Efficient circuit-designs using spintronic devices

Deb, Suman

2019

Deb, S. (2019). Efficient circuit-designs using spintronic devices. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/94466>

<https://doi.org/10.32657/10220/49470>

**NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE**

**Efficient Circuit-Designs Using
Spintronic Devices**

Suman Deb

School of Computer Science and Engineering

A thesis submitted to Nanyang Technological University Singapore
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

July 16, 2019

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarized materials, and has not been submitted for a higher degree to any other University or Institution.

16.07.2019

Date

Suman Deb

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

16.07.2019

Date

Prof. Anupam Chattopadhyay

Authorship Attribution Statement

This thesis contains material from 3 papers published in the following peer-reviewed journals or conferences where I was the first and the corresponding author.

Chapter 3 is published as: Deb, S., Chattopadhyay, A., Yu, H., “Energy Optimization of Racetrack Memory-Based SIMON Block Cipher”, *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, July 2016, pp: 431-436.

The contributions of the co-authors are as follows:

1. I proposed the idea in the paper.
2. I carried out the implementations.
3. The paper was written by me.
4. The co-authors advised me from time to time about how to carry out my work.

Chapter 4 is published as:

1. Deb, S., Vatwani, T., Chattopadhyay, A., Basu, A., Fong, X., “Domain Wall Motion-based XOR-like Activation Unit With A Programmable Threshold”, *International Joint Conference on Neural Networks (IJCNN)*, July 2018, pp:1-8.
2. Deb, S., Vatwani, T., Chattopadhyay, A., Basu, A., Fong, X., “Domain Wall Motion-based Dual-Threshold Activation Unit for Low-Power Classification of Non-Linearly Separable Functions”, *IEEE Transactions on Biomedical Circuits and Systems (TBioCAS)*, 2018.

The contributions of the co-authors are as follows:

1. I proposed the idea in the paper(s).
2. I carried out the implementations.
3. The paper was written by me.

4. The co-authors advised me from time to time about how to carry out my work.

16.07.2019

Date

A handwritten signature in blue ink that reads "Suman Deb". The signature is written over a horizontal dashed line.

Suman Deb

THESIS ABSTRACT

Efficient Circuit-Designs Using Spintronic Devices

by

Suman Deb

Doctor of Philosophy

Supervisor: Prof. Anupam Chattopadhyay

School of Computer Science and Engineering

Nanyang Technological University, Singapore

The last 50 years of Moore's Law have witnessed a continuous shrinkage of CMOS technology node in the sub-micron range. While this has facilitated more and more transistors to be accommodated in the same silicon area, thereby increasing the computation power of microprocessors, smaller transistors drain more power in their OFF state. Due to increasing standby or leakage power, they cannot be downsized further. This, so called, power wall ignited the interest in non-volatile technologies like Spintronics, Phase Change Memory (PCM) and Resistive RAM (ReRAM). Spintronics, with devices like, Spin Transfer Torque (STT)-based Magnetic Tunnel Junctions (MTJs) and Racetracks (RTs) in its arsenal, has emerged as a prospective paradigm for future logic- and storage-applications. Spintronics promise for efficient processing and storage of information lies in its attributes of non-volatility, excellent integration-density, near-unlimited endurance and compatibility with CMOS process-technology.

While spin devices make excellent candidates for storage, their capability to realize logic functions remains a relatively-new and less-chartered area of research. One of the primary reasons for this is that, despite multiple optimizations at technology-, device- and circuit-level, spin-based circuits suffer from poor energy-efficiency due to the high energy consumed by write operations. In this thesis, we first aim to address this challenge. We propose design optimizations to reduce the number of write operations in Domain Wall motion-based logic circuits, and therefore, achieve overall gain in energy performance. As a case study, we perform in-depth study of the cutting-edge cryptographic block cipher SIMON, using experimentally validated Verilog-A models of MTJ and Racetrack Memory. For this benchmark, simulations demonstrate $4.65\times$ reduction in computation energy, $2.66\times$ improvement in computation delay and $1.71\times$ reduction in transistor count compared to its base implementation using Racetrack Memory.

Recently, a great deal of scientific endeavour has been devoted to developing spin-based neuromorphic platforms owing to the ultra-low-power benefits offered by spin devices and the inherent correspondence between spintronic phenomena and the desired neuronal, synaptic behavior. Whereas domain wall motion-based threshold activation unit has previously been demonstrated for neuromorphic circuits, it remains well-known that neurons with threshold activation cannot completely learn non-linearly separable functions. Our research in the later half of the thesis addresses this fundamental limitation by proposing two novel domain wall motion-based dual-threshold activation units (AUs). Furthermore, new learning algorithms are formulated for neurons with these activation functions. We perform 100 trials of 10-fold training and testing of our neural networks on real-world data sets taken from the UCI machine learning repository. On an average, we observe that:

1. The learning algorithm for the first proposed-AU performs $1.08\times-1.82\times$ better than that of the perceptron learning algorithm.
2. The learning algorithm for the second AU achieves $1.04\times-6.54\times$ lower misclassification rate (MCR) than the traditional perceptron learning algorithm. In circuit-level simulation, the neural networks with the proposed activation unit are observed to outperform the perceptron networks by as much as $2.98\times$ in MCR. The energy consumption of a neuron having the proposed domain wall motion-based activation unit averages to 35 fJ approximately.

In the next step of the roadmap of this PhD work, we investigate another interesting application of a neuron with the latter AU (proposed above). As we know, a Boolean function, before being mapped to hardware, undergoes representation in terms of basic logic-primitives followed by its optimization (w.r.t. size, depth, etc.). Today's state-of-the-art EDA tools primarily use AND-Inverter Graphs (AIGs), Majority-Inverter Graphs (MIGs) and XOR-Majority Graphs (XMGs) for representing Boolean functions. To be able to utilize the existing EDA tools for implementing spin-based logic circuits, it is important that the logic primitives in these data structures can be natively realized by spin devices. We demonstrate how the XMGs and the AIGs synthesized by EDA flows can be more-efficiently mapped to spintronic fabric using a domain wall motion-based XOR-primitive. Extensive circuit-level simulations are carried out to benchmark this XOR-gate

over other domain wall motion-based gates. In addition, we develop a device-to-system simulation-framework to precisely evaluate the post-mapping (to domain-wall gates) performances of synthesized networks. Our study over several challenging benchmark-suites shows that the use of this XOR-gate improves the {size, depth, size·depth, energy, EDP} performances of mapped XMGs and AIGs by average values of {31.54%, 19.00%, 41.56%, 38.03%, 45.47% and {13.39%, 9.26%, 17.74%, 15.90%, 19%}, respectively.

Acknowledgement

First and foremost, I would like to express my deep gratitude to my advisor, Prof. Anupam Chattopadhyay, for his guidance, advice, and priceless supervision throughout my Ph.D. study. He has been of immense help during the years of my PhD without which this PhD won't have been possible. I express my heartfelt gratitude to him for giving me the freedom to often disobey him and pursue my own interest.

I would like to convey many thanks to my dear friends-cum-colleagues at the Hardware & Embedded Systems Lab (HESL) for their suggestions and encouragement, especially, Debjyoti Bhattacharya, Anubhab Baksi, Gaurav Chauhan, Arko Dutta and Ahmed Ibrahim Samir. I would also like to thank the laboratory executive, Chua Ngee Tat *a.k.a* Jeremiah, of HESL for providing logistic support.

I would also like to thank my close friends, Costerwell, Mayank, Bali, Sonu and Devadeep, in Singapore.

Last but not the least, I am indebted to my parents for understanding my wish for higher studies and giving their heart and soul to raise me to the better human being I am today.

Contents

1	Introduction	1
1.1	Background	2
1.1.1	Magnetic Tunnel Junction	2
1.1.2	Domain Wall Motion	5
1.2	Challenges and Motivation	7
1.3	Research Goals	9
1.4	Contribution	10
1.5	List of Publications	11
1.6	Thesis Outline	12
2	Literature Review	17
2.1	Tunnel Magneto-Resistance (TMR)	17
2.2	Write Avoidance Techniques	18
2.3	Spin-based Neuromorphic Computing	25
2.4	Motivation for Research	30
3	Energy Optimization of Spin-based Boolean Logic	32
3.1	Introduction	32
3.1.1	SIMON Block Cipher: A Case Study	33
3.2	Racetrack Memory-based Implementation of SIMON 32/64	34
3.2.1	Hardware Stages	36
3.2.2	Round Counter	38
3.2.3	Control Signals	39
3.2.4	Key Expansion	40
3.2.5	Encryption	41
3.2.6	Simulation and Models	42
3.2.7	Experimental Results	43
3.3	Energy Optimization	43
3.3.1	Hardware Stages	45
3.3.2	Round Counter and Control Signals	45
3.3.3	Key Expansion	45
3.3.4	Encryption	46
3.3.5	Results	47
3.4	Conclusion	47

4	Spintronic Activation Unit for Classifying Linearly Inseparable Functions	49
4.1	Introduction	49
4.2	Proposed Neurons	53
4.2.1	1 St Proposed Neuron (PN-1): Single Tunable-Threshold	55
4.2.1.1	Proposed Activation Unit	55
4.2.1.2	Proposed Learning Algorithm (LA-1)	57
4.2.2	2 nd Proposed Neuron (PN-2): Dual Tunable-Threshold	60
4.2.2.1	Proposed Neural Activation Unit	60
4.2.2.2	Proposed Learning Algorithm (LA-2)	61
4.3	Neural-Network Training and Neuromorphic-Circuit Simulation	64
4.4	Results and Analysis	71
4.4.1	Classification Performance	72
4.4.1.1	Learning Algorithm for Neurons with Dual-threshold AUs	72
4.4.1.2	Neuromorphic Implementations	74
4.4.2	Energy Performance of the Proposed Neurons	76
4.5	Conclusion	78
5	Efficient Mapping of XMG- and AIG-Synthesized Spintronic Circuits Using Domain Wall Motion-based XOR-Gate	80
5.1	Introduction	81
5.2	Proposed XOR-gate	83
5.2.1	Domain-Wall Device	83
5.2.2	XOR Operation	85
5.3	Device-to-System Simulation	87
5.3.1	Device Level	87
5.3.2	Gate Level	89
5.3.3	Network Level	90
5.3.3.1	AIG-based Synthesis	91
5.3.3.2	MIG-based Synthesis	92
5.3.3.3	XMG-based Synthesis	92
5.4	Results and Analysis	95
5.5	Conclusion	102
6	Conclusion and Future Research	103
6.1	Future Research	104
6.1.1	Further Optimization of Spin-based SIMON	104
6.1.2	Multi-Layered Network of Dual-Threshold Neurons	105
6.1.3	MRAM-based In-Memory Acceleration of ANNs	105

Bibliography	107
---------------------	------------

List of Figures

1.1	Comparison of programming energy of various non-volatile technologies [1]	3
1.2	Write Endurance <i>vs</i> Write Cycle Time for various technologies [2] .	3
1.3	Comparison of Emerging Technologies [3]	4
1.4	Magnetic Tunnel Junction	4
1.5	Writing into a Magnetic Tunnel Junction	14
1.6	Domain Wall nano-wire	15
1.7	Domain wall motion	15
1.8	Energy v/s Delay for switching an MTJ [4]	15
1.9	Variation of switching current and switching delay with transistor-width	16
2.1	1T1MTJ cell of an MRAM	20
2.2	Domain Wall motion-based MRAM-cell [5]	23
3.1	SIMON encryption scheme [6].	34
3.2	Circular shifting of bits in RT for: (a) Propagation of on-state (b) Round counter (c) Generating the LSB of C_i for i^{th} round of key expansion. The arrows indicate the sense of circular shifting.	36
3.3	RM-based circular shifter.	37
3.4	STT-RM-based AND logic unit.	38
3.5	Control signals.	39
3.6	SIMON encryption unit. O_1 , O_2 , O_3 and O_4 are the outputs of the intermediate logic operations shown in Fig. 3.1	41
3.7	STT-MTJ composite gate for encryption.	44
3.8	SIMON encryption with composite gates.	46
4.1	Artificial neuron model	50
4.2	Single-layered ANN	51
4.3	2-input OR	52
4.4	2-input XOR	52
4.5	XOR using threshold neurons	54
4.6	(a) Dual-threshold function (b) XOR using neuron with dual-threshold AU	54
54	. ((a)).	
54	. ((a)).	

54 . ((b)).	
54 . ((b)).	
4.7 Proposed DW motion-based neural AU for non-linearly separable functions.	56
4.8 Modeling the dual-threshold function.	57
4.9 Training of the proposed activation unit.	58
4.10 Proposed domain wall motion-based AU with two programmable thresholds.	60
4.11 Behavior of PN-2 during training.	63
4.12 Neuromorphic architecture of single-layered network of the proposed neuron. The reading circuitry is shown for one neuron only. For illustration purpose, the neurons are shown to be of the 2 nd proposed-type.	67
4.13 Effect of reduced input precision on network performance. The network here is of PN-2s.	68
4.14 Iris dataset	71
4.15 MONK-2 (test+train) dataset	72
4.16 User Knowledge Modeling dataset	72
4.17 Wall-Following Robot Navigation Data (sensor-readings-2)	73
4.18 Variation of network performance with the length of DW strip for Iris dataset	75
4.19 Variation of network performance with the length of DW strip for MONK-2 (test+train) dataset	76
4.20 Variation of network performance with the length of DW strip for User Knowledge Modeling dataset	76
4.21 Variation of network performance with the length of DW strip for Wall-Following Robot Navigation Data (sensor-readings-2)	77
5.1 Spin-Memristor Threshold Logic (SMTL) gate.	81
5.2 XOR using SMTL gates.	82
5.3 DW motion-based device for the proposed XOR-gate.	84
5.4 Circuit-level implementation of the proposed XOR-gate.	85
5.5 Timing diagram of the proposed XOR-gate.	86
5.6 Simulation framework for DW motion-based logic networks.	88
5.7 Domain wall motion-based device for realizing Inverter, AND and Majority functions.	90
5.8 Synthesized Circuit	91
5.9 Mapped circuit with DW-based buffers.	91
5.10 Phase sequence in different gate-levels of the mapped circuit.	92
5.11 Baseline XOR gate from Fig. 4(b) of [7].	98

List of Tables

1.1	<i>n</i> -variable Boolean Functions: Linearly Separable <i>vs</i> Non-linearly Separable	8
3.1	Truth Table of AND gate	41
3.2	Simulation parameters	42
3.3	Encryption energy of 2-input logic gate based implementation. . . .	43
3.4	Energy performance of composite gate-based SIMON 32/64.	47
4.1	Truth Table of Neuron with Eqn. 4.3 as Activation Unit	53
4.2	Split-up of Datasets for training, validation and testing	65
4.3	Physical Parameters of the Magnetic Strips used in the Simulation of a Domain Wall Motion-based AU	70
4.4	LA-2 <i>vs</i> LA-1 <i>vs</i> Perceptron LA	73
4.5	Classification Performance of Neuromorphic Implementations	74
4.6	Energy Performance of Neuron Implementations	77
4.7	Energy Dissipated in b_2 of PN-2	78
5.1	Truth Table of the Proposed Gate	87
5.2	Physical Parameters used in the Simulation of the Device Proposed in Fig. 5.3	89
5.3	Energy Consumption of Domain Wall Motion-based Logic Gates . .	93
5.4	Figures-of-Merit of XMGs Mapped to DW Motion-based Native Gates	94
5.5	Figures-of-Merit of {AND, Inverter}-Mapped AIGs	95
5.6	Figures-of-Merit of {XOR (proposed), AND, Inverter}-Mapped AIGs	96
5.7	Figures-of-Merit of {XOR (proposed), AND, Inverter}-Mapped AIGs	97
5.8	Energy Values of the Proposed and Baseline [7] XOR-gates	98

1

Introduction

The past 50 years of Moore's Law have witnessed an unprecedented amount of research effort carried out in academia as well as industry for continuous downscaling of the CMOS technology node. As the feature size of transistors continued to downsize in the deep sub-micron range [8], the cost and performance of computing improved, but the increasing leakage-current soon became a critical bottleneck. With billions of CMOS transistors integrated in a single chip, the leakage current contributed by all of them adds to a whopping amount of chip-energy. Consequently, the overall computation cost becomes unaffordable and, also, the reliability of the system deteriorates significantly. The quest for overcoming this key limitation to Moore's law motivated the researchers to investigate a number of alternative device-technologies. Non-volatile technologies, like

PCRAM, ReRAM, Spintronics etc. have especially emerged as promising candidates in the post-CMOS era. Of these, spintronic devices like Magnetic Tunnel Junctions (MTJs) [9] and Domain Wall nano-wires [10] render multiple superior features. For example, as shown in Fig. 1.1, the range of programming energy of STT-MRAM (spintronic memory made of MTJs) is lower than the other non-volatile technologies. Besides non-volatility, the other property that singly puts spintronics ahead of other post-CMOS technologies is its near-unlimited endurance (see Fig. 1.2) to write operations. Additionally, spin devices demonstrate excellent compatibility with CMOS fabrication process and can potentially achieve high integration-density (Fig. 1.3). Other useful properties of spin devices include high scalability, radiation hardness and 3-D integration [11, 12]. Since data is not stored in the form of charge in MTJs, it cannot be flipped or corrupted by charged radiation in outer space. Hence, these devices are suitable for storage applications in satellites. Other charge-based non-volatile emerging technologies are not radiation hardened. The presence of all these desirable features in a single device renders spintronic devices potential building blocks for implementing lightweight – low-power, low-area – architectures with applications in logic as well as storage. As a result, it makes sense to explore methods and techniques that can leverage the inherent benefits of spin devices for enhancing the performance of modern-day computing systems.

1.1 Background

1.1.1 Magnetic Tunnel Junction

Fig. 1.4 illustrates a Magnetic Tunnel Junction (MTJ) as a multi-layered device. It consists of an ultra-thin insulating layer of non-magnetic material, like MgO, sandwiched between two ferromagnetic (FM) layers. The upper FM layer is called the ‘Reference’ layer whereas the lower FM layer is called the ‘Free’ layer. The Reference layer has its magnetic orientation pinned along a permanent direction

FIGURE 1.1: Comparison of programming energy of various non-volatile technologies [1]

FIGURE 1.2: Write Endurance *vs* Write Cycle Time for various technologies [2]

with the help of additional layers (not shown here). On the other hand, the magnetization of the Free layer can be re-oriented externally. The magnetic orientation of the Free layer with respect to that of the Reference layer determines the net resistance of the MTJ. This phenomenon is called the Tunnel Magneto-Resistance (TMR) effect. The resistance of the MTJ is low when the Free layer is magnetized in the same direction as the Reference layer and high when they are anti-parallel

Feature	SRAM	eDRAM	STT-RAM	PCRAM
Density	Low	High	High	Very high
Speed	Very fast	Fast	Fast for read; slow for write	Slow for read; very slow for write
Dynamic power	Low	Medium	Low for read; very high for write	Medium for read; high for write
Leakage power	High	Medium	Low	Low
Nonvolatility	No	No	Yes	Yes
Scalability	Yes	Yes	Yes	Yes
* eDRAM: embedded DRAM; PCRAM: phase-change RAM; STT-RAM: spin-transfer torque magnetic RAM.				

FIGURE 1.3: Comparison of Emerging Technologies [3]

FIGURE 1.4: Magnetic Tunnel Junction

to each other. The strength of the TMR effect is measured by the ratio $\frac{(R_{ap}-R_p)}{R_p}$, where R_p and R_{ap} represent the resistances of the MTJ in the parallel and the anti-parallel states, respectively. Higher the TMR ratio, more the robustness of the device against sensing errors, since larger TMR creates a greater difference between R_p and R_{ap} . The existence of TMR effect allows the MTJ to be used as a medium for storing binary digital data. A bit is stored in an MTJ in the form of magnetization of its free layer relative to that of the fixed layer. Bits 1 and 0 are stored in the anti-parallel and parallel states of the MTJ respectively. A new bit is written in an MTJ by passing spin polarized current - having density greater than a critical value - through it. This current induced writing is based on transfer of spin angular momentum between the writing current and the magnetization of the free layer. The direction of the current determines the bit written in the MTJ. When the current flows from the fixed layer to free layer, a 0 is written. On the other hand, when the current flows from the free layer to the fixed layer, a 1 is

written. Fig. 1.5 depicts the resultant bit that is written into the MTJ for different directions of write current and different initial bits.

1.1.2 Domain Wall Motion

Another well-known spintronic phenomenon is the Domain Wall (DW) motion in ferromagnetic nano-wires. All magnetic objects are made up of numerous tiny regions called domains. Each domain has its own magnetization and acts like a permanent magnet. A DW is a tiny region of change of magnetization, that exists at the boundary between two domains with opposite magnetizations. As shown in Fig. 1.6, a series of DWs can exist along the length of a magnetic nano-wire. Notches are etched along the edges of the wire to act as pinning sites for these DWs. The DWs can be depinned from these notches and shifted along the wire by passing a current with density, J_{app} , greater than a minimum threshold, J_{th} , through the wire [13–17]. The magnitude of J_{th} depends on the width of the applied current-pulse. The DW moves in the direction of the incoming electrons and its speed is determined by J_{app} ($> J_{th}$). As can be seen in Fig. 1.7, reversing the applied current reverses the direction of DW motion.

The current-induced motion of DWs is caused by the physical phenomenon of ‘spin-momentum transfer’ [18], [19]. When current is passed through a ferromagnetic nano-wire, its constituent electrons get spin-polarized due to spin-dependent electron scattering in the wire. As a result, the current gains an overall spin angular momentum. When this current passes through a DW, its spin-polarization undergoes rotation in order to align with the local magnetization in the DW region. Assuming that the net spin momentum of the system remains constant, this change in the current’s spin angular momentum is transferred to the DW. This exerts a spin-transfer torque (STT) on the DW magnetization, thereby, causing the DW to move along the wire. The effect of current-induced STT on the overall

magnetization dynamics of a magnetic system is modeled by the Landau-Lifshitz-Gilbert-Slonczewski (LLGS) equation as:

$$\frac{\partial \mathbf{m}}{\partial t} = -|\gamma| \mathbf{m} \times \vec{\mathbf{H}}_{res} + \alpha (\mathbf{m} \times \frac{\partial \mathbf{m}}{\partial t}) + \vec{\tau}_{stt} \quad (1.1)$$

where, $\mathbf{m}$ is a unit vector in the direction of magnetization of the ferromagnetic layer, γ is the gyro-magnetic ratio obtained using the relation: $\gamma = g\mu_B/\hbar$ (such that, g is the Lande g-factor, μ_B is the Bohr magneton constant and $\hbar$ is the Planck constant), $\vec{\mathbf{H}}_{res} = \vec{\mathbf{H}}_{ani} + \vec{\mathbf{H}}_{ext} + \vec{\mathbf{H}}_{exch} + \vec{\mathbf{H}}_{demag} + \vec{\mathbf{H}}_{th}$ (where, $\vec{\mathbf{H}}_{ani}$ is the uniaxial anisotropy field, $\vec{\mathbf{H}}_{ext}$ the external magnetic field, $\vec{\mathbf{H}}_{exch}$ the exchange interaction field, $\vec{\mathbf{H}}_{demag}$ the demagnetization field and $\vec{\mathbf{H}}_{th}$ the thermally induced variable field acting on the magnetization of the ferromagnet), α is the Gilbert damping constant and $\vec{\tau}_{stt}$ is the Slonczewski term. The first term in the right hand side of Eq. (1.1) represents the precessional motion shown by the magnetization of the ferromagnet about the resultant magnetic field, $\mathbf{H}_{res}$. The middle term denotes the damping of the magnetization towards $\mathbf{H}_{res}$ and the last term signifies the spin transfer torque exerted by the spin-polarized current flowing through the layer and is given by [20]:

$$\vec{\tau}_{stt} = -(\frac{\gamma\hbar}{2eM_sV})[\mathbf{m} \times (\mathbf{m} \times \vec{\mathbf{I}}_s) + \beta(\mathbf{m} \times \vec{\mathbf{I}}_s)] \quad (1.2)$$

where, $\vec{\mathbf{I}}_s$ is the vector along the direction of current flow and β is the non-adiabatic STT strength.

Experiments carried out in the recent past have successfully demonstrated STT-driven DW motion with high DW velocity above 50 m/s [21] and low threshold current-density of the order of 10^6 A/cm^2 [22]. Moreover, shrinking the thickness and/or width of the ferromagnetic nano-wire can reduce the threshold current for DW depinning as well as the current needed to achieve a particular DW velocity. These optimistic findings and opportunities encourage the application of DW motion for realizing high-speed, low-power logic and memory units.

1.2 Challenges and Motivation

The domains in which spintronics has found application can broadly be classified into three main classes – *storage*, *boolean logic* and *neuromorphic computing*. While spin devices have traditionally been explored as a storage technology, its potential for performing Boolean logic and neuromorphic computing remains a relatively-new area of research. The major challenges that are encountered by engineers while designing spin-based circuits for logic and neuromorphic computing are as follows:

1. *Boolean Logic*: One of the major obstacles in the path of using spin-based devices for implementing logic functions (and also, high density storage) is that they have a high write energy. Current of the order of 100s of μA is needed to flip the magnetization of the free layer of MTJ cells. Fig. 1.8 from [4] shows the range of write currents that can be applied to write data into these devices. We can observe that the free layer magnetization can be switched by applying a longer pulse of current. Even for longer pulse-widths, the magnitude of the write current is a few hundred μAs .

To verify this issue, we simulated the write circuit in Fig. 1.9 and obtained the write currents and the corresponding delay by varying width of the transistors (STMicroelectronics 65nm technology library) in the write circuit. The MTJ model used here is described in later part of the report. The write current magnitudes and their corresponding delays are tabulated below. We can notice that in our simulations also, the current pulse amplitude for writing into an MTJ increases as the pulse width decreases. For fast operations, larger current pulses can be used. Higher write current increases the probability of the junction getting damaged. The minimum width of transistor that can be used for writing is $0.2\mu\text{m}$. Below this, the current sourcing ability of the transistor is insufficient for writing into the MTJ. For all value of transistor width, the write current remains in the range of 100-200 μA range. The lowest writing current is 127.8 μA . The high energy consumed by the write operations makes spin-based logic implementations inefficient. Also,

it increases the size of the access transistors of MTJ-cells in MRAM and impedes high storage density.

2. *Neuromorphic Computing*: Recently, a great deal of scientific endeavour has been devoted to developing spin-based neuromorphic platforms owing to the ultra-low-power benefits offered by spin devices and the inherent correspondence between spintronic phenomena and the desired neuronal, synaptic behavior. Domain wall motion-based threshold activation unit has previously been demonstrated in literature, for neuromorphic circuits. In spite of being hardware-friendly, threshold function suffers from limited functionality. It remains well known that threshold activation units can only learn linearly separable functions. But, functions very often lack the property of linear separability.

With increasing n in n -variable Boolean functions, the number of non-linearly separable functions grows much more rapidly than the number of linearly separable functions. The quantum of this growth is explicitly reflected in Table 1.1 [23]. For $n > 4$, the ratio of the number of linearly separable functions to the total number of n -variable Boolean functions, 2^{2^n} , becomes significantly small [24]. Besides, real-world applications, such as

TABLE 1.1: n -variable Boolean Functions: Linearly Separable *vs* Non-linearly Separable

n	No. of non-linearly separable functions	No. of linearly separable functions
1	2	2
2	8	8
3	184	72
4	64, 000	1, 536
5	4, 294, 881, 216	86, 080
6	$\approx 1.844 \times 10^{19}$	14, 487, 040
7	$\approx 3.4 \times 10^{38}$	8, 274, 797, 440
8	$\approx 1.15 \times 10^{77}$	17, 494, 930, 604, 032

classification, face recognition, etc., require a neural network to learn functions with high degree of linear inseparability. Their training data have multiple decision boundaries. Modeling such non-linearly separable functions accurately requires deep neural networks (DNNs) with multiple layers of hidden neurons. But, this comes at the costs of more-compute-intensive learning and evaluation processes and a significant increase in area, delay and energy overhead of the neuromorphic architecture. As a result, it becomes extremely challenging for system architects to design DNN-powered mobile platforms that can meet the tight energy constraints imposed by Internet of Things (IoT) applications

1.3 Research Goals

The research challenges reported in the previous section give rise to certain vital questions in mind. These are:

1. *How can we reduce the write-energy consumption of a spin-based logic-circuit by utilizing a Boolean property?* The advantage of this approach would be that it can be applied to supplement the already-existing device- and circuit-level approaches for mitigating the high write-energy of a spintronic logic circuit.
2. *Can an activation function capable of supporting learning non-linearly separable functions be realized using spintronic phenomenon? What will be the learning algorithm for a neuron with this special activation function? How will this algorithm perform on real-world datasets?*
3. *Can such an activation unit be utilized to reduce the overall performance of a spin-based Boolean-logic circuit?*

1.4 Contribution

The work presented in this thesis is an effort to answer the above-mentioned curiosities. The proposals made by this thesis basically aim at improving the prospects of spintronics in Boolean logic and neuromorphic computing applications. The contributions of this thesis are briefly summarized below. Whereas the first and last works in this list focus on Boolean logic, the remaining works aim at neuromorphic computing.

1. First, we propose design optimizations to reduce the number of write operations in spin-based logic circuits, and therefore, achieve overall gain in energy performance. As a proof-of-concept, we perform in-depth study of the cutting-edge cryptographic primitive SIMON using experimentally validated Verilog-A models of MTJ and domain wall. For this benchmark, simulations demonstrate $4.65\times$ reduction in computation energy and $2.66\times$ improvement in computation delay compared to its baseline implementation.
2. Second, we propose a novel domain wall motion-based dual-threshold activation unit with additional non-linearity in its function. Furthermore, a new learning algorithm is formulated for artificial neurons with this activation function. We perform 100 trials of 10-fold training and testing of our neural networks on real-world datasets taken from the UCI machine learning repository. On an average, the proposed algorithm achieves $1.04\times - 6.54\times$ lower mis-classification rate (MCR) than the traditional perceptron learning algorithm. In circuit-level simulation, the neural networks with the proposed activation unit are observed to outperform the perceptron networks by as much as $2.98\times$ in MCR. The energy consumption of a neuron having the proposed domain wall motion-based activation unit averages to 35 fJ approximately.
3. Third, we propose a variant of the above activation unit. The results suggest femto-Joule range energy consumption of a neuron with the proposed activation unit and $1.08\times - 1.82\times$ lower mis-classification rate (MCR) of

the proposed algorithm in comparison to the traditional perceptron learning algorithm.

4. Our next work is inspired by the above-proposed activation units and leads us to Boolean logic again. A Boolean function, before being mapped to hardware, undergoes representation in terms of basic logic-primitives followed by its optimization (w.r.t. size, depth, etc.). Today's state-of-the-art EDA tools primarily use AND-Inverter Graphs (AIGs), Majority-Inverter Graphs (MIGs) and XOR-Majority Graphs (XMGs) for representing Boolean functions. To be able to utilize the existing EDA tools for implementing spin-based logic circuits, it is important that the logic primitives in these data structures can be natively realized by spin devices. In this work, we demonstrate how the XMGs and the AIGs synthesized by EDA flows can be more-efficiently mapped to spintronic fabric using a domain wall motion-based XOR-primitive. Remember that XOR is a non-linearly separable function and can be realized by any of the activation units proposed above. We propose a XOR gate whose design derives its inspiration from these activation units. Extensive circuit-level simulations are carried out to benchmark this XOR-gate over other domain wall motion-based gates. In addition, we develop a device-to-system simulation-framework to precisely evaluate the post-mapping (to domain-wall gates) performances of synthesized networks. Our study over several challenging benchmark-suites shows that the use of this XOR-gate improves the {size, depth, size-depth, energy, EDP} performances of mapped XMGs and AIGs by average values of {31.41%, 18.93%, 41.42%, 37.85%, 45.28%} and {13.46%, 9.31%, 17.82%, 16%, 19%}, respectively.

1.5 List of Publications

The outcomes of the research done in this PhD are documented in the following publications:

1. **Deb, S.**, Chattopadhyay, A., Yu, H., “Energy Optimization of Racetrack Memory-Based SIMON Block Cipher”, *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, July 2016, pp: 431-436.
2. **Deb, S.**, Vatwani, T., Chattopadhyay, A., Basu, A., Fong, X., “Domain Wall Motion-based XOR-like Activation Unit With A Programmable Threshold”, *International Joint Conference on Neural Networks (IJCNN)*, July 2018, pp: 1-8.
3. **Deb, S.**, Vatwani, T., Chattopadhyay, A., Basu, A., Fong, X., “Domain Wall Motion-based Dual-Threshold Activation Unit for Low-Power Classification of Non-Linearly Separable Functions”, *IEEE Transactions on Biomedical Circuits and Systems (TBioCAS)*, 2018.
4. **Deb, S.**, Chattopadhyay, A., “Spintronic Device-Structure for Low-Energy XOR Logic Using Domain Wall Motion”, *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2019.
5. **Deb, S.**, Chattopadhyay, A., “Efficient Mapping of XMG- and AIG-Synthesized Spintronic Circuits Using Domain Wall Motion-based XOR-Gate”, *IEEE Transaction of Computer Aided Design of Integrated Circuits and Systems (TCAD)*, 2019 (submitted).

1.6 Thesis Outline

This thesis is organized as follows: Chapter II acquaints the reader with the literature related to spin-based logic and neuromorphic computing. Next, in chapter III, we describe an optimization method to reduce the number of write operations in spin-based logic circuits. Chapter IV introduces two novel domain wall motion-based neural activation units for classifying linearly inseparable functions. In chapter V, we present a domain wall motion-based XOR gate for improving

the mapping of XMG- and AIG-synthesized circuits to the spintronic fabric. Finally, we conclude the thesis with our closing remarks and suggestions for future research-directions in chapter VI.

FIGURE 1.5: Writing into a Magnetic Tunnel Junction

FIGURE 1.6: Domain Wall nano-wire

FIGURE 1.7: Domain wall motion

FIGURE 1.8: Energy v/s Delay for switching an MTJ [4]

FIGURE 1.9: Variation of switching current and switching delay with transistor-width

2

Literature Review

In the previous chapter, we identified some challenges in the two broad application-areas of spintronics – boolean logic and neuromorphic computing. This chapter introduces the reader to the previous state-of-the-art research done in these two areas.

2.1 Tunnel Magneto-Resistance (TMR)

The first experimental demonstration of TMR effect was in the paper [25]. It reported a TMR ratio of 14% for an $Fe/Ge/Co$ junction. This discovery ignited several other research efforts to achieve higher TMR ratios. Higher TMR ratio helps in achieving higher MRAM density and better Signal to Noise Ratio (SNR) in MTJ-based read heads in magnetic HDDs. The TMR effect is possible due

to spin-dependent scattering [26] of electrons in the ferromagnetic layers of the MTJ. $TMR = 2P_1P_2/(1 + P_1P_2)$ [25]. Therefore, more is the spin-polarization of the conduction electrons in the ferromagnetic layers, higher is the TMR ratio. [27] reports a $R_{antiparallel}/R_{parallel}$ ratio of 7.3 for $La_{0.7}Ca_{0.3}MnO_3/NdGaO_3$ MTJ. In [28], a $R_{antiparallel}/R_{parallel}$ ratio of 9.7 is shown for an MTJ device having $La_{0.67}Sr_{0.33}MnO_3$ as free and fixed layers and $SrTiO_3$ as the tunnel barrier material. In [29], Bowen *et al* demonstrated TMR of 1850% and an average spin polarization of at least of 95% for $La_{0.67}Sr_{0.33}MnO_3/SrTiO_3/La_{0.67}Sr_{0.33}MnO_3$ device. Though these studies were successful in achieving high TMR values, they were performed at low temperatures. These TMR values cannot be observed at room temperature as these materials have low Curie temperature. In 1995, Moodera *et al* [30] and Miyazaki and Tezuka [31] achieved TMRs of 11.8% and 18% respectively at room temperature using Al_2O_3 tunnel barrier. In 2004, Wang *et al* [32] replaced $NiFeCo$ with $CoFeB$ for the ferromagnetic layers and achieved TMR of 70.4% at temperature while using Al_2O_3 tunnel barrier. This value of TMR corresponds to a polarization factor $P = 0.61$ for $CoFeB$. $CoFeB$ offers higher TMR and has lower coercivity and coupling field for the free layer. Theoretical calculations [33, 34] suggested that higher TMR values can be achieved at room temperature for $Fe/MgO/Fe$. These motivated research interest in devices with MgO barrier. In 2004, Stuart Parkin of IBM Almaden Research Centre demonstrated TMR values upto 220% at room temp [35]. In the same year, similar results were reported in [36] using varying thickness of MgO barrier between Fe layers. The TMR ratio was found to increase with the increasing thickness of MgO .

2.2 Write Avoidance Techniques

Several techniques and approaches have been proposed at different abstraction levels to tackle the high write-energy consumption of MTJs. At the device level, sacrificing the non-volatility of MTJ cells has been proposed by [37, 38] to reduce their switching energy and delay. Switching energy can be reduced by

reducing the volume of the free layer, since fewer domains imply less energy ($= k_u * V$; k_u =anisotropy constant, V =volume) required to switch the MTJ. The non-volatility of an object is guaranteed as long as $k_u * V > k_B * T$ (k_B =Boltzman's constant; T =temperature). Since V is reduced to reduce the write energy, the volatility will be also be reduced. Lower write energy and delay makes MRAM more suitable for application in L2 cache. At the device-level, less current (and delay) is needed in switching Perpendicular Magnetic Anisotropy (PMA) MTJs than the in-plane counterparts [39]. It has been observed that employing Spin Orbit Torque (SOT) greatly reduces the minimum current needed for magnetization reversal [39].

[40] proposes an 'Early Write Termination' (EWT) technique at the circuit-level to minimize the energy consumed in write operations. They applied this technique on STT-based MRAM cache. The Fig. 2.1 shows an MTJ-based cell of MRAM array. It consists of an MTJ electrically connected to a pass transistor. The gate of the transistor is driven by the Word Line (WL). A current can pass for sensing or switching the MTJ only when the transistor is switched ON by its WL. The state of the MTJ can be sensed by applying a very small voltage of fixed polarity across the Bit Line (BL) and the Source Line (SL). On the other hand, the MTJ cell can be programmed by applying larger voltage of opposite polarities, depending on whether a 1 or a 0 is to be written. In EWT, the SL and the BL each have a pass transistor in them. When writing is initiated, EWT senses the current state of the MTJ by using the write current itself. As we know, write current is larger than read current. So, the sense amplifier used here to detect the state of the MTJ is different from the SA usually used for reading MTJs. Once read, the current state of the MTJ is stored in a CMOS latch. If the latched state is same as the new input, the writing is terminated by generating a signal to switch OFF the SL and BL. If the new input is not equal to the latched value, the writing process is uninterrupted and completes to flip the MTJ.

1. *Pros*: EWT reduces the writing delay in case of a redundant write, when the writing is interrupted. The writing delay is equal to the time required

for sensing. An average of 70% and 33% reduction in write and dynamic (read+write) energies is reported in comparison to STT-RAM L2 cache (without EWT) for different benchmarks. The Energy-Delay Product (EDP) of the STT-RAM cache improves by an average of 34% due to EWT.

2. *Cons:* This scheme uses extra circuitry like Multiplexer, latch, SA, pass transistors. This introduces energy and area overhead of 3.23% and 4.17% per cell per write.

FIGURE 2.1: 1T1MTJ cell of an MRAM

In [41], Bishnoi *et al.* proposed a technique that can avoid unnecessary write operations in MRAM. In this technique, whenever new data is to be stored in the MTJ, the current magnetization state of the MTJ is first read using the pre-charge sense amplifier (PCSA) based read circuit and is stored in a CMOS latch. The new input is first compared with the current state of the MTJ. The comparison is performed by a CMOS 2-input XOR gate. The inputs to the XOR gate are the new input - output of the PCSA read circuit - and the output of the latch. If the new input matches the current stored bit, the write operation is not performed. If they do not match, the write operation is performed. This technique prevents re-writing of the same data

into the MTJ cell. They tested this technique for power saving in MRAM writes and observed 68.9% reduction in total write power consumption. To achieve conditional writing, extra CMOS circuitry XOR and latch has been used. This has led to an increase in area by 0.68%. Since every memory-write operation is preceded by a read and a XOR operation, it incurs a delay overhead of 1.33

In the paper [7], H. P. Trinh *et al.* present a multi-bit adder circuit based on racetrack memory. The operands and results of addition are stored in racetrack memory. The adder circuit performs two operations - $Sum = A \oplus B \oplus Carry_{In}$ and $Carry_{Out} = AB + BCarry_{In} + Carry_{In}A$. The adder circuit performs a Pre-Charge Sense Amplifying (PCSA)-based sensing circuit to read the output of these two operations. The outputs are written into the write-head of racetracks. Shift pulses are applied to shift the output bits into the racetracks. Due to the bottleneck of high write and shift energies, the 8-bit proposed magnetic adder has a dynamic energy 6X that of CMOS 8-bit adder. To mitigate this bottleneck, they proposed a scheme that prevents redundant or unnecessary writes. The scheme comprises of a comparison circuit. The comparison circuit has two PCSA circuits. The clock input to the PCSA circuits are provided by the inverted outputs of the PCSA reading the new output. The PCSA circuits in the comparison circuit read the previous output when they are in the evaluation phase. The PCSA circuits in the comparison circuit each have a pair of complementary outputs. These outputs drive the transistors in a write circuit such that a 1 is written into the write-head of RM only when the previous output is 0 and vice versa. This prevents writing if the new output matches the previous output. Due to this scheme, the switching energy is reduced by an average of 50%. Due to the use of additional PCSA circuits, the sensing energy is increased 3X, but this energy is around 3 orders of magnitude less than the switching energy.

In modern day multi-core processors, the number of cores on-chip continue increasing. As a result, the number of SRAM caches also increase proportionately. SRAM caches have 6T cell structure in comparison to a 1T1C

structure in DRAM. The 6T structure offers high speed but suffers from high leakage power. Owing to multiple caches per chip, caches consume a significant area of the multi-core processor. This results in high leakage power. MRAM, due to its high write endurance and zero standby power, seems to be a promising candidate. In [42], Sun *et al.* propose replacing the SRAM in L2 cache with MRAM 3D stacked on top of the processor. The storage density of MRAM cache is 4X that of SRAM cache. This improves the hit rates for all the benchmarks tested by them. The read energy and read latency of MRAM cache are comparable to those of SRAM cache. However, the Instruction Per Cycle (IPC) decreases for most of the benchmark applications due to high write latency incurred by the MRAM cache. While MRAM improves the leakage energy 10X, the high write energy negatively affects the dynamic energy of the L2 cache. To overcome this bottleneck, they proposed the use of a write buffer to hide the write latency. They also apply a read pre-emptive policy wherein read accesses are given higher priority over write accesses. To mitigate the dynamic energy overhead due to the high energy consumed by write operations, they proposed a hybrid structure wherein the L2 cache is partitioned between SRAM and MRAM. The hybrid structure shows an average improvement of 4.91% and 73.5% in IPC and total power respectively.

In [5], Fukami *et al.* developed a new cell for MRAM that is based on current induced STT-driven Domain Wall (DW) motion. It uses DW-motion for writing rather than switching of the magnetization of the free layer. The rationale behind this is that STT-driven DW-motion consumes less current than STT-driven switching of MTJ. It has a 2T1MTJ structure as shown in Fig. 2.2. PMA material is used as it has lower critical current density for DW motion in comparison to in-plane (IMA) material. The cell-structure consists of a free layer. The two ends of the free layer have fixed magnetizations. Their magnetizations are pinned in opposite directions (1 and 0) using pinning layer. The middle portion of the free layer i.e. the portion between these two ends stores the data (1/0) and is free to be re-oriented by

a current applied along the free layer. The middle portion has a DW, since its magnetization will be aligned opposite to that of one of the two ends at any time. The cell has an MTJ arrangement to read the state of the middle portion. In the figure, if a 1 is to be written, a current is passed from left to right. A current in the opposite direction writes a 0. Critical writing current and delay as low as $100\mu\text{A}$ and 2ns respectively could be achieved using this cell. The authors fabricated a 4kB memory-array of these cells. Being mo-

FIGURE 2.2: Domain Wall motion-based MRAM-cell [5]

tivated by the above work, Venkatesan *et al.* proposed a cache architecture using DW-motion-based cells in [43]. They proposed two cells TAPESTRI 1-bit and TAPESTRI multi-bit. The former is similar to the cell proposed by Fukami *et al.* in [5], but with a slightly different arrangement of pass transistors and WL, BL and BLB lines. TAPESTRI 1-bit cell has separate paths for reading and writing currents. This reduces the electrical stress undergone by the tunnel barrier of the MTJ during magnetization switching of the free layer. TAPESTRI multi-bit cell stores multiple bits per cell. It uses a racetrack for storing multiple bits. The write head of the racetrack is constituted by a TAPESTRI 1-bit cell. The free layer of this TAPESTRI 1-bit cell is co-planar but orthogonal to the racetrack. A new bit is injected

into this cell by passing a DW-shifting current current along the free layer of the TAPESTRI 1-bit cell. Then, a shift-pulse is applied along the race-track to push the new bit into the cell. A number of bits can be stored in a TAPESTRI multi-bit cell by applying multiple pulses. A TAPESTRI multi-bit cell offers higher storage density in comparison to utilizing multiple TAPESTRI 1-bit cells for storing multiple bits. But, it also has the disadvantage of incurring higher latency due to multiple shift operations. They used multiple write heads along the racetrack to reduce the number of shift operations. The cache architecture proposed uses TAPESTRI multi-bit cells to for the data blocks and TAPESTRI 1-bit cells for tag arrays in L2 cache. The L1 cache is completely made using TAPESTRI 1-bit cells to avoid the delay due to multiple shift operations in the other cell. They also utilized the concept of pre-shifting to predict the block that will be accessed next from L2 cache and hence, overcome the delay penalty incurred due to shift operations. Their proposed cache achieves 8.2X and 1.63X improvements in energy in comparison to SRAM and STT-RAM caches respectively. The read and write latencies of DWM TAPESTRI cache is comparable to SRAM cache. Its write latency is much less than STT-RAM cache. Due to reduced write currents, the access transistors are smaller in comparison to STT-RAM cache. This also contributes to reduced leakage power in comparison to STT-RAM cache.

[44] proposes to improve the performance of STT-RAM based L2 cache using a scheme that reduces the high energy consumed by write operations. The scheme is based on the observation that on an average 68.4% and 54.02% of the bytes and words respectively written to the L2 cache consist only of 0s. The authors of this paper propose to use special flags to indicate this phenomenon. If the flag is set to 1, it indicates that all the bits in the byte or word are 0s. If the flag-value is 0, it indicates the presence of non-zero bits. During write accesses, the cache line to be written to the cache is examined to detect the null bytes or words. The flags corresponding to the null bytes or words are set to 1. Rest of the flags are set to 0. Only the content in the

cache line corresponding to the zero-valued flag are written. During data access, these flags are first read to get the null bytes or words, followed by reading the bytes or words corresponding to the flags set to 0. The energy consumed by write operations is reduced by 73.78% and 69.30% when this scheme is applied at byte- and word-levels respectively. Due to larger tag arrays needed for accommodating the all-zero flags, the leakage power is increased in comparison to an all-MRAM L2 cache implemented without this scheme. However, the leakage power is less than that of an SRAM L2 cache. [45] is another work which proposes to reduce the write-energy consumption by exploiting the bit-pattern in the data written back to the L2 cache. Their observation is that the upper bits of words in cache lines are not as frequently altered as the lower bits. So, they apply a verify-before-write policy on the upper bits. The MTJ-cells of the upper bits are written only if they do not match the corresponding bits of the new input.

As can be observed in this survey, the techniques (except [7]) for avoiding the energy-expensive write operations are primarily proposed for memory applications. This is because the mainstream research in spintronics has been dominated by the prospect of spintronics as a non-volatile storage technology.

2.3 Spin-based Neuromorphic Computing

Artificial Neural Network (ANN) is a well-known computing model that promises to realize the extraordinary cognitive abilities of the human brain. ANNs have conventionally been implemented on CMOS hardware. However, these CMOS-based neuromorphic systems haven't been able to achieve brain-level performance due to limitations such as, high leakage power and lack of physical characteristics mimicking the biological functions of neurons and synapses. This fuelled research into several post-CMOS technologies like, memristors, spintronics, phase change memory etc. Different neural networks have been demonstrated in literature using different spin devices. Next, we

will give you a brief overview of the major works that have been done in spintronics in this direction.

Mrigank Sharad *et al* in [46] presented an analog design of associative memory for facial recognition. In this design, a crossbar array of memristors is used to perform weighted summation of input signals. The conductance values of memristors in the array act as the weights in this sum. The sum is then fed to a threshold activation function which compares the input to a reference. The output of the function is binary – 0 or 1 – depending on whether the input to the function is greater or smaller than the reference value. This paper implements the thresholding function using a ferromagnetic nano-strip containing a domain wall. An MTJ on the ferromagnetic strip detects its resistance state. The domain wall is depinned and set to motion only when the input current from memristive array is greater than the threshold current for domain wall motion. This changes the resistance of the MTJ. If the current is less than the critical value, the domain wall remains in its initial position in the strip and the MTJ resistance doesn't change. A dynamic CMOS latch reads the resistance of the MTJ. The memory design in this paper exploits this domain wall motion-based neuron as a current-mode comparator for digitizing analog current levels. A winner-takes-all algorithm is proposed for this purpose. Simulation results show that this design consumes $1000\times$ less computation energy than 45nm CMOS-based digital baseline designs.

Deliang Fan *et al* [47] utilized the crossbar array of memristors and domain wall motion-based thresholding unit in [46] to realize hierarchical temporal memory (HTM). The architecture of HTM is a tree of processing nodes. Each processing node consists of temporal pooler, spatial pooler and winner-takes-all circuits. The input to the processing node goes to the spatial pooler whose output in turn feeds the temporal pooler. The output of the temporal pooler goes to the winner-takes-all circuit for calculation of the winner index – the final output of the processing node. The fundamental operation in the temporal and spatial poolers is dot product of inputs and reference matrices. This dot product is obtained by applying voltages proportional to the inputs

to a crossbar array of memristors. The conductances of the memristors are programmed to values proportional to the elements of the reference matrices. The output of the poolers is digital. So, the analog output of the memristive crossbar arrays has to be digitized. An SAR-ADC performs this analog to digital conversion. The authors employ the domain wall motion-based thresholding unit in [46] to implement the low-power comparator component of the SAR-ADC. Simulation results indicate that this HTM design is more than $200\times$ more energy efficient than a digital baseline design.

The above two works employ the domain wall motion-based device for implementing the hard-limiting transfer function, i.e. the threshold activation function. This is a step function. Apart from the hard-limiting transfer function, there is another class of functions popularly known as the soft-limiting transfer function. This includes functions like, logistic sigmoid, hyperbolic tangent and saturated linear. Unlike hard-limiting functions, a neuron with soft-limiting transfer function can produce a continuous range of activation levels between '0 and 1, thereby, conveying more information in its output. [48] is one work which implements a soft-limiting function using a crossbar array of memristors and a domain wall motion-based activation unit. The crossbar architecture is similar to that in the above works. The activation unit consists of a ferromagnetic nano-strip housing a domain wall in it. An MTJ sits on top of the region of domain wall motion. The resultant resistance of the MTJ is a rational function of the domain wall position and can realize a soft-limiting non-linear transfer function. An artificial neural network designed using these soft-limiting activation units is shown to consume $2\times$ less energy than corresponding analog and digital CMOS implementations.

While the above works are based on domain wall motion for realizing thresholding function, [49] utilizes MTJ for the same. In this work, a neural network is implemented using MTJs and a crossbar array of memristors. Positive as well as negative weights are implemented in this crossbar array.

Each weight of the neural network is implemented by a pair of memristors – one connected to $+V_{dd}$, another to $-V_{dd}$. For a positive weight, the conductance of memristor connected to $+V_{dd}$ is greater than that of the one connected to $-V_{dd}$. For a negative weight, the former is smaller than the latter. The currents supplied by the crossbar array pass through MTJs. MTJs exhibit threshold currents for transition from anti-parallel (AP) to parallel (P) states and vice-versa. The critical current for transition from AP to P states is smaller than that for transition from P to AP states. The MTJs in this neural network implementation are initially set to AP state. If the current supplied by the crossbar array to an MTJ is greater than the critical current for AP $\rightarrow$ P transition, the MTJ resistance is set to a low resistance state. Otherwise, its resistance remains high. This neuromorphic structure achieves $1.63\times - 1.79\times$ reduction in power consumption in comparison to a 45nm CMOS-based digital baseline implementation.

It can be observed that all the above works employ memristors for implementing the synaptic weights. Abhronil Sengupta in his paper [50] proposes an all-spin neuromorphic architecture. Unlike the above works, this design uses domain wall motion not only for realizing the threshold activation unit, but also for implementing the synaptic weights. These weights consist of an MTJ fabricated on top of a ferromagnetic strip containing a domain wall. The domain wall strip acts as the free layer of the MTJ. Different position of domain wall results in different resistance values of the MTJ. So, the weights can be programmed to desired values by SOT-driven motion of the domain wall in the strip. The all-spin neural network consisting of 20 hidden-layer neurons and 26 outer-layer neurons exhibited $100\times$ more energy efficiency than 45nm-based digital and analog baseline implementations.

Spiking Neural Networks (SNNs) represent a popular class of neural networks, that is considered to be functionally closer to the biological neurons and synapses. In this class of neural networks, information is transmitted as a train of spikes. Data is encoded as the number of spikes transmitted

in a fixed duration. [51] proposes an SNN design using MTJ as the activation unit. This is possible due to the property of probabilistic switching of an MTJ. The switching probability characteristic of an MTJ matches the sigmoid function to a reasonable extent. The synaptic weights are realized using a resistive crossbar array. A deep SNN is implemented using such neurons. Its performance in digit recognition is tested on the MNIST dataset. Overall, this design consumes $20\times$ less energy than a digital baseline using 45nm CMOS technology.

[52] is another work that proposes spin-based SNN design. The focus of this work is to demonstrate the potential of domain wall motion for realizing synaptic weights in SNNs. The structure of the domain wall motion-based synapses in this SNN is similar to that in [50]. The linear relationship between the domain wall position and the device conductance lends synaptic plasticity to this spin device. The activation unit used in this design is a CMOS-based Leaky-Integrate-Fire (LIF) circuit. Spin orbit torque-induced programming of this spintronic synapse consumes a maximum energy of ≈ 15 fJ.

Another interesting work that deserves mention is [53] by Deming Zhang *et al.* This work proposes synaptic weight and activation unit using multiple MTJs vertically stacked as a single device. The limitation over using MTJ as a synaptic device is that it, unlike memristors, can exhibit only two conductance states. The proposed device presents the first step towards realizing multi-state spintronic synapse. The proposed device with n MTJs stacked vertically can exhibit 2^n discrete conductance states. By having different thicknesses of the capping and MgO layers of the individual MTJs in the proposed device, the critical switching current and RA of these MTJs can be made different. So, it becomes possible to independently program the individual MTJs of this device to different conductance states. Additionally, the authors demonstrate an activation unit using the proposed device. Such an activation unit realizes a multi-step transfer function, thereby, encoding more information in its output. The proposed device with n MTJs stacked

vertically produces an $n + 1$ -step function. They discuss a 3-layered all-spin artificial neural network implemented using this device for synapses and neurons. However, the energy and delay performances of this network are not provided in the paper.

[54] proposes a 3-terminal MTJ-Heavy Metal (HM) multi-layer device to realize stochastic neurons and synapses for SNNs. It consists of an MTJ stack on top of a HM layer. A current through the HM layer can switch the free layer of the MTJ due to spin-Hall effect. During inference, the proposed synapse modulates the input voltage spike as per its conductance. Being a binary device, it exhibits 2 conductance levels. During the learning phase, depending on the relative timings of the pre- and the post-synaptic spikes, this synapse can be conditionally programmed by passing a current through its HM layer. The proposed device also acts as a stochastic neuron by virtue of its probabilistic switching in response to a write current through its HM layer. An all-spin SNN of this device for digit recognition is simulated. The energy per spike consumed by this synapse during learning is $\approx 1000\times$ less than CMOS-based SNN implementations.

2.4 Motivation for Research

Spintronics being a storage technology, the write avoidance techniques mentioned in Section 2.2 of this chapter were primarily proposed for reducing the dynamic energy of spintronic memories like STT-RAM. While the problem of high dynamic energy also faces STT-based logic, almost none of these techniques remain valid in logic applications. Only the write avoidance techniques in [40] and [41] are applicable to spin-based logic circuits. Due to the absence of such techniques in the logic domain, we propose composite gates for minimizing the number of writes in CMOS-MTJ hybrid logic circuits. This proposal is specific to logic and doesn't apply to memories. The write-avoidance techniques in [40] and [41] are orthogonal to our technique and can be applied in conjunction with it.

Section 2.3 mentions about the various spin-based neural networks available in literature. The activation unit in [46], [47], [48], [49] and [50] realize either step or sigmoid function. A single neuron with either of these activation functions cannot learn a simple non-linear function like, XOR. More than one layers of such neurons are needed to learn such non-linear functions. In Chapter 4, we propose a DW motion-based neuron whose activation unit has additional non-linearity and is capable of learning the XOR function. [53] proposes an activation unit with multiple MTJs vertically stacked on top of each other. It realizes a multi-step function and should be able to learn the XOR function. Because it uses MTJs, high energy is required to switch the state of the device from one level to another. The activation unit proposed by us is based on DW motion and hence consumes very low power.

3

Energy Optimization of Spin-based Boolean Logic

¹

3.1 Introduction

The techniques mentioned in Section 2.2 for reducing the number of write operations are aimed at MRAM-based memories. Whereas some of these techniques can also be utilized in spin-based logic applications, many of them are based on properties that are specific to the memory architecture and so, they cannot be used in logic applications. Whether there exist special properties of logic operations that can be exploited to reduce the number

¹The research documented in this chapter has been published in [55].

of writes is something that needs to be answered. This is exactly what we explore in this chapter. We particularly make two contributions here. First, we perform design and benchmarking of the state-of-the-art lightweight security kernel SIMON using Spin Transfer Torque (STT)-based racetrack memory [55]. Racetrack memory (RM) is a term coined by Dr. Stuart Parkin of IBM Almaden Research Center for a memory made of domain wall nanowires (Fig. 1.6) called racetracks (RTs). Second, as a proof-of-concept, we apply our proposed technique to minimize the number of write operations in this implementation of SIMON. We achieve $4.65\times$ reduction in the total computation energy and $1.71\times$ reduction in transistor count of RT-based SIMON [55].

3.1.1 SIMON Block Cipher: A Case Study

SIMON is a lightweight block cipher introduced by the National Security Agency (NSA) in 2013. Compared to other block ciphers, SIMON consumes very less area in hardware while providing similar level of security. This makes it extremely suitable for area- and power-constrained applications. SIMON operates in multiple rounds and on plaintext and key(s) with different widths depending on its version [6, Table 3.1]. SIMON $2n/mn$ refers to a version that encrypts a $2n$ -bit plaintext word using an n -bit key in each round. n -bit keys for the initial m rounds are provided by the user. The keys for the subsequent rounds are computed by a key-schedule function that is specific to the version. A version-dependent constant c and a sequence z_j of 1-bit constants are used for deriving these keys.

FIGURE 3.1: SIMON encryption scheme [6].

3.2 Racetrack Memory-based Implementation of SIMON 32/64

SIMON 32/64 has a plaintext size of 32 bits and it iterates for 32 rounds. The encryption logic is given by:

$$x_{i+2} = S^1 x_{i+1} \cdot S^8 x_{i+1} \oplus x_i \oplus S^2 x_{i+1} \oplus k_i \quad (3.1)$$

where, x_{i+1} and x_i are respectively the upper and the lower 16 bits of the plaintext. S^1 , S^2 and S^8 represent left circular shift by 1, 2 and 8 bits respectively; k_i is the 16-bit key for i^{th} round of encryption. Fig. 3.1 shows the encryption logic for each round. The key expansion logic of SIMON

32/64 is:

$$k_{i+m} = c \oplus (z_j)_i \oplus k_i \oplus (I \oplus S^{-1})(S^{-3}k_{i+3} \oplus k_{i+1}) \quad (3.2)$$

S^{-1} and S^{-3} represent right circular shift by 1 and 3 bits respectively. 16-bit keys for the first 4 ($= m$) rounds are uploaded by the user. As a result, the key expansion routine needs to be executed for 28 rounds only.

The core feature of this hardware design described in the following subsections is that it is divided into stages. The idea behind this design choice is to reduce the energy consumption of the circuit. This is based on the observation that at any instant of encryption, only one of the stages or constituent operations is used for computation. The other stages remain unused at that instant. However, these unused stages would still be ON and consume high energy at that instant. To eliminate this waste of energy and achieve a lightweight implementation, we divide the entire circuit into stages such that only one stage is active at any instant. All other stages remain switched off at that instant. The stages are switched on successively and the computation proceeds from one stage to another. This design choice saves the huge energy that would otherwise be consumed by the write circuits in the unused stages.

Due to very low power required for DW motion, we used racetrack memory to implement round counter and the circular shifter for the switching on the stages.

In order to further optimize the design, we realized the circular shifts in Eq. 3.1 and 3.2 by directly wiring a logic gate to the MTJ in the bit position (of the operand) post shift. This way we avoided performing actual shifting of bits in a racetrack, thereby saving energy and area.

FIGURE 3.2: Circular shifting of bits in RT for: (a) Propagation of on-state (b) Round counter (c) Generating the LSB of C_i for i^{th} round of key expansion. The arrows indicate the sense of circular shifting.

3.2.1 Hardware Stages

The RM-based circuit of SIMON 32/64 comprises of 8 successive stages. In any clock cycle, only one of these stages remains in on-state, whereas the others are in off-state. We utilize circular shifting of the bits in an RT to implement this scheme. The RT shown in Fig. 3.2(a) stores 8 bits, of which only one is a 0 and the rest are 1s. Each bit-position in the RT corresponds to a stage. If a bit-position has a 0, the corresponding stage is in on-state. Else, the stage is set to off-state. As a result, only one of the stages is in on-state at any instant. One round of encryption completes upon the traversal of the 0 from stage 1 to stage 8. Subsequent rounds are executed upon the circular shifting of the bits in the RT.

The RT-based implementation of this circular shifter is shown in Fig. 3.3. MTJs (read-heads) are placed on the RT, at the bit-positions that correspond to the stages. These MTJs have their free layers *electrically insulated* from the RT, but *magnetically coupled* to it. This is done to electrically isolate

FIGURE 3.3: RM-based circular shifter.

the read-head sensing operations from each other as well as from the write-head switching operations and the DW-shifting operations in the RT. The free layer of these MTJs have their magnetic state aligned to that of the RT beneath them. The RT, however, forms the free layer of the MTJ in the write-head. The bit in the write-head is called the Redundant Bit (RB). It stores a copy of the rightmost bit in the RT. The circular shifting of the bits in the RT is carried out in 2 steps: First, the output of the Sensing Amplifier (SA) [56] reading the rightmost bit is written into the write-head. This updates the RB. Next, a current pulse is injected into the right shifting-port of the RT. The STT effect induced by the current shifts the DWs towards right and hence, the bits are circularly shifted. If the output of an SA reading a bit-position in the RT is 0, the corresponding stage is in on-state, else it is in off-state.

During the on-state of a stage, the SAs in it operate in evaluation phase and produce complementary outputs. As a result, switching current is triggered in the write circuits at these outputs (see Fig. 3.4). When the stage is in off-state, the SAs in it remain in pre-charge phase and both their outputs are

FIGURE 3.4: STT-RM-based AND logic unit.

1s. This prevents the flow of switching current in the write circuits. Also, simulations show that the SAs drain less power in pre-charge phase than in evaluation phase. The current pulses applied to the RTs for shifting DWs are clock gated, so that no pulses are injected during the off-state. Thus, the power consumption of the circuit is reduced by having the idle stages in off-state.

3.2.2 Round Counter

The routines for key expansion and encryption iterate upto the 28th and the 32nd round respectively. We utilize RT-based circular shifters to generate signals indicating the completion of these rounds. The function of these signals are mentioned in the next sub-section. Fig. 3.2(b) shows the circular shifting of bits in two RTs – A and B. Count_28 and Count_32 are respectively the outputs of the SAs reading the bits at position 2 of RT A and at position 31 of RT B. Besides the redundant bit, RT A stores 32 bits – four 1s and twenty-eight 0s – in the order shown. These bits are circularly shifted left by 1 position in each round. As a result, the Count_28 signal evaluates to 1

during rounds 29 to 32, thereby, indicating the completion of the 28th round. RT B stores a redundant bit and 4 bits – three 1s and a 0 – in the order shown in Fig. 3.2(b). When **Count_28** evaluates to 1, the bits in RT B are circularly shifted right by 1 position in every round. During the on-state of the 7th stage in round 32, the **Count_32** signal evaluates to 0 to indicate the completion of the 32nd round. These circular shifters are implemented as the one in Fig. 3.3.

FIGURE 3.5: Control signals.

3.2.3 Control Signals

Fig. 3.5 shows the various control signals generated in the circuit. Signals $\overline{\text{ON}}_i$ and **Key_ON_** i are respectively used for clock gating the i^{th} stage ($i=1$ to 8) of the encryption and the key expansion units. The key expansion unit remains in off-state during the rounds in which **Count_28** evaluates to 1. When **Count_32** is 0, the encryption unit produces the final ciphertext

and a new plaintext can be input to the circuit for encryption. **New_Key_In** becomes 1 during the on-state of stage 7 in rounds 29 to 32. Initial keys for encrypting the new plaintext are input when **New_Key_In** is 1. So, a total of four 16-bit keys are input.

3.2.4 Key Expansion

In Eq. 3.2, $c = 0\text{x}\text{fffc}$ and $z_j = z_0$. z_0 is a sequence of 1-bit constants. Each bit of this sequence is used in a different round of key expansion. $c \oplus (z_0)_i$ can be denoted by a 16-bit constant C_i . So, Eq. 3.2 is re-written as:

$$k_{i+m} = C_i \oplus k_i \oplus (I \oplus S^{-1})(S^{-3}k_{i+3} \oplus k_{i+1}) \quad (3.3)$$

The LSB of C_i is different for each round. We use an RT to store the LSBs for all the rounds of key expansion. As shown in Fig. 3.2(c), the bits in this RT are circularly shifted by one position in every round. We store the upper 15 bits of C_i and their complements in MTJs. Since these bits do not vary from one round to another, these MTJs are never switched during the operation of the circuit.

As in conventional CMOS circuits, 2-input gates, here XOR, are used to implement the key expansion logic in Eq. 3.2. The operation of each gate comprises of: (i) Reading: An SA evaluates the XOR network of MTJs [7]; (ii) Writing: The complementary outputs of the SA are written into the write-head of RTs; (iii) DW-shifting: One or more current pulses are injected into the RT to replicate the output for non-zero fan-out. The key expansion module consists of 8 stages and operates similar to the encryption module described next. Due to lack of space, we are not able to depict or explain the key expansion module in detail.

TABLE 3.1: Truth Table of AND gate

S^8x_{i+1}	S^1x_{i+1}	R_{left}	$S^8x_{i+1} \cdot S^1x_{i+1}$
0	0	$R_p/2$	0
0	1	$(R_{ap} + R_p)/2$	0
1	0	$(R_{ap} + R_p)/2$	0
1	1	$R_{ap}/2$	1

3.2.5 Encryption

The encryption logic comprises of XOR and AND operations. Fig. 3.4 shows the 2-input AND gate used in its implementation. Its truth table is given in Table 3.1. The value of resistance $R_{reference}$ in Fig. 3.4 is such that $R_p/2 < (R_{ap} + R_p)/2 < R_{reference} < R_{ap}/2$. Similar to the 2-input XOR gate explained above, the operation of the AND gate also comprises of reading, writing and DW-shifting, .

FIGURE 3.6: SIMON encryption unit. O_1 , O_2 , O_3 and O_4 are the outputs of the intermediate logic operations shown in Fig. 3.1

Fig. 3.6 depicts the encryption module. $k_i[15] \dots k_i[0]$ are the key bits produced by the key-expansion module for i^{th} round of encryption. The read, write operations of 2-input AND/XOR gates are performed in the same stage, followed by DW-shifting performed in the subsequent stage. During the on-state of stage 7 in the 32nd round, the encryption unit produces the final ciphertext bits — CIPHER[31:0] — and accepts new plaintext bits — p_{new} [31:0] — for encryption.

3.2.6 Simulation and Models

We simulated the CMOS-RT hybrid circuit of SIMON 32/64 in Cadence Virtuoso Analog Design Environment using the ST Microelectronics 65nm Process Design Kit (PDK) and the SPINLIB Verilog-A compact model for PMA CoFeB RT [57]. The CoFeB/MgO/CoFeB PMA MTJ models in [58] and [59] were used for the write-head and the read-heads on RT respectively. The models for switching in the write-head MTJ and for DW motion in RT are based on current-induced STT phenomenon.

TABLE 3.2: Simulation parameters

Parameters	Description	Values
V_{write}	Writing voltage	2 V[7]
W_{write}	Width of transistors in writing circuits	0.6 μm
V_{read}	Reading voltage	1.2 V[7]
W_{read}	Width of transistors in read circuits	0.135 μm
I_{shift}	Amplitude of shifting pulses	176 μA [57]

Table 4.7 presents values of some of the important parameters related to the circuit. We chose to use DW shifting pulses having amplitude of 176 μA [57]. The pulse-width corresponding to this amplitude is 0.75ns. Among all the stages, stage 2 of the key expansion unit requires the maximum number of DW shifting pulses. In this stage, three pulses are applied to the RT. We observe the minimum permissible interval between any two consecutive pulses for DW motion to be 0.15ns. So, the duration of on-state of each stage is set to 2.7ns. Employing transistors wider than 0.6 μm in write circuits results in switching delay much less than 2.7ns and larger switching current. It causes the write circuits to unnecessarily drain large switching current for longer period of time. Also, transistors smaller than 0.6 μm are not able to switch MTJ within 2.7ns. So, we used 0.6 μm wide transistors to implement the write circuits in the encryption and the key expansion units.

TABLE 3.3: Encryption energy of 2-input logic gate based implementation.

Component	Energy (nJ)
Writing	25.56
Domain Wall shifting	2.84
Reading + CMOS gates	0.077
Total Energy	28.473

3.2.7 Experimental Results

The above RT-based implementation of SIMON 32/64 takes 691.2ns to compute a ciphertext. The various components of its computation energy are listed in Table 3.3. Almost 90% of the total energy is consumed by write operations. DW shifting consumes $\approx 9.8\%$ of the total energy. Energy consumed by the read circuits and the CMOS logic gates is extremely small in comparison to these two energies. The energy cost of RT-based SIMON circuit is significantly larger than its CMOS implementation. For example, CMOS implementation of SIMON 64/96 consumes 255 pJ and has a delay of 2.18 ns per round [60]. It can be seen that the energy consumption of CMOS-based SIMON 64/96 is less than that of RT-based SIMON 32/64 even though SIMON 64/96 involves more number of rounds (42) and larger key-size (32-bit) and block-size (64-bit) than SIMON 32/64. This test case shows that we should explore ways in which spin-based circuits can be optimized for enabling lightweight cryptography.

3.3 Energy Optimization

Since a major portion of the total energy consumed by the above circuit comes from the write operations, more energy efficient implementation of SIMON 32/64 can be realized by reducing the number of write operations. As shown in Fig. 3.4, each 2-input logic unit in the encryption and the key expansion modules consist of a write circuit that writes the output of the logic operation into the write head of an RT. Reducing the number of logic

units in the circuit will directly reduce the number of write operations. The number of logic units required to implement a Boolean expression can be reduced by performing larger logic per unit. Such logic units have more than 2 inputs and are called composite gates. Only one write circuit will be needed if the entire Boolean expression is evaluated in a single logic unit.

FIGURE 3.7: STT-MTJ composite gate for encryption.

To verify this idea, we implemented SIMON 32/64 using composite gates.

Fig. 3.7 shows the implementation of Eq. 3.1 as a network of MTJs connected together. This network is evaluated by an SA and the output is stored in an RT. The number of copies stored in the RT depends on the fan-out of the composite gate. Similarly, the key expansion logic in Eq. 3.3 can be rewritten in the form of Eq. 3.4 and implemented as a composite gate. Details of the new circuit are discussed next.

$$k_{i+m} = S^{-3}k_{i+3} \oplus k_{i+1} \oplus k_i \quad (3.4)$$

$$\oplus S^{-4}k_{i+3} \oplus S^{-1}k_{i+1} \oplus C_i$$

3.3.1 Hardware Stages

Like the previous implementation, this circuit also comprises of successive stages. But, the key expansion logic and the encryption logic are now performed in single logic units, thereby, reducing the number of stages. This circuit comprises of 3 stages only. As a result, the circular shifter for propagating the on-state from one stage to another uses a shorter RT. The circular shifter is implemented as the one shown in Fig. 3.3.

3.3.2 Round Counter and Control Signals

The round counter shown in Fig. 3.2(b) is used in this implementation as well. The control signals used in this implementation are similar to those in Fig. 3.3.

3.3.3 Key Expansion

The bits of C_i are stored in RTs in the same way as in the previous implementation. During the on-state of stage 1 of this module, the expression in Eq. 3.4 is evaluated and the output bits are written into the write-head of RTs. When the subsequent stages are in on-state, current pulses are injected

into the RTs, to replicate the bit in the write-heads for fan-out. This module runs upto the 28th round and produces a 16-bit key in every round. The module remains in off-state from rounds 29 to 32.

3.3.4 Encryption

The various stages of the encryption unit are shown in Fig. 3.8. When stage 1 is in on-state, the result of Eq. 3.1 is produced and the output bits are written into the write-head of RTs. Also, x_{i+1} (see Fig. 3.8) is read and written into the write-head of RTs, for use in the subsequent round. During the on-state of subsequent rounds, current pulses are applied to the RTs for DW motion. When stage 1 is in on-state in the 32nd round, the ciphertext bits - Cipher[31:0] - are produced and new plaintext - p_{new} [31:0] - is accepted for encryption.

FIGURE 3.8: SIMON encryption with composite gates.

TABLE 3.4: Energy performance of composite gate-based SIMON 32/64.

Component	Energy (nJ)	Reduction (%)
Writing	4.000	84.35
Domain Wall shifting	2.085	26.58
Reading + CMOS gates	0.042	45.45
Total Energy	6.127	78.48

3.3.5 Results

Table 3.4 shows the energy performance of the composite gate-based implementation of SIMON 32/64. $\approx 65\%$ of the computation energy is spent on write operations. DW shifting consumes $\approx 34\%$ of the total computation energy. The SAs and other CMOS circuitry consume only 0.685% of the total energy. The time required to encrypt a plaintext is 259.2ns.

Overall, the optimization achieves $4.65\times$ reduction in total energy and $2.66\times$ improvement in encryption delay. We also observed $1.71\times$ reduction in transistor count. But, we are not able to comment on the corresponding area comparison, as we did not carry out the layout implementation of these circuits.

3.4 Conclusion

In this chapter, we studied the performance of RM-based implementations of SIMON 32/64 block cipher. Our study shows that the use of composite gates can significantly reduce the number of energy-expensive write operations in Boolean logic circuits. Reduction in J_{th} and I_{th} ($= J_{th} \cdot Area$) for MTJ switching and DW propagation is necessary to further enhance the performance of spin-based SIMON 32/64. This can be achieved by using more efficient magnetization switching technology (e.g. SOT) and by scaling down the spin devices. The comparative improvements shown in this chapter will more or less remain intact as long as the choices of parameters are consistent in both the circuits. We are not sure but it may be that,

for very large number of inputs i.e. for very large number of MTJs, the correct detection of the composite-gate output can become more difficult or unreliable for the PCSA. The correctness of output-detection depends on the TMR of the MTJs and the sensitivity of the PCSA. However, with the advancement of device technology and circuit design, these factors are expected only to improve in future, thereby, mitigating this issue. This issue is only speculative and would require extensive exploration with a wide range of devices and PCSA designs to confirm. The large amount of simulation time and the availability of accurate models for only a limited number of device-types pose a significant limitation in this path. Also, we would like to emphasize again that the composite gates used for encryption and key generation have not-so-small fan-in. If a constituent composite-gate doesn't function correctly beyond a threshold fan-in, the overall circuits can still be designed using composite gates with smaller fan-in. This would fetch as much energy benefits as possible while ensuring the functionality of the spin circuit. Also, we would like to comment about the applicability of composite gates in general. Any Boolean logic can be implemented using composite gates. The number of inputs of the composite gate would, of course, depend on the size of the target Boolean expression. Larger the Boolean expression, larger the composite gate.

4

Spintronic Activation Unit for Classifying Linearly Inseparable Functions

1

4.1 Introduction

Millennia of biological evolution have engineered the human brain to be extremely efficient for performing tasks, like vision and cognition, that are critical to human survival. The efficiency exhibited by the human brain in performing these vital tasks is unmatched even by the most powerful of

¹The work stated in this chapter has been published in [61, 62] during the PhD.

the existing modern-day supercomputers. The human brain is estimated to encage a whopping 20 billion neurons and 140 trillion synapses [63]. In 2014, Japan's K Computer, the then 4th most powerful supercomputer in the world, took about 40 minutes to completely simulate one second's worth of neuronal activity of just 1% of the human brain [64]. The K Computer was then equipped with 705,024 SPARC64 processor cores and 1.4 million GB of RAM and consumed enough electricity to power 10,000 homes. On the other hand, the average power consumed by the brain of a typical adult human being is ≈ 12 Watts [65]. The low-power and the high-speed data processing potentials of human brain ignited research interest in brain-based computing models.

FIGURE 4.1: Artificial neuron model

Artificial Neural Network (ANN) is one such computing paradigm that was invented to reap the aforementioned benefits by emulating the computation and communication capabilities of the neurons and the synapses in a human brain. The basic processing unit of ANN is depicted in Fig. 4.1. Analogous to its biological counterpart, it consists of a synaptic weight, w_i , corresponding to the i^{th} input, x_i , of the unit. Its operation can be mathematically

FIGURE 4.2: Single-layered ANN

expressed as:

$$Y = f_a\left(\sum_{i=1}^n w_i x_i + b\right) \quad (4.1)$$

where, the bias, b , denotes a weight connected to an input of fixed value 1, f_a represents the activation function and Y is the output of the neuron. The ANN is a layered model with multiple neurons per layer. In its simplest form, the ANN consists of one neural layer only (see Fig. 4.2). In a multi-layered network, the output of one neuron is utilized as input to multiple neurons in the next layer. The synaptic weights determine the function computed by the ANN. For an ANN to be effectively able to classify input patterns, its weights are set to optimum values by iteratively updating the randomly initialized weights according to certain learning rules.

Although a variety of non-linear functions are available for use as activation function in neurons, the threshold or hard-limiting function is popular due to the ease of its implementation in CMOS [66] or spintronic hardware [67].

The behaviour of a neuron having threshold activation, also known as ‘perceptron’, is given by:

$$Y = \begin{cases} -1, & \text{if } \sum_{i=1}^n w_i x_i < -b \\ 1, & \text{otherwise} \end{cases} \quad (4.2)$$

FIGURE 4.3: 2-input OR

FIGURE 4.4: 2-input XOR

In spite of being hardware-friendly, threshold function suffers from limited functionality. All neurons using the threshold function as its activation can

learn a given Boolean function, $f(x_1, x_2, x_3, \dots)$, only if it is possible to determine an n -dimensional plane, $y = b + w_1x_1 + w_2x_2 + w_3x_3 + \dots + w_nx_n$ (where w_i is the i^{th} weight), that separates the 1s of this function from its -1 s (or, 0s). Such a function is said to be linearly separable. Boolean AND and OR are the most commonly seen examples of linearly separable functions. Their 1s can be separated from the -1 /0s by a straight line as shown in Fig. 4.3. A neuron with threshold activation function can realize this line by learning an appropriate set of weights. But, functions very often lack the property of linear separability. Such functions are known as non-linearly separable functions. The most common example of this class of functions is XOR. As can be seen in Fig. 4.4, the 1s of XOR cannot be separated from the -1 /0s by constructing a single straight line. Consequently, there exists no definite set of weights (i.e. a line) that can be learned by a neuron with threshold activation function for implementing XOR. This prevents the learning from converging. A neuron more capable of learning non-linearly separable functions can mitigate this challenge by enabling single layered neural network to have greater flexibility and greater generalization ability, thereby, bridging the performance gap between single layered and multi-layered NNs. At the same time, single layered NNs lead to far more lightweight (low-power cum low-area) neuromorphic implementations than deep neural networks.

4.2 Proposed Neurons

TABLE 4.1: Truth Table of Neuron with Eqn. 4.3 as Activation Unit

x_1	x_2	$x (= w_1x_1 + w_2x_2)$	$f(x)$
-1	-1	-0.7 ($x < t_1$)	-1
-1	1	-0.3 ($t_1 < x < t_2$)	1
1	-1	0.3 ($t_1 < x < t_2$)	1
1	1	0.7 ($x > t_2$)	-1

FIGURE 4.5: XOR using threshold neurons

((A))

((B))

FIGURE 4.6: (a) Dual-threshold function (b) XOR using neuron with dual-threshold AU

Being non-linearly separable, XOR cannot be learned by a neuron using threshold activation. A two-layered network (see Fig. 4.5) with three threshold neurons is necessary for implementing two-input XOR. In contrast, XOR can be realized using only one neuron with the activation function shown in Fig. 4.6((a)). The function has two thresholds, t_1 and t_2 ($> t_1$), and can be mathematically expressed as:

$$f(x) = \begin{cases} -1, & \text{if } x < t_1 \\ 1, & \text{if } t_1 \leq x < t_2 \\ -1, & \text{if } x \geq t_2 \end{cases} \quad (4.3)$$

where, x represents the input to the function. A two-input neuron having Eq. (4.3) as the activation function is shown in Fig. 4.6((b)).

Assuming an input weight vector $[w_1 \ w_2]^T = [0.5 \ 0.2]^T$ and a threshold vector $[t_1 \ t_2]^T = [0.0 \ 0.5]^T$, for instance, it can be easily seen in Table 4.1 that the resultant behavior of Eqn. 4.3 matches the XOR functionality. Hence, it can be inferred that *having additional non-linearity in the activation function can make a neuron computationally more powerful and achieve greater functionality*. This forms the basis of our idea of the proposed activation units (AUs). We propose to achieve this by realizing Eqn. 4.3 using DW motion.

4.2.1 1st Proposed Neuron (PN-1): Single Tunable-Threshold

4.2.1.1 Proposed Activation Unit

The proposed DW motion-based AU [61] comprises of two equally-sized ferromagnetic strips, S_1 and S_2 , connected as shown in Fig. 4.7. A neuron comprising of this AU has its bias, b , connected to the path $w_{1r} - w_{2l}$. The individual strips have their ends pinned along opposite magnetization directions. In each strip, the two oppositely-magnetized ends lead to the nucleation of a DW in the region between them. The magnetization at any point in this region can be switched by shifting the DW back and forth via STT. We call this region as ‘domain-wall-motion (DWM)’ region. DW pinning can occur only at the ends of this region. A Magnetic Tunnel Junction (MTJ) sensor is fabricated on top of this region. The MTJ can either have the

FIGURE 4.7: Proposed DW motion-based neural AU for non-linearly separable functions.

DWM region as its free layer or have its free layer electrically insulated from but magnetically coupled with the DWM region. Unlike the former structure, the latter offers the key benefit of decoupled read- and write-current paths. So, a thin layer of magnetic oxide [68] is sandwiched between the DWM region and the MTJ free-layer in Fig. 4.7. This oxide layer serves to insulate the read-current path from the write-current path and provides magnetic coupling between the MTJ free-layer and the DWM region. The magnetization of the MTJ free-layer, thus, locally follows the magnetization of the DWM region directly below it and different DW positions result in different resistances of the MTJ. The MTJs, here, are connected in series. The magnetization directions of the pinned domains of the strips and their MTJs are as shown in Fig. 4.7.

The three-staged operation of this neuron begins with the DWs positioned at the right end of the free regions of S_1 and S_2 . In the write stage, currents I_w ($\propto \sum_{i=1}^n w_i x_i$) and $I_w + I_b$ ($\propto \sum_{i=1}^n w_i x_i + b$) flow through S_1 and S_2 , respectively. Assuming I_{th} to be the threshold current for DW motion in S_1 and S_2 , the DW in S_1 moves leftward if $I_w > I_{th}$; otherwise, it remains pinned. Similarly, the occurrence of DW motion in S_2 is determined by

$I_w + I_b$. The sum of the final resistances of the MTJs is read in the read stage. A CMOS-based amplifier, e.g., Pre-Charged Sense Amplifier (PCSA) [56], can be used to read this sum. The PCSA compares the net resistance with a reference resistance, R_{ref} , by discharging pre-charged polarization voltage through them. The binary result of this comparison gives the neuron output, Y . In the reset stage, a reset current, I_{reset} , is passed through each strip to shift its DW back to the right end of the DWM region. This initializes the neuron for its operation on the next input. It is worthwhile to mention that *the proposed neuron, PN-1, achieves additional non-linearity using the same number of synaptic resources as a perceptron.*

4.2.1.2 Proposed Learning Algorithm (LA-1)

FIGURE 4.8: Modeling the dual-threshold function.

Due to the absence of any explicit learning algorithm for a neuron with this AU, we devise our own algorithm here. In this proposition, we model

the XOR-like function, f , in Eqn. 4.3 as the binary product of two single-threshold functions, f_1 and f_2 . Mathematically,

$$f = f_1 \cdot f_2 \quad (4.4a)$$

$$f_1 = \begin{cases} -1, & \text{if } x < t_1 \\ 1, & \text{otherwise} \end{cases} \quad (4.4b)$$

$$f_2 = \begin{cases} 1, & \text{if } x < t_2 \text{ } (t_2 > t_1) \\ -1, & \text{otherwise} \end{cases} \quad (4.4c)$$

Fig. 4.8 clearly portrays the proposed model. Adjusting any of the constituents, f_1 or f_2 , would produce corresponding changes in the resultant function, f . The idea of decomposing f into f_1 and f_2 derives its reasoning from the observation that S_1 and S_2 in Fig. 4.7 can respectively realize f_1 and f_2 .

For this AU, we should note that $t_1 = 0$ because I_w flowing through S_1 is always proportional to $\sum_{i=1}^n w_i x_i + 0$ and $t_2 = -b$ since $I_w + I_b$ through S_2 is always proportional to $\sum_{i=1}^n w_i x_i + b$.

FIGURE 4.9: Training of the proposed activation unit.

Algorithm 1 Proposed Learning Algorithm (Stochastic)

Input: current weight vector, $\vec{W} = [w_1 \ w_2 \ w_3 \dots w_n]^T$, and bias, b ;
input vector, $\vec{X} = [x_1 \ x_2 \ x_3 \dots x_n]^T$, and desired output, y_d ;
learning rate, η ; loop size for ‘test-and-halve’ operation, l_s

1. **if** $0 \leq \sum_{i=1}^n w_i x_i < -b$ **then**
 $y \leftarrow 1$ **Comment:** y is the actual output of neuron
else
 $y \leftarrow -1$
end if
2. $\delta \leftarrow \eta(y_d - y)$
3. $d_1 \leftarrow |0 - \sum_{i=1}^n w_i x_i|$, **Comment:** distance of $\vec{W} \cdot \vec{X}$
 $d_2 \leftarrow |(-b) - \sum_{i=1}^n w_i x_i|$ **Comment:** from t_1 and t_2
4. **if** $d_1 < d_2$ **then**
 $\delta_b \leftarrow 0$,
 $\vec{\Delta}_w \leftarrow \delta \vec{X}$
else if $d_1 > d_2$ **then**
 $\delta_b \leftarrow -\delta$,
 $\vec{\Delta}_w \leftarrow -\delta \vec{X}$
end if
5. **for** $i \leftarrow 1, l_s$ **do** **Comment:** test-and-halve
if $-(b + \delta_b) < 0$ **then**
 $\delta_b \leftarrow \frac{\delta_b}{2.0}$
end if
end for
6. $b \leftarrow b + \delta_b$,
 $\vec{W} \leftarrow \vec{W} + \vec{\Delta}_w$

Output: updated $\vec{W}$ and b for training on next input

The stochastic version of LA-1 is outlined in Algorithm 1. During training, the weights and the bias of the neuron are updated by employing perceptron learning rule according to either f_1 or f_2 . If the weighted sum, $\sum_{i=1}^n w_i x_i$, for a given training sample lies closer to t_1 , the perceptron learning rule is applied as per f_1 . Only the weights, and not the bias, are updated in this case, because the bias, b (I_b), doesn't contribute to the input to f_1 (S_1). On the other hand, if $\sum_{i=1}^n w_i x_i$ lies closer to t_2 , the weights and the bias are updated as per f_2 . Prior to adding an update to the bias, the algorithm carries out a test to check whether the condition $t_2 > t_1$ (Eqn. 4.3) would hold good for the resultant bias. If the resultant bias violates this condition, the update is

halved. As a precaution, we perform the ‘test-and-halve’ process twice for each training sample and it works fine for the real-world classification tasks discussed later in this paper. The behavior of PN-1 during the learning process can be qualitatively understood by looking at Fig. 4.9. Therefore, by programming b , this algorithm is able to learn the value of $t_2 - t_1$ that is optimum for a dataset.

4.2.2 2^{nd} Proposed Neuron (PN-2): Dual Tunable-Threshold

4.2.2.1 Proposed Neural Activation Unit

FIGURE 4.10: Proposed domain wall motion-based AU with two programmable thresholds.

The AU of PN-2 [62] comprises of two equally-sized ferromagnetic strips, S_1 and S_2 , connected as shown in Fig. 4.10. Besides having a traditional bias, b_1 , connected to the w_{1l} terminal of S_1 , it has an additional bias, b_2 , connected to the path $w_{1r} - w_{2l}$ between S_1 and S_2 . The rationale behind this design choice is justified in the following sub-sub-section. The individual strips are identical to the structure in Fig. 4.7. The magnetization direction of the pinned domains in these strips and their MTJs can be known from Fig. 4.10. As in the basic design, the MTJs, here, are serially connected.

The three-staged operation of a neuron with this AU also begins with the DWs positioned at the right end of the free regions of S_1 and S_2 . In the write stage, currents $I_w + I_{b1}$ ($\propto \sum_{i=1}^n w_i x_i + b_1$) and $I_w + I_{b1} + I_{b2}$ ($\propto \sum_{i=1}^n w_i x_i + b_1 + b_2$) flow through S_1 and S_2 , respectively. Assuming I_{th} to be the threshold current for DW motion in S_1 and S_2 , the DW in S_1 moves towards its left if $I_w + I_{b1} > I_{th}$; otherwise, it remains pinned. Similarly, the occurrence of DW motion in S_2 is determined by $I_w + I_{b1} + I_{b2}$. The read and reset stages of its operation are identical to those of the basic design.

4.2.2.2 Proposed Learning Algorithm (LA-2)

Here, we describe the algorithm that we proposed for this neuron. This algorithm is also based on the set of Eqs. (4.4a), (4.4b), (4.4c).

For PN-2, we should note that:

- (a) $t_1 = -b_1$, because the write current flowing through S_1 is always proportional to $\sum_{i=1}^n w_i x_i + b_1$.
- (b) $t_2 = -(b_1 + b_2)$, since the write current through S_2 is always proportional to $\sum_{i=1}^n w_i x_i + b_1 + b_2$.
- (c) $m = -(b_1 + b_2/2)$ such that, m denotes the mid-point between t_1 and t_2 in Fig. 4.8.

It is evident from these expressions that both t_1 and t_2 are programmable by tuning b_1 and b_2 . The stochastic version of LA-2 is outlined in Algorithm 2. During this stochastic training, the updates for the weights and the biases of a neuron are generated by employing perceptron learning rule according to either f_1 or f_2 . If the weighted sum, $\sum_{i=1}^n w_i x_i$, for a given training sample lies closer to t_2 than to t_1 , the weights and the bias b_2 are updated by applying the perceptron learning rule as per f_2 in Eq. (4.4c). No update is generated for b_1 in this case. On the other hand, if the value of $\sum_{i=1}^n w_i x_i$ is closer to t_1 , the weights and the bias b_1 are updated according to f_1 in Eq. (4.4b). Change in t_1 due to the addition of an update to b_1 also causes an equal change in the value of t_2 . If the update increases (decreases) t_1 , t_2

Algorithm 2 Proposed Learning Algorithm (Stochastic)

Input: current weight vector, $\vec{W} = [w_1 \ w_2 \ w_3 \dots w_n]^T$, and biases, b_1 and b_2 ; input vector, $\vec{X} = [x_1 \ x_2 \ x_3 \dots x_n]^T$, and desired output, y_d ; learning rate, η ; loop size for ‘test-and-halve’ operation, l_s

1. **if** $-b_1 \leq \sum_{i=1}^n w_i x_i < -(b_1 + b_2)$ **then**
 $y \leftarrow 1$ **Comment:** y is the actual output of neuron
else
 $y \leftarrow -1$
end if
2. $\delta \leftarrow \eta(y_d - y)$
3. $d_1 \leftarrow |(-b_1) - \sum_{i=1}^n w_i x_i|$, **Comment:** distance of $\vec{W} \cdot \vec{X}$
 $d_2 \leftarrow |(-(b_1 + b_2)) - \sum_{i=1}^n w_i x_i|$ **Comment:** from t_1 and t_2
4. **if** $d_1 < d_2$ **then**
 $\delta_{b1} \leftarrow \delta$,
 $\delta_{b2} \leftarrow -\delta_{b1}$,
 $\vec{\Delta}_w \leftarrow \delta \vec{X}$
else if $d_1 > d_2$ **then**
 $\delta_{b1} \leftarrow 0$,
 $\delta_{b2} \leftarrow -\delta$,
 $\vec{\Delta}_w \leftarrow -\delta \vec{X}$
end if
5. **for** $i \leftarrow 1, l_s$ **do** **Comment:** test-and-halve
if $-(b_1 + \delta_{b1}) > -(b_1 + \delta_{b1} + b_2 + \delta_{b2})$ **then**
 $\delta_{b2} \leftarrow \frac{\delta_{b2}}{2.0}$
end if
end for
6. $b_1 \leftarrow b_1 + \delta_{b1}$,
 $b_2 \leftarrow b_2 + \delta_{b2}$,
 $\vec{W} \leftarrow \vec{W} + \vec{\Delta}_w$

Output: updated b_1 , b_2 and $\vec{W}$ for training on next input

increases (decreases) equally. To negate this effect, we add a compensating update of equal magnitude but opposite sign to the value of b_2 .

Notice that, in both the above cases, an update is generated for b_2 . In either case, prior to adding the update to b_2 , the algorithm carries out a test to check whether the condition $t_1 < t_2$ (Eq. (4.3)) would remain valid for the resultant values of b_1 and b_2 . If the resultant biases lead to $t_1 \not< t_2$, the update for b_2 is halved. We call this operation the ‘test-and-halve’ step.

Next, the final values of updates are added to the corresponding weights and biases.

FIGURE 4.11: Behavior of PN-2 during training.

Algorithm 3 presents the batch version of the LA-2. It differs from the stochastic form in the test-and-halve and the weight-, bias-update steps. We introduce variables $\vec{E}_w$, e_{b1} and e_{b2} to store the cumulative errors (updates) for the weights and the biases during training in a batch. These variables are initialized with 0s at the onset of every batch. The computation of errors ($\vec{\Delta}_w$, δ_{b1} and δ_{b2}) at each training sample in a batch is done as in Algorithm 2. In the test-and-halve step, to evaluate whether $t_1 < t_2$ will continue to hold good, we compare the partial values of thresholds that would result if b_1 and b_2 are respectively updated with the sums $e_{b1} + \delta_{b1}$ and $e_{b2} + \delta_{b2}$. The resultant errors are then accumulated in $\vec{E}_w$, e_{b1} and e_{b2} . At the end of the batch, the final values of $\vec{E}_w$, e_{b1} and e_{b2} are, in turn, added to the individual weights and biases.

In both the stochastic and the batch versions of the LA-2, as a precautionary measure, we choose to perform the test-and-halve step twice (i.e., $l_s = 2$) for each training sample and it works fine for the real-world classification tasks discussed later in this paper. Fig. 4.11 provides a glimpse of the training

Algorithm 3 Proposed Learning Algorithm (Batch)

Input: current weight vector, $\vec{W} = [w_1 \ w_2 \ w_3 \dots w_n]^T$, and biases, b_1 and b_2 ;
current sum-of-errors for b_1 , b_2 and $\vec{W}$: e_{b1} , e_{b2} and $\vec{E}_w$;
input vector, $\vec{X} = [x_1 \ x_2 \ x_3 \dots x_n]^T$, and desired output, y_d ;
learning rate, η ; loop size for ‘test-and-halve’ operation, l_s

1. **do**
 steps 1 to 4 of Algorithm 2
2. **for** $i \leftarrow 1, l_s$ **do** **Comment:** test-and-halve
 if $-(b_1 + e_{b1} + \delta_{b1}) > -(b_1 + e_{b1} + \delta_{b1} + b_2 + e_{b2} + \delta_{b2})$ **then**
 $\delta_{b2} \leftarrow \frac{\delta_{b2}}{2.0}$
 end if
 end for
3. $e_{b1} \leftarrow e_{b1} + \delta_{b1}$,
 $e_{b2} \leftarrow e_{b2} + \delta_{b2}$,
 $\vec{E}_w \leftarrow \vec{E}_w + \vec{\Delta}_w$
4. **if** $end - of - batch$ **then**
 $b_1 \leftarrow b_1 + e_{b1}$,
 $b_2 \leftarrow b_2 + e_{b2}$,
 $\vec{W} \leftarrow \vec{W} + \vec{E}_w$
 $e_{b1} = 0$,
 $e_{b2} = 0$,
 $\vec{E}_w = [0 \ 0 \ 0 \ \dots \ 0]^T$
 end if

Output: updated b_1 , b_2 and $\vec{W}$ for training on next batch of input

carried out by LA-2. Thus, the proposed algorithm materializes the extra flexibility of this AU by intelligently tuning both its thresholds.

4.3 Neural-Network Training and Neuromorphic-Circuit Simulation

To analyze the performance of the proposed learning algorithms, we carry out offline training on some popular datasets. The networks trained are single layered and contain as many neurons as the number of classes in the datasets. Prior to the training, we perform mean subtraction followed by min-max normalization (between -1 and 1) of all the samples in a dataset.

TABLE 4.2: Split-up of Datasets for training, validation and testing

Dataset	Total size	Training size	Validation size	Test size
Iris	150	120	15	15
MONK-2	601	483	59	59
User Knowledge Modeling	403	323	40	40
Wall-Following Robot Navigation (sensor-readings-2)	5456	4370	543	543

These standard pre-processing steps are done to aid the fast convergence of training. We choose to employ 10-fold cross-validation in our training as it is very commonly used in applied machine learning [69]. We run 100 trials of 10-fold training, validation and testing. Each trial comprises of 10 folds and in each fold, we use 10% of the dataset for testing, another 10% for validation and the remaining 80% for training. The size of the training, validation and test sets in each fold are given in Table 4.2. The training, validation and test sets vary from one fold to another. In other words, no two folds have the same testing or validation or training sets. Each fold consists of the following steps in order:

- (a) We randomly initialize the weights and the biases of the networks in the ranges $[0.0, 1.0)$ and $[-1.0, 0.0)$, respectively. The bias b_1 of the PN-2 is initialized to zero, whereas b of the PN-1 and b_2 of the PN-2 are randomly initialized such that $t_2 > t_1$ (Eqs. (4.3), (4.4c)).
- (b) The initial sets of weights and biases are then multiplied with suitable factors for achieving better training results. The value of the multiplication factor is selected by trial and error and is different for different datasets.
- (c) If, corresponding to the initial weights of any neuron, $\sum_{i=1}^n w_i x_i$ for all training samples lie closer to either t_1 *only* or t_2 *only*, we assign m (mid-point) to the initial value of the programmable threshold lying isolated.

This step furnishes a more-balanced training by shifting the initially-isolated programmable threshold closer to the other one. Otherwise, it is observed that the learning – in most cases, throughout all the training epochs – takes place according to f_1 *only* or f_2 *only*. Consequently, the proposed neuron appears to have only one or no tunable-threshold and the other threshold remains unused throughout the training.

- (d) Next, the weights and biases are trained by executing the proposed algorithm iteratively for a pre-set number of epochs. At the end of each epoch, we evaluate the network’s MCR on the validation set. The training in a fold is registered only up to the epoch in which the lowest validation MCR – such that the MCR value remains conserved over a minimum (pre-defined) number of subsequent epochs – is achieved.
- (e) The performance of the trained network is assessed on the test set of the ongoing fold.

We conduct two sets of the above 100 10-fold trials – one using the stochastic version and another using the batch version of the algorithms. During the batch training, we use a batch size equal to the size of the training set. The set of weights and biases corresponding to the least test-MCR obtained in any fold of the 100 10-fold trials of the batch and the stochastic trainings is used for realizing an MCA-based neuromorphic implementation of single layered network (SLN). Whereas the values of these weights and biases can be either positive or negative, the physical conductance of a memristor in an MCA is solely positive in nature. To allow the mapping of negative weights, we represent the i^{th} weight of the j^{th} neuron in the SLN as the difference between two memristive conductances, G_{ij}^+ and G_{ij}^- , in the MCA. In this scheme, positive and negative weights are obtained by having $G_{ij}^+ > G_{ij}^-$ and $G_{ij}^+ < G_{ij}^-$, respectively. In our simulation, we utilize memristors with conductances in the range $[15\mu S, 150\mu S]$. The choice of using low synaptic conductances (i.e., high synaptic resistances) ensures that the voltage drops occurring across the synapses are significantly larger than the voltage drop across the AU, thereby, alleviating any non-linearity introduced into the

circuit due to the latter. We implement a high-precision write algorithm [70] to tune a synaptic memristor within 2% of the desired conductance level. This write process employs a tuning voltage, V_{tune} , of initial value 0.5 V and increasing/decreasing in steps of 10 mV until the desired precision is reached. We apply this algorithm on a dynamic switching model [71] of memristor to simulate the *ex-situ* [72] programming of the memristors in the MCA. In order to protect the memristors from breaking down, we restrict V_{tune} up to a maximum and a minimum of 4.8 V and -2.3 V, respectively.

FIGURE 4.12: Neuromorphic architecture of single-layered network of the proposed neuron. The reading circuitry is shown for one neuron only. For illustration purpose, the neurons are shown to be of the 2nd proposed-type.

The neuromorphic implementation of the SLN consisting of PN-2s is illustrated in Fig. 4.12. In this circuit, the difference between G_{ij}^+ and G_{ij}^- is achieved by connecting G_{ij}^+ to the input voltage, $V_{inp(i)}$, and G_{ij}^- to $-V_{inp(i)}$. Here, $V_{inp(i)}$ lies in the range $[-40\text{mV}, 40\text{mV}]$. We use low-voltage inputs to enable: i) non-destructive reads on memristors and ii) low-power operation of the overall network. The inputs to the MCA are 3-bit wide. As can be observed in Fig. 4.13, 3-bit input precision offers classification accuracy close to that achieved using full-precision inputs. The accuracy corresponding to 3-bit inputs deviates most in case of the 4th dataset; but, interestingly, it leads to $\approx 10.5\%$ increase in accuracy – which is desirable. The maximum

FIGURE 4.13: Effect of reduced input precision on network performance. The network here is of PN-2s.

drop in accuracy noted for any dataset is $\approx 1.2\%$ (dataset 1). Increasing the number of input bits can reduce the loss in accuracy, but it also leads to larger power consumption. Due to binary-weighted current sourcing, the input current grows exponentially with the increase in the number of input bits. Altogether, 3-bit precision presents a fair trade-off between classification performance and energy efficiency of the circuit. This holds good for neurons with PN-1 as well.

Let R_{s1} and R_{s2} respectively denote the resistances of S_1 and S_2 in PN-2. The currents supplied to these magnetic strips during the write phase of this

neuron are given by:

$$I_{s1} = \frac{\left(\sum_{i=1}^n (G_i^+ - G_i^-) V_{inp(i)} + (G_{b1}^+ - G_{b1}^-) V_{inp(b1)} + I_{offset} \right) - (G_{b2}^+ - G_{b2}^-) V_{inp(b2)} \left(\frac{R_{s2} G_{sum}}{1 + R_{s2} (G_{b2}^+ + G_{b2}^-)} \right)}{1 + G_{sum} \left(R_{s1} + \frac{R_{s2}}{1 + R_{s2} (G_{b2}^+ + G_{b2}^-)} \right)} \quad (4.5a)$$

$$I_{s2} = \frac{I_{s1} + (G_{b2}^+ - G_{b2}^-) V_{inp(b2)}}{1 + R_{s2} (G_{b2}^+ + G_{b2}^-)} \quad (4.5b)$$

where,

$$G_{sum} = \sum_{i=1}^n G_i^+ + \sum_{i=1}^n G_i^- + G_{b1}^+ + G_{b1}^- \quad (4.5c)$$

I_{offset} in Eq. (4.5a) is an additional current that is injected into its AU to balance the threshold current for DW motion in the magnetic strips. It is worth mentioning that, for very small values of R_{s1} and R_{s2} , the voltage drops across S_1 and S_2 become negligible and, therefore, I_{s1} and I_{s2} represent the weighted sums of the input voltages. The effect of increasing R_{s1} and R_{s2} on the accuracy of the neuromorphic network is discussed in the next section. The net synaptic currents injected into the strips of the PN-1 can be obtained by setting G_{b1}^+ and G_{b1}^- to zero in Eqs. (4.5a)–(4.5c), since b_1 is not present in this neuron structure.

We employ equations that capture the dynamics of current-induced DW depinning and STT-driven DW motion in the mCell [59] model for realizing the write operation of the neurons. These equations are:

$$t_{depin} = 4523 |J|^{-2.82} + 0.2285 \quad (4.6)$$

$$v_{DW} = \left(\frac{1 + \alpha\beta}{1 + \alpha^2} \right) \frac{g\mu_B P J}{2eM_s} \quad (4.7)$$

where, t_{depin} signifies the time (ns) required to depin the DW, J the current

TABLE 4.3: Physical Parameters of the Magnetic Strips used in the Simulation of a Domain Wall Motion-based AU

Parameters	Values
Length	20 nm
Width	10 nm [59]
Thickness	3 nm [73]
Resistivity	200 Ω .nm [59]
Length of MTJ	12 nm [74]
TMR of MTJ	150 % [75, 76]
RA (low) of MTJ	$1.8 \times 10^7 \Omega$.nm ² [76]

density (MA/cm²) applied along the direction of the intended DW motion, v_{DW} is the DW velocity (nm/s), α the Gilbert damping constant, β the nonadiabatic STT coefficient, g the Land factor, μ_B the Bohr magneton, P the spin polarization, J the electron current density, e the electron charge, and M_s the saturation magnetization. The device parameters used in our simulation are specified in Table 4.3. We utilize strips with low width in order to minimize the current required for DW motion. We use an I_{offset} of value 12 μ A to achieve nanosecond-long write operation. Note that the write, read and reset operations of the neuron are each set to have a duration of 1 ns [77]. A CMOS-based Pre-Charged Sense Amplifier (PCSA) is implemented for reading the output of the neuron. The pre-charge phase of the PCSA is overlapped with the write stage of the neuron in order to reduce the overall delay of the neuron. We utilize an I_{reset} of magnitude 14.9 μ A for performing the reset operation.

A crucial implementation concern that needs consideration is the effect of device-mismatch, noise and process variation upon the network performance. Random edge-roughness in the ferromagnetic strip and thermal fluctuations can give rise to variation in the programming of DW position in the AU. However, it is to be noted that multiple research works [78], [79], [80] have successfully demonstrated the occurrence of deterministic DW motion in nano-magnetic structures. In addition, fabricating notches along the edges

of the strip can improve programming accuracy by stabilizing and pinning the DW at desired locations [81]. It is worth mentioning that brain-inspired computing paradigms like ANNs are inherently resilient to inaccuracies in the constituent computations [82]. Hence, it is reasonable to expect that minor mismatches or variations in the memristive weights and the DW AU will not degrade the network performance significantly.

4.4 Results and Analysis

Let us now look into the performances of the proposed neurons and their learning algorithms in detail. The datasets used for the evaluation of the corresponding neurons and their LAs are taken from the UCI machine learning repository [83] and are presented in Figs. 4.4, 4.4, 4.4, 4.17 for visualization. Dimensionality-reduction techniques such as, Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA) are used for plotting multi-class and binary-class datasets, respectively, here. To ensure a fair comparison among the perceptron (baseline) and the proposed neurons, we adopt the same methodology of training, testing and hardware implementation for each of these.

FIGURE 4.14: Iris dataset

4.4.1 Classification Performance

4.4.1.1 Learning Algorithm for Neurons with Dual-threshold AUs

Table 4.4 highlights the test-MCR results averaged over 100 trials of 10-fold training and testing on these real-world datasets. The average-MCR results for the batch as well as the stochastic versions of LA-1 and LA-2 are compared with those of the perceptron LA. We can observe that *LA-1 and LA-2 can achieve better accuracy than the traditional perceptron LA*. However, LA-1 fails to surpass the performance of the perceptron learning

FIGURE 4.15: MONK-2 (test+train) dataset

FIGURE 4.16: User Knowledge Modeling dataset

TABLE 4.4: LA-2 vs LA-1 vs Perceptron LA

Dataset	Average MCR					
	Perceptron LA		LA-2		LA-1	
	stoch.	batch	stoch.	batch	stoch.	batch
Iris	0.386	0.386	0.07	0.059	0.224	0.212
MONK-2 (test + train)	0.637	0.637	0.612	0.611	0.720	0.723
User Know- ledge Modeling	0.724	0.724	0.505	0.498	0.558	0.567
Wall-Following Robot Navigation (sensor-readings-2)	0.598	0.601	0.529	0.531	0.555	0.545

algorithm in the MONK-2 dataset. Although both the proposed neurons allow the separation between t_1 and t_2 to be modulated during training, *LA-2 distinctly outperforms LA-1 in all the datasets*. It makes sense because PN-1 suffers from the limitation of having its t_1 pinned at 0 – which may not be an optimal setting for all application-datasets. This drawback is not present in the design of the more-flexible 2nd AU. In fact, the stochastic version of LA-2 achieves $1.03\times$ – $3.59\times$ lower MCR than that of LA-1 and $1.04\times$ – $6.54\times$ lower MCR than the stochastic version of perceptron LA in these datasets. Further, we would like to highlight that *the batch version of LA-1 and LA-2 perform more-or-less similar to their stochastic versions*.

FIGURE 4.17: Wall-Following Robot Navigation Data (sensor-readings-2)

4.4.1.2 Neuromorphic Implementations

The classification performances of the neuromorphic circuits (with 3-bit inputs) implemented are reported in Table 4.5.

TABLE 4.5: Classification Performance of Neuromorphic Implementations

Dataset	MCR		
	SLN of Perceptrons (SLN- <i>per</i>)	SLN of PN-2s (SLN-2)	SLN of PN-1 (SLN-1)
Iris	0.0	0.0	0.066
MONK-2 (test + train)	0.288	0.288	0.389
User Knowledge Modeling	0.275	0.225	0.15
Wall-Following Robot Navigation (sensor-readings-2)	0.796	0.267	0.313

- (a) SLN-1: The neuromorphic implementation of SLN-1 outperforms that of SLN-*per* in two of the datasets, but loses in the Iris and the MONK-2 datasets. Our analysis shows that the MCR of ideal (software) SLN-1 deteriorates from 0.0 to 0.066 when the precision of Iris test-inputs is reduced from full-precision to 3-bit. The hardware implementation of SLN-1 for the Iris dataset then retains the MCR of 0.066. SLN-2, on the other hand, maintains its MCR (= 0.0) consistently in software (for both 3-bit and full-precision inputs) as well as hardware. In the MONK-2 dataset, ideal SLN-1 delivers an MCR of 0.356 at full input-precision and an MCR of 0.389 at 3-bit input-precision. The relatively poor performance of SLN-1 in the MONK-2 dataset may be attributed to the fixed value of t_1 in its constituent AUs.
- (b) SLN-2: Neuromorphic SLN-2 achieves better accuracy than neuromorphic SLN-1 in all datasets other than the User Knowledge Modeling dataset. We observe that the MCR of ideal SLN-2 in the User Knowledge Modeling dataset degrades from 0.099 at full input-precision to 0.125 at 3-bit input-precision. The MCR value further deteriorates to

0.225 in hardware due to the non-idealities introduced by the non-zero resistance of the AUs. In contrast, the neuromorphic implementation of SLN-1 for this dataset is able to retain the accuracy of the corresponding ideal network with full-precision inputs. Another notable observation for this dataset is that the neuromorphic implementation of SLN-2, however, outperforms that of SLN-1 when the MCR value averaged over the test sets of all the 10 folds is considered. These MCR values are 0.413 and 0.29 for SLN-1 and SLN-2, respectively. In hardware, SLN-2 performs at par with SLN-*per* in the Iris and the MONK-2 datasets, but achieves $1.22\times$ and $2.98\times$ higher accuracy than SLN-*per* in the subsequent datasets. Nevertheless, when we compare their MCR values averaged over the test sets of all the 10 folds, we find that SLN-2 outperforms SLN-*per* by a factor of 4.1 in the Iris dataset and a factor of 1.06 in the MONK-2 dataset, respectively.

FIGURE 4.18: Variation of network performance with the length of DW strip for Iris dataset

Figs. 4.4.1.2, 4.4.1.2, 4.4.1.2, 4.21 depict the classification performances of these SLNs for different lengths of DW strip(s) in the constituent AUs. *In general, classification accuracy drops as the length of DW strip increases.* This falling trend is often non-monotonous and characterized by regions of constant accuracy.

FIGURE 4.19: Variation of network performance with the length of DW strip for MONK-2 (test+train) dataset

FIGURE 4.20: Variation of network performance with the length of DW strip for User Knowledge Modeling dataset

4.4.2 Energy Performance of the Proposed Neurons

Next, we turn our attention to Table 4.6 that presents the average energy ($=I^2Rt$) consumption – averaged over all possible combinations of 3-bit inputs – of the aforementioned neurons. *The average energy consumed by PN-1*

FIGURE 4.21: Variation of network performance with the length of DW strip for Wall-Following Robot Navigation Data (sensor-readings-2)

TABLE 4.6: Energy Performance of Neuron Implementations

Dataset	Average Energy (fJ)		
	Perceptron	PN-2	PN-1
Iris	25.468	37.609	28.927
MONK-2 (test + train)	33.936	38.789	34.4
User Knowledge Modeling	28.285	36.056	27.216
Wall-Following Robot Navigation (sensor-readings-2)	22.937	28.063	20.196

is found to be quite close to that by the perceptron. In fact, it consumes somewhat less energy than the perceptron in the User Knowledge Modeling and the Wall-Following Robot Navigation datasets. We also observe that, for each of the considered datasets, the energy consumption of PN-2 is more than those of perceptron and PN-1. The reason behind this can be better understood by referring to the data given in Table 4.7. If we subtract the energy dissipated in the additional bias, b_2 , from the total energy consumption of PN-2, we find that the resultant value lies close to the energy footprints of perceptron and PN-1. This observation proves that b_2 accounts for the higher energy consumption of PN-2. The average energy (across all

TABLE 4.7: Energy Dissipated in b_2 of PN-2

Dataset	Average Energy (fJ)
Iris	10.541
MONK-2 (test + train)	11.716
User Knowledge Modeling	8.158
Wall-Following Robot Navigation (sensor-readings-2)	6.096

the datasets) consumed by the DW motion-based perceptron and PN-2 and PN-1 quantify to 27.66 fJ, 35.13 fJ and 27.68 fJ, respectively. Another key remark that we would like to add here is that, *irrespective of the type of AU, the write energy dissipated in the memristive synapses dominates the total energy consumption of a neuron*. For example, 66.85%–77.6% of the perceptron’s energy consumption originates from Joule heating in its synapses. Along similar lines, PN-2 and PN-1 respectively dissipate 72.8%–80.32% and 62.2%–77.81% of their total energy in the synapses.

4.5 Conclusion

As a whole, we have demonstrated two novel DWM-based dual-threshold AUs (Fig. 4.10) for learning non-linearly separable functions. Necessary algorithms for training the weights and biases of these neuros have also been developed. The proposed neurons and the associated learning algorithms have been extensively benchmarked on real-world datasets. One of these algorithms is shown to surpass the classification accuracy of the perceptron learning algorithm by a peak factor of 6.54. The single layered networks of the proposed neurons yield competent accuracies (on the considered datasets) in hardware. Moreover, the proposed neurons evaluate to be fairly energy-efficient – consuming energy in the lower *femto*-Joule range. Note that the energy consumption of the neurons can be brought down further by utilizing more-efficient DW-switching mechanisms such as Spin Orbit

Torque (SOT). Although the proposed AUs are shown to be based on STT-driven DW motion, they can also be realized with SOT phenomenon – by fabricating an additional heavy metal under-layer (beneath each of its DW strips) that can conduct charge current for exerting SOT on the DWs.

Based on the aforementioned advantages, it can reasonably be argued that the proposed neurons present a potential step towards boosting the performance of neuromorphic architectures meant for the resource-constrained hand-held devices and IoT platforms. Investigating the implications of using these neurons in multi-layered networks such as CNNs, DNNs and RNNs, that are widely used for accomplishing real-world tasks, marks the next logical goal in the roadmap of our research. Lastly, we would like to conclude this chapter with the hope that the current proof-of-concept demonstration will ignite interest in the spintronic community for pursuing further research into developing new spin-based AUs (as well as the corresponding learning algorithms) that can enable a neuron to learn and compute more complex non-linearly separable functions while consuming low power.

5

Efficient Mapping of XMG- and AIG-Synthesized Spintronic Circuits Using Domain Wall Motion-based XOR-Gate

¹The results presented in this chapter have been submitted in parts for publication in IEEE ISCAS [84] and IEEE TCAD [85]. The decisions by the reviewers are being awaited during the submission of this thesis.

5.1 Introduction

To address the immense complexity of today's ICs, the industrial design-process of ICs is highly automated. One core component of the electronic design automation (EDA) flow is logic synthesis. Logic synthesis converts a digital design given at the register-transfer level to a gate-level or transistor-level implementation, while optimizing one or more of the following: (i) number of nodes, (ii) number of logic levels, (iii) switching activity. The performance of digital circuits is primarily dependent on the efficiency of logic representation structures and associated Boolean-function-optimization algorithms [86] employed in the logic synthesis flow. Several data structures and algorithms have been proposed for this purpose [87–91]. Homogeneous logic structures such as, AND-Inverter Graphs (AIGs) [92, 93] and Majority-Inverter Graphs (MIGs) [94, 95] are more widely-used and are implemented in state-of-the-art logic synthesis tools. AIGs and MIGs consist of {2-input AND, Inverter} and {3-input Majority, Inverter} gates, respectively. Recently, it has been shown that, compared to AIGs, XOR-Majority Graphs (XMGs) can lead to greater reduction in the size and depth of a network [96]. XMG is made of {2-input XOR, 3-input Majority and Inverter} gates. The high expressive-power of XOR enables XMG synthesis to achieve smaller logic-representations and, therefore, less memory footprint. Since smaller networks require less time to be optimized, the exact synthesis executes faster with the use of XMGs. So, XMGs are suitable for optimization flows based on exact synthesis.

FIGURE 5.1: Spin-Memristor Threshold Logic (SMTL) gate.

FIGURE 5.2: XOR using SMTL gates.

Here, it becomes relevant to cite the work done by Deliang Fan *et. al.* in [97]. In this paper, the authors propose spin-memristor threshold logic (SMTL) gate and use it as the basis for performing threshold-logic synthesis of some popular benchmark circuits. Fig. 5.1 shows an SMTL gate consisting of an array of memristive weights for summation and a Domain Wall strip for thresholding operation. As shown in Fig. 5.1, the activation unit of an SMTL is a nano-strip containing one DW. The motion of a DW occurs only when the density of the current applied along the length of the strip is greater than the threshold current density. This thresholding property of DW motion realizes the step/threshold activation function. As discussed in Section 4.1, this activation function can classify linearly separable functions. So, SMTL gate can realize basic linearly-separable functions such as AND, OR, Inverter and Majority. But, it cannot realize the XOR primitive natively. As shown in Fig. 5.2, a 2-layered network of SMTL gates is required to implement XOR. As a result, SMTL-based mapping of synthesized XMGs is inefficient and sacrifices the original structure and compactness of the XMGs. In the absence of a native XOR-gate, the SMTL-mapped networks of XMG don't achieve the size, delay and energy performances that they ideally should. This is contrary to a synthesized AIG/MIG whose actual performance is unaffected by mapping, as its constituent gates are natively available. Consequently, any comparison of the performances of SMTL-mapped XMGs with those of SMTL-mapped MIGs/AIGs remains incomplete. A low-power

spin-device capable of realizing high-speed XOR operation is required to efficiently map synthesized XMGs to spintronic fabric and, thereby, allow an accurate evaluation of mapped XMGs with respect to mapped AIGs/MIGs. In this chapter, we attempt to fulfill the above gap in spintronic logic. The primary contributions of this work [85] are as follows:

- (a) Design of a novel Domain Wall (DW) motion-based 2-input XOR gate.
- (b) Circuit-level simulation and functional validation of the proposed XOR gate.
- (c) Comparative study of the performances of this and other DW motion-based gates.
- (d) Analysis of the impact of the proposed XOR-gate on the mapping of synthesized AIGs and XMGs – studied over popular benchmark-circuits.

5.2 Proposed XOR-gate

5.2.1 Domain-Wall Device

We introduce the structure shown in Fig. 5.3 to realize XOR operation. It consists of a ferromagnetic layer with five magnetization regions – r_1 to r_5 . The magnetizations of r_1 , r_3 and r_5 are pinned along fixed directions. Whereas r_1 and r_5 are magnetized along the same direction, r_3 is magnetized in the opposite direction. Due to the presence of oppositely magnetized regions at the ends of r_2 and r_4 , a DW is nucleated in each of these two regions. The magnetization of any point in r_2 and r_4 depends on the DW's position relative to the point and can be manipulated by moving the DW back or forth. Suppose, I_{th2} and I_{th4} represent the threshold currents for DW motion in r_2 and r_4 , respectively. I_{th4} is set to be larger than I_{th2} . This is realized by fabricating r_3 , r_4 and r_5 wider than r_1 and r_2 . Although having r_3 , r_4 and r_5 thicker ('thickness', here, refers to the vertical height

FIGURE 5.3: DW motion-based device for the proposed XOR-gate.

of the layer) than r_1 and r_2 can also yield $I_{th4} > I_{th2}$, we don't adopt this alternative in our design as it costs extra steps to pattern a ferromagnetic layer with different thickness. MTJs M_1 and M_2 are fabricated on top of regions r_2 and r_4 , respectively, in order to read their magnetic states. Instead of directly having r_2 and r_4 as the free layers of M_1 and M_2 , respectively, these MTJs have free layers that are electrically insulated from r_2 and r_4 . It allows the paths of read and write currents to be isolated from each other, thereby, preventing the read current from affecting a DW position. To realize this design choice, a thin layer of magnetic oxide is sandwiched between r_2 and the free layer of M_1 as well as between r_4 and the free layer of M_2 . This oxide layer not only electrically isolates r_2 and r_4 from the free layers of the corresponding MTJs, but also introduces magnetic coupling between them. Consequently, the magnetization of any point in a free layer locally follows that of the point in r_2 or r_4 located directly below it. Thus, the resistance of an MTJ depends on the position of the corresponding DW. It is to be noted that M_1 and M_2 are connected in parallel. The net parallel-resistance, R_{out} , represents the output state of this device.

5.2.2 XOR Operation

FIGURE 5.4: Circuit-level implementation of the proposed XOR-gate.

Fig. 5.4 illustrates the XOR-gate implemented using the above DW-device. Its logic operation occurs in three consecutive phases – reset, write and read. The corresponding timing diagram is shown in Fig. 5.5. In the reset stage, Clk_{reset} goes high and the nMOS transistor, n_r , drains a current, I_{reset} , from the DW-device. I_{reset} flows from r_5 to r_1 of the device and shifts the DWs to the right ends of r_2 and r_4 , irrespective of their initial positions. This sets

FIGURE 5.5: Timing diagram of the proposed XOR-gate.

the logic device ready for performing its operation on the new pair of inputs. Next, in the write phase, the Boolean inputs, a and b , are applied to the gate terminals of the nMOS transistors, n_a and n_b , respectively. The sum (I_w) of the resultant currents supplied by n_a and n_b is injected into the device from r_1 to r_5 . Different input combinations produce different values of I_w . If $I_w < I_{th2}$, neither of the DWs gets depinned. If $I_{th2} < I_w < I_{th4}$, the DW in r_2 gets depinned and moves left due to STT, whereas the position of DW in r_4 remains unaffected. On the other hand, if $I_w > I_{th4}$, both the DWs shift leftward. So, the value of R_{out} varies with the inputs given to the gate. Note that, unlike the conventional design of spin-CMOS logic-gates [77, 97], the proposed gate doesn't use an array of resistive weights to produce I_w . This eliminates the power dissipated by the weights as well as the issues due to device mismatch and process variation in these weights. Also, our design doesn't require an additional offset current, I_{offset} , to balance the threshold current for DW depinning. Lastly, in the read stage, R_{out} is sensed. As in Fig. 5.4, a Pre-Charged Sense Amplifier (PCSA) can be used for this purpose. The operation of the PCSA consists of two phases – pre-charging ($Clk_{read} = 0$) and evaluation ($Clk_{read} = 1$). During the reset and write stages of the XOR-gate, the PCSA remains in the pre-charging phase, with both its outputs, O and $\bar{O}$, at 0. As a result, O and $\bar{O}$ are unable to drive the input nMOS-transistors of subsequent gates. During the read stage, the PCSA enters into the evaluation phase and compares R_{out} with a reference resistance, R_{ref} . The pre-charged polarization-voltage is discharged through each of these resistances to perform this comparison. If R_{out} is smaller than

TABLE 5.1: Truth Table of the Proposed Gate

$\mathbf{a}$	$\mathbf{b}$	$\mathbf{R}_{out}$ (r_p (r_{ap}) and R_p (R_{ap}) : resistances of M_1 and M_2 in $\parallel$ (anti- $\parallel$) state, resp.)	$\mathbf{O}$
0	0	$\frac{r_p \cdot R_{ap}}{r_p + R_{ap}} (< R_{ref})$	0
0	1	$\frac{r_{ap} \cdot R_{ap}}{r_{ap} + R_{ap}} (> R_{ref})$	1
1	0	$\frac{r_{ap} \cdot R_{ap}}{r_{ap} + R_{ap}} (> R_{ref})$	1
1	1	$\frac{r_{ap} \cdot R_p}{r_{ap} + R_p} (< R_{ref})$	0

R_{ref} , the output, O ($\bar{O}$), of the PCSA evaluates to 0 (1); otherwise, 1 (0). By choosing the value of R_{ref} appropriately, the behaviour of this gate can be matched with XOR functionality. Table 5.1 expresses this point in more detail.

5.3 Device-to-System Simulation

In this section, we describe our methodology for simulating the performances of various DW motion-based logic-networks. Fig. 5.6 illustrates the bottom-to-top simulation framework. As can be seen, it consists of multiple hierarchy-levels. These consecutive levels are discussed as follows:

5.3.1 Device Level

For simulating DW motion in a ferromagnet, we employ the mCell [59] Verilog-A compact model. The physics of DW depinning and propagation in the proposed device are modeled by the following equations [59]:

$$t_{depin} = 4523|J|^{-2.82} + 0.2285 \quad (5.1)$$

$$v_{DW} = \left(\frac{1 + \alpha\beta}{1 + \alpha^2} \right) \frac{g\mu_B P J}{2eM_s} \quad (5.2)$$

FIGURE 5.6: Simulation framework for DW motion-based logic networks.

where, t_{depin} represents the time (ns) required to depin the DW, J the current density (MA/cm²) applied along the direction of the intended DW motion, v_{DW} is the DW velocity (nm/s), α the Gilbert damping constant, β the non-adiabatic STT coefficient, g the Land factor, μ_B the Bohr magneton, P the spin polarization, J the electron current density, e the electron charge, and M_s the saturation magnetization. The salient parameters used in the simulation of the device in Fig. 5.3 are listed in Table 5.2. Note that we utilize strips of low width in order to reduce the current required for DW motion.

TABLE 5.2: Physical Parameters used in the Simulation of the Device Proposed in Fig. 5.3

Parameter	Value
Length	40 nm
Width	20 nm (r_1, r_2) 50 nm (r_3, r_4, r_5)
Thickness	3 nm [73]
Resistivity	200 Ω .nm [59]
Length of MTJ	12 nm [74]
TMR of MTJ	150 % [75, 76]
RA (low) of MTJ	$1.8 \times 10^7 \Omega$.nm ² [76]

5.3.2 Gate Level

Next, we develop a standard cell-library containing the energy and delay statistics of different DW motion-based gates. Apart from the proposed XOR-gate, the other gates in this library include AND, Majority and Inverter. As will be found in the following sub-section, this library contains the constituent gates of the final networks obtained after mapping the AIG-, MIG- and XMG-synthesized networks of various benchmark-circuits. We simulate CMOS-spin hybrid circuits of these gates in Cadence Virtuoso Analog Design Environment using the ST Microelectronics 40nm Process Design Kit (PDK) and the above device model. Unlike XOR, the Inverter, AND and Majority gates here can be realized by a single-DW device. Fig. 5.7 shows such a device. We use the same values of physical parameters for implementing this device as those listed in Table 5.2. Note that the parameter(s) given in Table 5.2 for r_3 , r_4 and r_5 do not apply to this device. The reset, write and read circuits for this device are same as shown in Fig. 5.4. The number of input transistors in the write circuit for this device, however, is determined by the number of Boolean inputs of the corresponding gate. The reset, write and read operations of all the gates in this library are each set to have a duration of 1.5 ns. Transistor n_r supplying the I_{reset} in these gates is connected to a 60 mV voltage source and the input transistors have

a 70 mV voltage source at their source terminals. In the proposed gate, n_r is sized to a width of $0.46 \mu\text{m}$ whereas the input transistors n_a and n_b are each $0.39 \mu\text{m}$ wide. The values of I_{reset} and I_w ensure 1.5 ns-long reset and write operations. A CMOS-based Pre-Charged Sense Amplifier (PCSA) is implemented for reading the output state of the DW device. The pre-charge phase of the PCSA is overlapped with the reset and write stages in order to minimize the overall gate-delay. We use an R_{ref} of value $18.2 \text{ k}\Omega$ in the PCSA of the proposed gate.

5.3.3 Network Level

FIGURE 5.7: Domain wall motion-based device for realizing Inverter, AND and Majority functions.

In this level, AIG-based and MIG-/XMG-based networks are generated using state-of-the-art logic synthesis tools like, ABC [91] and Cirkit [98], respectively. The logic synthesis, irrespective of the data structure, begins by providing a benchmark logic-netlist in the Berkeley Logic Interchange Format (.blif) as input to these tools.

FIGURE 5.8: Synthesized Circuit

FIGURE 5.9: Mapped circuit with DW-based buffers.

5.3.3.1 AIG-based Synthesis

The AIG-based synthesis is performed by executing ABC scripts like, *strash*, *resyn2*. We execute multiple iterations of these scripts to obtain an AIG that is as optimized as practically possible. The final AIG is then *balanced* and, thereafter, *mapped* using a library (.genlib) of basic gates. After mapping, we formally verify all results using ABC's *cec* command.

5.3.3.2 MIG-based Synthesis

We use the Cirkit tool from EPFL to carry out MIG- and XMG-based synthesis. The Cirkit tool has ABC integrated into it. First of all, the input logic-netlist is synthesized into an optimized AIG as above. Next, we perform MIG-based synthesis by executing the *xmglut -k 4 -noxor* command on the final AIG. This is followed by writing the MIG-synthesized network into a verilog (.v) file. Using ABC, we *balance* and then *map* the final MIG-based network to Majority and Inverter gates.

5.3.3.3 XMG-based Synthesis

The XMG-based synthesis begins with synthesizing the given logic-circuit into an AIG, followed by applying the *xmglut -k 4* command on the AIG. The output of this command is an XMG. We then *balance* and *map* this XMG. The library used here for mapping consists of gates that are native to the network – XOR, Majority and Inverter.

FIGURE 5.10: Phase sequence in different gate-levels of the mapped circuit.

Post synthesis and mapping, an in-house tool developed by us takes the final network as input. It performs the following functions:

TABLE 5.3: Energy Consumption of Domain Wall Motion-based Logic Gates

Gate	Reset Energy (fJ) (% of total energy)	Write Energy (fJ) (% of total energy)	Read Energy (fJ) (% of total energy)
Inverter	2.911 (53.45%)	1.713 (31.34%)	0.822 (15.09%)
AND	3.028 (48.63%)	2.367 (38.02%)	0.831 (13.35%)
Majority	3.028 (41.33%)	3.414 (46.60%)	0.884 (12.07%)
XOR (proposed)	5.479 (49.57%)	4.561 (41.27%)	1.012 (9.16%)

- (a) First, it parses through the mapped network and inserts DW motion-based buffer(s) between gates that are directly connected to each other but do not lie in consecutive gate-levels. {INV-1a, MAJ-3a}, {INV-1c, XOR-3a} and {XOR-2b, MAJ-4a} in Fig. 5.8 are examples of such pairs of gates. To understand why a buffer has to be inserted between the gates in these pairs, we first need to refer to Fig. 5.10. We can observe here that the write phase of the gates in a level always coincides with the read phase of those in the previous level. This is so because the input signals to a gate are produced during the read phase of its input gates. But, in case of the gate pairs mentioned above, the write phase of their output gates don't coincide with the read phase of their input gates. For instance, when the write phase of MAJ-3a occurs, INV-1a goes through its reset phase. During the reset phase, the PCSA of INV-1a cannot write into MAJ-3a. So, the resultant write operation of MAJ-3a would be incorrect. To cover this gap, a DW motion-based buffer, BUF-2a, is inserted in level 2. The input of BUF-2a is connected to INV-1a while its output is connected to MAJ-3a. The buffer is implemented using the device in Fig. 5.7. The write phase of BUF-2a occurs simultaneously with the read phase of INV-1a and its read phase coincides with the write phase of MAJ-3a. Fig. 5.9 depicts the buffers inserted corresponding to these gate pairs. The buffer BUF-4a

TABLE 5.4: Figures-of-Merit of MIGs
Mapped to DW Motion-based Native Gates

Bench- mark Circuit.	No. of Inputs/ Outputs	Mapped MIG				
		Size	Depth	Size· Depth	Energy (pJ)	EDP (10^{-18} J-s)
prom1	9/40	58585	42	2460570	4893.7	322.98
prom2	9/21	26498	46	1218908	2428.48	174.85
apex4	9/18	24781	42	1040802	2092.38	138.10
ex1010	10/10	23017	43	989731	1969.16	132.92
test4	8/30	18179	33	599907	1203.47	63.18
exps	8/38	10629	34	361386	729.85	39.41
alu4	14/8	9635	57	549195	1076.31	95.25
bench1	8/10	7684	34	261256	529.76	28.61
cavlc	10/11	5778	30	173340	358.57	17.21
test1	8/10	5732	34	194888	396.82	21.43
pn2112_table	7/32	5404	23	124292	261.23	9.80
max512	9/6	4776	37	176712	353	20.65
addm4	9/8	4567	34	155278	315.31	17.03
m4	8/16	4359	37	161283	328.21	19.20
dist	8/5	3839	31	119009	241.79	11.97

is added so that the output, OUT-3, is produced simultaneously with the other outputs. It is to be noted that more than one buffer may have to be inserted between any two directly-connected gates in the mapped circuit. The number of buffers inserted is equal to the number of intermediate levels between the two gates.

- (b) Second, it uses our standard-cell library to compute the size (no. of nodes), depth (no. of gate levels) and average-energy statistics of the final network.

TABLE 5.5: Figures-of-Merit of XMGs Mapped to DW Motion-based Native Gates

Bench- mark Circuit.	No. of Inputs/ Outputs	Mapped XMG				
		Size	Depth	Size- Depth	Energy (pJ)	EDP (10^{-18} J-s)
prom1	9/40	43842 (25.17%↓)	33 (21.43%↓)	1446786 (29.88%↓)	2949.68 (39.73%↓)	154.86 (52.05%↓)
prom2	9/21	19239 (27.39%↓)	37 (19.57%↓)	711843 (41.60%↓)	1444.2 (40.53%↓)	84.49 (51.68%↓)
apex4	9/18	19975 (19.39%↓)	32 (23.81%↓)	639200 (38.59%↓)	1318.28 (37.00%↓)	67.23 (51.32%↓)
ex1010	10/10	17607 (23.50%↓)	33 (23.26%↓)	581031 (41.29%↓)	1183.36 (39.91%↓)	62.13 (53.26%↓)
test4	8/30	13634 (25.00%↓)	27 (18.18%↓)	368118 (38.64%↓)	766.82 (36.28%↓)	33.36 (47.21%↓)
exps	8/38	8183 (23.01%↓)	26 (23.53%↓)	212758 (41.13%↓)	443.32 (39.26%↓)	18.62 (52.76%↓)
alu4	14/8	8049 (16.46%↓)	51 (10.53%↓)	410499 (25.25%↓)	820.33 (23.78%↓)	65.22 (31.53%↓)
bench1	8/10	6343 (17.45%↓)	25 (26.47%↓)	158575 (39.30%↓)	331.59 (37.41%↓)	13.43 (53.06%↓)
cavlc	10/11	5002 (13.43%↓)	30 (0.00%↓)	150060 (13.43%↓)	311.74 (13.06%↓)	14.96 (13.06%↓)
test1	8/10	4442 (22.51%↓)	24 (29.41%↓)	106608 (45.30%↓)	227.19 (42.75%↓)	8.86 (58.65%↓)
pn2112- _table	7/32	4808 (11.03%↓)	21 (8.70%↓)	100968 (18.77%↓)	215.82 (17.38%↓)	7.45 (24.00%↓)
max512	9/6	3562 (25.42%↓)	29 (21.62%↓)	103298 (41.54%↓)	211.96 (39.95%↓)	9.86 (52.27%↓)
addm4	9/8	3229 (29.30%↓)	27 (20.59%↓)	87183 (43.85%↓)	182.96 (41.97%↓)	7.96 (53.26%↓)
m4	8/16	3520 (19.25%↓)	25 (32.43%↓)	88000 (45.44%↓)	185.32 (43.53%↓)	7.51 (60.91%↓)
dist	8/5	2934 (23.57%↓)	26 (16.13%↓)	76284 (35.90%↓)	160.04 (33.81%↓)	6.72 (43.84%↓)

5.4 Results and Analysis

Now, we will evaluate the energy performance of the XOR-gate proposed in Fig. 5.3. First, we carry out a comprehensive comparison between the proposed design and the baseline design in Fig. 5.11. In [7], the input operands to the XOR gate are provided by switching the free layer of MTJs. It is

TABLE 5.6: Figures-of-Merit of {AND, Inverter}-Mapped AIGs

Bench- mark Circuit	No. of Inputs/ Outputs	{AND, Inverter}-Mapped AIG				
		Size	Depth	Size· Depth	Energy (pJ)	EDP (10^{-18} J-s)
test3	10/35	38231	34	1299854	2741.41	148.04
prom1	9/40	34709	30	1041270	2242.20	107.63
prom2	9/21	14992	29	434768	942.62	43.83
apex4	9/18	14844	33	178128	1055.88	55.43
ex1010	10/10	14280	33	471240	1000.88	52.55
test4	8/30	11450	26	297700	640.92	26.92
alu4	14/8	7500	23	172500	358.50	13.44
bench1	9/9	4703	27	126981	270.86	11.78
ex5p	8/63	4431	22	97482	210.62	7.58
pn2112_table	7/32	4035	23	92805	210.94	7.91
cavlc	10/10	3604	30	108120	234.48	11.26
test1	8/10	3408	29	98832	210.57	9.79
exam	10/9	3335	27	90045	195.29	8.49
max512	9/6	2530	26	65780	141.33	5.94
max1024	10/6	2065	20	41300	86.82	2.87

a well-proven fact that STT-driven DW motion is more energy- cum area-efficient than switching the free layer of an MTJ by STT [57, 99]. This gives our DW motion-based design a clear advantage over the design in [7]. To ensure a fair comparison, we implemented an improved version of the XOR gate in [7]. In this improved version (baseline), the original configuration of MTJ network remains unchanged. Instead of programming these MTJs by switching their free layers, we employ DW motion for the same. For example, shifting the DW left (right) in the mCell device can store a 1 (0) in the MTJ. The operation of the baseline implementation also comprises of reset, write and read phases – each being 1.5 ns long. Thus, *the total delay of the proposed and the baseline gates are same (=4.5 ns)*. While the reset and write operations are for storing the inputs in the MTJs of the network,

TABLE 5.7: Figures-of-Merit of {XOR (proposed), AND, Inverter}-Mapped AIGs

Bench- mark Circuit	No. of Inputs/ Outputs	{XOR (proposed), AND, Inverter}-Mapped AIG				
		Size	Depth	Size- Depth	Energy (pJ)	EDP (10^{-18} J-s)
test3	10/35	34123 (10.75%↓)	31 (8.82%↓)	1057813 (18.62%↓)	2240.28 (18.3%↓)	110.89 (25.09%↓)
prom1	9/40	31260 (9.94%↓)	30 (0.00%↓)	937800 (9.94%↓)	2013.93 (10.2%↓)	96.67 (10.18%↓)
prom2	9/21	13444 (10.33%↓)	29 (0.00%↓)	389876 (10.33%↓)	843.23 (10.5%↓)	39.21 (10.54%↓)
apex4	9/18	13748 (7.38%↓)	30 (9.09%↓)	41244 (76.85%↓)	894.73 (15.3%↓)	42.95 (22.53%↓)
ex1010	10/10	12965 (9.21%↓)	31 (6.06%↓)	401915 (14.71%↓)	857.07 (14.4%↓)	42.42 (19.26%↓)
test4	8/30	10223 (10.72%↓)	26 (0.00%↓)	265798 (10.72%↓)	572.17 (10.7%↓)	24.03 (10.73%↓)
alu4	14/8	7518 (0.24%↑)	24 (4.35%↑)	180432 (4.60%↑)	373.81 (4.27%↑)	14.58 (8.44%↑)
bench1	9/9	4282 (8.95%↓)	27 (0.00%↓)	115614 (8.95%↓)	247.51 (8.62%↓)	10.77 (8.62%↓)
ex5p	8/63	4308 (2.78%↓)	20 (9.09%↓)	86160 (11.61%↓)	186.74 (11.3%↓)	6.16 (18.73%↓)
pn2112- _table	7/32	3693 (8.48%↓)	22 (4.35%↓)	81246 (12.46%↓)	185.28 (12.20%↓)	6.67 (15.68%↓)
cavlc	10/10	3576 (0.78%↓)	31 (3.33%↑)	110856 (2.53%↑)	240.84 (2.71%↑)	11.92 (5.92%↑)
test1	8/10	3080 (9.62%↓)	26 (10.34%↓)	80080 (18.97%↓)	172.58 (18.00%↓)	7.25 (25.97%↓)
exam	10/9	3134 (6.03%↓)	27 (0.00%↓)	84618 (6.03%↓)	184.17 (5.69%↓)	8.01 (5.69%↓)
max512	9/6	2297 (9.21%↓)	26 (0.00%↓)	59722 (9.21%↓)	129.07 (8.68%↓)	5.42 (8.68%↓)
max1024	10/6	2182 (5.67%↑)	20 (0.00%↓)	43640 (5.67%↑)	92.45 (6.48%↑)	3.05 (6.48%↑)

the read operation performs the logic. Table 5.8 [84] highlights the phase-wise energy-consumption of the proposed and the baseline XOR gates for different inputs. The proposed gate spends an average of 49.5%, 36.4% and 13.5% of its total energy on reset, write and read operations, respectively. *Whereas the proposed gate has more-or-less similar read-energy as the baseline, it consumes 63.4% less reset-energy and 68.24% less write-energy than*

TABLE 5.8: Energy Values of the Proposed and Baseline [7] XOR-gates

Inputs		Reset (fJ)		Write (fJ)		Read (fJ)	
a	b	Pro- posed	Base- line	Pro- posed	Base- line	Pro- posed	Base- line
0	0	4.7027	12.8467	5.91e-06	14.45	1.295	1.2211
0	1	4.7026	12.8467	4.825	14.33	1.314	1.2234
1	0	4.7026	12.8467	4.825	14.20	1.314	1.2215
1	1	4.7026	12.8467	8.360	14.08	1.175	1.2232

FIGURE 5.11: Baseline XOR gate from Fig. 4(b) of [7].

the baseline. The total energy-consumption of the proposed gate varies from ≈ 6.0 fJ to 14.237 fJ and has an average value of ≈ 10.48 fJ. *On an average, the proposed gate is 63% more energy-efficient than the baseline.* This can be attributed to the fact that the XOR gate in [7] requires more copies of the input operands and, hence, more number of writes and resets in the DW motion-based implementation or more number of energy-hungry MTJ-switching operations in the original version. *The proposed gate has 2 MTJs while the baseline gate has 6 MTJs.*

Next, in Table 5.3, we present the average-energy consumed by the proposed gate and the other DW-gates used for mapping the networks synthesized in our simulation framework. Table 5.3 reveals that the proposed XOR-gate consumes more energy in each of the individual stages than the other three gates. *On an average, overall energy consumption of the proposed gate is 50.72%, 43.67% and 33.71% more than that of the Inverter, AND and Majority gates, respectively.* At this point, we would like to draw the reader's attention to the following key points:

- (a) As shown in Fig. 5.3, the proposed device possesses two ferromagnetic strips – one wider than the other. The DW in the wider strip, r_4 , requires higher current than the DW in the thinner strip, r_2 , for depinning and motion within 1-5 ns. In order to ensure that both the DWs are reset to their initial positions, I_{reset} has to be equal to the greater of these two current-values. Also, the set of Boolean inputs requiring change in resistance of both the MTJs, M_1 and M_2 , of the device and, hence, motion of both the DWs, result in larger write-current. On the other hand, the device in Fig. 5.7 has only one strip containing DW. Consequently, it requires less energy for reset and write operations, than the proposed device.
- (b) In addition, we can observe in Table 5.3 that: (a) *the energy-consumptions (total) of all the gates are dominated by their reset- and write-energies*
(b) *the read energies of these gates do not differ much from each other.*

The proposed XOR-gate, thus, proves to be more energy-hungry.

Next, we run the simulation framework in Fig. 5.6 on different circuits and generate their XMG-, MIG-synthesized networks. The framework maps the synthesized XMGs and MIGs to DW motion-based {XOR (proposed), Majority and Inverter} and {Majority, Inverter} gates, respectively, and computes their {size, depth, energy} performances. Note that XOR is used in conjunction with Majority and Inverter as its expressive power is not enough to alone map an entire network. The circuits used in our study are combinational circuits obtained from a mix of popular benchmark-suites including LGSynth'91, IWLS'93, IWLS 2005, ISCAS, Advanced Synthesis Cookbook, Espresso, EPFL, Generic and others [100]. Because of limited computing-power and lengthy measurement-time, we could obtain the performance estimates for only subsets of these benchmark-suites in the available time. Out of the 139 circuits processed, the performance figures for only the-fifteen-largest circuits are shown in Table 5.4 and Table 5.5 due to page limitation. Here, note that we draw our conclusions based on the performances in *all* the processed circuits. Our study demonstrates that the mapped XMGs generally outperform the mapped MIGs in all the four figures-of-merit. *On an average, the mapped XMGs have 31.54% fewer nodes and 19.00% less depth in comparison to the mapped MIGs. Also, we compare the size · depth metric of the mapped XMGs and MIGs and observe that the former leads by 41.56%. It is interesting to note that, despite the higher energy-consumption of the proposed XOR-gate, the mapped XMGs achieve 38.03% improved energy-performance than the mapped MIGs.* The reason behind this finding is two-fold:

- (a) *Synthesis-level*: Due to the the high expressive-power of XOR primitive, XMG synthesis is more effective at compressing the logic networks.
- (b) *Mapping-level*: The presence of XOR gate in the mapping library allows the mapping to natively recognize and preserve the XOR nodes, thereby, enabling the mapped circuits to inherit the reduced size and depth of the synthesized XMGs.

The energy savings brought about by these two factors effectively outweigh the higher energy-consumption of the proposed XOR-gates in the mapped circuits. Overall, *the mapped XMGs are 45.47% better than the mapped MIGs in terms of energy-delay product, thanks to the proposed XOR-gate.*

The study so far evaluates the performance of the proposed XOR-gate in mapping XMG-synthesized networks. To further explore the potential of this gate, we extend our analysis to understanding its impact on mapping synthesized AIGs. So, we map synthesized AIGs to two different libraries of DW motion-based gates – {XOR (proposed), AND, Inverter} and {AND, Inverter} – and then, compare their circuit-level performances. The framework in Fig 5.6 is re-utilized for obtaining the post-mapping performances of synthesized AIGs. A total of 150 combinational circuits from the above set of benchmark suites are processed using this framework. Table 5.6 and Table 5.7 present the results obtained for the-fifteen-biggest combinational circuits. Let us now look into the overall performances of the XOR-based and the XOR-free mappings in these benchmark circuits. Similar to the XMGs, the AIGs are also benefitted by the use of the proposed XOR-gate for mapping. For example, *the XOR-based mapping of AIGs achieves an average of 9.26% reduction in depth as compared to the XOR-free mapping.* In addition, *the {XOR (proposed), AND, Inverter}-mapped AIGs have 13.39% (average) fewer nodes than the {AND, Inverter}-mapped AIGs. Considering size-depth as a performance indicator, the former is found to be 17.74% better than the latter.* This decrease in size and depth yields energy saving which, in turn, counteracts the rise in network energy caused by the energy-expensive XOR gates. We observe that *the {XOR (proposed), AND, Inverter}-mapped AIGs improve on the energy consumption of the {AND, Inverter}-mapped AIGs by an average of 15.90%. EDP-wise, the former outperforms the latter by 19%.* Comparing these results with those for mapped XMGs, we can see that *the proposed XOR-gate has brought about more significant improvements (in %) in the performances of the mapped XMGs than in those of the mapped AIGs.* This observation can be explained by the fact that the XOR-based mapping

of XMGs, unlike that of AIGs, is preceded by XOR-based synthesis. XMG synthesis recognizes XORs and leads to greater compression of the networks. On the contrary, AIG synthesis is unable to utilize XORs explicitly. And, the technology mapping of AIGs is not as strong in XOR identification, as the XMG synthesis. Consequently, the XOR-mapped AIGs are not able to achieve performance improvements as large as those by the mapped XMGs.

5.5 Conclusion

To this end, the study presented in this chapter is an effort to improve the mapping of XMG- and AIG-synthesized circuits to spintronic fabric. We demonstrated how a DW motion-based XOR-gate can be utilized to effectively address this challenge. We performed extensive circuit-level simulations to benchmark this gate over other DW motion-based gates. Furthermore, a framework was developed for accurately evaluating the energy- and delay-performances of synthesized-cum-mapped DW-circuits. We observed that the use of this XOR-gate reduced the {size, depth, size·depth, energy, EDP} values of mapped MIGs by {31.54%, 19.00%, 41.56%, 38.03%, 45.47%} and of mapped AIGs by {13.39%, 9.26%, 17.74%, 15.90%, 19%}. The impact of the decrease in network size on the area comparison of mapped circuits is something that remains to be investigated. Note that the speed and energy efficiency of the mapped circuits can be further alleviated by using more-efficient phenomenon, like Spin Orbit Torque (SOT), for DW motion in the constituent gates. Altogether, we are optimistic that the findings of this study will reinforce the prospects of spin devices in replacing CMOS, at least partially, in the post-Moore era.

6

Conclusion and Future Research

Spintronics is emerging as a game-changer in the post-CMOS era of key areas like, memory, logic and neuromorphic computing, due to its attributes of zero-standby power, non-volatility, CMOS compatibility and extremely high endurance. The objective of this thesis has been to identify key issues in the design of spintronic circuits for Boolean and non-Boolean (neuromorphic) applications and propose circuit-techniques to alleviate these issues.

We started with proposing a design technique to minimize the number of write operations and, hence, the energy consumption of spin-CMOS logic circuits. We demonstrated the effectiveness of the proposed technique by applying it to the spin-based implementation of SIMON – a cryptographic block cipher. We observed that the proposed technique leads to impressive improvements in the energy and delay performances of the spin-based logic circuit.

Next, we worked towards providing a solution to tackle the fundamental challenge of classifying linearly-inseparable functions using a single neuron. We proposed two novel domain wall motion-based dual-threshold activation units with additional non-linearity in the transfer function. We also developed a new learning algorithm for training the weights of these neurons. We carried out extensive tests to examine the performance of these spin-based designs of neuron and their learning algorithms. The obtained results indicated the proposed learning algorithms can obtain better MCR results than the perceptron learning algorithm while the neurons consumed ultra-low computation energy.

The subsequent work derived its inspiration from the above work in neuromorphic computing. We proposed a variant of the above-proposed neurons to realize the XOR operation that is used frequently in the domain of Boolean logic. We demonstrated how this XOR-design can significantly improve the mapping of AIG- and XMG-synthesized logic networks to spintronic hardware. Simulation results indicated that the proposed gate can improve the performance of AIGs and XMGs on multiple fonts, like size, depth, size·depth, energy-consumption and EDP.

6.1 Future Research

We hope that the presented designs and techniques will contribute to improving the likelihood of spin devices and circuits being adopted by system designers for implementing embedded and IoT systems. Next, we outline some research directions that we think can be investigated as future research.

6.1.1 Further Optimization of Spin-based SIMON

In Chapter 3, we studied the impact of using spin-CMOS composite logic-gates on the hardware performance of SIMON. It is to be noted here that the output of these composite gates are stored in racetracks using the write

operation. The energy efficiency of these gates can be further improved if a shift-based write mechanism is used to store the output. The improvement rendered by this technique is something that needs to be studied in comparison with CMOS-based implementation of SIMON. Another observation that we have is that XOR is the most frequently occurring operation in SIMON. The XOR gate that we have proposed in chapter 5 can be leveraged to obtain a very light-weight spin-implementation of SIMON. It would be interesting to study the hardware performance of the XOR (proposed)-based implementation of SIMON and see where it stands in comparison to the CMOS implementation.

6.1.2 Multi-Layered Network of Dual-Threshold Neurons

In chapter 4, we studied the performance of the proposed neurons in a single-layered network. The study can be extended to a multi-layered neural network. If we visualize the training of the dual tunable-threshold neuron, we can understand that the slope of the two thresholds with respect to each other remains constant and only the distance between them can be modulated. The proposed neuron can be made more-suitable for a multi-layered neural network by having a design that also allows the relative slope of the thresholds to be programmed. Another extension that we think might be interesting to study in the context of this work is if more thresholds can be added to the neuron. If yes, will it be useful for any Boolean logic or machine learning application?

6.1.3 MRAM-based In-Memory Acceleration of ANNs

Though this thesis doesn't address any challenge(s) related to MRAM, we would, here, like to take the opportunity to suggest an interesting direction of research. As we know, MTJ is the most-matured spin-technology – so much that MRAMs are now commercially available. In-memory computing

is an emerging area that is gaining attention since it eliminates the energy- and delay-expensive operations of *load* and *store*. It performs computations and storage of their results in the same memory – here, MRAM. So, it makes sense to use the MRAM as an accelerator for machine learning models like ANNs and CNNs. It would be interesting to study the challenges involved in designing a scheme that employs the same MRAM for training as well as inference. Also, how would such an accelerator performs in terms of accuracy, area, energy, delay etc. on real-world applications?

Bibliography

- [1] H. S. P. Wong and S. Salahuddin, “Memory leads the way to better computing,” *Nature Nanotechnology*, vol. 10, no. 3, pp. 191–194, 2015.
- [2] S. H. Kang, “Embedded stt-mram for mobile applications: Enabling advanced chip architectures,” in *Non-volatile Memories Workshop*, 2010.
- [3] Y. Xie, “Modeling, architecture, and applications for emerging memory technologies,” in *IEEE Design Test of Computers*. IEEE, 2011, pp. 44–51.
- [4] M. Hosomi, H. Yamagishi, T. Yamamoto, K. Bessho, Y. Higo, K. Yamane, H. Yamada, M. Shoji, H. Hachino, C. Fukumoto *et al.*, “A novel nonvolatile memory with spin torque transfer magnetization switching: Spin-ram,” *Electron Devices Meeting, IEDM Technical Digest*, pp. 459–462, 2005.
- [5] S. Fukami, T. Suzuki, K. Nagahara, N. Ohshima, Y. Ozaki, S. Saito, R. Nebashi, N. Sakimura, H. Honjo, K. Mori *et al.*, “Low-current perpendicular domain wall motion cell for scalable high-speed mram,” *VLSI Technology, 2009 Symposium on*, pp. 230–231, 2009.
- [6] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, “The SIMON and SPECK families of lightweight block

- ciphers,” *IACR Cryptology ePrint Archive*, vol. 2013, p. 404, 2013. [Online]. Available: <http://eprint.iacr.org/2013/404>
- [7] H.-P. Trinh, W. Zhao, J.-O. Klein, Y. Zhang, D. Ravelsona, and C. Chappert, “Magnetic adder based on racetrack memory,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 6, pp. 1469–1477, 2013.
- [8] N. S. Kim, T. M. Austin, D. T. Blaauw, T. N. Mudge, K. Flautner, J. S. Hu, M. J. Irwin, M. T. Kandemir, and N. Vijaykrishnan, “Leakage current: Moore’s law meets static power,” *IEEE Computer*, vol. 36, no. 12, pp. 68–75, 2003. [Online]. Available: <https://doi.org/10.1109/MC.2003.1250885>
- [9] S. Ikeda, K. Miura, H. Yamamoto, K. Mizunuma, H. Gan, M. Endo, S. Kanai, J. Hayakawa, F. Matsukura, and H. Ohno, “A perpendicular-anisotropy coFeB–mgo magnetic tunnel junction,” *Nature materials*, vol. 9, no. 9, p. 721, 2010.
- [10] M. Hayashi, L. Thomas, R. Moriya, C. Rettner, and S. S. Parkin, “Current-controlled magnetic domain-wall nanowire shift register,” *Science*, vol. 320, no. 5873, pp. 209–211, 2008.
- [11] S. Wolf, D. Awschalom, R. Buhrman, J. Daughton, S. Von Molnar, M. Roukes, A. Y. Chtchelkanova, and D. Treger, “Spintronics: a spin-based electronics vision for the future,” *science*, vol. 294, no. 5546, pp. 1488–1495, 2001.
- [12] J.-P. Wang and X. Yao, “Programmable spintronic logic devices for reconfigurable computation and beyond history and outlook,” *Journal of Nanoelectronics and Optoelectronics*, vol. 3, no. 1, pp. 12–23, 2008.
- [13] M. Hayashi, L. Thomas, R. Moriya, C. Rettner, and S. S. Parkin, “Current-controlled magnetic domain-wall nanowire shift register,” *Science*, vol. 320, no. 5873, pp. 209–211, 2008.

- [14] S. Fukami, T. Suzuki, Y. Nakatani, N. Ishiwata, M. Yamanouchi, S. Ikeda, N. Kasai, and H. Ohno, “Current-induced domain wall motion in perpendicularly magnetized cobe nanowire,” *Applied Physics Letters*, vol. 98, no. 8, p. 082504, 2011.
- [15] L. Thomas, R. Moriya, C. Rettner, and S. S. Parkin, “Dynamics of magnetic domain walls under their own inertia,” *Science*, vol. 330, no. 6012, pp. 1810–1813, 2010.
- [16] L. Thomas, S.-H. Yang, K.-S. Ryu, B. Hughes, C. Rettner, D.-S. Wang, C.-H. Tsai, K.-H. Shen, and S. S. Parkin, “Racetrack memory: a high-performance, low-cost, non-volatile memory based on magnetic domain walls,” in *Electron Devices Meeting (IEDM), 2011 IEEE International*. IEEE, 2011, pp. 24–2.
- [17] G. Tatara and H. Kohno, “Theory of current-driven domain wall motion: Spin transfer versus momentum transfer,” *Physical review letters*, vol. 92, no. 8, p. 086601, 2004.
- [18] L. Berger, “Emission of spin waves by a magnetic multilayer traversed by a current,” *Physical Review B*, vol. 54, no. 13, p. 9353, 1996.
- [19] J. C. Slonczewski, “Current-driven excitation of magnetic multilayers,” *Journal of Magnetism and Magnetic Materials*, vol. 159, no. 1-2, pp. L1–L7, 1996.
- [20] A. Brataas, A. D. Kent, and H. Ohno, “Current-induced torques in magnetic materials,” *Nature materials*, vol. 11, no. 5, p. 372, 2012.
- [21] S. Fukami, M. Yamanouchi, K.-J. Kim, T. Suzuki, N. Sakimura, D. Chiba, S. Ikeda, T. Sugibayashi, N. Kasai, T. Ono *et al.*, “20-nm magnetic domain wall motion memory with ultralow-power operation,” in *Electron Devices Meeting (IEDM), 2013 IEEE International*. IEEE, 2013, pp. 3–5.

- [22] K. Ikeda, H. Awano *et al.*, “Direct observation of domain wall motion induced by low-current density in tbfecu wires,” *Applied physics express*, vol. 4, no. 9, p. 093002, 2011.
- [23] S. Muroga, T. Tsuboi, and C. R. Baugh, “Enumeration of threshold functions of eight variables,” *IEEE Transactions on Computers*, vol. 100, no. 9, pp. 818–825, 1970.
- [24] I. Aizenberg, N. N. Aizenberg, and J. P. Vandewalle, *Multi-Valued and Universal Binary Neurons: Theory, Learning and Applications*. Springer Science & Business Media, 2013.
- [25] M. Julliere, “Tunneling between ferromagnetic films,” *Physics letters A*, vol. 54, no. 3, pp. 225–226, 1975.
- [26] C. Chappert, A. Fert, and F. N. Van Dau, “The emergence of spin electronics in data storage,” *Nanoscience And Technology: A Collection of Reviews from Nature Journals*, pp. 147–157, 2010.
- [27] M.-H. Jo, N. Mathur, N. Todd, and M. Blamire, “Very large magnetoresistance and coherent switching in half-metallic manganite tunnel junctions,” *Physical Review B*, vol. 61, no. 22, p. R14905, 2000.
- [28] J. Sun, D. Abraham, K. Roche, and S. Parkin, “Temperature and bias dependence of magnetoresistance in doped manganite thin film trilayer junctions,” *Applied physics letters*, vol. 73, no. 7, pp. 1008–1010, 1998.
- [29] M. Bowen, M. Bibes, A. Barthélémy, J.-P. Contour, A. Anane, Y. Lemaitre, and A. Fert, “Nearly total spin polarization in $\text{La}_{2/3}\text{Sr}_{1/3}\text{MnO}_3$ from tunneling experiments,” *Applied Physics Letters*, vol. 82, no. 2, pp. 233–235, 2003.
- [30] J. S. Moodera, L. R. Kinder, T. M. Wong, and R. Meservey, “Large magnetoresistance at room temperature in ferromagnetic thin film tunnel junctions,” *Physical review letters*, vol. 74, no. 16, p. 3273, 1995.

- [31] T. Miyazaki and N. Tezuka, "Giant magnetic tunneling effect in fe/al₂o₃/fe junction," *Journal of magnetism and magnetic materials*, vol. 139, no. 3, pp. L231–L234, 1995.
- [32] D. Wang, C. Nordman, J. M. Daughton, Z. Qian, and J. Fink, "70% tnr at room temperature for sdt sandwich junctions with cofeb as free and reference layers," *IEEE Transactions on Magnetism*, vol. 40, no. 4, pp. 2269–2271, 2004.
- [33] W. Butler, X.-G. Zhang, T. Schulthess, and J. MacLaren, "Spin-dependent tunneling conductance of fe—mgo—fe sandwiches," *Physical Review B*, vol. 63, no. 5, p. 054416, 2001.
- [34] J. Mathon and A. Umerski, "Theory of tunneling magnetoresistance of an epitaxial fe/mgo/fe (001) junction," *Physical Review B*, vol. 63, no. 22, p. 220403, 2001.
- [35] S. S. Parkin, C. Kaiser, A. Panchula, P. M. Rice, B. Hughes, M. Samant, and S.-H. Yang, "Giant tunnelling magnetoresistance at room temperature with mgo (100) tunnel barriers," *Nature materials*, vol. 3, no. 12, p. 862, 2004.
- [36] S. Yuasa, T. Nagahama, A. Fukushima, Y. Suzuki, and K. Ando, "Giant room-temperature magnetoresistance in single-crystal fe/mgo/fe magnetic tunnel junctions," *Nature materials*, vol. 3, no. 12, p. 868, 2004.
- [37] C. W. Smullen, V. Mohan, A. Nigam, S. Gurumurthi, and M. R. Stan, "Relaxing non-volatility for fast and energy-efficient stt-ram caches," *High Performance Computer Architecture (HPCA), 2011 IEEE 17th International Symposium on*, pp. 50–61, 2011.
- [38] A. Jog, A. K. Mishra, C. Xu, Y. Xie, V. Narayanan, R. Iyer, and C. R. Das, "Cache revive: architecting volatile stt-ram caches for enhanced performance in cmps," *Proceedings of the 49th Annual Design Automation Conference*, pp. 243–252, 2012.

- [39] S. Fukami, H. Sato, M. Yamanouchi, S. Ikeda, F. Matsukura, and H. Ohno, “Advances in spintronics devices for microelectronics from spin-transfer torque to spin-orbit torque,” *Design Automation Conference (ASP-DAC), 2014 19th Asia and South Pacific*, pp. 684–691, 2014.
- [40] P. Zhou, B. Zhao, J. Yang, and Y. Zhang, “Energy reduction for stt-ram using early write termination,” *Proceedings of the 2009 International Conference on Computer-Aided Design*, pp. 264–268, 2009.
- [41] R. Bishnoi, F. Oboril, M. Ebrahimi, and M. B. Tahoori, “Avoiding unnecessary write operations in stt-mram for low power implementation,” *Quality Electronic Design (ISQED), 2014 15th International Symposium on*, pp. 548–553, 2014.
- [42] G. Sun, X. Dong, Y. Xie, J. Li, and Y. Chen, “A novel architecture of the 3d stacked mram l2 cache for cmps,” *High Performance Computer Architecture, 2009. HPCA 2009. IEEE 15th International Symposium on*, pp. 239–249, 2009.
- [43] R. Venkatesan, M. Sharad, K. Roy, and A. Raghunathan, “Dwm-tapestri-an energy efficient all-spin cache using domain wall shift based writes,” *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013*, pp. 1825–1830, 2013.
- [44] J. Jung, Y. Nakata, M. Yoshimoto, and H. Kawaguchi, “Energy-efficient spin-transfer torque ram cache exploiting additional all-zero-data flags,” *Quality Electronic Design (ISQED), 2013 14th International Symposium on*, pp. 216–222, 2013.
- [45] J. Ahn and K. Choi, “Lower-bits cache for low power stt-ram caches,” *Circuits and Systems (ISCAS), 2012 IEEE International Symposium on*, pp. 480–483, 2012.
- [46] M. Sharad, D. Fan, and K. Roy, “Ultra low power associative computing with spin neurons and resistive crossbar memory,” in *The*

- 50th Annual Design Automation Conference 2013, DAC '13, Austin, TX, USA, May 29 - June 07, 2013*, 2013, pp. 107:1–107:6. [Online]. Available: <https://doi.org/10.1145/2463209.2488866>
- [47] D. Fan, M. Sharad, A. Sengupta, and K. Roy, “Hierarchical temporal memory based on spin-neurons and resistive memory for energy-efficient brain-inspired computing,” *IEEE Trans. Neural Netw. Learning Syst.*, vol. 27, no. 9, pp. 1907–1919, 2016. [Online]. Available: <https://doi.org/10.1109/TNNLS.2015.2462731>
- [48] D. Fan, Y. Shim, A. Raghunathan, and K. Roy, “STT-SNN: A spin-transfer-torque based soft-limiting non-linear neuron for low-power artificial neural networks,” *CoRR*, vol. abs/1412.8648, 2014. [Online]. Available: <http://arxiv.org/abs/1412.8648>
- [49] A. Sengupta and K. Roy, “Spin-transfer torque magnetic neuron for low power neuromorphic computing,” in *2015 International Joint Conference on Neural Networks, IJCNN 2015, Killarney, Ireland, July 12-17, 2015*, 2015, pp. 1–7. [Online]. Available: <https://doi.org/10.1109/IJCNN.2015.7280306>
- [50] A. Sengupta, Y. Shim, and K. Roy, “Proposal for an all-spin artificial neural network: Emulating neural and synaptic functionalities through domain wall motion in ferromagnets,” *IEEE Trans. Biomed. Circuits and Systems*, vol. 10, no. 6, pp. 1152–1160, 2016. [Online]. Available: <https://doi.org/10.1109/TBCAS.2016.2525823>
- [51] A. Sengupta, M. Parsa, B. Han, and K. Roy, “Probabilistic deep spiking neural systems enabled by magnetic tunnel junction,” *CoRR*, vol. abs/1605.04494, 2016. [Online]. Available: <http://arxiv.org/abs/1605.04494>
- [52] A. Sengupta, A. Banerjee, and K. Roy, “Hybrid spintronic-cmos spiking neural network with on-chip learning: Devices, circuits

- and systems,” *CoRR*, vol. abs/1510.00432, 2015. [Online]. Available: <http://arxiv.org/abs/1510.00432>
- [53] D. Zhang, L. Zeng, K. Cao, M. Wang, S. Peng, Y. Zhang, Y. Zhang, J. Klein, Y. Wang, and W. Zhao, “All spin artificial neural networks based on compound spintronic synapse and neuron,” *IEEE Trans. Biomed. Circuits and Systems*, vol. 10, no. 4, pp. 828–836, 2016. [Online]. Available: <https://doi.org/10.1109/TBCAS.2016.2533798>
- [54] G. Srinivasan, A. Sengupta, and K. Roy, “Magnetic tunnel junction enabled all-spin stochastic spiking neural network,” in *Design, Automation & Test in Europe Conference & Exhibition, DATE 2017, Lausanne, Switzerland, March 27-31, 2017*, 2017, pp. 530–535. [Online]. Available: <https://doi.org/10.23919/DATE.2017.7927045>
- [55] S. Deb, A. Chattopadhyay, and H. Yu, “Energy optimization of racetrack memory-based SIMON block cipher,” in *IEEE Computer Society Annual Symposium on VLSI, ISVLSI 2016, Pittsburgh, PA, USA, July 11-13, 2016*, 2016, pp. 431–436. [Online]. Available: <https://doi.org/10.1109/ISVLSI.2016.103>
- [56] W. Zhao, C. Chappert, V. Javerliac, and J.-P. Noziere, “High speed, high stability and low power sensing amplifier for mtj/cmos hybrid logic circuits,” *IEEE Transactions on Magnetics*, vol. 45, no. 10, pp. 3784–3787, 2009.
- [57] Y. Zhang, W. Zhao, D. Ravelosona, J.-O. Klein, J.-V. Kim, and C. Chappert, “Perpendicular-magnetic-anisotropy cofeb racetrack memory,” *Journal of Applied Physics*, vol. 111, no. 9, p. 093925, 2012.
- [58] Y. Zhang, W. Zhao, Y. Lakys, J.-O. Klein, J.-V. Kim, D. Ravelosona, and C. Chappert, “Compact modeling of perpendicular-anisotropy cofeb/mgo magnetic tunnel junctions,” *IEEE Transactions on Electron Devices*, vol. 59, no. 3, pp. 819–826, 2012.

- [59] D. M. Bromberg and D. H. Morris, “mcell model,” Jan 2015. [Online]. Available: <https://nanohub.org/publications/13/2>
- [60] B. A. Banik, S. and F. Regazzoni, “Exploring energy efficiency of lightweight block ciphers,” *International Conference on Selected Areas in Cryptography*, pp. 178–194, 2015.
- [61] S. Deb, T. Vatwani, A. Chattopadhyay, A. Basu, and X. Fong, “Domain wall motion-based xor-like activation unit with A programmable threshold,” in *2018 International Joint Conference on Neural Networks, IJCNN 2018, Rio de Janeiro, Brazil, July 8-13, 2018*, 2018, pp. 1–8. [Online]. Available: <https://doi.org/10.1109/IJCNN.2018.8489146>
- [62] —, “Domain wall motion-based dual-threshold activation unit for low-power classification of non-linearly separable functions,” *IEEE Trans. Biomed. Circuits and Systems*, vol. 12, no. 6, pp. 1410–1421, 2018. [Online]. Available: <https://doi.org/10.1109/TBCAS.2018.2867038>
- [63] D. A. Drachman, “Do we have brain to spare?” *Neurology*, vol. 64, no. 12, pp. 2004–2005, 2005.
- [64] M. Sparkes, “Supercomputer models one second of human brain activity,” *The Telegraph*, 2014.
- [65] F. Jabr, “Does thinking really hard burn more calories?” *Scientific American*, 2012.
- [66] P. L. Bartlett and T. Downs, “Using random weights to train multilayer networks of hard-limiting units,” *IEEE Trans. Neural Networks*, vol. 3, no. 2, pp. 202–210, 1992. [Online]. Available: <https://doi.org/10.1109/72.125861>
- [67] M. Sharad, D. Fan, and K. Roy, “Ultra low power associative computing with spin neurons and resistive crossbar memory,” in *The*

- 50th Annual Design Automation Conference 2013, DAC '13, Austin, TX, USA, May 29 - June 07, 2013*, 2013, pp. 107:1–107:6. [Online]. Available: <https://doi.org/10.1145/2463209.2488866>
- [68] D. Bromberg, M. Moneck, V. Sokalski, J. Zhu, L. Pileggi, and J.-G. Zhu, “Experimental demonstration of four-terminal magnetic logic device with separate read-and write-paths,” in *Electron Devices Meeting (IEDM), 2014 IEEE International*. IEEE, 2014, pp. 33–1.
- [69] R. Kohavi, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes*, 1995, pp. 1137–1145. [Online]. Available: <http://ijcai.org/Proceedings/95-2/Papers/016.pdf>
- [70] F. Alibart, L. Gao, B. Hoskins, and D. B. Strukov, “High-precision tuning of state for memristive devices by adaptable variation-tolerant algorithm,” *CoRR*, vol. abs/1110.1393, 2011. [Online]. Available: <http://arxiv.org/abs/1110.1393>
- [71] F. M. Bayat, B. Hoskins, and D. B. Strukov, “Phenomenological modeling of memristive devices,” *Applied Physics A*, vol. 118, no. 3, pp. 779–786, 2015.
- [72] I. Kataeva, F. Merrikh-Bayat, E. Zamanidoost, and D. B. Strukov, “Efficient training algorithms for neural networks based on memristive crossbar circuits,” in *2015 International Joint Conference on Neural Networks, IJCNN 2015, Killarney, Ireland, July 12-17, 2015*, 2015, pp. 1–8. [Online]. Available: <https://doi.org/10.1109/IJCNN.2015.7280785>
- [73] D. Morris, D. M. Bromberg, J. J. Zhu, and L. T. Pileggi, “mlogic: ultra-low voltage non-volatile logic circuits using STT-MTJ devices,” in *The 49th Annual Design Automation Conference 2012, DAC '12*,

- San Francisco, CA, USA, June 3-7, 2012*, 2012, pp. 486–491. [Online]. Available: <https://doi.org/10.1145/2228360.2228446>
- [74] J. J. Nowak, R. P. Robertazzi, J. Z. Sun, G. Hu, J.-H. Park, J. Lee, A. J. Annunziata, G. P. Lauer, R. Kothandaraman, E. J. OSullivan *et al.*, “Dependence of voltage and size on write error rates in spin-transfer torque magnetic random-access memory,” *IEEE Magnetics Letters*, vol. 7, pp. 1–4, 2016.
- [75] Z. Diao, D. Apalkov, M. Pakala, Y. Ding, A. Panchula, and Y. Huai, “Spin transfer switching and spin polarization in magnetic tunnel junctions with mgo and alo x barriers,” *Applied Physics Letters*, vol. 87, no. 23, p. 232502, 2005.
- [76] S. Kanai, F. Matsukura, and H. Ohno, “Electric-field-induced magnetization switching in cofeb/mgo magnetic tunnel junctions with high junction resistance,” *Applied Physics Letters*, vol. 108, no. 19, p. 192406, 2016.
- [77] Z. He and D. Fan, “Energy efficient reconfigurable threshold logic circuit with spintronic devices,” *IEEE Trans. Emerging Topics Comput.*, vol. 5, no. 2, pp. 223–237, 2017. [Online]. Available: <https://doi.org/10.1109/TETC.2016.2633966>
- [78] S. Emori, U. Bauer, S.-M. Ahn, E. Martinez, and G. S. Beach, “Current-driven dynamics of chiral ferromagnetic domain walls,” *Nature materials*, vol. 12, no. 7, p. 611, 2013.
- [79] K.-S. Ryu, L. Thomas, S.-H. Yang, and S. Parkin, “Chiral spin torque at magnetic domain walls,” *Nature nanotechnology*, vol. 8, no. 7, p. 527, 2013.
- [80] D. Bhowmik, M. E. Nowakowski, L. You, O. Lee, D. Keating, M. Wong, J. Bokor, and S. Salahuddin, “Deterministic domain wall motion orthogonal to current flow due to spin orbit torque,” *Scientific reports*, vol. 5, p. 11823, 2015.

- [81] D. Lacour, J. Katine, L. Folks, T. Block, J. Childress, M. Carey, and B. Gurney, “Experimental evidence of multiple stable locations for a domain wall trapped by a submicron notch,” *Applied physics letters*, vol. 84, no. 11, pp. 1910–1912, 2004.
- [82] S. Venkataramani, A. Ranjan, K. Roy, and A. Raghunathan, “Axnn: energy-efficient neuromorphic systems using approximate computing,” in *Proceedings of the 2014 international symposium on Low power electronics and design*. ACM, 2014, pp. 27–32.
- [83] M. Lichman, “Uci machine learning repository,” 2013. [Online]. Available: <http://archive.ics.uci.edu/m>
- [84] S. Deb and A. Chattopadhyay, “Spintronic device-structure for low-energy xor logic using domain wall motion,” in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2019 (submitted).
- [85] —, “Efficient mapping of xmg- and aig-synthesized spintronic circuits using domain wall motion-based xor-gate,” *IEEE Trans. Computer Aided Design of Integrated Circuits (TCAD)*, 2019 (submitted).
- [86] G. D. Micheli, *Synthesis and optimization of digital circuits*. McGraw-Hill Higher Education, 1994.
- [87] R. E. Bryant, “Graph-based algorithms for boolean function manipulation,” *IEEE Trans. Computers*, vol. 35, no. 8, pp. 677–691, 1986. [Online]. Available: <https://doi.org/10.1109/TC.1986.1676819>
- [88] R. K. Brayton, R. L. Rudell, A. L. Sangiovanni-Vincentelli, and A. R. Wang, “MIS: A multiple-level logic optimization system,” *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 6, no. 6, pp. 1062–1081, 1987. [Online]. Available: <https://doi.org/10.1109/TCAD.1987.1270347>

- [89] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. Sangiovanni-Vincentelli, “Sis: A system for sequential circuit synthesis,” 1992.
- [90] C. Yang and M. J. Ciesielski, “BDS: a bdd-based logic optimization system,” *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 21, no. 7, pp. 866–876, 2002. [Online]. Available: <https://doi.org/10.1109/TCAD.2002.1013899>
- [91] R. K. Brayton and A. Mishchenko, “ABC: an academic industrial-strength verification tool,” in *Computer Aided Verification, 22nd International Conference, CAV 2010, Edinburgh, UK, July 15-19, 2010. Proceedings*, 2010, pp. 24–40. [Online]. Available: https://doi.org/10.1007/978-3-642-14295-6_5
- [92] A. Mishchenko, S. Chatterjee, and R. K. Brayton, “Dag-aware AIG rewriting a fresh look at combinational logic synthesis,” in *Proceedings of the 43rd Design Automation Conference, DAC 2006, San Francisco, CA, USA, July 24-28, 2006*, 2006, pp. 532–535. [Online]. Available: <https://doi.org/10.1145/1146909.1147048>
- [93] A. Kuehlmann, V. Paruthi, F. Krohm, and M. K. Ganai, “Robust boolean reasoning for equivalence checking and functional property verification,” *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 21, no. 12, pp. 1377–1394, 2002. [Online]. Available: <https://doi.org/10.1109/TCAD.2002.804386>
- [94] L. G. Amarù, P. Gaillardon, and G. D. Micheli, “Majority-inverter graph: A novel data-structure and algorithms for efficient logic optimization,” in *The 51st Annual Design Automation Conference 2014, DAC '14, San Francisco, CA, USA, June 1-5, 2014*, 2014, pp. 194:1–194:6. [Online]. Available: <https://doi.org/10.1145/2593069.2593158>

- [95] L. Amarú, P.-E. Gaillardon, and G. De Micheli, “Boolean logic optimization in majority-inverter graphs,” in *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE*. IEEE, 2015, pp. 1–6.
- [96] W. Haaswijk, M. Soeken, L. G. Amarù, P. Gaillardon, and G. D. Micheli, “A novel basis for logic rewriting,” in *22nd Asia and South Pacific Design Automation Conference, ASP-DAC 2017, Chiba, Japan, January 16-19, 2017*, 2017, pp. 151–156. [Online]. Available: <https://doi.org/10.1109/ASPDAC.2017.7858312>
- [97] D. Fan, M. Sharad, and K. Roy, “Design and synthesis of ultralow energy spin-memristor threshold logic,” *IEEE Transactions on Nanotechnology*, vol. 13, no. 3, pp. 574–583, 2014.
- [98] M. Soeken, “Cirkit.” [Online]. Available: <https://github.com/msoeken/cirkit>
- [99] R. Venkatesan, V. J. Kozhikkottu, C. Augustine, A. Raychowdhury, K. Roy, and A. Raghunathan, “Tapecache: a high density, energy efficient cache based on domain wall memory,” in *International Symposium on Low Power Electronics and Design, ISLPED’12, Redondo Beach, CA, USA - July 30 - August 01, 2012*, 2012, pp. 185–190. [Online]. Available: <https://doi.org/10.1145/2333660.2333707>
- [100] P. Fišer and J. Schmidt, “A comprehensive set of logic synthesis and optimization examples,” in *12th. Int. Workshop on Boolean Problems (IWSBP)*, 2016, pp. 151–158.