Investigating Probabilistic Computing: Devices, Circuits, and Systems

A THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

BY

AMINA HAROON

(PHD18105)

UNDER THE GUIDANCE OF

Dr. SNEH SAURABH

PROFESSOR, IIIT-DELHI

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING,

INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY DELHI

NEW DELHI-110020

September, 2025

Investigating Probabilistic Computing: Devices, Circuits, and Systems

BY

AMINA HAROON

(PHD18105)

UNDER THE GUIDANCE OF

Dr. SNEH SAURABH

PROFESSOR, IIIT-DELHI

A THESIS

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING,

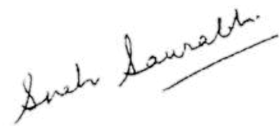
INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY DELHI

September, 2025

Certificate

This is to certify that the thesis titled **Investigating Probabilistic Computing: Devices, Circuits, and Systems**, submitted by **Amina Haroon**, to the Indraprastha Institute of Information Technology Delhi, for the award of the degree of Doctor of Philosophy, is an original research work carried out by her under my supervision. In my opinion, the thesis has reached the standard fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree or diploma.



Dr. Sneh Saurabh
(Thesis Advisor)
Professor,
Department of Electronics and Communication Engineering,
Indraprastha Institute of Information Technology Delhi-110020, India.

Place: New Delhi
Date: September, 2025

Declaration

This is certified that the thesis entitled **Investigating Probabilistic Computing: Devices, Circuits, and Systems**, submitted by me to the Indraprastha Institute of Information Technology Delhi, for the award of the degree of **Doctor of Philosophy**, is a bonafide work carried out by me. This research work has been carried out under the supervision of **Dr. Sneh Saurabh**. The study pertaining to this thesis has not been submitted in part or in full, to any other University or Institution for the award of any other degree.

Amina Haroon
Ph.D. Student,
Department of Electronics and Communication Engineering,
Indraprastha Institute of Information Technology Delhi-110020, India.

Place: New Delhi
Date: September, 2025

Abstract

The semiconductor industry has been driven by digital computation using binary digits, following Moore's law for over half a century. However, the demands of emerging applications like artificial intelligence (AI), cloud computing, exascale computing, and the Internet of Things (IoT) require scalable computational resources. Implementing these applications sometimes requires solving a certain class of problems, such as optimization problem, which requires stochasticity to arrive at the solution. It should be noted that the optimization problems can be implemented using digital computation that includes a pseudo-random number generator. However, the area- and power-overhead to implement a pseudo-random number generator is significant. One way to reduce the area- and power-overhead is to choose the computational paradigm where the stochasticity is inherently present in the fundamental building block. A recently proposed computational paradigm, probabilistic computing has shown promising results as an energy- and area-efficient alternative to digital computation.

The fundamental building block of probabilistic computing is a probabilistic bit or a p-bit. A p-bit is a classical entity similar to bits with logic levels 0 and 1, however, unlike bits, a p-bit fluctuates between the two logic levels. The p-bits can be realized using different semiconductor devices such as metal-oxide-semiconductor field-effect transistor (MOSFET), diodes, low-barrier magnets (LBM), etc. Among these alternatives, the LBM-based implementation have shown promising results in reducing the area and power overhead.

This work provides a comprehensive overview of probabilistic computing, from material physics to the system level, to optimize the entire stack from algorithms to device design and address challenges in hardware implementation to enhance the performance and robustness of applications using probabilistic computing.

In this thesis, the design of an LBM for integration in the one transistor-one magnetic tunnel junction spin-transfer torque magnetic random access memory (1T-1MTJ

STT-MRAM) structure is investigated. A method is proposed for the selection of material parameters in the LBM-based p-bit implementation to improve flips per second (fps), a critical system-level metric. The significance of specific material properties in the LBM design is highlighted. It is demonstrated that, beyond material selection, several design parameters are significantly influenced by process-induced variations and are, therefore, critical to the development of a robust p-bit-based computational system.

Subsequently, the work systematically investigates the impact of non-idealities, such as process variations, environmental factors, and ageing, on the performance of p-bit networks. An analytical model is proposed to incorporate these non-idealities, and the model's predictions are validated using numerical and SPICE simulations. For demonstration, the image completion problem for digits (0 to 9) using non-ideal p-bits is implemented. The analytical model closely aligns with the behavioral model, revealing that non-idealities in p-bits significantly affect the performance of probabilistic computing. Furthermore, the impact of these non-idealities in p-bit-based implementations is demonstrated in circuit simulations using SPICE models. This work highlights the importance of considering process-induced variations when designing p-bit networks. By incorporating these considerations, the performance and robustness of p-bit networks can be enhanced, paving the way for their real-world application.

In line with efforts to enhance the robustness of p-bit networks, this work investigates the impact of faults arising from fabrication defects, ageing, and variability in the p-bits that lead to stuck-at faults, which can degrade system functionality. The effect of such faults is examined using the Modified National Institute of Standards and Technology (MNIST) dataset, and a mutual information-based criticality score (CS) is proposed to guide fault-tolerance strategies. To further improve fault resilience, testable, isolatable, and fault-tolerant p-bit architectures are proposed and validated through Simulation Program with Integrated Circuit Emphasis (SPICE) simulations using $14nm$ Fin Field-Effect Transistor (FinFET) technology. Testable p-bits integrate conventional p-bits with scan cells, introducing controllability and observability into the network. Isolatable p-bits enable the disconnection of faulty p-bits, while fault-tolerant p-bits restore network functionality by activating faulty p-bits with redundant counterparts. By selectively replacing only the most critical p-bits, accuracy degradation is minimized with

limited overhead, thus demonstrating an effective framework for fault-tolerant p-bit systems.

Next, the exploration of probabilistic computing from an algorithmic perspective is discussed. The effectiveness of a p-bit system in tasks such as image completion, where the system uses partially clamped inputs, such as images of digits (0 to 9), to generate a complete output is demonstrated. Additionally, a method is proposed to sparsify a probabilistic computing network by leveraging mutual information, a concept from information theory. The findings show that the proposed method is computationally efficient and can produce a sparse network with only 42% of the original connections while delivering accuracy comparable to the fully connected network.

In summary, this work comprehensively investigates probabilistic computing, spanning device materials, circuit-level implementations, and system-level design considerations, focusing on understanding non-idealities and enabling fault tolerance for practical applications.

Dedicated to the loving memory of my mother...

Acknowledgements

"No one who achieves success does so without the help of others. The wise and confident acknowledge this help with gratitude."

– Alfred North Whitehead

First and foremost, I am truly grateful to the Almighty, whose guidance, strength, and unwavering support sustained me throughout the course of this research. In addition to my own efforts, many individuals have supported, guided, and encouraged me throughout this journey, contributing significantly to both my academic and personal growth, and making this thesis possible.

I would like to express my deepest gratitude to my advisor, Dr. Sneh Saurabh, for his invaluable guidance, mentorship, and unwavering support throughout the course of this research. His insightful feedback, patience, and encouragement have been instrumental in shaping the direction of this work. His high standards, dedication to research, and attention to detail have profoundly influenced my academic growth. It has been a privilege to work under his supervision and learn from his vast experience.

I would like to express my sincere gratitude to my collaborator, Dr. Ram Krishna Ghosh, for valuable technical discussions and insightful feedback, which significantly contributed to refining the quality and depth of this work.

I sincerely thank the members of my thesis committee, Dr. Sumit J. Darak and Dr. Debajyoti Bera, for their time, insightful feedback, and valuable suggestions throughout the course of this work. I am grateful for their encouragement, and for the constructive feedback they provided, which significantly contributed to the improvement of this research.

Special thanks go to the members of Nanoscale Devices and Circuits research group, especially Dr. Shelly Garg, Dr. Abhinav Gupta, Ms. Jasmine Kaur, and Ms. Pooja Beniwal for their help in simulations, reviews, and critical brainstorming sessions. I also wish to acknowledge the recent additions to our research group, Mr. Yashwardhan

Tyagi, Mr. Rithu Sagar, Mr. Nitin Dwivedi, and Ms. Garima, whose thoughtful questions about research directions, the purpose of pursuing a Ph.D, and broader academic themes led to many thought-provoking discussions. I would also like to thank Mrs. Kainat Yasmeen, Mr. Syed Asrar Ul Haq, Mr. Mohd. Aamir, and Ms. Somya Sharma for their support and motivation. The bonds of friendship formed with them are truly special and will remain close to my heart forever. I also extend my heartfelt thanks to the academic staff and the IT support team at IIIT Delhi for their prompt assistance throughout the course of my research journey.

Beyond academia, I am deeply thankful to my family. My husband, Mr. Rafeequl Islam, has been a pillar of strength and unwavering support throughout my research journey. I owe my deepest gratitude to my family, whose unwavering support made this journey possible. To my parents, for their endless love and belief in my abilities. This achievement stands on the foundation they built for me. To my siblings and their families, thank you for your constant emotional support throughout this process. I could not have done this without you all.

I am also deeply grateful to my extended family and in-laws for their constant support, patience, and kindness throughout this journey. Their encouragement provided a much-needed space of comfort and calm during the most stressful times.

Finally, I would like to thank IIIT Delhi for providing an excellent infrastructure and research environment. I would also like to thank the Tata Consultancy Services Research Scholarship Program (TCS-RSP) for sponsoring my research.

Contents

Abstract **i**

Acknowledgements **v**

List of Figures **xi**

List of Tables **xii**

List of Algorithms **xiii**

List of Abbreviations **xiv**

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Objectives . . . 3
- 1.3 Contributions . . . 4
- 1.4 List of Publications . . . 5
- 1.5 Organization . . . 6

2 Background and related work **8**
- 2.1 Introduction to probabilistic computing . . . 8
  - 2.1.1 Design of a single p-bit . . . 10
  - 2.1.2 Design of interconnected network of p-bits . . . 19
- 2.2 Applications of probabilistic computing . . . 23

3 Implementation of Probabilistic bits using Low Barrier Magnets **26**
- 3.1 Material optimization oriented design of a p-bit . . . 26
  - 3.1.1 Design of LBM for p-bit implementation . . . 26
- 3.2 Conclusion . . . 33

4 Impact of Non-Idealities on the Behavior of Probabilistic Computing **34**
- 4.1 Model of a Non-ideal p-bit . . . 34
- 4.2 Modeling Variations in a p-bit implementation . . . 39
- 4.3 Impact of non-idealities in a p-bit on a p-bit network . . . 44
- 4.4 Image Completion using a p-bit network with non-ideal p-bits . . . 48
- 4.5 Assessing impact of variations in 1T-1MTJ network . . . 52
- 4.6 Conclusion . . . 56

5 Fault-Tolerant Design Framework for Probabilistic-Bit (P-Bit) Systems **58**
- 5.1 Designing a network of p-bits . . . 58
- 5.2 Identifying critical p-bits in a p-bit network . . . 60
- 5.3 Fault detection in a p-bit network . . . 67

5.3.1 Testable P-bits 68
5.3.2 Detect and identify faulty p-bits in a network 70
5.4 Isolatable p-bit 72
5.5 Fault-tolerant P-bit 75
5.6 Performance evaluation of Proposed P-bits 80
5.6.1 Delay 80
5.6.2 Power dissipation 81
5.7 Conclusion 82

6 Image completion using Sparse Probabilistic Computing network 84
6.1 Network optimization oriented realization of sparse p-computing network 84
6.1.1 Deriving Fully Connected probabilistic computing network . 85
6.1.2 Deriving sparse probabilistic computing network 87
6.1.3 Results 93
6.2 Conclusion 96

7 Conclusion and Future Work 98
7.1 Summary 98
7.2 Future Work 99

References 101

Publications 117

Brief Biodata of the Author 118

List of Figures

2.1 (a) A generic model of a p-bit featuring input-output terminals denoted as W-R. (b) Illustrates the variation of m_i with I_i ranging from -4 to $+4$. The average of m_i, denoted as $< m_i >$, is plotted for $10,000$ samples of m_i at each I_i value. (c) Probability distributions of m_i for $\beta = 1$ and $I_i = \{-1, 0, +1\}$. [1]. 10

2.2 (a) The schematic of an Low Barrier Magnet (LBM). The energy barrier, E_b for (b) $E_b >> k_BT$ (c) $E_b \approx k_BT$. For $E_b \approx k_BT$, the thermal fluctuations are sufficient to change the state of the LBM from $+1$ to -1 or vice versa. 12

2.3 Two types of p-bit design using stochastic MTJ. (a) 1T-1MTJ STT MRAM based design of a p-bit. (b) Equivalent circuit of 1T-1MTJ STT MRAM based p-bit [1]. (c) Spin-Orbit Torque (SOT) MRAM based implementation of the p-bit. (d) Equivalent circuit of Spin-Orbit Torque (SOT) MRAM based p-bit [2]. 13

2.4 (a) Input-output characteristics of the modified 1T-1MTJ-based p-bit with V_{IN} varying from $-V_{DD}/2$ to $+V_{DD}/2$. (b) Probability distributions of V_{OUT} for three values of $V_{IN} = \{-V_{DD}/2, 0, V_{DD}/2\}$, shown in yellow, red, and green, respectively ($V_{DD} = 0.8\,V$ for the 14nm HP-FinFET technology) [2]. 15

2.5 A diagram illustrating the creation of a pinhole [3]. 17

2.6 The schematic for two-input AND gate implemented using p-bits. The h and J values represent the weighted connections [1]. 20

2.7 (a-d) The probability distribution for AND gate working in the forward mode i.e., $(A, B) \in \{(-1, -1), (-1, 1), (1, -1), (1, 1)\}$ and p-bit C is floating. The numbers along the X-axis represent the decimal encoding of the states represented as (ABC). For example, the states $(-1, -1, -1)$, $(-1, -1, 1)$, and $(1, 1, 1)$ are encoded as 0, 1, and 7, respectively. (e,f) The probability distribution for the AND gate working in the backward mode [1]. 21

2.8 Circuit diagram for a two-input AND gate. The weight values represented by h and J are mapped to resistance values as detailed in [1]. 22

2.9 Applications of probabilistic computing [1], [4]. 24

3.1 (a) Schematic illustration of an LBM. (b) Thermal fluctuations between two logic levels -1 and $+1$ separated by E_b. Magnetization dynamics on the Bloch sphere for circular IMA: (c) $E_b < 2\,k_BT$, $H_k \approx 100\,Oe$, (d) $2\,k_BT < E_b < 4\,k_BT$, $H_k > 500\,Oe$ [5]. 28

3.2 Plot of fps versus H_k for different values of M_s. The analysis is performed for 30 values of M_s, separated linearly in the range of $(300, 2000)\,emu/cc$. However, the graph is plotted only for 14 values of M_s to improve readability. 31

3.3 Average of V_{OUT} ($<V_{OUT}>$) for different values of V_{IN} in the range $(-V_{DD}/2, V_{DD}/2)$. For LBM-{A, B, C, D, E, and F}, for each value of V_{IN}, V_{OUT} is averaged over $500\,ns$. For LBM-{G, H, and I}, the average window is increased to $1\,\mu s$ to incorporate more fps. The inset shows the comparison of the sigmoid response of LBM-{A and D} with LBM-{F, G, H, and I} for $V_{IN} \approx 0\,V$. 32

4.1 Illustration of impact of q and r on $f_{\mathbf{H}}(h)$. The shaded area represents the p-bits' probability p being +1. 38
4.2 Circuit implementation of 1T-1MTJ-based p-bit using SPICE models [2]. 40
4.3 Comparison of the results obtained from the circuit simulation of 1T-1MTJ-based p-bit implementation and the analytical model. A scaling factor $m = 88.49/V$ is included such that, $\mathbf{Y} = \text{sgn}(\tanh(m * V_{IN}) + \mathbf{E})$ models the output response of the p-bit. 41
4.4 Comparison of the results obtained from the circuit simulation and the analytical model. The nominal value of the fin height $h_{fin}^{nom} = 21nm$ is marked in blue. A scaling factor $c = -0.232/nm$ is included such that, $\mathbf{Y} = \text{sgn}(\tanh(m * V_{IN} - c * (h_{fin} - h_{fin}^{nom})) + \mathbf{E})$ models the output response of the p-bit. . 42
4.5 Comparison of the results obtained from the circuit simulation and the analytical model. The nominal value of $G_0^{nom} = 46\mu S$ is marked in blue. A scaling factor $k = 0.05/\mu S$ is included such that, $\mathbf{Y} = \text{sgn}(\tanh(m * V_{IN}) + (\mathbf{E} - k * (G_0 - G_0^{nom}))$ models the output response of the p-bit. 43
4.6 Impact of non-ideality in a p-bit on the functionality of the AND gate. Only one term among (q_A, r_A), (q_B, r_B), and (q_C, r_C) is considered to be non-zero in each figure. 46
4.7 Illustration of canceling effect of non-idealities of p-bits in an AND gate. . 48
4.8 (a) Training images for a p-bit network with fifteen p-bits. (b) Test images where six out of fifteen pixels in an image is clamped. (c) The images generated by the p-bit network implemented using ideal p-bits. The colormap at the top indicates the colors corresponding to the respective pixel values. . . . 50
4.9 Results of Monte Carlo (MC) simulations for (a) $\mu_q = \mu_r = 0$, and $\sigma_q = \sigma_r = 0.01$, (b) $\mu_q = \mu_r = 0$, and $\sigma_q = \sigma_r = 0.02$, and (c) $\mu_q = \mu_r = 0$, and $\sigma_q = \sigma_r = 0.05$. (d) Expected value of E_{RMS} with varying standard deviation of the q and r. 51
4.10 The connection to a single p-bit in a 26- p-bit network. The p-bit shown here has a V_{fix}^{i} indicating the clamped input. 53
4.11 Prediction of the image classification circuit with nominal design parameters for the test cases: (a) $T1$ (b) $T2$. 53
4.12 Predictions of the image classification circuit with global variations in the G_0 values of the MTJ for each p-bit. For $G_0 = 30\mu S$, the plot shows the result for (a) $T1$ and (b) $T2$. For $G_0 = 70\mu S$, the plot shows the result for (c) $T1$ and (d) $T2$. 54
4.13 Results of MC simulations for $\mu_{G_{0_i}} = 46\mu S$ (a) $\sigma_{G_{0_i}} = 0.01 \times \mu_{G_{0_i}}$ and (b) $\sigma_{G_{0_i}} = 0.2 \times \mu_{G_{0_i}}$. (c) Expected value and (d) standard deviation of the E_{RMS}^{TC} with varying standard deviation of the G_0 values of the MTJ. . . . 56

5.1 Accuracy versus the number of p-bits with stuck-at-1 faults (chosen randomly) in the p-bit network. 60

5.2 Identification of the critical p-bits based on mutual information. The criticality of a p-bit is encoded by color, where yellow represents the critical most p-bit. There are 784 input p-bits in the network, however, for illustration, a 500 p-bit network is shown here. 63
5.3 Accuracy versus the number of p-bits with stuck-at-1 faults in the p-bit network: when critical p-bits are chosen (red) and when randomly chosen (black). 64
5.4 (a) An example Bayesian Network (BN) is shown with three nodes Sprinkler (S), Rain (R), and Wet-Grass (W). There are directed connections from the parent (S and R) to the child node (W) according to CPTs. A selector block maps the probability (P) to h_S, h_R, and h_W as shown in table in (b). (b) Probability to h_i mapping to determine the required h_i that achieves P. (c) The corresponding output response for the p-bits S, R, and W. (d) The corresponding probability distribution for all the p-bits. 65
5.5 Proposed framework for fault detection (a) testable p-bit (b) internal details of scan cell (c) connecting a testable p-bit with other p-bits. 68
5.6 Validation of proposed testable p-bit circuit using SPICE simulations. The m_a values are shown in terms of the probability of occurrence of state $1(0.4V)$ and $-1(-0.4V)$. 69
5.7 Connections of two testable p-bits a and b. 71
5.8 Validation of testable p-bits a and b shown in Fig. 5.7 using SPICE simulations. 71
5.9 Comparison of accuracy obtained for a p-bit network with faulty p-bits and without faulty p-bits (assumed to be removed using isolatable p-bits). The inset shows the complete plot on a logarithmic scale. 73
5.10 Proposed isolatable p-bits and their connections. 73
5.11 Validation of isolatable p-bits shown in Fig. 5.10 using SPICE simulations. The p-bit a is identified to be faulty during the test mode. 74
5.12 SPICE simulation results demonstrating isolation of the faulty p-bit a. . . 75
5.13 Proposed fault-tolerant p-bit. 76
5.14 Validation of fault-tolerant p-bit using SPICE simulations. 77
5.15 Accuracy versus number of fault-tolerant p-bits. The values (M, N) indicate that there are M fault-tolerant p-bits and randomly-chosen N p-bits with stuck-at-1 fault. 79

6.1 (a) Digit image set TT, $violet$ and $yellow$ color represents pixel values -1 and 1 (b) Partial images with 6 out of 15 p-bits clamped to a unique pattern for every digit. $Green$ color represents the unclamped p-bits. 87
6.2 No. of removed $Connections_J_{ij}$ bounded by the E_{RMS} value. Maximum number of removed weights for every method is shown in $green$ color. . . 94
6.3 Image completion: Colormap of the p-bit outputs: (a,b) Fully connected p-bit network. Sparsely connected p-bit network realized using (c,d) step-wise (SW) method, (e,f) binary search (BS) method, and (g,h) information gain (IG) method. 96

List of Tables

2.1 Comparison of bits, p-bits, and qubits 9
2.2 Parameters used for p-bit simulations 15

3.1 Material parameters for different LBM 30

4.1 Probabilities for a non-ideal p-bit obtained using analytical model and simulation (R=5) 38
4.2 Probabilities for a non-ideal p-bit obtained using analytical model and simulation (R=5) 39
4.3 Parameters used in SPICE simulations of p-bits 40
4.4 E_{RMS} for ideal and non-ideal p-bit based image completion models 50

5.1 Impact on probability distribution due to stuck-at fault in various p-bits in the given BN 67
5.2 Modes of operation of testable/fault-tolerant p-bit system 69
5.3 Comparison of area overhead for testable p-bits 78
5.4 Comparison of delay for proposed p-bits in functional mode 80
5.5 Comparison of Power dissipation in the proposed p-bits 82

6.1 Input parameters for Algo. 3 85
6.2 Comparison of the order of complexity of sparsification methods . . 94

List of Algorithms

1 Compute_criticality: Determines criticality score of all p-bits in a network . . . 62
2 Get_Neighbors: Determines common neighbours . . . 63
3 Training probabilistic computing network: TRAIN_PC . . . 86
4 Collect sample from probabilistic computing network: PC_network 87
5 Sparsely connected p-bit network: Step-Wise (SW) method . . . 89
6 Sparsely connected p-bit network - Binary Search (BS) method . . 90
7 Sparsely connected p-bit network: Information gain (IG) method . . 93

List of Abbreviations

AI Artificial Intelligence

BN Bayesian Network

CMOS Complementary Metal Oxide Semiconductor

CPT Conditional Probability Table

DNN Deep Neural Networks

FinFET Fin Field-Effect Transistor

GAA Gate All-Around

GSHE Giant Spin Hall Effect

HDL Hardware Description Language

IMA In-Plane Magnetic Anisotropy

IoT Internet-of-Things

IRDS International Roadmap for Devices and Systems

KL Kullback-Leibler

LBM Low Barrier Magnet

LFSR Linear Feedback Shift Register

Max-SAT Maximum Satisfiability

MNIST Modified National Institute of Standards and Technology

MRAM Magnetoresistive Random-Access Memory

NM Nanomagnet

PDF Probability Density Function

PMA Perpendicular Magnetic Anisotropy

SOT Spin-Orbit Torque

SPICE Simulation Program with Integrated Circuit Emphasis

Chapter 1

Introduction

1.1 Motivation

For over 60 years, transistor scaling guided by Moore's Law has driven sustained growth in the semiconductor industry. As transistor dimensions decreased, a greater number of transistors could be fabricated on a single chip, enabling more functionality per chip. However, the continued downscaling of transistors has slowed down due to several critical challenges at advanced process nodes. These challenges have been partially addressed through the introduction of high-k gate dielectrics (for example, hafnium oxide) and strained silicon technologies [6]. Despite using advanced architectures like Fin Field-Effect Transistor (FinFET) and Gate All-Around (GAA) transistors [7], current semiconductor process technologies face fundamental limitations in reducing device dimensions.

It is observed that the deterministic computing with Von Neumann architecture is very efficient in arithmetic and bit operations. However, it can be more energy-efficient in areas where it is difficult to specify precise logic or rules, or in problems that are known to be computationally demanding. Pattern recognition, language models in our brains, and the optimization problems (such as protein folding, chemical reactions) that find possible answers in a large number of cases are some of the examples. These problems are solved in nature with a very small amount of energy. Hence, alternative computing paradigms with fundamentally improved energy efficiency are being explored. Among the most promising are neuromorphic, quantum, and probabilistic computing.

Moreover, existing computational paradigms are considered inadequate to meet the demands of future applications, including Artificial Intelligence (AI), edge-AI, big data analytics, autonomous systems, exascale computing, and the Internet-of-Things (IoT) [8]. To this end, the International Roadmap for Devices and Systems (IRDS) projected two technological paradigms: 1) extending the functionality of Complementary

Metal Oxide Semiconductor (CMOS) circuits via the integration with other technologies (also known as 'More Moore'), and 2) invention of new computational paradigms (also known as 'Beyond CMOS'). Beyond-CMOS technologies have shown promising improvements in terms of area and energy efficiency by incorporating more efficient and compatible devices, circuits and architectures needed for the new era of computing [9]. The improvement in the performance of emerging Beyond-CMOS technologies is possible because of the tailored devices, circuit design and architectures for different applications. Additionally, there is a focus on domain-specific hardware and architecture. The idea is to design application-targeted hardware and combine multiple integrated circuits to realize an efficient system. This approach of domain-specific computing can provide the best performance.

There is a class of optimization problems in which the approximate solutions are acceptable such as simulated annealing (SA), parallel tempering (PT), and Markov Chain Monte Carlo (MCMC), etc. Taking inspiration from nature, one of the attempts in solving these problems on hardware is to employ stochastic computing. The idea of including stochasticity in computational elements was originally proposed by John von Neumann's seminal work on probabilistic logic [10]. However, incorporating stochasticity to solve problems was first introduced in the 1960s for logic circuit design [11], [12]. The state-of-the-art implementation at that time incorporated an external source of stochasticity using a pseudo-random number generator. Therefore, [11] also highlighted that existing hardware implementations gave promising results, but they are not low in cost, and therefore can not be commercialized.

Two decades later, Richard Feynman in his seminal paper redefined this idea of domain-specific computing, and articulated that the nature of the problem to be solved should define the underlying computational paradigm [13]. Apart from the advancement of quantum computation, the same principle can be applied to probabilistic computing. The basic building block of probabilistic computing, a probabilistic bit (p-bit), has deterministic states similar to classical bits, but fluctuates between them rather than reaching a stable value [1]. The fluctuating characteristic of a p-bit is the source of stochasticity in a probabilistic computing system. A truly remarkable feat is the design of hardware implementation of p-bits using Low Barrier Magnet (LBM), with the true

randomness originating from the physical phenomenon, for example, thermal fluctuation in the devices [14], [15]. However, in addition to the hardware implementation of a p-bit, it is required to analyze a device-to-system implementation of a probabilistic computing system. At the device level, the rate of fluctuation of the LBM governs the speed of operation of the p-bit system, which is a crucial figure of merit. To this end, in this thesis, the impact of various material parameters on the output response of a p-bit is analyzed. Additionally, there are fabrication-related implications on the performance of the probabilistic computing system. Hence, a mathematical model is proposed to analyze the impact of process-induced variations on the performance of the system implemented for a given application. Apart from process-induced variations, another source of practical challenge is the defects and faults in a p-bit. Subsequently, a framework is proposed to implement a fault-tolerant p-bit system. Additionally, at the system-level, a methodology is proposed to realize a sparse probabilistic computing network to reduce the area and power overhead. In summary, this work comprehensively investigates probabilistic computing, spanning device materials, circuit-level implementations, and system-level design considerations. At the system-level, the focus is on understanding non-idealities and enabling fault tolerance for practical applications.

1.2 Objectives

The goal of this work is to analyze the full stack of probabilistic computing. This entails gaining insights into the state-of-the-art hardware required to implement the basic building block of the probabilistic computing known as p-bits. Furthermore, the focus is on optimizing the design of p-bits and enhancing their figures of merit at every level. Specifically, the objectives of this work are as follows.

- To investigate the impact of material parameters on the performance of p-bits, and to propose a methodology to optimize the figures of merit at the device-level that impacts the performance at the circuit- and system-level.
- To analyze the performance of a realistic p-bit with process-induced variations using an analytical model and a Simulation Program with Integrated Circuit Emphasis (SPICE) model, and to propose a methodology for deriving application-specific constraints for non-idealities in a p-bit to achieve desired figures of merit.

- To develop a fault-tolerant design framework for p-bit systems and to minimise accuracy loss and area overhead while achieving desired fault-tolerance levels in p-bit systems.
- To propose a methodology to identify the relative importance of connections in a p-bit network and employ this information to sparsify a fully connected probabilistic network to improve its figures of merit.

1.3 Contributions

The major contributions of this work are summarized below.

- A one transistor and one magnetic tunneling junction (1T-1MTJ) -based p-bit is employed to analyze the dependency of flips per second (fps) on various material parameters. The fps of an LBM is dependent on the energy barrier E_b. The E_b is dependent on the material parameters of the LBM, namely uniaxial anisotropy H_k, saturation magnetization M_s, and the volume Ω of the LBM. A methodology to identify the realistic range of material parameters that optimize fps is proposed. The LBMs with different material parameters are integrated into the 1T-1MTJ spin-transfer-torque (STT) magnetic random access memory (MRAM) based p-bit architecture. The impact of deviation from the nominal range of material parameters, potentially due to process-induced variations, on the output response of a p-bit is investigated.
- The influence of non-idealities in a p-bit and a p-bit network is analyzed. A mathematical model for a p-bit accounting for its non-idealities is derived and shown to agree well with the SPICE simulations. The robustness of a p-bit network, allowing it to retain functionality despite non-idealities, is demonstrated. The averaging and cancelling effects of various random non-idealities are studied. A statistical method to account for non-idealities is proposed, which can be employed to derive application-specific realistic design constraints on the p-bit implementation. The impact of variations on the accuracy of the probabilistic computing network is demonstrated using SPICE simulations.
- The impact of faults on p-bit systems is examined using a subset of the Modified

National Institute of Standards and Technology (MNIST) handwritten dataset. A methodology based on mutual information is proposed to evaluate the criticality of individual p-bits. A testable p-bit is proposed to detect faults and its functionality is validated through SPICE simulations. Additionally, isolatable and fault-tolerant p-bits are proposed to tackle accuracy loss caused by stuck-at faults. A framework is proposed to demonstrate the development of fault-tolerant capabilities in p-bit systems implemented using stochastic elements susceptible to stuck-at faults. Furthermore, the trade-off between various figures of merit is shown, which could be employed to achieve desirable robustness for a system with a minimal overhead.

- A methodology is proposed to realize a sparse probabilistic computing network by computing the mutual information to predict the strength of interconnections between the p-bits. The p-bit network is trained to identify the images of digits 0 to 9 of size 5×3 pixels. Subsequently, the invertibility property of probabilistic computing is employed to recover a full image from a given partial image. The sparsely connected p-bit network to implement the image completion task is derived using three methods. The first two methods are conventional: Step-Wise (SW) removal of weights and the Binary Search (BS) method. However, both are iterative and, therefore, computationally expensive. An alternative method is proposed using Information Gain (IG) based on information theory. The results shows that the sparsely connected network successfully completes the partial images in all three cases.

1.4 List of Publications

1. **A. Haroon** and S. Saurabh, "Image Completion using a Sparse Probabilistic Spin Logic Network," in 35th *International Conference on VLSI Design (VLSID)*, 2022, pp. 281–286, doi: 10.1109/VLSID2022.2022.00061

2. **A. Haroon**, R. K. Ghosh, and S. Saurabh, "Implementation of Probabilistic Bits (Pbits) using Low Barrier Magnets: Investigation and Analysis," in 36th *International Conference on VLSI Design (VLSID)*, 2023, pp. 307-312, doi: 10.1109/VLSID57277.2023.00069.

3. **A. Haroon**, R. K. Ghosh, and S. Saurabh, “Impact of Non-Idealities on the Behavior of Probabilistic Computing: Theoretical Investigation and Analysis,” in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 71, no. 12, pp. 6279–6291, 2024, doi: 10.1109/TCSI.2024.3461770.

4. **A. Haroon** and S. Saurabh, “Fault-Tolerant Design Framework for Probabilistic-Bit (P-Bit) Systems: Proposal and Analysis,” in *IEEE Transactions on Circuits and Systems I: Regular Papers*, doi: 10.1109/TCSI.2025.3569769.

1.5 Organization

The rest of this thesis is organized as follows:

- **Chapter 2 Background and related work:** The background and related work relevant to this thesis are presented. The chapter begins with a historical overview and a discussion of the technological and computational demands that have led to the development of probabilistic computing. An overview of the recent advancements in the field is provided. Finally, the application space of probabilistic computing is discussed, emphasizing its relevance across emerging domains.

- **Chapter 3 Implementation of Probabilistic bits using Low Barrier Magnets:** An approach to identify optimal design parameters for LBM in 1T-1MTJ-based p-bit implementations is proposed. This study investigates a realistic range of the material parameters that can enhance p-bit performance.

- **Chapter 4 Impact of Non-Idealities on the Behavior of Probabilistic Computing:** The non-idealities in a p-bit arising from process- and environment-induced variations, as well as aging of circuit components, are investigated. The impact of these non-idealities on the functionality and robustness of a p-bit network is demonstrated through an analytical model and numerical simulations. Furthermore, Monte Carlo simulations are performed using a 1T-1MTJ-based p-bit implementation. It is shown that the results from numerical simulations and circuit-level simulations are in agreement with the predictions of the proposed analytical model. Therefore, the analysis presented in this chapter can be utilized to de-

rive application-specific constraints on the non-idealities, offering critical design criteria for future p-bit implementations.

- **Chapter 5 Fault-Tolerant Design Framework for Probabilistic-Bit (P-Bit) Systems:** The impact of faults in the p-bit systems implemented with stochastic nanomagnets is evaluated using the MNIST dataset. A mutual information-based criticality score is proposed to identify essential p-bits. The testable, isolatable, and fault-tolerant p-bit designs were introduced and validated through SPICE simulations using $14nm$ FinFET technology. By selectively replacing critical p-bits, fault resilience is enhanced with minimal overhead. Additionally, a performance evaluation of the proposed p-bits is performed.

- **Chapter 6 Image completion using Sparse Probabilistic Computing network:** A methodology to optimize the p-bit network is proposed. A case study to perform image completion of the partial digit images is presented. A fully connected p-bit network effectively recovers the missing pixels. Additionally, an area-efficient sparsely connected p-bit network is introduced using weight pruning and mutual information based on the perspective of information theory. Using the information gain method, 42% of the interconnections are removed. And, lastly, the performance comparison of the fully connected and the sparsely connected p-bit network is presented.

- **Chapter 7 Conclusion and Future Work :** The thesis is concluded and potential directions for future research are discussed.

Chapter 2

Background and related work

In this chapter, the basic concepts of probabilistic computing is discussed in detail alongside the current state-of-the-art implementation techniques. Additionally, the application space of probabilistic computing is discussed.

The presence of noise in computational systems was first recognized by John von Neumann, who investigated the correctness of implemented logic in the presence of noisy electrical components [10]. In the mid-1900s, implementation of emerging applications such as chemical plant control systems, aerospace navigation controls, and other large-scale, complex systems posed challenges that conventional digital, analog, or hybrid computing systems could not adequately address. A significant obstacle at that time was the limited computational speed and the need for efficient algorithmic implementation on hardware, which often required analog components. These analog systems introduced their own challenges in terms of cost, physical size, reliability, and integration with digital logic.

The introduction of integrated circuits (ICs) provided a partial solution to these limitations. Nevertheless, it became evident that while stochastic computing held promise for solving certain classes of problems, it was not cost-effective for commercial scale deployment [16]. Despite these hurdles, pioneers such as Ted Poppelbaum, Sergio Ribeiro, and Brian R. Gaines continued to advance the theoretical foundations of stochastic computing, viewing stochasticity not as a hindrance, but as a viable computational paradigm [11], [17], [18].

2.1 Introduction to probabilistic computing

The stochastic computational paradigm continued to evolve, with problems being addressed through the integration of pseudo-random noise sources. A common approach involved the use of an Linear Feedback Shift Register (LFSR) to generate stochastic

bitstreams, effectively serving as a source of randomness.

In 1982, Richard Feynman, during the seminal keynote address, hypothesized that the nature of a computational problem should define the underlying computational paradigm. He emphasized that to efficiently simulate quantum mechanical systems, the computing architecture itself must be inherently quantum [13]. This insight laid the foundation for the field of quantum computing. Decades later, a research group led by Dr. Kerem Y. Camsari and Dr. Supriyo Datta at Purdue University developed a tunable true-randomness based fundamental building block specifically designed for probabilistic computing [1]. This building block is called the *probabilistic bit*, or *p-bit*, and it marked a crucial step in probabilistic computing. In this thesis, the terms *p-bit* and *probabilistic computing* will be used to describe and explore the principles and operations of this emerging computational framework.

The p-bit is characterized by deterministic logic levels, the states 0 and 1, similar to those of binary digits or bits. However, unlike classical bits, the state of a p-bit fluctuates between these two logic levels, offering a middle ground between classical bits and quantum qubits. This unique ability of probabilistic toggling makes p-bits

Table 2.1: Comparison of bits, p-bits, and qubits

Characteristics	**Bits**	**P-bits**	**Qubits**
State Space	Either 0 or 1	Fluctuate between 0 and 1	Superposition of 0 and 1
Implementation technique (one of them)	CMOS/Stable magnets	Unstable magnets	Single electron spin
Temperature	Room Temperature ($300K$)	Room Temperature ($300K$)	Cryogenic Temperature ($\approx 0K$)
Measurement properties	Readout without state alteration	Readout without state alteration	State altered during readout
Scalability	Mature scaling (billions of transistors on-chip) [19]	Emerging technology (Not commercially scalable, $\approx$ 3000 p-bits on CPU) [20]	Emerging technology (scalable upto $\approx$ few hundered qubits [21])
Commercial availability	Ubiquitous	Limited specialized hardware	Limited specialized hardware

highly suitable for stochastic computations and solving optimization problems. Tab. 2.1 presents a comparison of the characteristics of bits, p-bits, and qubits. The working

mechanism and design principles of an ideal p-bit using the behavioural model and the subsequent circuit implementation are examined in the following subsection [1].

2.1.1 Design of a single p-bit

Earlier attempts at designing a building block for applications that require probabilistic computations such as in Bayesian Network (BN) used magnetoelectric device that can be programmed using magnetic field [22]. The idea is to design a circuit that can perform computation on discrete values, a representation of probabilities in BN. This technique uses non-volatile devices to realize energy-efficient implementation by concepts similar to in-memory computing [23]. Therefore, the device could perform probabilistic computations, but they were still far from being an inherently probabilistic device.

Similar to deterministic logic circuits, where the key properties of gain and directionality allow billions of devices to be interconnected, the p-bits must have an input(write) - output(read) (W-R) isolation and a transistor-like gain [24]. The behavioural modelling and the subsequent circuit implementation using SPICE models are demonstrated in the next part of this section.

Behavioural model of a p-bit

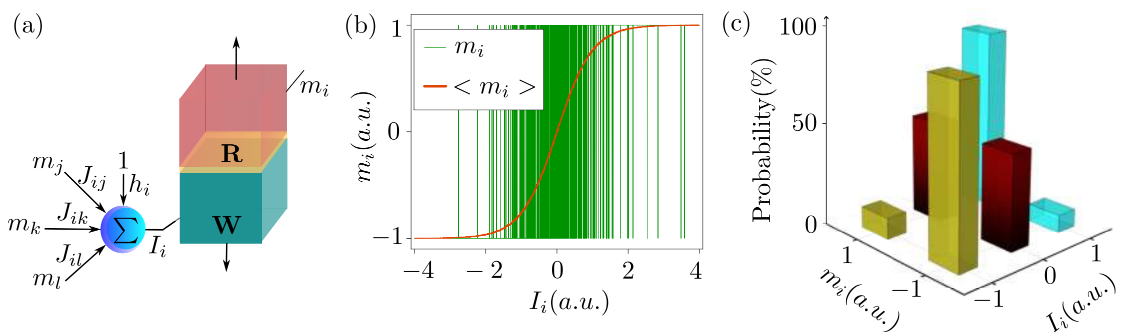


Fig. 2.1: (a) A generic model of a p-bit featuring input-output terminals denoted as W-R. (b) Illustrates the variation of m_i with I_i ranging from -4 to $+4$. The average of m_i, denoted as $< m_i >$, is plotted for $10,000$ samples of m_i at each I_i value. (c) Probability distributions of m_i for $\beta = 1$ and $I_i = \{-1, 0, +1\}$. [1].

The behaviour of an ideal p-bit is described by the following equation [1]:

$$m_i(t) = \text{sgn}\{\tanh(I_i(t) + \text{rand}(-1, 1)\} \tag{2.1}$$

where m_i is the output of the p-bit and takes on a value of -1 or 1, sgn is the signum function, rand$(-1, 1)$ represents a uniformly distributed random number in the interval $[-1, +1]$, and $I_i(t)$ is the input to the p-bit at time step t, evaluated according to the following equation [1]:

$$I_i(t) = \beta\{h_i + h_i^{fix} + \sum_j J_{ij} m_j(t)\} \tag{2.2}$$

where h_i and h_i^{fix} are the local and fixed bias input to the ith p-bit. A generic model of a p-bit is shown in Fig. 2.1(a). To realize a meaningful functionality, a system of p-bits is interconnected with synapse-like weighted connections, J_{ij}, where the $(i, j)^{th}$ element of the weight matrix J represents the influence of the behavior of p-bit j on p-bit i, and β represents the global strength of interconnections between the p-bits [1]. The p-bits in the p-bit network are updated sequentially. Therefore, the convergence of the p-bit network depends on the update of all the p-bits in the network, and on the sufficient number of samples that need to be collected from the p-bit network to get the valid states [25].

The p-bit's output response m_i for changing input I_i is depicted in Fig. 2.1(b). For $I_i = 0$, the p-bit behaves like a true random number generator (TRNG). However, its randomness can be tuned by providing a non-zero I_i, as described by Eq. 2.1. With the increasing $|I_i|$, the effect of randomness diminishes, and the output tends to be deterministic. A possible probability distribution for the two states is illustrated in Fig. 2.1(c) for three values of $I_i = \{-1, 0, 1\}$ [1]. In the next part of this section, different circuit implementation techniques for p-bit are demonstrated.

Hardware implementation of a p-bit

Low barrier magnet (LBM) inherently provides stochasticity through thermal fluctuations, enabling energy-efficient probabilistic switching without requiring complex additional circuitry. In contrast, CMOS implementations rely on pseudo-random number generators or noisy analog circuits, microcontrollers, or microprocessors, which increases the design complexity and power consumption [26], [27], [28].

The spintronic devices that are widely used in memory applications can be modi-

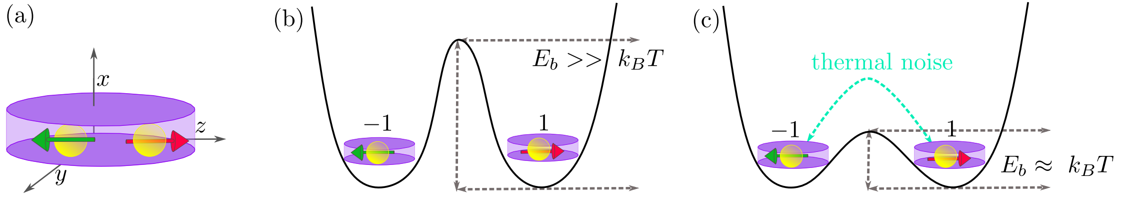


Fig. 2.2: (a) The schematic of an LBM. The energy barrier, E_b for (b) $E_b >> k_BT$ (c) $E_b \approx k_BT$. For $E_b \approx k_BT$, the thermal fluctuations are sufficient to change the state of the LBM from $+1$ to -1 or vice versa.

fied to be used as a p-bit. In memories, the stability of a logic state is dependent on the energy barrier (E_b) of the nanomagnet, as shown in Fig. 2.2. For a stable nanomagnet, $E_b \approx$ 40-60 k_BT, as shown in Fig. 2.2(b). For a p-bit implementation, the magnets are designed such that the $E_b \approx k_BT$, as shown in Fig. 2.2(c) [1], [5]. The E_b in turn controls the correlation time τ_c of the nanomagnet according to the equation $\tau_c = \tau_0 e^{E_b/k_BT}$ where k_B is the Boltzmann constant, τ_0 is a constant, and T is the temperature. The inverse of τ_c, which is flips per second (fps), is defined as the rate of fluctuation in a nanomagnet. In probabilistic computing, the time-to-solution is dependent on the fps [29]. A fast fluctuating p-bit is able to traverse the solution space quickly. Based on these observations, a nanomagnet-based implementation of a p-bit is proposed in [5]. A SPICE model-based proof of concept for the design of LBM-based implementation of a p-bit and its successful integration to realize a BN is demonstrated in [24].

A p-bit implementation based on an elliptical nanomagnet is presented in [30]. An experiment-based comparison of the performance of circular and elliptical nanomagnets is demonstrated in [15]. It was found that the circular nanomagnet was faster and energy efficient than the elliptical nanomagnet. Recently, a hardware implementation of perpendicular spin-transfer-torque (p-STT) based p-bit with its corresponding peripheral circuitry is reported in [31]. However, in circular nanomagnets, an In-Plane Magnetic Anisotropy (IMA) magnet has better performance than a Perpendicular Magnetic Anisotropy (PMA) magnet [5].

Fig. 2.3 illustrates the p-bit designs incorporating the IMA-LBM-based structures. Fig. 2.3(a) shows the 1T-1MTJ STT-MRAM, which operates as a series-resistance-controlled device where the input voltage modulates the resistance of the transistor R_{NMOS}. An alternative implementation involves the use of a one resistor one-MTJ (1R-1MTJ) Spin-Orbit Torque (SOT) Magnetoresistive Random-Access Memory (MRAM)-

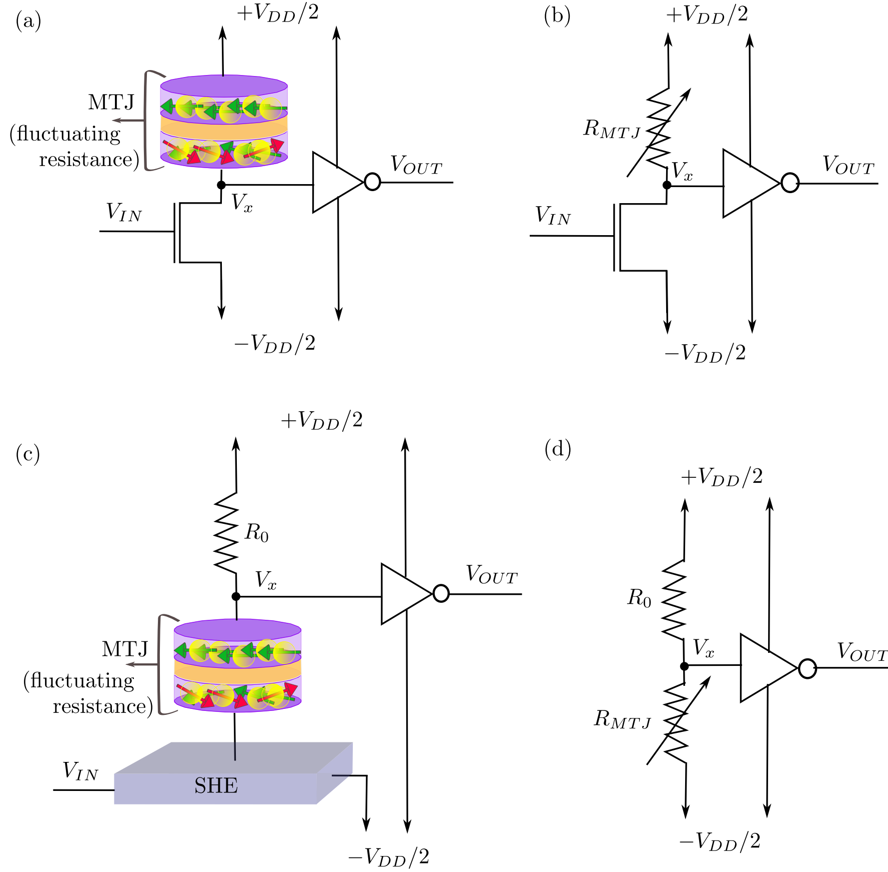


Fig. 2.3: Two types of p-bit design using stochastic MTJ. (a) 1T-1MTJ STT MRAM based design of a p-bit. (b) Equivalent circuit of 1T-1MTJ STT MRAM based p-bit [1]. (c) Spin-Orbit Torque (SOT) MRAM based implementation of the p-bit. (d) Equivalent circuit of Spin-Orbit Torque (SOT) MRAM based p-bit [2].

based p-bit design. In the 1R-1MTJ SOT-MRAM design, the input spin current generated by the Giant Spin Hall Effect (GSHE) layer is used to pin the free-layer magnetization of the MTJ, as illustrated in Fig. 2.3(c). The corresponding equivalent circuit implementation is presented in Fig. 2.3(d) [32], [33], [34]. In addition to fabrication challenges such as interlayer oxide defects, lattice mismatch between the interface layers, ensuring thermal and structural stability faced by the integration of MTJ with the CMOS technology, fabrication of SOT-MRAM encounters additional challenges as compared to STT-MRAM [35]. The SHE layer in SOT-MRAM is ultra-thin, therefore, the etching process to fabricate the MTJ-stack on top of the SHE layer needs to stop precisely at the edge of the SHE layer. Any minor deviations in the device dimensions can affect the performance and reliability of the SOT MRAM device [36]. In terms of performance characteristics, the two device designs are compared based on the steady-state response, correlation time (τ_c), response time, and the power consumption [32]. Since the IMA

magnets require high current values to pin the LBM in any particular direction, the unpinned LBM contributes to the stochastic behavior of the p-bit. Similarly, both devices can be simplified as a voltage divider, and therefore have a comparable power dissipation. Additionally, the correlation time of the LBM is determined by the magnets' parameters. The key difference lies in the response time, which is the time it takes for the p-bit output to respond to the input stimuli. For the STT-MRAM-based design, the response time depends on the transistor physics and for the SOT-MRAM-based design, it depends on the magnet physics [32], [37].

The 1T-1MTJ-based implementation of p-bit is considered a milestone in probabilistic computing because the STT MRAM is a widely researched architecture and a commercialized technology [14], [38]. Additionally, the 1T-1MTJ of STT-MRAM cell is integrable with the existing CMOS technology [1], [2], [32], [37]. In this work, the circuit implementation of the p-bit adopts the 1T-1MTJ STT-MRAM-based design. The circuit parameters for implementing a p-bit shown in Fig. 2.3(b) are listed in Tab. 2.2 [2]. The conductance of the MTJ is described by the following equation:

$$G_{MTJ} = G_0 \left(1 + m_z \frac{TMR}{2 + TMR} \right) \tag{2.3}$$

where $G_0 = (G_P + G_{AP})/2$ is the average conductance of the MTJ, $TMR = (G_P - G_{AP})/G_{AP}$ is the tunnel magnetoresistance, m_z is the magnetization of the magnet, and G_P and G_{AP} are the parallel ($m_z = 1$) and antiparallel ($m_z = -1$) conductance of the MTJ [2]. The circuit in Fig. 2.3(b) is effectively a voltage divider with voltage V_x given as follows:

$$V_x = \frac{V_{DD}}{2} \left(\frac{R_{NMOS} - R_{MTJ}}{R_{NMOS} + R_{MTJ}} \right) \tag{2.4}$$

where V_{DD} is the supply voltage, R_{NMOS} and R_{MTJ} are the resistances of the transistor and the MTJ, respectively. Additionally, an inverter is added to drive the fluctuations at V_x towards the high and low supply voltages [2].

The selection of G_0, TMR, sizing of the $NMOS$, and the inverter are crucial design criteria. The value of R_{NMOS} and R_{MTJ} should be such that in the absence of stimulus to a p-bit, the voltage at the node x is sufficient to drive a small-sized inverter. The

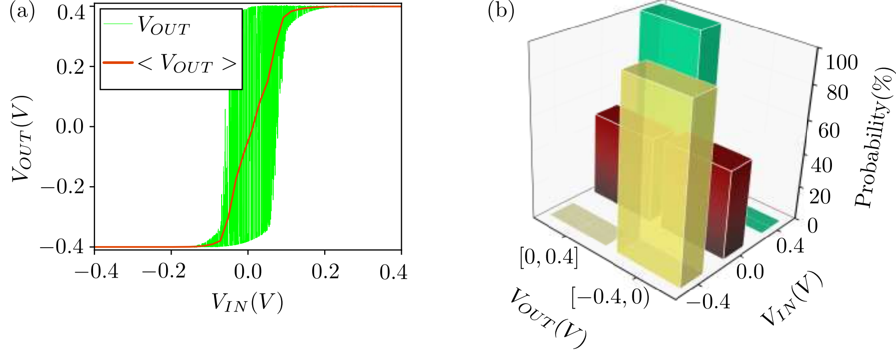


Fig. 2.4: (a) Input-output characteristics of the modified 1T-1MTJ-based p-bit with V_{IN} varying from $-V_{DD}/2$ to $+V_{DD}/2$. (b) Probability distributions of V_{OUT} for three values of $V_{IN} = \{-V_{DD}/2, 0, V_{DD}/2\}$, shown in yellow, red, and green, respectively ($V_{DD} = 0.8\,V$ for the 14nm HP-FinFET technology) [2].

Table 2.2: Parameters used for p-bit simulations

Parameters	**Value**
Gilbert damping coefficient (α)	0.01 [2]
Volume of the free layer (Ω)	$4.16 \times 10^{-19}\,cc$
Saturation magnetization (M_s)	$1200\,emu/cc$
In-plane uniaxial anisotropy field (H_k)	$100\,Oe$
Temperature (T)	$27\,°C$
Tunnel magnetoresistance (TMR)	200%
Average conductance of MTJ (G_0)	$46\,\mu S$
Transistor technology models	14-nm HP-FinFET [40]
Time step for transient simulation (Δt)	$1ps$ [2]

output response of a p-bit obtained using SPICE simulation is shown in Fig. 2.4(a). The green trace shows the instantaneous value and the red trace shows the average response over $100ns$ for $V_{IN} \in (-0.4V, 0.4V)$. The average output response can be approximated according to the following equation which has a similar form as Eq. 2.1 [39]:

$$m_i(t) = C_1\left(\text{sgn}\left\{\text{rand}(-1,1) + \tanh\left(\frac{I_i(t)}{C_2}\right)\right\}\right) \tag{2.5}$$

where C_1 is the supply voltage and C_2 denotes the sigmoidal range of the p-bit and is primarily determined by the transistor characteristics [2], [39]. Fig. 2.4(b) shows the probability distribution derived from the output samples binned into two states as: *state* -1, if $V_{OUT} \in [-0.4V, 0V)$ and *state* 1, if $V_{OUT} \in [0V, 0.4V]$.

Probabilistic computing has demonstrated promising performance in solving NP-hard problems such as integer factorization, spurring exploration of p-bit implementa-

tions using existing devices. However, significant modifications are required to harness true randomness [14], [38]. Among the notable advancements to realize a p-bit, single ferroelectric field effect transistor (FeFET) [41], one-Transistor-one-Resistor (1T1R) Resistance Random Access Memory (RRAM) [42] and $Pr_xCa_{1-x}MnO_3$ based probabilistic RRAM [43], $Cu_{0.1}Te_{0.9}/HfO_2/Pt$ (CTHP) diffusive memristor [44], perovskite nickelates [45], single photon avalanche diodes (SPAD) CMOS based p-bits [46], [47], magnetoelectric RAM (MELRAM) [48], superconductor single flux quantum (SFQ) circuit-based superconductor random number generator (SRNG) [49], and a variation-tolerant p-bit implemented using stochastic MTJs, resistors, capacitor, and a comparator [50] are reported in the literature.

Non-ideal behaviour in a p-bit

There has been extensive research into material synthesis and alternative architectures to enhance the efficiency and functionality of probabilistic computing networks [51], [52], [50], [53]. However, one of the major challenges in any p-bit implementation is the impact of process-induced variations and defects during fabrication [54], [55], [56], [57]. The figures of merit for a p-bit depend on the material and electrical properties of the transistors and the MTJ [58]. The critical material properties of the MTJ impacting the p-bit design include the uniaxial anisotropy (H_k), saturation magnetization M_s, tunnel magneto-resistance ratio (TMR), and the oxide thickness (t_{ox}) of the insulating layer. However, the impact of process-induced variations in the material parameters of the MTJ on the performance of a p-bit remains largely unexplored. There have been a few efforts in this regard, such as in-situ training for machine learning (ML) applications [55], adaptation through the applied stimulus to the p-bits [59], or the variation-tolerant circuits at the cost of extra circuit components [60]. An exhaustive analysis of the strong sensitivity of the behavior of the p-bit to slight geometric variations is demonstrated in [56]. An exhaustive analysis of the effect of process- and reliability-induced variations on different components of a p-bit, namely the MTJ, NMOS, and the inverter, is discussed in [61]. An alternate approach to train around the variations in the p-bits is discussed in [55].

It is worth pointing out that probabilistic computing depends on the time-dependent

random fluctuations in the computing element (such as thermal noise induced randomness in an LBM). This type of randomness is desirable and essential for probabilistic computing. The probabilistic circuit is designed with some nominal parameters, assuming that the computing element will have the above time-dependent randomness without bias. If the fabricated circuit also has nominal values of circuit parameters, the probabilistic circuit is expected to produce the desired results. However, during fabrication, the circuit parameters, such as the width/length of transistors, oxide thickness, threshold voltage, etc., can move away from their nominal values due to process-induced variations. Unlike the ideal time-dependent unbiased randomness in the computing element, these deviations are undesirable and inevitable, can be biased, persist throughout the circuit operation, and potentially disrupt functionality. A detailed discussion of the impact of process-induced variations in the MTJ and the transistors on the performance of the p-bit is reported in [61]. An on-chip training of the Boltzmann machines to counter the device-to-device variations is demonstrated in [55]. Moreover, non-ideal behavior may also arise due to aging-induced reliability issues in the transistors commonly used in various p-bit implementations [61]. Therefore, the impact of such non-idealities on the behavior of a p-bit, as well as on the performance metrics at the system level, such as accuracy, must be investigated. In this thesis, the influence of non-idealities in a p-bit and a p-bit network is systematically investigated. Furthermore, an analytical model is proposed that accounts for the non-idealities in a p-bit. The SPICE simulations using 1T-MTJ-based p-bit implementation demonstrated that the analytical model can capture the non-idealities present in the hardware implementation of a p-bit.

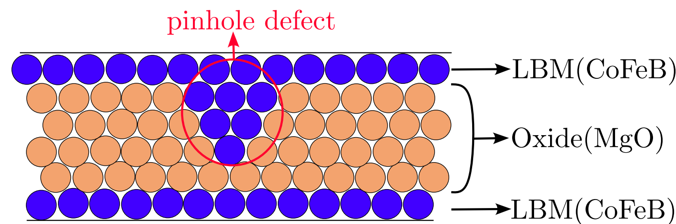


Fig. 2.5: A diagram illustrating the creation of a pinhole [3].

Another challenge that can cause a p-bit to deviate from its expected behavior is the presence of defects and faults in the 1T-1MTJ STT-MRAM design [3], [62], [63], [64], [65]. A defect can occur from patterning proximity effects, line-edge and line-width roughness, polish variations for shallow trench isolation and variations in gate dielectric

in the manufacturing of the transistor in the front-end-of-line (FEOL) phase [54]. With extremely thin gate oxide, a pinhole defect can occur due to rough deposition of oxide layer, as illustrated in Fig. 2.5 [3], [66]. Additionally, stuck-at faults can manifest as open- and short-circuit between various terminals. The open-circuit faults are typically caused by salicidation, incompletely filled vias, or electromigration in the interconnects. Similarly, short-circuit faults may form during the transistor fabrication or the back-end-of-line (BEOL) MTJ integration processes [67]. Recent studies have shown that during the BEOL production phase, defects occur due to pinholes and thickness variations in the insulating layer of the MTJ, MgO/CoFeB (insulating layer/magnetic material layer) interface roughness, re-depositions on MTJ sidewalls, etc [68], [69], [70].

Among the above defects, the pinholes are the most common in the insulating layer of the MTJ. A pinhole defect occurs due to an unoptimized deposition process. The unintended thinning (thickening) of the insulating layer results in metallic shorts (open) in the MTJ. It is observed that localized heating by high current flowing through the pinholes results in the early oxide breakdown of the MTJ [71]. Furthermore, it is reported that the pinholes may grow in size after the circuit has been in use, and may lead to early breakdown of the device [72]. Fabrication steps such as ion beam etching can result in a short in the insulation layer of the MTJ [73], [74]. Additionally, the switching of resistance states in MTJ between low resistance state (parallel) and high resistance state (anti-parallel) depends on the geometry and the interface properties of the insulating layer and the magnetic layer of the MTJ. These dependencies become particularly significant in the nano-scale devices [75], [76]. Another potential cause of a stuck-at fault is when the magnetization of the free layer becomes fixed in one state, leading to the MTJ being stuck in either the parallel or anti-parallel configuration [77], [78].

The above studies suggest that STT-MRAMs are susceptible to various types of defects and faults. As energy-efficient p-bit implementations are derived from STT-MRAM technology, a thorough analysis of the impact of faults in a p-bit system is required. Previous works have reported the impact of process-induced variations on the performance of p-bit systems and have demonstrated remarkable resilience against these variations [55], [56], [59], [61], [79], [80], [81], [82]. Nevertheless, a p-bit can have faults due to defects introduced during fabrication, or while operating due to age-

ing. Therefore, it is crucial to investigate the performance of a p-bit system under these scenarios.

It is worth noting that there are related stochastic counterparts of p-bits, namely the ternary stochastic neuron (TSN) and the analog stochastic neuron (ASN), where the material properties of LBM are modified to achieve the desired output response [83], [84]. The p-bit fluctuates between two discrete logic levels, $\{-1, 1\}$. In comparison, a ternary stochastic neuron (TSN) encodes three discrete states $\{-1, 0, 1\}$, while an analog stochastic neuron (ASN) can take any continuous value within the range $[-1, 1]$. Despite these differences in state representation, p-bits, TSNs, and ASNs leverage inherent stochasticity to compute their output responses. A TSN achieves more data processing per neuron unit by employing three states per neuron, resulting in savings in area, energy consumption, and number of interconnections. However, if a stuck-at fault occurs in a TSN, its impact on the system performance can be more severe because of the greater deviation in the behavior of the faulty TSN from the non-faulty TSN. ASNs, on the other hand, serve as the stochastic counterparts of analog neurons and find applications in time-series prediction using reservoir computing. For an ASN, fault models that are relevant to analog circuits, such as single and multiple parametric fault models, can be more relevant and require thorough investigation. In this thesis, the fault tolerance in a p-bit system is investigated; exploring the impact of faults in TSN- and ASN-based systems remains an important direction for future research.

2.1.2 Design of interconnected network of p-bits

A given Boolean function can be implemented using a p-bit network by appropriately selecting the h and J matrices. Multiple approaches exist for determining these weight matrices. Early methods were based on the Ising Hamiltonian, initially developed for quantum computing [85], [86], as well as principles from Hopfield networks [87]. A numerical method for incorporating hidden p-bits is presented in [1].

Additionally, fundamental logic components such as AND, OR, and XOR gates can be incorporated to realize larger arithmetic units like adders and multipliers, as demonstrated in [1], [88], [89], [90], [91]. A generalized automated design flow from a standard Hardware Description Language (HDL) is also proposed in [92]. Further-

more, machine learning has been extensively implemented using probabilistic computing paradigm, as demonstrated in [55], [93], [94], [95], [96], [97].

A p-bit network has a unique property of *invertibilty*. This property allows the p-bit network to operate in forward mode (input $\rightarrow$ output) and backward mode (output $\rightarrow$ input). This phenomenon occurs because the input and output p-bits are not differentiated in a p-bit network. Also, a p-bit network is an energy-based network, similar to a Boltzmann machine [1], [98]. Therefore, the inherent randomness in a p-bit allows the p-bit network to explore the energy landscape.

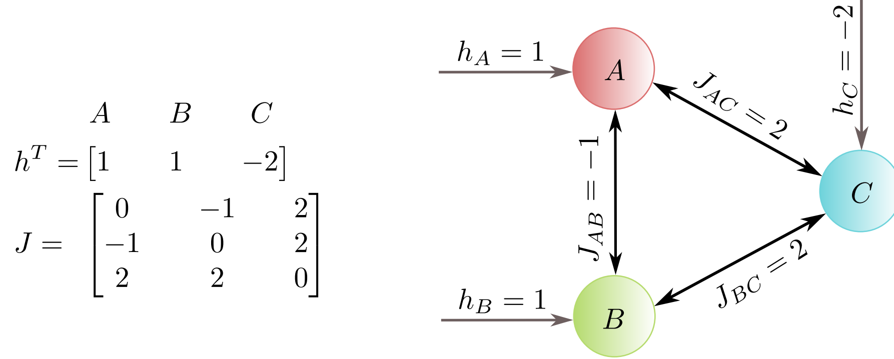


Fig. 2.6: The schematic for two-input AND gate implemented using p-bits. The h and J values represent the weighted connections [1].

As an illustration, a two-input AND gate is implemented using the h and J matrices as shown in Fig. 2.6 [1]. The p-bits A, B, and C in the network are ideal p-bits encapsulated by Eq. 2.1. When the input p-bits (A and B) are fixed to one of the following input combinations $AB = \{(-1,-1),(-1,1),(1,-1),(1,1)\}$, the probability distribution of the output p-bit C gets adjusted to represent the output value $\{-1,-1,-1,1\}$, respectively, delivering the functionality of an AND gate. For all the input combinations, the probability distribution for the p-bits of the AND gate obtained using simulation is shown in Fig. 2.7(a-d). Similarly, when the output p-bit C is fixed to $\{-1,1\}$, the probability distribution shows the dominant contribution of the corresponding valid states i.e., $ABC = \{(-1,-1,-1),(-1,1,-1),(1,-1,-1)\}$ and $\{(1,1,1)\}$, respectively, as shown in Fig. 2.7(e) and (f) [1].

The equivalent circuit implementation using 1T-1MTJ-based p-bits is shown in Fig. 2.8. The design parameters for the p-bits A, B and C are listed in the Tab. 2.2. The scaling parameter A_0, where $A_0 = 1/\beta$, is set to 0.45 to manage the global strength of the interactions. Similar to the behavioural model, the probability distribution for different

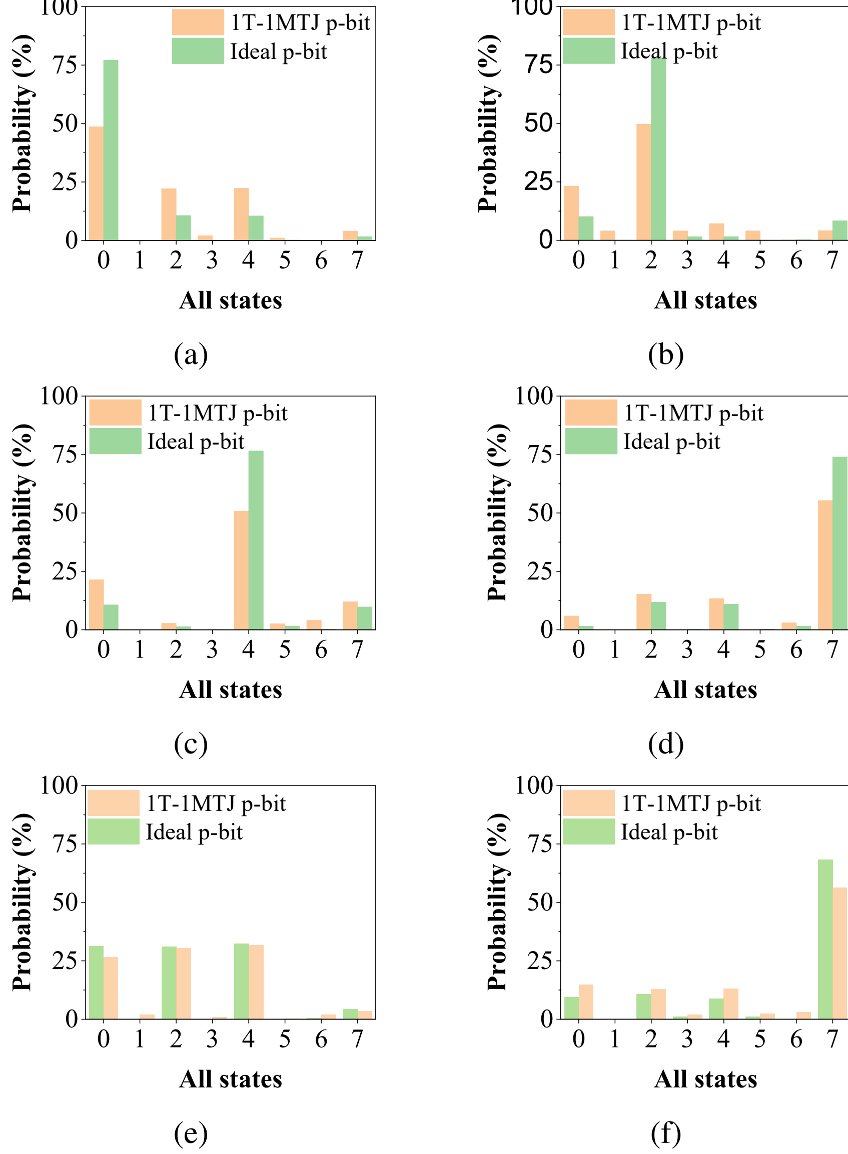


Fig. 2.7: (a-d) The probability distribution for AND gate working in the forward mode i.e., $(A, B) \in \{(-1,-1),(-1,1),(1,-1),(1,1)\}$ and p-bit C is floating. The numbers along the X-axis represent the decimal encoding of the states represented as (ABC). For example, the states $(-1,-1,-1)$, $(-1,-1,1)$, and $(1,1,1)$ are encoded as 0, 1, and 7, respectively. (e,f) The probability distribution for the AND gate working in the backward mode [1].

input patterns, where $(V_{fix}^{A}, V_{fix}^{B}) \in \{(-0.4V,-0.4V),(-0.4V,0.4V),(0.4V,-0.4V),$ $(0.4V,0.4V)\}$ is shown in Fig. 2.7. To realize a floating p-bit, where no stimuli is to be applied, the V_{fix}^{C} is connected to high resistance ($\approx 1M\Omega$). A bias voltage V_{bias} is applied to all the p-bits to emulate the fixed bias h_i, as defined in Eq. 2.2. The probability distributions obtained using circuit simulations, as shown in Fig. 2.7(a-d), exhibit behavior consistent with the behavioral model.

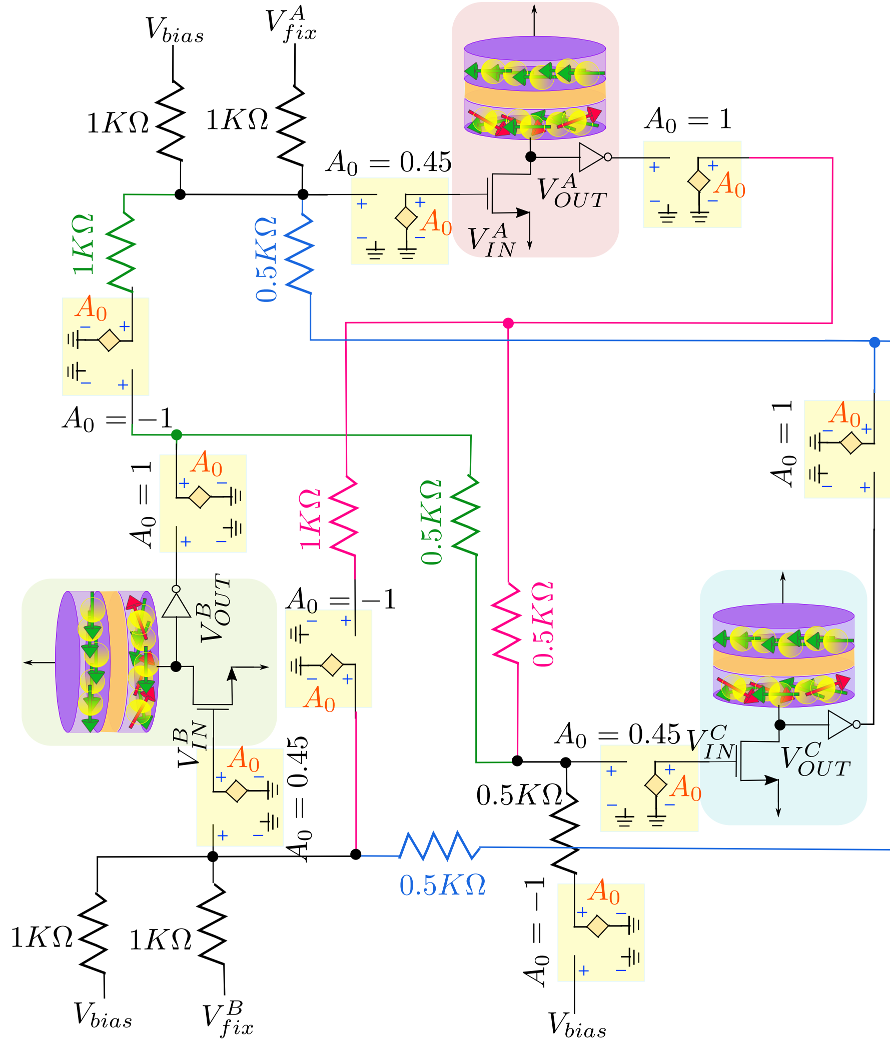


Fig. 2.8: Circuit diagram for a two-input AND gate. The weight values represented by h and J are mapped to resistance values as detailed in [1].

In terms of hardware requirements, the p-bit network can be implemented using three techniques, namely purely CMOS-based implementation, purely stochastic devices based implementation, and the hybrid approach.

In the purely CMOS-based implementation, the p-bit network is implemented on FPGA, where each p-bit requires a PRNG, a look-up table for calculating the hyperbolic tangent function, and a memory on-chip for storing weights with sufficient precision. A recent report compared the area and energy consumption of three p-bit implementations. The first, a 32-bit LFSR-based design, required 5150 transistors and consumed $145fJ$ of energy. The second, a 1T-1MTJ-based p-bit, similar to one considered in this thesis, used four devices (three transistors and one MTJ) and consumed $2fJ$ of energy. The third, a high-quality 32-bit Xoshiro RNG-based design required $11,546$ transistors and

consumed $293 fJ$ of energy.

In the purely stochastic device based implementation, p-bits are realized using MTJs, NMOS transistors, source resistors, and comparators (Operational amplifier). The weighted connections can be realized using the resistors, capacitors, and buffers. For example, to implement a full adder with five p-bits and fifteen weighted interconnections, five sets of MTJs, NMOS transistors, source resistors, and comparators are required. Additionally, for weighted connections, fifteen sets of resistors, capacitors, and buffers are required [55]. The 1T-MTJ-based p-bit consumes about $20\mu W$ [32]. For the overall circuit including the weighted connections and the weight update circuit, the total power consumed is reported $\approx 300\mu W$.

In the hybrid approach, the p-bit network is implemented on the FPGA using the PRNGs, and the stochastic MTJs are used as a clock to the FPGAs. Therefore, even though the next state is pre-determined in PRNG, the unpredictability in the arrival of the clock emulates true randomness. For example, the stochastic devices are implemented using stochastic MTJ, NMOS transistors, resistors, and comparators. The p-bits are implemented on the Kintex UltraScale KU040 FPGA Development Board [60].

Additionally, ≈ 5000 p-bit based deep Boltzmann machine network is implemented on FPGA. Currently, this is the largest p-bit network that could be implemented on the AMD Alveo U250 (Virtex UltraScale+ XCU250) FPGA [25].

In the next section, different applications of the probabilistic computing network are discussed.

2.2 Applications of probabilistic computing

Several proof-of-concept problems, primarily in machine learning applications, have been implemented using probabilistic computing, shown in Fig. 2.9. One notable contribution is the Probabilistic Inference Network Simulator (PIN-Sim), in which p-bits were used to replace the neurons in Restricted Boltzmann Machines (RBM), resulting in improved accuracy compared to traditional methodologies [99]. An approximate generative adversarial network (ApGAN) for accelerating GANs from both algorithm and hardware implementation perspectives is proposed in [100]. Also, Boltzmann machines (BM) are considered a benchmark problem for the MCMC methods. A hybrid quantum-

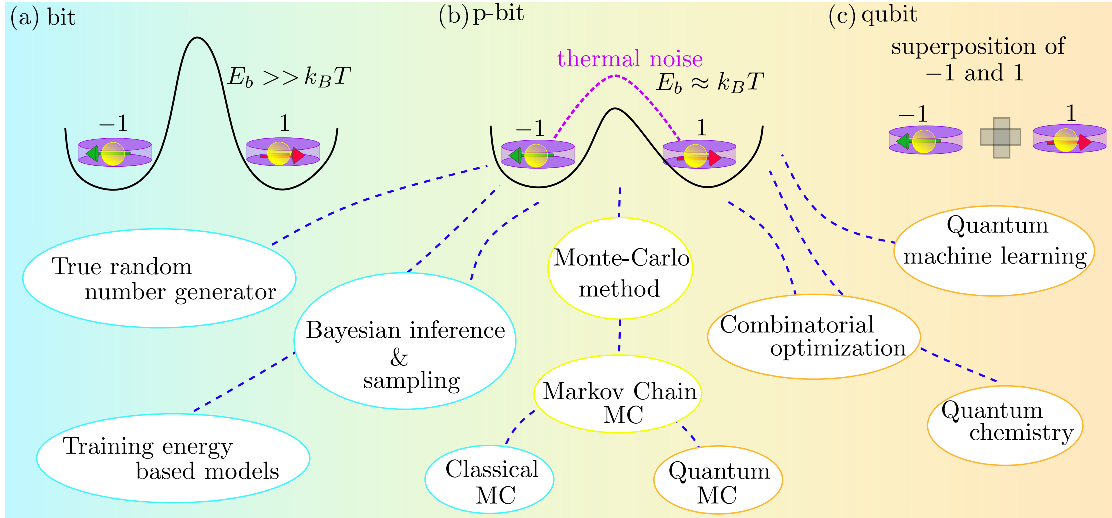


Fig. 2.9: Applications of probabilistic computing [1], [4].

classical method for learning BM for a generative and discriminative task is proposed in [101]. A mean-field assisted training of fully connected Boltzmann machine is demonstrated in [102]. A Deep Boltzmann Machine (DBM), a multilayer generalization of the original Boltzmann Machine, is trained on standard datasets including the MNIST, Fashion MNIST, and the Canadian Institute for Advanced Research 10-class dataset (CIFAR-10). The model achieved accuracy comparable to conventional methodologies and operated at a significantly greater speed than the existing traditional CPUs [25].

A considerable effort has gone into designing a clockless p-bit network that can autonomously arrive at the solution [29], [103]. A clockless autonomous training of the Boltzmann machine learning algorithm is demonstrated in [93]. Alternatively, other machine learning applications, such as in-hardware training of Deep Neural Networks (DNN) that harnessed the invertibility property of the probabilistic computing to perform back-propagation are discussed in [94], [96]. A performance comparison of the training hardware for convolutional neural networks (CNNs) implemented with probabilistic computing and classical computing is reported in [95].

The probabilistic network is an energy-based model and is prone to getting stuck in local minima. A self-adaptive gate control for high convergence rates to the global minimum energy is demonstrated in [104], and shows the convergence probabilities to the global minimum a few times higher than those based on conventional methods.

Similarly, other applications, such as the parallel tempering and the Maximum Sat-

isfiability (Max-SAT) problem implemented with thousands (≈ 3000) of p-bits, showed that the p-bit-based implementation outperformed the conventional algorithms in terms of the number of iterations required to arrive at the solution [105]. On a similar note, a sparse network is used to achieve massive parallelism. The sparsely connected network allows parallel update instead of the sequential update generally used in the traditional Gibbs sampling, thereby significantly improving the sampling speed and efficiency [29], [106], [107].

A noise-augmented chaotic bit (c-bit) is employed to enhance the performance of combinatorial optimization and sampling tasks. Additionally, an Adaptive Parallel Tempering (APT) algorithm, which outperforms fully deterministic c-bits executing simulated annealing, is demonstrated in [108].

On the other extreme, a full-stack view of probabilistic computing incorporates hardware, architecture, and algorithmic perspectives to cater to the applications of probabilistic computers ranging from probabilistic machine learning and AI to combinatorial optimization and quantum simulation [4]. A physics-based, modular, CMOS-compatible modelling approach connecting the microscopic physics of spins and magnets all the way up to circuits and system is demonstrated in [109].

Chapter 3

Implementation of Probabilistic bits using Low Barrier Magnets

In this chapter, the material optimization in the realm of probabilistic computing is explored. A methodology is proposed to select material parameters for an LBM-based p-bit implementation that improves flips per second (fps), a crucial system-level figure of merit (FOM). Subsequently, the proposed p-bit is integrated with a magnetic tunnel junction (MTJ), and the performance of p-bit implementations with different material parameters is evaluated. The work presented in this chapter is published in [80].

3.1 Material optimization oriented design of a p-bit

In this section, a methodology to optimize the material for designing an LBM-based p-bit is presented. The energy barrier E_b of a Nanomagnet (NM) is given as $E_b = H_k M_s \Omega/2$, where H_k is the in-plane uniaxial anisotropy, M_s is the saturation magnetization and Ω is the volume of the NM [110], [111]. A small-E_b NM is a crucial requirement for a p-bit, as E_b directly controls the stochasticity of the p-bit. The two methods of reducing E_b are 1) by reducing H_k and 2) by reducing $M_s\Omega$ [37]. However, simply reducing E_b does not ensure the optimal performance of the p-bit [15]. Therefore, the motivation of this work is to determine the realistic range of material parameters that not only reduce E_b, but also ensure the optimal performance at the system level. Hence, the results presented in this chapter are expected to provide direction to the material exploration for p-bit implementations.

3.1.1 Design of LBM for p-bit implementation

Memory applications are characterized by the use of stable, high barrier magnets with $E_b \approx 40\, k_B T$ to $60\, k_B T$. However, for p-bit functionality, spontaneous fluctuation

between two logic states is required. The rate at which the LBM fluctuates is a key parameter in defining the performance at the system level. In literature, it is characterized as the flips per second (fps). The fps is related to another parameter, the correlation time τ_c of a magnet, and is approximated as $1/\tau_c$, where τ_c is defined as the attempt time to change the state of the magnet [29]. However, recent work showed that τ_c is not the same for all LBMs and is given by the following equations [37]:

$$\tau_c = \sqrt{8ln(2)}\frac{1}{\gamma}\sqrt{\frac{M_s\Omega}{H_D k_B T}} \quad : E_b < k_B T \tag{3.1}$$

$$\tau_c = \tau_0 \frac{e^{E_b/k_B T}}{1 + \frac{H_D}{2H_k}} \quad : 2k_B T < E_b < 10k_B T \tag{3.2}$$

where gyromagnetic ratio $\gamma = 1.76 \times 10^7 \,/s\text{-}Oe$, demagnetization field $H_D = 4\pi M_s$ and $\tau_0 \approx 0.1 - 1ns$. However, fps in a p-bit can be increased if τ_0 lower than $100ps$ can be realized for p-bit materials [103]. For all the SPICE simulation-based analysis reported in this thesis, $\tau_0 = 1ns$ is used [37]. Furthermore, for an IMA LBM, the Eq. 3.2 is also valid in the range $k_B T < E_b < 2k_B T$. The factor $(1 + H_D/2H_k)$ shows the contribution of both easy-plane anisotropy and in-plane uniaxial anisotropy fields [110]. For $E_b > 10k_B T$, the $e^{E_b/k_B T}$ term dominates, and the conventional equation of τ_c, given by Arrhenius model as $\tau_c = \tau_0 e^{E_b/k_B T}$ is applicable [112].

The small fps value signifies that the magnet is almost pinned to a logic level (-1 or 1). When this type of magnet is integrated into a 1T-1MTJ architecture, it shows a *distorted sigmoid*. A distorted sigmoid has a step-like characteristic and is not the desired behavior for a p-bit. For $E_b < k_B T$, Eq. 3.1 shows that fps is dependent on volume Ω. The volume Ω is decided by the process technology and the desired spin orientation of the magnet (in-plane or out-of-plane). It should be small enough to have a monodomain behavior (depends on the diameter), and the thickness should be large enough for m to be in-plane instead of out-of-plane [113]. For example, the change in the orientation of magnetic anisotropy with the thickness of the cobalt (Co) thin film NM is demonstrated in [114].

The LBM with circular in-plane magnetic anisotropy (IMA), as shown in Fig. 3.1(a), is characterized by a telegraphic nature (m_z close to $\pm z$-axis) in contrast to the relatively continuous magnetization observed in perpendicular magnetic anisotropy (PMA). Ad-

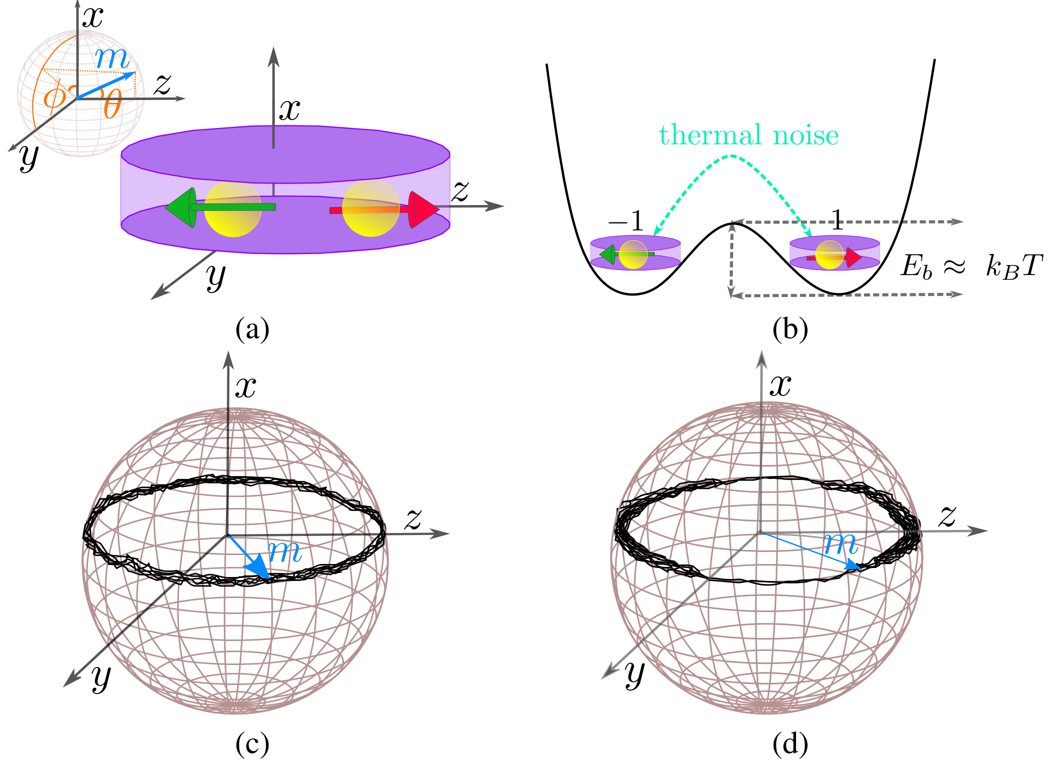


Fig. 3.1: (a) Schematic illustration of an LBM. (b) Thermal fluctuations between two logic levels -1 and $+1$ separated by E_b. Magnetization dynamics on the Bloch sphere for circular IMA: (c) $E_b < 2\,k_BT$, $H_k \approx 100\,Oe$, (d) $2\,k_BT < E_b < 4\,k_BT$, $H_k > 500\,Oe$ [5].

ditionally, a higher fps rate is exhibited by IMA magnets compared to PMA magnets. Therefore, the IMA magnet is preferred over the PMA magnet for the p-bit design [5]. In an LBM, thermal fluctuations are sufficient to overcome the energy barrier, as illustrated in Fig. 3.1(b). In this work, the model geometry of the magnet provided in [110] has been followed. The components of magnetization m of the LBM along the coordinate axes are given by $(m_x, m_y, m_z) \equiv (m\sin(\theta)\cos(\phi),\ m\sin(\theta)\sin(\phi),\ m\cos(\theta))$, as depicted in inset in Fig. 3.1(a). The quantities m_x, m_y, and m_z are linearly dependent since $|m| = m_x^2 + m_y^2 + m_z^2 = 1$, implying that m always remains on the Bloch sphere. Here, m_z corresponds to m_i in Eq. 2.1. For circular IMA, the magnetization vector m is constrained to remain in-plane due to the large demagnetization field $H_D = 4\pi M_s$, and it rotates freely in the easy plane (y-z), as shown in Fig. 3.1(c). However, when the energy barrier is in the range $2k_BT < E_b < 10k_BT$, a telegraphic behavior begins to emerge in m_z, with confined fluctuations appearing closer to the z-axis, as shown in Fig. 3.1(d) [37].

The fluctuation dynamics of the LBM is captured by the stochastic Landau-Lifshitz-Gilbert (sLLG) equation [2]:

$$\begin{aligned}(1+\alpha^2)d\hat{m}/dt = &-|\gamma|\,\hat{m}\times\vec{H} - \alpha\,|\gamma|\left(\hat{m}\times\hat{m}\times\vec{H}\right)\\ &+1/qN\left(\hat{m}\times\vec{I_S}\times\hat{m}\right)+\left(\alpha/qN\left(\hat{m}\times\vec{I_S}\right)\right)\end{aligned} \tag{3.3}$$

where normalized magnetization $\hat{m} = m/M_s$, m is the magnetization, α is the Gilbert damping parameter, q is the electron charge, $N = M_s\Omega/\mu_B$ is the total number of spins in the LBM, and μ_B is the Bohr magneton. $\hat{m}$ can be controlled by the effective field $\vec{H}$ or the spin current $\vec{I_S}$. $\vec{I_S}$ is the spin-polarized charge current flowing from the fixed to free layer or vice-versa. The $\vec{H} = 4\pi M_s m_z\hat{z} + \vec{H}_N$ has two terms: H_D and the isotropic thermal noise field $\vec{H}_N$. $\vec{H}_N$ is a three dimensional uncorrelated thermal noise having a Gaussian distribution with $\langle H_N\rangle = 0$ and standard deviation $\left\langle {H_N}^2\right\rangle = 2\alpha k_B T/(|\gamma|\, M_s\Omega)$ along each coordinate axis. The Fokker-Planck equations have been used to model the LBM for circuit simulations [115], [116].

A small E_b is achieved by selecting materials with small values of H_k, M_s, and Ω [15]. In this work, two methods have been explored, namely, tuning the H_k and M_s values, to realize a magnet with E_b in the range of $(1,4)\,k_BT$. The methodology to select H_k and M_s value is explained in the next section. It should be noted that the chosen values of H_k and M_s lie within the range typically observed in real materials. The exact values of H_k, M_s, and Ω are presented in the subsequent sections.

Optimizing fps with H_k selection

The magnetization m changes from zero to M_s depending on several factors. One factor that strongly affects m is anisotropy. This term means that the magnetic properties depend on the direction of measurement. It is a widely researched subject and is exploited in the design of magnetic devices of commercial importance. There are different types of anisotropies, namely 1) crystalline anisotropy, also known as magnetocrystalline anisotropy, 2) shape anisotropy, 3) stress anisotropy, and 4) exchange anisotropy, etc [113].

In literature, for a p-bit, CoFeB is the most widely used material for designing an LBM. CoFeB has body-centered-cubic (bcc) crystal structure [117]. An experimental

analysis of p-bit implementation using circular disk nanomagnet of $Co_{60}Fe_{20}B_{20}$ (wt%) is demonstrated in [15]. Furthermore, NiFe based alloys (permalloys) are considered *soft material*. An extensive study on the structural and magnetic properties of $Ni_{81}Fe_{19}$ (wt%) is presented in [118]. Therefore, different magnetic properties of the *soft materials* can be exploited to implement a p-bit [113].

The LBM considered in this work is in the $y - z$ plane, and H_k is along the z-axis. For low values of H_k ($\approx$ few Oe), m_z fluctuates freely in the easy plane ($y - z$ plane) as shown in Fig. 3.1(c). The value of H_k depends on the material of the LBM, the shape of the magnet (circular or elliptical), etc. So, for an LBM with high H_k ($\approx 200\,Oe$) the movement of the m_z is restricted closer to the z-axis as shown in Fig. 3.1(d). However, if H_k is very high ($\approx 600\,Oe$), then that magnet is a *hard magnet*, and it is not suitable for p-bit implementation because of the smaller fps and distorted behavior of the sigmoid.

Table 3.1: Material parameters for different LBM

Material	H_k (Oe)	M_s (emu/cc)	E_b (k_BT)	fps ($/s$)
LBM-A	115.86	2000.00	1.16	3.43×10^{10}
LBM-B	134.48	1706.89	1.15	2.56×10^{10}
LBM-C	153.10	1472.41	1.13	1.99×10^{10}
LBM-D	302.07	1296.55	1.96	3.95×10^{9}
LBM-E	357.93	1120.68	2.01	2.78×10^{9}
LBM-F	395.17	1003.45	1.98	2.33×10^{9}
LBM-G	506.90	1062.06	2.69	9.59×10^{8}
LBM-H	544.14	1237.93	3.37	5.27×10^{8}
LBM-I	600.00	1296.55	3.89	2.98×10^{8}

The methodology to find suitable H_k and M_s is as follows. The energy value varies in the range of $0.5\,k_BT$ to $4\,k_BT$. For $E_b < k_BT$, fps is dependent on Ω. For $\Omega = 4.16 \times 10^{-19} cc$, fps $= 8.37 \times 10^{9}/s$. However, for $k_BT < E_b < 10k_BT$, fps is governed by the Eq. 3.2 and is a) $\propto M_s$, b) $\propto 1/H_k$, and c) $\propto 1/e^{E_b/k_BT}$. The pairs of H_k and M_s values considered for materials examined in this work are listed in Tab. 3.1. The variation of fps with H_k for different values of M_s is shown in Fig. 3.2.

The LBM-{A, B, C, D, and E} is preferred over the LBM-{F, G, H, and I}, because the H_k values for LBM-{F, G, H, and I} are fairly high and are considered as *hard magnets*. LBM-{E, F, and G} have M_s corresponding to CoFeB. The feasibility of achieving

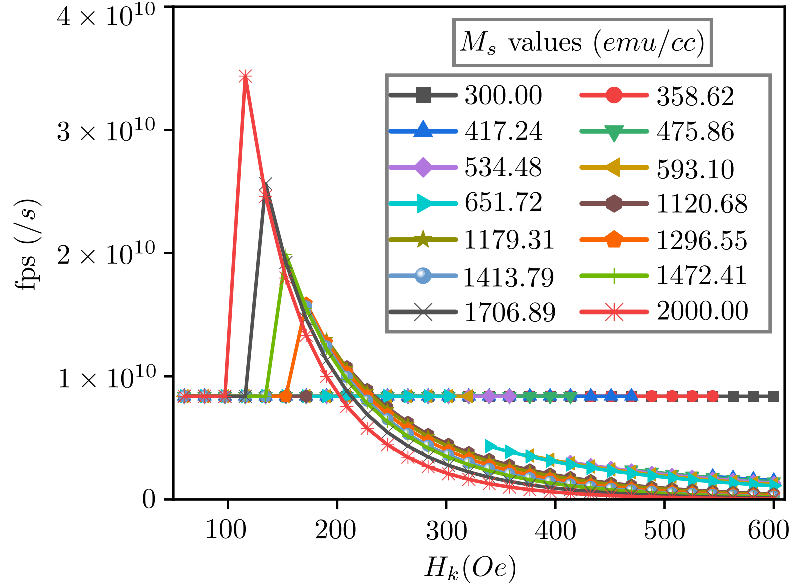


Fig. 3.2: Plot of fps versus H_k for different values of M_s. The analysis is performed for 30 values of M_s, separated linearly in the range of $(300, 2000)$ emu/cc. However, the graph is plotted only for 14 values of M_s to improve readability.

M_s corresponding to LBM-{A, B, and C} is assessed in the following subsection.

Optimizing fps with M_s selection

The maximum intensity of magnetization (i.e., the degree of alignment of magnetic domains) is represented by M_s. However, M_s is influenced by temperature T, volume Ω, and the crystalline structure. A decrease in M_s is observed with an increase in temperature. In the analysis presented in this work, the temperature T has been kept constant at $27\,^{\circ}\mathrm{C}$. With an increase in the thickness of the LBM, the saturation magnetization is observed to increase and then saturate at the maximum achievable M_s [113], [119]. It has been reported that a few *soft materials* may serve as alternative candidates for the LBM in p-bit implementations. For instance, NiFe has been shown to exhibit an increase in M_s with increasing thickness [120]. A comprehensive investigation of Heusler alloys has been presented in [121], [122], [123], where it has been demonstrated that the magnetic properties can be controlled by tuning the material composition of the alloy. Therefore, the magnetic material composition can be adjusted to obtain suitable H_k and M_s pairs corresponding to LBM-{A, B, and C}.

Based on the above analysis, the process of selecting magnetic materials for probabilistic computing applications is summarized as follows: 1) if $E_b < k_BT$, fps is

adjusted by decreasing Ω of the LBM. 2) if $k_BT < E_b < 10\,k_BT$, for a given fps, a few combinations of (H_k, M_s) are selected, as shown in Fig. 3.2 and Tab. 3.1. Next, the materials corresponding to the desired M_s values are identified. Once a material is selected, the adjustment in H_k and M_s values can be made via the geometry of the LBM.

Comparing the two methods

Fig. 3.3 shows the response of the 1T-1MTJ-based p-bit implemented using LBM materials examined above. The sigmoid trend shows the tunability of the p-bit. For

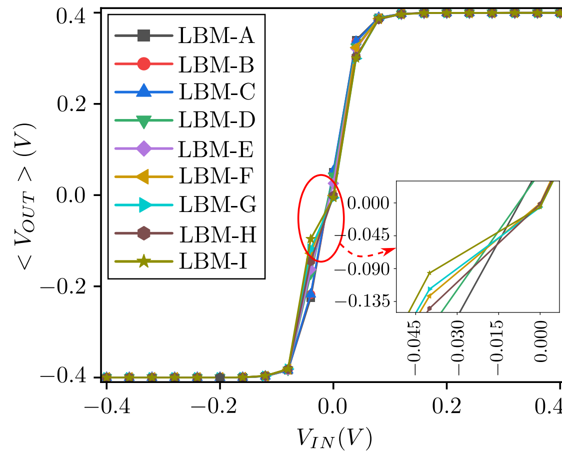


Fig. 3.3: Average of V_{OUT} (<V_{OUT}>) for different values of V_{IN} in the range $(-V_{DD}/2, V_{DD}/2)$. For LBM-{A, B, C, D, E, and F}, for each value of V_{IN}, V_{OUT} is averaged over $500\,ns$. For LBM-{G, H, and I}, the average window is increased to $1\,\mu s$ to incorporate more fps. The inset shows the comparison of the sigmoid response of LBM-{A and D} with LBM-{F, G, H, and I} for $V_{IN} \approx 0\,V$.

$V_{IN} \approx 0\,V$, the R_{NMOS} and R_{MTJ} are adjusted such that a $0\,V$ (on an average) is obtained at V_x. This is possible when the MTJ has high fps, otherwise, a pinning effect is observed. For LBM-{G, H, and I}, E_b is less than $4\,k_BT$. This energy barrier lies in the recommended design limit. However, the pinning effect is observed in the output response because of the high value of H_k. The experiments suggested that a high value of M_s and a low value of H_k such that $E_b < 4\,k_BT$ does not distort the sigmoid. Therefore, the selection of the H_k value is more crucial than the M_s, as the high value of H_k is equivalent to a *hard magnet*. The distorted sigmoid signifies a strong telegraphic nature of p-bit that does not fluctuate between the two logic levels. Such p-bits are unsuitable hardware for probabilistic computing applications because fps encapsulates

the stochasticity of the p-bit implementation.

In this work, the range of material parameters for low barrier magnet design is proposed assuming that the p-bit circuit can be designed that is able to capture the fast fluctuations in the MTJ. It is also projected that integrated solutions such as 2D material-based FETs may have the potential to realize p-bits fluctuations in GHz range [60] [61].

3.2 Conclusion

The impact of material parameters, namely H_k, M_s, and Ω, and E_b on the output response of a p-bit is investigated. Furthermore, a method of selecting material parameters for the LBM-based p-bit implementation that improves fps, a crucial system-level figure of merit, without sacrificing the ideal sigmoid-type output response is proposed. Additionally, the criticality of H_k on the output response of a p-bit is highlighted. Hence, the impact of H_k changes, for example, because of process-induced variations should be considered, while designing a p-bit-based computational system. The results presented in this work provides the direction to the material exploration for the LBM-based p-bit implementation. It should be noted that the conclusions drawn in this study for 1T-1MTJ STT MRAM may not be applicable to the 1T-1MTJ spin-orbit torque (SOT) MRAM-based p-bit implementation. However, the methodology outlined in this work for identifying suitable material parameters can also be followed in re-evaluating 1T-1MTJ SOT MRAM p-bits.

Chapter 4

Impact of Non-Idealities on the Behavior of Probabilistic Computing

The ideal p-bit implementation, given by Eq. 2.1 in Chapter 2, is mathematically abstracted as an ideal p-bit, while the hardware implementation of a p-bit (discussed in Chapter 2) deviates from this ideal sigmoidal behavior [80]. These deviations in a p-bit can be caused by process-induced variations, aging-induced reliability vulnerabilities, or implementation-specific non-idealities. Moreover, non-idealities at the p-bit level manifest at the circuit and system levels, potentially disrupting functionality. In this chapter, this aspect of a p-bit network is investigated. First, an analytical model is developed to assess the impact of non-idealities in a p-bit, and its application in a 1T-1MTJ-based p-bit implementation is demonstrated. Then, this model, along with the understanding of p-bits, is utilized to analyze their impact on a p-bit network implemented using SPICE models. The work presented in this chapter is published in [82].

4.1 Model of a Non-ideal p-bit

An ideal p-bit can be modelled using random variables as follows:

$$\mathbf{Y} = \text{sgn}(\tanh(\mathbf{X}) + \mathbf{E}) \tag{4.1a}$$

$$= \text{sgn}(\mathbf{G}) \tag{4.1b}$$

The random variable $\mathbf{Y}$ is the output produced by a p-bit, the random variable $\mathbf{X}$ models the input received by the p-bit, and the random variable $\mathbf{E}$ models the inherent randomness in the p-bit. A random variable $\mathbf{G}$ is introduced for easy explanation. For an ideal p-bit, $\mathbf{E}$ is a continuous random variable uniformly distributed in the range $[-1, 1]$. Additionally, it is assumed that $\mathbf{X}$ is uniformly distributed in the range $[-R, +R]$. The sgn

function in the above equation ensures that the output $\mathbf{Y}$ is bipolar, similar to Eq. 2.1. Hence, $\mathbf{Y}$ is considered as a Bernoulli's random variable with its characteristics defined by a parameter p, where p is its probability of being 1. The value of p is computed as follows:

$$p = \int_{0}^{\infty} f_{\mathbf{G}}(g)dg \tag{4.2}$$

where $f_{\mathbf{G}}(g)$ is the Probability Density Function (PDF) of $\mathbf{G}$. For an ideal p-bit, in the absence of the input stimulus, i.e. for $\mathbf{X} = 0$, $\mathbf{Y}$ is equal to sgn($\mathbf{E}$). Therefore, by evaluating the area under the curve of $f_{\mathbf{G}}(g)$ according to Eq. 4.2, $p = 0.5$ is obtained.

In a realistic circuit where a p-bit receives inputs from other connected p-bits, the input $\mathbf{X}$ will be biased. This bias forces a p-bit to favour one of the two states and delivers the required functionality of a p-bit network. Further, the bias can be due to non-ideal interactions between p-bits and undesirable variations in the input due to environmental and process-induced factors. These effects are modelled using a parameter q and a shifted random variable $\mathbf{V} = \mathbf{X} - q$. Additionally, a realistic p-bit can be inherently biased. The bias can be due to limitations in the implementation, process-induced variations and aging-induced reliability vulnerabilities [61]. These inherent biases in a p-bit are modelled using a parameter r and a shifted random variable $\mathbf{Z} = \mathbf{E} - r$. Hence, a non-ideal p-bit is described as follows:

$$\mathbf{Y} = \text{sgn}(\tanh(\mathbf{X} - q) + (\mathbf{E} - r)) \tag{4.3a}$$

$$= \text{sgn}(\tanh(\mathbf{V}) + \mathbf{Z}) \tag{4.3b}$$

$$= \text{sgn}(\mathbf{U} + \mathbf{Z}) \tag{4.3c}$$

$$= \text{sgn}(\mathbf{H}) \tag{4.3d}$$

The random variables $\mathbf{U}$ and $\mathbf{H}$ is introduced for easy explanation. The behavior of a non-ideal p-bit is analyzed by deriving the probability p for the output $\mathbf{Y}$ as described below. The PDF of $\mathbf{V}$ and $\mathbf{Z}$ are the shifted versions of the PDF of $\mathbf{X}$ and $\mathbf{E}$, respectively. Since tanh is a strictly monotonic differentiable function, the PDF of $\mathbf{U}$ is

computed as follows [124]:

$$f_{\mathbf{U}}(u) = \left.\frac{f_V(v)}{\frac{du}{dv}}\right|_{v=\tanh^{-1}(u)} \tag{4.4}$$

The PDF of $\mathbf{U}$ is given by the following equation:

$$f_{\mathbf{U}}(u) = \left\{ \begin{array}{cc} \frac{2}{(1-u^2)\ln(\Theta\cdot\Phi)} & -\tanh(R+q) \leq u \\ & \leq \tanh(R-q) \\ & \\ 0 & \text{Otherwise} \end{array} \right\}$$

where $\Theta = \frac{1+\tanh(R+q)}{1-\tanh(R+q)}$ and $\Phi = \frac{1+\tanh(R-q)}{1-\tanh(R-q)}$. Similarly, the PDF of $\mathbf{Z}$ is given by the following equation:

$$f_{\mathbf{Z}}(z) = \left\{ \begin{array}{cc} \frac{1}{2} & -1-r \leq z \leq 1-r \\ 0 & \text{Otherwise} \end{array} \right\}$$

Subsequently, the PDF of $\mathbf{H} = \mathbf{U} + \mathbf{Z}$ is calculated using the convolution ($\circledast$) of $\mathbf{U}$ and $\mathbf{Z}$, and is given by the following equations:

$$\begin{aligned} f_{\mathbf{H}}(h) &= f_{\mathbf{U}}(u) \circledast f_{\mathbf{Z}}(z) \\ &= \int_{-\infty}^{+\infty} f_U(u) \cdot f_Z(h-u) du \end{aligned}$$

The convolution is computed over the following ranges:

1. $-1 - \tanh(R+q) - r \leq h \leq -1 + \tanh(R-q) - r$

$$f_{\mathbf{H}}(h) = \int_{-\tanh(R+q)}^{1-r+h} \frac{2}{(1-u^2)\ln(\Theta\cdot\Phi)} \cdot \frac{1}{2} \cdot du$$

2. $-1 + \tanh(R-q) - r \leq h \leq 1 - \tanh(R+q) - r$

$$f_{\mathbf{H}}(h) = \int_{-\tanh(R+q)}^{\tanh(R-q)} \frac{2}{(1-u^2)\ln(\Theta\cdot\Phi)} \cdot \frac{1}{2} \cdot du$$

3. $1 - \tanh(R+q) - r \leq h \leq 1 + \tanh(R-q) - r$

$$f_{\mathbf{H}}(h) = \int_{-1-r+h}^{\tanh(R-q)} \frac{2}{(1-u^2)\ln(\Theta \cdot \Phi)} \cdot \frac{1}{2} \cdot du$$

The PDF of $\mathbf{H}$ is given by Eq. (4.6):

$$f_{\mathbf{H}}(h) = \begin{Bmatrix} \frac{\ln\left(\Theta \cdot \frac{2+r+h}{-r-h}\right)}{2\ln(\Theta \cdot \Phi)} & -1 - \tanh(R+q) - r \leq h \\ & \leq -1 + \tanh(R-q) - r \\ \frac{1}{2} & -1 + \tanh(R-q) - r \leq h \\ & \leq 1 - \tanh(R+q) - r \\ \frac{\ln\left(\Phi \cdot \frac{2-r-h}{r+h}\right)}{2\ln(\Theta \cdot \Phi)} & 1 - \tanh(R+q) - r \leq h \\ & \leq 1 + \tanh(R-q) - r \\ 0 & \text{Otherwise} \end{Bmatrix} \tag{4.6}$$

Finally, p is obtained using the PDF of $\mathbf{H}$ and Eq. 4.2. The probability p represents the area under the curve along the positive X-axis for this PDF. The non-idealities introduced in the above p-bit model modify this area and change the output response of a p-bit. If these modifications are solely because of intended input modulation, a p-bit network works as designed and is expected to be functionally correct. However, if significant unintended non-idealities exist, such as due to environmental or process-induced variations, the functionality implemented using the p-bit network can be disrupted. Therefore, the impact of these non-idealities on the output of a p-bit should be examined carefully.

Fig. 4.1(a) shows the PDF $f_{\mathbf{H}}(h)$ for an ideal p-bit. It is symmetric around the Y-axis and will produce $p = 0.5$. Fig. 4.1(b) and (c) show $f_{\mathbf{H}}(h)$ when some bias r is added. A shift of the PDF along the X-axis is observed, depending on the value of r. Consequently, the p-bit output $\mathbf{Y}$ is expected to favor one of the two states. Fig. 4.1(d) and (e) show $f_{\mathbf{H}}(h)$ when the bias q is added. It is observed that the shape of the PDF changes on the two sides of the Y-axis (but it is not shifted along the X-axis), and a preference for one of the two states is expected from the p-bit. Fig. 4.1(f) shows $f_{\mathbf{H}}(h)$ when both the terms q and r are non-zero. The curves are shifted, and their shapes also

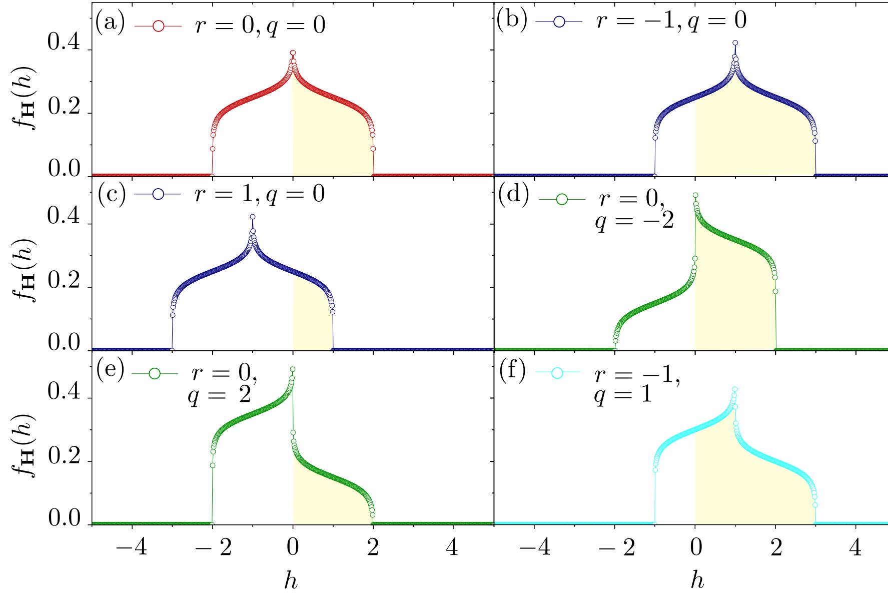


Fig. 4.1: Illustration of impact of q and r on $f_{\mathbf{H}}(h)$. The shaded area represents the p-bits' probability p being +1.

change, and it is expected that the p-bit will favor one of the two states.

Table 4.1: Probabilities for a non-ideal p-bit obtained using analytical model and simulation (R=5)

P-bit parameters	Analytical model (p)	Simulation results (p) Number of samples (N)		
		$N=100$	$N=1000$	$N=10000$
$r=0, q=0$	0.50	0.55	0.50	0.51
$r=-1, q=0$	0.78	0.85	0.75	0.78
$r=1, q=0$	0.22	0.15	0.21	0.22
$r=0, q=-2$	0.70	0.67	0.69	0.69
$r=0, q=2$	0.30	0.35	0.30	0.30
$r=-1, q=1$	0.73	0.71	0.73	0.73

The computed probability p for the non-ideal p-bit using the analytical model described in Eq. 4.6 is shown in Tab. 4.1. This table also reports the probability p computed numerically using Eq. 5(a) for varying number of samples of random variables. It is observed that the results obtained by numerical simulations change with the number of samples and are expected to become stable when N is large. Furthermore, it is found that the value of p computed using the analytical model matches that obtained from numerical simulation for large N. This agreement is expected, as the analytical model

inherently captures the asymptotic behavior of probabilities. This is an advantage of the analytical model over the numerical simulation.

From Tab. 4.1, it should be noted that both the terms q and r modify p-bits' probability p. However, the nature of their impact is different, as illustrated in Fig. 4.1. In some cases, their impact can be in the opposite direction (one increasing p and the other decreasing p), reducing their overall effect on p. In contrast, in others, they can be in the same direction, and their impact can add up. The canceling nature of these terms can average out their effects and make a p-bit network more tolerant against variations. For example, consider the non-ideal p-bits shown in Tab. 4.2. For $r = 0, q = -0.5$, the value of p becomes 0.55. However, p will come closer to the ideal case of 0.5 when the variation due to $r = 0.08$ cancels the effect of $q = -0.5$. Similarly, the effect of $r = 0.1$ is canceled by $q = -0.45$, as shown in Tab. 4.2. These considerations are critical in making p-bit networks more resilient to errors and controlling their dynamic response and robustness.

Table 4.2: Probabilities for a non-ideal p-bit obtained using analytical model and simulation (R=5)

P-bit parameters	Analytical model (p)	Simulation results (p) Number of samples (N)		
		$N = 100$	$N = 1000$	$N = 10000$
$r = 0, q = -0.5$	0.55	0.54	0.57	0.55
$r = 0.08, q = -0.5$	0.52	0.55	0.50	0.52
$r = 0.1, q = 0$	0.47	0.50	0.48	0.47
$r = 0.1, q = -0.45$	0.51	0.52	0.52	0.51

In the next section, the 1T-1MTJ-based implementation of a p-bit and the relevance of the above model in capturing the non-idealities in the implementation is demonstrated.

4.2 Modeling Variations in a p-bit implementation

Fig. 4.2 shows the equivalent circuit implementation of a p-bit employed in SPICE simulation. The design parameters of the p-bit are listed in Tab. 4.3. In this section, it is illustrated that the characteristics of a p-bit obtained through SPICE simulation can

be modelled using the mathematical formulation of an ideal p-bit, as described by Eq. 4.1.

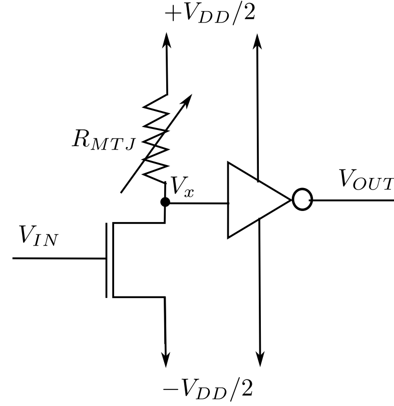


Fig. 4.2: Circuit implementation of 1T-1MTJ-based p-bit using SPICE models [2].

Table 4.3: Parameters used in SPICE simulations of p-bits

Parameters	**Value**
Gilbert damping coefficient (α)	0.01 [2]
Volume of the free layer (Ω)	$4.16 \times 10^{-19}\, cc$
Saturation magnetization (M_s)	$1200\, emu/cc$
In-plane uniaxial anisotropy field (H_k)	$100\, Oe$
Temperature (T)	$27\,°C$
Tunnel magnetoresistance (TMR)	Nominal value: 200%
Average conductance of MTJ (G_0)	Nominal value: $46\, \mu S$
Transistor technology models	14-nm HP-FinFET [40]
Time step for transient simulation (Δt)	$1ps$ [2]

Fig. 4.3 compares the simulated characteristics for varying V_{IN} with the mathematical model of Eq. 4.1. Note that the X of the mathematical model is obtained using $X = m * V_{IN}$, where m is a scaling factor. From Fig. 4.3, it is inferred that the 1T-1MTJ-based p-bit implementation with nominal parameters (as shown in Tab. 2.2) can be modeled well by the proposed ideal p-bit analytical model.

However, a realistic 1T-1MTJ-based p-bit implementation is prone to process-induced variations impacting the attributes of both the transistor and the MTJ. The process-induced variations and non-idealities can result in variations in the dimensions of the transistors (such as width, length, fin height, and oxide thickness), threshold voltage, and physical parameters such as mobility [61]. The attributes of an MTJ can be impacted by process-induced variations in the thickness of the oxide layer and the cross-

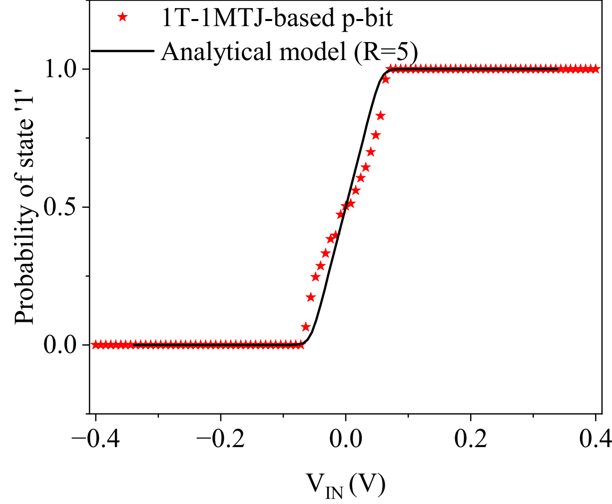


Fig. 4.3: Comparison of the results obtained from the circuit simulation of 1T-1MTJ-based p-bit implementation and the analytical model. A scaling factor $m = 88.49/V$ is included such that, $\mathbf{Y} = \text{sgn}(\tanh(m * V_{IN}) + \mathbf{E})$ models the output response of the p-bit.

sectional area of the MTJ stack [62]. In the context of the p-bits, the variations in the p-bits are manifested as the misaligned average response, i.e., the equal contribution of the valid states does not occur at the zero stimuli to the input of the p-bit, and the distorted shape of the average response [14], [55]. The misalignment of the output response is the consequence of the mismatch between the R_{NMOS} and R_{MTJ}, a crucial design metric for a p-bit [61]. Furthermore, the attributes of the fixed and the free layers of the MTJ also control the stochastic behavior of the p-bit. The magnetization of the magnetic material m_z is a function of the uniaxial anisotropy (H_k), the saturation magnetization (M_s), the volume of the nanomagnet, and the spin polarization current (I_s) [2]. These parameters are impacted by the geometry of the LBM [37], [80]. Thus, non-idealities can arise for various reasons in a realistic p-bit implementation.

Next, it is illustrated that the mathematical model proposed in Eq. 4.3 is capable of capturing these non-idealities; hence, the simplified mathematical model can be used to assess the impact of such non-idealities in realistic p-bit implementations. For the sake of illustration, the impact of variations in the following parameters in the 1T-1MTJ-based p-bit implementation is considered: variations in the fin height (h_{fin}) of the transistor and the average conductance (G_0) of the MTJ.

The variation in h_{fin} of the FinFET impacts its drive current. Therefore, the variation in h_{fin} can support or oppose the external bias applied to the p-bit and is expected

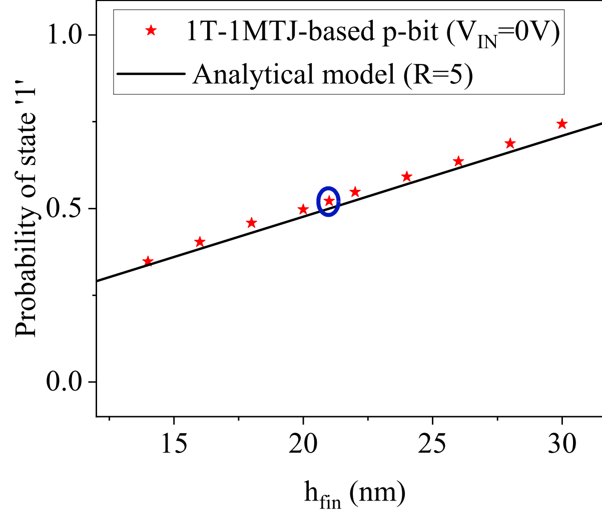


Fig. 4.4: Comparison of the results obtained from the circuit simulation and the analytical model. The nominal value of the fin height $h_{fin}^{nom} = 21nm$ is marked in blue. A scaling factor $c = -0.232/nm$ is included such that, $\mathbf{Y} = \text{sgn}(\tanh(m * V_{IN} - c * (h_{fin} - h_{fin}^{nom})) + \mathbf{E})$ models the output response of the p-bit.

to favor one of the two states. Hence, the parameter q in the analytical model of Eq. 4.3 is used, which works together with the external bias, to capture the variations in the fin height. Fig. 4.4 shows the variation in the probability of observing state '1' in the p-bits with the variation in h_{fin} obtained using SPICE simulation. Additionally, the results obtained using the mathematical model of Eq. 4.3 are shown. It should be noted that the q of the mathematical model is obtained from the variation in the fin height using $q = c * (h_{fin} - h_{fin}^{nom})$, where c is a scaling factor and h_{fin}^{nom} is the nominal height of the fin. These results illustrate that the effects of variations can be captured well using the parameter q.

Next, the modeling of the impact of variations in the MTJ in the p-bit implementation is explored. The average conductance G_0 depends on the parallel (G_P) and the anti-parallel (G_{AP}) conductance of the MTJ. These conductances vary with the thickness of the oxide layer and the cross-sectional area of the fixed and free layers and are prone to process-induced variations. The MTJ is the primary source of stochasticity in the 1T-1MTJ-based p-bit implementation, and the variations in G_0 can introduce an inherent bias in the p-bit implementation. Hence, the parameter r is used in the analytical model of Eq. 4.3, which provides a bias to the inherent randomness, to capture the variations in G_0. Fig. 4.5 shows the variation in the probability of observing state '1' in the p-bits with the variation in G_0 obtained using SPICE simulation. Additionally, it

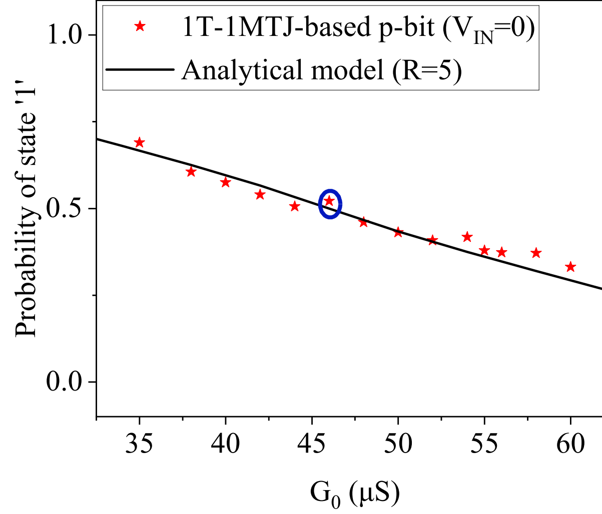


Fig. 4.5: Comparison of the results obtained from the circuit simulation and the analytical model. The nominal value of $G_0^{nom} = 46\mu S$ is marked in blue. A scaling factor $k = 0.05/\mu S$ is included such that, $\mathbf{Y} = \text{sgn}(\tanh(m * V_{IN}) + (\mathbf{E} - k * (G_0 - G_0^{nom}))$ models the output response of the p-bit.

is demonstrated using the mathematical model of Eq. 4.3. It should be noted that the r of the mathematical model is obtained from the variation in the average conductance using $r = k * (G_0 - G_0^{nom})$, where k is a scaling factor and G_0^{nom} is the nominal average conductance of the MTJ. These results illustrate that the effects of G_0 variations can be captured reasonably well using the parameter r.

Furthermore, the canceling effect of non-idealities in the SPICE simulations is observed similar to one predicted by the analytical model (shown in Section 4.1). For example, when there is a variation in the fin height from the nominal value of $h_{fin}^{nom} = 21nm$ to $h_{fin} = 16nm$, the probability of state '1' becomes 0.38 (less than the ideal probability of 0.5), while when there is a variation in the average conductance from the nominal value of $G_0^{nom} = 46\mu S$ to $G_0 = 35\mu S$, the probability of state '1' becomes 0.68 (more than the ideal probability of 0.5). However, when both these variations occur together ($h_{fin} = 16nm$ and $G_0 = 35\mu S$), the probability of state '1' is observed to be 0.52. Hence, the non-idealities with the opposite direction of impact can cancel their effect in p-bit implementation, similar to the observation derived from the mathematical model. The canceling behavior of the non-idealities indicates a possible direction to make a p-bit network more robust.

The above illustrations demonstrate the usefulness of the proposed analytical model in capturing the non-idealities in a p-bit implementation using the model parameters q

and r. However, it should be highlighted that the extent of the variations in the device parameters depends on the specific fabrication technology, and the device variations considered above are only for illustrative purposes. Moreover, other transistor/MTJ parameters, such as the threshold voltage, mobility of carriers, dimensions of the fixed and free layers, interface properties, and the systematic variations in material parameters, may require considering the appropriate combination of q and r values in the proposed model. The variation in the threshold voltage impacts the conductivity of the NMOS transistor. For example, the increase in V_{th} decreases the conductivity of the NMOS. Therefore, the resistance of the NMOS increases. As a result, V_x is pulled up to $+V_{DD}/2$, and V_{OUT} is pulled down to $-V_{DD}/2$. Similarly, for decrease in V_{th} value, the V_{OUT} is pulled up to $+V_{DD}/2$. In essence, the impact on the probability distribution would be opposite to that of the trend observed for h_{fin}. The variation in the TMR value affects the shape of the sigmoidal response of the p-bit. The variation in the temperature affects the mobility of charge carriers, leakage current and the threshold voltage in the CMOS circuit. Similarly, the thermal fluctuations in the MTJ are impacted by variations in the temperature. Additionally, the variation in the supply voltage may also impact the performance of the p-bits. This aspect of the model calibration for other types of non-idealities requires further investigation and assessment, and is intended for future work. Additionally, the proposed model will not cover some types of variations in the p-bit attributes, such as those related to the dynamic response (e.g., flips per second for the nanomagnets). Moreover, the variations in p-bit interconnections are implicitly captured within the q parameter. This is a valid assumption because the input to a p-bit is the sum of weighted outputs of the interconnected p-bits, according to Eq. 2.2. However, SPICE models based circuit simulations to analyze the impact of variations in the interconnections is intended as future work.

4.3 Impact of non-idealities in a p-bit on a p-bit network

For a given p-bit network, some states deliver the required functionality, and these states are referred to as valid states. Conversely, the states that are prohibited within the same p-bit network are referred to as invalid states. For example, a two-input AND gate implemented using three p-bits - A, B, and C is considered. Here, A and B are assumed to

be the inputs, while C is the output of the AND gate. Encoding the states as (A, B, C), the valid states are $\{(-1,-1,-1),(-1,1,-1),(1,-1,-1),(1,1,1)\}$ and the invalid states $\{(-1,-1,1),(-1,1,1),(1,-1,1),(1,1,-1)\}$. In an ideal p-bit network with no external bias, all valid states have an equal probability of occurrence, while all invalid states have zero probabilities.

In practice, a p-bit network exhibits non-ideal behavior, and a suppression of probabilities of valid states and enhancement in the probabilities of invalid states is observed. Any unexpected modulation in the incoming signal or skewness in the inherent randomness (e.g., due to process-induced variations) can potentially force a p-bit to be biased towards an incorrect state, disrupting the functionality of the network. To quantify these non-idealities in a p-bit network, a parameter called observed non-ideality NI_obs is defined as follows:

$$NI_obs = \sqrt{\frac{\sum_{S_i}(\hat{P}(S_i) - \overset{*}{P}(S_i))^2}{N}} \tag{4.7}$$

where N is the total number of states in a p-bit network, S_i is the i-th state, $\hat{P}(S_i)$ is the observed probability for S_i, and $\overset{*}{P}(S_i)$ is the ideal probability of the state S_i. For a two-input AND gate, $N = 8$, the ideal probabilities of all valid states are $1/4$, and the ideal probabilities of all invalid states are zero.

Next, a p-bit-based implementation of an AND gate, as shown in Fig. 2.6, is considered. Assuming that the p-bits (A, B, and C) are ideal and described by Eq. 4.1, the observed probability $\hat{P}(S_i)$ may differ from the ideal probability $\overset{*}{P}(S_i)$ when a limited number of samples is taken, even for an ideal p-bit. As the number of time samples increases to infinity, keeping β constant, $\hat{P}(S_i)$ will converge towards $\overset{*}{P}(S_i)$, and NI_obs is expected to approach zero for an ideal p-bit. However, in practice, each p-bit can have non-idealities, as described by Eq. 4.3. The non-idealities in the p-bits A, B, and C are denoted as (q_A, r_A), (q_B, r_B), and (q_C, r_C), respectively. First, the impact of these non-ideality terms is examined separately on the NI_obs for the AND gate. Fig. 4.6(a-f) shows the NI_obs with the variation in q_A, r_A, q_B, r_B, q_C, and r_C, respectively.

Fig. 4.6(a) plots the NI_obs for various values of q_A, keeping all other non-ideality terms as 0. It is observed that the NI_obs increases as q_A moves further away from zero. For $q_A = -1$, p-bit A gets biased towards the state 1. Therefore, the states $\{(1,-1,-1),(1,1,1)\}$ dominate ($\approx 40\%$ contribution of each state), as shown in Fig.

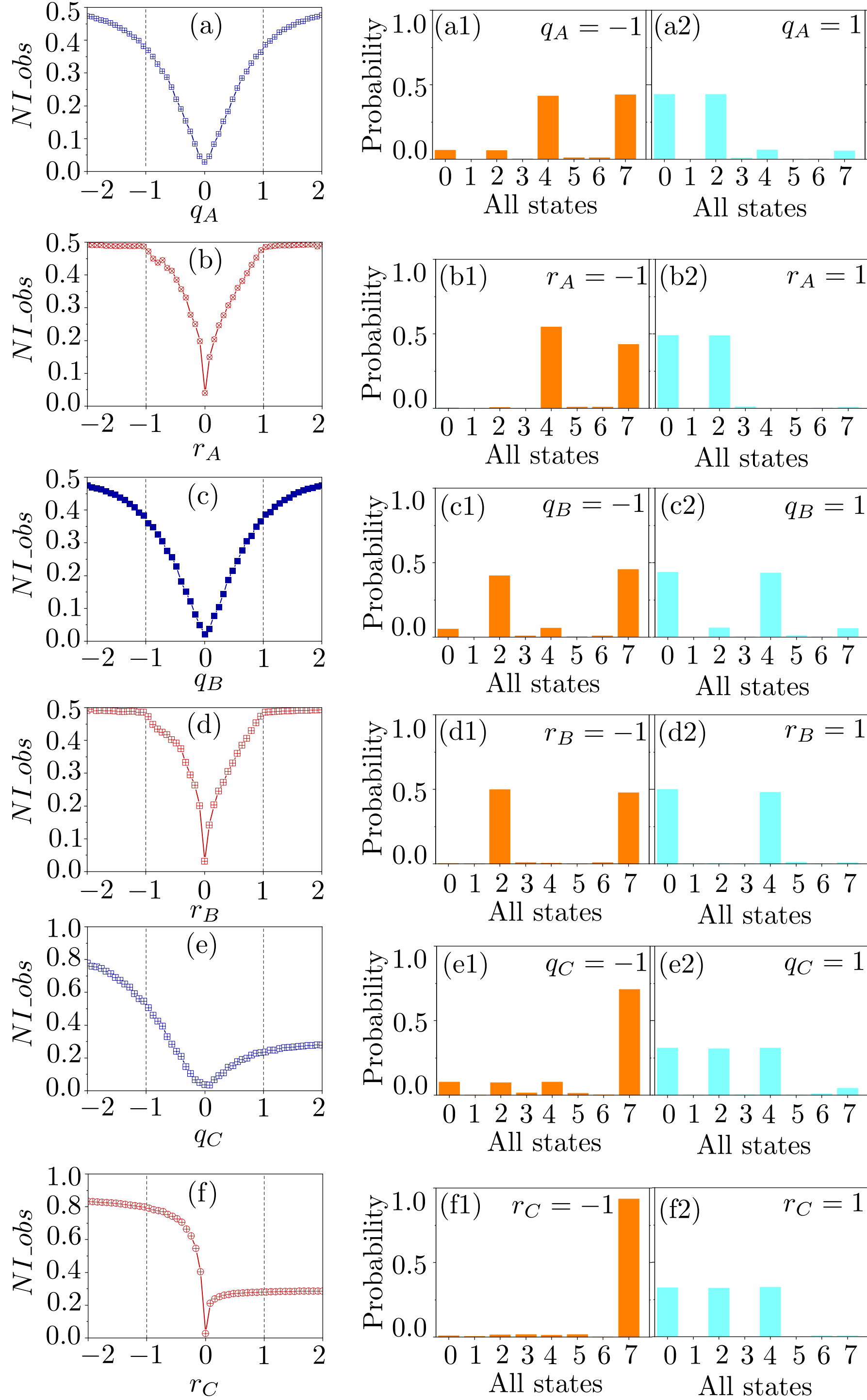


Fig. 4.6: Impact of non-ideality in a p-bit on the functionality of the AND gate. Only one term among (q_A, r_A), (q_B, r_B), and (q_C, r_C) is considered to be non-zero in each figure.

4.6(a1). The other valid states, $\{(-1,-1,-1),(-1,1,-1)\}$ get suppressed. Note that the connections among p-bits (h and J matrices) can still force the fluctuations in B and C such that invalid states are not inflated. This observation indicates the robustness of the p-bit network. Similarly, for $q_A = 1$, the states $\{(-1,-1,-1),(-1,1,-1)\}$ appear

with high probabilities, as shown in Fig. 4.6(a2). Similar behavior of p-bits getting biased towards -1 or 1 is observed when r_A, q_B, and r_B are considered separately, as shown in Fig. 4.6(b-d). In all these cases, the invalid states are found to be suppressed, indicating no actual loss in the functionality of the AND gate.

When the p-bit C has non-ideality, the impact on the NI_obs is more severe and asymmetric for positive and negative values of q_C and r_C. A vital attribute of a p-bit network is that the input and output p-bits are inherently indistinguishable, making a p-bit network invertible. It implies that when the output p-bit gets biased, it forces the input p-bits to align to a subset of valid states defined by the weighted connections (h and J matrices). For negative values of q_C and r_C, the p-bit C gets biased to the state 1. Hence, only one valid state for the case $C = 1$ (i.e., the state $\{(1, 1, 1)\}$ dominates and has a contribution of $\approx$ 80%, as shown in Fig. 4.6(e1) and (f1), respectively. Since the NI_obs quantifies RMS errors, a large non-ideality in one of the states results in a large NI_obs. For the positive values of q_C and r_C, the p-bit C gets biased to the state -1, and the states $\{(-1, -1, -1), (-1, 1, -1), (1, -1, -1)\}$ dominate with $\approx$ 33% contribution from each state, as shown in Fig. 4.6(e2) and (f2), respectively. Small errors from multiple states result in a smaller NI_obs compared to when q_C and r_C take negative values. It should be noted that the invalid states are still suppressed in all these cases. The J and h matrices are robust enough to enable the p-bit network to retain its intended functionality. It is worth pointing out that the range of q and r values in the above illustrations are chosen to explore the behavior of the p-bit network under various scenarios. In circuit implementations of a p-bit network, which are investigated in the subsequent sections, the extent of the variation in the device parameters will depend on the fabrication technology.

Next, the compensating effect of the non-idealities of a p-bit on the NI_obs for the AND gate is examined. In the previous section, it was observed that the impact of negative values of q could be canceled by a suitable positive value of r for a p-bit. Therefore, an experimental determination is carried out to assess whether such canceling effects in a p-bit can also reduce the non-idealities in an AND gate, and the results are presented in Fig. 4.7. It is found that for a given negative q_C value, a positive r_C value (shown in red) diminishes NI_obs significantly (it reduces below

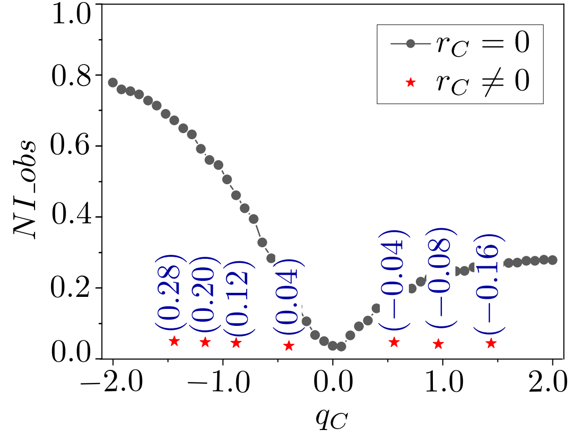


Fig. 4.7: Illustration of canceling effect of non-idealities of p-bits in an AND gate.

0.05). Similarly, for a given positive q_C value, a negative r_C value (shown in red) diminishes NI_obs significantly. These results demonstrate that the canceling effects of non-idealities can make a p-bit network inherently more immune to random process-induced variations. Further, these results suggest compensations that can be made to deal with aging-related reliability issues. The cancellation effect of non-idealities in a p-bit network may be achieved by pairing p-bits with opposite biases. The tuning can be done through the weighted output response of the connected p-bit and may cancel out the impact of non-ideality in a p-bit. However, a behavioral and circuit simulation-based analysis is required to validate the hypothesis, and is a promising direction for future research. In the next section, the impact of the non-idealities is examined, and the robustness of a larger p-bit network is analyzed for image completion.

4.4 Image Completion using a p-bit network with non-ideal p-bits

The invertible property of a p-bit network opens up new opportunities for employing p-bits in various applications [4], [88], [105], [125], [126]. However, there are accuracy and scalability challenges in the backward operation of a p-bit network. Hence, the non-idealities that can potentially impact accuracy are worth examining for the backward operation of a p-bit network. This section explores a p-bit network that completes a partial image using its invertibility and examines the impact of non-idealities.

The images of digits represented using 5×3 pixels is considered, as shown in Fig.

4.8(a). The value of a pixel is taken as -1 for violet and $+1$ for yellow. A fifteen p-bit fully connected network is trained to visit these valid states (corresponding to the ten digits) with a high probability. The training is performed using the Boltzmann machine learning algorithm demonstrated in [127]. The weights are represented using a matrix J of dimension 15×15. The elements of this matrix J_{ij} are updated as follows [93]:

$$J_{ij}(n+1) = J_{ij}(n) + \alpha(T_iT_j - V_iV_j) - \lambda J_{ij}(n) \tag{4.8}$$

where n is the training instant and $\alpha = 0.05$ is the learning rate. The training dataset (T) is 10×15, and the size of the sample data collected from the p-bit network (V) is 10000×15. The terms T_iT_j and V_iV_j are the correlation between the ith and the jth columns of T and V, and $\lambda = 1 \times 10^{-4}$ denotes the regularization parameter [128]. The training process is monitored using the Kullback-Leibler (KL) divergence. It should be noted that other variants of the Boltzmann machines, including the one presented here have been reported previously [93], [25]. However, in this study, the impact of non-idealities in a p-bit on the performance of a p-bit network is analyzed. A detailed explanation of the training algorithm is presented in [129]. Subsequently, the partial images of the digits is provided by clamping six out of fifteen p-bits, as shown in Fig. 4.8(b), as an input to the p-bit network. It is expected that the p-bit network will complete the image appropriately by assigning valid values to the rest of the nine p-bits, as shown in Fig. 4.8(c). The completed image is imperfect because p-bits also visit invalid states, though with a low probability (p-bits can be considered as a mix of violet and yellow based on their probability distribution).

To assess the accuracy of image completion, the following average root mean square (RMS) error metric E_{RMS} is defined:

$$E_{RMS} = \frac{1}{|S_N|} \sum_{S_N} \sqrt{\frac{\sum_{k=1}^{K} (\hat{P}(k) - \overset{*}{P}(k))^2}{K}} \tag{4.9}$$

where S_N is the set of test images, $K = 15$ is the number of pixels in an image, $\hat{P}(k)$ is the observed probability of being $+1$ for the pixel k in the given image (collected using 10000 time samples), and $\overset{*}{P}(k)$ is the ideal probability of being $+1$ for the pixel k in the given image. The error measure quantifies RMS error in completing the image

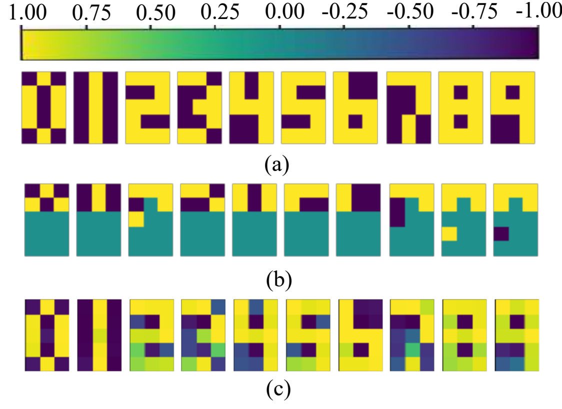


Fig. 4.8: (a) Training images for a p-bit network with fifteen p-bits. (b) Test images where six out of fifteen pixels in an image is clamped. (c) The images generated by the p-bit network implemented using ideal p-bits. The colormap at the top indicates the colors corresponding to the respective pixel values.

averaged over the entire image set S_N.

The E_{RMS} for the image completion obtained using simulation is reported in Tab. 4.4. The first row with $E_{RMS} = 0.21$ corresponds to the case when all p-bits are ideal. The second row corresponds to the case when all fifteen p-bits have a non-ideality term $q = -0.5$. As expected, E_{RMS} increases to 0.58. Similarly, the third row corresponds to when all p-bits have a non-ideality term $r = 0.1$, and E_{RMS} increases to 0.55. Thus, the non-idealities existing in all p-bits lead to a significant increase in the error. However, some errors can be nullified if compensatory non-idealities exist in the p-bits. Similar to a single p-bit case (Tab. 4.2), if all p-bits have $q = -0.5, r = 0.08$ and $q = -0.45, r = 0.1$, the E_{RMS} reduces to 0.44 and 0.43, respectively. Thus, compensatory errors can also reduce errors in a p-bit network for the backward operation and can make p-bit network more resilient to errors.

Table 4.4: E_{RMS} for ideal and non-ideal p-bit based image completion models

S. No.	Values of q and r	E_{RMS}
1	$q = 0$, $r = 0$	0.21
2	$q = -0.5$, $r = 0$	0.58
3	$q = 0$, $r = 0.1$	0.55
4	$q = -0.5$, $r = 0.08$	0.44
5	$q = -0.45$, $r = 0.1$	0.43

For the experiments in Tab. 4.4, it is assumed that the exact same non-ideality impacts all the p-bits in the system. In practice, such a scenario can occur due to

global variations that similarly impact all the fabricated devices (as assumed in corner-based analysis in traditional design flows) and can be artificially pessimistic. The non-idealities in p-bits can also be due to random local variations. The nature of such random non-idealities will strongly depend on the p-bits' implementation and manufacturing technology. In such cases, depending on the size of the p-bit network and the nature of randomness, the non-idealities can exhibit averaging or canceling effects.

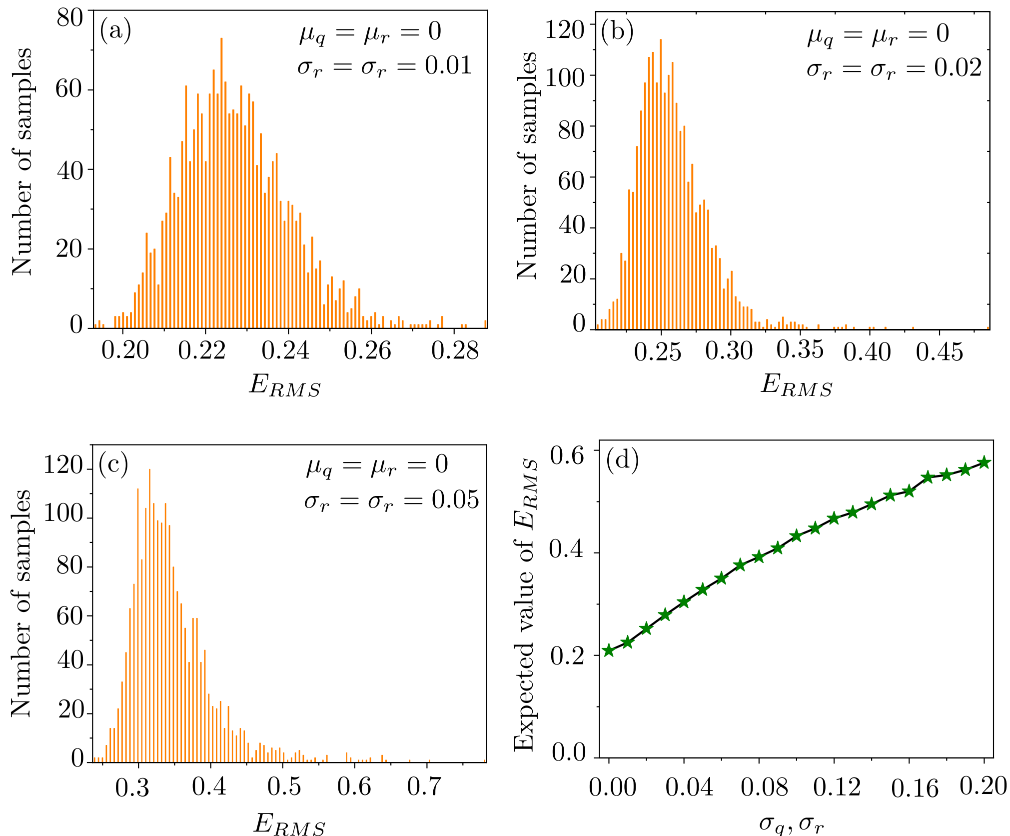


Fig. 4.9: Results of Monte Carlo (MC) simulations for (a) $\mu_q = \mu_r = 0$, and $\sigma_q = \sigma_r = 0.01$, (b) $\mu_q = \mu_r = 0$, and $\sigma_q = \sigma_r = 0.02$, and (c) $\mu_q = \mu_r = 0$, and $\sigma_q = \sigma_r = 0.05$. (d) Expected value of E_{RMS} with varying standard deviation of the q and r.

To assess the impact of random non-idealities in the p-bits more realistically, statistical methods are required to be employed. One such widely used method is the Monte Carlo (MC) simulation. Therefore, as an illustration, for the image completion task, MC simulations are performed with 2000 samples. It is assumed that the parameters q and r of the p-bits in Eq. 4.3 have normal distribution $\mathcal{N}(\mu_q, \sigma_q)$ and $\mathcal{N}(\mu_r, \sigma_r)$, respectively. Each p-bit is assumed to have non-ideality defined by the above distribution for a given MC sample point. The output of a p-bit is determined by the weighted contributions of other p-bits in the p-bit network and the applied input stimulus. The

output of a p-bit is updated according to Eq. 2.1. The output values of the p-bits are recorded once all the p-bits in the network have been updated. Fig. 4.9(a-c) shows the number of samples observed for each E_{RMS} interval. The μ_q and μ_r values are taken as zero, and the values of σ_q and σ_r are varied. It is observed that due to the random distribution of non-idealities and compensatory or adding effects, the E_{RMS} shows a wide variation. Furthermore, as σ_q and σ_r increase, the expected value of E_{RMS} also increases. Fig. 4.9(d) shows this trend of the expected value of E_{RMS} for different σ_q and σ_r values. These results suggest that, given an application, its tolerable error limit will define the constraints on the acceptable non-idealities in the p-bits. For example, the expected value and the standard deviation of the E_{RMS} obtained for the samples collected from a p-bit network and safety margins needed for the application (such as 3-sigma) will determine the tolerable non-idealities in p-bits. These constraints for the p-bits can be extended to the design criterion for the p-bit network. Thus, the application and the attributes of the p-bit network will essentially determine the design space for the p-bit implementation.

4.5 Assessing impact of variations in 1T-1MTJ network

In this section, the impact of non-idealities in a 1T-1MTJ-based p-bit on the behavior of a p-bit network is assessed using SPICE simulations. For illustration, a fully connected Boltzmann machine is considered using $N = 26$ p-bits having $N(N+1)/2 = 351$ weights and biases. Since each p-bit consists of three transistors (including two transistors for the inverter), the network consists of 78 transistors and 26 MTJ devices. The size of the network is chosen to be large enough to allow observation of its average response, and small enough to enable SPICE simulation with reasonable computational resources and runtime. For demonstration, the image classification problem on the MNIST handwritten digit image dataset is considered [130]. The original 28×28 pixel images are modified to 5×5 pixel images with bipolar pixel values to map the problem on the above 78-transistor network. Moreover, the network is trained (derived the weights and biases) for recognizing only two classes, namely letter '0' and '1'. The mapping of weights and biases to the resistors is demonstrated in [1].

The connections to a single p-bit in the circuit is illustrated in Fig. 4.10 (the com-

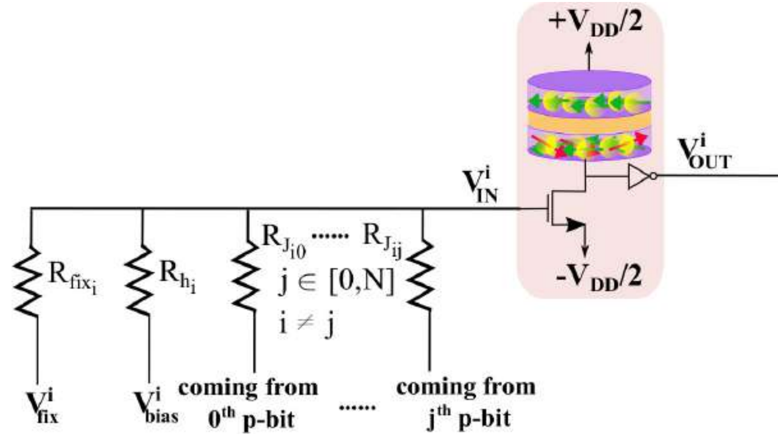


Fig. 4.10: The connection to a single p-bit in a 26- p-bit network. The p-bit shown here has a V^i_{fix} indicating the clamped input.

plete circuit is not shown here for clarity). The V^i_{fix} voltages ($+V_{DD}/2$ or $-V_{DD}/2$) are applied to the p-bits corresponding to the pixel values in the 5×5 pixel image. Hence, 25 p-bits are biased as per the given image, while the 26th p-bit is left floating and its output voltage denotes the label for the image. For the demonstration, two types of testcases, $T1$ and $T2$, were considered. In $T1$, the p-bits are clamped such that the floating p-bit is expected to be at $+V_{DD}/2$, while in $T2$, the floating p-bit is expected to be at $-V_{DD}/2$.

Firstly, the results of the SPICE simulation for the network with all device parameters fixed to their nominal values (as mentioned in Tab. 4.3) is reported in Fig. 4.11(a) and (b). The circuit was simulated for $25ns$ to obtain sufficient samples. However, the

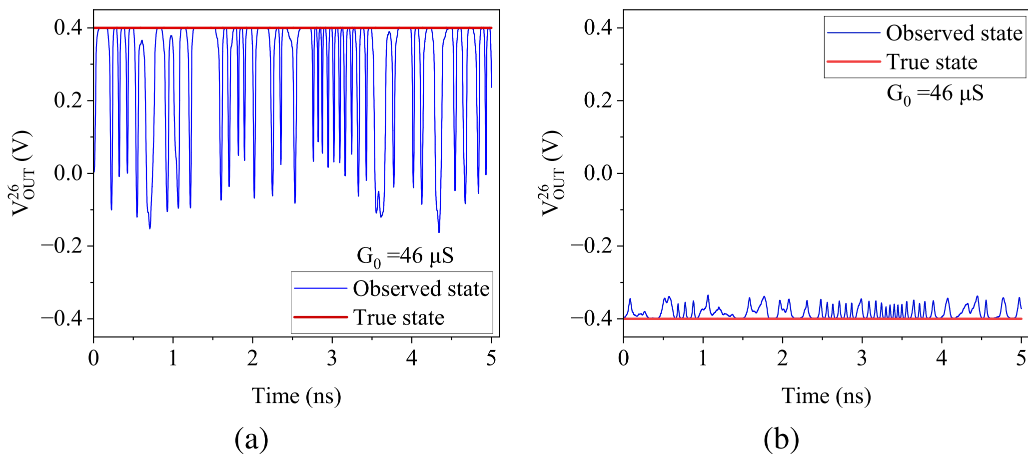


Fig. 4.11: Prediction of the image classification circuit with nominal design parameters for the test cases: (a) $T1$ (b) $T2$.

output is shown for only $5ns$ to improve clarity. It is observed that the network pro-

duces the correct response for both testcases $T1$ and $T2$ when the device parameters are at their nominal values.

Next, the impact of global process-induced variations in devices is analyzed. The global variations impact all the devices in a circuit in a similar manner. The extent and the type of device parameters that are impacted by the global variations depend on the fabrication process. However, for the sake of illustration, the SPICE simulation results when the average conductance $G_0 = 46\mu S$ changes to $30\mu S$ and $70\mu S$ are shown in Fig. 4.12. For $G_0 = 30\mu S$, the p-bits are biased towards the state $+V_{DD}/2$, and the network

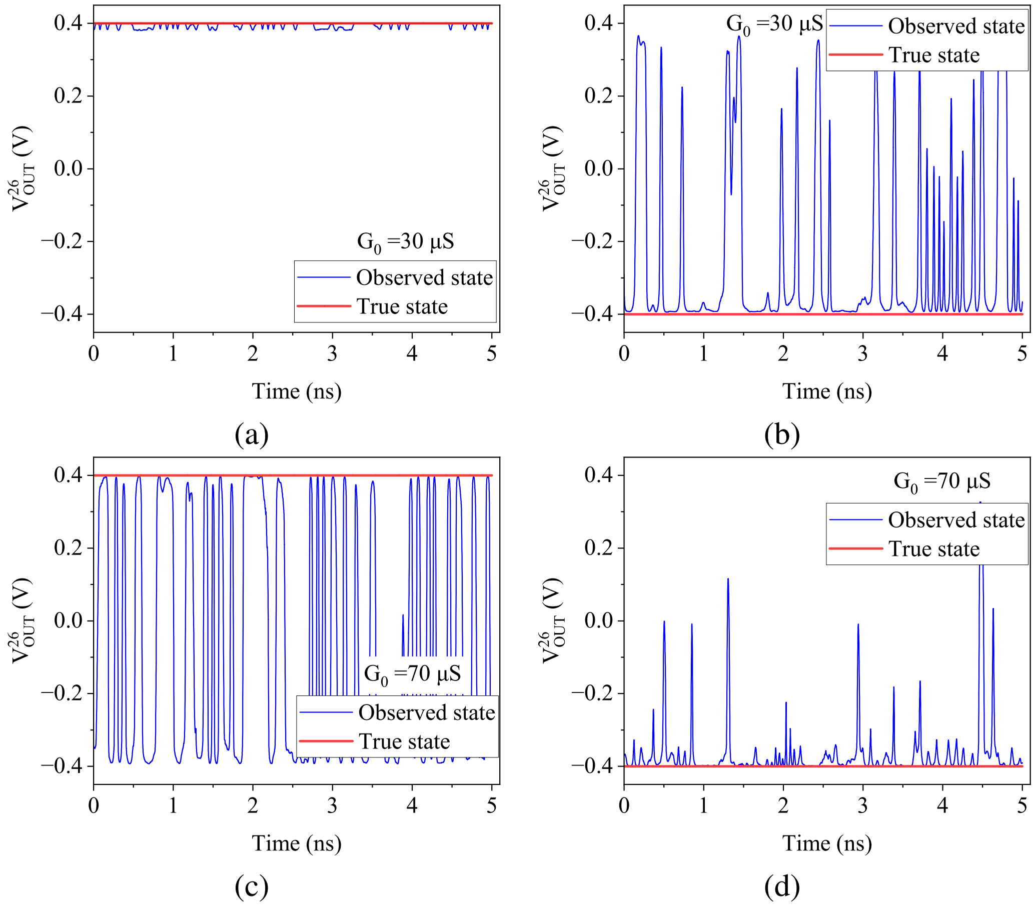


Fig. 4.12: Predictions of the image classification circuit with global variations in the G_0 values of the MTJ for each p-bit. For $G_0 = 30\mu S$, the plot shows the result for (a) $T1$ and (b) $T2$. For $G_0 = 70\mu S$, the plot shows the result for (c) $T1$ and (d) $T2$.

produces the correct prediction for the $T1$ with a high probability. However, for $T2$, due to the bias of the p-bits in the system, the network produces the correct prediction for the letter '0' with a lower probability. Similarly, for $G_0 = 70\mu S$, the p-bits are biased towards the state $-V_{DD}/2$, leading to a worse prediction for $T1$ and an excellent prediction for $T2$. These results show that global variations in a p-bit implementation

can add bias in the network, potentially making the prediction erroneous.

Next, the impact of local variations in devices on the behavior of the network is analyzed using SPICE simulations. Local variations can impact various devices in a circuit in different ways. For illustration, the local variations in the average conductance G_0 around its nominal value is considered in this work. Monte Carlo (MC) simulations with 2000 samples were carried out. For each sample, the G_0 values for all the p-bits in the network were drawn from a Gaussian distribution with an average $\mu_{G_{0_i}} = 46\mu S$ and a varying standard deviation $\sigma_{G_{0_i}}$. The output of the p-bit is encoded as follows:

$$V^i_{obs}(t) = \begin{Bmatrix} -1 & V^i_{OUT}(t) < 0V \\ 1 & V^i_{OUT}(t) \geq 0V \end{Bmatrix} \tag{4.10}$$

where $V^i_{OUT}(t)$ is the voltage at the output of the ith p-bit at the t time instant. To quantify the accuracy of the network, the error in the image classification E^{TC}_{RMS}, is calculated for the test cases $T1$ and $T2$ as follows:

$$E^{TC}_{RMS} = \sqrt{\frac{((\overline{V}^{i=26}_{obs} - V_{ideal})^2|_{T1} + (\overline{V}^{i=26}_{obs} - V_{ideal})^2|_{T2})}{2}} \tag{4.11}$$

where $V_{ideal} = +V_{DD}/2$ and $-V_{DD}/2$ corresponds to the true label for $T1$ and $T2$, respectively, and $\overline{V}^{i=26}_{obs}$ is the average value for all the time samples for the 26th p-bit (floating p-bit). Fig. 4.13(a) and (b) show the number of samples observed for each E^{TC}_{RMS} interval for $\sigma_{G_{0_i}} = 0.46\mu S$ and $\sigma_{G_{0_i}} = 9.2\mu S$, respectively. It is observed that the expected value of E_{RMS} and its standard deviation increase with the increase in variations around the nominal value.

Fig. 4.13(c) and (d) also show this trend for different $\sigma_{G_{0_i}}$ values. The geometric parameter variation of $\pm 10\%$ in M_s, H_k, α, t_{ox}, and TMR in STT-MRAM devices are considered, and their impact on the performance of read and write errors in memory devices is reported in [131]. Additionally, the extent of the impact of variations on the performance of the MTJ also depends on scaling the dimensions of the MTJ devices [131]. It should be highlighted that the extent and mechanics of process-induced variations are technology-dependent and technology-specific realistic variations are not considered in this work. Nevertheless, the above observations and trends obtained us-

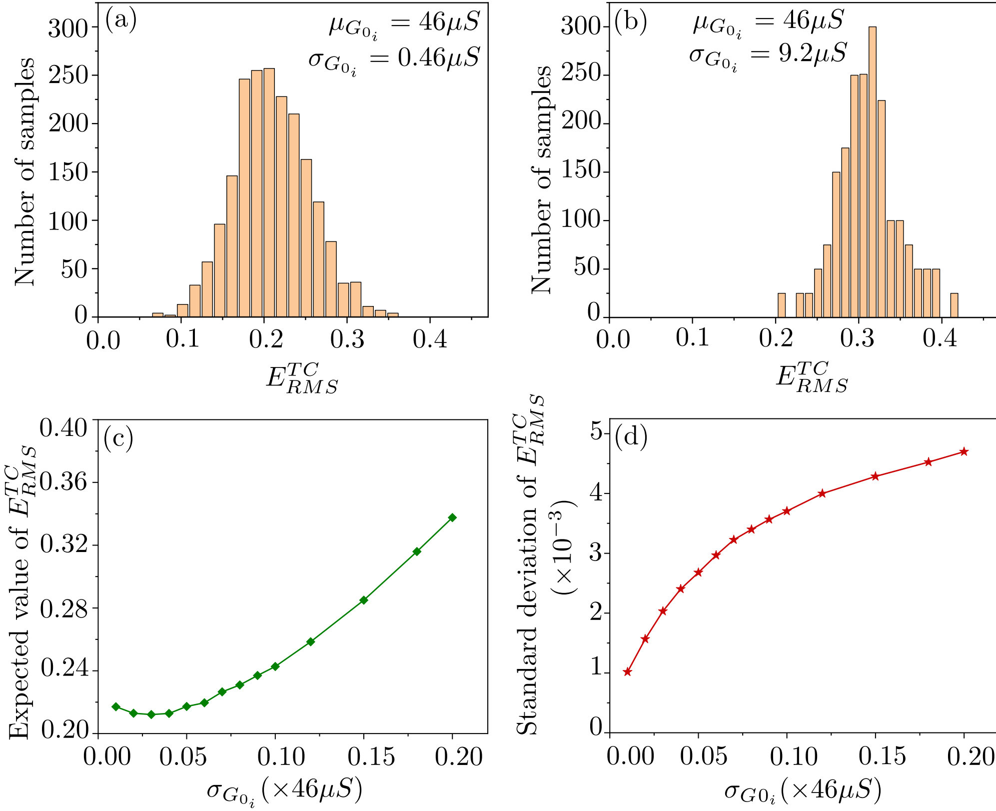


Fig. 4.13: Results of MC simulations for $\mu_{G_{0_i}} = 46\mu S$ (a) $\sigma_{G_{0_i}} = 0.01 \times \mu_{G_{0_i}}$ and (b) $\sigma_{G_{0_i}} = 0.2 \times \mu_{G_{0_i}}$. (c) Expected value and (d) standard deviation of the E_{RMS}^{TC} with varying standard deviation of the G_0 values of the MTJ.

ing SPICE simulation and the results obtained by the mathematical model (as shown in Fig. 4.9) agree and highlight the importance of considering non-idealities in assessing the performance of a p-bit network. Moreover, the results show that the proposed mathematical model can be used to gain insights into such non-idealities and analyze their impact on the performance of a p-bit network.

4.6 Conclusion

The results presented in this chapter indicate that careful consideration must be given to the non-idealities in p-bit implementations. Furthermore, circuit simulations for the 1T-1MTJ-based p-bit implementation have demonstrated how specific variations in design parameters can lead to biased outputs and affect performance in practical applications. Hence, the importance of accounting for process-induced variations in the design of p-bit networks is underscored by this work. Through the incorporation of these con-

siderations, the performance and reliability of p-bit networks can be improved, thereby facilitating their deployment in real-world applications.

Chapter 5

Fault-Tolerant Design Framework for Probabilistic-Bit (P-Bit) Systems

In addition to the non-idealities arising from process-induced variations, an added challenge of defects and faults needs to be tackled in the hardware implementation of a p-bit. In this chapter, first, the impact of stuck-at faults on the performance of a p-bit system is analyzed using the Modified National Institute of Standards and Technology (MNIST) hand-written dataset. The experiments showed that a p-bit system, in general, is resilient to stuck-at faults. However, some p-bits are more critical to the functionality of the system, and faults in these p-bits adversely impact the functionality. Using mutual information, a concept adapted from information theory, a methodology is proposed to identify these critical p-bits. Additionally, a testable p-bit is proposed to detect and identify the existence of a stuck-at fault in a p-bit integrated into a given p-bit system. A testable p-bit helps increase the observability and controllability of p-bits, and its effectiveness is demonstrated using SPICE simulations. Furthermore, the impact of some faults can be reduced by identifying and isolating a faulty p-bit from the rest of the system using the proposed isolatable p-bit. However, critical p-bits are integral to the functionality of a p-bit system and isolating them may not restore the required functionality. Hence, to enhance the robustness of a p-bit system, a p-bit with redundancy is proposed to restore the functionality of critical p-bits and make a p-bit system robust and fault-tolerant. The work presented in this chapter is published in [132].

5.1 Designing a network of p-bits

The impact of faults on the performance of a p-bit system is demonstrated by artificially introducing faults in a p-bit network trained for the image classification problem using MNIST hand-written dataset [130]. The images for letters ‘0’ and ‘1’ are extracted.

Additionally, the pixel values for all the images are transformed to bipolar values i.e., -1 and $+1$. There are 784 input p-bits such that each pixel is mapped to a particular p-bit. The output labels are encoded using two p-bits. There are no hidden p-bits in the network. A fully connected Boltzmann machine (BM) network with 786 p-bits is trained according to the following equation:

$$J_{ij}(n+1) = J_{ij}(n) + \alpha(T_iT_j - V_iV_j) - \lambda J_{ij}(n) \tag{5.1}$$

where n is the training instant and $\alpha = 1 \times 10^{-5}$ is the learning rate. The training dataset has 12665 images and the test dataset has 2115 images. Therefore, the size of the training dataset (T) is 12665×786. The mathematical model given by Eq. 2.1 is used to simulate the p-bit in the network and 10000 samples are collected from the p-bit network. Therefore, the size of the sampled data (V) is 10000×786. The terms T_iT_j and V_iV_j are the correlation between the ith and the jth columns of T and V, and $\lambda = 1 \times 10^{-6}$ denotes the regularization parameter [128]. The model is trained for 2000 iterations. It should be noted that mini-batches are not used for training because of the simplistic nature of the problem. After training, the training accuracy $= 99.5\%$ and the test accuracy $= 99.6\%$ were obtained.

Next, the impact of stuck-at faults on the performance of the p-bit network is analyzed. A few p-bits are randomly selected and their output is set to '1'. This signifies a stuck-at-1 fault such that the state of the p-bit is always '1' despite variations in the external stimuli (h_i^{fix}) and the influence of other connected p-bits.

Since the p-bit computing network is an energy-based model, the stuck-at p-bits disrupt the energy profile of the solution space. Fig. 5.1 shows that the accuracy is not degraded even if as many as ≈ 550 p-bits out of 786 p-bits have stuck-at faults. However, this observation should not be considered as a demonstration of the general resilience of the p-bit network because it could be an artefact of the simplistic problem of classifying between two letters 0 and 1. A simplistic problem is considered for this work because the motivation for this study is to investigate the relative importance of p-bits in the network, which can be easily demonstrated using this example. For a larger p-bit network and complicated applications, the accuracy can degrade significantly with less number of units having stuck-at faults, similar to other computational frameworks,

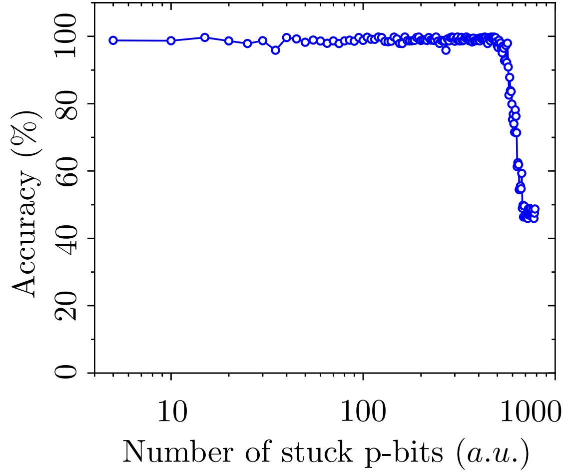


Fig. 5.1: Accuracy versus the number of p-bits with stuck-at-1 faults (chosen randomly) in the p-bit network.

such as deep neural network [133], [134]. Moreover, the extent of the impact on the accuracy differs for various p-bits, i.e., some p-bits are more critical than others in determining the overall accuracy of the network for a given application [135]. In the next section, a method is proposed to identify critical p-bits in a p-bit network for a given application.

5.2 Identifying critical p-bits in a p-bit network

The criticality of p-bits in a p-bit network is determined using the measure of information gain. To assess the importance of an individual p-bit, the importance of the connections of that p-bit with the other p-bits in the network is evaluated [129]. In an N-p-bit network, for a given training dataset T, the mutual information (MI) between two p-bit pairs i and j, given the common neighbors $k_1, k_2, ..., k_n$ is calculated according to the Eq. 5.2 [136]:

$$\begin{aligned}MI(T[i];T[j] \mid T[k_1],T[k_2],\ldots,T[k_n]) \;=\; & H(T[i] \mid T[k_1],T[k_2],\ldots,T[k_n]) \\ & +H(T[j] \mid T[k_1],T[k_2],\ldots,T[k_n]) \\ & -H(T[i],T[j] \mid T[k_1],T[k_2],\ldots,T[k_n]) \\ \forall\, i,j,k_1,k_2,\ldots,k_n \in [1,N];\; i \neq j \neq k_1 \neq k_2 \ldots \neq k_n &\end{aligned} \tag{5.2}$$

where $T[i]$ denotes a 1D-array with all the training instances for the ith p-bit i.e., $T[i][1], T[i][2], ...T[i][12665]$ and $H(T[i]|T[k_1], T[k_2], ..., T[k_n])$ is the conditional entropy evaluated according to the following equation [136]:

$$\begin{aligned} H(T[i] \mid T[k_1], T[k_2], \ldots, T[k_n]) &= H(T[i], T[k_1], T[k_2], \ldots, T[k_n]) \\ &\quad - H(T[k_1], T[k_2], \ldots, T[k_n]) \end{aligned} \tag{5.3}$$

Substituting Eq. 5.3 into Eq. 5.2 and simplifying, the following equation is obtained [136]:

$$\begin{aligned} MI(T[i]; T[j] \mid T[k_1], T[k_2], \ldots, T[k_n]) &= H(T[i], T[k_1], T[k_2], \ldots, T[k_n]) \\ &\quad + H(T[j], T[k_1], T[k_2], \ldots, T[k_n]) \\ &\quad - H(T[i], T[j], T[k_1], T[k_2], \ldots, T[k_n]) \\ &\quad - H(T[k_1], T[k_2], \ldots, T[k_n]) \\ \forall\, i, j, k_1, k_2, ..., k_n \in [1, N];\ i \neq j \neq k_1 \neq k_2 ... \neq k_n & \end{aligned} \tag{5.4}$$

Further, using the definition of joint entropy, $H(T[i], T[k_1], T[k_2], \ldots, T[k_n])$ is calculated as follows [136]:

$$\begin{aligned} H(T[i], T[k_1], T[k_2], \ldots, T[k_n]) &= -\sum P(T[i], T[k_1], T[k_2], \ldots, T[k_n]) \cdot \\ &\quad \log_2 \left(P(T[i], T[k_1], T[k_2], \ldots, T[k_n]) \right) \\ \forall\, i, k_1, k_2, ..., k_n \in [1, N];\ i \neq k_1 \neq k_2 ... \neq k_n & \end{aligned} \tag{5.5}$$

where $P(T[i], T[k_1], T[k_2], \ldots, T[k_n])$ is the joint probability distribution evaluated for all the instances in the training dataset [136].

The algorithm $Compute_Criticality$ (shown in Algo. 1) determines the criticality score of all p-bits in a given network. The inputs to the algorithm are the training dataset T, the number of pixels in a row/column of the image M (a square-shaped image is taken), and the number of p-bits N in the network. In the example considered in this work, $M = 28$ and $N = 784$. The algorithm's output is the criticality score (CS) for each p-bit in the network.

First, m_info (an $N \times N$ matrix) is initialized to collect the mutual information between various p-bit pairs (i, j), the MIS (an $N \times N$ matrix) to store the mutual

Algorithm 1 Compute_criticality: Determines criticality score of all p-bits in a network

Input: T, M and N
Output: CS

1: Initialize: m_info of size $[N, N]$, MIS of size $[N, N]$, and CS of size $[1, N]$ with all 0
2: **for** $i = 1 : N$ **do** {Loop 1}
3: $neighbors_i \leftarrow$ Get_Neighbors(i, M)
4: **for** $j \in neighbors_i$ **do** {Loop 2}
5: $link_nodes \leftarrow \phi$
6: **for** $x \in neighbors_i$; $x \neq j$ **do** {Loop 3}
7: **if** $MIS[i, x]$ > 0 and $MIS[j, x]$ > 0 **then**
8: $link_nodes \leftarrow link_nodes \cup x$
9: **end if**
10: **end for** # Loop 3
11: $m_info[i, j] \leftarrow MI(T[i]; T[j]|T[link_nodes])$
12: **end for** # Loop 2
13: $max \leftarrow$ Get index of p-bit $j \in neighbors_i$ with the maximum $m_info[i, j]$
14: **if** $m_info[i, max] > 0$ **then**
15: $MIS[i, max]$ = $MIS[max, i] \leftarrow m_info[i, max]$
16: **end if**
17: **end for** # Loop 1
18: **for** $i = 1 : N$ **do**
19: $CS[i] = \sum_{q \in [1,N]; i \neq q} MIS[i, q]$
20: **end for**
21: **return** CS

information-based score between p-bit pairs and CS (a $1 \times N$ array) to store the criticality score of the p-bits with all zero elements. The $link_nodes$ corresponds to $k_1, k_2, ..., k_n$ in the Eq. 5.4, and the $neighbors_i$ denotes the surrounding pixels of the ith p-bit. To reduce the complexity in computing mutual information, a 3×3 pixel block sliding over the entire image is considered, and the mutual information is computed for the pixels lying within this window. All the p-bits are iterated over, and their neighbors are found using the algorithm $Get_Neighbors$. For the p-bit pair (i, j), the common neighbors are identified and stored in $link_nodes$. In step 11, the mutual information is calculated between the p-bit pair (i, j) according to the Eq. 5.4, where $T[link_nodes]$ is same as $T[i], T[k_1], T[k_2], \ldots, T[k_n]$. Among the p-bit pairs, the pair with the maximum mutual information is picked and the mutual information-based score MIS is updated with the corresponding mutual information value. In this way, the mutual information-based score between different p-bit pairs is computed for all the N p-bits in the network. It should be noted that the mutual information between the p-bit pairs (i, j) can be zero

if the pixel values remain the same for the entire training dataset. For example, the pixels at the boundary of the image remain the same for all the letters, and therefore, are not the deciding factor in distinguishing the letters 0 and 1. Finally, in step 19, the criticality score of a p-bit is determined by adding the mutual information-based score of all its connections in the network. The algorithm $Get_Neighbors$ (shown in Algo. 2), computes the set of indices of p-bits lying in the neighborhood of the ith p-bit within a 3×3 window.

Algorithm 2 Get_Neighbors: Determines common neighbours

Input: i and M
Output: $neighbors_i$
1: $neighbors_i \leftarrow \phi$
2: $r = \left\lfloor \frac{i-1}{M} \right\rfloor + 1$
3: $c = (i-1) \bmod M + 1$
4: **for each** $dr \in \{-1, 0, 1\}$ **do**
5:     **for each** $dc \in \{-1, 0, 1\}$ **do**
6:         **if** $dr \neq 0$ **or** $dc \neq 0$ **then**
7:             $r' = r + dr$ and $c' = c + dc$; $r', c' \in [1, M]$
8:             $neighbor_pbit = (r' - 1) * M + c'$
9:             $neighbors_i \leftarrow neighbors_i \cup neighbor_pbit$
10:         **end if**
11:     **end for each**
12: **end for each**
13: **return** $neighbors_i$

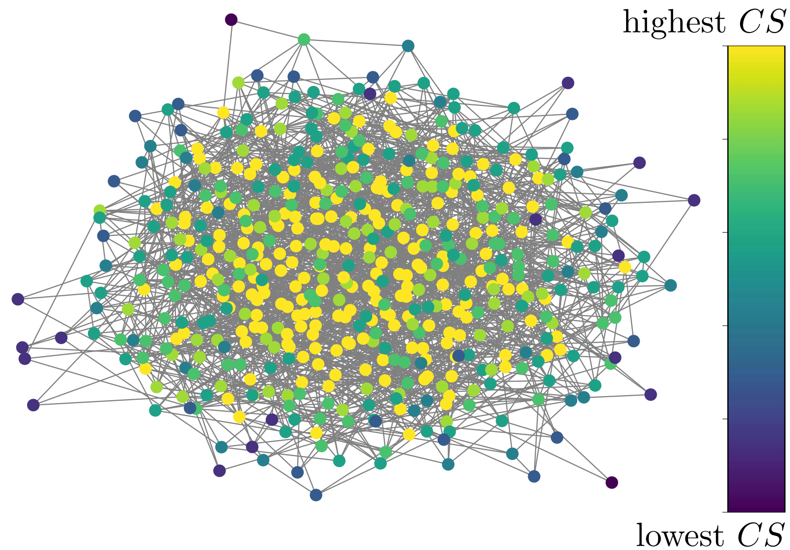


Fig. 5.2: Identification of the critical p-bits based on mutual information. The criticality of a p-bit is encoded by color, where yellow represents the critical most p-bit. There are 784 input p-bits in the network, however, for illustration, a 500 p-bit network is shown here.

Fig. 5.2 shows the criticality score of 500 p-bits in a network. The CS value of a p-bit is encoded with colors, yellow to violet, where the p-bit shown in yellow color has the highest CS value.

It should be noted that the output p-bits are the most critical p-bit from the perspective of faults, and are not considered in the above computation. If an output p-bit has a fault, the functionality can be recovered only by replacing the faulty p-bit, as discussed in the later sections.

Once the critical p-bits are identified, the performance of the p-bit network can be analyzed under the condition that the critical p-bits exhibit stuck-at faults. Fig. 5.3 shows that the accuracy degrades significantly if a fault occurs in the critical p-bits (red curve) in contrast to the same number of randomly selected p-bits (black curve). Hence, detecting faults in critical p-bits and making a p-bit system tolerant to them is important. It should be noted that the results presented in this work focus on the stuck-at-1 fault. Given the binary nature of the problem, the analyses for the stuck-at-(-1) fault gave similar results. However, for a multiclass classification problem, the analysis of the impact of stuck-at-(-1) is required [137]. The analysis for the transient faults is not performed in this work, and is intended for future work.

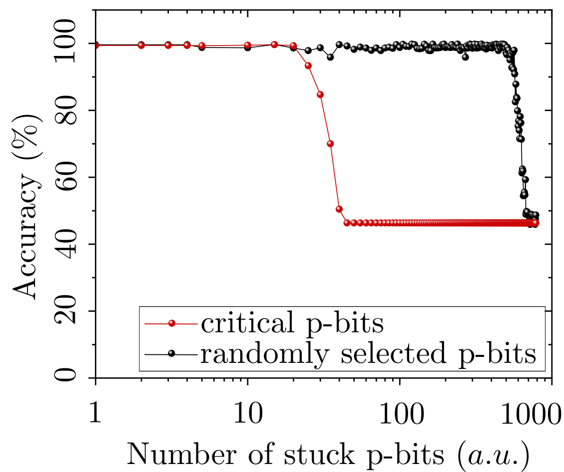


Fig. 5.3: Accuracy versus the number of p-bits with stuck-at-1 faults in the p-bit network: when critical p-bits are chosen (red) and when randomly chosen (black).

The method of identifying critical p-bits outlined above determines the criticality of p-bits using the available training dataset T. Nevertheless, the mutual information framework can be potentially extended to other p-bit applications for assessing the relative importance or criticality of individual p-bits in the network. To demonstrate this,

an example is presented in which a p-bit circuit is used to emulate a Bayesian Network (BN), allowing the correlation between real-world variables to be derived from the output response of the p-bits in the network.

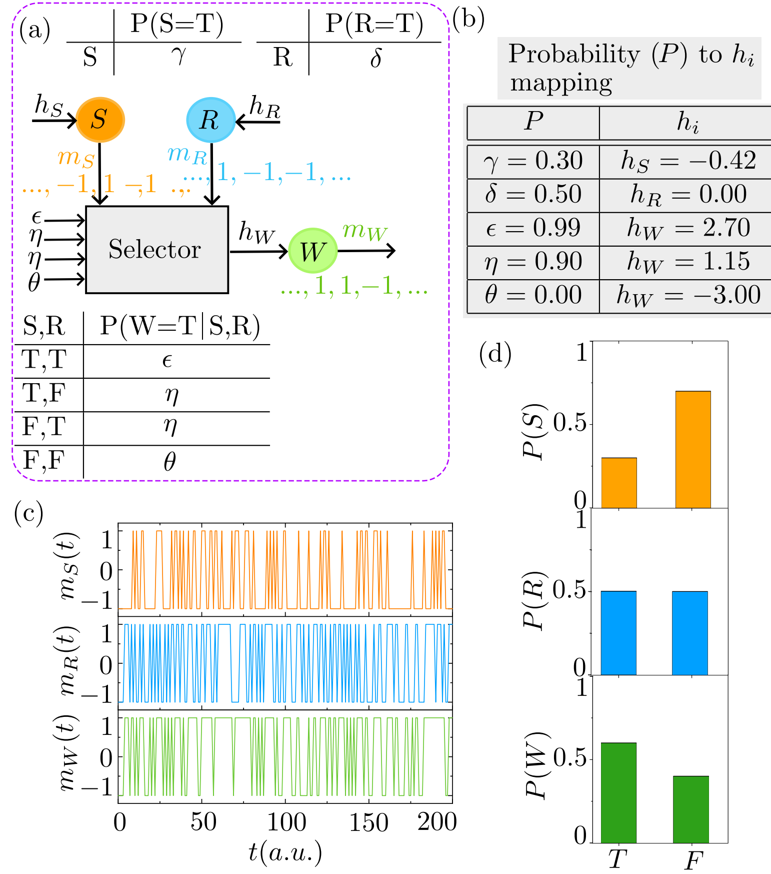


Fig. 5.4: (a) An example Bayesian Network (BN) is shown with three nodes Sprinkler (S), Rain (R), and Wet-Grass (W). There are directed connections from the parent (S and R) to the child node (W) according to CPTs. A selector block maps the probability (P) to h_S, h_R, and h_W as shown in table in (b). (b) Probability to h_i mapping to determine the required h_i that achieves P. (c) The corresponding output response for the p-bits S, R, and W. (d) The corresponding probability distribution for all the p-bits.

For illustration, a simple BN with three nodes: Sprinkler (S), Rain (R), and Wet-Grass (W) is considered. Each node takes a binary state True (T) or False (F) according to the local probability table known as a Conditional Probability Table (CPT), as shown in Fig. 5.4(a). In this example, the directed arrows indicate that the Sprinkler (S) and the Rain (R) nodes are the parent nodes to the child node Wet-Grass (W). The mapping of CPT to h_i values is given in the table shown in Fig. 5.4(b). For p-bits S and R, the values of h_S and h_R are chosen such that m_S and m_R have probability distributions similar to the CPT of S and R, respectively. For the p-bit W, the h_i is selected by the

selector block that finds the appropriate h_W corresponding to different combinations of the states of S and R such that the probability distribution of W corresponds to the CPT of W [125], [138].

Fig. 5.4(c) shows the output response of each node in the p-bit network. The probability distribution of the child node (W) is obtained by marginalizing over the parent nodes (S and R), as given by the following equation [125]:

$$P(W) = \sum_{S,R} P(W|S,R)P(S)P(R) \tag{5.6}$$

Using Eq. 5.6, the probability distribution $P(W = T) = 0.59$ is analytically obtained for the W node. In Fig. 5.4(d), the probability distribution for both states of all nodes is shown, and it is evident that the simulation results are in alignment with the analytical predictions.

Next, MI between the nodes is used to determine the relative criticality of p-bits in the BN. Given the CPT for a BN, the MI between the two nodes given by Eq. 5.4 is modified using the relation between the entropy and probability distribution [136]. The MI between the nodes S and W is given by Eq. 5.7:

$$MI(W;S) = \sum_{i=\{T,F\}} P(S=i) \times \sum_{j=\{T,F\}} P(W=j|S=i) \times \log_2 \left(\frac{P(W=j|S=i)}{P(W=j)} \right) \tag{5.7}$$

Using Eq. 5.7, the mutual information values are calculated as follows: $MI(W;S) = 0.19$ and $MI(W;R) = 0.38$. Therefore, it is concluded that p-bit R has a stronger influence on p-bit W as compared to p-bit S. Consequently, a fault in p-bit R is expected to exhibit more detrimental impact on the result. The validity of this deduction is confirmed by quantifying the impact of a stuck-at fault in the p-bit by evaluating the resulting deviation in the probability distribution using the error metric E_{RMS}, as follows:

$$\begin{aligned} E_{\text{RMS}} = \Bigg(\frac{1}{2} \Big(& (P(W=T)|_{\text{SA-1}} - P(W=T)|_{\text{NF}})^2 \\ & + (P(W=T)|_{\text{SA-(-1)}} - P(W=T)|_{\text{NF}})^2 \Big) \Bigg)^{1/2} \end{aligned} \tag{5.8}$$

where $P(W=T)|_{\text{NF}}$ is the probability of p-bit $W = T$ when the p-bit W does not have a stuck-at fault. $P(W = T)|_{\text{SA-1}}$ and $(P(W = T)|_{\text{SA-(-1)}}$ is the probability obtained

when a parent p-bit of the p-bit W has a stuck-at 1 and (-1) fault, respectively. Table 5.1 presents the E_{RMS} values for stuck-at faults in p-bit R and p-bit S, indicating that the E_{RMS} value is higher when a fault occurs in p-bit R compared to p-bit S. This illustrates that mutual information can be utilized to assess the relative criticality of p-bits in Bayesian inference applications. However, it is important to note that the example presented here is a simple BN. For more complex BNs, the method may require appropriate adjustments to account for intricate dependencies. The methodology to calculate CS is also expected to change with the depth of the network, for example, for a deep neural network. In a recurrent neural network (RNN), the output of a neuron is fed back to the neuron itself or to another neuron at different time steps. Therefore, the proposed methodology to calculate CS needs to be re-evaluated to include the time step dependencies. Next, the circuit-level solutions are proposed to detect faults in critical

Table 5.1: Impact on probability distribution due to stuck-at fault in various p-bits in the given BN

P-bit	**Type of Fault**	**P(W=T)**	E_{RMS}
S	stuck-at-1	0.95	0.27
	stuck-at-(-1)	0.46	
R	stuck-at-1	0.94	0.34
	stuck-at-(-1)	0.27	

p-bits and make a p-bit system tolerant to faults.

5.3 Fault detection in a p-bit network

A critical p-bit can be deep inside a p-bit network, including in hidden layers, and detecting and identifying a fault in it using the network's input-output behavior can be challenging. This challenge can be overcome by making a p-bit more controllable and observable through the primary inputs and outputs of the network. Based on the 1T-1MTJ STT MRAM-based p-bit [2], a testable p-bit is proposed that is capable of detecting stuck-at faults.

5.3.1 Testable P-bits

Fig. 5.5(a) shows the proposed testable p-bit. A 2-to-1 multiplexer and a D-type scan cell is added to increase the controllability and observability of a p-bit. A D-type scan cell is widely used in scan design methodologies for digital circuits and its internal structure is shown in Fig. 5.5(b). However, since p-bits work in bipolar logic mode, the

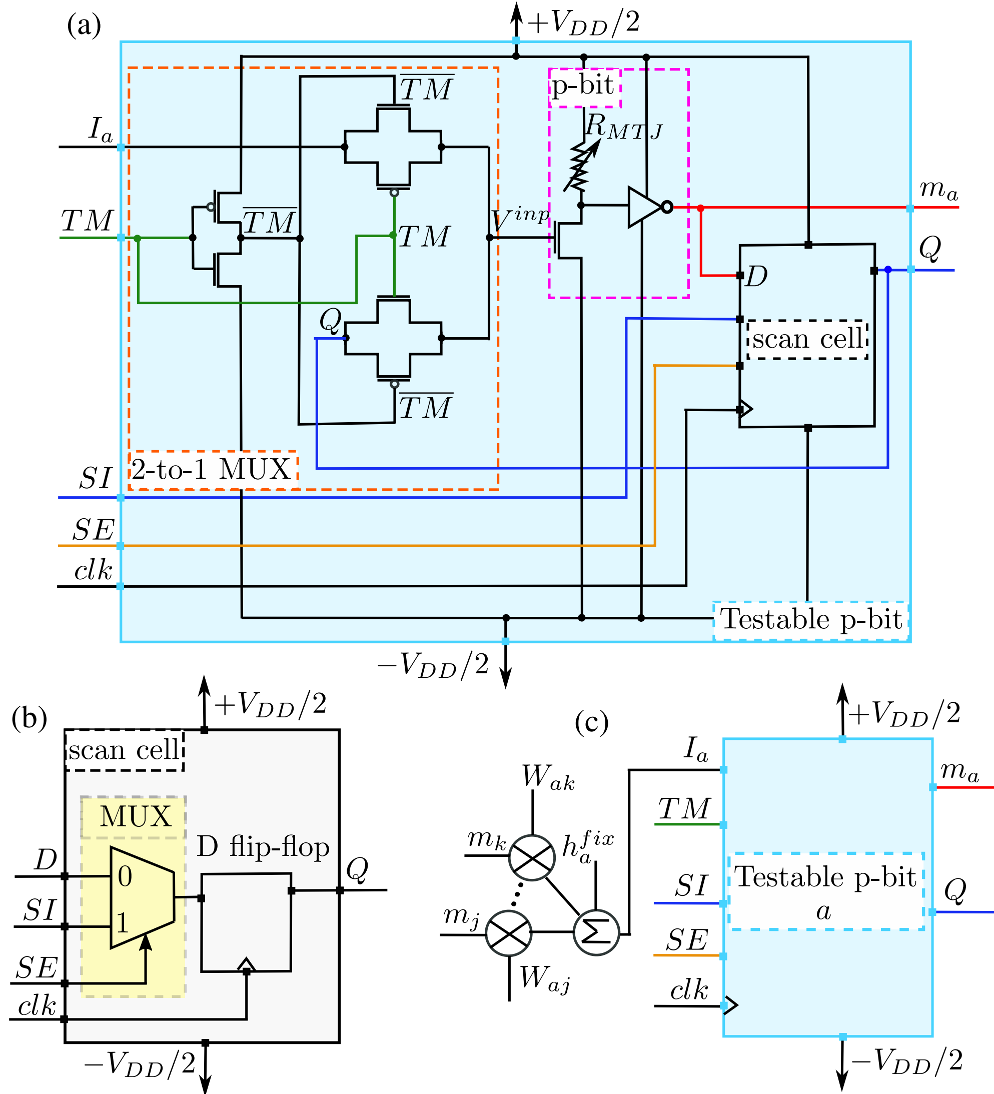


Fig. 5.5: Proposed framework for fault detection (a) testable p-bit (b) internal details of scan cell (c) connecting a testable p-bit with other p-bits.

2-to-1 multiplexer and the D-type scan cell are designed to work in the voltage range $[-V_{DD}/2, +V_{DD}/2]$, with $-V_{DD}/2$ and $+V_{DD}/2$ correspond to -1 and $+1$, respectively.

The testable p-bit has the following inputs: I_a (input), TM (test mode), SI (scan input), SE (scan enable), and clk (clock). The input I_a receives inputs from the neigh-

boring p-bits according to Eq. 2.2. The inputs TM and SE determine the mode in which the circuit works, as explained in Tab. 5.2. A p-bit network works normally in the functional mode (TM and SE are at $-0.4V$). During testing TM is kept at logic 1. The SE input is set to logic 1 when the scan cell of the p-bit is to be loaded with a given value applied at the SI pin. When TM is at logic 1, the p-bit receives the input V^{inp} from the output of the scan cell Q. The testable p-bit also has a clk input, despite probabilistic logic being an asynchronous system, for synchronizing the testing operation and is relevant only when TM is set to logic 1.

Table 5.2: Modes of operation of testable/fault-tolerant p-bit system

Mode	TM (V)	SE (V)
Functional	-0.4	-0.4
Test (shift)	0.4	0.4
Test (capture)	0.4	-0.4

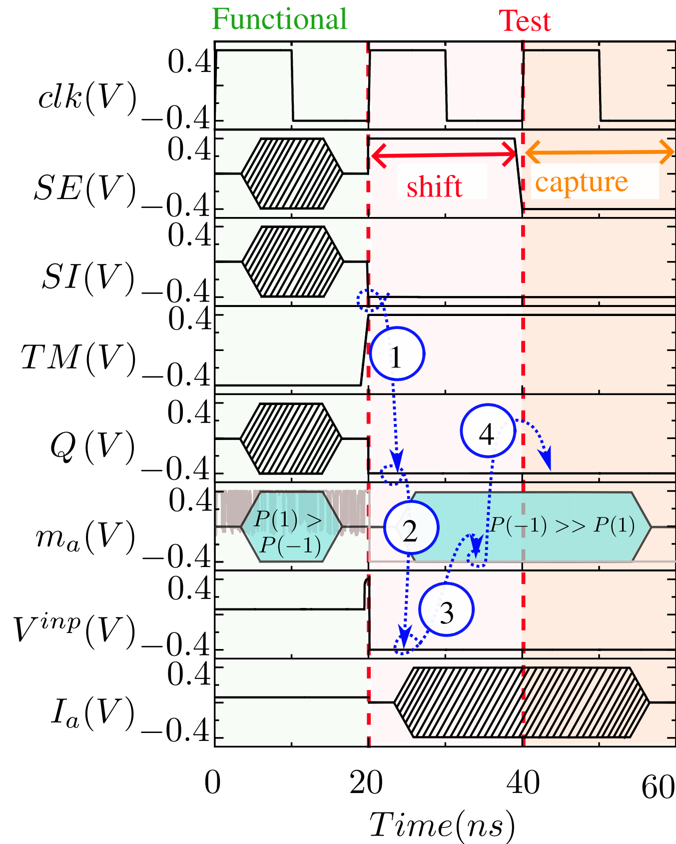


Fig. 5.6: Validation of proposed testable p-bit circuit using SPICE simulations. The m_a values are shown in terms of the probability of occurrence of state $1(0.4V)$ and $-1(-0.4V)$.

The proposed p-bit is implemented using 14 nm HP-FinFET technology [40]. Fig.

5.6 shows the timing diagram for the testing p-bit a obtained using SPICE simulation [116]. From $t = 0$ to $20ns$, p-bit a is in the functional mode ($TM = -0.4V$). The input I_a is routed to V^{inp} and the output of the p-bit, m_a, is determined by V^{inp}. Since $I_a > 0V$, in this case of simulation, the probability $P(1) > P(-1)$. If $I_a < 0V$ is applied, $P(1) < P(-1)$ is observed. For large values of $I_a \approx 0.4V$, $P(1) \approx 1$ is obtained, and the p-bit is observed to virtually provide a deterministic logic 1, as shown in Fig. 2.4(a). In the functional mode, the values at SE, SI, clk and Q are irrelevant.

At $t = 20ns$, $TM = 0.4V$, $SE = 0.4V$ and $SI = -0.4V$ are set. This takes the p-bit to the test (shift) mode. The value at SI input propagates to Q on the arrival of the clk edge (arrow 1), which further propagates to V^{inp} (arrow 2) due to $SE = 0.4V$. Thus, the p-bit receives the value at SI instead of I_a. The output of the p-bit m_a responds to V^{inp} and exhibits $P(-1) \approx 1$ (arrow 3).

At $40ns$, $SE = -0.4V$ is set, and the p-bit moves to the capture mode. The output produced by m_a is latched at the Q pin (arrow 4). If the p-bit was faulty and stuck-at 1, m_a would exhibit $P(1) \approx 1$, instead of $P(-1) \approx 1$ (arrow 3), and the faulty behaviour will be recorded by the testable p-bit at the output of the scan cell. Thus, the proposed testable p-bit can detect stuck-at 1 fault in the p-bit. Similarly, it can detect stuck-at -1 fault, if $SI = 0.4V$ is provided in the scan mode. Moreover, the fault recorded at the scan cell output is observed on a primary output of the network, as explained below.

5.3.2 Detect and identify faulty p-bits in a network

The connection of testable p-bits in a chain (shift register) configuration is demonstrated for the purpose of applying the required test pattern and observing the response at the primary inputs and outputs, respectively. This is illustrated using two testable p-bits, a and b, as shown in Fig. 5.7. The SI input of the testable p-bit a receives the Q output of the previous testable p-bit in the chain. Similarly, the SI input of the testable p-bit b is connected to the Q pin of p-bit a, and the Q pin of b goes to the SI of the other testable p-bit. The SI pin of the first testable p-bit in this chain is connected to a primary input through which the required test pattern is shifted in the shift mode, and the Q pin of the last testable p-bit in this chain is connected to a primary output through which the response received is read by a testable p-bit in the capture mode.

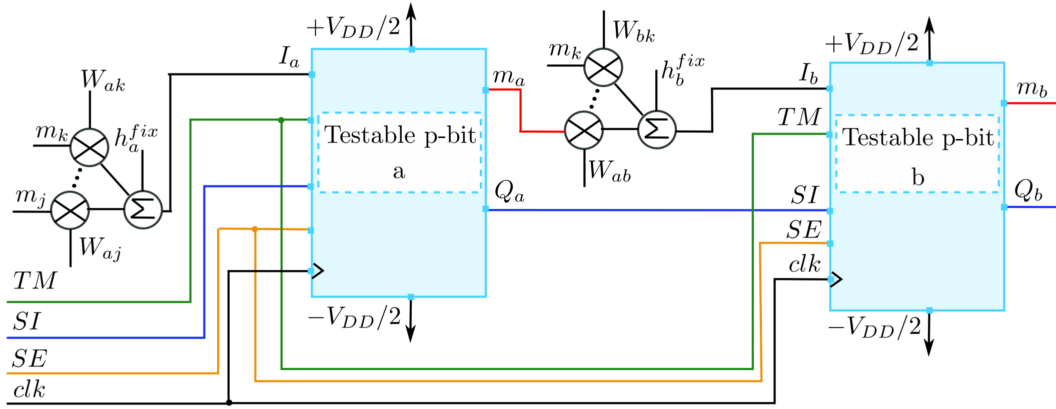


Fig. 5.7: Connections of two testable p-bits a and b.

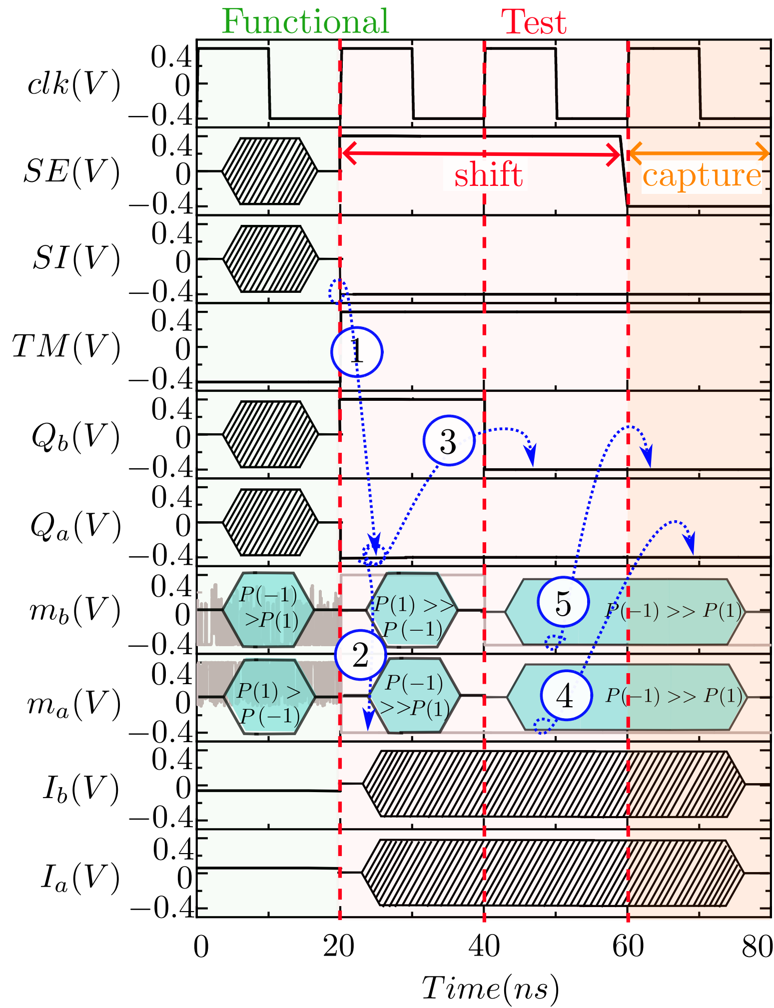


Fig. 5.8: Validation of testable p-bits a and b shown in Fig. 5.7 using SPICE simulations.

Fig. 5.8 shows the SPICE simulation results obtained for the above circuit. From $t = 0ns$ to $20ns$, $TM = -0.4V$ is set, and the circuit works in the normal functional mode. The values obtained at m_a and m_b, have a greater probability of being in state 1

and -1, respectively, because I_a and I_b are set to values greater than and less than $0\,V$, respectively. The values at other pins are irrelevant in this case.

At $t = 20ns$, $TM = 0.4V$ and $SE = 0.4V$ is set, and the circuit goes into the scan mode. At the arrival of the clock edge, the value at the SI pin (state -1) of the p-bit a goes to its Q pin (arrows 1 and 2), and at the next clock edge, this value goes to the Q pin of the p-bit b. Thus, the testable p-bits is loaded with the required test pattern.

At $t = 60ns$, $TM = 0.4V$ and $SE = -0.4V$ is set, and the circuit goes into the capture mode. The response of the p-bits m_a and m_b gets captured at Q_a (step 4) and Q_b (step 5), respectively (arrows 4 and 5). It is assumed that both p-bits are non-faulty, hence the state -1 is observed for both of them. If any p-bit was stuck-at 1, the value at its Q pin would have been different. Next, the circuit is operated in the scan mode and the observed response is shifted out at the Q pin to the primary output where the faulty p-bit is identified by its respective position in the scan chain. The next test pattern can be simultaneously shifted in this mode, if needed.

It should be noted that, in general, an ensemble of p-bit output responses is required to estimate the probability of occurrence of state 1 and -1. However, to detect stuck-at faults, $+V_{DD}/2$ or $-V_{DD}/2$ is applied, which practically biases it to the required state and taking one sample would be sufficient.

In the next section, a circuit-based solution is presented to make the p-bit system more tolerant to faults by isolating the faulty p-bits from the network.

5.4 Isolatable p-bit

When a stuck-at fault occurs in a p-bit, it is often better to isolate that p-bit from the rest of the network rather than allow it to produce a constant stuck-at value. For example, if a stuck-at fault is detected in the p-bit i, it is isolated by forcing $m_i = 0$. It should be noted that for bipolar logic, stuck-at faults produce either $m_i = -V_{DD}/2$ or $m_i = V_{DD}/2$. Forcing $m_i = 0$ is equivalent to making $J_{ij} = 0; \forall(j \in (1, N), i \neq j)$, hence, isolating the faulty p-bit from the network. Fig. 5.9 shows the accuracies obtained when the faulty p-bits are isolated from the network trained for the image classification problem illustrated in Section 5.1. It is observed that isolating the faulty p-bit helps improve the accuracy in certain cases for a p-bit network.

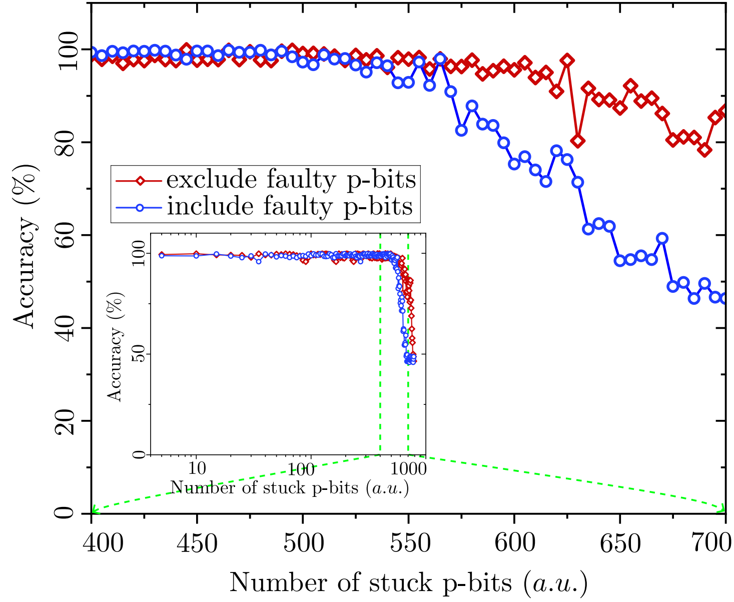


Fig. 5.9: Comparison of accuracy obtained for a p-bit network with faulty p-bits and without faulty p-bits (assumed to be removed using isolatable p-bits). The inset shows the complete plot on a logarithmic scale.

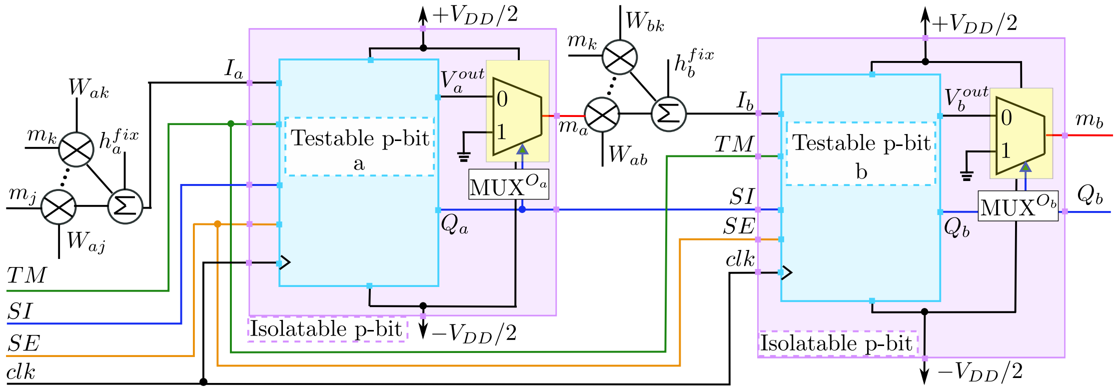


Fig. 5.10: Proposed isolatable p-bits and their connections.

Next, an isolatable p-bit is proposed, as shown in Fig. 5.10. A 2-to-1 multiplexer forces its output to $0V$ when needed, thus, isolating it from the rest of the network. For instance, if a fault is detected in the p-bit a, Q_a is forced to $0.4V$ through the scan chain and make the multiplexer select the grounded input or $0V$.

Figure 5.11 shows the simulation results obtained for two p-bits shown in Fig. 5.10 assuming that there is a stuck-at 1 fault in the p-bit a. From $t = 0ns$ to $20ns$, the circuit is in the functional mode, m_a is stuck-at-1 (stuck-at $0.4V$), while m_b follows I_b. At $t = 20ns$, the circuit is taken to the test mode, and in two clock cycles the test pattern $Q_a = -0.4V, Q_b = -0.4V$ is shifted in through the SI primary input (arrows 1, 2, and 3). Next, at $t = 60ns$ the circuit is taken to the capture mode, and $Q_a = 0.4V$

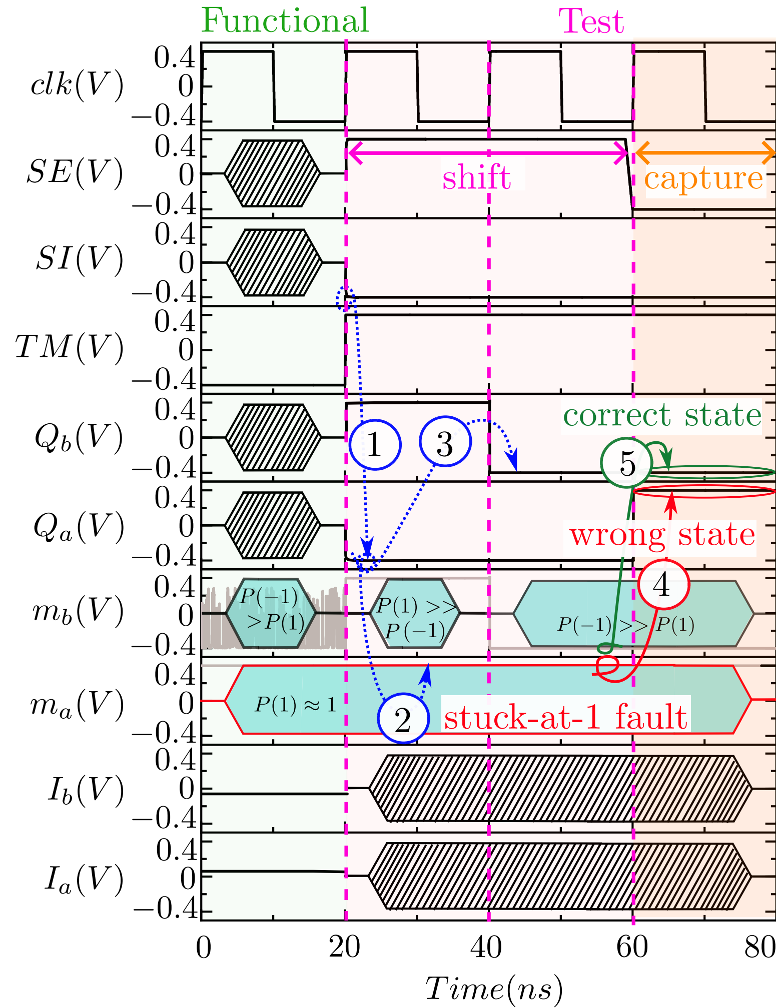


Fig. 5.11: Validation of isolatable p-bits shown in Fig. 5.10 using SPICE simulations. The p-bit a is identified to be faulty during the test mode.

and $Q_b = -0.4V$ are latched (arrows 4 and 5). By shifting these values to the primary output and comparing them with the golden values $Q_a = -0.4V$ and $Q_b = -0.4V$, a fault in the p-bit a is detected.

Next, it is demonstrated that the faulty p-bit a can be isolated by loading the isolatable p-bits with an appropriate mask pattern. To isolate or connect a p-bit, its Q pin is set to either $0.4V$ or $-0.4V$, respectively, by shifting the appropriate mask pattern through the scan chain. The mask pattern is loaded before taking the circuit to functional mode. Fig. 5.12 shows the simulation results for the p-bits a (with fault) and p-bit b (without fault). In the first clock cycle, $Q_a = -0.4V$, and in the next clock cycle $Q_a = 0.4V, Q_b = -0.4V$ (arrow 1). The value at m_a is forced to $0V$ (arrow 2), and p-bit a gets isolated, while the p-bit b produces output based on I_b.

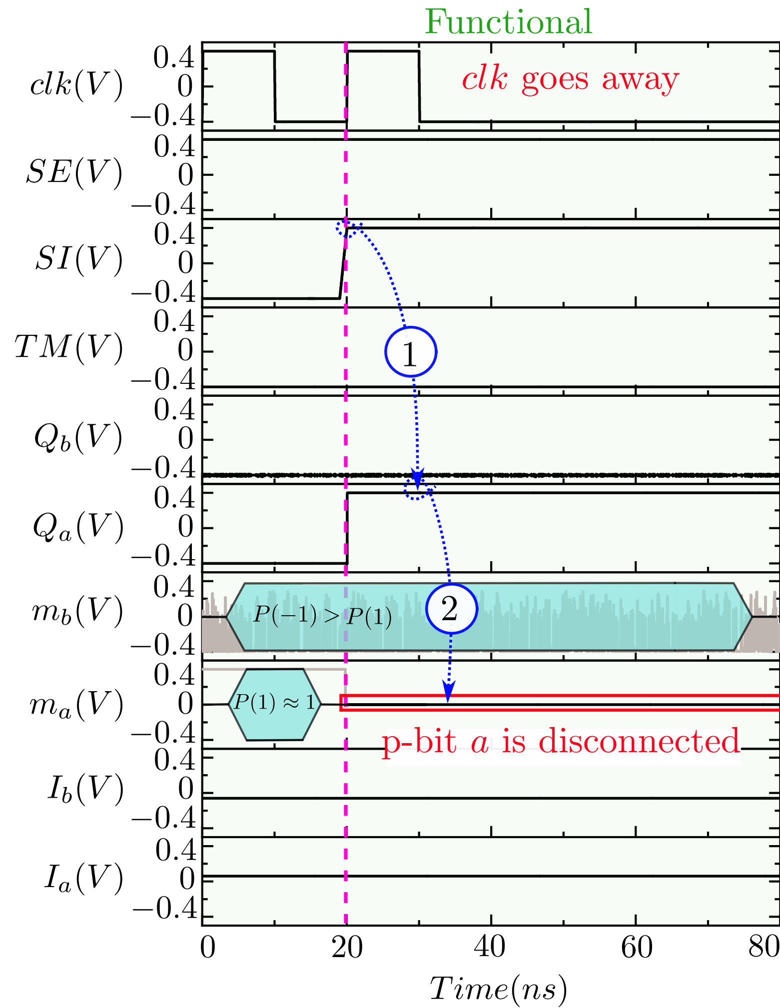


Fig. 5.12: SPICE simulation results demonstrating isolation of the faulty p-bit a.

Isolating faulty p-bits can help improve accuracy in some cases. However, for extremely critical p-bits, replacement of a faulty p-bit with a fault-free p-bit may be required [139], [140]. In the next section, the use of redundant p-bits is proposed to make a p-bit system more tolerant of faults.

5.5 Fault-tolerant P-bit

Fig. 5.13 shows the circuit diagram of the proposed fault-tolerant p-bit. It has a redundant p-bit b' besides a testable p-bit b. When a fault is detected in b, the redundant p-bit b' can deliver the required functionality. The input I_b drives both the p-bits b and b'. The multiplexer selects the outputs from one of the p-bits b or b' based on the value of the select line Q_b. On detecting a fault in the p-bit b, Q_b is forced to logic 1 using an

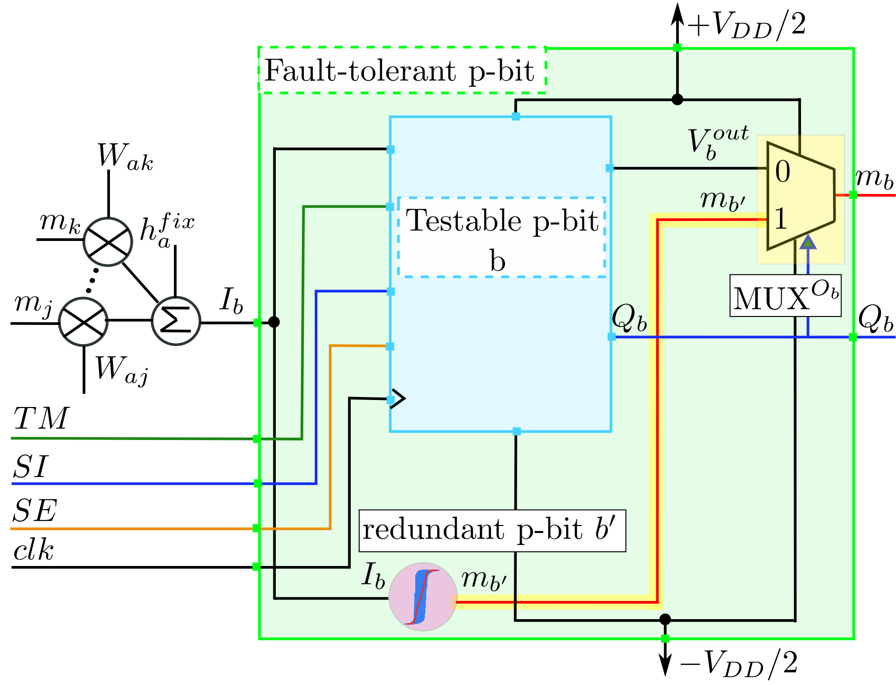


Fig. 5.13: Proposed fault-tolerant p-bit.

appropriate mask pattern, thus producing a fault-free result using the redundant p-bit b'.

Fig. 5.14 shows the timing diagram for the redundant p-bit obtained using SPICE simulations. It is assumed that the p-bit b has a stuck-at-1 fault. From $t = 0ns$ to $20ns$, the circuit is in the functional mode ($TM = -0.4V$). It is observed that V_b^{out} (highlighted in red) has a constant value of $0.4V$ irrespective of the I_b due to the fault. The output m_b is also stuck at 1 due to select signal $Q_b = -0.4V$. The fault in b is detected using the methodology outlined in Section 5.3. Subsequently, Q_b is loaded with the mask pattern $0.4V$ to enable b' to produce the output. The circuit is taken to the scan mode ($TM = 0.4V$ and $SE = 0.4V$), and then $SI = 0.4$ is applied, and the value is propagated to Q_b on the arrival of the next clock edge (arrow 1). If the fault-tolerant p-bit was part of a long chain, multiple clock cycles are needed to load the masking patterns at the Q_b pin. The masking value allows the fault-tolerant p-bit to produce m_b using the fault-free p-bit b' (arrow 2) through the highlighted yellow path in Fig. 5.13. Subsequently, the clock is removed and the circuit is moved to the functional mode, and it is expected to produce the correct result with the help of fault-free p-bit b'. It should be noted that the functionality cannot be restored if b' also has a stuck-at fault.

The above results demonstrate the effectiveness of testable, isolatable, and fault-free

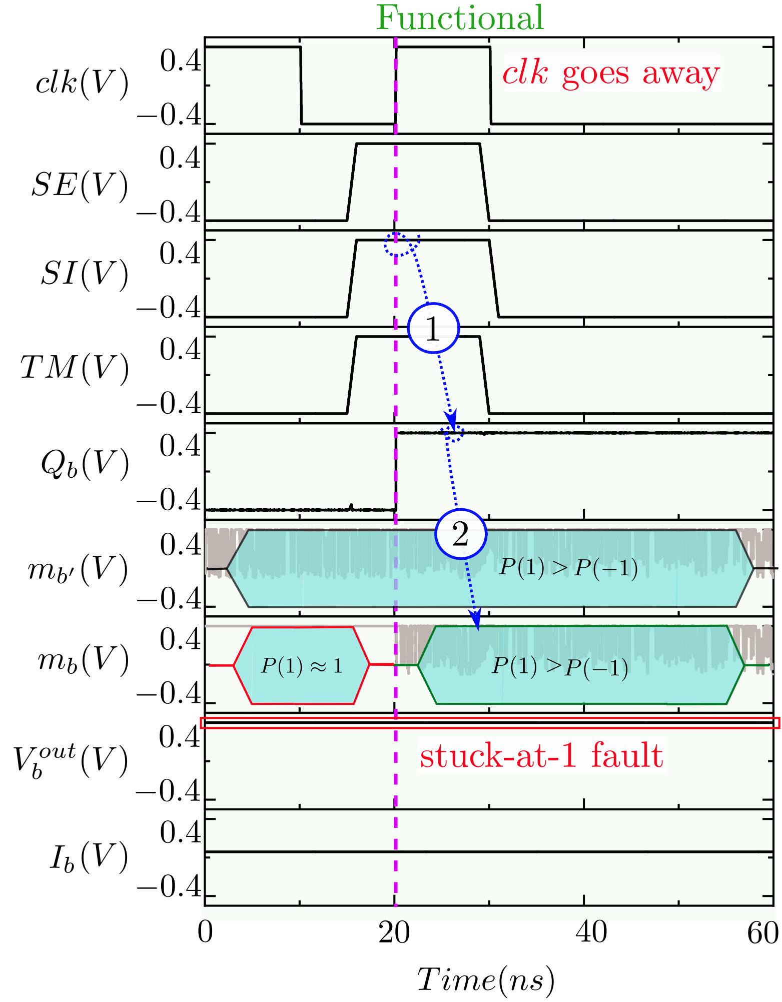


Fig. 5.14: Validation of fault-tolerant p-bit using SPICE simulations.

p-bits in making a p-bit system more robust. However, the proposed p-bits consume extra resources. Tab. 5.3 reports the extra number of resources needed by the proposed p-bits in comparison to the conventional p-bits. For the $14nm$ HP-FinFET technology, the area of the FinFET is estimated as $A_{FinFET} = L_{fin} \times (n_{fin} \times t_{fin} + (n_{fin} - 1) \times f_p)$, where L_{fin} is the length of the fin, t_{fin} is the thickness of the fin, f_p is the fin pitch, and n_{fin} is the number of fins [143]. In this work, $L_{fin} = 18nm$, $t_{fin} = 10nm$, and $n_{fin} = 1$, which gives $A_{FinFET} = 180nm$ [40]. Therefore, for testable p-bits, the area overhead of 24 transistors is $4.32 \times 10^{-3} \mu m^2$. It should be noted that an accurate estimate of the area footprint requires layout design, which accounts for design rules, routing congestion, and parasitics that are not captured at the circuit-level analyses. Thus, there is a trade-off between the reliability and the area/resource cost. This trade-off can be carefully evaluated as follows. For a given application, first, a set of critical

Table 5.3: Comparison of area overhead for testable p-bits

Device	# of Transistor	# of MTJ
Conventional p-bit	3T[a]	1
Testable p-bit	6T[b]+3T[a]+ $6T$[b]+ 12T[c]= 27T	1
Isolatable p-bit	27T +6T[b]= 33T	1
Fault-tolerant p-bit	27T +6T[b]+3T[a]= 36T	2

[a] T: Transistor. Here 3T consists of 1 NMOS and 1 inverter [2].
[b] MUX: Multiplexer. 2-to-1 MUX consists of 6 transistors [141].
[c] D flip-flop consists of 12 transistors [142]

p-bits must be determined. Then, a subset of these critical p-bits can be converted to fault-tolerant p-bits based on their criticality scores, acceptable area overhead, and reliability targets.

The above trade-off is demonstrated in Fig. 5.15 for the network discussed in Section 5.1. If 640 p-bits of the network are randomly chosen to be stuck at 1, the accuracy falls from 99.6% to 62.5%. However, if just 20 most critical p-bits are converted to fault-tolerant p-bits the accuracy increases to more than 90%. Moreover, if the 30 most critical p-bits are converted to fault-tolerant p-bits, the accuracy restores to 99%. Hence, the proposed fault-tolerant p-bits can improve the fault tolerance of a p-bit system, with minimal area overhead. It should be noted that a fault in the output p-bits in a p-bit network will result in erroneous results irrespective of other p-bits, and they must always be converted to fault-tolerant p-bits. Additionally, the above trade-off is application-specific and an appropriate trade-off can be explored for a given application using the proposed framework.

Next, we examine the feasibility of implementing post-deployment fault correction in a p-bit system that has been designed and fabricated for a specific application. By leveraging the built-in self-test (BIST) strategy, which is widely used in integrated circuits, faults in a p-bit can be detected during runtime after fabrication. A BIST controller can be designed to internally generate test patterns, propagate them through the scan chain, and identify faulty p-bits by comparing the observed response with the expected response. Moreover, based on the position of the erroneous bit in the observed bit-pattern, the BIST controller can determine the appropriate mask pattern required to activate the redundant p-bit b' in the proposed fault-tolerant p-bit shown in Fig. 5.13.

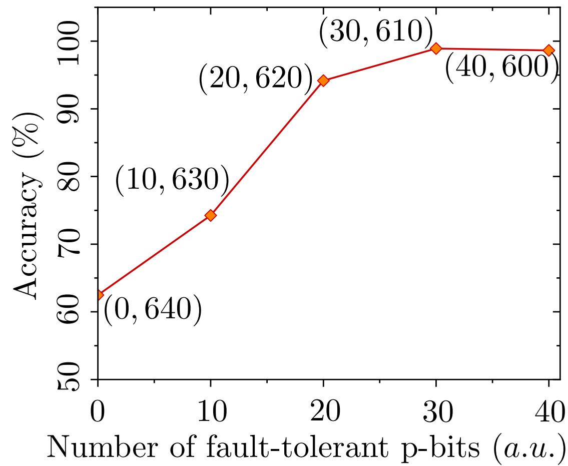


Fig. 5.15: Accuracy versus number of fault-tolerant p-bits. The values (M, N) indicate that there are M fault-tolerant p-bits and randomly-chosen N p-bits with stuck-at-1 fault.

Thus, the proposed p-bit system can recover from faults after the system has been in operation, enhancing its reliability and robustness. However, the overhead introduced by the BIST controller in the p-bit system should also be considered.

It should be noted that the proposed methodology enables static fault tolerance, where the target application is known prior to fabrication and critical p-bits are identified and converted to fault-tolerant p-bits during the design phase before hardware is built. Implementing a fault-tolerant methodology is more challenging for versatile systems in which the application is dynamically configurable because critical p-bits cannot be determined before fabrication in this case. One possible approach to introduce dynamic fault tolerance is to configure the p-bit array such that a certain proportion of p-bits (for example, 80% conventional p-bits and 20% fault-tolerant p-bits) are reserved without being hardwired during fabrication. Instead, their interconnections would be electrically programmable post-fabrication. While this method could simplify initial hardware design, supporting flexible, application-specific mapping and interconnectivity would introduce significant complexity and system-level overhead, possibly greater than the overhead of the fault-tolerant p-bits alone. Addressing these challenges would require a detailed system-level investigation. Thus, enabling dynamic fault tolerance in versatile p-bit systems would be a promising direction for future research.

5.6 Performance evaluation of Proposed P-bits

In this section, the delay and power dissipation of the proposed p-bits is compared with the conventional p-bit.

5.6.1 Delay

The delays exhibited by the proposed p-bits in the functional and test modes are different. Hence, these behaviours are reported separately.

Functional mode

In the functional mode, the comparison of delay incurred in testable p-bits is reported in Tab. 5.4. For illustration, a load of four inverters is considered at the output of the p-bit. The input I_i to the p-bit is a step input with a rising edge at $t = 0s$ and a falling edge at $t = 2.5ns$. For each p-bit, the rise (tp_{LH}) and fall delay (tp_{HL}) are evaluated and the total average delay is reported. For a conventional p-bit, the total average delay is $5.2ps$. For the testable p-bit, an additional delay of the 2-to-1 multiplexer is introduced, while for the isolatable and fault-tolerant p-bits, additional delays of two 2-to-1 multiplexers are introduced. The fault-tolerant p-bit exhibits a slightly increased delay due to the increased load on the driver due to the redundant p-bit. It should be noted that the scan cells do not directly affect the delay of the proposed p-bits in the functional mode because they do not appear in the input-to-output path in this mode.

Table 5.4: Comparison of delay for proposed p-bits in functional mode

Device	**Total Average Delay** (ps)
Conventional P-bit	5.2
Testable P-bit	5.6
Isolatable P-bit	7.0
Fault-tolerant P-bit	7.4

Test mode

In the test mode, the main component of the delay is the clk-to-Q delay in the scan cell during a clock cycle. The clk-to-Q delay is observed to be $10.2ps$ in the p-bit

implemented in this work (assuming a clock slew of $1ps$ and a load of four inverters at the output). For a p-bit network with N testable p-bits connected in a chain, a total of $2N + 1$ clock cycles are required in testing: N clock cycles in loading the test bits into the scan cells, one cycle in recording the response, and N clock cycles in observing the response at the output of the chain [144].

5.6.2 Power dissipation

In a p-bit circuit, the power is dissipated in the MTJ branch (MTJ and NMOS) and the inverter branch (its internal PMOS and NMOS). For the MTJ branch, the equivalent resistance $R = R_{MTJ} + R_{NMOS}$ and for the inverter branch, the drain-to-source resistances R_{NMOS} and R_{PMOS} are responsible for the power dissipation. The resistance of the MTJ is given by the following equation:

$$R_{MTJ} = \left(G_0 \left(1 + m_z \frac{TMR}{2 + TMR} \right) \right)^{-1} \tag{5.9}$$

where $G_0 = (G_P + G_{AP})/2$ is the average conductance of the MTJ, $TMR = (G_P - G_{AP})/G_{AP}$ is the tunnel magnetoresistance, m_z is the magnetization of the magnet, and G_P and G_{AP} are the parallel ($m_z = 1$) and antiparallel ($m_z = -1$) conductance of the MTJ [2]. For $TMR = 200\%$ and $G_0 = 46\mu S$, the R_P and R_{AP} calculated according to the Eq. 5.9 is equal to $14.5K\Omega$ and $43.5K\Omega$, respectively.

Functional mode

In the functional mode, the power dissipation values for different types of p-bits are listed in Tab. 5.5. A pulse input I_i with the voltage levels $0.4V$ and $-0.4V$ and a time period of $1ns$ is applied to the p-bit. The rise and fall time of the I_i is taken as $1ps$. When $I_i = 0.4V$, the NMOS in the MTJ branch is turned ON and the power is dissipated in that branch. As the I_i switches to $-0.4V$, a switching power is dissipated in the inverter. However, when $I_i = -0.4V$ is reached, the NMOS in the MTJ branch is turned OFF, therefore, the power dissipated in this branch becomes negligible [32]. In the testable p-bit, for $TM = 0$, the 2-to-1 multiplexer also contributes to the total power dissipation. Similarly, in the isolatable and fault-tolerant p-bits, the two 2-to-

1 multiplexers also contribute to the power dissipation. In the fault-tolerant p-bit, the redundant p-bit also contributes to the power dissipation. The extra power dissipated in the redundant p-bit can be eliminated by forcing the NMOS in the MTJ branch to switch off when it is inactive.

Table 5.5: Comparison of Power dissipation in the proposed p-bits

Device	MTJ branch (μW)	Inverter branch (μW)	2-to-1 MUX (μW)	Total average power (μW)
Conventional P-bit	12.9	0.1	-	13.0
Testable P-bit	12.9	0.1	0.1	13.1
Isolatable P-bit	13.0	0.2	0.1	13.3
Fault-tolerant P-bit	25.7	0.3	0.1	26.1

Test mode

In the test mode, power is dissipated within the scan cell (shown in Fig. 5.5(b)) also during the scan operation. With a clock period of $20ns$, the average power dissipation in the scan cell is measured to be $115nW$ for the p-bits simulated in this work.

The additional delay and power dissipation in the proposed p-bit implementations, as highlighted above, underscore the importance of considering these factors, alongside area overhead, when converting a p-bit into a fault-tolerant p-bit.

5.7 Conclusion

In this chapter, the impact of stuck-at faults on the accuracy of a p-bit system is analyzed. An information-theory-based methodology is proposed to evaluate the criticality of various p-bits in the network. Further, the proposed p-bits can detect faults and their effectiveness is validated using simulations. P-bit architectures that enable isolation and fault tolerance are also proposed, and it is demonstrated how these architectures can mitigate accuracy loss caused by faults in a p-bit system. Furthermore, the proposed fault detection framework can be integrated into hybrid probabilistic computing systems, which have demonstrated significant improvements in computational speed and energy efficiency that are several orders of magnitude higher than conventional computing paradigms. The proposed architectures have resource/area overheads, which can

be kept acceptable by prioritizing critical p-bits. However, the impact of faults in the interconnections between p-bits and other parts of the p-bit system, such as peripherals is not considered. For practical applications, these should also be considered in future work.

A notable feature of the proposed p-bits and fault detection/repair methodology is that they can be employed in the field to detect/repair faults due to ageing and other reasons, besides detecting/repairing faults arising due to fabrication. Moreover, in this work, the proposed methodology is demonstrated using nanomagnet-based p-bit. However, it could be employed for p-bits implemented using other stochastic elements which are susceptible to being stuck in one of the states.

It is important to highlight that the proposed fault-tolerant design framework is applicable when the target application is known in advance, allowing critical p-bits to be selectively converted into fault-tolerant p-bits before fabrication. In the general case of a versatile p-bit system, where the application is determined post-fabrication, the current framework would not be directly applicable and would require significant adaptation. Addressing this broader scenario remains an important direction for future work.

Chapter 6

Image completion using Sparse Probabilistic Computing network

In this chapter, a methodology to achieve sparsity in the probabilistic computing network and the trade-off between performance and network optimization is investigated in detail. The work presented in this chapter is published in [129].

6.1 Network optimization oriented realization of sparse p-computing network

In this section, the optimization of a sparse probabilistic computing network is explored. A case study involving image completion for letters 0 to 9 of size 5×3 pixels is presented. Furthermore, the performance of probabilistic computing with a fully connected probabilistic network and a sparsely connected probabilistic network is evaluated. An attempt is made to extend the energy efficiency claim of the probabilistic computing. As discussed in [97], significant contributions to the area and power dissipation of the circuit are made by weighted connections, which are reduced in a sparsely connected probabilistic network.

A sparsely connected p-bit network to implement the image completion task is proposed to be derived using three methods. The first two methods are conventional: Step-Wise (SW) removal of weights and the Binary Search (BS) method. Third, a weight connection prediction method is proposed using Information Gain (IG) based on information theory.

6.1.1 Deriving Fully Connected probabilistic computing network

The training algorithm TRAIN_PC for fully connected p-bit network is demonstrated in Algo. 3. The probabilistic computing network is trained on a ten-digit images for letters 0 to 9 of size 5×3 pixels. Every letter contributed 10% to the ideal probability distribution, P_{data}. The K p-bit probabilistic computing network generated a probability distribution, P_{model}, that has the contribution of all the 2^K states as shown in Algo. 4. The discrepancy between the probabilities P_{data} and P_{model} during training is evaluated by the Kullback-Leibler (KL) divergence according to the Eq. 6.1 [145]:

$$KL(P_{data}||P_{model}) = \sum_{x=1}^{N} P_{data}(x) * \log_2 \frac{P_{data}(x)}{P_{model}(x)} \tag{6.1}$$

where N is the no. of instances in the digit image set TT, and $P_{data}(x)$ and $P_{model}(x)$ are the probability of occurrence of xth state in TT and VV, respectively. Here, VV is the samples from the p-bit network, collected using Algo. 4 (which is described later). The input to the algorithm TRAIN_PC and their values are shown in Tab. 6.1

Table 6.1: Input parameters for Algo. 3

Symbol	Description	Value
TT	Digit image set	-
N	Number of instances in TT	10
S	Number of instances in VV	10000
K	Number of p-bits	15
α	Learning rate	0.05
λ	Regularization parameter	1×10^{-4}
ε	Tolerance value	9×10^{-4}
$max_iterations$	Maximum number of iterations	1500
P_{data}	Ideal probability distribution	-
WS	Window size	10
$Connections_J_{ij}$	J matrix	-

In step (1) all the parameters are initialized. Step (3) collects samples from the PC_network. Next, KL divergence is calculated in step (4). Steps (5) to (7) calculates the average KL divergence value over WS. If the average change in the KL divergence is less than ε, step (9) stops the training and goes to step (24). Otherwise, the update of J and h is carried out through the steps (12) to (22). Finally, at step (24) , the algorithm returns the h and J matrices.

Algorithm 3 Training probabilistic computing network: TRAIN_PC

Input: TT, N, S, K, α, λ, ε, $max_iterations$, P_{data}, WS, and $Connections_J_{ij}$.
Output: h and J

1: **Initialize:** $KL_present \leftarrow \infty$, $KL_previous \leftarrow \infty$, J of size $[K \times K]$, h of size $[1 \times K]$ and KL_values of size $[1 \times max_iterations]$ with all 0.
2: **for** $no_iter = 1 : max_iterations$ **do**
3: (VV, P_{model}) = PC_network(J, h, S, K)
4: $KL_values[no_iter] \leftarrow KL$ acc. to Eq. 6.1
5: **if** $no_iter \,\%\, WS = 0$ **then**
6: $KL_present = \frac{1}{WS}\sum_{w=1}^{WS} KL_values[no_iter - w]$
7: **end if**
8: **if** $|KL_present - KL_previous| < \varepsilon$ **then**
9: break /* stop the training /*
10: **end if**
11: $KL_previous \leftarrow KL_present$
12: **for** i, j in $Connections_J_{ij}$ **do**
13: $pos_ij = \frac{1}{N}\sum_{t=1}^{N} TT[t][i] * TT[t][j]$
14: $neg_ij = \frac{1}{S}\sum_{t=1}^{S} VV[t][i] * VV[t][j]$
15: $J[i][j] = J[i][j] + \alpha * (pos_ij - neg_ij) - \lambda * J[i][j]$
16: $J[j][i] = J[i][j]$
17: **end for**
18: **for** $i = 1 : K$ **do**
19: $pos_i = \frac{1}{N}\sum_{t=1}^{N} TT[t][i]$
20: $neg_i = \frac{1}{S}\sum_{t=1}^{S} VV[t][i]$
21: $h[i] = h[i] + \alpha * (pos_i - neg_i)$
22: **end for**
23: **end for**
24: **return** h and J

The samples from probabilistic computing network VV are collected from PC_network as shown in Algo. 4. The input to the algorithm are J, h, S, and K, and returns VV and P_{model}. In step (4) all the p-bits are initialized to a random state. The samples are independent of the order of p-bit updates, hence the shuffling of $update_sequence$ in step (6) ensures the random update of the p-bits. The input to the p-bit i, I_i and the output, m_i are calculated in steps (8) and (9). Step (11) collects all the samples in VV. Steps (12) and (13) evaluates the P_{model}. Finally, VV and P_{model} is returned and the algorithm is concluded in step (15).

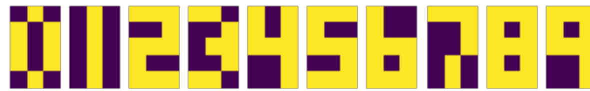
Algorithm 4 Collect sample from probabilistic computing network: PC_network

Input: J, h, S, and K
Output: VV, P_{model}

1: **Initialize** : VV of size $[S \times K]$, m of size $[1 \times K]$, $update_sequence$ of size $[1 \times K]$, and P_{model} of size $[1 \times 2^K]$ with all 0.
2: $\beta = 1$.
3: $update_sequence[i] \leftarrow i \qquad \forall i \in [1, K]$
4: $m[i] \leftarrow sign(random(-1, 1)) \qquad \forall i \in [1, K]$
5: **for** $t = 1 : S$ **do**
6: Shuffle: $update_sequence$
7: **for** i in $update_sequence$ **do**
8: $I[i] = \beta * (J[i] * m^T + h[i])$
9: $m[i] = \text{sgn}(\text{rand}(-1, 1) + \tanh(I[i]))$
10: **end for**
11: $VV(t) = m$
12: $x = \sum_{q=1}^{K} 2^{K-q} * m[q]$
13: $P_{model}[x] \leftarrow P_{model}[x] + 1/2^K$
14: **end for**
15: **return** VV and P_{model}

6.1.2 Deriving sparse probabilistic computing network

Recently, sparsity has been explored to demonstrate a massively parallelized probabilistic computing network in which speed-up in runtime was achieved at the expense of an increased number of p-bits in the network [106]. With a sparsely connected p-bit network, an area-efficient implementation is intended. However, reduced performance is expected from the sparsely connected p-bit network due to the removed connections. Nevertheless, it is shown that a certain level of sparsity can be tolerated in this case.



(a)

(b)

Fig. 6.1: (a) Digit image set TT, $violet$ and $yellow$ color represents pixel values -1 and 1 (b) Partial images with 6 out of 15 p-bits clamped to a unique pattern for every digit. $Green$ color represents the unclamped p-bits.

The pixel pattern in TT is shown in Fig. 6.1(a). Given a unique clamping pattern for all the digits, a full image is recovered from a partial image by the image completion network, as shown in Fig. 6.1(b). This clamped pattern is represented by the value of h_i^{fix} in Eq. 2.2. Next, an RMS measure E_{RMS} is defined to assess the best possible sparsity achieved through a certain methodology, as given by Eq. 6.2.

$$E_{RMS} = \frac{1}{no_trials} \sum_{trial=1}^{no_trials} \frac{1}{N} \sum_{x=1}^{N} \left(\sqrt{\frac{1}{K} \sum_{k=1}^{K} (Ref_{pixel}(k) - Gen_{pixel}(k))^2} \right) \quad (6.2)$$

where E_{RMS} is averaged over specified no_trials. For example, Ref_{pixel} takes the pixel values of letter 0 and Gen_{pixel} is averaged over S samples obtained from the trained probabilistic computing network provided that h_i^{fix} is clamped to the pixel values of letter 0 shown in Fig. 6.1(b).

Here, the trained network is evaluated based on the following assumption. Ideally, the states corresponding to all the ten letters in the digit image set contribute equally, i.e. 10% to the P_{model}. However, the p-bit network is probabilistic, and therefore, the P_{model} also contains the contribution of the invalid states. The p-bit network is assumed to be trained if the valid states corresponding to each letter in the digit image set contribute 5% or more to the P_{model}. Hence, the contribution of all the valid states is effectively 50%. This threshold ensures completed images remain visually distinguishable while allowing further refinement using the global strength parameter β. Additionally, it is observed that a network satisfying this criterion during training exhibited $E_{RMS} < 0.3$ during image completion, with $no_trials = 10$, and samples collected from the probabilistic computing network S set to 100000. This condition has been used as the exit criterion in the following paragraphs. However, an alternative method is to train the p-bit network for a fixed number of iterations. Alternatively, the training is stopped when the change in the interconnection weights between the two training steps is less than a pre-defined threshold value. Another alternative is to evaluate using a validation set with different mask patterns such that the E_{RMS} is below a specified threshold value.

Step-wise removal of weights

In this method, the weakest connection J_{ij} between the p-bit pair i and j is removed at every step. The probabilistic computing network is re-trained for every removed connection, and therefore, step-wise (SW) is the most computationally expensive method. The methodology to obtain the best possible sparsely connected p-bit network is presented in Algo. 5. The input to the Algo. 5 is similar to that of Algo. 3, except the

Algorithm 5 Sparsely connected p-bit network: Step-Wise (SW) method

Input: TT, N, S, K, α λ, ε, $max_iterations$, $error_limit$, P_{data}, and WS
Output: h and J

1: **for** $n = 1 : K(K-1)/2$ **do**
2: $\quad Connections_J_{ij}[n] = (i, j) \qquad \forall i, j \in [1, k]; i \neq j$
3: **end for**
4: $(h,\ J)$ = TRAIN_PC(TT, N, S, K, α λ, ε, $max_iterations$, P_{data}, WS, and $Connections_J_{ij}$)
5: **for** $no_weights = 1 : K(K-1)/2$ **do**
6: $\quad (i_{low}, j_{low})$ = index, (i, j) of $min|J[i][j]|$
7: $\quad Connections_J_{ij} \leftarrow Connection_J_{ij} - (i_{low}, j_{low})$
8: $\quad$ Repeat step (4)
9: $\quad$ Evaluate E_{RMS} acc. to Eq. 6.2
10: $\quad$ **if** $E_{RMS} > error_limit$ **then**
11: $\quad\quad$ break /* stop the weight removal process /*
12: $\quad$ **end if**
13: **end for**
14: **return** h and J

parameter $Connections_J_{ij}$, which is initialized in step (2). In Step (4), a fully connected p-bit network is trained. Once a fully connected p-bit network is obtained, the lowest unsigned magnitude in the J is selected and removed in steps (6) and (7). To compensate for the removed connection, the network is re-trained in the step (8). If the E_{RMS} calculated in step (9) is greater than the $error_limit$, the weight removal process is stopped as shown in step (11). The algorithm at completion returned the h and J matrices. The zero entries in the J signify the sparsity of the probabilistic computing network.

Weight removal with Binary Search method

The next method to obtain the best possible sparsely connected p-bit network is based on the binary search (BS) method. As the name suggests, the sparsity is obtained by

removing half of the total number of connections between the p-bits, where all the connections are sorted in the increasing weight values. Therefore, the minimum valued connections are removed first, and the E_{RMS} value is checked. If the E_{RMS} is not satisfied, only the lower half-of-half (one-fourth) of the minimum-valued connections are removed and E_{RMS} is checked. If the E_{RMS} is satisfied, the method is recursively applied to the remaining half of the weights. The methodology to realize sparse p-bit network using binary search algorithm is presented in Algo. 6. The input to the algorithm

Algorithm 6 Sparsely connected p-bit network - Binary Search (BS) method

Input: TT, N, S, K, α λ, ε, $max_iterations$, $error_limit$, no_trials, P_{data}, and WS

Output: h and J

1: **Initialize** : $upper_no_{weights} \leftarrow K(K-1)/2$, $lower_no_{weights} \leftarrow 0$, and $total_removed$ of size $[1 \times no_trials]$ with 0
2: **for** $n = 1 : K(K-1)/2$ **do**
3: $\quad Connections_J_{ij}[n] = (i, j) \qquad \forall i, j \in [1, k]; i \neq j$
4: **end for**
5: (h_R, J_R) = TRAIN_PC(TT, N, S, K, α λ, ε, $max_iterations$, P_{data}, WS, and $Connections_J_{ij}$)
6: **for** $trial = 1 : no_trials$ **do**
7: $\quad$ **while** $|upper_no_{weights} - lower_no_{weights}| > 1$ **do**
8: $\quad\quad extra_weights = ceil((upper_no_{weights} - lower_no_{weights})/2)$
9: $\quad\quad (i_{low}, j_{low})$ = indices, $extra_weights$ no. of $min|J_R[i][j]|$
10: $\quad\quad Connection_J_{ij} \leftarrow Connection_J_{ij} -$ all (i_{low}, j_{low})
11: $\quad\quad$ Repeat step (5)
12: $\quad\quad$ Evaluate E_{RMS} acc. to Eq. 6.2
13: $\quad\quad$ **if** $E_{RMS} < error_limit$ **then**
14: $\quad\quad\quad lower_no_{weights} \leftarrow extra_weights$
15: $\quad\quad$ **else**
16: $\quad\quad\quad upper_no_{weights} \leftarrow extra_weights$
17: $\quad\quad$ **end if**
18: $\quad$ **end while**
19: $\quad total_removed[trial] = extra_weights$
20: **end for**
21: **return** h, J, and $total_removed$

is similar to Algo. 5, except the no_trials parameter. Since, the probabilistic computing network is stochastic, the final sparse probabilistic computing network is reported after $no_trials = 10$. In step (1), for a fully connected p-bit network, $upper_no_{weights}$ is $K(K-1)/2$ and $lower_no_{weights}$ is 0. Therefore, in step (3), $Connections_J_{ij}$ has all the $K(K-1)/2$ p-bit pairs. In step (5), a fully connected p-bit network is trained.

For an even numbered p-bit network, the removal can be done until the $upper_no_{weights}$ becomes equal to the $lower_no_{weights}$ but, the $K(K-1)/2$ for this network is an odd number hence the condition in step 7 is adjusted. In steps (9) and (10), 50% of the lowest existing weights in J_R are removed and the indices of remaining weights are stored in $Connections_J_{ij}$. To compensate for the removed weights, probabilistic computing network is re-trained in the step (11). If the E_{RMS} calculated in step (12) is less than the $error_limit$, the removed weights are accepted and the value of $lower_no_{weights}$ is updated, otherwise, $upper_no_{weights}$ is updated as shown in steps (13) to (17). The algorithm returns h, J, and the no. of weights removed in each trial attempt. The final sparsity is reported corresponding to the most frequently occurring $extra_weights$ in $total_removed$ parameter.

Predicting weights in sparsely connected p-bit network using Information Gain

In this section, a method to generate a sparsely connected p-bit network using Information Gain (IG) is proposed. Link prediction in probabilistic graphical models has been widely explored [146], [147]. The prediction of network structure for a simplified version of BM, the Restricted Boltzmann Machine (RBM), has been discussed in [148]. In the proposed method, the existence of the weight J_{ij} between the p-bits i and j is predicted prior to training the p-bit network. In contrast, simpler methods like removing weights from a fully connected network require multiple training and evaluation to remove the connections. Therefore, the proposed method based on IG is computationally efficient as it is based on a bottom-up strategy.

A fully connected p-bit network is described as a regular undirected graphical model in which every p-bit is connected to all other p-bits in the network. To obtain a sparsely connected p-bit network, the Mutual Information (MI) between two p-bit pairs (Y, W), given their common neighbors $Z_1, Z_2, \ldots, Z_n$ p-bits, is evaluated according to Eq. 6.3.

$$\begin{aligned} MI(Y,W|Z_1,Z_2,...,Z_n) = \quad & H(Y|Z_1,Z_2,...,Z_n) + H(W|Z_1,Z_2,...,Z_n) \\ & -H(Y,W|Z_1,Z_2,...,Z_n) \\ & \forall Y,W,Z_n \in [1,K]; Y \neq W \neq Z_n \end{aligned} \tag{6.3}$$

where Y denotes a 1D-array with all the training instances for the Yth p-bit i.e., $Y[0], Y[1]$,

...., $Y[10]$ for ten letters in the digit image set, and $H(Y|Z_1, Z_2, ..., Z_n)$ is the conditional entropy evaluated according to the following equation [136]:

$$H(Y|Z_1, Z_2, ..., Z_n) = H(Y, Z_1, Z_2, ..., Z_n) - H(Z_1, Z_2, ..., Z_n) \quad (6.4)$$

Substituting Eq. 6.4 into Eq. 6.3 and simplifying, the following equation is obtained [136]:

$$\begin{aligned} MI(Y, W|Z_1, Z_2, ..., Z_n) = \quad & H(Y, Z_1, Z_2, ..., Z_n) + H(W, Z_1, Z_2, ..., Z_n) \\ & -H(Y, W, Z_1, Z_2, ..., Z_n) - H(Z_1, Z_2, ..., Z_n) \\ \forall Y, W, Z_n \in [1, K];\ & Y \neq W \neq Z_1 \neq Z_2 ... \neq Z_n \end{aligned} \quad (6.5)$$

Further, using the definition of joint entropy, $H(Y, W, Z_1, Z_2, \ldots, Z_n)$ is calculated as follows [136]:

$$\begin{aligned} H(Y, Z_1, Z_2, ..., Z_n) = -\sum P(Y, Z_1, Z_2, ..., Z_n) \cdot \log_2 \left(P(Y, Z_1, Z_2, ..., Z_n)\right) \\ \forall Y, Z_1, Z_2, ..., Z_n \in [1, K];\ Y \neq Z_1 \neq Z_2 ... \neq Z_n \end{aligned} \quad (6.6)$$

where $P(Y, Z_1, Z_2, ..., Z_n)$ is the joint probability distribution evaluated for all the instances in the training dataset [136].

The input to the Algo. 7 is a Graph G with vertices V and edges E. The rest of the inputs are similar to that of the Algo. 6. In step (1), V has all the p-bits, and E is empty. The $strength_MI$ stores the MI between all the p-bits in V. An adjacency matrix Adj_Matrix indicates the existence of a connection between p-bits i and j. In step (2), MI is calculated according to the Eq. 6.5. All the p-bits are disconnected at the start of the algorithm, hence the contribution of common neighbors in MI is zero. Next, the p-bit pairs with maximum MI are recognized as connected, and the Adj_Matrix is updated in step (4). Now, the p-bits will form new connections bounded by the degree $deg(V)$ of any p-bit in V. In step (9), the p-bits with non zero degrees are picked, and the MI to all the other p-bits is evaluated in steps (10) to (12). Again, in step (14), new connections are updated in Adj_Matrix based on the maximum MI, calculated in step (12). The process of edge creation is repeated until $deg(pbit)$ is achieved for every p-bit. In step (19) $Connections_J_{ij}$ are updated and is given to the TRAIN_PC algorithm in

Algorithm 7 Sparsely connected p-bit network: Information gain (IG) method

Input: $G = (V, E)$, TT, N, S, K, α λ, ε, $max_iterations$, $error_limit$, no_trials, P_{data}, and WS

Output: h and J

1: Initialize: $V \in [1, K]$, $E \leftarrow \phi$, $Connections_J_{ij}$ of size $[1 \times K(K-1)/2]$, $strength_MI$ and Adj_Matrix of size $[K \times K]$ with all 0
2: $strength_MI(Y, W) \leftarrow MI(Y, W)\, \forall Y, W \in [1, K]; Y \neq W$
3: **for** max valued p-bit pairs, (Y_{max}, W_{max}) in $strength_MI(Y, W)$ **do**
4: $\quad Adj_Matrix[Y_{max}, W_{max}] \leftarrow 1$
5: **end for**
6: **for** $degree_pbit = 1 : K - 1$ **do**
7: $\quad$ **while** any $deg(pbit) < degree_pbit$ **do**
8: $\quad\quad$ Initialize: $strength_MI$ with all 0
9: $\quad\quad$ **for** $A =$ nonzero $deg(pbit)$ **do**
10: $\quad\quad\quad$ **for** $B = 1 : K; A \neq B$ **do**
11: $\quad\quad\quad\quad obs_node \leftarrow (neighbor_A \cap neighbor_B)$
12: $\quad\quad\quad\quad strength_MI(A, B) = (MI(A, B|obs_node)$
13: $\quad\quad\quad\quad$ **for** max valued p-bit pairs, (A_{max}, B_{max}) in $strength_MI(A, B)$ **do**
14: $\quad\quad\quad\quad\quad Adj_Matrix[A_{max}, B_{max}] \leftarrow 1$
15: $\quad\quad\quad\quad$ **end for**
16: $\quad\quad\quad$ **end for**
17: $\quad\quad$ **end for**
18: $\quad$ **end while**
19: $\quad Connections_J_{ij} \leftarrow Connections_J_{ij} + (i, j)$ **if** $Adj_Matrix[i][j] \neq 0$
20: $\quad (h,\ J)$ = TRAIN_PC(TT, N, S, K, α λ, ε, $max_iterations$, P_{data}, WS, and $Connections_J_{ij}$)
21: $\quad$ Calculate E_{RMS} acc. to Eq. 6.2
22: $\quad$ **if** $E_{RMS} < error_limit$ **then**
23: $\quad\quad$ break */ optimal $Connections_J_{ij}$ achieved /*
24: $\quad$ **end if**
25: **end for**
26: **return** h and J

step (20). In step (22), if the $error_limit$ criteria is not satisfied, the algorithm repeats to increase the degree of the p-bits.

6.1.3 Results

A quantitative assessment of the image completion using the probabilistic computing network is performed in this section. It should be noted that the number of pairwise connections J_{ij} in a fully connected p-bit network is given by $K(K-1)/2$. Therefore, the J_{ij} terms increase quadratically with K.

Fig. 6.2 shows the best possible sparsity from the three methods. The broken line

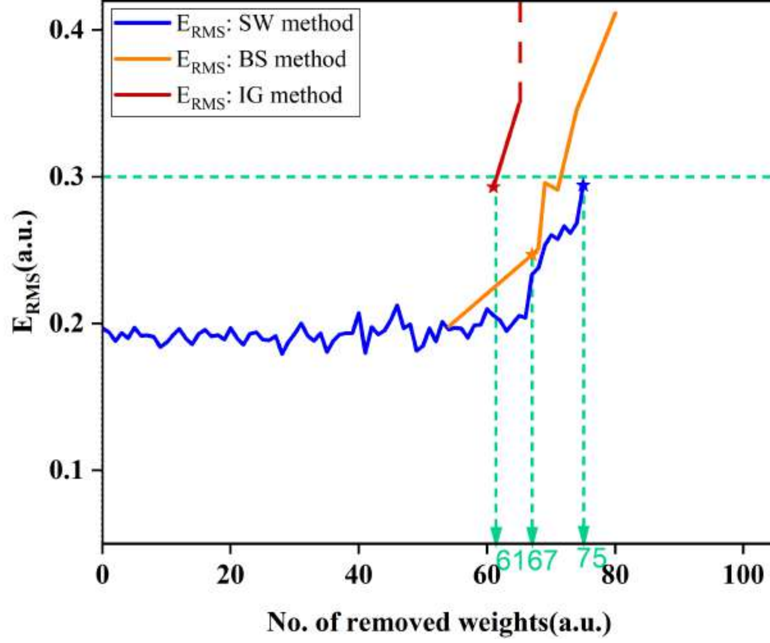


Fig. 6.2: No. of removed $Connections_J_{ij}$ bounded by the E_{RMS} value. Maximum number of removed weights for every method is shown in $green$ color.

extending the IG method signifies that the TRAIN_PC algorithm could not train the sparsely connected p-bit network because the number of connections are insufficient to satisfy the KL divergence criteria.

The SW method can remove the maximum possible connections, i.e., 75 out of 105, achieving the sparsity of 71.43%. Similarly, BS and IG methods achieved sparsity of 63.81% and 58.09%. The order of complexity for SW method is $O(C)$, where C is the number of connections in a fully connected p-bit network. The number of connections is quadratically related to the number of p-bits in the network. The complexity of BS method is $O(\log(C))$ because the number of connections that can be removed is halved at every step. For the IG method, the maximum number of attempts is equal to the maximum degree of a p-bit, which is equal to $K - 1$. Additionally, the mutual information is calculated all the p-bit pairs in the network. Therefore, its order is $O(K)$ Tab. 6.2

Table 6.2: Comparison of the order of complexity of sparsification methods

Method	Order of complexity	# of attempts to achieve sparse p-bit network
Step-Wise	$O(C)$	75
Binary Search	$O(\log(C))$	$7(\times no_trials)$
Information gain	$O(K)$	4

shows the order of complexity and the number of attempts to sparsify a fully connected p-bit network. Although the sparsity of IG is slightly lower than that of BS, it requires significantly lower computational effort. It should be noted that the MI is calculated for all the connections in the p-bit network. For a small problem, similar to one demonstrated in this work, the iterative nature of the Step-Wise (SW) and Binary Search (BS) methods seems to be computationally expensive in comparison to the bottom-up strategy of the IG method. However, in a large p-bit network, calculating MI for all p-bit pairs may pose a significant computational bottleneck.

Moreover, increasing the value of β, where β represents the strength of interconnections between the p-bits (given by Eq. 2.2), increases the strength of input stimuli to the p-bits. The increase in the strength of input stimuli results in a reduction of the fluctuations of the p-bits in the p-bit network. Additionally, the performance of every method is evaluated for the high precision and low precision of h and J weight values. It is required that the performance should not deteriorate at the low precision weight values because the low precision weight values represent the mapping of weight value to the resistance values in hardware implementation as discussed in [97].

Fig. 6.3 shows the image completion performed using the fully connected and the sparsely connected p-bit network. The pixel value is calculated as the average of the output of the p-bit over S sampling instances. Therefore, the pixel value varies in the range -1 and 1, and is encoded by the colormap, as shown in Fig. 6.3. The pixel value around -1 (violet colour) and 1 (yellow colour) signifies that the output of the p-bit is almost pinned to state -1 or 1, respectively. However, a pixel value around 0 (green colour) signifies unpinned output of the p-bit. For a fully connected p-bit network, Fig. 6.3(a,b) shows that the pixel values for high precision weight values and optimized β are closer to -1 (violet colour) and 1 (yellow colour), as compared to the low precision weight values. Fig. 6.3(c,d) shows the image completion for a sparsely connected p-bit network derived using the SW method. However, for both high precision and low precision weight values, digit images for 'four', 'five', and 'six' have a few pixels with around 0 values, and the digits are not clearly distinguishable from one another. Fig. 6.3(e,f) and (g,h) illustrate the image completion for BS and IG methods, respectively. For both methods, it is evident that the performance of the p-bit network with high

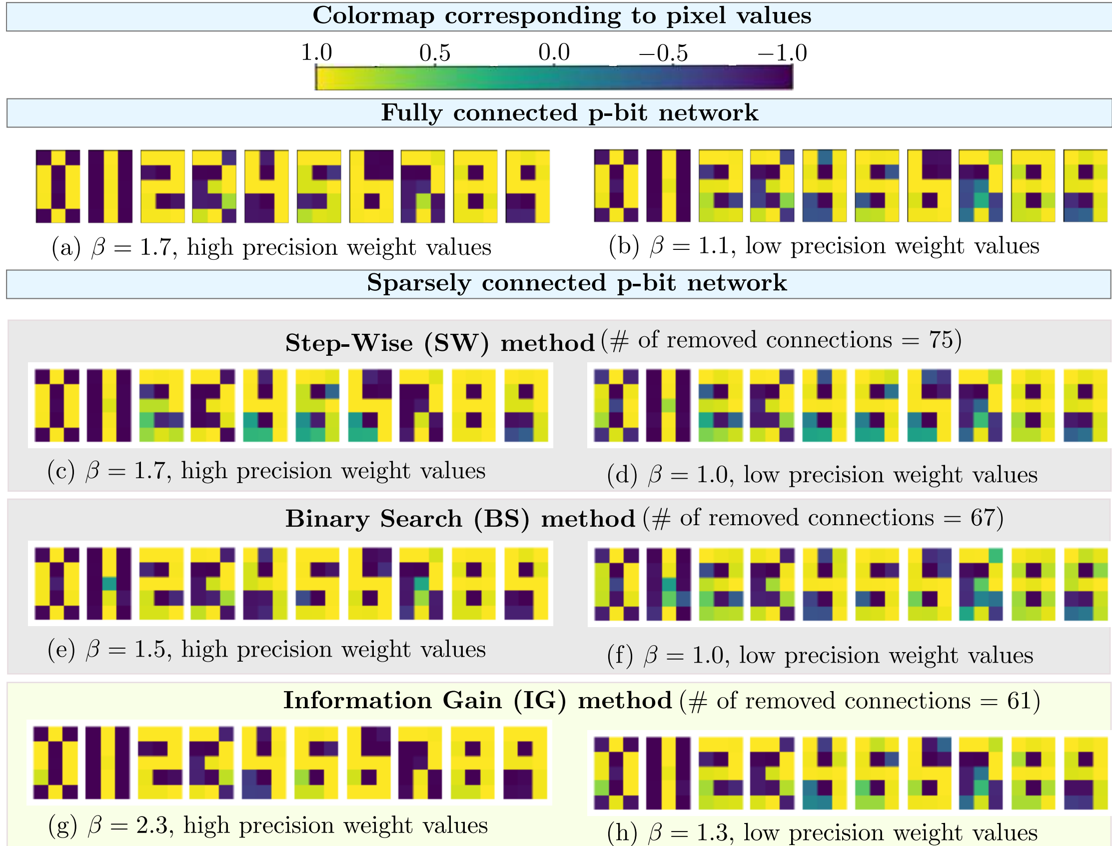


Fig. 6.3: Image completion: Colormap of the p-bit outputs: (a,b) Fully connected p-bit network. Sparsely connected p-bit network realized using (c,d) step-wise (SW) method, (e,f) binary search (BS) method, and (g,h) information gain (IG) method.

precision weight values is better than the low precision weight values. Additionally, the accuracy for all three methods is similar. However, the proposed method based on IG is computationally more efficient, as demonstrated earlier.

6.2 Conclusion

A methodology to sparsify a fully connected probabilistic network is proposed to reduce the area of the p-bit network. However, an accurate estimate can be provided only after the post-layout simulations, and is intended for future work. The sparsely connected p-bit network derived from the proposed algorithm using the IG approach successfully performed the image completion task, with a sparsity of 58.09%. The IG concept can also identify the groups of closely related p-bits and parallelize the systems to speed up the computational run time of a p-bit system. However, this requires quantitative validation and is a key direction for future research. Furthermore, IG based sparsifica-

tion method generalizes to p-bit network performing tasks beyond image completion, provided the training data or prior knowledge of the probability distribution of the valid and invalid states is available. It is important to note that the power dissipation is also expected to reduce with the reduction in the area. However, an exhaustive comparison with the existing pruning methods would be a valuable addition, and is intended for future work.

Chapter 7

Conclusion and Future Work

7.1 Summary

Probabilistic computing is an emerging computational paradigm that leverages the intrinsic stochastic behavior of probabilistic bits (p-bits) to perform efficient and invertible computation. This work explores multiple aspects of the probabilistic computing systems, including architectural, material, and robustness considerations.

From a hardware perspective, simulations are conducted to evaluate the feasibility of implementing p-bits using 1T-1MTJ structures with LBM. The influence of material parameters on p-bit performance particularly flips per second (fps) is analyzed, and design guidelines are proposed to optimize system-level performance without compromising sigmoid-like behavior.

To investigate the impact of process-induced variations, a mathematical model for an ideal p-bit is proposed and extended to incorporate non-idealities, such as bias mismatches and asymmetries in the transfer characteristics. The model is validated against SPICE simulations and numerical computations, demonstrating close agreement under both ideal and non-ideal conditions.

The robustness of p-bit systems is also investigated, especially in the presence of faults such as stuck-at faults resulting from fabrication defects, aging, or operational variations. Using the MNIST dataset as a benchmark, the impact of these faults on the system accuracy is assessed. A mutual information-based criticality score (CS) is introduced to identify highly impactful p-bits within the network, enabling selective application of fault-tolerance strategies. Testable and isolatable p-bit designs are proposed and validated through SPICE simulations using $14nm$ FinFET technology, offering a practical framework for building resilient p-bit systems with minimal area overhead.

Additionally, the invertibility property of fully connected p-network is exploited for partial image completion using digit image set. An area-efficient sparsely connected p-

bit network is derived using the proposed information gain method resulting in retaining only $\approx 42\%$ connections.

In summary, this work enables employing probabilistic computing for real-world applications by tackling several practical challenges. A comprehensive performance benchmarking of the proposed frameworks against CMOS alternatives is a key direction for future research.

7.2 Future Work

While this thesis addresses several aspects of the design and modelling of p-bits and p-bit systems, critical challenges persist in achieving scalable hardware architectures, managing interconnect complexity, ensuring efficient mapping of the algorithm on the hardware, harnessing the inherent stochasticity of the device, and tackling reliability issues. Therefore, several important research directions remain open for future investigation:

- **Fabrication-driven Validation:** Although proposed devices and circuits are studied using simulation, benchmarking using fabrication-driven experiments will give more insights into practical issues and solutions.

- **Optimizing Device-Circuit Co-Design**: The area and power overhead of CMOS-based p-bits limit the network scalability to beyond one million p-bits. In terms of reducing area- and power-overhead, nanomagnet-based p-bit implementation seems promising. To this end, the future work should address key challenges such as device-to-device variability, the interplay between continuously fluctuating stochastic devices with CMOS circuits to realize scalable p-bit systems.

- **Realizing scalable p-bit systems**: Despite advancement at the device-level, the experimental results show that the p-bit can only attain up to 10^5 flips per second [149]. This happens because of the frequent charging/discharging of the capacitances present in the circuit elements. Additionally, a larger network necessitates realizing a fast synapse such that the fast fluctuating p-bits can communicate and deliver the correct functionality. Addressing these challenges can contribute to building a scalable p-bit network.

- **Aging and Reliability Modeling:** Aging-induced non-idealities in NMOS and MTJ components can degrade performance over time [61]. Incorporating such long-term effects into the mathematical model would offer a more realistic estimation of lifetime reliability.
- **Redundancy and Reconfigurability:** Future designs can explore more sophisticated redundancy strategies, such as dynamic fault-tolerant topologies. Reconfigurable p-bit arrays could allow dynamic isolation and replacement of faulty units in runtime.
- **CAD Toolchain for p-bit Synthesis:** A hardware description language (HDL)-to-p-bit compiler and design automation framework (as discussed in [92]) would help bring probabilistic logic design into mainstream VLSI workflows.
- **ML and Inference Accelerators:** Extending the p-bit network to accelerate probabilistic inference and machine learning workloads including Boltzmann machines [99, 25] could unlock new directions in low-power neuromorphic computing.
- **Probabilistic Security Primitives:** The inherent stochastic behavior of p-bits can be exploited to create entropy sources for true random number generators and physical unclonable functions (PUFs).
- **Cross-Paradigm Comparisons:** A comparative study between p-bits, stochastic CMOS [59], c-bits [108], and quantum-inspired computing platforms could provide insights into the niche applications best served by each paradigm.

In conclusion, this thesis lays a strong foundation for the scalable and robust deployment of p-bit-based probabilistic computing systems. With further development in hardware, design automation, and emerging applications, p-bits have the potential to serve as a practical bridge between classical deterministic logic and future quantum-inspired computing platforms.

References

[1] K. Y. Camsari, R. Faria, B. M. Sutton, and S. Datta, "Stochastic *p*-Bits for Invertible Logic," *Physical Review X*, vol. 7, no. 3, Jul. 2017, Art. no. 031014.

[2] K. Y. Camsari, S. Salahuddin, and S. Datta, "Implementing p-bits With Embedded MTJ," *IEEE Electron Device Letters*, vol. 38, no. 12, pp. 1767–1770, 2017.

[3] W. Zhao *et al.*, "Failure analysis in magnetic tunnel junction nanopillar with interfacial perpendicular magnetic anisotropy," *Materials*, vol. 9, no. 1, p. 41, 2016.

[4] S. Chowdhury *et al.*, "A Full-Stack View of Probabilistic Computing With p-Bits: Devices, Architectures, and Algorithms," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 9, no. 1, pp. 1–11, 2023.

[5] R. Faria, K. Y. Camsari, and S. Datta, "Low-Barrier Nanomagnets as p-Bits for Spin Logic," *IEEE Magnetics Letters*, vol. 8, pp. 1–5, 2017.

[6] C.-H. Jan *et al.*, "A 45nm low power system-on-chip technology with dual gate (logic and I/O) high-k/metal gate strained silicon transistors," in *2008 IEEE International Electron Devices Meeting*. IEEE, 2008, pp. 1–4.

[7] E. M. Bazizi *et al.*, "GAA Technology Innovations for 2nm Logic node and Beyond," in *2024 8th IEEE Electron Devices Technology & Manufacturing Conference (EDTM)*. IEEE, 2024, pp. 1–3.

[8] M. S. Lundstrom and M. A. Alam, "Moore's law: the journey ahead," *Science*, vol. 378, no. 6621, pp. 722–723, 2022.

[9] IEEE International Roadmap for Devices and Systems, "Beyond CMOS and Emerging Materials Integration," *Institute of Electrical and Electronics Engineers*, 2023, doi: 10.60627/0P45-ZJ55.

[10] J. Von Neumann, “Probabilistic logics and the synthesis of reliable organisms from unreliable components,” *Automata studies*, vol. 34, no. 34, pp. 43–98, 1956.

[11] W. Poppelbaum, C. Afuso, and J. Esch, “Stochastic computing elements and systems,” in *Proceedings of the November 14-16, 1967, fall joint computer conference*, 1967, pp. 635–644.

[12] B. R. Gaines, “Stochastic computing systems,” *Advances in Information Systems Science: Volume 2*, pp. 37–172, 1969.

[13] R. P. Feynman, “Simulating Physics with Computers,” *International Journal of Theoretical Physics*, vol. 21, no. 6, pp. 467–488, 1982.

[14] W. A. Borders, A. Z. Pervaiz, S. Fukami, K. Y. Camsari, H. Ohno, and S. Datta, “Integer factorization using stochastic magnetic tunnel junctions,” *Nature*, vol. 573, no. 7774, pp. 390–393, 2019.

[15] P. Debashis, R. Faria, K. Y. Camsari, and Z. Chen, “Design of stochastic nanomagnets for probabilistic spin logic,” *IEEE Magnetics Letters*, vol. 9, pp. 1–5, 2018.

[16] B. R. Gaines, “Stochastic computer thrives on noise,” *Electronics*, vol. 40, no. 14, p. 72, 1967.

[17] B. R. Gaines, “Stochastic and fuzzy logics,” *Electronics Letters*, vol. 11, no. 9, pp. 188–189, 1975.

[18] B. R. Gaines, “Techniques of identification with the stochastic computer,” in *Proceedings International Federation of Automatic Control Symposium on Identification, Progue*, 1967.

[19] M. Liu and H.-S. P. Wong, “The path to a 1-trillion-transistor gpu: Ai’s boom demands new chip technology,” *IEEE Spectrum*, vol. 61, no. 7, pp. 22–27, 2024.

[20] M. Khan and O. Hassan, “Benchmarking of Probabilistic-bit based Algorithm for Max-cut Problem,” in *2022 12th International Conference on Electrical and Computer Engineering (ICECE)*, 2022, pp. 453–456.

[21] S. Krinner *et al.*, "Engineering cryogenic setups for 100-qubit scale superconducting circuit systems," *EPJ Quantum Technology*, vol. 6, no. 1, p. 2, 2019.

[22] S. Khasanvis *et al.*, "Physically equivalent magneto-electric nanoarchitecture for probabilistic reasoning," in *Proceedings of the 2015 IEEE/ACM International Symposium on Nanoscale Architectures (NANOARCH ´15)*, 2015, pp. 25–26.

[23] S. Khasanvis *et al.*, "Self-similar magneto-electric nanocircuit technology for probabilistic inference engines," *IEEE Transactions on Nanotechnology*, vol. 14, no. 6, pp. 980–991, 2015.

[24] B. Behin-Aein, V. Diep, and S. Datta, "A building block for hardware belief networks," *Scientific reports*, vol. 6, no. 1, p. 29893, 2016.

[25] S. Niazi, S. Chowdhury, N. A. Aadit, M. Mohseni, Y. Qin, and K. Y. Camsari, "Training deep Boltzmann networks with sparse Ising machines," *Nature Electronics*, pp. 1–10, 2024.

[26] H.-Y. Maeng *et al.*, "Homeothermic P-Bit Computing Hardware with Stochastic Operations Beyond Limit of Non-Stochastic Materials," *Advanced Functional Materials*, p. 2417552, 2024.

[27] A. Z. Pervaiz, B. M. Sutton, L. A. Ghantasala, and K. Y. Camsari, "Weighted *p*-Bits for FPGA Implementation of Probabilistic Circuits," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 6, pp. 1920–1926, 2018.

[28] A. Z. Pervaiz, L. A. Ghantasala, K. Y. Camsari, and S. Datta, "Hardware emulation of stochastic p-bits for invertible logic," *Scientific Reports*, vol. 7, no. 1, Dec. 2017.

[29] B. Sutton, R. Faria, L. A. Ghantasala, R. Jaiswal, K. Y. Camsari, and S. Datta, "Autonomous Probabilistic Coprocessing With Petaflips per Second," *IEEE Access*, vol. 8, pp. 157 238–157 252, 2020.

[30] P. Debashis, R. Faria, K. Y. Camsari, J. Appenzeller, S. Datta, and Z. Chen, "Experimental demonstration of nanomagnet networks as hardware for Ising com-

puting," in *2016 IEEE International Electron Devices Meeting (IEDM)*. IEEE, 2016, pp. 34–3.

[31] H.-S. Jun *et al.*, "Hardware Implementation of a Fully Functional Stochastic p-STT Neuron for Probabilistic Computing," *Advanced Electronic Materials*, p. 2300821, 2024.

[32] O. Hassan, R. Faria, K. Y. Camsari, J. Z. Sun, and S. Datta, "Low-barrier magnet design for efficient hardware binary stochastic neurons," *IEEE Magnetics Letters*, vol. 10, pp. 1–5, 2019.

[33] V. Ostwal and J. Appenzeller, "Spin–orbit torque-controlled magnetic tunnel junction with low thermal stability for tunable random number generation," *IEEE Magnetics Letters*, vol. 10, pp. 1–5, 2019.

[34] X. Li *et al.*, "True random number generator based on spin–orbit torque magnetic tunnel junctions," *Applied Physics Letters*, vol. 123, no. 14, 2023.

[35] A. Thapa and B. Sharma, "A Review on Novel Low-Dimensional Materials based Magnetic Tunnel Junctions: Opportunities, Challenges, and Applications," *Advanced Materials Technologies*, p. e00133, 2025.

[36] V. Nguyen, S. Rao, K. Wostyn, and S. Couet, "Recent progress in spin-orbit torque magnetic random-access memory," *npj Spintronics*, vol. 2, no. 1, p. 48, 2024.

[37] O. Hassan, S. Datta, and K. Y. Camsari, "Quantitative Evaluation of Hardware Binary Stochastic Neurons," *Physical Review Applied*, vol. 15, Jun 2021, Art. no. 064046.

[38] D. E. Nikonov, "Stochastic magnetic circuits rival quantum computing," 2019.

[39] O. Hassan, K. Y. Camsari, and S. Datta, "Voltage-Driven Building Block for Hardware Belief Networks," *IEEE Design & Test*, vol. 36, no. 3, pp. 15–21, 2019.

[40] Predictive Technology Model (PTM). Accessed: Jan 8, 2025. [Online]. Available: https://mec.umn.edu/ptm

[41] S. Luo, Y. He, B. Cai, X. Gong, and G. Liang, "Ferroelectric Probabilistic Bits based on Thermal Noise induced Randomness for Stochastic Computing," in *2023 7th IEEE Electron Devices Technology & Manufacturing Conference (EDTM).* IEEE, 2023, pp. 1–3.

[42] Y. Liu *et al.*, "Probabilistic circuit implementation based on p-bits using the intrinsic random property of RRAM and p-bit multiplexing strategy," *Micromachines*, vol. 13, no. 6, p. 924, 2022.

[43] D. Khilwani *et al.*, "$Pr_xCa_{1-x}MnO_3$ based stochastic neuron for Boltzmann machine to solve "maximum cut" problem," *APL Materials*, vol. 7, no. 9, 2019.

[44] K. S. Woo, J. Kim, J. Han, W. Kim, Y. H. Jang, and C. S. Hwang, "Probabilistic computing using $Cu_{0.1}Te_{0.9}/HfO_2/Pt$ diffusive memristors," *Nature Communications*, vol. 13, no. 1, p. 5762, 2022.

[45] T. J. Park *et al.*, "Efficient Probabilistic Computing with Stochastic Perovskite Nickelates," *Nano Letters*, vol. 22, no. 21, pp. 8654–8661, 2022.

[46] W. Whitehead, Z. Nelson, K. Y. Camsari, and L. Theogarajan, "CMOS-compatible Ising and Potts annealing using single-photon avalanche diodes," *Nature Electronics*, vol. 6, no. 12, pp. 1009–1019, 2023.

[47] W. Whitehead, W. Oh, and L. Theogarajan, "CMOS Single-Photon Avalanche Diode Circuits for Probabilistic Computing," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 10, pp. 49–57, 2024.

[48] K. Y. Camsari, R. Faria, O. Hassan, B. M. Sutton, and S. Datta, "Equivalent Circuit for Magnetoelectric Read and Write Operations," *Physical Review Applied*, vol. 9, no. 4, p. 044020, 2018.

[49] R. Yamanaka, N. Yoshikawa, and Y. Yamanashi, "Implementation of Bayesian Network Based on Ultra-High-Speed Superconductor Random Number Generators," *IEEE Access*, 2024.

[50] Y. Lv, R. P. Bloom, and J.-P. Wang, "Experimental Demonstration of Probabilistic Spin Logic by Magnetic Tunnel Junctions," *IEEE Magnetics Letters*, vol. 10, pp. 1–5, 2019.

[51] K. Hayakawa *et al.*, "Nanosecond random telegraph noise in in-plane magnetic tunnel junctions," *Physical review letters*, vol. 126, no. 11, p. 117202, 2021.

[52] C. Safranski, J. Kaiser, P. Trouilloud, P. Hashemi, G. Hu, and J. Z. Sun, "Demonstration of nanosecond operation in stochastic magnetic tunnel junctions," *Nano letters*, vol. 21, no. 5, pp. 2040–2045, 2021.

[53] G. M. Gutiérrez-Finol, S. Giménez-Santamarina, Z. Hu, L. E. Rosaleny, S. Cardona-Serra, and A. Gaita-Ariño, "Lanthanide molecular nanomagnets as probabilistic bits," *npj Computational Materials*, vol. 9, no. 1, p. 196, 2023.

[54] K. J. Kuhn *et al.*, "Process technology variation," *IEEE Transactions on Electron Devices*, vol. 58, no. 8, pp. 2197–2208, 2011.

[55] J. Kaiser, W. A. Borders, K. Y. Camsari, S. Fukami, H. Ohno, and S. Datta, "Hardware-aware in situ learning based on stochastic magnetic tunnel junctions," *Physical Review Applied*, vol. 17, no. 1, p. 014016, 2022.

[56] R. Rahman and S. Bandyopadhyay, "The strong sensitivity of the characteristics of binary stochastic neurons employing low barrier nanomagnets to small geometrical variations," *IEEE Transactions on Nanotechnology*, vol. 22, pp. 112–119, 2023.

[57] M. G. Morshed, S. Ganguly, and A. W. Ghosh, "A Deep Dive Into the Computational Fidelity of High-Variability Low Energy Barrier Magnet Technology for Accelerating Optimization and Bayesian Problems," *IEEE Magnetics Letters*, vol. 14, pp. 1–5, 2023.

[58] A. H. Lone, S. Shringi, K. Mishra, and S. Srinivasan, "Cross-sectional area dependence of tunnel magnetoresistance, thermal stability, and critical current density in MTJ," *IEEE Transactions on Magnetics*, vol. 57, no. 2, pp. 1–10, 2020.

[59] J. L. Drobitch and S. Bandyopadhyay, "Reliability and scalability of p-bits implemented with low energy barrier nanomagnets," *IEEE Magnetics Letters*, vol. 10, pp. 1–4, 2019.

[60] N. S. Singh *et al.*, "CMOS plus stochastic nanomagnets enabling heterogeneous computers for probabilistic inference and learning," *Nature Communications*, vol. 15, no. 1, p. 2685, 2024.

[61] J. Daniel *et al.*, "Experimental demonstration of an on-chip p-bit core based on stochastic magnetic tunnel junctions and 2D MoS2 transistors," *Nature Communications*, vol. 15, no. 1, p. 4098, 2024.

[62] A. Chintaluri, A. Parihar, S. Natarajan, H. Naeimi, and A. Raychowdhury, "A model study of defects and faults in embedded spin transfer torque (STT) MRAM arrays," in *2015 IEEE 24th Asian Test Symposium (ATS)*. IEEE, 2015, pp. 187–192.

[63] A. Chintaluri, H. Naeimi, S. Natarajan, and A. Raychowdhury, "Analysis of defects and variations in embedded spin transfer torque (STT) MRAM arrays," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 6, no. 3, pp. 319–329, 2016.

[64] L. Wu, S. Rao, M. Taouil, E. J. Marinissen, G. S. Kar, and S. Hamdioui, "Testing STT-MRAM: Manufacturing defects, fault models, and test solutions," in *2021 IEEE International Test Conference (ITC)*. IEEE, 2021, pp. 143–152.

[65] A. Shukla *et al.*, "A true random number generator for probabilistic computing using stochastic magnetic actuated random transducer devices," in *2023 24th International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2023, pp. 1–10.

[66] J. Gomez *et al.*, "Experimental validation of a compact pinhole latent defect model for MOS transistors," *IEEE Transactions on Electron Devices*, vol. 69, no. 9, pp. 4796–4802, 2022.

[67] B. Kruseman, A. Majhi, C. Hora, S. Eichenberger, and J. Meirlevede, "Systematic defects in deep sub-micron technologies," in *2004 International Conferce on Test*. IEEE, 2004, pp. 290–299.

[68] M. Fieback *et al.*, "Device-aware test: A new test approach towards DPPB level," in *2019 IEEE International Test Conference (ITC)*. IEEE, 2019, pp. 1–10.

[69] A. Aouichi *et al.*, "Device Aware Diagnosis for Unique Defects in STT-MRAMs," in *2023 IEEE 32nd Asian Test Symposium (ATS)*. IEEE, 2023, pp. 1–6.

[70] L. Wu *et al.*, "Defect and fault modeling framework for STT-MRAM testing," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 2, pp. 707–723, 2019.

[71] Y. Wang *et al.*, "Compact model of dielectric breakdown in spin-transfer torque magnetic tunnel junction," *IEEE Transactions on Electron Devices*, vol. 63, no. 4, pp. 1762–1767, 2016.

[72] L. Wu *et al.*, "Pinhole defect characterization and fault modeling for STT-MRAM testing," in *2019 IEEE European Test Symposium (ETS)*. IEEE, 2019, pp. 1–6.

[73] K. Sugiura *et al.*, "Ion beam etching technology for high-density spin transfer torque magnetic random access memory," *Japanese Journal of Applied Physics*, vol. 48, no. 8S1, p. 08HD02, 2009.

[74] G. Panagopoulos, C. Augustine, and K. Roy, "Modeling of dielectric breakdown-induced time-dependent STT-MRAM performance degradation," in *69th Device Research Conference*. IEEE, 2011, pp. 125–126.

[75] J. Wang *et al.*, "Atomistic Investigation of Interface Edge Defect in CoFeB/MgO Ferromagnetic Nano-Dots," *IEEE Transactions on Magnetics*, vol. 59, no. 2, pp. 1–4, 2023.

[76] C. Engel, S. Goolaup, H. K. Teoh, and W. S. Lew, "Effect of Geometrical Modulation on pMTJ Magnetization Reversal," *IEEE Transactions on Magnetics*, vol. 53, no. 12, pp. 1–7, 2017.

[77] M. Kuepferling *et al.*, "Vortex dynamics in Co-Fe-B magnetic tunnel junctions in presence of defects," *Journal of Applied Physics*, vol. 117, no. 17, 2015.

[78] R. Bishnoi, F. Oboril, and M. B. Tahoori, "Design of Defect and Fault-Tolerant Nonvolatile Spintronic Flip-Flops," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 4, pp. 1421–1432, 2017.

[79] M. A. Abeed and S. Bandyopadhyay, "Sensitivity of the power spectra of thermal magnetization fluctuations in low barrier nanomagnets proposed for stochastic computing to in-plane barrier height variations and structural defects," in *Spin*, vol. 10, no. 01. World Scientific, 2020, p. 2050001.

[80] A. Haroon, R. K. Ghosh, and S. Saurabh, "Implementation of Probabilistic Bits (Pbits) using Low Barrier Magnets: Investigation and Analysis," in *2023 36th International Conference on VLSI Design and 2023 22nd International Conference on Embedded Systems (VLSID)*, 2023, pp. 307–312.

[81] H. Pourmeidani, P. Debashis, Z. Chen, R. F. DeMara, and R. Zand, "Electrically-tunable stochasticity for spin-based neuromorphic circuits: self-adjusting to variation," in *2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, 2020, pp. 81–84.

[82] A. Haroon, R. K. Ghosh, and S. Saurabh, "Impact of Non-Idealities on the Behavior of Probabilistic Computing: Theoretical Investigation and Analysis," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 71, no. 12, pp. 6279–6291, 2024.

[83] R. Rahman, S. Ganguly, and S. Bandyopadhyay, "Reconfigurable stochastic neurons based on strain engineered low barrier nanomagnets," *Nanotechnology*, vol. 35, no. 32, p. 325205, 2024.

[84] R. Rahman and S. Bandyopadhyay, "Ternary stochastic neuron-implemented with a single strained magnetostrictive nanomagnet," *Nanotechnology*, 2025.

[85] J. Biamonte, "Nonperturbative k-body to two-body commuting conversion Hamiltonians and embedding problem instances into Ising spins," *Physical Review A—Atomic, Molecular, and Optical Physics*, vol. 77, no. 5, p. 052331, 2008.

[86] J. D. Whitfield, M. Faccin, and J. Biamonte, "Ground-state spin logic," *Europhysics Letters*, vol. 99, no. 5, p. 57004, 2012.

[87] L. Personnaz, I. Guyon, and G. Dreyfus, "Collective computational properties of neural networks: New learning mechanisms," *Physical Review A*, vol. 34, no. 5, p. 4217, 1986.

[88] S. C. Smithson, N. Onizawa, B. H. Meyer, W. J. Gross, and T. Hanyu, "Efficient CMOS Invertible Logic Using Stochastic Computing," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 6, pp. 2263–2274, 2019.

[89] K. Nishino *et al.*, "Study of stochastic invertible multiplier designs," in *2018 25th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*. IEEE, 2018, pp. 649–650.

[90] N. Onizawa *et al.*, "A design framework for invertible logic," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 4, pp. 655–665, 2020.

[91] N. Onizawa and T. Hanyu, "CMOS invertible logic: Bidirectional operation based on the probabilistic device model and stochastic computing," *IEEE Nanotechnology Magazine*, vol. 16, no. 1, pp. 33–46, 2021.

[92] M. Kato, N. Onizawa, and T. Hanyu, "Design automation of invertible logic circuit from a standard HDL description," *Journal of Applied Logics*, vol. 8, no. 5, pp. 1311–1333, 2021.

[93] J. Kaiser, R. Faria, K. Y. Camsari, and S. Datta, "Probabilistic Circuits for Autonomous Learning: A Simulation Study," *Frontiers in Computational Neuroscience*, vol. 14, Feb 2020, Art. no. 14.

[94] N. Onizawa, S. C. Smithson, B. H. Meyer, W. J. Gross, and T. Hanyu, "In-Hardware Training Chip Based on CMOS Invertible Logic for Machine Learn-

ing," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 5, pp. 1541–1550, 2020.

[95] D. Shin, N. Onizawa, W. J. Gross, and T. Hanyu, "Training hardware for binarized convolutional neural network based on CMOS invertible logic," *IEEE Access*, vol. 8, pp. 188 004–188 014, 2020.

[96] S. D. Pyle, J. D. Sapp, and R. F. DeMara, "Leveraging stochasticity for in situ learning in binarized deep neural networks," *Computer*, vol. 52, no. 5, pp. 30–39, 2019.

[97] R. Zand, K. Y. Camsari, S. Datta, and R. F. Demara, "Composable Probabilistic Inference Networks Using MRAM-based Stochastic Neurons," *ACM J. Emerging Technol. Comput. Syst*, vol. 15, no. 2, 2019.

[98] P. Huembeli, J. M. Arrazola, N. Killoran, M. Mohseni, and P. Wittek, "The physics of energy-based models," *Quantum Machine Intelligence*, vol. 4, no. 1, p. 1, Jan 2022. [Online]. Available: https://doi.org/10.1007/s42484-021-00057-7

[99] M. Eisinger, R. Zand, and R. F. DeMara, "Training Optimization of Restricted Boltzmann Machines using a Contrastive Divergence Algorithm," in *2019 IEEE Southeastern Conference (SECon-2019)*, 2019.

[100] A. Roohi, S. Sheikhfaal, S. Angizi, D. Fan, and R. F. DeMara, "ApGAN: Approximate GAN for Robust Low Energy Learning From Imprecise Components," *IEEE Transactions on Computers*, vol. 69, no. 3, pp. 349–360, 2019.

[101] S. Srivastava and V. Sundararaghavan, "Generative and discriminative training of Boltzmann machine through quantum annealing," *Scientific Reports*, vol. 13, no. 1, p. 7889, 2023.

[102] S. Chowdhury, S. Niazi, and K. Camsari, "Mean-Field Assisted Deep Boltzmann Learning with Probabilistic Computers," in *Machine Learning with New Compute Paradigms*, 2023. [Online]. Available: https://openreview.net/forum?id=92YxqzuGSu

[103] J. Kaiser, A. Rustagi, K. Y. Camsari, J. Z. Sun, S. Datta, and P. Upadhyaya, "Sub-nanosecond fluctuations in low-barrier nanomagnets," *Physical Review Applied*, vol. 12, no. 5, 2019, Art. no.054056.

[104] N. Onizawa, K. Yano, S. Shin, H. Fujita, and T. Hanyu, "Self-Adaptive Gate Control for Efficient Escape From Local Minimum Energy on Invertible Logic," *IEEE Access*, vol. 11, pp. 44 923–44 931, 2023.

[105] A. Grimaldi *et al.*, "Spintronics-compatible approach to solving maximum-satisfiability problems with probabilistic computing, invertible logic, and parallel tempering," *Physical Review Applied*, vol. 17, no. 2, 2022, Art.024052.

[106] N. A. Aadit *et al.*, "Massively parallel probabilistic computing with sparse Ising machines," *Nature Electronics*, vol. 5, no. 7, pp. 460–468, 2022.

[107] S. Nikhar, S. Kannan, N. A. Aadit, S. Chowdhury, and K. Y. Camsari, "All-to-all reconfigurability with sparse and higher-order Ising machines," *Nature Communications*, vol. 15, no. 1, p. 8977, 2024.

[108] K. Lee, S. Chowdhury, and K. Y. Camsari, "Noise-augmented chaotic Ising machines for combinatorial optimization and sampling," *Communications Physics*, vol. 8, no. 1, p. 35, 2025.

[109] K. Selcuk *et al.*, "Connecting physics to systems with modular spin-circuits," *arXiv preprint arXiv:2404.19345*, 2024.

[110] J. Z. Sun, "Spin-current interaction with a monodomain magnetic body: A model study," *Physical Review B*, vol. 62, pp. 570–578, Jul 2000.

[111] Y. Xie, J. Ma, S. Ganguly, and A. W. Ghosh, "From materials to systems: a multiscale analysis of nanomagnetic switching," *Journal of Computational Electronics*, vol. 16, no. 4, pp. 1201–1226, 2017.

[112] L. Lopez-Diaz, L. Torres, and E. Moro, "Transition from ferromagnetism to superparamagnetism on the nanosecond time scale," *Physical Review B*, vol. 65, no. 22, 2002, Art. no. 224406.

[113] B. D. Cullity and C. D. Graham, *Introduction to magnetic materials*. John Wiley & Sons, 2011, ch. 7, pp. 197–237.

[114] A. Lisfi, J. Lodder, H. Wormeester, and B. Poelsema, “Reorientation of magnetic anisotropy in obliquely sputtered metallic thin films,” *Physical Review B*, vol. 66, no. 17, 2002, Art. no. 174420.

[115] Group: Modular Approach to Spintronics, Circular Nanomagnets: Experiment vs Theory. [Online]. Available: https://nanohub.org/groups/spintronics/circular_magnets

[116] Cadence Virtuoso Schematic Editor. Accessed: Jan 8, 2025. [Online]. Available: https://www.cadence.com/en_US/home/tools/custom-ic-analog-rf-design/circuit-design/virtuoso-schematic-editor.html

[117] A. K. Kaveev *et al.*, “Laser MBE-grown CoFeB epitaxial layers on MgO: Surface morphology, crystal structure, and magnetic properties,” *Physical Review Materials*, vol. 2, no. 1, 2018, Art. no. 014411.

[118] D. Olekšáková, S. Roth, P. Kollár, and J. Füzer, “Soft magnetic properties of NiFe compacted powder alloys,” *Journal of magnetism and Magnetic Materials*, vol. 304, no. 2, pp. e730–e732, 2006.

[119] K. Ounadjela, H. Lefakis, V. Speriosu, C. Hwang, and P. Alexopoulos, “Thickness dependence of magnetization and magnetostriction of NiFe and NiFeRh films,” *Le Journal de Physique Colloques*, vol. 49, no. C8, pp. 1709–1710, 1988.

[120] H. Kuru, N. Ç. Aytekin, H. Köçkar, M. Haciismailoğlu, and M. Alper, “Effect of NiFe layer thickness on properties of NiFe/Cu superlattices electrodeposited on titanium substrate,” *Journal of Materials Science: Materials in Electronics*, vol. 30, no. 19, pp. 17 879–17 889, 2019.

[121] S. Mitra, A. Ahmad, S. Chakrabarti, S. Biswas, and A. K. Das, “Investigation on structural, electronic and magnetic properties of Co_2FeGe Heusler alloy: experiment and theory,” *Journal of Magnetism and Magnetic Materials*, vol. 552, 2022, Art. no. 169148.

[122] A. Ahmad, S. Mitra, S. Srivastava, and A. Das, "Size-dependent structural and magnetic properties of disordered Co_2FeAl Heusler alloy nanoparticles," *Journal of Magnetism and Magnetic Materials*, vol. 474, pp. 599–604, 2019.

[123] K. Elphick *et al.*, "Heusler alloys for spintronic devices: review on recent development and future perspectives," *Science and technology of advanced materials*, vol. 22, no. 1, pp. 235–271, 2021.

[124] S. Miller and D. Childers, *Probability and Random Processes: With Applications to Signal Processing and Communications*. Elsevier Science, 2004.

[125] S. Hong, "Solving inference problems of Bayesian networks by probabilistic computing," *AIP Advances*, vol. 13, no. 7, 2023.

[126] S. Chowdhury, K. Y. Camsari, and S. Datta, "Accelerated quantum Monte Carlo with probabilistic computers," *Communications Physics*, vol. 6, no. 1, p. 85, 2023.

[127] M. Á. Carreira-Perpiñán and G. E. Hinton, "On Contrastive Divergence Learning," in *International Conference on Artificial Intelligence and Statistics*, 2005. [Online]. Available: https://api.semanticscholar.org/CorpusID:17861266

[128] A. Y. Ng, "Feature selection, L 1 vs. L 2 regularization, and rotational invariance," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 78.

[129] A. Haroon and S. Saurabh, "Image Completion using a Sparse Probabilistic Spin Logic Network," in *35th International Conference on VLSI Design (VLSID)*, 2022, pp. 281–286.

[130] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998. [Online]. Available: http://yann.lecun.com/exdb/mnist/

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

[132] A. Haroon and S. Saurabh, "Fault-Tolerant Design Framework for Probabilistic-Bit (P-Bit) Systems: Proposal and Analysis," *IEEE Transactions on Circuits and Systems I: Regular Papers*, pp. 1–13, 2025.

[133] T.-H. Nguyen, M. Imran, J. Choi, and J.-S. Yang, "CRAFT: Criticality-Aware Fault-Tolerance Enhancement Techniques for Emerging Memories-Based Deep Neural Networks," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 10, pp. 3289–3300, 2023.

[134] J. J. Zhang, T. Gu, K. Basu, and S. Garg, "Analyzing and mitigating the impact of permanent faults on a systolic array based neural network accelerator," in *2018 IEEE 36th VLSI Test Symposium (VTS)*, 2018, pp. 1–6.

[135] A. Ruospo, G. Gavarini, I. Bragaglia, M. Traiola, A. Bosio, and E. Sanchez, "Selective Hardening of Critical Neurons in Deep Neural Networks," in *2022 25th International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS)*, 2022, pp. 136–141.

[136] T. Cover and J. Thomas, *Elements of Information Theory*. John Wiley & Sons, 2006, ch. 2, pp. 13–24.

[137] A. Q. Zhang, A. M. Tosson, D. Ma, R. Fang, and L. Wei, "Stuck-at Faults Tolerance and Recovery in MLP Neural Networks Using Imperfect Emerging CNFET Technology," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, vol. 9, no. 2, pp. 168–175, 2023.

[138] I. K. Baek *et al.*, "Implementation of Bayesian networks and Bayesian inference using a $Cu_{0.1}Te_{0.9}/HfO_2/Pt$ threshold switching memristor," *Nanoscale Advances*, vol. 6, no. 11, pp. 2892–2902, 2024.

[139] B. Zhou, Y. Sun, D. Bau, and A. Torralba, "Revisiting the Importance of Individual Units in CNNs via Ablation," *CoRR*, vol. abs/1806.02891, 2018. [Online]. Available: http://arxiv.org/abs/1806.02891

[140] R. A. Amjad, K. Liu, and B. C. Geiger, "Understanding neural networks and individual neuron importance via information-ordered cumulative ablation," *IEEE*

Transactions on Neural Networks and Learning Systems, vol. 33, no. 12, pp. 7842–7852, 2021.

[141] Y. Leblebici and S. M. Kang, *CMOS Digital Integrated Circuits: Analysis and Design*. McGraw-Hill New York, 1996, ch. 7, p. 299.

[142] U. Ko and P. Balsara, “High-performance energy-efficient D-flip-flop circuits,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 1, pp. 94–98, 2000.

[143] Y. Zhang *et al.*, “Large-Area 1.2-kV GaN Vertical Power FinFETs With a Record Switching Figure of Merit,” *IEEE Electron Device Letters*, vol. 40, no. 1, pp. 75–78, 2019.

[144] S. Saurabh, *Introduction to VLSI Design Flow*. Cambridge University Press, 2023, ch. 21, pp. 461–467.

[145] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, “A Learning Algorithm for Boltzmann Machines,” *Cogn. Sci.*, vol. 9, no. 1, pp. 147–169, 1985.

[146] F. Tan, Y. Xia, and B. Zhu, “Link Prediction in Complex Networks: A Mutual Information Perspective,” *PLoS ONE*, vol. 9, no. 9, 2014.

[147] P. Kumar and D. Sharma, “A potential energy and mutual information based link prediction approach for bipartite networks,” *Scientific Reports*, vol. 10, no. 1, pp. 1–14, 2020.

[148] Z. Chen, N. L. Zhang, D. Y. Yeung, and P. Chen, “Sparse Boltzmann Machines with Structure Learning as Applied to Text Analysis,” *31st AAAI Conference on Artificial Intelligence*, pp. 1805–1811, 2017.

[149] M. Ferdosi, A. Gholamidavoodi, and H. Mohimani, “Measuring mutual information between all pairs of variables in subquadratic complexity,” in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 4399–4409.

Publications

1. **A. Haroon** and S. Saurabh, "Image Completion using a Sparse Probabilistic Spin Logic Network," in 35th *International Conference on VLSI Design (VLSID)*, 2022, pp. 281–286, doi: 10.1109/VLSID2022.2022.00061

2. **A. Haroon**, R. K. Ghosh, and S. Saurabh, "Implementation of Probabilistic Bits (Pbits) using Low Barrier Magnets: Investigation and Analysis," in 36th *International Conference on VLSI Design (VLSID)*, 2023, pp. 307-312, doi: 10.1109/VLSID57277.2023.00069.

3. **A. Haroon**, R. K. Ghosh, and S. Saurabh, "Impact of Non-Idealities on the Behavior of Probabilistic Computing: Theoretical Investigation and Analysis," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 71, no. 12, pp. 6279–6291, 2024, doi: 10.1109/TCSI.2024.3461770.

4. **A. Haroon** and S. Saurabh, "Fault-Tolerant Design Framework for Probabilistic-Bit (P-Bit) Systems: Proposal and Analysis," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, doi: 10.1109/TCSI.2025.3569769.

Brief Biodata of the Author

Amina Haroon obtained her Bachelor's Degree (B.Tech) in Electronics Engineering from Aligarh Muslim University in 2016 followed by a Master's Degree (M.Tech) in Electronics & Communication Engineering from Jamia Millia Islamia in 2018 where she was awarded a Gold Medal for securing First Rank in Master's Degree. Currently, she is pursuing a Ph.D. in the Department of Electronics and Communications Engineering at the Indraprastha Institute of Information Technology Delhi (IIIT Delhi), India. Her research interests include exploring spintronic devices for probabilistic computing with a focus on advanced materials and low-power computing solutions.