 22nm at target $\Delta = 20$. From Biot-Savart law (Eqn. 2.4), magnetic field at a point is stronger when distance between a point and the current loop is closer. Therefore, in compact cells, each MTJs exerts more H_{stray} on the victim cell. Since M_s direction in IMTJ is in y-direction in 3 by 3 array, the sum of H_{stray} from neighboring cell at victim cell is larger when MTJs are compact. In the case of PMTJ, the sum of H_{stray} from neighboring MTJs on the victim cell decreases because the direction of M_s of MTJs is in the z-direction. When distance between neighboring and victim MTJs is too close, the direction of H_{stray} from neighboring cell deviate significantly, which results in less H_{stray} in the direction of M_s on victim cell.



(a) Variation in Δ with nominal $\Delta = 20$

(b) Variation in Δ with nominal $\Delta = 40$

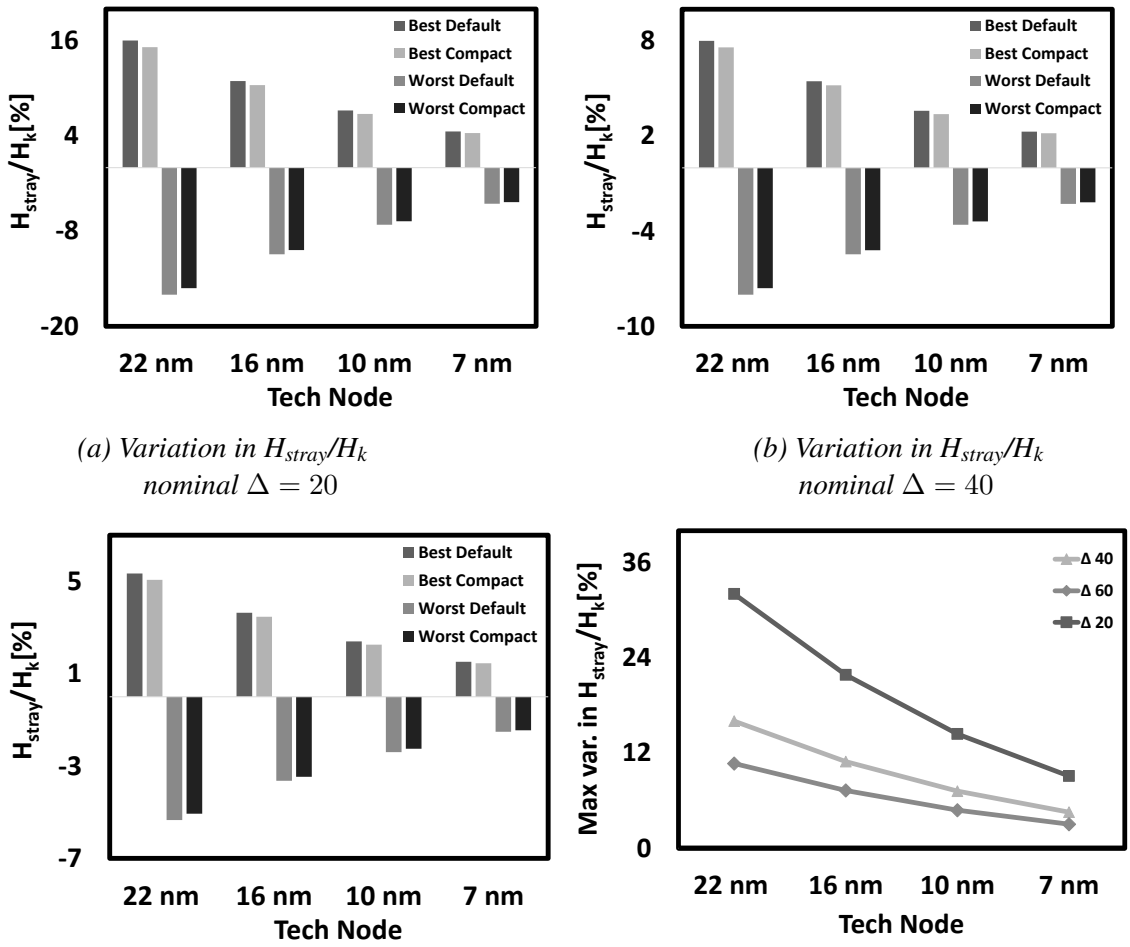
(c) Variation in Δ with nominal $\Delta = 60$

(d) Maximum Variation in Δ

Figure 2.14: Variation of Δ in CPMTJ with respect to technology nodes, data patterns and cell array configuration (a) Variation with nominal Δ = 20 (b) Variation with nominal Δ = 40 (c) Variation with nominal Δ = 60 (d) Maximum variation of Δ across combinations of data pattern and cell array configuration in technology node.

Interface-induced perpendicular MTJ

Fig.2.17 shows how $\frac{H_{\text{stray}}}{H_{\text{k}}}$ changes across technology nodes, target Δ in different cell sizes and data patterns. Fig. 2.16 shows how much variation it caused in Δ. The effect of magnetic coupling on the Δ of IPMTJ is similar to that of CPMTJ. The only difference between IPMTJ and CPMTJ in terms of Δ variation is the magnitude of variation. The reason for this difference lies in the relationship between Eqn.(2.2) and (2.5). For IMTJ and CPMTJ, as technology node scales, H_k is increased to compensate the loss in Δ caused by decreasing volume of MTJ. In IPMTJ, decreasing volume automatically increases H_k because decreasing t_f increases H_k. Therefore, H_k in IPMTJ is smaller than H_k in IMTJ and



(a) Variation in H_{stray}/H_k nominal $\Delta = 20$

(b) Variation in H_{stray}/H_k nominal $\Delta = 40$

(c) Variation in H_{stray}/H_k nominal $\Delta = 60$

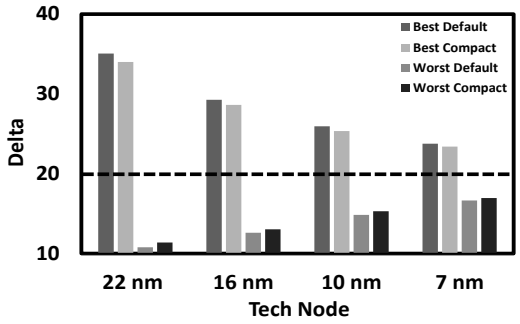
(d) Maximum Variation in $\frac{H_{stray}}{H_k}$

Figure 2.15: Variation of H_{stray}/H_k in CPMTJ with respect to technology nodes, data patterns and cell array configuration (a) Variation with nominal Δ = 20 (b) Variation with nominal Δ = 40 (c) Variation with nominal Δ = 60 (d) Maximum variation of H_{stray}/H_k across combinations of data pattern and cell array configuration in technology node.

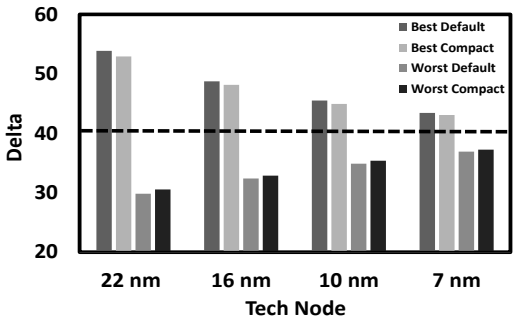
CPMTJ. It results in large variation in $\frac{H_{\text{stray}}}{H_{\text{k}}}$ and Δ.

Comparison of the Effect of Magnetic Coupling on Δ across MTJ types

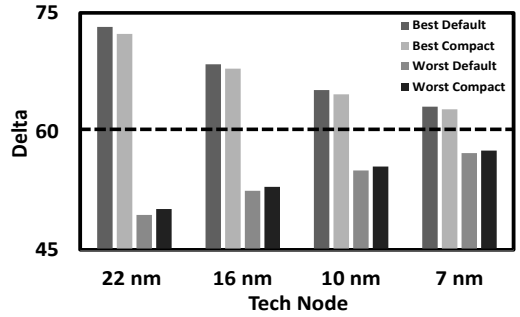
Fig. 2.18 summarizes the maximum Δ variation for IMTJ, CPMTJ and IPMTJ across target Δ and technology nodes. As we discussed above, Δ variation due to magnetic coupling is in the order: IPMTJ, IMTJ and CPMTJ. The conclusion from thermal stability analysis is that the Δ variation will not become a big problem as technology node decreases, which is



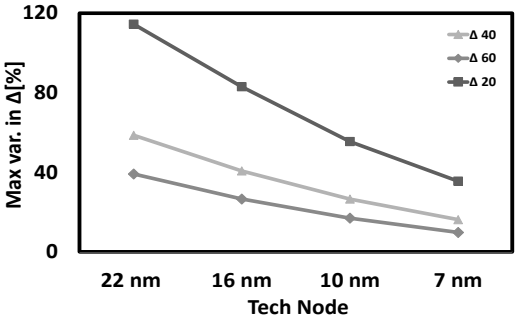
(a) Variation in Δ with nominal $\Delta = 20$



(b) Variation in Δ with nominal $\Delta = 40$



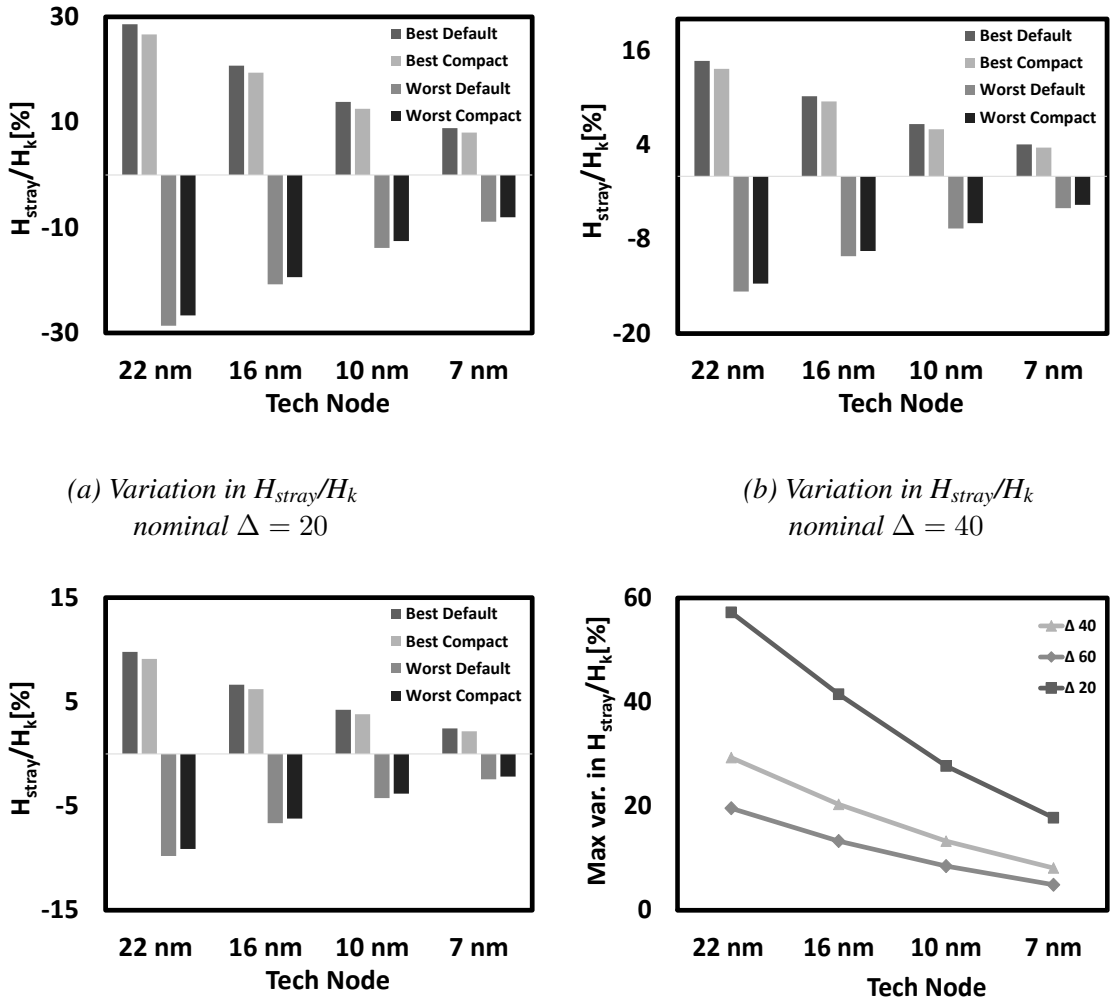
(c) Variation in Δ with nominal $\Delta = 60$



(d) Maximum Variation in Δ

Figure 2.16: Variation of Δ in IPMTJ with respect to technology nodes, data patterns and cell array configuration (a) Variation with nominal Δ = 20 (b) Variation with nominal Δ = 40 (c) Variation with nominal Δ = 60 (d) Maximum variation of Δ across combinations of data pattern and cell array configuration in technology node.

counter intuitive. From intuition, we expect that magnetic coupling will become a severe problem as technology node scales because STT-MRAM array will become denser. However, if we allow for scaling laws and adjust H_k as technology node decreases, the effect of magnetic coupling on thermal stability diminishes, since the the stray field is is normalized by H_k. For the same reason, magnetic coupling has minimal effect when $\Delta = 60$ since H_k is higher than the H_k of MTJs with lower target Δ.



(a) Variation in H_{stray}/H_k nominal $\Delta = 20$

(b) Variation in H_{stray}/H_k nominal $\Delta = 40$

(c) Variation in H_{stray}/H_k nominal $\Delta = 60$

(d) Maximum Variation in $\frac{H_{stray}}{H_k}$

Figure 2.17: Variation of H_{stray}/H_k in CPMTJ with respect to technology nodes, data patterns and cell array configuration (a) Variation with nominal Δ = 20 (b) Variation with nominal Δ = 40 (c) Variation with nominal Δ = 60 (d) Maximum variation of H_{stray}/H_k across combinations of data pattern and cell array configuration in technology node.

2.3.2 Effect of Magnetic Coupling on Retention Time

The average retention time (τ) of MTJ is exponentially dependent on the Δ [64]

$$\tau = \tau_0 exp(\frac{K_u V}{k_B T}) = \tau_0 exp(\Delta) \tag{2.7}$$

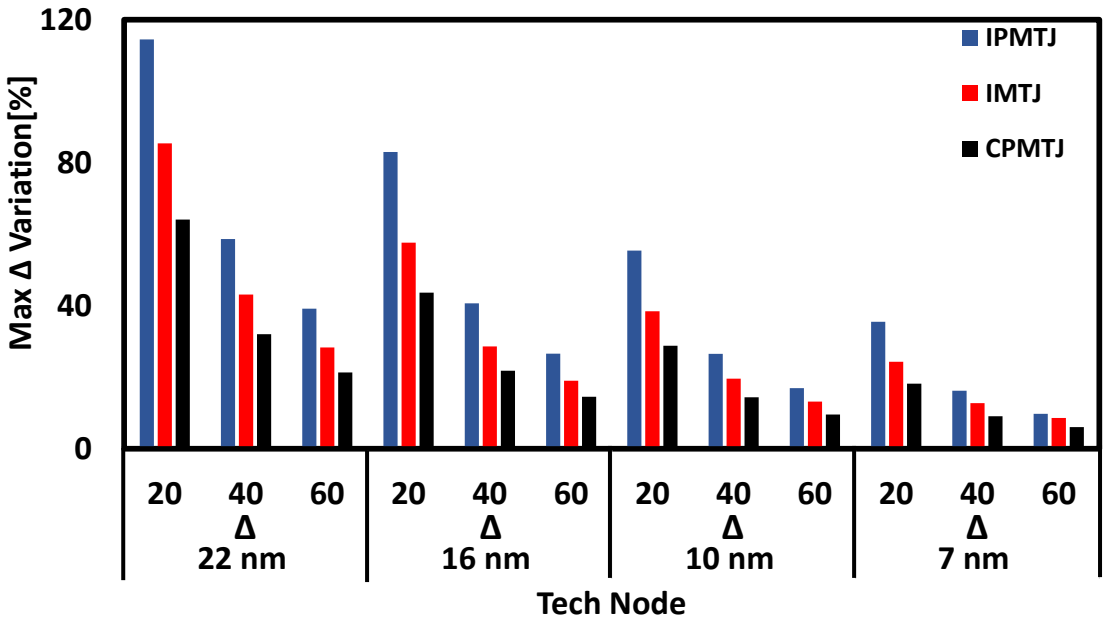


Figure 2.18: Maximum variation of Δ in IMTJ, CPMTJ and IPMTJ across combinations of data pattern and cell array configuration in technology node and nominal Δ.

where τ_0 is 1ns. Since Δ is affected by magnetic coupling (2.6), the retention times for MTJ bits are affected as well. The loss of retention from the nominal retention time is exponential of the Δ variation and it exhibits the same trend as Δ variation in three MTJs.

Fig.2.19 represents the maximum variation in retention across nominal Δ and technology nodes. Since the variation in retention is in exponential relationship with variation in Δ, the variation tends to be very large in a cell with 22nm and Δ = 20 and it exhibits decreasing trend in retention variation as technology node decreases and nominal Δ increases.

2.4 Magnetic coupling effect on thermal stability with synthetic anti-ferromagnet fixed layer

So far, we have explored the effects of magnetic coupling on static and dynamic characteristics of STT-MRAM under the assumption that the fixed layer of MTJs are ferromagnets and exert magnetic fields in their neighborhood. Therefore, large portion of external mag-

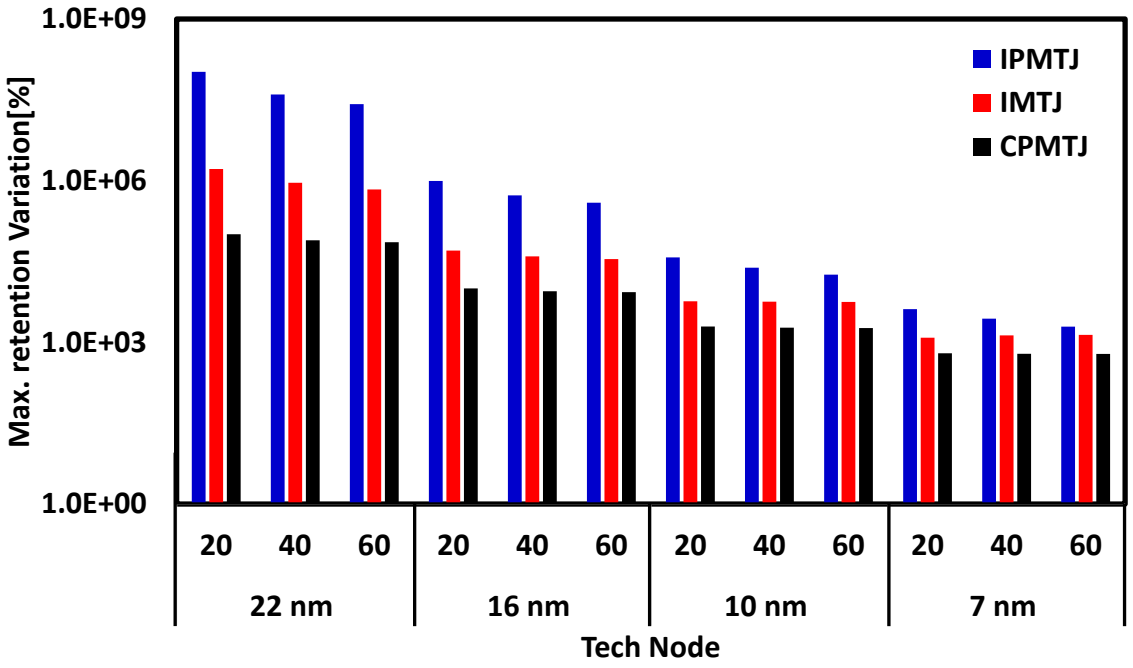


Figure 2.19: Maximum variation of retention time in IMTJ, CPMTJ and IPMTJ across combinations of data pattern and cell array configuration in technology node and nominal Δ.

netic field exerted on the victim cell is emanated from fixed layers of MTJs in adjacent cells. However, when fixed layer of an MTJ is a synthetic anti-ferromagnet, i.e. the magnetic field lines from the fixed layer will close on itself, the results of the same analysis is expected to be very different. In order to compare the results, we model the interaction of STT-MRAM bit-cells without any magnetic field contribution from the fixed layer. We observe that the variation in thermal stability across different technology nodes and cell sizes reduce significantly but they are still not negligible in scaled nodes.

Fig.2.20 shows the maximum variation of thermal stability in IMTJ, CPMTJ and IPMTJ across various data patterns. The variation has decreased to approximately $\frac{1}{10}^{th}$ of what we observed in Fig.2.18. Since the variation in J_{c0} and t_{wr} of a cell exhibit less variation compared to thermal stability variation, we can deduce that the variation in J_{c0} and t_{wr} is greatly reduced when fixed layer of an MTJ does not exert any magnetic field on this neighbors. However, given the exponential relationship between Δ and the retention time, care must be taken when data is stored in the STT-MRAM array over long periods of time.

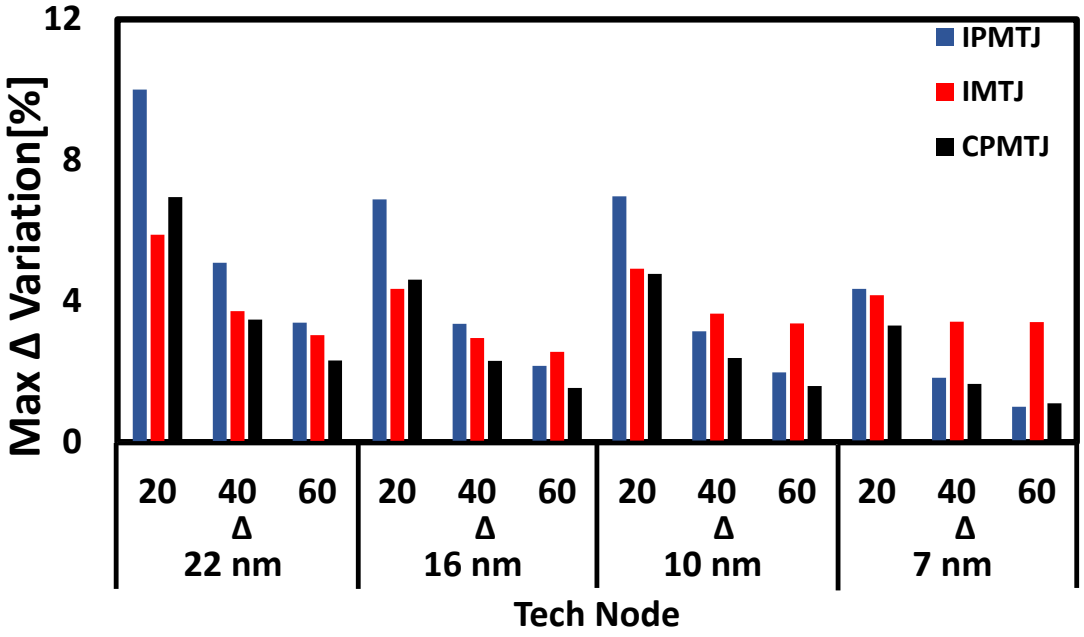


Figure 2.20: Maximum variation of Δ in IMTJ, CPMTJ and IPMTJ across combinations of data pattern and cell array configuration in technology node and nominal Δ.

2.5 Summary

In this research, we present a model of magnetic field induced coupling between adjacent bits in an STT-MRAM array. A comprehensive analysis, across four technology nodes and different MTJ technologies has been presented. We have analyzed the role of the magnetic coupling on electrical performance, both static and dynamic. We conclude that for MTJ technologies with dense memory bits and lower stored energy, the coupling field can cause significant change in the average retention time. Data patterns that activate the worst and best case scenarios have also been explored. It should be noted that the research explores ultra-dense memory bit cells with cell sizes which are $15F^2$ and $6F^2$.[75] The state-of-the art bit-cells are significantly larger ($2\times$ to $3\times$ larger) and effects such as magnetic coupling will be reduced. However, key observations such as the data pattern dependence of retention, will remain unchanged and as the technology matures and denser bit-cells are enabled, magnetic field induced coupling will play a key role in both design and test.

CHAPTER 3

RETENTION TEST CHALLENGES OF STT-MRAM ARRAYS

As we mentioned in the introduction, STT-MRAMs can suffer from retention failures. The non-volatility (or retention characteristics) of the bit can be measured by the thermal stability factor Δ. [37][38] describe retention failure as a bit-flip in a cell caused by thermal noise. The thermal activation model of STT-MRAM in [37] suggests that a bit flip has a poisson distribution with time constant of $\tau.e^{\Delta}$ where $\tau \approx 1ns$. Conventional test methods for retention have very large number of test times. we explore worst case test patterns and propose a Memory Built In Self-Test (MBIST) architecture that can detect the retention failures along with read and write faults in a time-efficient manner. We propose EMACS as an efficient MBIST architecture that can perform in-situ read, write and retention (stochastic test) tests on STT-MRAM arrays. This work is based on a vertically-integrated, device to array modeling infrastructure that we have developed to analyze the physics of MTJ operation (amidst variations and thermal fluctuations) for various types of MTJ cells. The rest of the chapter is divided as follows. In section I, we discuss the challenges and necessary test patterns for retention testing. The MBIST architecture and circuits are discussed in section II. Performance Analysis of EMACS is discussed in section III. Some practical challenges and efficiency of EMACS is discussed in section IV. Finally conclusions are drawn in section V.

3.1 Test and Characterization of cell Retention

In STT-MRAM, retention time is defined as the time it takes for a cell to flip, a stochastic phenomenon, caused by thermal noise [64]. The average retention time is quantified as: $\tau = \tau_0 exp(\Delta)$ and $\Delta = \frac{K_u V}{k_B T} = \frac{H_k M_s V}{2 k_B T}$ [64]. In order to ensure system reliability, each cell in an array must have enough thermal stability(Δ = 60 to guarantee 10 years of re-

tention) against stochastic bit flip induced by thermal noise. With high Δ, a cell can have long retention but high Δ affects increase in write time and current. Due to this trade-off between power consumption and retention, [76][77][78] propose the use of quasi-stable cells with lower Δ to be used in caches. Whatever the design target may be, determining Δ in post-Silicon characterization and manufacturing tests is of utmost importance. However (1) the statistical nature of thermally activated bit-flips, (2) low failure probabilities, (3) large dependence on temperature and process parameters (M_S, H_K, t) and (4) exponential dependence of retention times and retention failure probability on Δ make it a challenging test problem, as has been noted in the Intel publication [37].

3.1.1 Challenges in Retention and Thermal Stability Tests

Very little work exists in published literature on test schemes and challenges from testing retention and thermal stability. While discussing the challenges in [37], Intel proposes a possible test methodology based on the thermal activation model.

$$P_{\text{sw}} = 1 - exp\left(-t/exp\left(\Delta\left(1 - \frac{I_{\text{WWR}}}{I_{\text{c0}}}\right)\right)\right) \tag{3.1}$$

P_{sw} is a switching probability of a cell and I_{WWR} is a Weak Write (WWR) current. The model is is used to obtain the values of I_{c0} and Δ by fitting bit-level experimental/test data [37][38][79][80]. From the thermal activation model for the case:

$$\frac{t_{\text{p}}}{\tau_0 exp(\Delta(1 - \frac{I_{\text{WWR}}}{I_{\text{c0}}}))} << 1 \tag{3.2}$$

using Taylor expansion and ignoring higher order terms [38][37]:

$$ln(P_{\text{sw}}) = ln(\frac{t_{\text{p}}}{t_0}) - \Delta(1 - \frac{I_{\text{WWR}}}{I_{\text{c0}}}) \tag{3.3}$$

where t_p is the pulse width for switching current. This model links P_{SW} and Δ under application of I_{WWR}. Since the thermal activation model is a stochastic model, a large number of successive tests is required to obtain statistically significant results. Also, the model is accurate when low switching current is applied during the long pulse width [37]. Experimental data from [38][79] suggests that switching current ratio of $\frac{I_{\mathrm{WWR}}}{I_{\mathrm{c0}}} \leq 0.8$ and switching pulse width of t_p = 100ns are the upper bounds of the thermal activation model for $P_{\mathrm{sw}} \leq 1e-3$ [37] (Fig. 3.1). P_{sw} of $1e-3$ with ± 1 percent error margin and 99 percent confidence requires 5e+5 number of tests [81][37]. Based on this model, [37] proposes a test scheme where 100ns I_{WWR} pulses are applied and each bit read to determine a possible bit flip. After 5e+5 such tests with 10 different values of $\frac{I_{\mathrm{WWR}}}{I_{\mathrm{c0}}}$, generated by an embedded MBIST, we can obtain statistically significant test data to determine Δ through post-processing. Based on [37] the test algorithm is shown below:

Result: Obtain P_{sw} for every cells in an array
initialization;
N_{row} = total number of rows;
Iwwr[N] = array that contains N number of I_{WWR} value;
M = total number of experiments per each I_{WWR};
for *i = 0; i <N_{row}; i++* **do**
  for *j = 0; j <N; j++* **do**
    Write test patterns into the line;
    for *k = 0; k <M; k++* **do**
      Apply current I_{WWR}[j] for t_p;
      Read the line value;
      if *value ≠ test pattern* **then**
        error counter of cells with error++;
        rewrite correct value to the row;
      end
    end
  end
end

Algorithm 2: Retention test algorithm with weak WR current

Using an MBIST the total test time is approximately 16mins to test two thousand lines of array when N is 5e5 with 10 I_{WWR}, t_p =100ns. Even though parallelism at a sub-array

level can help to reduce retention test time, there is a clear limit in reducing the total retention test time. With increasing size of cell array, the retention test time with this MBIST is not feasible. Therefore, there is a strong need for efficient retention test algorithm which can reduce test time significantly. We address this issue in the next section.

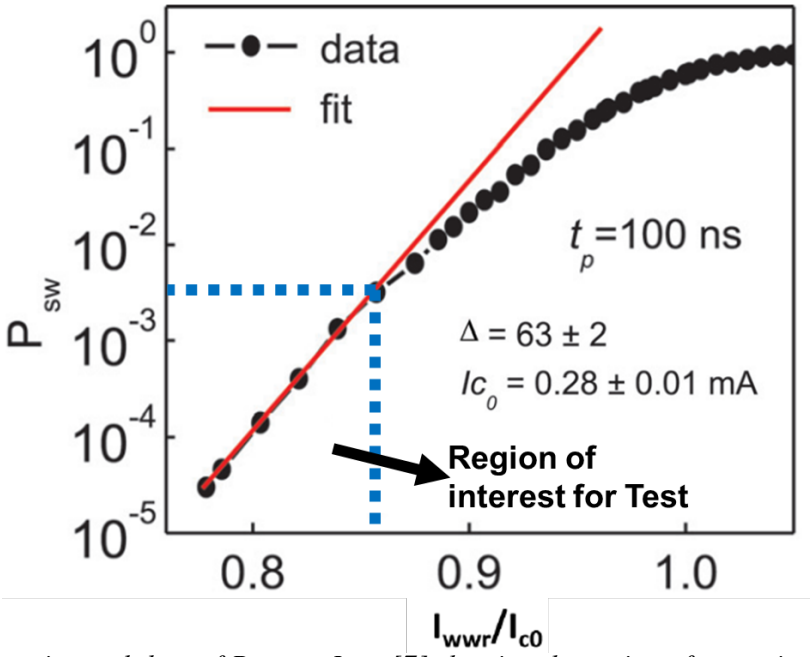


Figure 3.1: Experimental data of P_{SW} vs. Iwwr[7] showing the region of operation for test where the exponential thermal model is valid.

3.1.2 Test patterns for retention test: Role of Magnetic Coupling

From the analysis of magnetic coupling, we identified the worst case data patterns for retention testing under magnetic coupling. Fig. 2.10 indicates the worst data patterns of IMTJ and PMTJ cells under which magnetic field coupling degrades the thermal stability the most. In order to consider magnetic coupling effect in retention test, we need to set the test pattern which has most impact on thermal stability. We first write the data pattern based on Fig. 2.10; and then perform retention test for cells under magnetic coupling. Fig. 3.2 indicates the two block data patterns for testing worst case stability in I-MTJ arrays. For P-MTJ the worst case pattern is all-ones.

	C0	C1	C2	C3	C4	C5
R0	1	0	1	0	1	0
R1	1	1	1	1	1	1
R2	1	0	1	0	1	0
R3	1	1	1	1	1	1
R4	1	0	1	0	1	0
R5	1	1	1	1	1	1

(a) IMTJ block data pattern A

	C0	C1	C2	C3	C4	C5
R0	0	1	0	1	0	1
R1	1	1	1	1	1	1
R2	0	1	0	1	0	1
R3	1	1	1	1	1	1
R4	0	1	0	1	0	1
R5	1	1	1	1	1	1

(b) IMTJ array data pattern B

Figure 3.2: IMTJ worst case data patterns for retention shown in a 5×5 grid. For PMTJ the worst case pattern is all-ones.

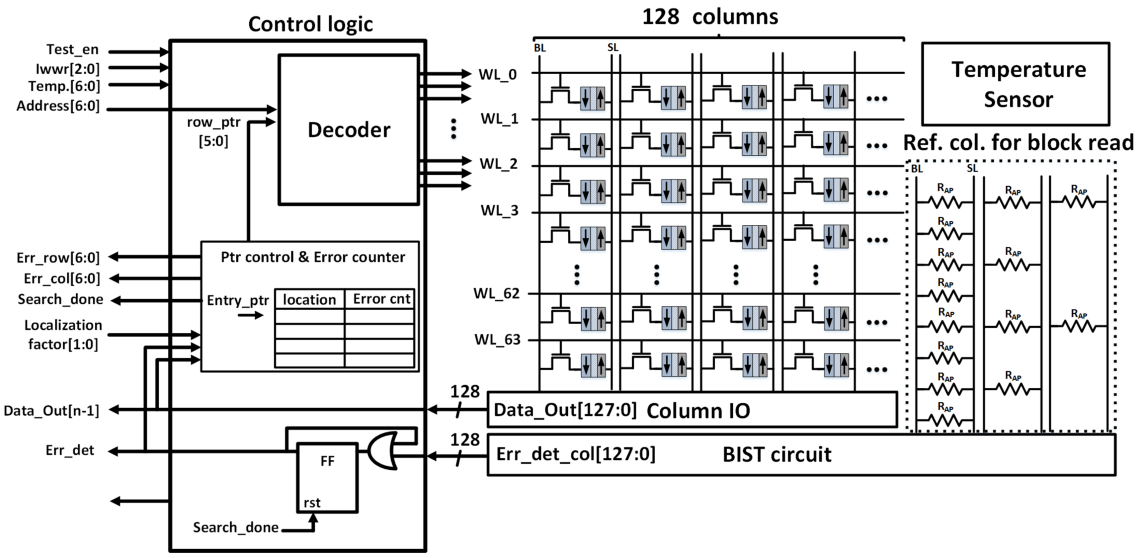


Figure 3.3: System architecture of EMACS MBIST applied to a 64×128 array. EMACS is capable of read, write and statistical retention tests.

3.2 Proposed MBIST for Retention Testing

We extend EMACS to perform in-situ, statistical, retention testing of large STT-MRAM arrays. From the retention BIST algorithm [37], we apply a weak current and read the value of cells row by row to obtain P_{sw}. The principle drawbacks of the above scheme that we identify are:

(1) The retention test time increases linearly when the row size of an array increases.

(2) The retention tests have to be carried out in an operating region where P_{SW} is very low. For example, for a cell with Δ = 60, applying $\frac{I_{WWR}}{I_{c0}}$ from 0.76 to 0.82 for t_p = 100ns sets P_{sw} to be 5.573e-5 to 0.002 based on Eqn 3.3. It indicates that for most of the iterations, a bit flip will not happen; which means most of the read operations after applying current are not necessary.

These two problems are main bottlenecks for improving speed of retention test. The retention test methodology and MBIST architecture that we propose focuses on how to overcome these two bottlenecks. If an error can be detected in an entire cell array with a fixed number of memory operations, we can decouple array size from the factors affecting retention time. Also, since the probability of occurrence of bit flip is low, rather than reading rows each time after applying current in search of a bit-flip, reading rows only after an error is detected will reduce retention test time. The proposed architecture reduces retention time significantly by:

1. Detecting errors column-wise
2. Avoiding search (reading rows) when error is not detected

By testing multiple rows in a column at the same time and searching for errors after error detection, retention time testing reduces significantly. The retention test is divided into two processes, (1) Error Detection (ED) and (2) Error Search (ES).

3.2.1 EMACS System Architecture for Statistical Retention Tests

Fig. 3.3 presents the top level system diagram of the proposed MBIST circuitry. Normal memory operation and test operation are distinguished by the test_en signal. For retention test, Error Detection (ED) and Error Search (ES) logic are parts of the control logic. Based on the outputs of the MBIST circuit, Error Detection logic asserts err_det signal and while err_det is asserted, Error Search is conducted. Error Search controls which rows to assert

from a localization factor (to be described in the next subsection) and it outputs error location to the output of control logic as soon as it identifies error locations. Search_done signal is asserted if Error Search is over and it resets err_det signal. I_{WWR} bus is used to control voltage of bitline and word-line, which leads to different magnitudes of I_{WWR} current. Column of different resistors are used as a references for finding errors in blocks of rows and temperature sensor are located inside a sub array to monitor temperature inside a sub array. Each characterization test, which produces an experimental determination of Δ, is qualified by a temperature data. The proposed scheme allows massive parallelism in test and enables a fine trade-off between localization of weak cells and test time.

3.2.2 Error Detection (ED)

The ED architecture is based on the MTJ property that any change in data (bit-flip) results in a change in resistance of the cell, which in-turn changes the current flowing through the cell. [82] uses this property to detect read disturb errors, by monitoring current difference (before and after the bit-flip) due to change in resistance.

In the proposed scheme (Fig. 3.4): (1) data patterns based on Fig. 3.2 are first written into the array, (2) retention test started by turning on multiple word-lines simultaneously, (3) I_{WWR} current injected through each cell which is storing a 1, (4) multiple read operations are conducted while passing I_{WWR} to check for a possible bit-flip, (5) next data pattern applied for full-coverage. For IMTJ, two block data patterns are identified in Fig. 3.2. To enable multiple simultaneous tests, odd numbered columns (C1,C3,..) are tested first Fig. 3.4a with block data pattern A, followed by testing of even numbered columns (C0, C2,...) using pattern B. Then the pattern is shifted vertically by one row and the process repeats. For PMTJ, the worst case pattern under magnetic coupling is all-ones, and hence all the columns can be tested simultaneously. Turning multiple word-lines in a column connects the MTJ resistance in parallel as shown in Fig. 3.4. The resistance of a MTJ is set to R_{ap} since cells store bit 1 in the figure. When I_{WWR} causes a bit flip in a cell, the

resistance of a MTJ will change from R_{ap} to R_p as shown. The current flowing through source line of a column (I_{SL}) changes due to the resistance change. By detecting difference in I_{SL}, we can detect the existence of errors in a column.

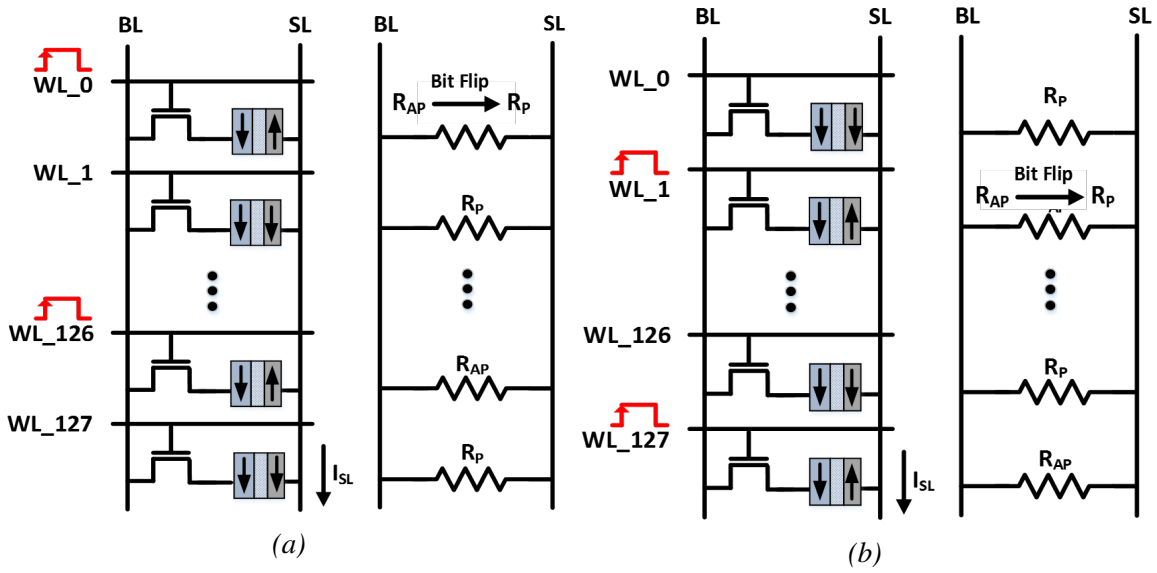


Figure 3.4: Multiple word-lines simultaneously turned ON to detect bit-flips according to worst case patterns identified in Fig.10 for IMTJ. (a) Simultaneous testing with block data pattern A applied to C1,C3,.., (b) Simultaneous testing with block data pattern B applied to C0,C2,... For PMTJ the worst case pattern is all-ones so all word-lines are turned on simultaneously.

However, due to low (150%[83]) TMR($= \frac{R_{ap}-R_p}{R_p}$) of the MTJ, the number of rows that can be simultaneously turned on and a bit-flip detected, is limited. With low TMR, the difference of total resistance of a column between a case with no errors and a case with a single error decreases and it affects difference in I_{SL}. Fig. 3.5 presents a trade-off between number of activated rows and the current difference of no error case and one error case with respect to different TMR values. It exhibits decreasing I_{SL} difference in percent as number of activated rows increases with different TMR. Due to process variation and temperature fluctuation, appropriate number of activated row must be set to gain enough margin in current difference. In this work, we limit the number of activated rows to 16, to distinguish between the "no error" and a "single error" in a column.

In the proposed test scheme, unlike the testing scheme from [37], we check errors while

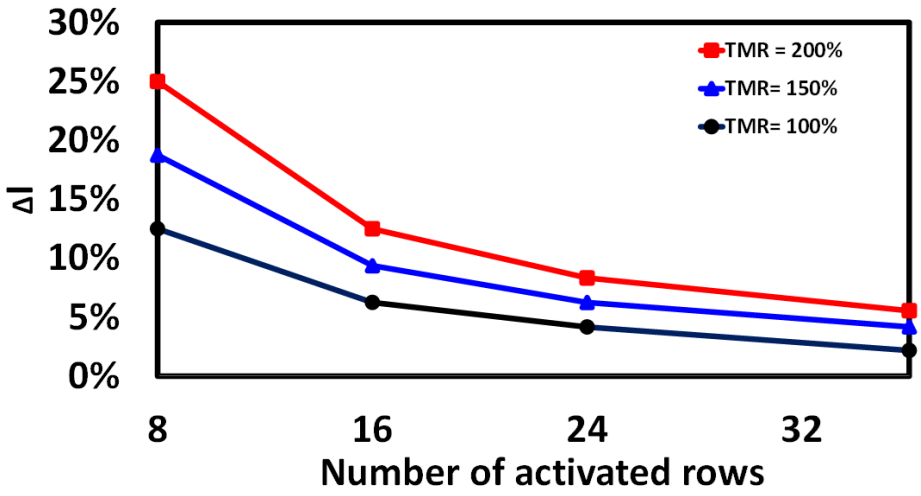


Figure 3.5: ΔI_{SL} vs. number of rows activated as a function of TMR

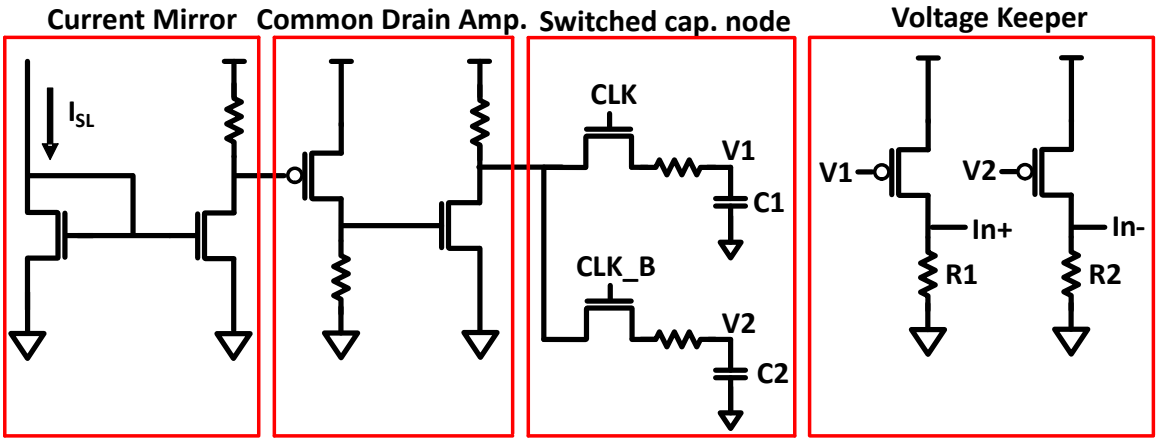


Figure 3.6: Error Detection circuit for a column with 16 rows

supplying I_{WWR} through 16 rows of cells. Since test scheme must apply $I_{WWR} \leq 0.8I_{c0}$ for t_p and it is same as a strong read/weak write operation, we can detect errors within 16 rows in a column by monitoring I_{SL}. The read operation overhead after weak write is removed for the case when no error is detected during t_p. From this scheme, we can reduce t_{read} by $(1 - P_{sw})N$ per I_{WWR} iteration, N is the number of test per I_{WWR} Fig. 3.6 shows the scheme for detecting a change in I_{SL} caused by a bit flip of a cell. The change in I_{SL} is amplified by current mirror and it is transferred to voltage difference and further amplified by multi stage common drain amplifier. Switched capacitors C1 and C2 sample the voltage at the common drain amplifier alternatively based on CLK and CLK_B signals. When bit-flip happens,

the voltage difference between C1 and C2 is developed and maintained for a half clock cycle. Since the node voltages at C1 and C2 fluctuate when they are directly connected to the inputs of sense amplifier due to their small size, we implemented voltage keepers in between to avoid voltage fluctuation. By calibrating value of R1 and R2, in+ port is set to be always 10mV higher than in- to prevent metastability issue in sense amplifier. When sense amplifier enable is on, the sense amplifier fully differentiates the in+ and in- to VDD and GND. Fig. 3.7 presents waveform of switched capacitor control signals(CLK, CLK_B) and sense amplifier enable. Once WLs are asserted to supply I_{WWR} for t_p, CLK and CLK_B toggle to sample the voltage to C1 and C2. After capacitor C1 and C2 develop common mode voltage within t_{dev}, sense amplifier enable signal is asserted in the middle of every half clock cycle. Discharging of C1 or C2 must be finished before sense amplifier enable is asserted to apply maximum voltage difference in port in+ and in- of sense amplifier.

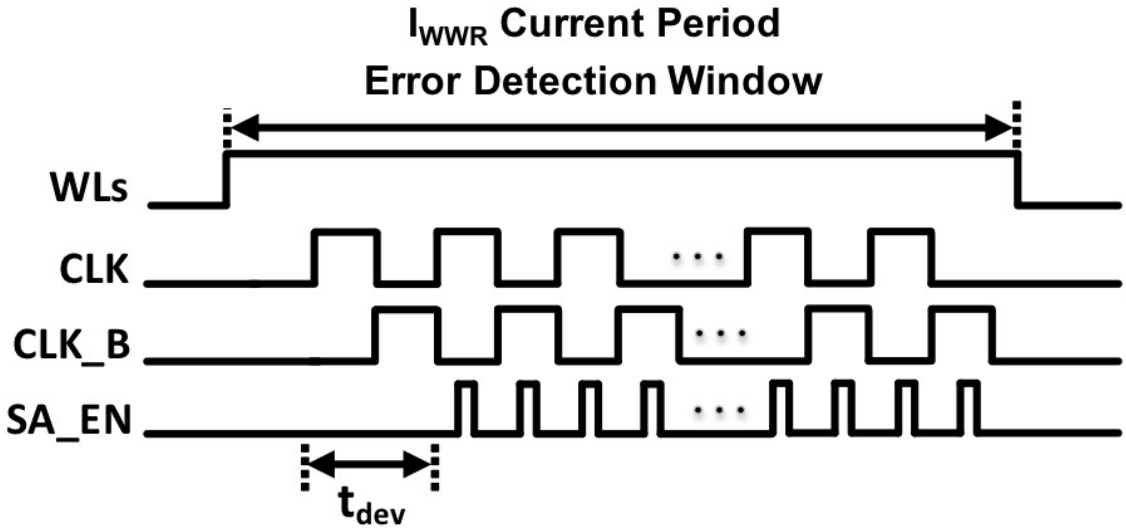
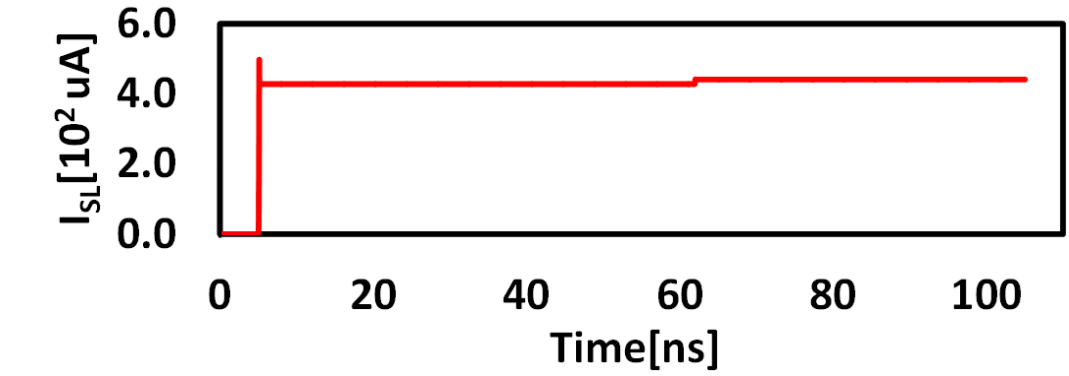
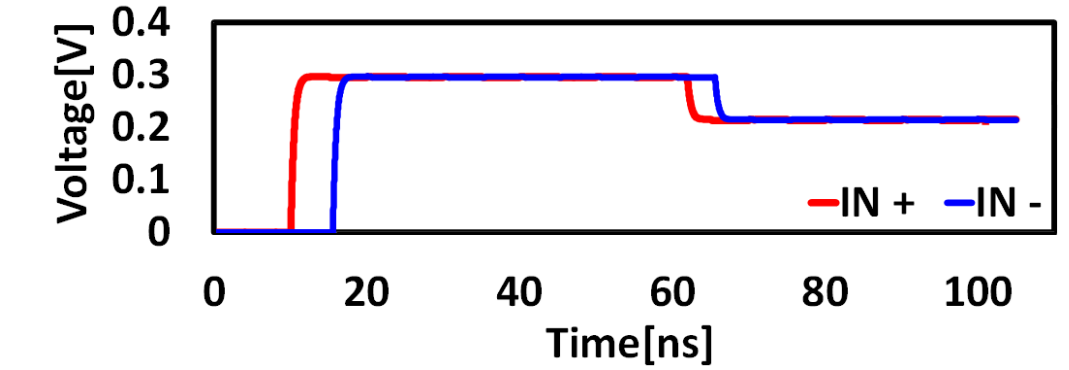


Figure 3.7: Timing Diagram illustrating the operation of the MBIST retention test

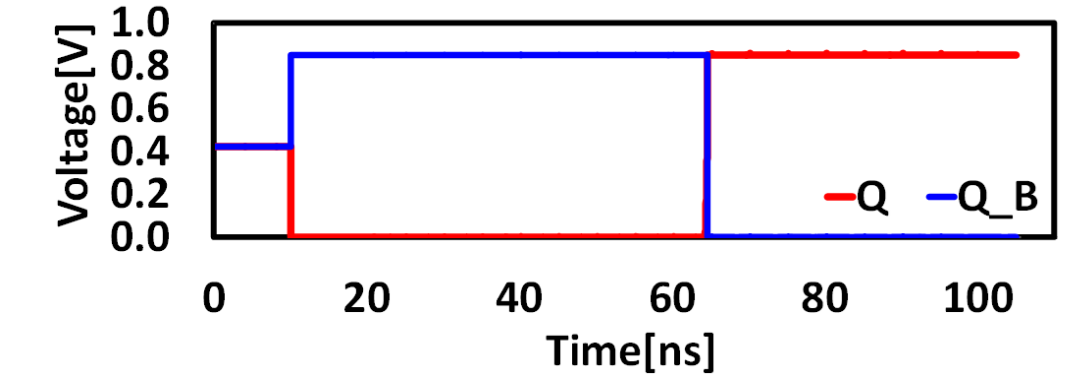
Fig. 3.8 shows the voltage across switched capacitors(C1, C2) and sense amplifier output when bit-flip happens. Around 60ns in Fig. 16a, current through SL is seen to increase due to the change in resistance ($R_{AP} \rightarrow R_P$) from a bit flip. Voltage difference across switched capacitor is maintained for half clock cycle in Fig. 16b and the sense amplifier resolves the voltage difference to VDD and GND when sense amp. enable is on. Fig. 3.9a



(a) Current through SL



(b) Voltage at the two switched cap.

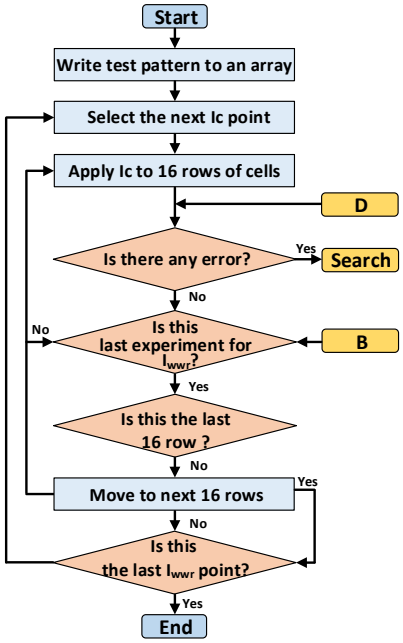


(c) Sense Amp. Output

Figure 3.8: Transient analysis for error detection

summarizes the test-procedure and Fig. 3.9b shows the corresponding algorithm. Test flows A, B, C, D in yellow bounding boxes are presented in individual diagrams in Fig. 3.12,3.13. The main differences between proposed test scheme and [37] are error detection and search algorithms. In [37], the authors propose to apply I_{WWR} for t_p and read a row for every rows in an array. Instead, the proposed test scheme applies I_{WWR} to a block of rows and search

for errors only when existence of error is identified by detecting a change in current through source line of a column. Fig. 3.10 illustrates a particular simulation run showing infrequent bit flips happening over time which are recorded in the current scheme. This allows estimation of P_{sw} and finally extrapolated to obtain Δ via Eqn. 3.1. The P_{sw} for a cluster of cells within an 8KB subarray is shown in Fig. 3.11. After ED, a search algorithm to localize the bit-flip is used and is discussed next.

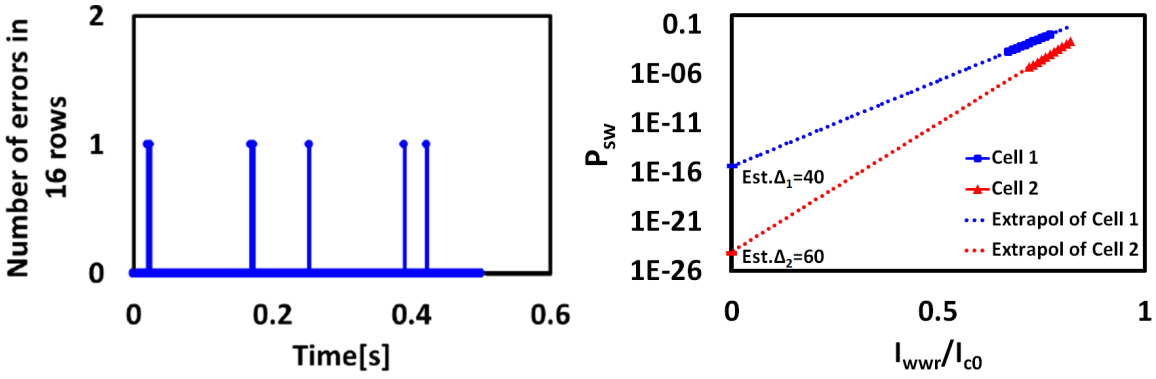


(a)

Result: Detect errors within N_{act} rows in each column
N_{row} = total number of rows;
$I_{wwr}[N]$ = array that contains N number of I_{wwr} value;
M = total number of experiments per each I_{wwr};
Write 1 to all cells;
for $k = 0; k < N; k{+}{+}$ **do**
  for $i = 0; i < N_{row}/N_{act}; i{+}{+}$ **do**
    for $j = 0; j < M; j{+}{+}$ **do**
      while *Apply* $I_{wwr}[k]$ *to* N_{rows} *rows for* t_p **do**
        if $In+ \neq In-$ **then**
          error = 1;
        end
      end
      if *error* **then**
        Search;
      end
    end
    select next N_{act} rows;
  end
end

(b)

Figure 3.9: (a) Flow chart (b) algorithm for bit-flip detection in a column



(a) Bit Errors as a measure of bit-flips over time *(b) P_{sw} vs.. I_{WWR}/I_{c0}*

Figure 3.10: Estimating P_{sw} and Δ through EMACS

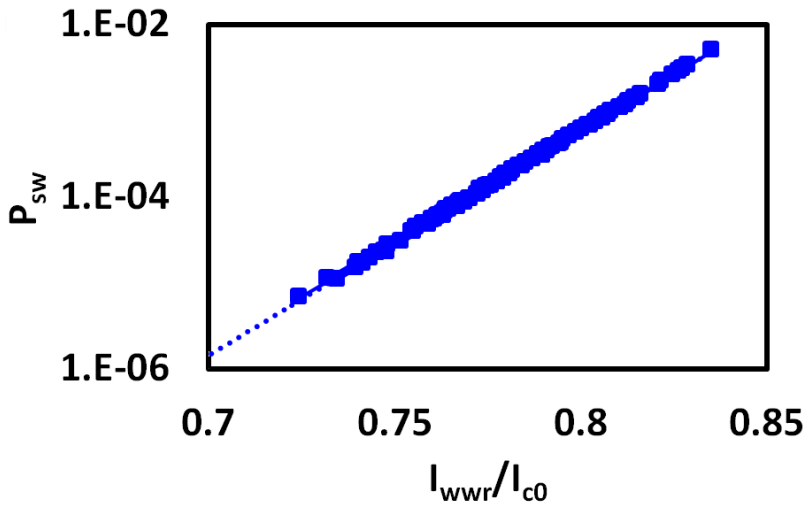


Figure 3.11: P_{sw} on a cluster of cells in an 8KB array showing a scatter which can be extrapolated to obtain Δ

3.2.3 Error Search and Localization

After detecting the existence of errors using the scheme above, searching the location of errors within the activated rows is necessary in order to obtain P_{sw} and thermal stability of cells. In this paper, we present three different error search schemes (exhaustive search, temporal locality search and search localization).

Exhaustive Search

The algorithm used after detecting the first error is exhaustive search. In exhaustive search, every row in a block of activated rows are read to locate errors. Once it obtains location of an error, the test scheme stores the location in a error table and re-writes original test pattern to a row with an error. When the last row in a block is read, it goes back to error detection flow algorithm. Error location stored in a table is used in a search which exploits temporal locality. Fig. 3.12 demonstrates each steps in exhaustive search.

Temporal locality search

Temporal locality search can reduce error search time when process variation on thermal stability of cells is large. The efficiency of search improves when performing manufactur-

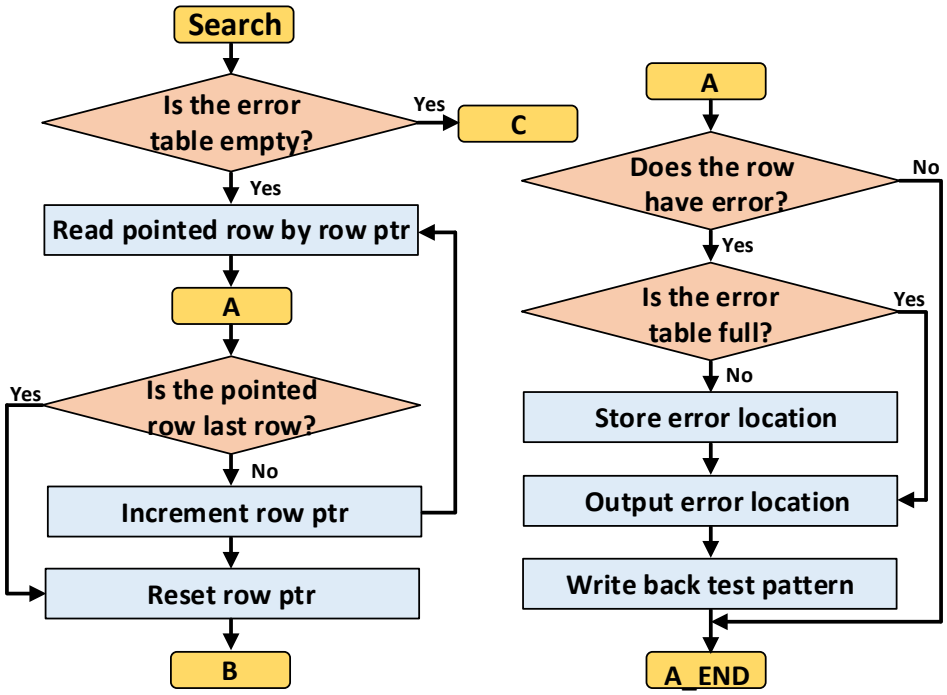


Figure 3.12: Flow chart for exhaustive error search

ing test, the test that identifies cells which do not meet target retention. Fig. 3.13 presents each steps in temporal locality search. Once error table is filled from exhaustive search, temporal locality search first reads rows in the table to locate errors. If the row specified in the table contains an error, it updates number of errors associated with the row in a table. When no error is found in the rows from the table, it switches to exhaustive search to find errors in other rows and add a row to a table when error is found in the row. After it finds an error, it reads the block of rows to ensure it corrected all errors.

Error Localization Search

Both exhaustive search and temporal locality search identify all locations of errors in the array. In terms of search time, however, both search scheme can be time consuming if the block size of activated rows for error detection is large. Instead of identifying which row contains errors for each column, we can set a block size in terms of row(N_{loc}) and search whether the block contains errors. For example, searching errors within 4 rows each time is 4 times faster than exhaustive row search. By reducing accuracy of error

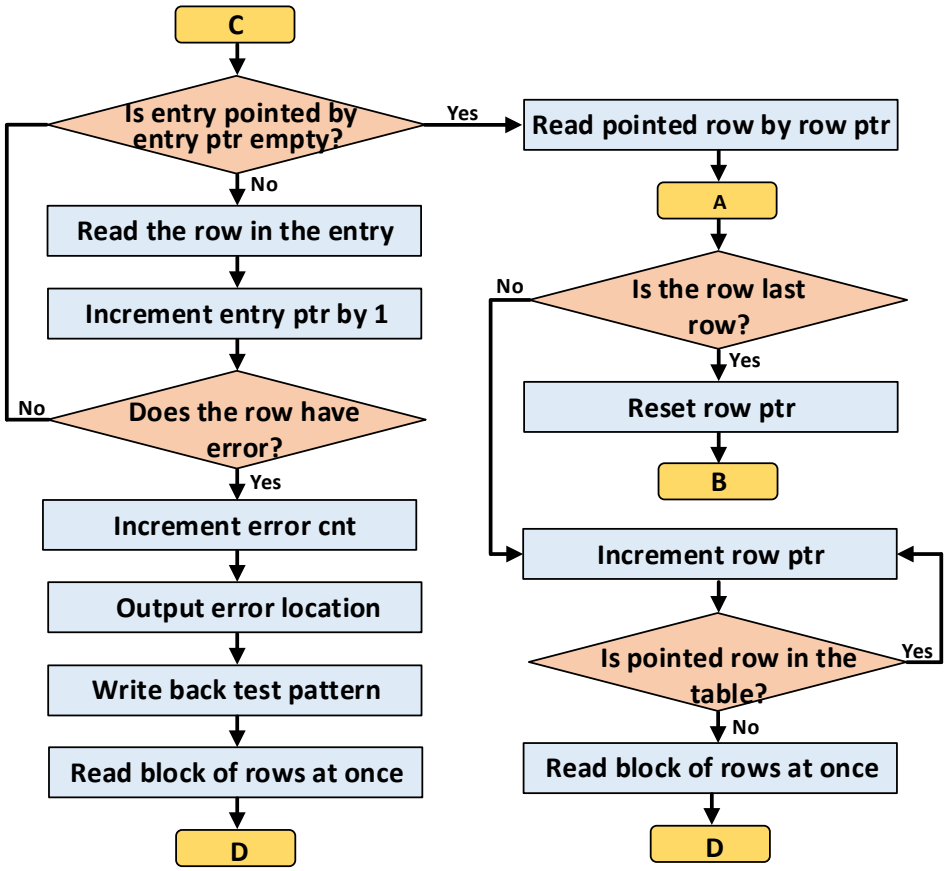


Figure 3.13: Flow chart for temporal locality search

position, we reduces search time linearly as N_{loc} increases. Fig. 3.14 presents how search time varies based on the size of N_{loc}. The search time is compared with 5 different levels of localization. Table IV indicates how localization level maps to N_{loc}. Since search is conducted when error is detected, search time is a multiplication of error probability(P_{sw}), read time and $\frac{N_{act}}{N_{loc}}$. N_{act} is the number of rows activated for error detection. Search time in the Fig. 3.14 is calculated with the assumption that P_{sw} = 3e-3, number of I_{wwr} = 10 and number of experiments per I_{wwr} = 5e5. As we mentioned earlier, the search time decreases linearly when N_{loc} increases in the figure.

$$t_{\text{search}} = P_{\text{sw}} \times t_{\text{read}} \times \frac{N_{\text{act}}}{N_{\text{loc}}} \tag{3.4}$$

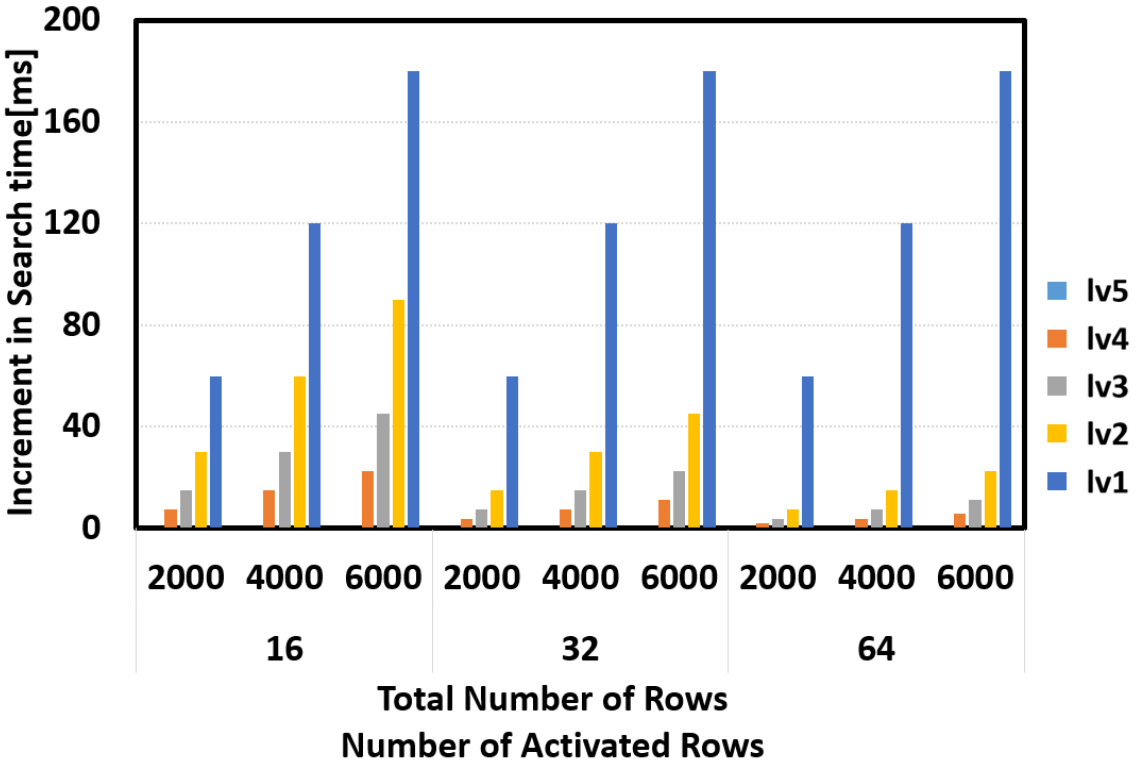


Figure 3.14: Search time increment w.r.t. localization level

Table IV: Localization level in terms of no. of rows

Localization level	Block size(row)
5	N_{act}
4	0.5 N_{act}
3	0.25 N_{act}
2	0.125 N_{act}
1	1

3.2.4 Overhead of internally storing data

Retention test requires at least $8N_{cell}$ (total number of cells in an array) bits of memory to store number of bit flips per cell under assumptions that the maximum P_{sw} = 3e-3 and number of experiments per I_{wwr} is 5e5 to calculate thermal stability of each cells. Instead of storing error counts in the memory, test scheme can output row & column information when error is detected to calculate thermal stability outside the chip. However, it adds complexity to test mode control logic and outputting error location is also time consuming. It should also be noted that block level identification of cell stability allows us to apply

redundancy easily. Once a particular column is identified as having weak (low Δ) cells, we can swap it with a redundant column. So in manufacturing tests, localization at the granularity of a column is sufficient.

3.3 Performance analysis

3.3.1 Test time Comparison

The retention test time of proposed test scheme can be calculated using the equation;

$$t_{\text{ret}} = [(t_{\text{p}} + t_{\text{search}}) \times \frac{N_{\text{row}}}{N_{\text{act}}}] \times M \times N_{\text{I}_{\text{wwr}}} \tag{3.5}$$

N_{row} is the total number of rows in an array, M is the number of experiments required for each I_{wwr} and $N_{I_{wwr}}$ is the total number of I_{wwr} needed to extrapolate P_{sw} vs. I_{wwr} to obtain cell retention. t_{search} is defined in equation 3.4. Fig. 3.15 presents the performance analysis in terms of time between [37] and EMACS. For testing retention for an array with 2000 rows, test scheme from [37] takes 16 mins to complete and proposed test scheme takes 1 min with N_{act}. If we increase N_{act} to be 32 and 64, the test time reduces to $\frac{1}{32}$, $\frac{1}{64}$ of the test time from [37].

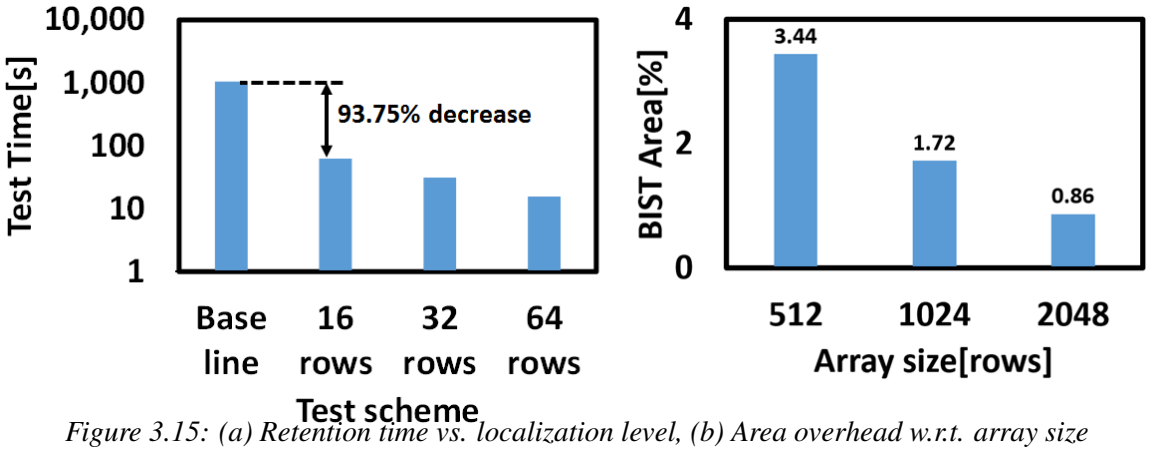


Figure 3.15: (a) Retention time vs. localization level, (b) Area overhead w.r.t. array size

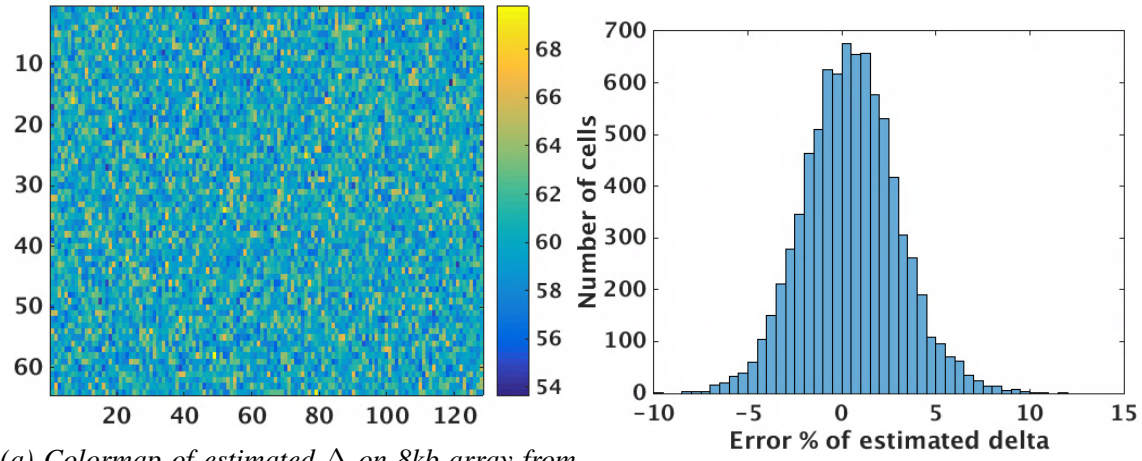
3.3.2 Area overhead

Fig. 3.15(b) presents the area overhead of the proposed retention MBIST. For this analysis, we did not use any column mux techniques to reduce number of test circuit by half. Each column contains one set of test circuit including sense amp. described in Fig. 3.6 in the analysis. We assumed that each cell size in the array is $30F^2$ to calculate the area overhead of test circuit with respect to total array area. From Fig. 3.15(b), we deduce that area overhead of test scheme decreases linearly with respect to number of rows in the array. With 512 rows, area overhead is 3.44% of the total cell array size and it reduces by half when number of rows doubles.

3.4 Array Level Testing and Challenges

The proposed test-scheme, albeit a practical and faster test methodology, is still a statistical test enabled by an MBIST and suffers from measurement errors arising due to temperature changes and process variations. Since retention times are heavily dependent on temperature, we propose (1) to use embedded thermal sensors within the subarray to qualify each sub-array measurement with the corresponding temperature, or (2) insert idle states in between applying I_{wwr} and error detection process to maintain stable temperature. Another potential problem in the test-scheme is the process induced mismatches between cells. When a block of cells are written and read simultaneously, the I_{WWR} is not equally divided between the cells. This creates loss of accurate measurement of Δ and needs to be accounted for as a design guard-band. We carried out simulations of the EMACS test scheme by running tests under temperature and process variations and trying to estimate Δ on an 8KB subarray amidst all the non-idealities. Fig. 3.16(a) presents the estimated thermal stability of 8kb cell subarray and Fig. 3.16b shows the accuracy of the test methodology for the collection of 8KB cells. It can be seen that the proposed scheme has bounded error of $< \pm 5\%$ and 93.75% decrease in test-time with respect to [37] and demonstrates the

effectiveness of the proposed test methodology.



(a) Colormap of estimated Δ on 8kb array from EMACS

(b) Error in estimation of Δ

Figure 3.16: Estimated Δ and Error of estimation from 8kb array using EMACS. The colormap represents cells in a 64×128 array.

3.5 Summary

This chapter presents a comprehensive test methodology for STT-MRAM arrays. An MBIST architecture (EMACS) capable of collecting statistical data in an STT-MRAM sub-array to estimate the thermal stability and retention is proposed. The proposed MBIST shows 93.75% improvement in test-time compared to a brute-force approach [37] with less that 5% estimation error.

CHAPTER 4

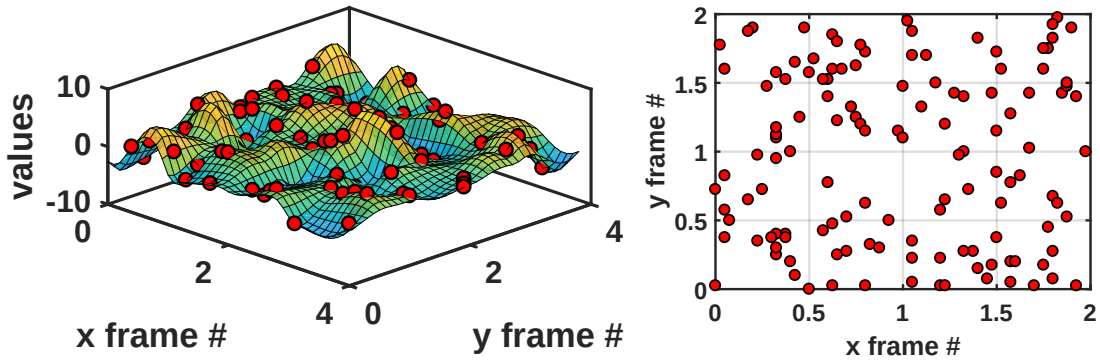
FERROFET BASED IN-MEMORY PROCESSING ARCHITECTURE

Modern computing systems based on the Von-Neumann architecture rely on a clear distinction between logic and memory, and processes information by executing a sequence of precise atomic instructions with periodic uploads to the memory. Such systems are the foundation of the digital revolution which began with the demonstration of the self-aligned planar-gate silicon MOSFET in the sixties and was accelerated by rapid advances in transistor technology. However, in the last one decade, the volume of data collected by distributed sensors and networks has grown exponentially. Ingesting, processing and extracting actionable intelligence out of this abundant data requires large amount of data traffic between logic and memory blocks leading to the problem of memory bottleneck. This requires novel ways of architecting the compute platform. For example, by embedding processing elements in the memory sub-array itself in so called Processing-In-Memory (PIM) architectures [39, 40, 41, 42], the traditional Von-Neumann bottleneck can be addressed and significant acceleration and improved power-efficiency can be achieved. In order to solve the memory bottleneck problem, current research focuses on architectures and memory arrays that can accelerate memory-based processing for machine learning applications. Designs explore the use of SRAM arrays [84], crossbar arrays with ReRAMs [85, 86, 87], memristors [88, 89] and spintronic MRAMs [90].

Apart from inference, one ubiquitous algorithm in signal processing and autonomous systems is optimization – in particular, convex optimization. Least squares minimization is such a template problem and is the focus of this research. We demonstrate that distributed convex optimization via least squares method can be efficiently implemented in a iterative dynamical system using a systolic PIM architecture, with breakthrough energy-efficiency and performance. In particular, the iterative and parallel nature of memory-read makes the

systolic PIM a good candidate for the proposed algorithm. This is further made possible by a parallel development in device technologies– namely, the advent of multiple embedded non-volatile memories (eNVM). Among all competing eNVM technologies, FerroFETs have emerged as promising candidates due to their compact size, multi-level storage, nano-second read-write and high energy-efficiency. We demonstrate that a systolic PIM architecture, using FerroFET pseudo-cross-point array can solve least squares minimization with $21\times$ improvement in energy-efficiency compared to an SRAM PIM architecture.

4.1 CONVEX LEAST SQUARE MINIMIZATION



(a) *(b)*

Figure 4.1: (a) 2D continuous function f(u,v) with non-uniform samples. (b) Spatial location of the non-uniform samples.

Before discussing the systolic PIM architecture, we present a brief overview of distributed least squares minimization as a template problem, with wide-spread applications in discrete signal processing. In particular, it is a common tool for signal reconstruction where the process of sampling is non-uniform[91, 92] such as in Computerized tomography (CT), magnetic resonance imaging (MRI) [93], radar signal processing, LIDAR systems etc. Consider (1) u and v are the horizontal and the vertical arguments of a continuous signal. (2) x and y are the discrete coordinate indexes. (3) ω_x and ω_y are horizontal and vertical spatial frequencies. Let $f(u, v)$ be a band-limited signal in $\mathbb{R}^2$. The signal is non-uniformly sampled and are stored in vector $\boldsymbol{b}$, which are referred to as $f(x, y)$. The

objective is to use the non-uniform samples to obtain complete reconstruction of $f(u,v)$ in $N_x \cdot N_y$ dimensional subspace. Fig. 4.1 shows an example of $f(u,v)$ and the results of non-uniform sampling. In this algorithm, we assume that $f(u,v)$ lies in an $N_x \cdot N_y$ dimensional subspace. To reconstruct the signal accurately we have used 2D lapped orthogonal transform (LOT) cosine-IV harmonics as the basis functions. A smoothing function $g(u,v)$ is applied to all the basis functions to avoid distortions. Equation (4.1) shows a general LOT cosine-IV basis function. Here, $f(u,v)$ is split into K_x by K_y frames and $[k_x, k_y]$ represent a specific frame, ω_x and ω_y indicate the harmonic in horizontal and vertical directions.

$$\psi_{k_x,\omega_x,k_y,\omega_y}(u,v) = \sqrt{2} \cdot g(u-k_x, v-k_y) \cdot cos((\omega_x + \frac{1}{2})\pi(u-k_x)) cos((\omega_y + \frac{1}{2})\pi(v-k_y)) \quad (4.1)$$

Since $f(u,v)$ lies in a $N_x \cdot N_y$ dimensional subspace, it can be expressed as:

$$f(u,v) = \sum_{\omega_x=1}^{N_x} \sum_{\omega_y=1}^{N_y} \sum_{k_x=1}^{K_x} \sum_{k_y=1}^{K_y} \alpha(k_x, \omega_x, k_y, \omega_y) \psi_{k_x,\omega_x,k_y,\omega_y}(u,v) \quad (4.2)$$

The key point to note here would be that LOT cosine-IV has compact support and the different frames are loosely coupled to each other. In fact, for samples in each frame, the nontrivial dependence would extend only to the adjacent frames apart from itself. According to (4.2), we can write an equation for each sample and collect them into matrix-vector product form and the coefficients can be found by solving the inverse-linear problem of

$$\boldsymbol{A}\boldsymbol{z} = \boldsymbol{b} \quad (4.3)$$

Here $\boldsymbol{b}$ is the sample vector, $\boldsymbol{z}$ is the coefficient vector obtained by stacking the coefficients $\alpha(k_x, \omega_x, k_y, \omega_y)$, and $\boldsymbol{A}$ is referred to as the Grammian (Gram) matrix of the basis.

When the size of $\boldsymbol{A}$ matrix is large (as in most applications) a direct solution is not possible. Therefore, alternatively we follow an iterative approach, the Jacobi method. A general update of $\boldsymbol{z}$ in j^{th} component at the k_{th} iteration is given as (4.4), where $\boldsymbol{B} = \boldsymbol{A}^T\boldsymbol{A}$

and $\boldsymbol{c} = \boldsymbol{A}^T \boldsymbol{b}$.

$$\boldsymbol{z}_j^k = \boldsymbol{B}_{jj}^{-1}(\boldsymbol{c}_j - \sum_{i \neq j} \boldsymbol{B}_{ji} \boldsymbol{z}_i^{k-1}) \tag{4.4}$$

Some observations are worth emphasizing: (1) To update $\boldsymbol{z}_j^k$, only values from previous iterations are need. (2) Columns of $\boldsymbol{A}$ are coupled only with neighboring frames, which leads to simpler computation of $\boldsymbol{B}_{ji}$. Such a system maps naturally to a systolic PIM architecture with (1) near neighbor connections and (2) embedded linear algebraic operators on the periphery of the sub-array – as will be described in the following sections.

4.2 FerroFet PIM Architecture and End-to-end Tool Chain Development

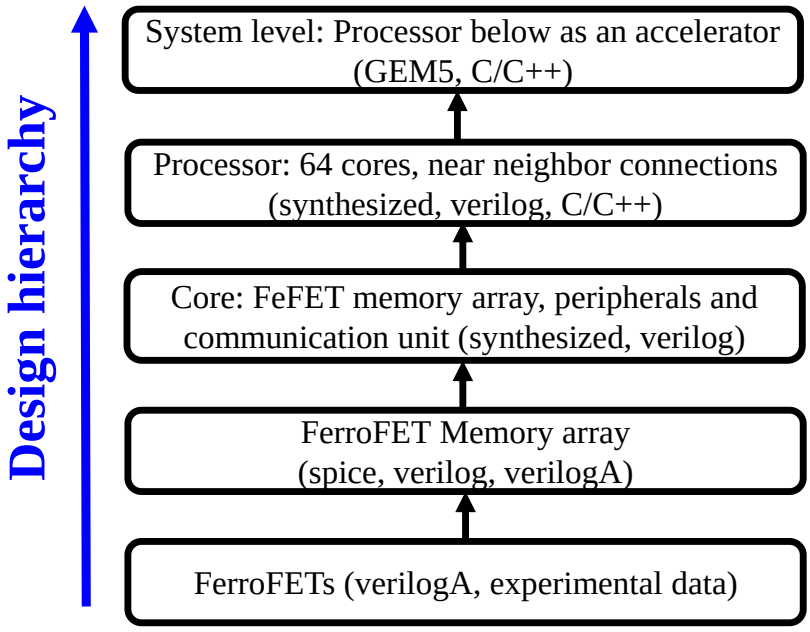


Figure 4.2: Flow-chart of design hierarchy from device to system.

In this research, we explore the FerroFET memory based processing in memory (PIM) architecture in a hierarchical manner.A short description of each layer of the design abstraction is provided here. Fig. 4.2 provides the flowchart of the entire design cycle from devices to the PIM architecture. The salient features are as follows:

1) There are 64 cores, 8 rows, with each row containing 8 cores. With respect to section II

this implies $N_x = N_y = 8$.

2) Each core is capable of performing Jacobi-iterations with subspace dimensions, K_x and K_y (horizontal and vertical dimensions) equal to 8. The subspace dimensions determine the core-complexity and the accuracy of signal reconstruction. From our analysis we identified 8x8 subspace dimensions is sufficient for signal-processing applications in hand.

3) Analog to Digital converters (ADCs) are critical in terms of determining the latency and power consumption. In order to explore the design space properly we have used analog-to-digital converters (ADCs) with different resolutions and design constraints.

4) For the current design the B-coefficients ($B_{jj}^{-1}B_{ji}$) and z-coefficients (z_j^k) are represented in 12 bit fixed point representations where the MSB 6 bits represent the integer part and last 6 bits represent the fractional part.

5) To model the system we have used Spice for simulating bit-cells, Verilog and VerilogA models for array-level circuit architecture simulations and gem5 for architectural simulations.

4.2.1 FerroFET cell structure

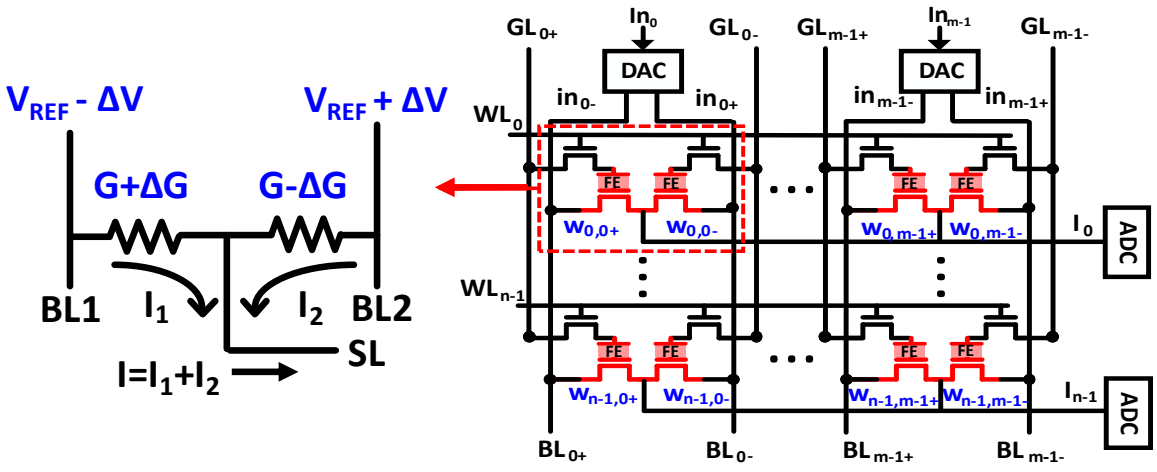


Figure 4.3: FerroFET cell schematic (a) Conceptual (b) Transistor level implementation

Fig.4.3 shows the schematic diagram for a differential FerroFET memory cell. The

cell, apart from storage, provides the facility to compute 12-bit by 3-bit in-memory multiplications. Unlike previous work[94][95][96], the proposed bit-cell allows both positive and negative values for stored values as well as the inputs. During a read operation the WL is fully-turned on, appropriate V_{GS} values are provided through GL1 and GL2. The entire row is read simultaneously through the current that is accumulated on SL. The accumulated current corresponding to ΔG and ΔV is given by:

$$I_1 = -\Delta V.(G - \Delta G), I_2 = \Delta V.(G + \Delta G) \tag{4.5}$$

$$I = \Delta V.(-G + \Delta G + G + \Delta G) = \Delta V.(2\Delta G) \tag{4.6}$$

The weights of B-coefficients are encoded as multiples of $2\Delta G$ and the inputs or z-coefficients are coded as multiples of ΔV. Here, both the ΔG (B-coefficients) and ΔV (z-coefficients) can be positive or negative; or in other words no additional peripheral structure is required that is determined by the sign of the number being multiplied. The FerroFET based product evaluation has been done by implementing the full design through spice simulation.

This cell structure allows in-situ analog computation of multiply and accumulate (with both positive and negative operands) in the memory array itself.

4.2.2 Core Architecture

Fig. 4.4 shows the block-diagram for the entire core and provides the detail structure of the FerroFET memory array. Cores can be divided into three major blocks: (1) the FerroFET memory array that computes vector dot product (sum of products), (2) peripheral blocks, and (3) the communication block. The memory array and the peripheral blocks together form the compute unit. Each core has a maximum of 8 compute units corresponding to each neighbor. The details of the architecture and the sub-blocks are shown as a part of the

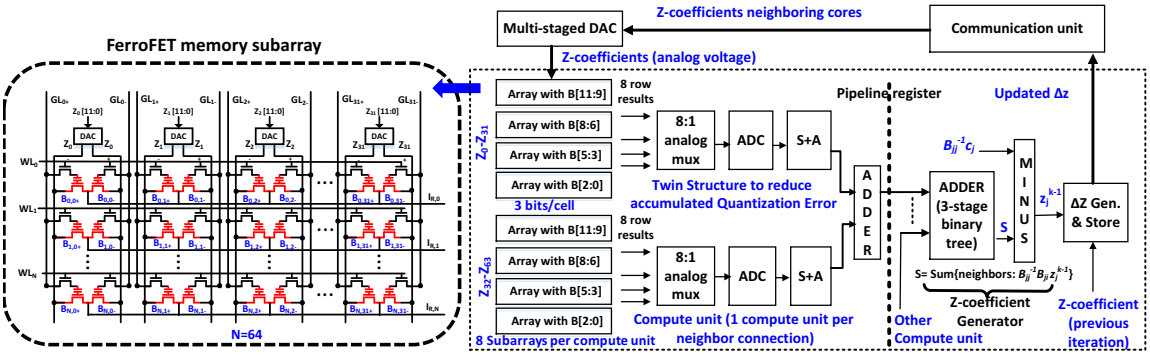


Figure 4.4: Schematic of a typical core.

supplementary material. Here we discuss the salient features only.

FerroFET memory array structure

The hierarchy of the FerroFET memory array has been shown in detail in the Fig. 4.4. In each iteration, the memory array performs matrix-vector product of B and z using a pseudo-crossbar architecture.

Peripheral blocks

The current summing FerroFET subarrays have per-column analog-to-digital converters (ADCs) to digitize the summation of the inner-products. The peripheral blocks include, shift plus add (S+A) arrays, adders to collect the output of each compute unit, followed by a subtraction block. Once these blocks finish their operation the z-coefficients are computed and sent to the communication blocks. Each core receives inputs from the neighboring cores. Digital to analog converters (DAC) produce voltage signals corresponding to digital value of z-coefficients and these voltages are asserted on bit-lines (BL1, BL2) of the memory array.

Communication unit

Communication between cores is done through an asynchronous mechanism. In this design, a 4-phase handshake protocol has been used because of reduced logical complexity

Table 4.1: Specifications of baseline Von Neumann architecture in 28nm CMOS process

Parameter	Value
Simulation Mode	Syscall Emulation
CPU Type	DerivO3CPU
CPU Width	3
L1 Inst. Cache Size	64kB
L1 Data Cache Size	64kB
L2 Cache Size	2MB
Main Memory	32GB DDR4

and competitive power and area efficiency when compared with respect to a 2-phase protocol. The details of the protocol has been discussed in the Summplementary material.

4.2.3 System Architecture

The proposed architecture comprises of 8 rows with 8 cores in each. The entire design is syntehsized in the 28nm CMOS process. To simulate and obtain latency and power estimations for the baseline Von-Neumann architecture, we used the gem5 simulator[97] and McPAT[98]. Table 4.1 shows the system specifications for the gem5 simulator. For each iteration of the baseline Von-Neumann architecture, we collect a set of workload statistics. The system configuration and the data for a single iteration are then run through McPAT to obtain power estimations.

Simultaneously, we construct an SRAM PIM to compare its performance with the proposed FerroFET based PIM architecture. In this design we use single read and write ports and peripheral adders and multipliers to design a compute unit. The structure of cores in the SRAM PIM are identical to that of the FerroFET PIM. The SRAM PIM prototype also consists of 64 cores.

4.3 Design Space Exploration

Fig. 4.5(a) illustrates how the average normalized error changes with respect to the number of iterations for a varying number of bits per FerroFET cell. The average normalized error is defined as the L2 norm of the difference of Z between the proposed architecture and a corresponding floating point architecture. In our design, we use 2/3/4/7 bits/cell to store 12 bits(excluding sign bit) of fixed point (6 bits for integer and 6 bits for the decimal). For example, the range corresponding to 2 bits with sign bit, i.e., [-4,3] is represented by 3bits/cell (due to the cell architecture). In our design the default ADC resolution is 16 bits; and we also study the effect of 16 bit data-converters on the design. We use the linear part of the FerroFET's conductance, as discussed above.

We observe that the average normalized error increases as the number of bits/cell increases as shown in Fig. 4.5(a). This is attributed to the fact that the use of a larger number of bits/cell requires higher ADC and DAC bit resolution to maintain precision. average normalized error from 7 bits/cell FerroFET array is much larger than 2,3,4bit/cell FerroFET array mainly due to the loss of precision during data-conversion. A higher resolution from the data-converters beyond 16b requires noise-shaping and advanced architectures that are not amenable for low-power designs.

In order to quantify the effect of the finite resolution of the ADC/DAC on the fidelity of the final results, we plot the average normalized error of Z in Fig. 4.5(b). Three cases corresponding to the ADC/DAC resolution of 12 bits, 14 bits and 16 bits are studied. Here the number of bits per FerroFET cell is considered to be 3. we observe that an ADC/DAC of 14 bit resolution results convergence, whereas the quantization offered resulting for a 12 bit ADC/DAC is unacceptable. This leads to the design point where 14 bit ADC/DACs are used in the peripherals.

So far, we have studied the effect of the peripheral circuits and storage architecture on the convergence of the optimization algorithm. FerroFETs, in spite of their multi-state

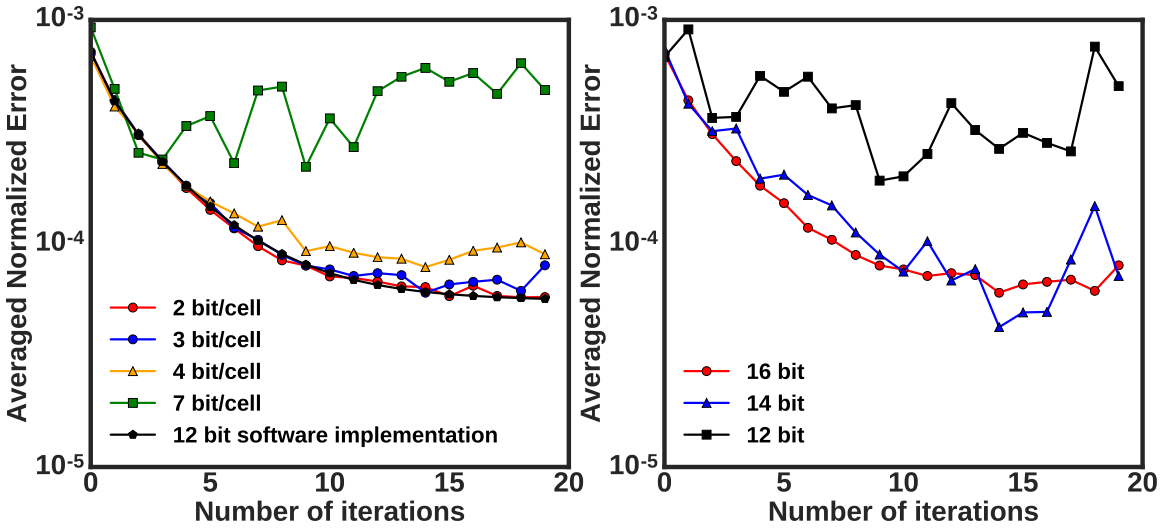


Figure 4.5: (a) Average normalized error in signal reconstruction via distributed least-squares method as a function of the number of bits/cell of FerroFET. The ADC bit resolution is fixed to 16 (b) Average normalized error of Z in non-uniform sampling algorithm with respect to different ADC bit resolution

storage capability, suffer from inherent non-linearities where the conductance does not change linearly with the number of pulses. We analyze the effect of this non-linearity in conductance on the average normalized error of Z in Fig. 4.6. The non-linearity in conductance of FerroFET is modeled as a normalized sigmoid function.

$$G(x) = \frac{\beta e^{\alpha x}}{1 + e^{\alpha x}} + G_{min}, \beta = G_{max} - G_{min} \tag{4.7}$$

where G_{max} and G_{min} are the maximum and minimum conductance values, α is an empirically derived parameter. This is in contrast to the convex/concave functions that have been used in [99][94][43] to model non-linearity. We note that in the case of FerroFETs the sigmoidal function is (1) a better fit and (2) physically meaningful. The sigmoidal conductance response manifests from the approximately Gaussian distribution of coercive fields among individual domains within the ferroelectric. Therefore, an amplitude modulated pulse scheme, which in essence, integrates across the domain distribution is expected to produce sigmoidal characteristics.

Fig. 4.6(a) shows the nonlinear conductance of FerroFET as a function of the number of

write pulses and (b) shows how non-linearity in conductance affects the average normalized error. In this design the number of bits per FerroFET cell is assumed to be 3. It is shown that if α is greater than 0.1, the average normalized error increases as the number of iterations progresses. This illustrates that the use of FerroFETs in optimizations for PIM architectures require linear changes in conductance during potentiation and depression. In [99], the authors have shown that when resistive processing units (RPUs) are used in cross-point architectures for solving inference in deep neural network architectures, the resistive units need high degrees of linearity. We arrive at a similar conclusion when such resistive elements are used in solving optimization problems. This motivates further research in the device community to address the issue of non-linearity when PIM architectures are used for solving linear-algebraic problems.

We study the effect of the effect of the design space on critical system parameters such as compute time, energy, power and area. The number of bits that can be stored in a FerroFET decides the FerroFET array size. Our baseline design uses a cell with 4 bits/cell. We also consider the case of 5 bits/cell where we need 64x256 memory cells (8 subarrays of 64x32 dimension) to store all the B-coefficients. As we decrease the number of bits/cell, the total number of memory cells required increases. For example, a design with 3 bits/cell requires a total memory size of 64x384 cells (12 subarrays of 64x32 cells per subarray), and so on.

Similarly, the DAC resolution also affects the compute unit area and other critical metrics. In this architecture, the multi-stage DAC resolution can be configured to 2, 3, 6 and 12 bits. The main role of the DAC is to provide analog values of the z-coefficients which are represented in a 12-bit fixed point format. As we reduce the DAC resolution, there are two options that can be pursued in the design: (1) duplicate the subarrays to compute in parallel and maintain the compute time at the expense of area overhead (2) perform the computations sequentially. The sequential computation can be explained by the following simple example. For a 6 bit DAC we first evaluate the sum with 6 LSB bits of all

the z-coefficients and in the next cycle we evaluate the sum with the 6 MSB bits for all z-coefficients and eventually add them with appropriate scales using shift+add blocks. We define the first approach as parallel-computation which results in higher throughput but lower area-efficiency and the second approach as sequential-computation which consumes lower area at the cost of lower throughput. Another important fact to note is that decreasing the number of bits/cell or the DAC resolution reduces the dynamic range of the read current out of SL lines resulting in simpler peripheral design. In our case studies, we have optimized the read peripheral circuits and ADCs based on the DAC configuration [100].

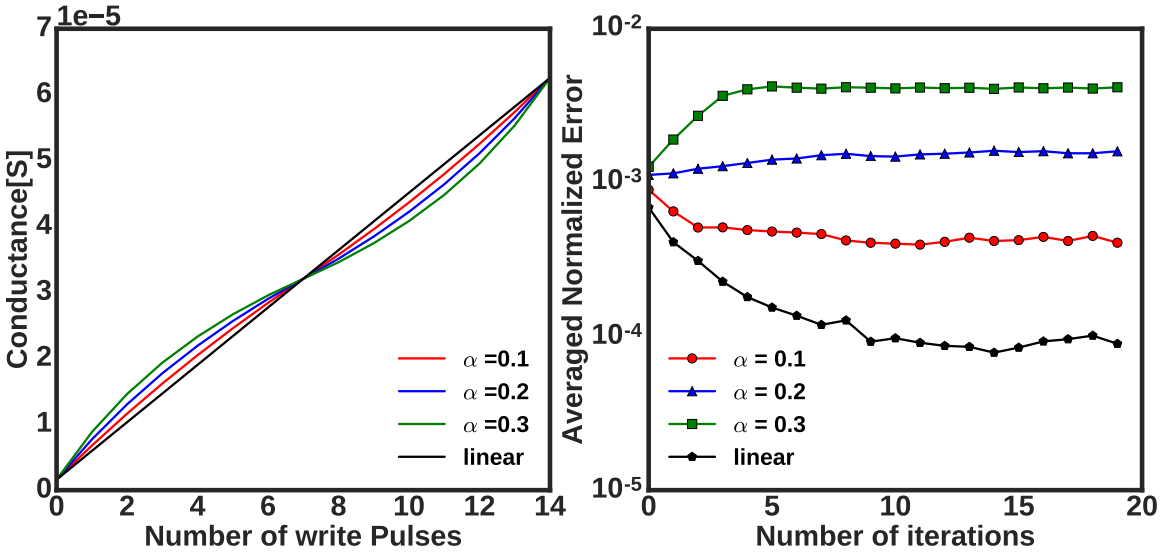


Figure 4.6: (a) Nonlinear conductance of 4bit/cell FerroFET (b) Average normalized error of as a function of the nonlinear conductance of FerroFETs. (4 bits/cell FerroFET and 16 bit ADCs are considered)

Fig. 4.7 and Fig. 4.8 illustrate the compute time and energy as the DAC resolution and number of bits/cell are varied for the parallel-computation and sequential-computation approach, respectively. It can be clearly seen from the two figures that in case of a sequential approach the computation time is 2-3X higher when compared to the parallel-computation approach. For parallel-computation (Fig. 4.7a-d), we observe a trend that the compute time goes up as the DAC resolution increases. This is because the ADC starts to dominate the system latency. As we increase the DAC resolution, to maintain the same quantization error for the read current a higher resolution ADC is required and ADC latency increases

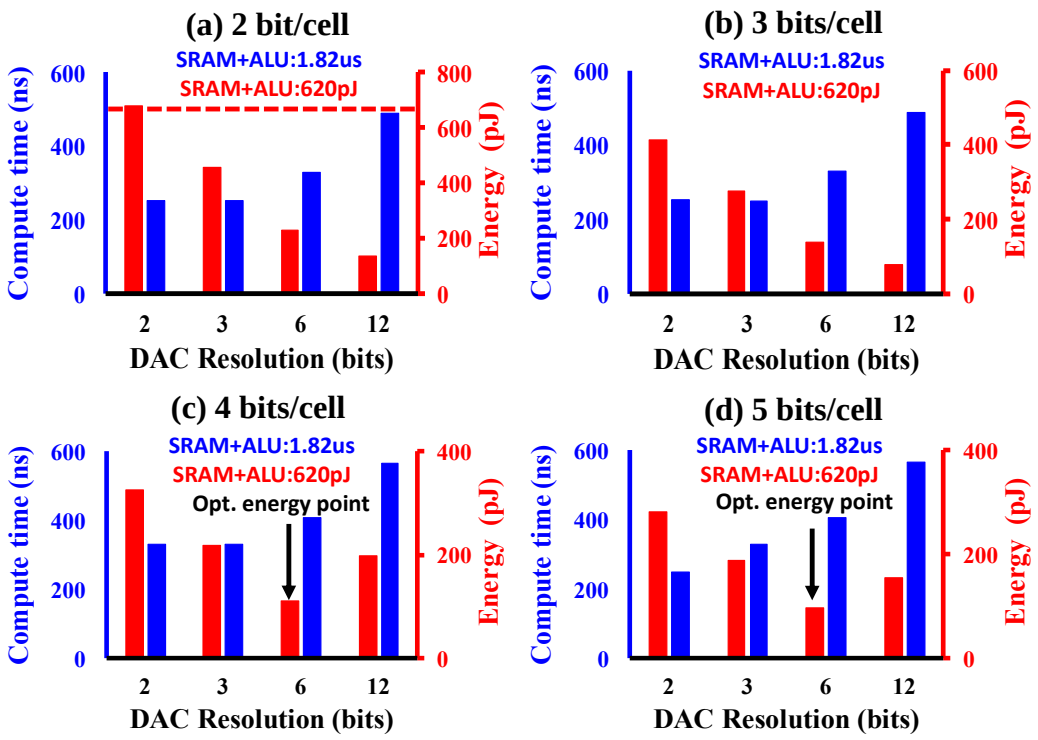


Figure 4.7: Compute time and energy behaviour of the compute unit versus DAC resolution for the parallel-computation approach and storage per FerroFET memory cell is (a) 2bit/cell (b) 3bits/cell (c) 4bits/cell and (d) 5bits/cell.

super-linearly as the resolution increases. In Fig. 4.7a and Fig. 4.7b, a monotonic decrease in energy is noted as the DAC resolution increases. This is because for both cases, the parallel memory-array and associated peripheral hardware overhead is the dominant factor, which decreases as the DAC resolution increases and eventually causes a reduction in the overall energy consumed. However, for Fig. 4.7c and Fig. 4.7d that have higher bits/cell (4 and 5 bits respectively) the ADC overhead starts to be significant. As mentioned before, as the DAC resolution for these two cases increase, we have to switch to a higher resolution ADC that adds to the energy consumed and off-sets the improvement due to reduction of the parallel subarrays and adders.

Fig. 4.8 exhibits an increasing trend of compute time as the DAC resolution and bits/cell decrease. With less bits/cell and DAC resolution, it results in multiple iterations of compute cycle since the number of sub-arrays are fixed. Due to the energy trade off between peripheral units and the ADC (discussed above), the trend for energy dissipation is similar

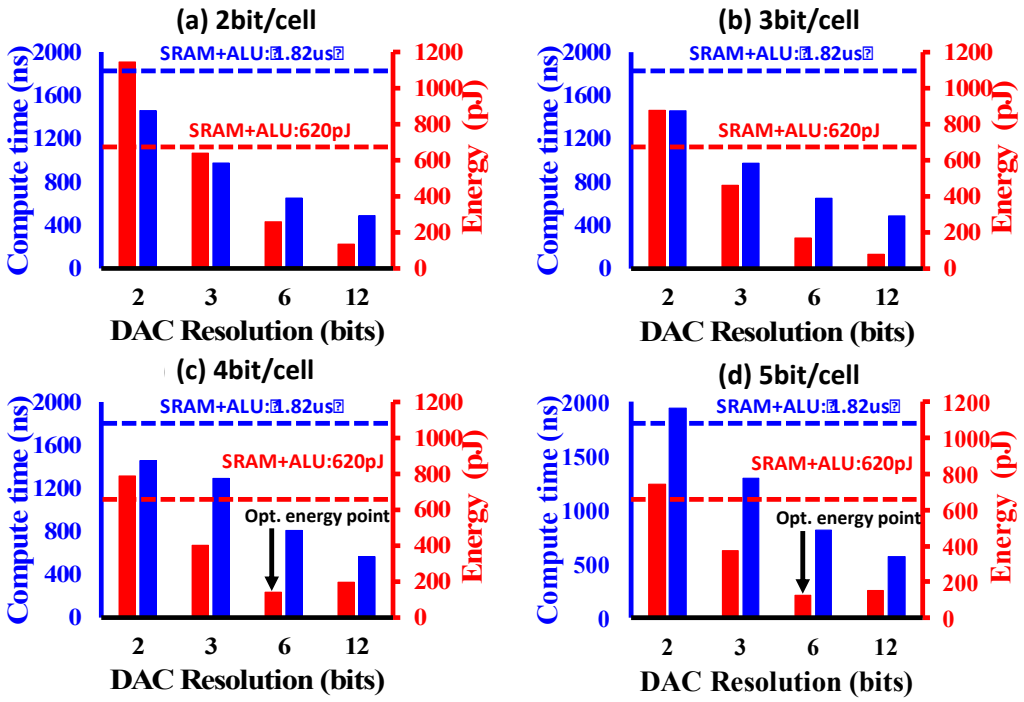


Figure 4.8: Compute time and energy behaviour of the compute unit versus DAC resolution for the sequential-computation approach and storage per FerroFET memory cell is (a) 2bit/cell (b) 3bits/cell (c) 4bits/cell and (d) 5bits/cell.

to Fig. 4.7. Also it can be noted that sequential approach consumes higher energy than the parallel approach due to the multiple iterations that are required. The comparison with an SRAM PIM structure has been shown using a dotted line in each of the histograms. The proposed design outperforms SRAM PIM structure in terms of compute time and energy for majority of design cases, as has been shown.

Fig. 4.9 shows the total power of the computation unit when the number of bits/cell and DAC resolution are varied for the parallel and sequential cases. From both Fig. 4.9(a) and Fig. 4.9(b) we observe that power consumption reduces as we increase either the number of bits/cell or the DAC resolution. From this we conclude that the total power consumed is determined by both the memory sub-arrays and peripheral logic. As the number of bits/cell or the DAC resolution increase, we observe a reduction in number of Shift+add array stages and memory subarrays, and this reduction causes an overall reduction in power. Further when Fig. 4.9(a) and Fig. 4.9(b) are compared to each other the parallel computa-

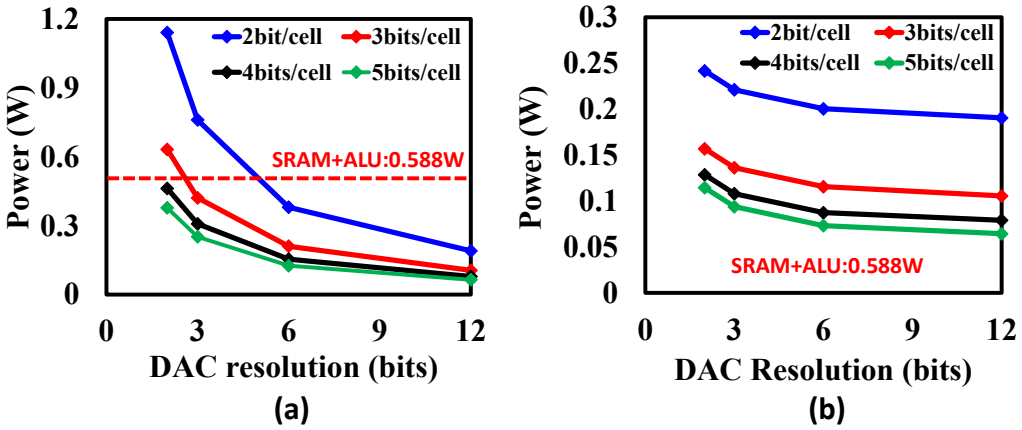


Figure 4.9: Power consumption of the compute unit when bits/memory cell and DAC resolution are varied for (a) parallel-computation (b) sequential-computation.

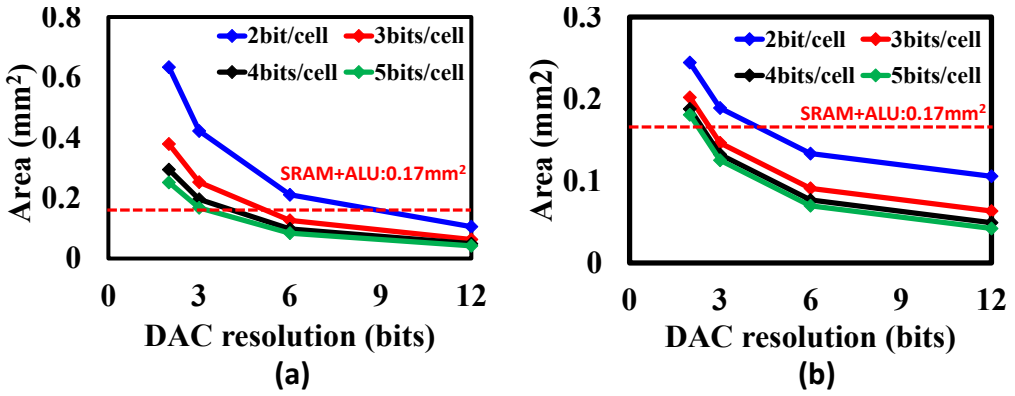


Figure 4.10: Estimated area of the compute unit when bits/memory cell and DAC resolution are varied for (a) parallel-computation (b) sequential-computation.

tion approach consumes higher power because of additional memory array and associated peripheral hardware requirements.

Fig. 4.10 shows the total area of the computation unit when the number of bits/cell and the DAC resolution are varied for the parallel and sequential cases. For the parallel computation approach (Fig. 4.10(a)), the area is larger than the sequential approach (Fig. 4.10(b)) since the computations are executed in parallel with a higher number of memory subarrays and peripheral blocks. As the DAC resolution and the number of bits/cell increase the total area increases because the memory subarray, shift+add and multi-stage adders required are lesser in number, and they dominate any increase caused by the ADC area. For all the figures the dotted lines show the performance of a corresponding SRAM+ALU Von-Neumann

architecture (baseline).

Table 4.2 presents the architectural results of compute time and energy for the baseline, SRAM PIM and FerroFET PIM architectures of 64 cores. FerroFET PIM shows 3x improvement in compute time and 21x improvements in energy efficiency compared to SRAM PIM.

Table 4.2: Compute time and energy comparison in different architectures

Architecture	**Baseline**	**SRAM PIM**	**FerroFET PIM**	**Performance**
Compute Time[s]	83μ	1.83μ	0.57μ	3x wrt SRAM PIM
Energy[J]	$1.36m$	460μ	21μ	21x wrt SRAM PIM

4.4 APPLICATIONS

As examples of prototypical problems that can be solved using the proposed algorithm and architecture, we present two applications. (1) Signal reconstruction from 1D EEG Signals and (2) Recovery of CT Images used in medical imaging.

Typical examples have been shown in Fig. 4.11(a) and (b). Both the Peak signal-to-noise ratio (PSNR) & Structural similarity (SSIM) are shown in Fig. 4.12. We note that increasing the sub-space dimension increases the fidelity of the reconstruction process. This justifies the use of a subspace dimension of 8×8 for the current applications in hand. It also shows the power of iterative algorithms in systolic PIM architectures for solving distributed convex optimization.

4.5 Summary

This chapter presents a systolic PIM architecture based on analog FerroFet pseudo-crosspoint arrays with in-situ computation to enable distributed convex optimization via least square minimization. Key contributions of the research are:

- A FerroFET based cell based which allows both positive and negative operands for matrix operations

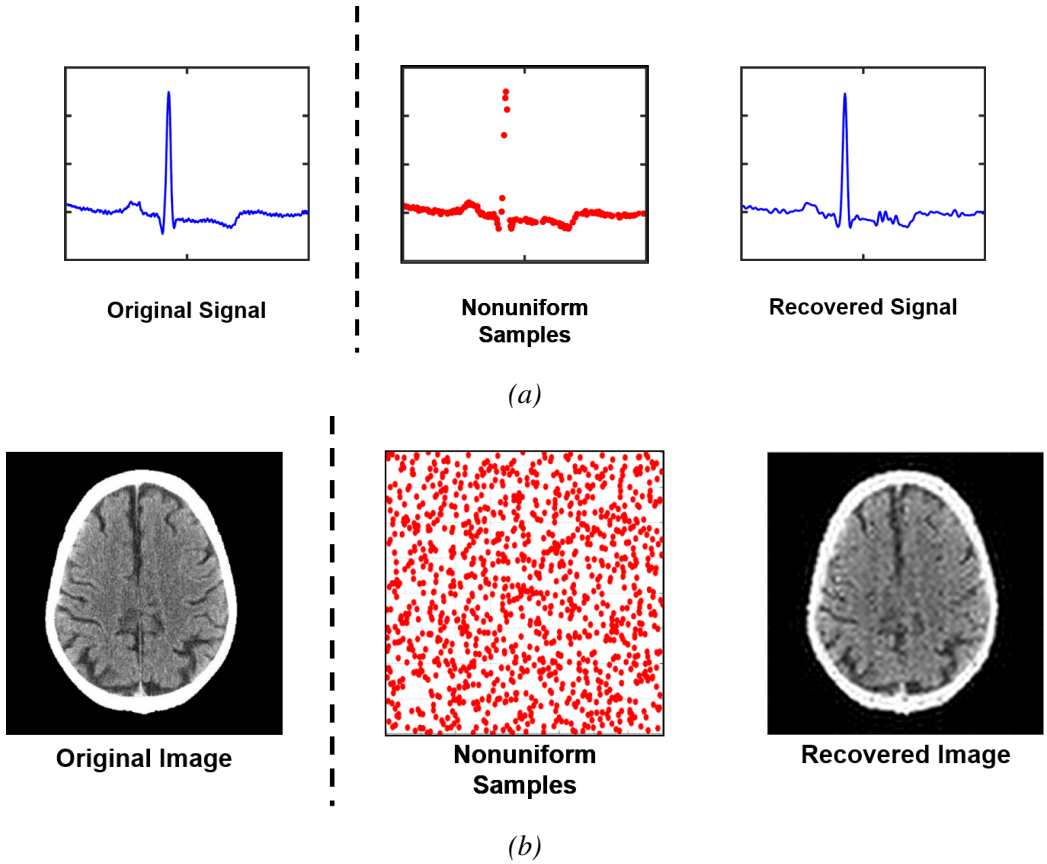


Figure 4.11: Reconstruction steps. (a) 1D Example: Recovery of EEG Signal Profile. (b) 2D Example: Brain Computed Topography Recovery.

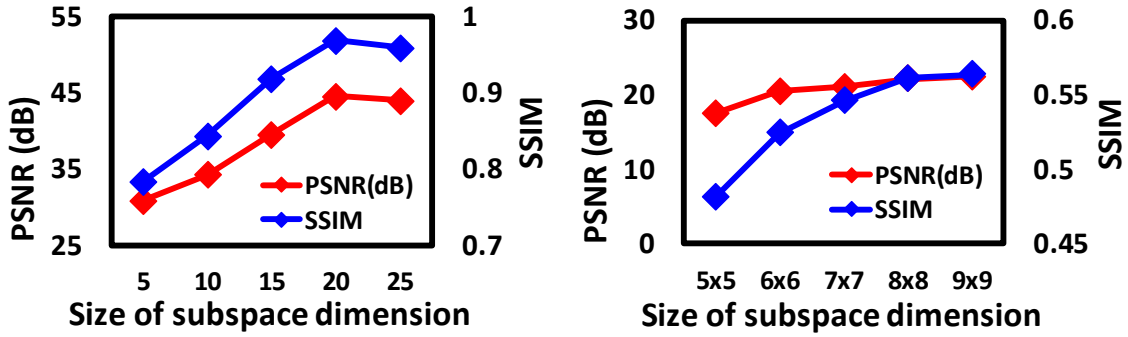


Figure 4.12: Peak signal-to-noise ratio (PSNR) & Structural similarity (SSIM). (a) 1D Example: Recovery of a non-uniformly sampled 1D signal from an EEG probe. (b) 2D Example: Recovery of a sampled image from the CT scan of a brain.

- A FerroFET based Processing-In-Memory architecture for solving a least squares minimization

- Develpment of a complete end-to-end tool chain and demonstration of $21\times$ in energy efficiency and $3\times$ in compute time compared to an SRAM based Processing- In-

Memory (PIM) architecture

We demonstrate that cross-bar resistive architectures are not only capable of accelerating machine-learning algorithms, but also distributed optmizations in a systolic array.

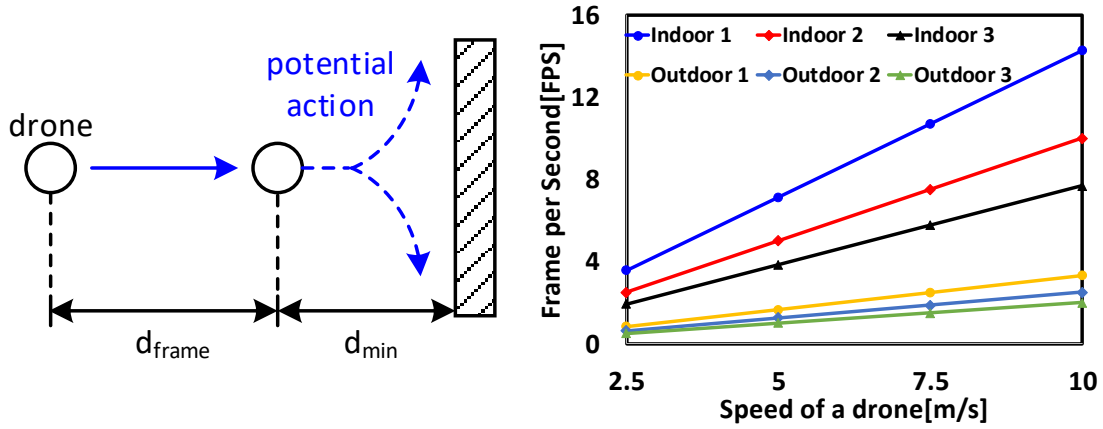
CHAPTER 5
STT-MRAM BASED SYSTEM FOR REINFORCEMENT LEARNING ON A DRONE

In this chapter, we propose a transfer learning (TL) followed by reinforcement learning (RL) algorithm mapped onto a hierarchical embedded memory system to meet the stringent power budgets of autonomous drones. The power reduction is achieved by 1. TL on meta-environments followed by online RL only on the last few layers of a deep convolutional neural network (CNN) instead of end-to-end (E2E) RL and 2. Mapping of the algorithm onto a memory hierarchy where the pre-trained weights of all the conv layers and the first few fully connected (FC) layers are stored in dense, low standby leakage Spin Transfer Torque (STT) RAM eNVM arrays and the weights of the last few FC layers are stored in the on-die SRAM. This memory hierarchy enables real-time RL as the drone explores unknown territories and the system only reads the weights from eNVM (that are slow and power hungry to write otherwise) for inference and uses the on-die SRAM for low latency training through both write and read of the weights of the last few layers. The proposed system is extensively simulated on a virtual environment and dissipates 83.5% lower energy per image frame as well as 79.4% lower latency as compared to E2E RL without any loss of accuracy. The speed of the drone is improved by a factor of 3X due to higher frame rates as well.

5.1 Introduction

Over the past decade, applications such as reconnaissance, surveying, rescuing and mapping with Unmanned Aerial Vehicles (UAVs) or drones have achieved substantial success. For all these applications of UAVs, navigating autonomously in varied environments with camera based inputs is considered a key enabling feature. Recently, reinforcement learning

(RL) on robotic tasks such as real-time drone navigation and collision avoidance has been extensively explored[101] [102]. However, online, real-time RL continues to be computationally challenging despite its recent success and its bio-mimetic approach. In typical RL systems, a deep convolutional neural network (CNN) is used to achieve a functional mapping images (system states) to the best possible action. In the case of RL for real-time collision avoidance, a major latency bottleneck arises from the need to train a CNN with the current image frame, which must be completed before the next image frame is captured [102][103].



(a) (b)

Environment	d_{min}[m]
Indoor 1	0.7
Indoor 2	1
Indoor 3	1.3
Outdoor 1	3
Outdoor 2	4
Outdoor 3	5

	Image frame per second[FPS]					
v_{drone}[m/s]	Indoor 1	Indoor 2	Indoor 3	Outdoor 1	Outdoor 2	Outdoor 3
2.5	3.571	2.5	1.923	0.833	0.625	0.5
5	7.142	5	3.846	1.666	1.25	1
7.5	10.71	7.5	5.769	2.5	1.875	1.5
10	14.28	10	7.692	3.333	2.5	2

(c)

Figure 5.1: (a) Definition of minimum distance required for obstacle avoidance (d_{min}). d_{frame} = distance that drone moves between frames. (b) Frame per second vs. speed of a drone for sample indoor and outdoor environments (c) d_{min} setting for different environment and minimum FPS needed for obstacle avoidance for different environments

This is illustrated in Fig. 5.1 where we show the relationship between the speed of a drone and the required frame per second (fps) of the image acquisition system. As shown in Fig. 5.1(a), For a given velocity of the drone, we can calculate the minimum fps require-

ment of the camera for collision avoidance based on the corresponding distance traveled between two frames (d_{frame}), and the minimum distance between the drone and its obstacles (a measure of clutter in the environment). From Fig. 5.1(b), we observe that the fps requirement increases as the speed of a drone increases. Since the minimum distance between a drone and its obstacles is lower in typical indoor environments compared to outdoor environments (i.e., the indoor environment is more cluttered than outdoor environments), drones in the indoor environment require higher fps compared to outdoor environments. As the fps increases, the time available to perform real-time RL decreases necessitating high-performance of the computing system. For small power-constrained drones, it requires significant hardware resources to execute the training process in RL within the latency and power targets. Further, embedded non-volatile memory (eNVM) [104][105][3] has emerged as a potential candidate for DRAM replacement for its high density and low standby power. This is particularly useful to store model weights of CNNs that can achieve RL in embedded systems, such as small drones. However, all eNVM technologies, including Spin Transfer Torque Magnetic RAM (STT-MRAM) exhibit high write latency and energy and does not meet the write energy and latency targets for real-time RL.

To address this fundamental challenge, we propose an algorithm-hardware co-design where we show:

1. Context-aware transfer-learning (TL) augmented with RL. Before deployment, an agent (drone) is trained in complex meta-training-environments (indoor and outdoor) in a virtual simulation platform. During training, the agent captures an image of an environment (called state) and a CNN provides the optimal action based on the current state to maximize some notion of long-term reward. Training the policy network is accomplished via RL. Once the system reaches the target performance, the trained weights of CNN in complex meta-environments are ready to be transferred to a drone (Transfer Learning) at the time of deployment.

2. At the time of deployment, the correct meta-model (indoor or outdoor model) obtained from TL is downloaded to the drone. In our studies, we consider a prototypical embedded platform consisting of a large, stacked-eNVM array and a smaller (30 MB) on-die SRAM. As a part of this study, we consider spin-transfer-torque (STT-RAM) as the NVM of choice. A part of the model (last few layers of the neural network) are stored in the on-die SRAM and rest of the model is stored in STT-MRAM stack.

3. After deployment, the drone performs real-time RL; the drone first reads the weights stored in NVM to perform inference to determine the best action (forward propagation of the CNN) based on the current state (acquired image). Once the drone receives the next state after the execution of inferred action, RL evaluates the error and train the weights in CNN. But instead of learning all the weights in every layer of the CNN, the system only trains the last few layers of CNN whose weights are stored in the on-die SRAM. This results in only read accesses from the e-NVM array during flight (inference/ forward propagation of data) and all the necessary write operations are executed on the on-die SRAM. In the process of inference (forward propagation of data), the system only reads the weights of the model from the eNVM to the SRAM. The weights of the last few layers stay in the SRAM and the updated weights of the last few layers are written to SRAM at the end of training. We also show a typical case where a small portion of the weights stored in the STT-MRAM array is updated in real-time. Since the convolution layers of the network stores the coarser features of the environment (obtained from TL), the proposed algorithm works successfully as the drone needs to learn only the environment specific finer features (online RL) in real-time. We show that the using TL followed by environment-specific RL over the last few layers achieves comparable accuracy as E2E RL. While E2E RL on an environment is not feasible with NVM based embedded platforms (in terms of latency and energy requirements), our proposed solution archives real-time operation with 79.4% (83.45%) decrease in latency (energy) in PE array compared to a baseline E2E RL system. Due to the stringent power constraints of a drone, the system employs STT-MRAM instead

of DRAM because using STT-MRAM can save the amount of energy used for refresh operation from DRAM since refresh operation is not required in STT-MRAM. With 83MB of weights stored in STT-MRAM, dissipated energy over 1000 iterations of STT-MRAM presents 58% decrease compared to the energy dissipated from DRAM in the case of on-line training of last 4 layers.

5.2 Reinforcement Learning for Drone Navigation

5.2.1 Basics of Reinforcement Learning

Before going into the details of the platform architecture, let us briefly review RL in the context of autonomous flight in small form-factor drones. Reinforcement learning (RL), inspired by behavioral psychology, learns by interacting with the environment in discrete time steps [101][106][107][108]. As opposed to supervised learning, RL doesn't have direct access to the data labels. The labels for RL can be thought of as dynamic and are generated and updated online until convergence is achieved. The agent is placed in the training environment and is allowed to take actions to explore the environment. With every action taken, the agent is presented with a reward based on a user defined goal. The reward quantifies the underlying goal; if the agent took an action that was in accordance with the goal, the reward would be higher and vice versa. The objective of RL is to learn a control policy that predicts actions maximizing these long-term rewards. For the case of autonomous flight, the RL problem will be formulated as follows. The goal is to avoid crashing into the obstacles, hence the notion of distance between the drone and the nearest obstacle can be used as a reward. A set of feasible actions is defined for the action space (in our case moving forward, moving left and moving right). The agent is only allowed to select among these set of actions. Resized RGB images from the drone's camera are used as states. Once the goal, state and action space are defined, the agent is placed in the training environment. At time step t, the drone observes the current state s_t, takes an action at from the action space and moves to a new position and observes a new state s_{t+1}. These

current and new state pair along with the actions taken are used to generate a reward r_t (s_t,a_t, s_{t+1}). For each step, these four quantities together define an RL data-tuple (s_t,a_t,s_{t+1},r_t). The objective of RL is to predict set of subsequent actions, leading to the maximization of the long-term discounted return

$$R_{\mathrm{t}} = \sum_{i=t}^{T} \gamma^{\mathrm{i\text{-}t}} r_{\mathrm{i}}$$

where, γ is the discount factor). This is done by converting the data-tuples into sets of training pairs. The effectiveness of taking an action a_t from a given state s_t is quantified by its corresponding Q-value Q(s_t,a_t). The greater the Q-value, the more favorable the action is. These Q values are updated online using the Bellman Equation

$$Q(s_{\mathrm{t}}, a_{\mathrm{t}}) = r + \gamma max_{\mathrm{a}} Q(s_{\mathrm{t+1}}, a) \tag{5.1}$$

The training pairs (s_t,Q(s_t,a_t)) are then used as the input-output pairs for training the network. At any given state, the network predicts the action with the maximum Q value $a' = max_{\mathrm{a}} Q(s_{\mathrm{t}}, a)$. RL for obstacle avoidance and path-planning has been successfully applied in prototypical robotic vehicles [109][110] and in Parrot AR drones [111] and interested readers are pointed to [110] for a detailed overview.

5.2.2 RL in Camera Based Navigation in Drones

We focus on the implementation of a camera based drone system that performs end-to-end navigation via collision avoidance (long term goal) as shown in Fig.5.2. The navigation problem is mapped to the RL problem as follows. The state at time instant t, $s_t \in S$ is the image frame of the environment from the camera. At any given state, the drone takes any action $a_t \in A$ where A is the action space. In this proposed system, the action space is limited to have five values A =0,1,2,3,4. 0 in action space A indicates that the drone moves forward, 1 and 3 mean that the drone turns left with turn angles 25 degree and 55 degree respectively. Similarly, 2 and 4 means turning right with turn angles 25 degree and 55

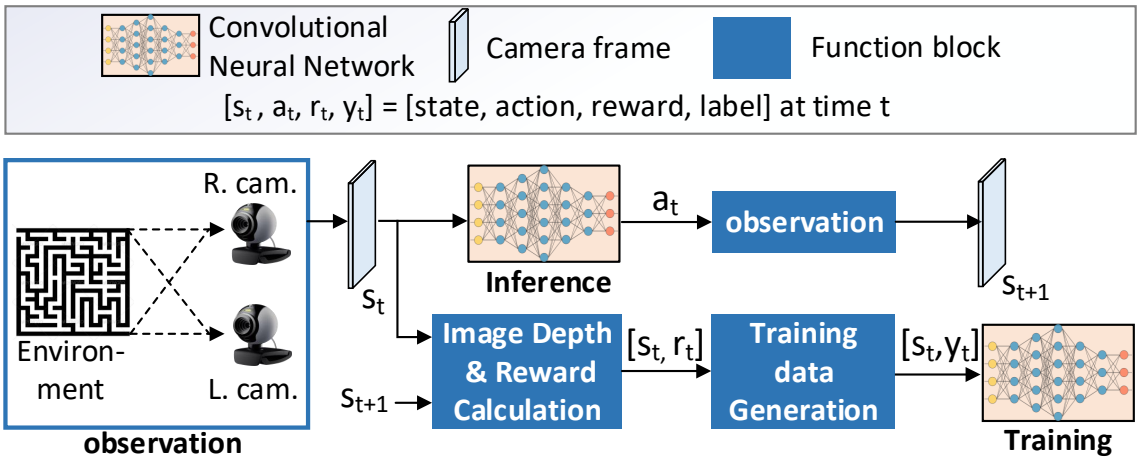
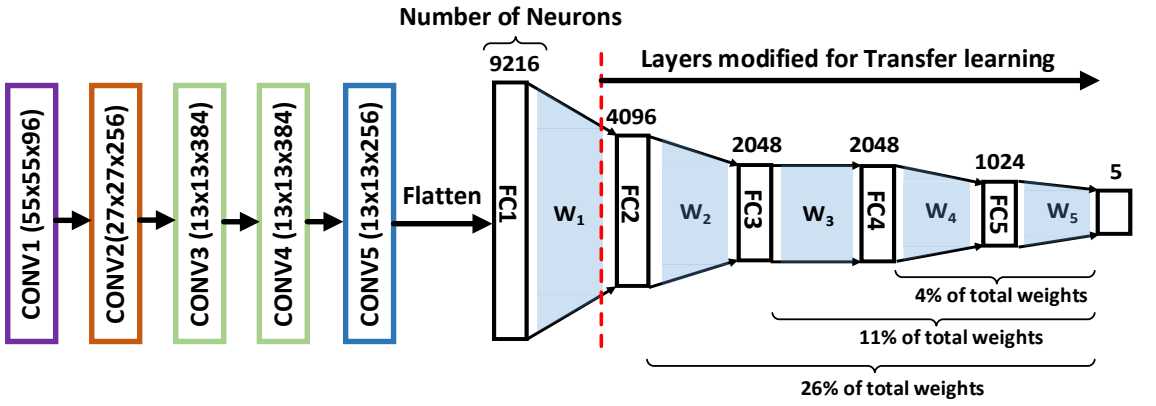


Figure 5.2: Reinforcement Learning(RL) network architecture for camera based navigation in drones

degree. These five actions are sufficient for the drone to navigate in its surrounding. When the image frame is captured from the stereo camera, a disparity map is used to generate an approximate depth map of the image frame [102]. From the generated depth map, a reward is generated in a manner described in [103]. In the process of reward generation, the depth map is segmented into smaller window at the center and the average depth of this center window correlates to the value of reward. Therefore, the reward becomes smaller when the drone is closer to obstacles because the average depth in the center window is less. The Q values for the states are estimated using a deep convolutional network (CNN). The image frame obtained from the camera is the state at time t, $s_t \in R^{nxn} wheren = 224$ and becomes an input to the CNN.

In order to have the network architecture optimized for autonomous navigation, we modified the AlexNet model [112] and used it as the CNN. It consists of 5 convolutional layers and 5 fully connected layers. The detailed network architecture and parameters are shown in Fig.5.3. During the online RL when the drone is flying, the CNN learns the weights of the model and keeps on improving the functional mapping from the state to the action.



Filter = 11x11x96, followed by ReLU, norm, Maxpool

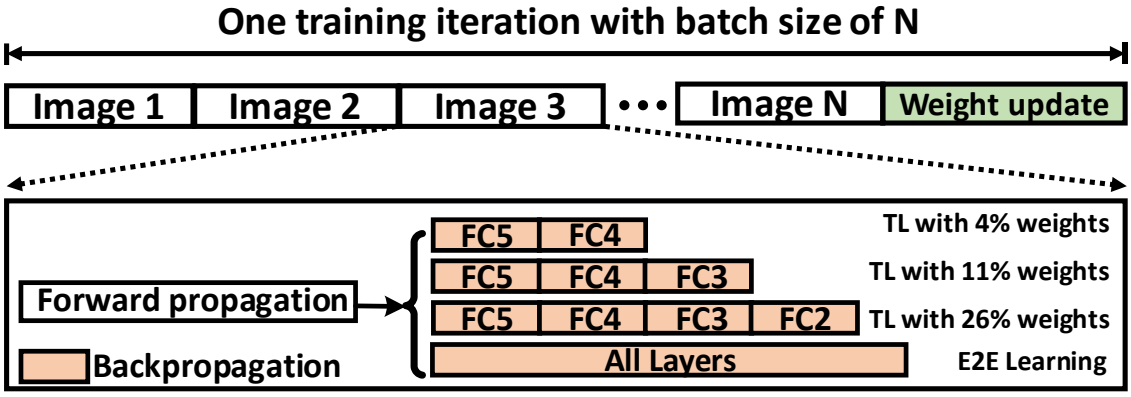
Filter = 5x5x256, followed by ReLU, norm, Maxpool

Filter = 3x3x384, followed by ReLU

Filter = 3x3x256, followed by ReLU, Maxpool

Layers	# neurons	# weights	% total weights	% cumulative weights
FC1	9216	37752832	67.18	93.33
FC2	4096	8390656	14.93	26.14
FC3	2048	4196352	7.468	11.21
FC4	2048	2098176	3.734	3.743
FC5	1024	5125	0.009	0.009
output	5			
sum	18437	52443141		

(a)



(b)

Figure 5.3: Reinforcement Learning(RL) network architecture for camera based navigation in drones. (a) Modified AlexNET [112] for the proposed system (b) 3 configurations where 4,11 and 26% weights are learnt in real-time. This is in contrast to E2E RL, where the entire network is learnt in real-time.

5.2.3 Challenges of End-to-End(E2E) RL in Embedded Systems

In a true biologically-inspired system, an autonomous drone should learn to navigate via E2E RL, where the reinforcement learning algorithm trains the weights in every layers of the CNN [103]. From starting with randomly initialized weights of the model, the drone

should learn the model that efficiently maps state to action from iterative interactions with the environment. Although feasible [103], this faces two fundamental challenges:

1. During exploration, the drone will take random actions and they are often incorrect actions, especially at the beginning of the flight. This can lead to collisions with obstacles. The sequence of incorrect action that lead to collisions can cause damage to the drone or the environment.

2. Further, E2E RL is computationally extremely challenging. Since E2E RL requires training of all weight parameters in every layers of CNN, it is almost impossible to achieve autonomy via RL in small form factor drones, without additional off-board infrastructure [103]. Improvements in both design [112][2] as well as technology [7][11][5][14][113][12] continue to make CNNs a reality on resource constrained edge-devices. In particular, eNVM is a promising replacement for DRAM to achieve high energy-efficiency. Among competing eNVM technologies, such as RRAM [11][12], PCRAM [14], FerroFETs [113], STT-MRAM [5][4] is considered more mature and exhibits high density, endurance and nano-second read speeds. However, the write latency and energy of STT-MRAM is expensive and is a major bottleneck in real-time RL and continuous weight updates.

5.3 Proposed Approach Using Transfer Learning(TL) with Real-Time RL

In transfer learning, pre-existent knowledge of the source tasks from one or more domain is used to learn target task in another domain. Transfer learning approach to solve various problems in deep learning has been there for over a decade. It has been used in the past for the purpose of mitigating convergence issue, faster convergence, improving target performance, reducing the time of convergence and addressing the issue insufficient data [114][115][116], where the weights of the deep network learnt for one problem is used as initial weights for some another similar problem. The network is then fine-tuned, end-to-end on the new data set converging faster. It is a well understood fact that [117], for a complex enough task, deep network's performance increases by increasing the num-

ber of hidden layers (given the amount of training data scales too). So, for an acceptable performance, the network should be deep enough, which comes with additional computational cost. This increased computational cost requires heavy computational resources (like GPUs) and cannot be executed on a resource constrained system/edge node (say a drone). To the best of our knowledge all the TL papers in the past discuss TL as tool/approach to address the above-mentioned issues without worrying much about the computational cost required to train a deep neural network. In this paper we show we can use Transfer learning, to segment a deep network into trainable and non-trainable part reducing the training computations, for underlying task without compromising too much on its performance. We use transfer learning with real-time RL as an algorithmic solution that maps to a hierarchical memory system consisting of stacked STT-MRAM and on-die SRAM. This alleviates the challenges of E2E RL and enables a practical hardware solution to realize autonomous flight with environment specific RL. In our proposed system the agent learns on an embedded platform in the following steps:

1. The CNN is first trained in complex meta-environments in simulation. Once the training is finished, the pre-trained CNN model is downloaded to the system memory as a meta model. We explore two types of meta-environments: outdoor and indoor. Other types of environments can be added depending on the types of real environments that drone is expected to be deployed in. 2. The downloaded meta model is located in STT-RAM and the weights of the last few layers of the CNN are transferred to an on-die SRAM. During real-time learning, the system reads the weights of each layer of STT-MRAM to SRAM for inference and once inference is finished, we train the weights of the last few fully connected (FC) layers of the model and write the updated weights back to the SRAM. By performing inference with the weights from TL and training the last few fully connected layers of the network via RL, we can reduce the latency and energy of the system significantly. This extends the drone's battery life and enables the system to support a higher speed as illustrated

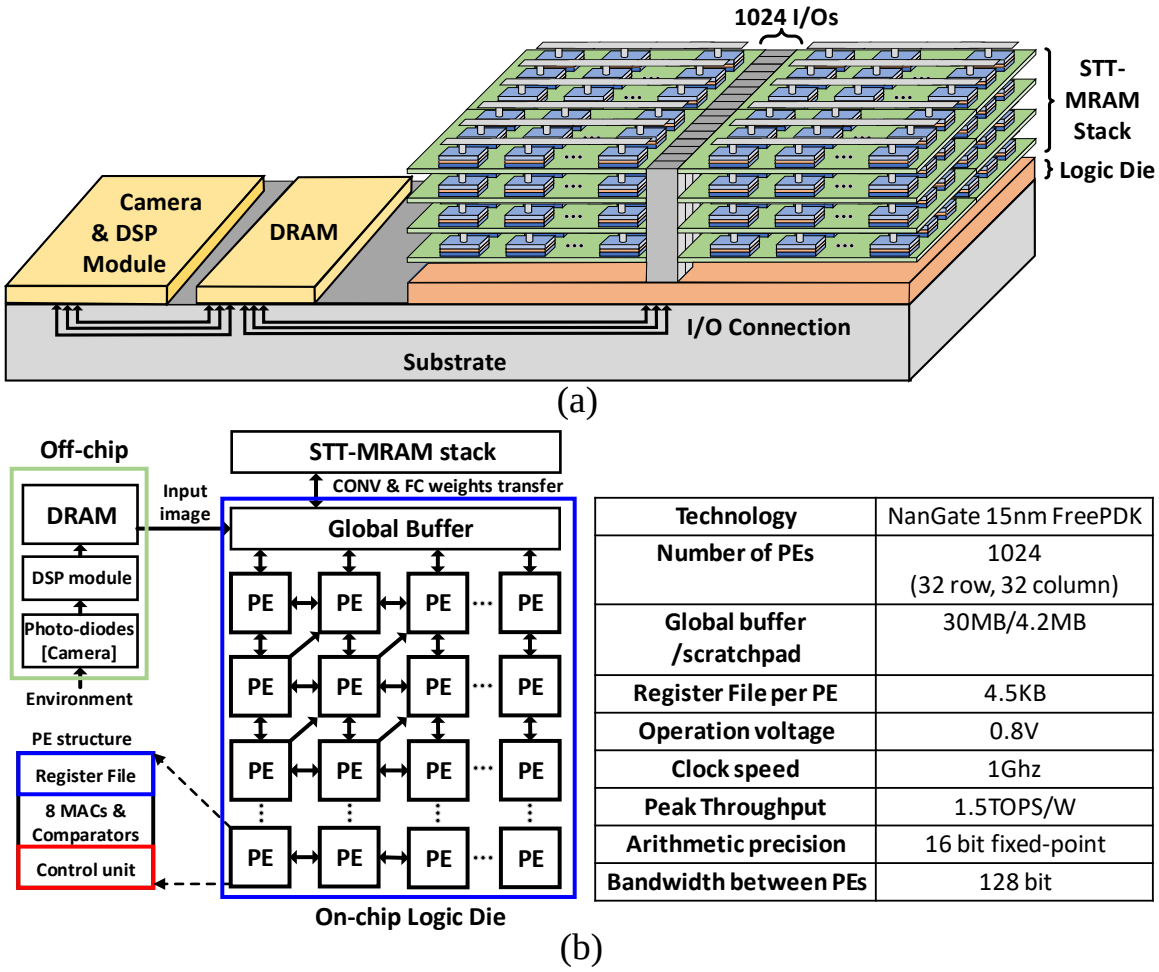
in Fig. 5.1. Fig. 5.3(a) presents three different architectures for training. Based on the on-die SRAM capacity, we can store 26% (FC2+FC3+FC4+FC5), 11% (FC3+FC4+FC5) and 4% (FC4+FC5) of the total weights of the network in the SRAM. The procedure of on-line training is described in Fig. 5.3(b). In order to complete one training iteration with batch size of N images, the system performs N number of computation, which is defined as taking one image at a time and complete forward and backward propagation. In the following sections, we compare the system performance of TL followed by RL, which train the last 2/3/4 layers of the network, and E2E learning (baseline), the algorithm that trains all parameters in the network.

5.4 Proposed System Architecture

The system architecture includes a logic die that contains of a systolic array of processing elements [112] and a global buffer (on-die SRAM) and a STT-MRAM stack on top of the logic die (Fig. 5.4). The architecture of the sub-array organization and local/global IO of STT-MRAM stack is same as the DRAM-based High Bandwidth Memory (HBM) architecture from JEDEC [118]. The DRAM subarrays of DRAM-based HBM have been replaced with STT-MRAM. By using STT-MRAM in the DRAM-based HBM architecture, we provide a realistic and emerging platform for an embedded system with high-bandwidth IO, based on HBM JEDEC specification [118]. A system with camera, image processing DSP module and DRAM buffer memory is integrated on a substrate (which can be a silicon interposer or a package substrate) as shown in Fig. 5.4(a). The connections between each module and the logic die are assumed to be DDR6 links.

5.4.1 Off-chip to On-chip Data Movement

The camera with a DSP module and buffer-DRAM are located off-chip on a shared substrate. Once an image is captured by the camera, the DSP module resizes the image to 224 by 224 as described above and stores the output to a DRAM memory buffer. The image is



Technology	NanGate 15nm FreePDK
Number of PEs	1024 (32 row, 32 column)
Global buffer /scratchpad	30MB/4.2MB
Register File per PE	4.5KB
Operation voltage	0.8V
Clock speed	1Ghz
Peak Throughput	1.5TOPS/W
Arithmetic precision	16 bit fixed-point
Bandwidth between PEs	128 bit

Figure 5.4: (a) 3D view of the hardware platform (b) System architecture and parameters as extracted post-synthesis in 15nm Nangate PDK.

serially read from DRAM buffer to the logic die as input to the CNN and stored in the on-chip global buffer. During the inference process, the image from global buffer is distributed to the register files in the PE array.

5.4.2 On-chip System Architecture with Stacked STT-MRAM

The logic die that contains of the spatial PE array and a global buffer located on a common substrate [118] and 3D-STT-MRAM [105][3] is stacked on top of the logic die in the same way as DRAM-based HBM is currently stacked. STT-MRAM stack is used as a weight storage and it contains all weights from each layers of the network. The systolic array of PE has 1024 PEs in total (32 rows, 32 columns) and the bit width of the connections

between PEs is 128 bit. One a PE is connected with 5 nearby PEs (top, bottom, left, right and upper right) [112][2]. The bit width of the connections between the global buffer and the 32 PEs at the first row of the PE array is 4096 and the global buffer can broadcast the same data to each PEs in the first row. STT-MRAM stack has 1024 I/O connections (each I/O has 2Gbit/s of bandwidth) with the global buffer [118]. Each PE has a register file, 8 MACs for convolution and vector-matrix multiplication and 8 comparators for rectified linear and maxpool operations. Fig. 5.4(b) shows a complete list of system parameters. The whole system is designed, synthesized and in the 15nm nangate technology [119]. All results discussed here are post-synthesis.

5.4.3 Mapping the CNN Model to the Memory System

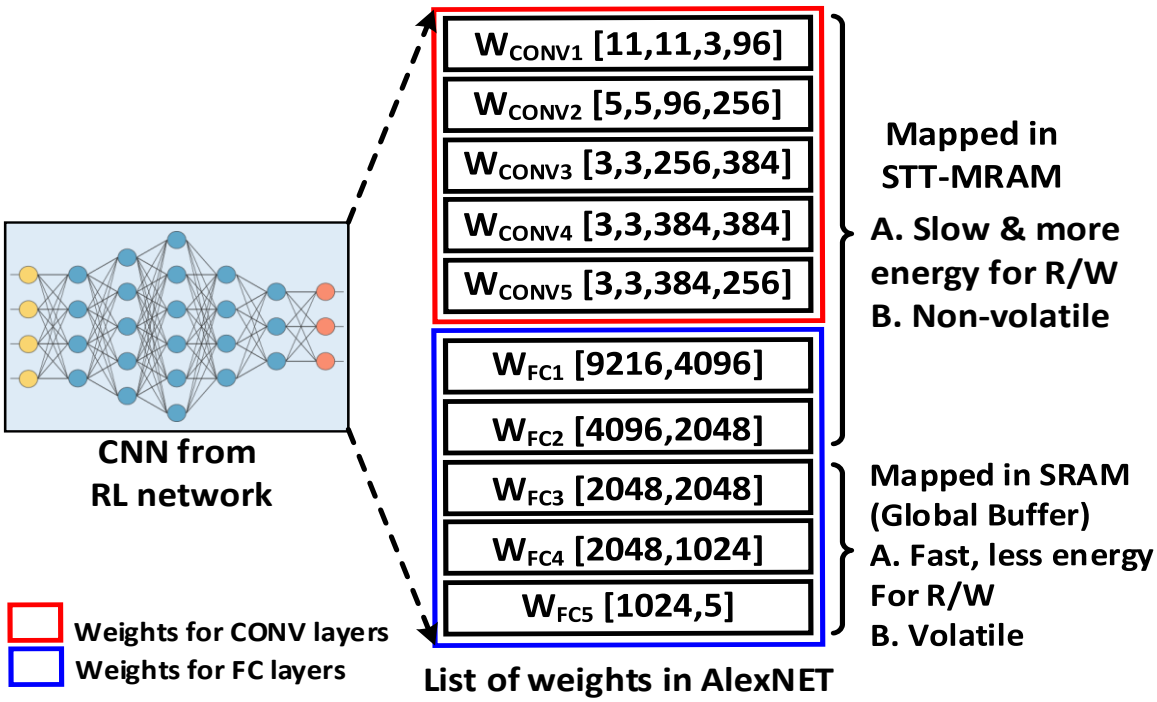


Figure 5.5: Mapping the weights of the proposed CNN (modified AlexNET) to stacked-STT-MRAM and on-die SRAM in the system

Fig. 5.5 presents how the model weight of the CNN is mapped to the memory system comprising of the stacked-STT-MRAM and on-die SRAM. The size of on-chip SRAM-based global buffer must be large enough to store the weights of the last 2/3/4 fully con-

nected layers of the network since the system performs real-time update of these weight parameters inside the global buffer. Since each parameter is 16 bit fixed point, the size of the SRAM should be 29.38MB if we store all weights from the last 4 fully connected layers. In the proposed design, we store the weights from the last three layers to the global buffer and the sum of all weights of the last three layer is 12.6MB. The rest of the weights from all convolutional (CONV) layers and the first and the second fully connected layers (FC1, FC2) add up to 100MB and they reside in the STT-MRAM array. In addition to this, the weight and bias gradients of the last 3 layers of the network are stored in the global buffer for the weight update in RL. Once we have the sum of gradients of weights and bias after processing a batch size of N, we need to update the weights as shown in a manner shown in Fig. 5.3(b) and this requires an additional 12.6MB of global buffer. In summary, the global buffer uses 25.2 MB of space to store weights of the last three layers for forward propagation and the sum of the weights and bias gradients from the last three layers used during backpropagation. Lastly, scratchpad for loading/storing intermediate results, input and weight parameters to the PE array takes 4.2MB of space in the global buffer. In summary, we need on-chip SRAM size to be 29.4MB, which is at-par with the on-die SRAM capacity of practical embedded systems.

5.5 Forward Propagation Through the CNN

5.5.1 Forward Propagation in Convolution (CONV) layers

A row stationary dataflow architecture is used in the systolic array for convolution during forward propagation [120]. The basic steps are:

1. Input image to the convolution layer is loaded from the global buffer to the local register file (RF) in each PE. Once the input image is stored in the RF of each PE, the row of the image is transferred to the nearby PEs by using diagonal connection to maximize data reuse within the PE array.

2. Each row of filter weights is broadcasted from the global buffer to the RF in each PE in

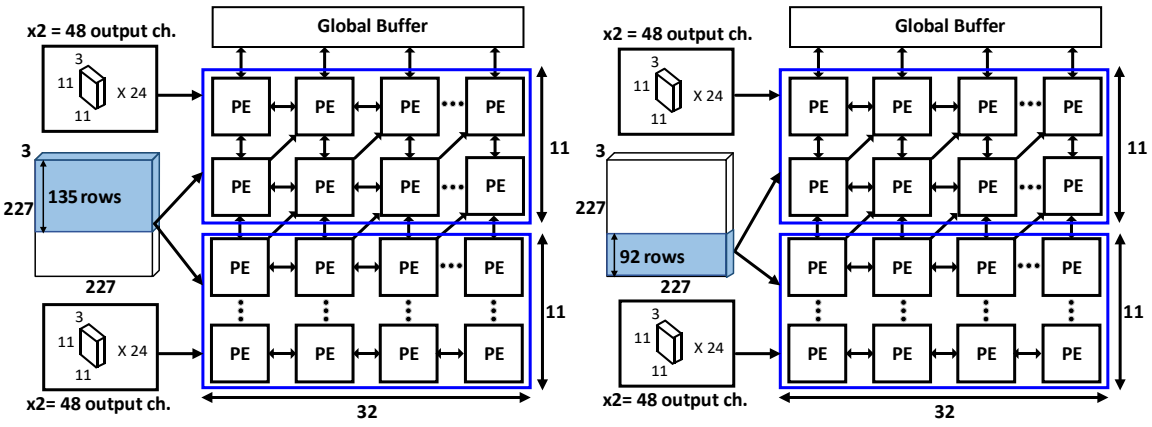
the same row of the PE array.

3. MAC units inside each PE computes row-wise convolution of image row and filter row and the result of the convolution (pSUM) is stored in the RF.

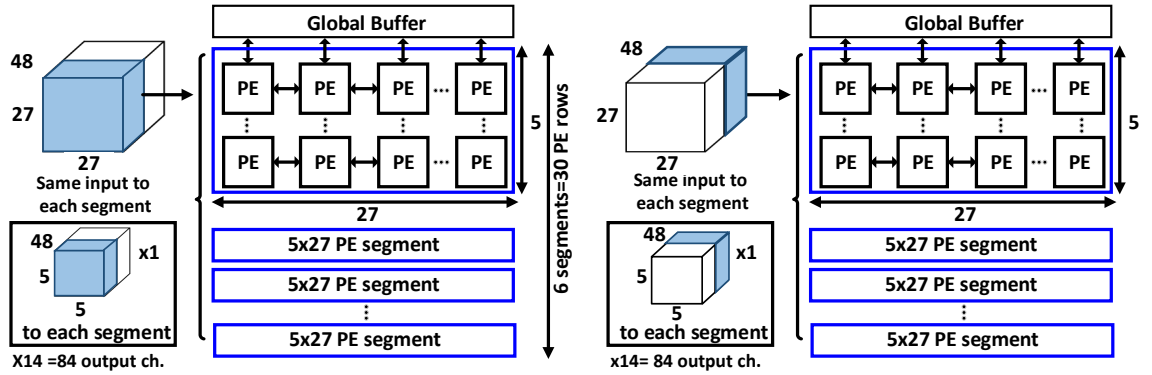
4. pSUM from each PEs in the same column are accumulated vertically to the PE in the first row and the accumulated values from the first row of the PE array are written back to the global buffer

In order to effectively utilize the hardware resource for computing convolution, we have three ways of partitioning PE arrays into segments based on the height of the filter in CONV layers.

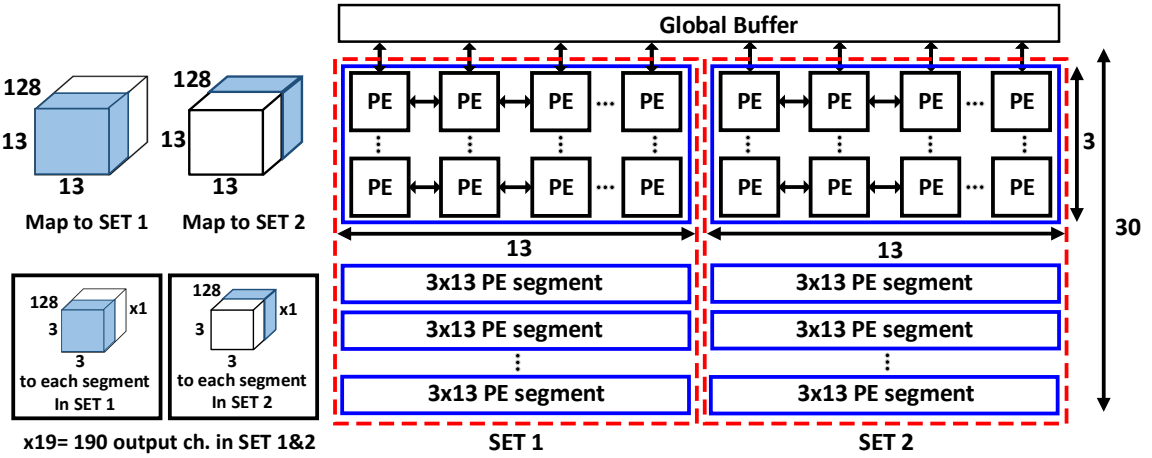
Based on the partitioning, the data mapping of the filter weights and the input are determined. The major factors that determine the partitioning the PE array are the size of RF inside the PE, the dimension of PE array and the filter size of the CONV layers. Fig. 5.6 shows all types of partitioning of PE arrays and the corresponding data mapping techniques. Fig. 5.6a shows how Type I partition is applied to the first convolution (CONV1) layer, whose filter dimension is (11,11,3,96) and stride is 4. In CONV1, each row of filter and image data with all input channels can fit into the RF of each PE in the same row. The PE array is partitioned into two segments whose dimensions are 32 x 11. Since the height of the filter is the same as the height of the segment, each row of the filter is mapped to each row of the PEs in the segment. The same image data is loaded to two segments of the PEs and filters with 24 different output channels are mapped to each segment. Depending on the RF size, the number of output channels of the filters can vary. The number of columns inside the segments is equal to the number of rows of images that the system can convolve per cycle. For example, TYPE I configuration produces the convolution results of 135 rows of input image in a single cycle. (135 = 32*stride + filter height) because the number of columns in the segment is 32. Fig. 5.6b presents the TYPE II mapping scheme of data for the second convolution layer (CONV2). In this case, the RF in a PE cannot fit the row of the image and the filter with all input channels because the data size is too large.



(a) Type I mapping used in CONV1



(b) Type II mapping used in CONV2



(c) Type III mapping used in CONV3,CONV4, CONV5

Figure 5.6: Strategies for mapping weights and data for processing the convolutional layers

Therefore, TYPE II divides input channels of filter and images into two parts and loads them into segments of the PE array. Since the filter height of CONV2 is 5, the dimension of each segment is 27 x 5 and the PE array is partitioned into 6 segments. Instead of using

all 32 columns of PE, 27 columns are utilized because each column generates one row of convolution output. The same image data is mapped to all 6 segments and each segment is mapped with the corresponding filters and each segment generates distinct outputs at the end of computation. Fig. 5.6c presents the TYPE III mapping scheme of data for CONV3. The main difference between TYPE II and TYPE III mapping is the existence of set, which is defined as a cluster of PE segments. Since the filter width and height decreases from CONV2 to CONV3, we can map 2 sets of 10 segments (each segment dimension is 3 x 10 PE) to PE array for CONV3. In the TYPE III mapping scheme, the segment size of the PE is 3 x 13 because the filter dimension is (3,3) and the stride is 1. Because the dimension of the segments is lower, we partition the PE array into 2 sets of 10 segments (total 30x26 PE array). Due to the high number of input channels of input and filter to CONV3, we split the input channel of filter and inputs into two parts. Unlike TYPE II, the two parts of inputs and filters are mapped to each set of the PE array, which enables us to map the input and the filter with all the input channels. After completing pSUM in step 4, the convolution results in the first row of set 2 must be transferred to the first row of set 1. For example, the output from PE at 14th column (PE in the 1st column in set 2) must be transferred to the PE in the 1st column in set 1. Then the two results from set 1 and set 2 are added together to complete the convolution. Since the filter height and width (3,3) in CONV4 and 5 are the same as the filter height and width in CONV3, the TYPE III mapping scheme is used for CONV4 and 5 as well.

5.5.2 Forward Propagation in Fully Connected(FC) Layers

Vector-matrix multiplication is the core computation in the forward propagation of Fully Connected layers. Fig. 5.7 describes how the input vector and the weight matrix are mapped to each PE in the array to perform vector-matrix multiplication. Once the values of the weight matrix are loaded to the PE array, the input vector is loaded to the first column of the PE array. Then the values in the input vector are propagated row-wise in the PE

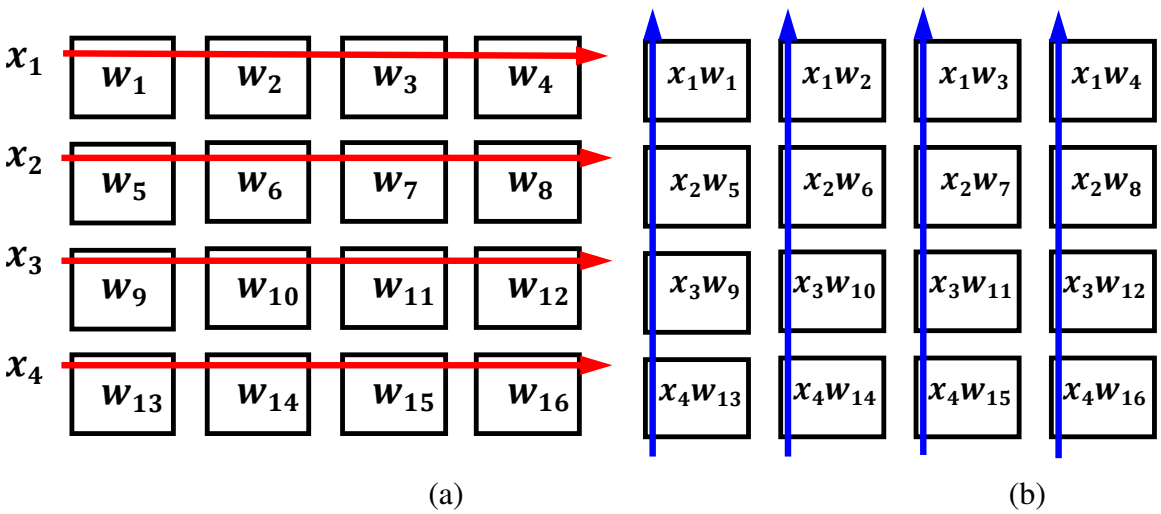


Figure 5.7: (a) Row-wise vector propagation in PE array for calculating pSUM (b) Vertical pSUM accumulation for vector-matrix multiplication in forward propagation of FC layers

array and we perform multiplication in each PE. The outcomes of computation (pSUMs) in each PE in the same column are propagated and accumulated vertically. The accumulated results in the first row of PE array are transferred to the global buffer.

5.6 Backpropagation and Gradient Descent

For TL followed by online RL, we train last 2/3/4 FC layers of the network. Backpropagation consists of two major computational steps: finding gradients of weights and their biases. Since we use our system to serially process one image at a time for training, the system must store the sum of weight and bias gradient of each image in the global buffer.

5.6.1 Backpropagation architecture of Fully-Connected Layer

The gradient of the weight is the result of multiplication of every vector element in a layer of neurons and every vector element in the gradient of the loss function computed with respect to the neurons in previous layer. Since there is no pSUM accumulation involved in calculating weight gradients, the results of multiplication of each PE are directly transferred to global buffer. The gradient of the bias in an FC layer is calculated by multiplying the vector of the gradient of Loss with respect to neurons in previous layer and the trans-

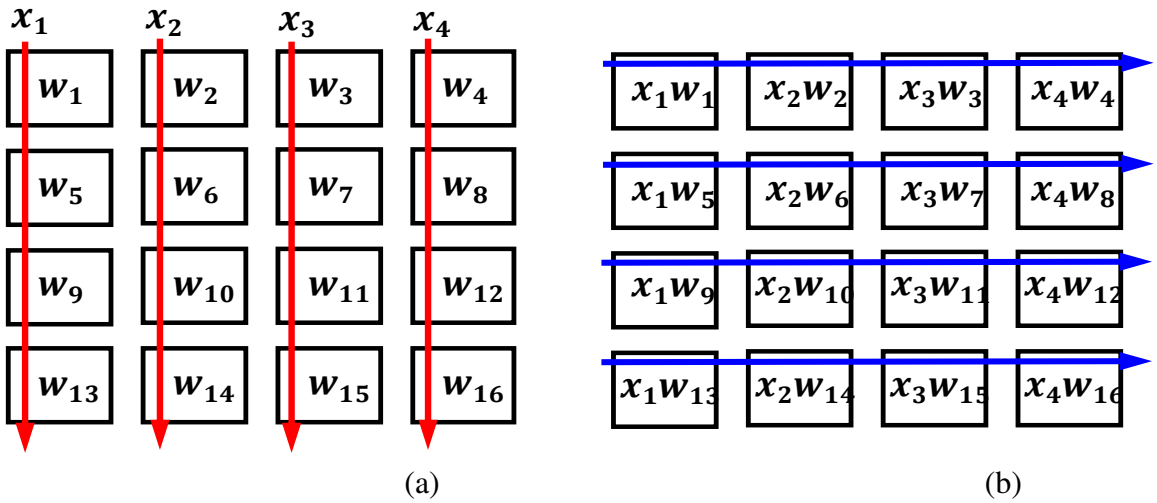


Figure 5.8: (a) Column-wise vector propagation in PE array for calculating pSUM (b) Row-wise pSUM accumulation for vector-transposed matrix multiplication in backpropagation of FC layers

posed weight matrix. The structure of the systolic array enables vector-transposed matrix multiplication without transposing the matrix itself, in a manner describe in [120] Fig. 5.8 describes the structure of vector-transposed matrix multiplication in the PE array. The vector elements are propagated downwards in each column of the array and the pSUM from each PE are accumulated row-wise. The computation is complete when PEs in the last column transfer their results to the global buffer.

5.6.2 Backpropagation architecture of CONV

The backpropagation of CONV layers only happen when evaluating the E2E RL in the system, which is our baseline design. For comparison to the baseline, we benchmark the backpropagation architecture for the entire network. For CONV layers, we use GEMM [121], where the system first reads the data from the STT-MRAM array to the logic die, and expands the inputs to each CONV layers in a 2D matrix. Once the expansion is complete, the backpropagation of CONV becomes same as the backpropagation of FC layers. After the weights of the CONV layers are updated, we write the weights back to the STT-MRAM array. We account for the additional on-chip SRAM requirement for storing the results of the intermediate compute steps.

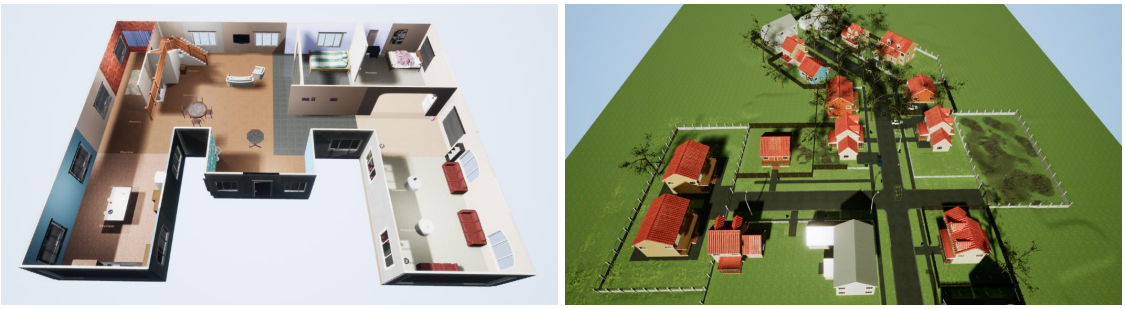
5.7 Simulation Setup

5.7.1 Hardware Architecture Simulation

We used NanGate 15nm FreePDK cell library to evaluate the hardware system performance [119]. We perform synthesis and place-and-route of the entire system and the results cited here (along with Fig. 5.4) are obtained post-synthesis.

5.7.2 Simulation Setup

The algorithm is tested on a simulated environment with the dynamics of realistic drones. Simulations were carried out on two types of simulated environments, Indoor and Outdoor. For each of the two categories, complex meta-environments and separate test environments were designed to train and test the performance of the proposed methodology respectively. We used the Unreal Engine 4 (UE4), used for video game development to design the simulation environments and emulate the necessary physics. For each of the two environment categories, a complex meta-environment and two test environments were designed for training and testing purposes. Hence a total of 6 (3 indoor, 3 outdoor) 3D were used in the simulation.



(a) (b)

Figure 5.9: Screenshots of the complex meta environments developed using UE 4.

The layouts and screenshots of these environments can be seen in Fig.5.9 and Fig.5.10 This engine interfaces with Tensorflow to train a drone via TL and RL. TensorFlow is used as the deep learning framework. AirSim [122] was used to interface the custom gen-

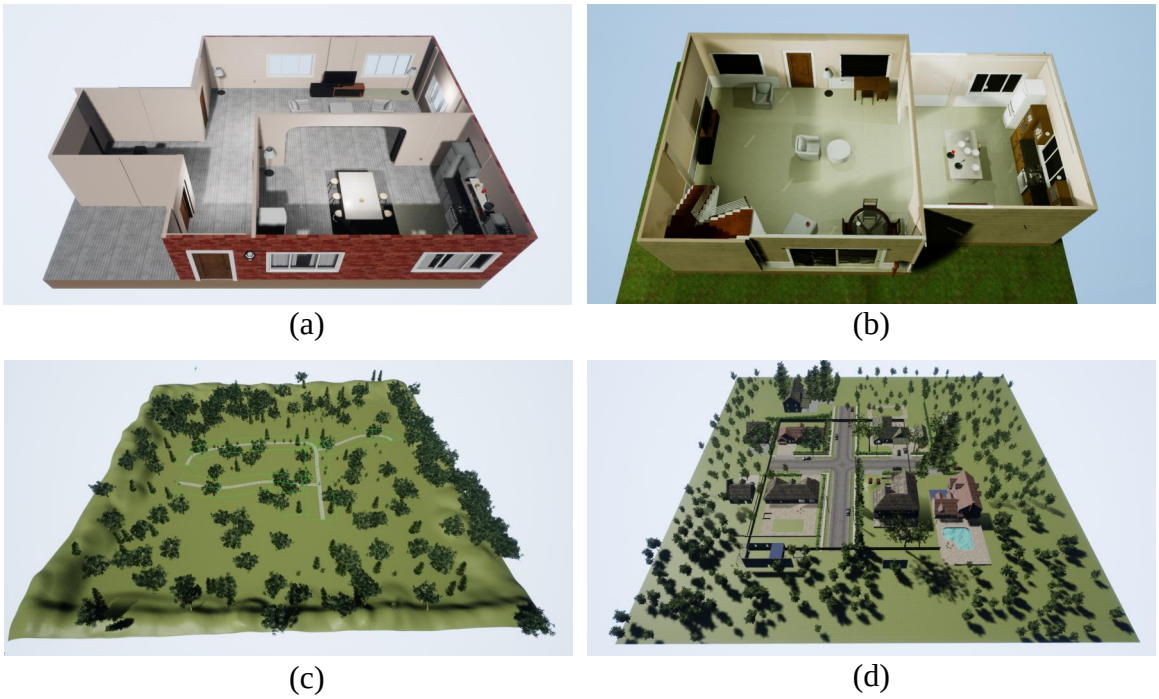


Figure 5.10: Screenshots of the test environments (a)Indoor Apartment (b)Indoor House (c)Outdoor Forest (d)Outdoor Town developed using UE 4.

erated 3D environments with python. The simulation and training was carried out on a workstation equipped with core i7 processor and NVIDIA GTX1080 GPU. The web-link for the suite of the environments, videos and corresponding data sets can be found here: https://tinyurl.com/y9wgpq4b and the implementation details are beyond the scope of this paper.

5.7.3 Training on Meta Environments

The drone is trained in the meta-environment for 60K iterations, initialized with ImageNet [123][124] weights. For the training, depth maps generated from stereo cameras are used, as shown in Fig.5.11. The drone is equipped with two cameras (left and right). The scene is captured using these two cameras and the disparity map is generated based on the distance between the corresponding pixels are in the left and right images. The disparity map is passed through a low pass filter to generate a depth map. A typical example is shown in Fig. 5.11. The training is carried out in two phases. In the first phase the DNN is trained

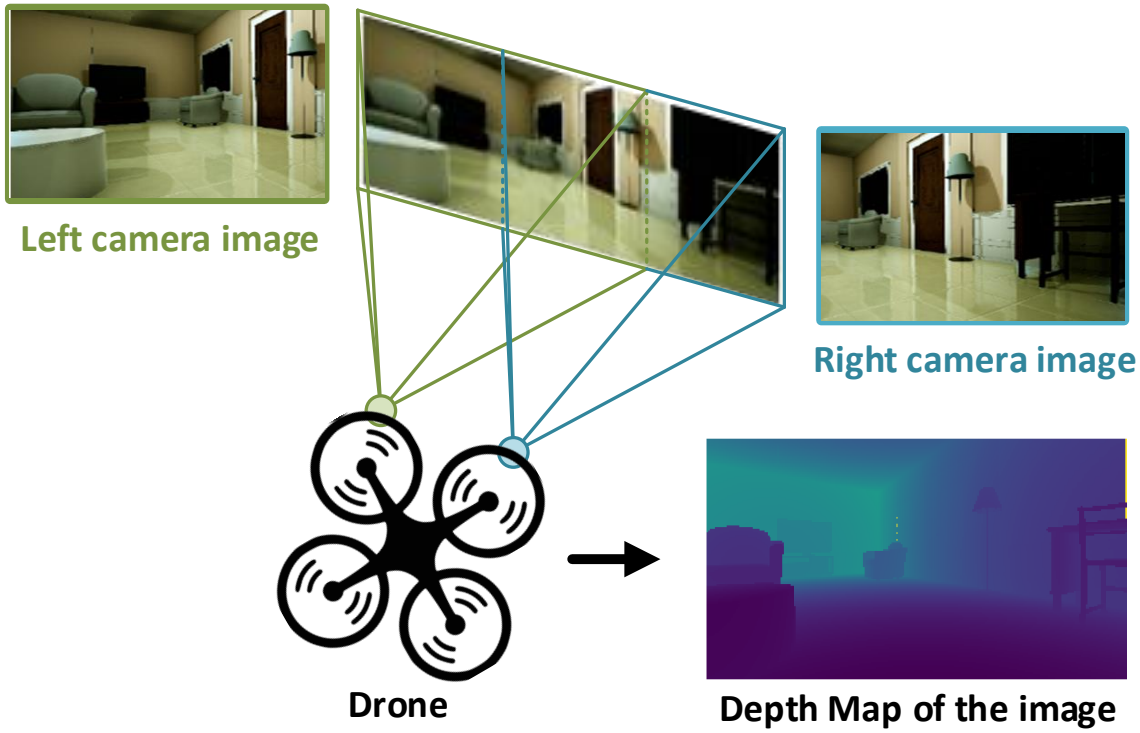


Figure 5.11: Stereo Vision based Depth Map Generation

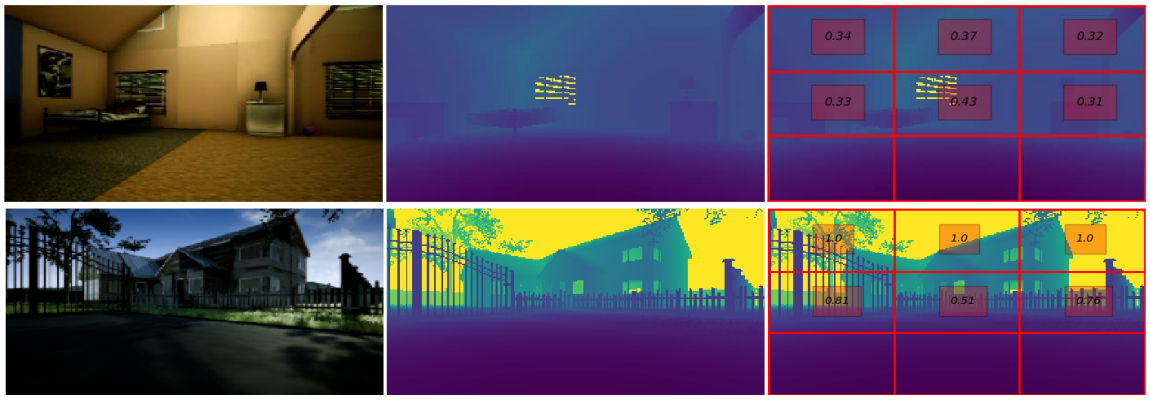


Figure 5.12: Feature extraction for SVM Classifier – On the left, the actual camera frame is shown. The depth map (in the center) is divided into windows and the top 6 windows are used towards feature extraction (right image).

on the complex meta indoor and outdoor environments separately. This DNN is initialized with ImageNet weights (and with random truncated normal weights for the additional layers i.e. FC3, FC4, FC5). In order to help converge the loss, various techniques discussed in [103], such as double deep Q-learning network (DDQN) and clipped temporal difference (TD) error are used. Apart from training the modified AlexNet network for the set of initial

weights, the meta-environment is also used to train a small binary SVM classifier in a supervised manner to differentiate between the indoor and outdoor category of environment. Since outdoor environments typically have objects placed at a larger distance as compared to the indoor environments, the use of the depth map (instead of the raw camera frames) for training the classifier comes as a natural choice. For each of the indoor and outdoor meta-environment, 1000 depth maps are collected. These 2D depth maps are converted into a feature vector of size 6 x 1 which is used as input to the binary SVM classifier to categorize what category of the environment these depth maps belong to. For each of the 2D depth map the feature vector is generated by slicing the depth map into 9 equal parts. The feature vector is the concatenation of the average of the largest 30% pixel values in the top 6 windows as shown in Fig. 5.12. The complete block diagram is shown in Fig 5.12. The classifier is trained on these feature vectors with training accuracy of 98.5% and it is tested on 200 data points from unseen indoor and outdoor environments with an accuracy of 97.02%.

5.7.4 Training on Test Environments

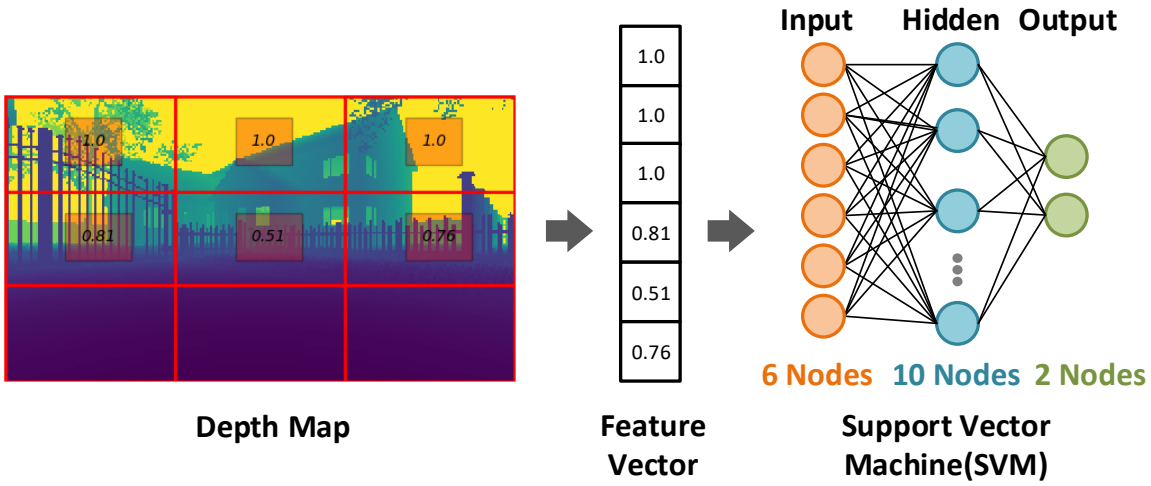


Figure 5.13: SVM Classifier Block Diagram

Once this training is completed for both the indoor and outdoor meta-environments separately, the transfer learning phase begins. In this phase, for each of the outdoor and

Table 5.1: List of hyper parameters for training

Learning Rate	N target	Batch size	Iteration
1e-6	500	32	60k

indoor category, a DNN is trained for the two test environments separately. The drone is placed in the test environment and uses the trained SVM classifier (Fig. 5.13) to categorize the environment it is in. The drone collects the depth map by rotating N times with an angle of 360/N degrees. Above mentioned features are extracted from these depth maps and fed to the classifier. Based on the majority label predicted by the binary classifier, the DNN is initialized with the respective trained meta-environment (Indoor or outdoor). Table 5.1 lists the hyper parameters used for training. N target is the number of training iteration after which the weights from the target network is copied into primary network in DDQN.

Algorithm 1 Reward generation using the depth map

function $f_r(s_t, a_t, s_t^{'})$
$\mathrm{d}(\mathrm{s_t}) \leftarrow depth\ map\ of\ s_t$
$\mathrm{d}(s_t^{'}) \leftarrow depth\ map\ of\ s_t^{'}$
$d^l(s_t), d^c(s_t), d^r(s_t) = DepthValues\big(d(s_t)\big)$
$d^l(s_t^{'}), d^c(s_t^{'}), d^r(s_t^{'}) = DepthValues\big(d(s_t^{'})\big)$
if $a_t = a_F$ **then** $r_t = d^c(s_t^{'})$
else if $a_t = a_L$ **then** $r_t = d^c(s_t^{'}) + \alpha\left(\mathrm{d^l}(s_t) - d^r(s_t)\right)$
else $r_t = d^c(s_t^{'}) + \alpha\big(\mathrm{d^r}(s_t) - d^l(s_t)\big)$
if $d^c(s_t^{'}) < \mathrm{d_{thresh}}$ **then** $\mathrm{r_t} = \mathrm{r_{crash}}$
return $\mathrm{r_t}$

Algorithm 3: Reward generation using the depth map. The superscript l,r,c with d denotes left, right, center value of depth map. the subscript F,L,R with a denotes the forward, left and right action. r and s are reward and state.[103]

Algorithm 3 describes how the depth map is used to generate a reward function for RL with the long-term goal of exploring an area without any collisions. The trained weights

are then used as initial weights for RL in the respective test environments For RL, we use 4 topologies, E2E (end-to-end RL) and L_2, L_3, and L_4, where L_i represents TL followed by RL where the last i-layers are trained online.

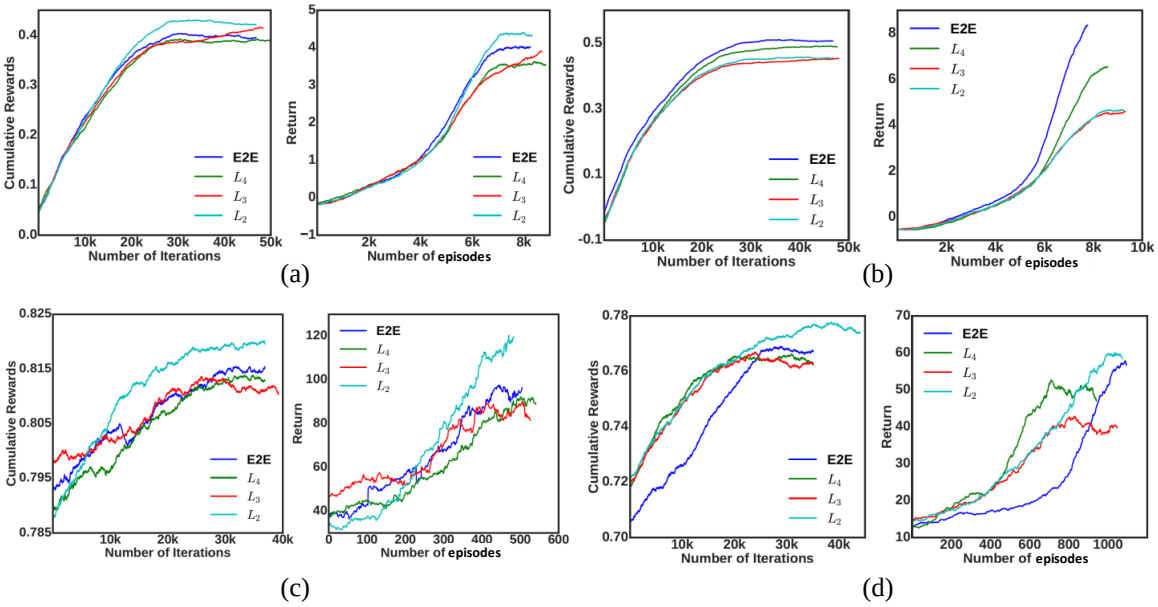


Figure 5.14: Cumulative rewards and return results in indoor (a)apartment (b)house and outdoor (c)forest (d)town test environments. The legend L_i indicates TL with last i-layers. All the algorithms show convergence and improving return loss indicating successful learning.

Fig. 5.14 reports the results for these test environments in terms of cumulative rewards and return while the safe flight is plotted in Fig 5.15. Cumulative reward is the moving average of last N rewards received by the agent and is given by $R_{\mathrm{i}} = \frac{1}{N}\sum_{j=i-N}^{i} r_{\mathrm{j}}$ where $i \geq N$ and N is a smoothing constant and was taken to be 15000. The return is the moving average of the sum of rewards across episodes. With each iteration, the agent takes an action and a reward is presented. These rewards are accumulated until the drone crashes and is given by $\frac{1}{N_{\mathrm{k}}}\sum_{j=i-N_{\mathrm{k}}}^{i} r_{\mathrm{j}}$where $\mathrm{N_k}$ is the number of actions taken between the $\mathrm{k^{th}}$ and $\mathrm{(k\text{-}1)^{th}}$crash. The return graph from Fig. 5.14 shows how the learned network performs, on average. Since the goal is not to get to a destination position, but rather to keep on moving around the arena, the return (cumulative reward before crashing) can theoretically become as large as possible. Hence as the system keeps on learning, the return graph will keep on increasing unless the topology itself isn't capable of learning

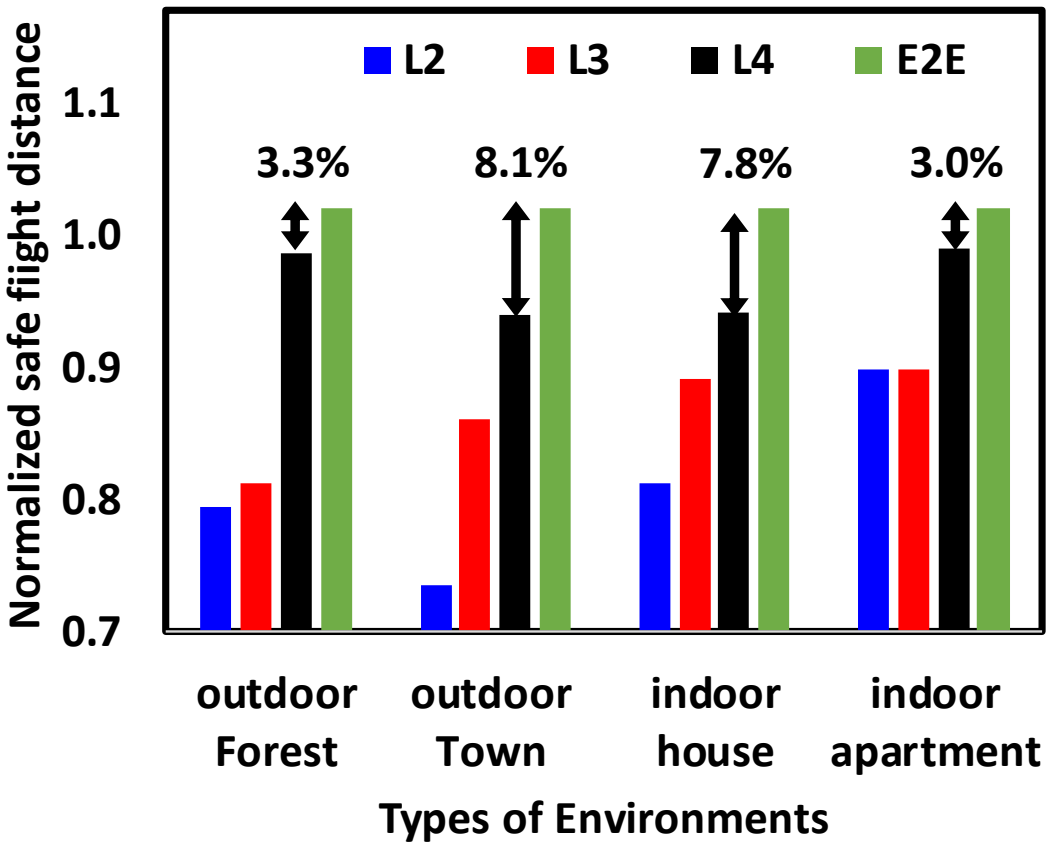


Figure 5.15: Normalized safe flight distance (SFD) with respect to different environments.

anymore due to limited number of trainable weights. This increase in the return may vary across topologies due to the random nature of the epsilon-greedy exploration [125]. The important takeaway from Fig.5.14 however, is that the return graph for the topologies with less number of trainable weights (L2, L3) doesn't get saturated at lower value of returns. This signifies superior learning capability of these topologies when initialized with meta-network weights, allowing the proposed technique to have comparable performance as E2E RL. Fig.5.15 plots the normalized Safe Flight Distance (SFD) across the topologies. The safe flight [103] is the average distance (in meters) traveled by the drone before it crashes and gives a more quantitative measure of how good the drone is in avoiding obstacles. From Fig. 5.14 we note that the system converges (saturating reward) for all the three scenarios showing the efficacy of the proposed algorithm. The normalized SFD shows acceptable degradation in performance (3% to 8.1%). In outdoor town environments the

meta-environment and test environments show large disparities (the type of houses, trees, cars etc. that the drone encounters) and shows the largest degradation. This can be further improved by performing TL on richer meta-environments.

5.8 Hardware Power-Performance Results

Layer	Processing Latency(ms)	Num. of Active PE	Power(mW)	Energy(mJ)
CONV1+ReLU+Maxpool	0.245	704	4134	1.012
CONV2+ReLU+Maxpool	1.087	960	5571	6.056
CONV3+ReLU	0.804	960	5674	4.564
CONV4+ReLU	1.28	960	5692	7.289
CONV5+ReLU+Maxpool	1.116	960	5672	6.33
FC1+ReLU	5.365	1024	6799	36.48
FC2+ReLU	1.189	1024	6800	8.091
FC3+ReLU	0.562	1024	6408	3.603
FC4+ReLU	0.28	1024	6410	1.8
FC5+ReLU	0.0005	160	1910	0.0009
total	**11.9285**	**880**	**5507**	**75.2259**

(a) Forward propagation system results

Layer	Processing Latency(ms)	Num. of Active PE	Power(mW)	Energy(mJ)	NVM Write
FC5+ReLU	0.0027	160	2094	0.006	**No**
FC4+ReLU	0.594	1024	6548	3.89	
FC3+ReLU	1.182	1024	6162	7.284	
FC2+ReLU	3.839	1024	5390	20.69	**Yes**
FC1+ReLU	29.19	1024	5390	157.3	
CONV5+ReLU+Maxpool	4.661	208	1888	8.804	
CONV4+ReLU	5.579	260	2112	11.78	
CONV3+ReLU	4.71	260	2112	9.947	
CONV2+ReLU+Maxpool	5.518	432	2850	15.73	
CONV1+ReLU+Maxpool	38.95	1024	5390	209.9	
total	**94.2257**	**644**	**3993.6**	**445.331**	

(b) Backward propagation system results

Figure 5.16: Latency, power and energy of each layers in forward and backward propagation

The hardware system is evaluated and the post-synthesis results are summarized in Fig.

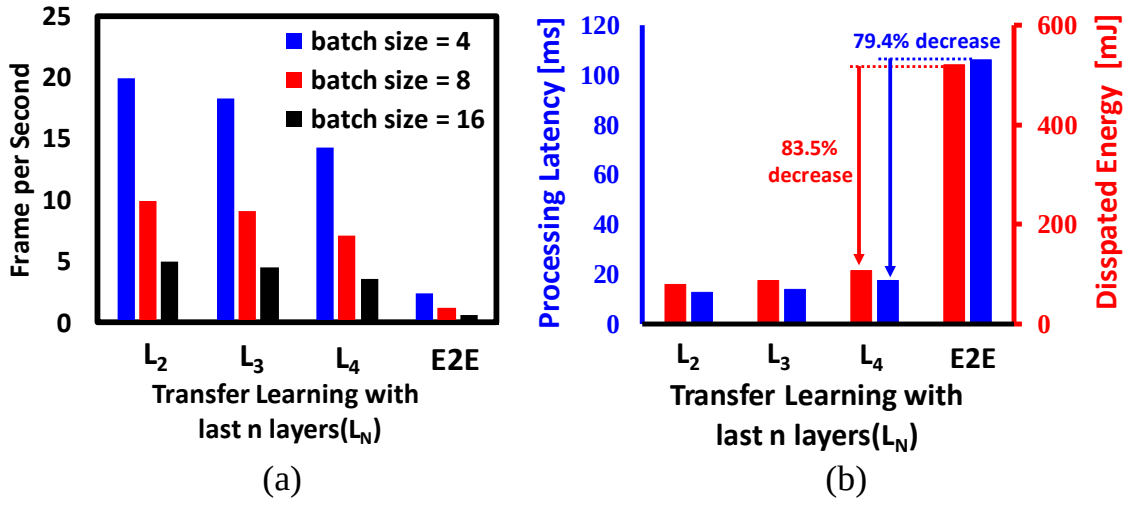


Figure 5.17: (a) Maximum fps supported by different algorithms as a function of batch size. (b) Estimated processing latency and energy dissipation

5.16 and Fig.5.17. The latency, energy and number of active PEs for the forward and backward propagation of data for each of the layers is shown in Fig. 5.16. The major bottleneck of the network layer in terms of processing latency in forward propagation is the first fully connected layer (FC1). Due to the size of its weights (75MB), most of the latency is attributed to data movement; fetching the weights from STT-MRAM to global buffer and distributing it from global buffer to the RF in the PE array. In backpropagation of the last three layers, the system does not access the STT-MRAM because the weights for the last three layers are stored in the global buffer. For backpropagation of the other layers, the weights from STT-MRAM are accessed to find the gradients of input to these layers and to store the gradients of the weights. In Fig. 5.17, we plot the maximum fps that can be supported in the proposed system vis-a-vis a baseline E2E RL system. We note that for a batch-size of 4, we can support 15fps for L_4, compared to just 3fps for E2E RL. This directly translates to more than 3X increase in the velocity of the drone (Fig. 5.1). We also achieve a 79.4% (83.45%) decrease in latency (energy) compared to the baseline. While E2E RL is not feasible in terms of energy and latency for small drones, the proposed solution opens up exciting opportunities for successful autonomous flight under strict power budgets.

Table 5.2: STT-MRAM[104][105][3] and HBM[126] energy parameters used in the system

Memory	Operation	Energy(pJ/b)
DRAM based HBMs	Read/Write	7
STT-MRAM	Read	0.7
	Write	4.5
HBM IO	Transmit/receive	5

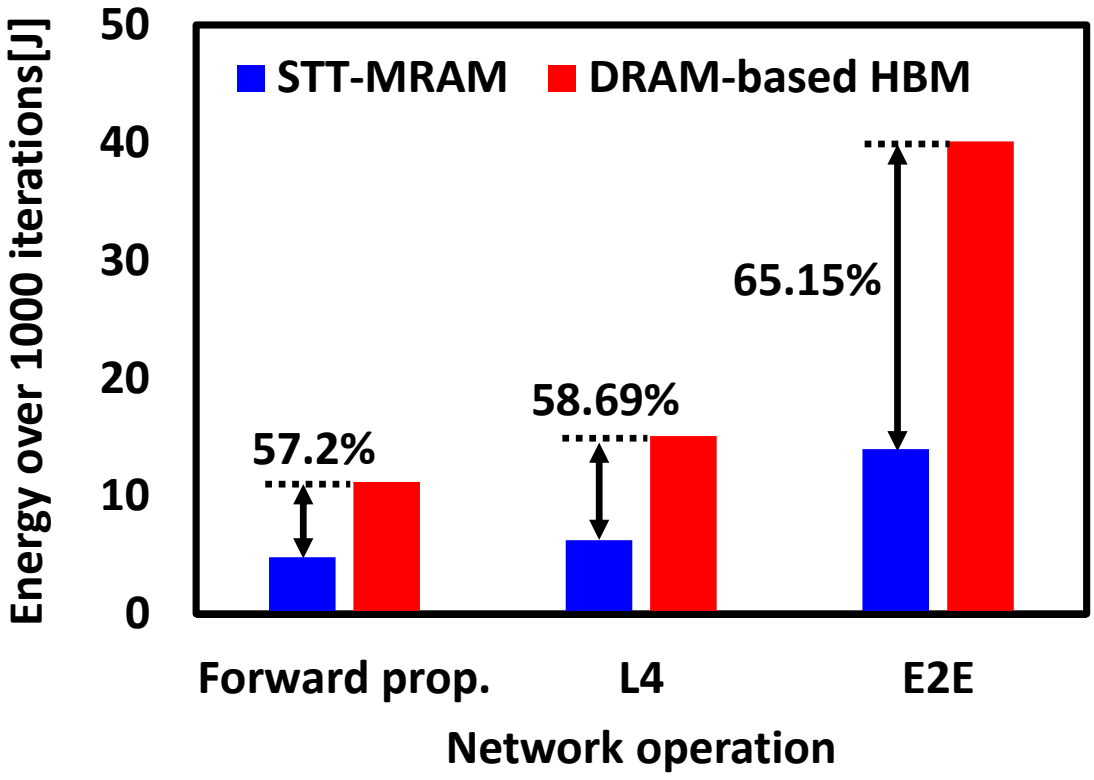


Figure 5.18: : Energy dissipation from DRAM-based HBM and STT-MRAM memory stack (off-chip) in case of Forward propagation, last 4 layer training (L_4) and E2E learning

In order to assess the need for eNVM in energy-efficient embedded systems, we compare the proposed STT-MRAM based system over a traditional DRAM-based HBM system. We use the parameters from Table. 5.2 to estimate the dissipated energy from each memory stack in three cases: (1) forward propagation, with no backpropagation, (2) forward propagation followed by learning the parameters of the last 4 layers (L4), and (3) End-to-End RL that involves forward propagation and full layer backpropagation across all the layers. Since the weights of the last three layers of the network reside in the global

buffer, the energy dissipation from the memory stack for L_2 and L_3 are same as that of forward propagation. The DRAM arrays in the DRAM-based HBM is refreshed every 64ms we consider the power cost of refreshing the entire 100MB following the JEDEC specifications.

Fig. 5.18 shows the energy dissipation of DRAM-based HBM and STT-MRAM based designs (off-chip) for 1000 iterations of forward propagation (no backpropagation), and training (gradient descent and backpropagation) for L_4 and E2E. The energy dissipation for the DRAM-based HBM from the figure is the sum of refresh, read/write and IO energy dissipation. Since STT-MRAM does not have refresh operation, the energy dissipation of STT-MRAM is the sum of read/write operation and IO energy dissipation. From the figure we observe that the energy dissipation from DRAM-based HBM is 2X greater than the energy dissipation from NVM in case of forward propagation. The difference in energy dissipation between DRAM-based HBM and STT-MRAM increases from L_4 to E2E since the number of refresh operations in the DRAM-based HBM is higher. This is attributed to the fact that it takes significantly longer to complete E2E compared to L_4.

5.9 Summary

In this research, we present a hardware-algorithm frame-work for STT-MRAM based embedded systems for application to small drones. We show that TL followed by RL on the last few layers of a deep CNN provides comparable performance compared to an E2E RL system, while reducing latency and energy by 79.4% and 83.45% respectively.

CHAPTER 6
CONCLUSION

In this dissertation, the research on post-CMOS memory based systems for machine learning and distributed optimization algorithm is presented. Among many post-CMOS memory devices, we focused on STT-MRAM and FerroFET because of their characteristics for in/near memory computing architecture. In the first chapter, prior works on STT-MRAM and FerroFET are presented and identified the major challenges of STT-MRAM and FerroFET based system at the device, circuit and system levels.

In chapter 2, a device level challenge of STT-MRAM, magnetic coupling of bit-cells of STT-MRAM, is presented. Since STT-MRAM is a nano-magnet, the external magnetic field from neighboring STT-MRAM cells can cause bit filp at the victim cell as the distance between other cells decreases as a result of memory scaling. The research provides the modeling of external magnetic field of neighboring STT-MRAM cells and the best and worst case data patterns for magnetic coupling are discovered based on the modeling. A comprehensive analysis of the effect of magnetic coupling on thermal stability of STT-MRAM, a property that affects the data retention of STT-MRAM is conducted. From the analysis, the research concluded that as the STT-MRAM memory becomes denser, the coupling field can cause significant change in the average retention time when STT-MRAM memory has lower thermal stability.

In chapter 3, a circuit level challenge of STT-MRAM, long retention testing time for STT-MRAM memory, is presented. In the conventional retention test scheme, the time for retention test takes too much time as the thermal stability of STT-MRAM increases because the bit flip in STT-MRAM happens stochastically. Therefore, in order to use STT-MRAM for commercial products, there is a strong need to shorten the retention test time. we propose an MBIST architecture (EMACS) capable of collecting statistical data in an

STT-MRAM subarray to estimate the thermal stability and retention. From our new retention test scheme, 93.75% improvement in test-time is shown compared to a brute-force approach [37] with less that 5% estimation error.

In chapter 4, a system level challenge of FerroFET, designing FerroFET based in-memory computing architecture with the limitations from FerroFET, is presented. First, the research identified the limitation of FerroFET as a computing element. Then it chooses distributed convex optimization via least square minimization as a template problem to show how much system performance and power gain can be achieved when a systolic PIM architecture based on analog FerroFET pseudo-crosspoint arrays with in-situ computation is used. The research also presents the accuracy loss of results from FerroFET based PIM architecture due to the device limitations. Finally, it summarizes the trade-off between system performance and accuracy loss of results in FerroFET based PIM architecture.

In chapter 5, a system & application level challenge of STT-MRAM based system, designing STT-MRAM based near-memory computing architecture for real-time reinforcement learning algorithm of drone applications, is presented. Since the target application is real-time reinforcement learning for a drone, it requires fast read and write latency and low energy dissipation from a memory. The research first shows that STT-MRAM has high write latency but it has lower energy dissipation and read latency. However, with the help of transfer learning, STT-MRAM can be used as a weight storage of neural network for reinforcement learning. With large on-chip SRAM buffer, the system can perform real-time training of last few layers of the neural network and the fixed weights of the rest of neural network are retrieved by reading STT-MRAM. The research demonstrates that the system can still exhibits almost equal performance in obstacle avoidance task of a drone.

REFERENCES

[1] C. Lin et al., "45nm low power cmos logic compatible embedded stt mram utilizing a reverse-connection 1t/1mtj cell", *IEDM*, 2009.

[2] Y. Chen et al., "Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks", *ISCA*, 2016.

[3] Q. Dong et al., "A 1mb 28nm stt-mram with 2.8ns read access time at 1.2v vdd using single-cap offset-cancelled sense amplifier and in-situ self-write-termination", *ISSCC*, 2018.

[4] K. Rho et al., "A 4gb lpddr2 stt-mram with compact 9f2 1t1mtj cell and hierarchical bitline architecture", *ISSCC*, 2017.

[5] O. Golonzka et al., "Mram as embedded non-volatile memory solution for 22ffl finfet technology", *IEDM*, 2018.

[6] A. Chen, "A review of emerging non-volatile memory (nvm) technologies and applications", *Solid-StateElectronics 125*, 2016.

[7] C. Lin et al., "45nm low power cmos logic compatible embedded stt mram utilizing a reverse-connection 1t/1mtj cell", *IEDM*, 2009.

[8] L. Luo et al., "Functionality demonstration of a high-density 2.5v self-aligned split-gate nvm cell embedded into 40nm cmos logic process for automotive microcontrollers", *IMW*, 2016.

[9] D. Shum et al., "40nm embedded self-aligned split-gate flash technology for high-density automotive microcontrollers", *IMW*, 2016.

[10] D. Kang et al., "256 gb 3 b/cell v-nand flash memory with 48 stacked wl layers", *JSSC*, 2017.

[11] S. Yu et al., "Binary neural network with 16 mb rram macrochip for classification and online training", *IEDM*, 2016.

[12] Q. Luo et al., "8-layers 3d vertical rram with excellent scalability towards storage class memory applications", *IEDM*, 2017.

[13] W. Kim et al., "Ald-based confined pcm with a metallic liner toward unlimited endurance", *IEDM*, 2016.

[14] J. Wu et al., "A 40nm low-power logic compatible phase change memory technology", *IEDM*, 2018.

[15] Y. Huai, "Spin-transfer torque mram (stt-mram): Challenges and prospects", *AAPPS bulletin*, vol. 18, no. 6, pp. 33–40, 2008.

[16] A. Fert, "Origin, development, and future of spintronics (Nobel lecture)", *Angewandte Chemie - International Edition*, vol. 47, no. 32, pp. 5956–5967, 2008.

[17] Y. Song et al., "Highly functional and reliable 8mb stt-mram embedded in 28nm logic", *IEDM*, 2016.

[18] A. Chintaluri et al., "Analysis of defects and variations in embedded spin transfer torque (stt) mram arrays", *JETCAS*, 2016.

[19] J. Muller et al., "Ferroelectricity in simple binary ZrO2 and HfO2", *Nano letters*,

[20] S. Mueller et al., "Next-generation ferroelectric memories based on fe-hfo 2", in *ISAF/ISIF/PFM, 2015 Joint IEEE International Symposium*.

[21] S. Oh et al., "Hfzro x-based ferroelectric synapse device with 32 levels of conductance states for neuromorphic applications", *IEEE EDL*,

[22] J. Woo et al., "Improved synaptic behavior under identical pulses using alox/hfo2 bilayer rram array for neuromorphic systems", *Electron Device Lett.,*, pp. 994–997, 2016.

[23] S. Park et al., "Neuromorphic speech systems using advanced reram-based synapse", *International Electron Devices Meeting(IEDM)*, 2013.

[24] H. Mulaosmanovic et al., "Evidence of single domain switching in hafnium oxide based FeFETs: Enabler for multi-level FeFET memory cells", in *Electron Devices Meeting (IEDM), 2015 IEEE International*, 2015.

[25] C. Chappert, A. Fert, and F. N. Van Dau, "The emergence of spin electronics in data storage.", *Nature materials*, vol. 6, no. 11, pp. 813–823, 2007.

[26] C. Augustine et al., "Numerical analysis of typical STT-MTJ stacks for 1T-1R memory arrays", *IEDM*, no. c, p. 22, 2010.

[27] K. C. Chun et al., "A Scaling Roadmap and Performance Evaluation of In-Plane and Perpendicular MTJ Based STT-MRAMs for High-Density Cache Memory", *Solid-State Circuits, IEEE Journal of*, 2013.

[28] Y. Chen et al., "Design margin exploration of spin-transfer torque RAM (STT-RAM) in scaled technologies", *IEEE TVLSI*, vol. 18, 2010.

[29] J. Kim et al., "A Technology-Agnostic MTJ SPICE Model with User- Defined Dimensions for STT-MRAM Scalability Studies", *CICC*, 2015.

[30] J. Kim et al., "Scaling analysis of in-plane and perpendicular anisotropy magnetic tunnel junctions using a physics-based model", *DRC*, 2014.

[31] E. Chen et al., "Advances and Future Prospects of Spin-Transfer Torque Random Access Memory", *Transactions on Magnetics*, 2010.

[32] J. H. Park et al., "Enhancement of data retention and write current scaling for sub-20nm STT-MRAM by utilizing dual interfaces for perpendicular magnetic anisotropy", *VLSIT*, 2012.

[33] W. Kim et al., "Extended scalability of perpendicular STT-MRAM towards sub-20nm MTJ node", *IEDM*, 2011.

[34] A. Raychowdhury et al., "Design space and scalability exploration of 1T-1STT MTJ memory arrays in the presence of variability and disturbances", *IEDM*, 2009.

[35] A. C. et al., "Analysis of defects and variations in embedded spin transfer torque (stt) mram arrays", *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2016.

[36] A. Chintaluri et al., "A Model Study of Defects and Faults in Embedded Spin Transfer Torque (STT) MRAM Arrays", *2015 IEEE 24th Asian Test Symposium (ATS)*, 2015.

[37] H. Naeimi, C. Augustine, A. Raychowdhury, S.-l. Lu, and J. Tschanz, *Intel Technology Journal, STTRAM Scaling and Retention Failure*, 1. 2013, vol. 17.

[38] R. Heindl et al., "Validity of the thermal activation model for spin-transfer torque switching in magnetic tunnel junctions", *Journal of Applied Physics*, 2011.

[39] M. Gokhale et al., "Processing in memory: The terasys massively parallel pim array", *Computer*, 1995.

[40] J. Draper et al., "The architecture of the diva processing-in-memory chip", in *International Conference on Supercomputing*, 2002.

[41] J. Suh et al., "A performance analysis of pim, stream processing, and tiled processing on memory-intensive signal processing kernels", in *ACM SIGARCH Computer Architecture News*.

[42] M. Hall et al., “Mapping irregular applications to diva, a pim-based data-intensive architecture”, in *Proceedings of the 1999 ACM/IEEE SC*.

[43] M. Jerry, “Ferroelectric fet analog synapse for acceleration of deep neural network training”, *IEEE International Electron Devices Meeting (IEDM)*, 2017.

[44] T. Böscke et al., “Ferroelectricity in hafnium oxide: CMOS compatible ferroelectric field effect transistors”, in *IEDM*, 2011.

[45] ——, “Phase transitions in ferroelectric silicon doped hafnium oxide”, *Applied Physics Letters*, 2011.

[46] J. Müller et al., “Ferroelectricity in yttrium-doped hafnium oxide”, *Journal of Applied Physics*, 2011.

[47] K. Florent et al., “First demonstration of vertically stacked ferroelectric Al doped HfO_2 devices for NAND applications”, in *VLSIT*, 2017.

[48] H. Mulaosmanovic, J. Ocker, S. Müller, M. Noack, J. Müller, P. Polakowski, T. Mikolajick, and S. Slesazeck, “Novel ferroelectric FET based synapse for neuromorphic systems”, in *VLSI Technology, 2017 Symposium on*, IEEE, 2017, T176–T177.

[49] R. Materlik et al., “The origin of ferroelectricity in Hf1-xZrxO2: A computational investigation and a surface energy model”, *Journal of Applied Physics*, 2015.

[50] B. L. Jackson et al., “Nanoscale electronic synapses using phase change devices”, *ACM Journal on Emerging Technologies in Computing Systems (JETC)*,

[51] D. Kuzum et al., “Synaptic electronics: Materials, devices and applications”, *Nanotechnology*,

[52] S. H. Jo et al., “Nanoscale memristor device as synapse in neuromorphic systems”, *Nano letters*, 2010.

[53] G. Indiveri et al., “Integration of nanoscale memristor synapses in neuromorphic computing architectures”, *Nanotechnology*, 2013.

[54] S. Yu et al., “A low energy oxide-based electronic synaptic device for neuromorphic visual systems with tolerance to device variation”, *Advanced Materials*, 2013.

[55] R. Sutton et al., *Introduction to Reinforcement Learning*. 1998.

[56] F. Sadeghi et al., “Cad2rl: Real single-image flight without a single real image”, *arXiv:1611.04201 [cs.LG]*,

[57] M. Anwar et al., "Navren-rl: Learning to fly in real environment via end-to-end deep reinforcement learning using monocular images", *arXiv:1807.08241 [cs.LG]*,

[58] H. Yang et al., "Threshold switching selector and 1s1r integration development for 3d cross-point stt-mram", *IEDM*, 2017.

[59] G. Jan et al., "Demonstration of fully functional 8mb perpendicular stt-mram chips with sub-5ns writing for non-volatile embedded memories", *VLSIT*, 2014.

[60] Q. Dong et al., "A 1mb 28nm stt-mram with 2.8ns read access time at 1.2v vdd using single-cap offset-cancelled sense amplifier and in-situ self-write-termination", *ISSCC*, 2018.

[61] M. Hosomi et al., "A Novel Nonvolatile Memory with Spin Torque Transfer Magnetization Switching :Spin-RAM", *IEDM*, 2005.

[62] S. Ikeda et al., "A perpendicular-anisotropy cofeb–mgo magnetic tunnel junction", *Nature Materials*, 2010.

[63] H. Sato et al., "Comprehensive study of CoFeB-MgO magnetic tunnel junction characteristics with single- and double-interface scaling down to 1X nm", *IEDM*, 2013.

[64] A. V. Khvalkovskiy et al., "Basic principles of STT-MRAM cell operation in memory arrays", *Journal of Physics D: Applied Physics*, 2013.

[65] S. Yuasa et al., 2013.

[66] Y. Zhang, W. Wen, and Y. Chen, "The Prospect of STT-RAM Scaling From Readability Perspective", *Transactions on Magnetics, 2012 IEEE International*, vol. 48, pp. 3035–3038, 2012.

[67] W. H. Choi, J. Kim, I. Ahmed, and C. H. Kim, "Comprehensive Study on Interface Perpendicular MTJ Variability", *Device Research Conference*, pp. 9–10, 2015.

[68] D. J. Griffiths, "Introduction to Electrodynamics, 4th Edition", 2012.

[69] W. Kang et al., "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.

[70] ——, "Yield and reliability improvement techniques for emerging nonvolatile STT-MRAM", *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 5, no. 1, pp. 28–39, 2015.

[71] I. Yoon et al., "EMACS: Efficient MBIST Architecture for Test and Characterization of STT-MRAM Arrays", *International Test Conference*, 2016.

[72] *15th IEEE VLSI Test Symposium (VTS'97), April 27-May 1, 1997, Monterey, California, USA*, IEEE Computer Society, 1997, ISBN: 0-8186-7810-0. [Online]. Available: `http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=4653`.

[73] T. Kawahara, K. Ito, R. Takemura, and H. Ohno, "Spin-transfer torque RAM technology: Review and prospect", *Microelectronics Reliability*, 2012.

[74] K. et al., "Basic principles of stt-mram cell operation in memory arrays", *Journal of Physics D: Applied Physics*, vol. 46, no. 7, p. 074 001, 2013.

[75] A. Raychowdhury et al., "Design space and scalability exploration of 1T-1STT MTJ memory arrays in the presence of variability and disturbances", *IEEE International Electron Devices Meeting(IEDM)*, 2009.

[76] A. Nigam, C. W. Smullen, V. Mohan, E. Chen, S. Gurumurthi, and M. R. Stan, "Delivering on the promise of universal memory for spin-transfer torque RAM (STT-RAM)", *Proceedings of the International Symposium on Low Power Electronics and Design*, vol. 1, pp. 121–126, 2011.

[77] A. Jog, A. K. Mishra, C. Xu, Y. Xie, V. Narayanan, R. Iyer, and C. R. Das, "Cache revive: Architecting volatile STT-RAM caches for enhanced performance in CMPs", *Proceedings of the 49th Annual Design Automation Conference (DAC)*, pp. 243–252, 2012.

[78] Z. Sun, X. Bi, H. H. Li, W.-F. Wong, Z.-L. Ong, X. Zhu, and W. Wu, "Multi retention level STT-RAM cache designs with a dynamic refresh scheme", *Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture*, pp. 329–338, 2011.

[79] A. Driskill-Smith et al., "Non-volatile spin-transfer torque RAM (STT-RAM): An analysis of chip data, thermal stability and scalability", *IMW*, 2010.

[80] M. Pakala et al., "Ciritical Current distribution in spin-transfer-switched magnetic tunnel junctions", *Journal of Applied Physics*, 2005.

[81] D. Montgomery and G. Runger, *Applied Statistics and Probability for Engineers*. John Wiley and Sons, 2010.

[82] R. Bishnoi, F. Oboril, and M. B. Tahoori, "Read Disturb Fault Detection in STT-MRAM", *International Test Conference*, pp. 1–7, 2014.

[83] G. Jan, L. Thomas, S. Le, Y.-j. Lee, H. Liu, J. Zhu, R.-y. Tong, K. Pi, Y.-J. Wang, D. Shen, R. He, J. Haq, J. Teng, V. Lam, K. Huang, T. Zhong, T. Torng, and P.-k. Wang, "Demonstration of fully functional 8Mb perpendicular STT-MRAM chips with sub-5ns writing for non-volatile embedded memories", *2014 Symposium on VLSI Technology (VLSI-Technology): Digest of Technical Papers*, vol. 093008, no. 2012, pp. 1–2, 2014.

[84] S. Gonugondla et al., "A 42pj/decision 3.12tops/w robust in-memory machine learning classifier with on-chip training", in *ISSCC*, 2018.

[85] P. Chi et al., "Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory", in *ISCA*, 2016.

[86] A. Shafiee et al., "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars", in *ISCA*, 2016.

[87] Z. Chen et al., "Optimized learning scheme for grayscale image recognition in a rram based analog neuromorphic system", in *IEDM*, 2015.

[88] M. Prezioso et al., "Training and operation of an integrated neuromorphic network based on metal-oxide memristors", *Nature*, 2015.

[89] G. W. Burr et al., "Large-scale neural networks implemented with non-volatile memory as the synaptic weight element: Comparative performance analysis (accuracy, speed, and power)", in *IEDM*, 2015.

[90] D. Fan et al., "Stt-snn: A spin-transfer-torque based soft-limiting non-linear neuron for low-power artificial neural networks", *T.Nanotech.*, 2015.

[91] P. J. S. Ferreira, "The stability of a procedure for the recovery of lost samples in band-limited signals", *Signal Processing*,

[92] R. Marks, "Restoring lost samples from an oversampled band-limited signal", *IEEE Transactions on Acoustics, Speech, and Signal Processing*,

[93] H. Stark, "Polar, spiral, and generalized sampling and interpolation", in *Advanced Topics in Shannon Sampling and Interpolation Theory*.

[94] S. Yu et al., "Scaling-up resistive synaptic arrays for neuro-inspired architecture: Challenges and prospect", in *IEDM*, 2015.

[95] M. Hu et al., "Memristor crossbar based hardware realization of bsb recall function", *Proc. Int. Joint Conf. Neural Networks*, 2012.

[96] B. Li et al., "Training itself: Mixed-signal training acceleration for memristor-based neural network", *Proc. 19th Asia and South Pacific Design Automation Conf.*, 2014.

[97] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator", *SIGARCH Comput. Archit. News*, vol. 39, no. 2, pp. 1–7, Aug. 2011, ISSN: 0163-5964. DOI: 10.1145/2024716.2024718. [Online]. Available: http://doi.acm.org/10.1145/2024716.2024718.

[98] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "Mcpat: An integrated power, area, and timing modeling framework for multicore and manycore architectures", in *Proceedings of the 42Nd Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO 42, New York, New York: ACM, 2009, pp. 469–480, ISBN: 978-1-60558-798-1. DOI: 10.1145/1669112.1669172. [Online]. Available: http://doi.acm.org/10.1145/1669112.1669172.

[99] T. Gokmen and Y. Vlasov, "Acceleration of deep neural network training with resistive cross-point devices: Design considerations", *Frontiers in neuroscience*, vol. 10, 2016.

[100] B. Murmann, "Adc performance survey 1997-2017,available: Http://web.stanford.edu/ murmann/adcsurvey.html",

[101] R. Sutton et al., "Introduction to reinforcement learning", 1998.

[102] F. Sadeghi et al., "Cad2rl: Real single-image flight without a single real image", *arXiv:1611.04201 [cs.LG]*,

[103] M. A. A. et al., "Navren-rl: Learning to fly in real environment via end-to-end deep reinforcement learning using monocular images", *arXiv:1807.08241 [cs.LG]*,

[104] H. Yang et al., "Threshold switching selector and 1s1r integration development for 3d cross-point stt-mram", *IEDM*, 2017.

[105] G. Jan et al., "Demonstration of fully functional 8mb perpendicular stt-mram chips with sub-5ns writing for non-volatile embedded memories", *VLSIT, doi: 10.1109/VLSIT.2014.6894357*, 2014.

[106] R. Sutton, "Learning to predict by the method of temporal differences", *Machine Learning. Springer. 3: 9–44. doi:10.1007/BF00115009*,

[107] M. van Otterlo et al., "Reinforcement learning and markov decision processes", *Reinforcement Learning. Adaptation, Learning, and Optimization*, 2012.

[108] L. Kaelbling et al., “Reinforcement learning: A survey”, 1996.

[109] A. Amaravati et al., “A 55nm time-domain mixed-signal neuromorphic accelerator with stochastic synapses and embedded reinforcement learning for autonomous micro-robots”, *ISSCC*, 2018.

[110] ——, “A 55-nm, 1.0-0.4v, 1.25-pj/mac time-domain mixed-signal neuromorphic accelerator with stochastic synapses for reinforcement learning in autonomous mobile robots”, *JSSC*, 2019.

[111] A. Anwar et al., “Navren-rl: Learning to fly in real environment via end-to-end deep reinforcement learning using monocular images”, *M2VIP*, 2018.

[112] Y. Chen et al., “Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks”, *JSSC*, 2017.

[113] M. Jerry et al., “Ferroelectric fet analog synapse for acceleration of deep neural network training”, *IEDM*, 2017.

[114] D. S. et al., “Transfer learning for multiagent reinforcement learning systems”, *IJCAI*, 2016.

[115] M. Taylor et al., “Transfer learning for reinforcement learning domains: A survey”, *JMLR*, 2009.

[116] ——, “Cross-domain transfer for reinforcement learning”, *ICML*, 2007.

[117] I. Goodfellow et al., “Deep learning”, 2016.

[118] “Jedec standard high bandwidth memory(hbm) dram speficication”, *JESD235B*, 2015.

[119] M. Martins et al., “Open cell library in 15nm freepdk technology”, in *ISPD*, 2015.

[120] D. O’Leary, “Systolic arrays for matrix transpose and other reorderings”, *IEEE Trans. Computers, vol. C-36, no. 1.*,

[121] J. Bottleson et al., “Clcaffe: Opencl accelerated caffe for convolutional neural networks”, *IPDPSW*, 2016.

[122] S. Shah et al., “Airsim: High-fidelity visual and physical simulation for autonomous vehicles, field and service robotics”, 2017.

[123] A. Krizhevsky et al., “Imagenet classification with deep convolutional neural networks”, *Commun. ACM 60, 6*, 2017.

[124] J. Deng et al., “Imagenet: A large-scale hierarchical image database”, *CVPR*, 2009.

[125] C. Watkins, “Learning from delayed rewards”, *PhD thesis, University of Cambridge, Cambridge, England*, 1989.

[126] “Https://www.cs.utah.edu/thememoryforum/mike.pdf”,