Non-volatile in-memory computing

Wang, Yuhao

2015

Wang, Y. (2015). Non-volatile in-memory computing. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/62147>

<https://doi.org/10.32657/10356/62147>

NANYANG
TECHNOLOGICAL
UNIVERSITY

NON-VOLATILE IN-MEMORY COMPUTING

Thesis

Submitted to the School of Electrical and Electronic Engineering
of the Nanyang Technological University

by

Wang Yuhao

for the Degree of Doctor of Philosophy

December 19, 2014

Acknowledgement

Firstly, I would like to express my sincerest gratitude to my advisor Prof. Yu Hao for giving me the opportunity to pursue my Ph.D. degree. I want to thank him for his countless support and valuable guidance throughout the years, without which I could not grow as a researcher. His profound knowledge has enlightened me to come up with many ideas; his scholarly inputs helped me greatly to improve many manuscripts of my works; his generosity on conference travel support allowed me to have the chances to engage with the best researchers around the world. He taught me to always set high standards for myself and made me realize the importance of global competence. For me, he sets a good example of what a researcher should be like: have enthusiasm towards research, always think out of box, and work hard to solve problems.

Secondly, I want to express my deepest appreciation to the faculty members, Prof. Siek Liter, Prof. Dennis Sylvester, Prof. Zhang Wei, Prof. Chang Chip Hong, Prof. Lew Wen Siang, Prof. Huang Guangbin, and Prof. Goh Wang Ling, who have given me a lot of guidance during my research. I also want to thank the Virtus, Chipes and EEE GSO staffs Dorothy, Gek Eng, David Robert, Jeremiah Chua, Karen, Christina, and many other staffs who have been really friendly and patient to help me many times on administrative matters. Moreover, I would like to express my thanks to my wonderful colleagues and group members, Yan Mei, Fei Wei, Shang Yang, Huang Xiwei, Wu Wei, Zhang Chun, Wang Kanwen, Fu Haipeng, Song Yang, Zhang Yonghao, Wang Fei, Wu Sih-Sian, P.D. Sai Manoj, Li Peng, Ma shunli, Guo Jing, Shen Shanlan, Jin Xin, Xu Dongjun, Wang Xiaolong, Chen Shuai, Huang Hantao, Yang Chang, and many other people who accompanied me to walk through my most memorable years.

Finally, I should give great appreciation to my family, my mother, my father, my grandmother and grandfather, my aunt, my uncle and my cousin for their love and endless support. And of course my lovely wife, Cai Chang, who gives me courages to tackle the difficulties and challenges. I feel happy and lucky to have them all with me.

Abstract

The analysis of big-data at exa-scale (10^{18} bytes or flops) has called for an urgent need to re-examine the existing hardware platform that can support intensive data-oriented computing. A big-data-driven application requires huge bandwidth and yet able to ensure low-power density. For example, web-searching application involves crawling, comparing, ranking, and paging of billions of web-pages with extensive memory access. The existing memory technologies have critical challenges of scaling at nano-scale due to process variation, leakage current and I/O access limitations. Recently, the emerging non-volatile memory (NVM) technologies such as resistive-RAM (ReRAM), spin-transfer torque RAM (STT-RAM), domain-wall nanowire racetrack memory etc., have all shown significantly reduced standby power and increased integration density, not forgetting the close-to DRAM/SRAM access speed. Therefore, they are considered as promising candidates of universal memory for future big-data applications.

The primary challenge to validate a hybrid design with both CMOS and nonvolatile devices is the lack of design platform that can validate the large-scale NVM circuit and system design accurately and efficiently. In addition, due to the use of non-electrical states of emerging NVM devices, new cells structures and their agreeing circuits for both read and write operations are needed to harness non-volatile memory with unique operations. For example, the transistor-free crossbar array that associates with NVM is different from conventional access transistor based memory structure. What is more, leveraging the NVM for computing, one also needs to examine the potential logic-in-memory computing architecture with significantly improved bandwidth and reduced power. In order to tackle above challenges ranging from device to system levels, this PhD thesis has explored the development of NVM design platform to support designs of non-volatile memories, readout and logic circuit designs, as well as the in-memory computing architecture.

For the NVM design platform, the target is to perform accurate yet efficient circuit level simulation. The previous approaches either ignore dynamic effect without considering non-volatile states for dynamic behavior, or need equivalent circuits with high complexity to curve-fit non-linearity of those devices. We proposed a SPICE simula-

tor named NVM-SPICE. This tool takes advantages of its new modified nodal analysis (MNA) framework, which can effectively support the non-electrical state variables of emerging non-volatile devices, such as ReRAM and spintronics devices. Due to the physics based modeling approach, NVM-SPICE is able to perform hybrid NVM/CMOS circuits efficiently and accurately. Compared to the equivalent circuit model based approach, the NVM-SPICE simulator exhibits more than 117x faster simulation speed for spintronics category devices and 40x faster speed for RRAM category devices.

For NVM in-memory architecture, both memory elements and logic elements are implemented by emerging spintronics devices, which leads to a system purely composed of non-volatile devices. The detailed non-volatile memory and logic circuits are explored within the NVM-SPICE platform. In addition, logic is built inside the memory so that the I/O workload can be alleviated. Applications such as data retention, encryption, machine learning that play critical roles for big-data computing are explored within the non-volatile in-memory architecture. The evaluation results show that the purely non-volatile memory based platforms with in-memory architecture greatly contribute to power efficiency and throughput improvement for big-data oriented applications, and thus are potential candidates to be next generation information and communication technology.

Contents

Acknowledgment	i
Abstract	iii
List of Acronyms	ix
List of Figures	xi
List of Tables	xvii
1 Introduction	1
1.1 Motivation	1
1.1.1 Non-volatile Memory	1
1.1.2 Non-volatile Computing	3
1.1.3 In-memory Architecture	3
1.2 Challenges and Contributions	4
1.3 Organization of the thesis	7
2 Fundamentals and Literature Review	9
2.1 Memory Design	9
2.2 Traditional Semiconductor Memories	12
2.2.1 Overview	12
2.2.2 Nano-scale Limitations	17
2.3 Recent Nano-scale Non-volatile Memories	20
2.3.1 Overview	20
2.3.2 NVM Design Challenges	29
2.4 Non-volatile In-memory Computing	30
2.4.1 Memory-logic-integration Architecture	30
2.4.2 Logic-in-memory Architecture	31
3 Non-volatile State Identification and NVM SPICE	35
3.1 SPICE Formulation with New Nano-scale NVM Devices	35
3.1.1 Traditional Modified Nodal Analysis	36
3.1.2 New MNA with Non-volatile State Variables	36

3.2	ReRAM Device Model	38
3.2.1	Memristor	38
3.2.2	Conductive Bridge	42
3.2.3	ReRAM Model Extension	45
3.2.4	A Case Study: ReRAM Synapse for Analog Learning	46
3.3	Spintronics Device Model	48
3.3.1	STT-MTJ	48
3.3.2	Topological Insulator	52
3.3.3	Racetrack and Domain-wall	61
3.3.4	Spintronics Model Extension	64
4	Non-volatile Circuit Design	65
4.1	Memory and Readout Circuit	65
4.1.1	Crossbar Resistive Memory	65
4.1.2	1T-1R Spintronic Memory	75
4.1.3	Domain-wall Spintronic Memory	82
4.2	Non-volatile Domain-wall Logic Circuit	87
4.2.1	XOR	87
4.2.2	Adder	90
4.2.3	Domain-wall multiplication	92
4.2.4	LUT	92
4.2.5	A Case Study: Matrix Multiplication By Domain-wall Logic	94
5	Non-volatile Memory Computing System	101
5.1	Hybrid Memory System with NVM	101
5.1.1	Overview of CBRAM based Hybrid Memory System	103
5.1.2	Block-level Incremental Data Retention	105
5.1.3	Design Space Exploration and Optimization	109
5.1.4	Performance Evaluation and Comparison	113
5.2	In-memory Computing System with NVM	115
5.2.1	General Purpose Big Data Computing	116
5.2.2	Application Specific Computing: AES Encryption	120
5.2.3	Domain Specific Computing: Machine Learning	139
6	Conclusions and Future Work	149
6.1	Conclusions	149
6.2	Recommendations for Further Work	150

A	NVM-SPICE Design Examples	153
A.1	Memristor Model Card in NVM-SPICE	153
A.2	Transient CMOS/Memristor Co-simulation Examples	155
A.3	STT-MTJ Model Card in NVM-SPICE	157
A.4	Transient CMOS/STT-MTJ Co-simulation Examples	159
B	Magnetization Physics	163
B.1	Basic Magnetization Process	163
B.2	Magnetization Damping	164
B.3	Spin-Transfer Torque	165
B.4	Magnetization Dynamics	167
B.5	Domain-wall Propagation	170
C	Publication list	171
C.1	Book	171
C.2	Tool Developed	171
C.3	Journal	171
C.4	Conference	172

List of Acronyms

NVM	non-volatile memory
DRAM	dynamic random-access memory
SRAM	static random-access memory
CBRAM	conductive bridging random-access memory
DWM	domain-wall memory
DW	domain wall
ReRAM	resistive random-access memory
MRAM	magnetic random-access memory
STT-MTJ	spin-transfer torque magnetic tunnel junction
STT-RAM	spin-transfer torque random-access memory
MTJ	magnetic tunnel junction
TI	topological insulator
PCRAM	phase change random-access memory
FeRAM	ferroelectric random-access memory
PROM	programmable read-only memory
GMR	giant magnetoresistance
CMOS	complementary metaloxidesemiconductor
SPICE	simulation program with integrated circuit emphasis
DAE	differential algebraic equation

MNA	modified nodal analysis
BSIM	Berkeley short-channel IGFET model
AES	advanced encryption standard
ALU	arithmetic logic unit
ASIC	application-specific integrated circuit
KCL	Kirchhoff's current law
KVL	Kirchhoff's voltage law
LLG	Landau-Lifshitz-Gilbert equation
LUT	look-up table
SVD	singular value decomposition
TSV	through-silicon via
XOR	exclusive or
FIFO	first in, first out
ELM	extreme learning machine

List of Figures

1.1	Typical memory read/write speed performance (latency measured in processor cycles)	2
2.1	Memory organization in H-tree network	9
2.2	The structure of memory array	10
2.3	2 to 4 decoder logic and truth table	11
2.4	Latch-type sense amplifier	11
2.5	A 6T SRAM cell structure with leakage paths in standby state. The bit-lines are precharged high and assume the stored data is ‘1’ at Q . . .	13
2.6	The circuit diagram of 1T1C DRAM cell structure	15
2.7	The cross section of a floating gate transistor	16
2.8	Two common layouts for flash memory: NOR flash memory and NAND flash memory	17
2.9	SRAM write failure	18
2.10	SRAM read failure	18
2.11	SRAM hold failure	19
2.12	Illustration of SRAM thermal runaway failure by positive feedback between temperature and leakage power	19
2.13	Memory architecture in a typical system	20
2.14	Memory hierarchy with typical volume and access cycles	21
2.15	Current status of researches on emerging non-volatile memories towards ideal universal memory	22
2.16	The relations among the fundamental elements and the prediction of the fourth element: memristor	23
2.17	STM photo of memristor array from HP Labs [8]	23
2.18	The diagram of TiO_2/TiO_{2-x} based memristor cell structure	24
2.19	The diagram of conductive bridging memory cell structure	24
2.20	The diagram of toggle MRAM cell structure	25
2.21	The diagram of spin-transfer torque based MRAM cell structure	26

2.22	The diagram of racetrack memory nanowire structure	27
2.23	(a) The cross-section of phase change memory cell; (b) Temperature profile of chalcogenide material in write operation; (c) mutual phase-change between the amorphous phase and polycrystalline phase	28
2.24	In-memory computing architecture at memory cell level	32
2.25	In-memory computing architecture at memory block level	33
3.1	New MNA with (a) component symbols and state variables; (b) large signal and (c) small signal KCL	37
3.2	Structure of memristor and nonlinear effects for dynamic model: (a) slow-down effect at boundary, (b) exponential relation between drift velocity and electric field	39
3.3	Validated memristor model with Joglekar window function for memristor hysteresis loop in [49]	39
3.4	An equivalent circuit of memristor model	41
3.5	Simulation time consumption comparison between non-volatile state variable based approach and equivalent circuit based approach for memristor	41
3.6	(a) Working mechanism of CBRAM with the shape morphing of conductive filaments in several phases illustrated between ON-state and OFF-state; (b) cross section of CBRAM device with defined geometric variables	42
3.7	CBRAM model validation against the published measurement data [16, 26]	45
3.8	Transient response of CBRAM device Set and Reset when v_w amplitude of 1.8V is applied	46
3.9	A simple hybrid CMOS-memristor circuit to model the dynamic conditioning behavior	47
3.10	The simulation results of the hybrid CMOS-memristor design for simple classic conditioning	48
3.11	Spherical coordinates with two magnetization angles: θ and ϕ	50
3.12	Equivalent circuit of STT-MTJ model	51
3.13	(a) Device structure; (b) schematic diagram of quantum Hall conductance; and (c) abstracted equivalent device circuit model	53
3.14	Cell circuit of topological insulator based memory	55
3.15	One 4×4 topological insulator based NVM array	56
3.16	Validation of the switching time and external magnetic field relationship for magnetization dynamics	58

3.17	Dynamic response of topological insulator new state variables	59
3.18	Read operation for topological insulator based memory cell	60
3.19	Timing diagram for word write and read operations in a 4×4 TI array .	60
3.20	(a) Schematic of domain-wall nanowire structure with access port and shift port; (b) magnetization of free-layer in spherical coordinates with defined magnetization angles; and (c) typical R-V curve for MTJ	62
4.1	Crossbar operations and peripheral circuits	66
4.2	Two approaches for 3D stacked CBRAM-crossbar fabrication (a) crossbar structure integrated within interconnect layers, and CBRAM devices fabricated at the bottom of vias (b) crossbar structure fabricated on the top of interconnect layers	67
4.3	Conventional crossbar read operation with incurred sneak-path issue . .	67
4.4	Readout circuit design for CBRAM crossbar structured memory	68
4.5	Monte Carlo simulation of 100×100 crossbar-array with device resistance variations	69
4.6	RC-delay model of one CBRAM-crossbar for: (a) read operation; (b) write operation	71
4.7	Verification of proposed CBRAM-crossbar specific wire delay model against simulation results, with fitting parameter $\alpha = 1.2$	73
4.8	Verification of proposed CBRAM-crossbar specific power model for read/write operations against simulation results	74
4.9	Circuit diagrams of 1T-1R STT-RAM memory cell: (a) simplified memory cell for write 1; (b) simplified memory cell for write 0; and (c) 16-bit STT-RAM with 4 bit-lines and 4 word-lines	76
4.10	The existing schemes for STT-RAM readout: (a) basic STT-RAM readout (b) destructive self-reference readout in [99] (c) non-destructive self-reference readout in [105]	78
4.11	The measured R-I sweep curve of a typical MgO based MTJ in [99] . .	79
4.12	Diagram of proposed single-sawtooth pulse based readout	80
4.13	Transient response of bit-line voltage, first derivative, and second derivative to the applied sawtooth pulse for pure OPAMP, hybrid and pure RC based circuit, respectively	81
4.14	Macro-cell of DWM with: (a) single access-port; and (b) multiple access-ports	83
4.15	Sensing circuit design for domain-wall nanowire	85
4.16	Low power XOR-logic implemented by two domain-wall nanowires . .	88

4.17	The timing diagram of DWL-XOR with SPICE-level simulation for each operation	89
4.18	The carry out logic achieved by domain-wall nanowires	90
4.19	The sum logic achieved by domain-wall nanowires	91
4.20	The NVM-SPICE simulation results for carry logic (a) $A = 1$, $B=1$, and $C_{in}=1$; (b) $A=0$, $B=1$, and $C_{in}=0$	92
4.21	(a) Domain-wall memory cell structure; (b) LUT by domain-wall nanowire array with parallel output and serial output	93
4.22	Power characterization for DW-LUT in different sizes	94
4.23	System overview of the proposed NVM-based logic-in-memory computing platform	95
4.24	Matrix multiplication mapping to proposed domain-wall nanowire based computing platform	97
5.1	(a) Conventional computing system power profile; (b) ideal instant on/off computing system power profile	102
5.2	Data retention for leakage reduction by (a) drowsy memory; (b) data back-up and recovery	103
5.3	More power efficient instant on/off computing by incremental back-up as well as 3D CBRAM-crossbar memory	104
5.4	3D hybrid memory system with CBRAM-crossbar based data retention	104
5.5	Circuit diagram for dirty bit set-up at active mode	107
5.6	Circuit diagram for dirty-data write-back at hibernating transition	108
5.7	Hibernating power and time reduction by incremental dirty-data write-back	109
5.8	Design space exploration of 3D hybrid memory with CBRAM-crossbar	111
5.9	(a) TSV density and (b) mode transition power under different architecture-level parameters	112
5.10	The overview of in-memory architecture with distributed memory for data server	115
5.11	The overview of the general purpose big data computing platform by domain-wall nanowire devices	116
5.12	(a) The runtime dynamic power of both DRAM and DWM under Phoenix and SPEC2006 ;(b) the normalized intended memory accesses	119
5.13	The per core ALU power comparison between CMOS design and DW-logic based design	119
5.14	The flow chart of AES algorithm with gate utilization analysis	121

5.15	Data organization of state matrix by domain-wall nanowire devices in distributed manner	124
5.16	SubBytes step with S-box function achieved by domain-wall memory based look-up table	125
5.17	ShiftRows transformation by domain-wall nanowire shift operations . .	127
5.18	AddRoundKey step with XOR logic achieved by domain-wall nanowire	128
5.19	MixColumns transformation by DW-LUT and DW-XOR	129
5.20	(a)DW-AES without pipeline;(b) Pipelined DW-AES by inserting DW-FIFO ;(c) Stages balancing by the cycles delay of DW-FIFO	131
5.21	Example (a) timing diagram and (b) block diagram of pipelined DW-AES with multi-issue	133
5.22	The breakdown of (a) area (b) latency (c) dynamic power (d) leakage power of DW-AES, pipelined DW-AES and multi-issued DW-AES . . .	136
5.23	In-memory encryption throughput, power and energy efficiency comparisons between different AES platforms	137
5.24	The working flow of extreme learning machine	141
5.25	The overview of the in-memory computing architecture	142
5.26	Detailed in-memory domain-wall nanowire based machine learning platform in Map-Reduce fashion	143
5.27	Domain-wall nanowire based full adder with sum operation by DW-XOR logic and carry operation by resistor comparator	144
5.28	(a) Sigmoid function implemented by domain-wall nanowire based look-up table (DW-LUT); (b) DW-LUT size effect on the precision of the sigmoid function	146
5.29	(a) Original image before ELM-SR algorithm (SSIM value is 0.91); (b) Image quality improved after ELM-SR algorithm by DW-NN hardware implementation (SSIM value is 0.94); (c) Image quality improved by GPP platform (SSIM value is 0.97)	147
A.1	1T1R structure for memristor device based memory cell	155
A.2	Dynmaics of doping ratio in memristor set operation under transient analysis	156
A.3	Plot of time-varying resistance of memristor for verification	157
A.4	(a) STT-MTJ writing circuit in 1T1R memory cell structure and (b) STT-MTJ sensing circuit by resistance comparator	159
A.5	Plot of time-varying internal state theta of STT-MTJ	160
A.6	Plot of time-varying resistance of STT-MTJ	160

A.7	Switch of STT-MTJ under pulse signal	161
A.8	Simulation results for STT-MTJ sensing circuit as shown in Figure A.4(b)	162
B.1	The magnetization precession.	164
B.2	The magnetization precession with damping.	165
B.3	The spin-transfer torque effect	166

List of Tables

3.1	Simulation time comparison for STT-RAM array using different simulation approaches (unit in second)	51
3.2	Notation for topological insulator device modeling	52
3.3	Performance comparison for different memory technologies	60
4.1	Device resistance variation caused sense amplifier failure rate for different crossbar-array sizes	70
4.2	Performance comparison of 16MB SRAM, DRAM, PCRAM, and CBRAM memories	76
4.3	Comparison of different readout schemes for STT-RAMs	82
4.4	Performance comparison of 128MB memory-bank implemented by different structures	87
4.5	Platform comparison	98
5.1	Notations	105
5.2	Optimized performance for different design objectives	112
5.3	Data-retention performance comparison for different leakage reduction schemes	113
5.4	Cache performance comparison between block-level data retention (CBRAM) and bit-level data retention (FeRAM) in active mode	114
5.5	System configuration	118
5.6	Notation for domain-wall device based DW-AES implementation	123
5.7	AES for 128 bits encryption performance comparisons	135
5.8	System configurations	138
5.9	Area, Power, Throughput and Energy Efficiency Comparison between In-Memory Architecture and Conventional Architecture for ELM-SR	145
A.1	The list of supported parameters for nonlinear dynamic memristor model	154
A.2	The list of supported parameters for STT-MTJ model	158

Chapter 1

Introduction

1.1 Motivation

As reported by IBM [1], there will be 2.5 exa-bytes (10^{18}) of data created everyday in the year 2012, and the volume is doubling every 40 months [2]. The big-data analytics is critically important for many emerging technologies such as artificial intelligence and machine learning. The analysis of big-data at exa-scale (10^{18} bytes or flops) has introduced emerging need to re-examine the existing hardware platform that can support intensive data-oriented computing. A big-data-driven application requires huge bandwidth and yet ensures high energy efficiency. For example, web-searching application involves crawling, comparing, ranking, and paging of billions of web-pages with extensive memory access. At the same time, the analysis of such a huge data at exa-scale is national interest due to cyber-security need. One needs to provide scalable big-data storage and processing solution that can detect malicious attack from the sea of data, which is beyond the capability from a pure software based data analytic solution. The key bottleneck is from the current data storage and processing hardware, which has the well-known memory-wall and power-wall with limited accessing bandwidth but also large leakage power at advanced CMOS technology nodes. One needs to design an energy-efficient hardware platform for future big-data storage that can also support data-intensive processing. In this part, the challenges and potential solutions will be discussed from three aspects: memory, logic and memory-logic-integration.

1.1.1 Non-volatile Memory

Memory is any physical device that is able to temporarily or permanently hold the state of information. All well-established memory technologies introduced above have certain limitations. Static random-access memory (SRAM) is the fastest memory technology

Figure 1.1: Typical memory read/write speed performance (latency measured in processor cycles)

currently available, but significant leakage power is experienced to retain the stored data, which gets worsen when scaled down to deep-submicron regime. In addition, its capacity is limited to a few megabytes due to its high cost led by area-consuming 6-transistor structure. Dynamic random-access memory (DRAM) is second to SRAM in terms of speed, but this capacitor based memory needs to be refreshed periodically, which produces large power consumption for DRAM in large capacity; Flash memory overcomes the power issue by its non-volatility, but has slower speed and very limited endurance; thus it mainly serves as storage for data that does not need to be frequently accessed.

Imagine how life would be if computer can start in the blink of an eye, without having to wait for the operating system to load, or to transfer full-length high-definition movie using memory stick in seconds rather than hours. These can materialize if a universal non-volatile memory (NVM) can retain information without needing an external power source and be accessed in high speed. In general, the following criteria characterize the new memory technologies:

- Scalability for high-density integration;
- Low energy consumption for mobile access;
- High endurance capable of 10^{12} writing/erasing cycles.

Featured with fast access speed, high density and zero standby power, the emerging

NVM at nano-scale such as spin-transfer-torque (STT-RAM) [3, 4], phase-change (PC-RAM)[5, 6], resistive-RAM (Re-RAM)[7, 8], and domain-wall (DW-RAM)[9, 10], have introduced promising future for universal memory in big-data computing platforms. The speed performance that compares conventional memories and emerging non-volatile memories is shown in Figure 1.1.

The tremendous advantages over the currently prevailing SRAM/DRAM/Flash devices have made the new nano-scale NVMs not just candidates for next-generation massive storage, but to replace DRAM and SRAM in the computing systems. For example, STT-RAM has exhibited the great potential due to its fast speed (< 10 ns), high integration density ($6\sim 8 F^2$, where F is the feature size), and virtually unlimited endurance ($> 10^{15}$).

1.1.2 Non-volatile Computing

Current logic circuits are all based on CMOS gates, which suffer significant leakage power at the deep sub-micro regime. Similar to the idea of non-volatile memory, it would be beneficial (power-wise) to conduct computing in non-volatile fashion by exploiting the non-volatile devices. The use of NVM devices for computing will also make it very interesting that the whole system could be built purely based on non-volatile memory devices, both the logic and storage.

Due to some unique physical effects of emerging non-volatile memories, this is actually possible by exploiting some physical effects of non-volatile memory devices, which conventional memory technologies do not possess. By building an unified non-volatile computing platform, where both the memory and computing resources are based on NVM devices with instant-switch-on as well as to ultra-low leakage current, low power and high throughput (or energy efficient) for big-data computing can be achieved. This can result in significant power reduction due to the non-volatility.

1.1.3 In-memory Architecture

Conventionally, all data is maintained within a memory that is separated from the processor and they are connected by the I/O bus. Therefore, during data execution, all information needs to migrate to the processor and be written back afterwards. In data-oriented applications, however, this will incur significant I/O congestions, which greatly degrades the overall performance.

Theoretically, it is feasible to overcome the bandwidth issue by adding more I/O pins or operating at higher frequency. Practically, however, the I/O frequency is limited by the signal propagation delay and signal integrity issues, and I/O number is limited

by the packaging technology, thus the bandwidth can hardly be improved further. Instead of improving on the memory bandwidth, it is possible to reduce the required data communication traffic between memory and processor. The basic idea behind is that, instead of feeding processor large volume of raw data, it is beneficial to pre-process the data and provide processor only intermediate result. The key to lower communication traffic is the operands reduction. For example, to perform a sum of ten number, instead of transmitting ten number to processor, in-memory architecture is able to obtain the sum by in-memory logic and transmit only one result thus reduce traffic by 90%. To perform in-memory logic, it is necessary to implement logic inside memory so that pre-processing logic can be done. Note that if the logic can be implemented in a non-volatile fashion as mentioned above, the logic-memory-integration inside the memory could be easily achieved. Such logic-in-memory architecture is one promising solution for both the memory-wall and power-wall issues for big-data computing.

1.2 Challenges and Contributions

Although the non-volatile in-memory computing seems to be a promising solution for future big-data storage and processing system, the emerging nano-scale NVM technologies are still in their infancy due to the following unresolved challenges:

- From device level, it lacks simulator to perform hybrid NVM/CMOS circuit simulation as the NVM devices are not supported in current SPICE simulator. Without accurate geometry related SPICE model built for NVM devices, it is hard and inefficient to further validate any design that contains NVM devices.
- From circuit level, it is necessary to explore memory and logic design based on emerging NVM devices. Conventional memory designs such as cell structure and sensing circuit may not work for emerging NVM or not optimal for NVM. Logic side, their computing ability brought by unique physical effects have not been well studied.
- From system level, big-data orientated architecture with NVM has not been explored. The in-memory computing architecture for NVM needs to be designed with concrete examples.

This PhD thesis has explored the development of NVM design platform to support designs of non-volatile memories, readout and logic circuit designs, as well as the in-memory computing architecture. The main contributions of this thesis are listed as follows.

For the NVM design platform, the target is to perform accurate yet efficient circuit level simulation. The previous approaches either ignore dynamic effect without considering non-volatile states for dynamic behavior, or need equivalent circuits with high complexity to curve-fit the non-linearity of NVM devices. We proposed a new modified nodal analysis for NVM devices with identified non-electrical state variables for dynamic behavior. In details, a compact SPICE-like implementation (NVM-SPICE) has been developed for the new NVM devices in the hybrid NVM/CMOS designs. Particularly, this PhD thesis has summarized the following works:

- Firstly, a memristor SPICE simulator is introduced based on new modified nodal analysis (MNA) framework, which effectively supports the non-conventional state variable like doping ratio for memristor. Therefore, similar to BSIM MOSFET model, the intrinsic memristor model can be stamped into state matrix and solved by Newton iteration. Compared with equivalent circuit based approach, the new MNA based approach exhibits 40x faster simulation speed for a 32×32 memristor crossbar circuit. A hybrid CMOS memristor circuit for classic conditioning training has also been explored within the developed SPICE simulator.
- Secondly, a spintronics SPICE simulator is developed for accurate and efficient simulator of spin-transfer torque magnetic tunneling junction (STT-MTJ) with all-physics-based-model. Compared to the equivalent circuit model based approach, the proposed simulator exhibits more than 117x faster simulation speed for large-array STT-RAM (64×64 array and above).

For NVM circuit designs, the target is to develop high performance and low power memory structures as well as corresponding memory cell read and write circuits that best exploit the uniqueness of new NVM devices. Particularly, this PhD thesis has summarized the following works:

- A single-saw-tooth pulse based NVM readout is proposed for STT-RAM, which is able to achieve reliable readout in the presence of large MTJ resistance variation. This scheme is nondestructive and is able to reduce the read latency into one cycle. Validated by the NVM-SPICE, the proposed scheme achieves about 2X faster read latency with similar sensing margin, or 8X larger sensing margin when compared to the existing approaches.
- The domain-wall nanowire based NVM logic is explored towards NVM computing. The shift-operation in domain-wall nanowire has been adapted to perform logic operations such as XOR-logic for comparison as well as addition, which are

dominant in web-searching. Therefore, one can build a highly logic-in-memory computing platform with both memory and logic implemented by domain-wall nanowire devices. We find that the proposed platform can reduce 92% leakage power and 16% dynamic power in terms of NVM memory; and can also reduce 31% both dynamic and 65% leakage power in terms of NVM logic.

For NVM based in-memory computing, the applications such as data retention, encryption, machine learning that play critical roles for big-data computing are implemented purely by non-volatile memory at block level. Particularly, this PhD thesis has summarized the following works:

- One 3D-integrated hybrid memory is designed for data retention in data storage. By stacking one tier of CBRAM-crossbar with tiers of SRAM and DRAM, the tier of CBRAM-crossbar is deployed for data retention during power gating of SRAM/DRAM tiers. One corresponding block level data retention is developed to only write back dirty data from SRAM/DRAM to CBRAM-crossbar. When compared to phase-change random-access-memory (PCRAM) based system level data retention, our design achieves 11x faster data-migration speed and 10x less data-migration power. When compared to ferroelectric random-access-memory (FeRAM) based bit-level data retention, our design also achieves 17x smaller area and 56x smaller power under the same data-migration speed.
- The domain-wall nanowire device is explored for in-memory AES (DW-AES) computing. All AES operations can be fully mapped to exploit the unique properties of this emerging technology. For example, the DW-XOR logic is proposed for the dominant XOR operation; the domain-wall shift is exploited for the row-shift operation; and the DW-LUT is utilized for the S-box operation. The experiment results show that, the proposed DW-AES exhibits the best energy efficiency (24 pJ/bit), which is 9X and 6.5X times better than CMOS ASIC and memristive CMOL implementations, respectively. In addition, it has 6.4X higher throughput and 29% power saving compared to a CMOS ASIC implementation.
- The domain-wall nanowire device is explored for in-memory machine learning neural network, called DW-NN, is studied. In the proposed DW-NN, domain-wall nanowire based logic customized for machine learning is integrated within the image storage data such that machine-learning based image processing can be performed locally within the memory. We show that all operations involved in machine learning on neural network can be mapped to a logic-in-memory architecture by non-volatile domain-wall nanowire. The experimental results show

that the I/O load in the proposed DW-NN is greatly alleviated with an energy efficiency improvement by 56x and throughput improvement by 11.6x compared to the conventional image processing system by general purpose processor.

1.3 Organization of the thesis

This thesis covers the entire design flow from device, circuit to system perspectives for emerging NVMs, which can be organized into the following chapters. Chapter 2 covers the basics of memory, review of existing memory technologies and emerging NVM technologies, and also review of in-memory computing architecture. Chapter 3 details the device characterization for the emerging NVM by non-electrical states, and the NVM-SPICE implementation. Chapter 4 explores the circuit level design techniques for both the NVM and its non-volatile logic. Chapter 5 presents the system level non-volatile computing architectures with applications such as data-retention, encryption and machine learning. Finally, Chapter 6 concludes the thesis, and shows some potential topics as future work.

Chapter 2

Fundamentals and Literature Review

2.1 Memory Design

Figure 2.1: Memory organization in H-tree network

Before introducing specific memory technologies, it is important to understand the basic electronic components of which the memory is made up of. A memory chip consists of millions to billions of memory cells, and takes binary address as input and finds target cells correspondingly, so that read and write operations can be performed. In order to efficiently perform this, memory cells are organized in certain fashion, as illustrated in Fig. 2.1.

Enormous data cells are divided into multiple data arrays, which are connected by H-tree network. The input address is logically divided into two parts, and will be interpreted respectively. First part of the address indicates the position of the data array, in

which the target cells are kept. Second part of the address reveals the position of the target cells inside the data array. The data array identifier will be used by the predecoders along the H-tree network paths, to route electrical signals to target data array. The most noticeable advantage of H-tree network is that it can ensure an identical signal propagating distance to every data array. This is important to ensure the system is deterministic and the access latency is fixed.

Figure 2.2: The structure of memory array

The storage unit in the memory is the data array, whose structure is shown in Fig. 2.2. All memory cells lie at the crosspoints of the word-lines and bit-lines. Word-lines and bit-lines are metal wires that propagate signals and will incur certain wire delay due to its parasitic wire resistance and capacitance; therefore the larger the data array is, the longer access latency can be expected. Each cell stores one bit of information, represented by high or low logic, and its value can be read out by sense amplifiers. If every single cell is directly connect with outside I/O, billions of I/Os will be required which is practically impossible to achieve; therefore the decoders are used to take binary address to operate on designated cells.

A decoder converts binary address from n input bits to a maximum of 2^n unique output lines. The circuit of a 2-4 decoder with its truth table is shown in Fig. 2.3. In the memory array, the output lines of word-line decoders are connected to the word-

Figure 2.3: 2 to 4 decoder logic and truth table

lines, which enable an entire row of data array specified by the address. Because the electrical signals are bidirectional on the bit-line, that is, bit-line can drive cell in write operation and cell can drive bit-line in read operation, the bit-line decoder output lines are connected to the multiplexer, which selectively allows electrical signal of specific column to pass.

Figure 2.4: Latch-type sense amplifier

The objective of the readout circuit is to distinguish the bistable states of the memory cells. For conventional memory technologies like static random-access memory (SRAM) and dynamic random-access memory (DRAM), the memory state is represented by electrical voltage: V_{DD} for logic 1 and GND for logic 0. Therefore, a readout circuit can be designed as a voltage comparator that compares state voltage with intermediate voltage, namely $V_{DD}/2$. However in practice, due to the charge sharing between memory cell and bit-line parasitic capacitor, the detectable voltage margin is reduced to less than one-tenth of previous value, more sensitive readout circuit is required. Figure 2.4 shows the circuit of a voltage mode typical latch-type sense amplifier, which is able to detect voltage difference smaller than 100mV at nanosecond scale. Its working mech-

anism can be described as follows. At first, the BLP and BLN are precharged at $V_{DD}/2$ and EN is kept low. When the readout is performed, the BLP voltage will increase or decrease slightly (ΔV) due to charge share. If differential structure is used, the BLN will change $-\Delta V$. Note that the ΔV is determined by the ratio of memory cell capacitance and bit-line capacitance, and when the array is too large, ΔV will become too small to detect. This is the major reason why the memory cannot be made into one single array and the division is required. After that, the sense amplifier is enabled by set EN signal high. This isolates the BLP and BLN with Q and QB , and Q and QB will be compared by the latch. The latch compares voltages in a positive feedback fashion. Assume there is a ΔV_Q , and as the input of inverter at right hand side ($M3$ and $M4$) it will lower the voltage of V_{QB} , which can be seen in the transfer curve of inverter. And as the input of inverter at left hand side ($M1$ and $M2$), the decrease in V_{QB} will in turn increase the value of ΔV_Q . As such, the two cross-coupled inverters reinforce each other and enter a positive feedback loop until they reach the final stable state that the ΔV_Q is V_{DD} and V_{QB} is 0. Without the pass transistor $M5$ and $M6$, the latch will have to drive the entire bit-line, which greatly affects the convergence speed and incur more energy consumption.

2.2 Traditional Semiconductor Memories

2.2.1 Overview

Semiconductor memories refer to the silicon-transistors based physical devices in computer systems that are used to temporarily or permanently store programs or data. In contrast to storage media like hard disks and optical discs, of which the accesses have to follow a predetermined order due to the mechanical drive limitations, semiconductor memories possess the property of random access, which means that the time it takes to access any data is identical regardless of the data location.

According to volatility, the semiconductor memories can be further classified into volatile memory and non-volatile memory. Volatile memory stores data as electrical signals, and loses its data when the chip is powered off. Currently, the most commonly used volatile memories are static random-access memory (SRAM) and dynamic random-access memory (DRAM), whose data is indicated by the electrical voltage levels. On the contrary, non-volatile memory is able to retain its data even when chip is turned off, as its data is mostly preserved by non-electrical states. For instance, bits in programmable read-only memory (PROM) are denoted by whether the fuses of individual memory cells are burned. In this part, the principles and operations of volatile SRAM/DRAM, and non-volatile flash memory will be briefly introduced.

2.2.1.1 Static Random-access Memory

Figure 2.5: A 6T SRAM cell structure with leakage paths in standby state. The bit-lines are precharged high and assume the stored data is ‘1’ at Q

A typical CMOS static random-access memory (SRAM) cell consists six transistors, shown in Fig. 2.5. The flip-flop formed by M1 to M4 holds the stored bit. The term *static* is derived from the fact that the cell does not need to be refreshed like dynamic RAM, and the data can be retained as long as the power is supplied. The M5 and M6, connected with word-line and two bit-lines, are used as access transistors to select target cells.

There are three operation states for a SRAM cell, write, read and standby. To a write operation, the value to be written needs to be applied on both bit-lines, namely BL and $\overline{BL}$, in a complementary manner. Assume we wish to write ‘0’ to the cell, i.e. Q to be ‘0’ and $\overline{Q}$ to be ‘1’, the BL is driven low and $\overline{BL}$ high. Once the M5 and M6 are turned on by setting WL ‘1’, the bit-line drivers will override the previous stored value. In order to easily override the previous state in the self-reinforced flip-flop, the bit-line drivers are required to be designed stronger than the transistors in flip-flop.

For a read operation, both bit-lines are precharged high before the start of the read cycle, and the turning on of the word-line signifies the start of read operation. Because of the opposite voltages at Q and $\overline{Q}$, one of the bit-lines will be pulled down by the cell, and the discharging of one of the bit-lines is then detected by the bit-line sense amplifier. In voltage mode sensing scheme, the sign of bit-line voltage difference ΔV (V_{BL} minus $V_{\overline{BL}}$) determines the value of stored bit. A ΔV in tens of millivolts is significant enough to efficiently distinguish which bit-line is being discharged. For example, assume the stored bit is ‘1’ at Q and ‘0’ at $\overline{Q}$, and once the WL is asserted, the $\overline{BL}$ will discharge towards ‘0’; when a positive ΔV of tens of millivolts is gained, the latch based sense

amplifier will amplify the small voltage difference with positive feedback, and finally output logic '1' as result.

When the word-line (WL) is connected to ground, turning off the two access transistors, the cell is in the standby state. During the standby state, the two cross-coupled inverters will reinforce each other due to the positive feedback, the value is preserved as long as the power is supplied. One prominent problem regarding SRAM in standby state is severe subthreshold leakage. Subthreshold leakage is the current between drain and source of a transistor when the gate-to-source voltage is less than the threshold voltage, namely in the subthreshold region or weak-inversion region. The subthreshold current depends exponentially on threshold voltage, which results in large subthreshold current in deep sub-micron regime. Figure 2.5 shows three leakage paths in one SRAM cell, assuming the stored bit is '1' at Q . Note that the BL and $\overline{BL}$ are always precharged to V_{DD} to facilitate future read operation. Regardless of the stored value, there always will be three transistors consuming leakage power.

Compared to other memory technologies, SRAM is able to provide fastest access speed, but the advantage comes as a tradeoff against density and power. As one SRAM cell requires silicon area for six transistors, SRAM has very limited density and hence is more expensive than other memory technologies. In addition, it is very power consuming due to the leakage problem at standby state. Therefore, SRAM serves best in the applications where high performance is the main concern and the capacity is not significant, namely the caches for processors.

2.2.1.2 Dynamic Random-access Memory

The philosophy behind DRAM is simplicity. Unlike SRAM where one cell is composed of six transistors, each individual DRAM cell consists only one capacitor and one access transistor. The data '0' or '1' is represented by whether the capacitor is fully charged or discharged. However, the electrical charge on the capacitor will gradually leak away, and after a period of time, the voltage on the capacitor is so low for the sense amplifier to differentiate between '1' and '0'. Therefore, unlike SRAM that the data can be retained as long as the power is supplied, the retention time for DRAM is finite and all DRAM cells need to be read out and written back periodically to ensure data integrity. Typically, the cells are refreshed once every 32ms or 64ms. This process is known as *refresh*, and this is how the name of dynamic RAM is derived. Figure 2.6 shows the circuit diagram of such 1T1C structure DRAM cell.

To write a DRAM cell, the bit-line is first set high or low based on the value to write. After the access transistor is turned on by asserting word-line, the capacitor in

Figure 2.6: The circuit diagram of 1T1C DRAM cell structure

the selected cell is charged to ‘1’ or discharged to ‘0’. Because the access takes place through an NMOS transistor, there exists a V_{th} drop during the write ‘1’. In order to prevent this V_{th} drop and maintain a long refresh period, the word-lines driving voltage is usually boosted to $V_{PP} = V_{DD} + V_{th}$.

To read a DRAM cell, the bit-line is precharged to $V_{DD}/2$ and then the word-line is enabled. Due to the charge sharing, the bit-line voltage will slightly decrease or increase depending on the voltage that the capacitor was previously charged to, i.e. V_{DD} or 0. If it is previously charged, the charge sharing will slightly boost the bit-line voltage; otherwise, some charge will be distributed from bit-line to cell capacitor. In both cases, the voltage of storage capacitor will be changed after read operation, thus the read operation is called destructive, and an instant write back is required. A slight voltage change on the bit-line can be calculated by

$$\Delta V = \pm \frac{V_{DD}}{2} \frac{C_{cell}}{C_{cell} + C_{bitline}} \quad (2.1)$$

The sign of ΔV depends on the state of storage capacitor. For typical DRAM devices, the capacitance of a storage capacitor is far smaller than the capacitance of the bitline. It is reported that the capacitance of a storage capacitor is about one-tenth of the capacitance of bitline [11]. In this low relative capacitance ratio scenario, the change of charge in DRAM cell will only result in subtle voltage level change on the bitline, which is difficult to measure in an absolute sense. Current solution exploits the use of a differential sense amplifier that compares the voltage of the bitline to a reference voltage.

2.2.1.3 Flash Non-volatile Memory

Flash memory is the most widely used non-volatile memory technology today. The key device in this prevailing memory is floating gate transistors. A figure of cross section of a floating gate transistor is shown in Fig. 2.7. Unlike a MOSFET transistor, an additional floating gate is added between the control gate and channel. Isolated by oxide layers,

floating gate is able to trap charges and keep them for years. Therefore, the FG-transistor encodes data based on whether electrons are trapped, and is able to retain data without power. That is where ‘non-volatile’ is derived from.

The principle of read operation can be described as follow. When no charges are trapped in floating gate, the FG-transistor has a threshold voltage of V_{th0} ; when negative charges are trapped, they attract positive charges of control gate, thus higher control gate voltage is required to turn on the channel, which produces a higher threshold voltage V_{th1} . By applying intermediate control gate voltage that is between V_{th0} and V_{th1} , and measuring current, the state of device can be known.

Figure 2.7: The cross section of a floating gate transistor

The write operation of FG-transistor involves injecting or pulling electrons across the oxide barrier. There are two ways to achieve this, quantum tunneling or hot electron injection. In quantum tunneling scenario, high voltage is applied on control gate, quantum tunneling will take place between the floating gate and channel, and electrons can travel across the oxide barrier. For hot electron injection scenario, electrons are accelerated under high electrical field in the channel till its energy is high enough to penetrate the oxide layer. Note that electrons with high energy will damage the oxide lattice, and such damage will accumulate and lead to a limited write cycles, which is typically around 10^5 cycles.

There are two common layouts for flash memory, shown in Fig. 2.8, NAND flash memory with FG-transistors in series and NOR flash memory with FG-transistors in parallel. The name NAND and NOR are derived from the fact that their connection fashion in series or parallel resemble a NAND gate or NOR gate. NAND layout has the density advantage over NOR layout because each row only has one ground connection, thus is widely used for external storage. NOR layout has lower latency thus is widely used in embedded systems, where high performance is required. Fig. 2.8 also shows how the read of NAND and NOR flash memory can be achieved. The relationship between the different applied voltage magnitude is shown as follows:

$$V_{OFF} < V_{th0} < V_{INT} < V_{th1} < V_{ON} \ll |V_{HIGH}| \quad (2.2)$$

Figure 2.8: Two common layouts for flash memory: NOR flash memory and NAND flash memory

2.2.2 Nano-scale Limitations

As the technology scaling advances into nano-scale, the conventional memory design faces certain limitations. In the regime where the classic physics still rules, such limitations mainly come from two major aspects. Firstly, due to process variation, the mismatch among transistors may lead to functional failures. Secondly, positive feedback loop between leakage power and heat may result in thermal runaway failure. In this part, we review the physical mechanisms of such failures induced in nano-scale designs,

2.2.2.1 Functional Failures by Variation

The work in [12] well illustrated the SRAM functional failures caused by threshold variation in nano-scale regime. The write, read and hold failures are illustrated in Fig. 2.9 to 2.11. The SRAM states (cross-coupled inverters) and operations are represented by a variable plane, with two stable equilibrium states either at top-left or bottom-right corner. There are two convergent regions that are divided by a crossing line, named *separatrix*. Given enough time, operating point in any region will converge to the nearest stable equilibrium state.

Write Failure The write operation aims to pull the operating point in equilibrium state to cross the *separatrix*, so that it converges to the closest equilibrium state after write operation, as shown by point B in Fig. 2.9. A change in threshold voltage due to variation, however, alters the transistor driving strength, which may lead to write failure.

Figure 2.9: SRAM write failure

A write failure occurs when the operating point cannot cross the *separatrix* before access transistors are turned off. In this case, it will go back to the initial state instead of target state. For example, an increase of V_{th} in M6 along with the decrease of V_{th} in M4 can result in difficulty to pull down v_2 . On the variable plane, it becomes more difficult for operating point to move towards the target state.

Read Failure Before the read operation, the bit-lines BR and BL are pre-charged high. During the read operation, the bit-line will charge “0” side of SRAM, i.e. operating point pulled towards the *separatrix* when reflected on the variable plane. A successful read will ensure access transistor shut before operating point pulled beyond the *separatrix*. However, if there exists mismatch caused by variation, for example between M4 and M6, unbalanced pulling strength will make v_2 can be pulled up more quickly, which may result in operating point cross the *separatrix*, as point B in Fig. 2.10.

Figure 2.10: SRAM read failure

Hold Failure During the standby state of SRAM, external noise leads to perturbation of operating points. Unless noise is large enough, the operation points are not likely to cross the *separatrix* without variation. However, with threshold variation of M1-4 present, it leads to the shift of the *separatrix*, and together with the noise level will determine the likelihood of hold failure, as shown in Fig. 2.11.

Figure 2.11: SRAM hold failure

2.2.2.2 Functional Failure by Thermal Runaway

Thermal runaway [13, 14] can be defined as a rise in temperature causes certain conditions to change in a way that further increases temperature in a positive feedback fashion. In memory system, it is associated with the electrical-thermal coupling between leakage power and temperature.

Figure 2.12: Illustration of SRAM thermal runaway failure by positive feedback between temperature and leakage power

As the technology node scales down, the controlling ability of transistors becomes weaker, and hence larger leakage current will be experienced. As such, thermal runaway becomes a prominent limitation for large-scale big-data memory integration in advanced technology nodes. The course of potential thermal runaway is illustrated of

the in Fig. 2.12. At the very beginning, memory works at room temperature with moderate leakage current, which will consistently produce heat. If the thermal source grows much faster than heat removal ability of heat-sink, there will be thermal accumulation and lead to temperature increase of memory chip. Due to the exponential relationship between temperature and leakage current, the increase of memory temperature will in turn provoke larger leakage current, which in turns increases the leakage current. Such uncontrolled positive feedback will continue and finally lead to the destructive high temperature, melting silicon and permanently damaging the memory cells.

2.3 Recent Nano-scale Non-volatile Memories

2.3.1 Overview

All well-established memory technologies introduced above have certain limitations. SRAM is the fastest memory technology currently available, but significant leakage power is experienced to retain the stored data, which gets worsen when scaled down to deep-submicron regime. In addition, its capacity is limited to a few megabytes due to its high cost led by area-consuming 6-transistor structure. DRAM is second to SRAM in terms of speed, but this capacitor based memory needs to be refreshed periodically, which produces large power consumption for DRAM in large capacity; Flash memory overcomes the power issue by its nonvolatility, but has slower speed and very limited endurance; thus it mainly serves as storage for data that does not need to be frequently accessed.

Figure 2.13: Memory architecture in a typical system

In order to exploit the strengths and avoid the drawbacks of different memories, the

Figure 2.14: Memory hierarchy with typical volume and access cycles

memory hierarchy has been formed based on the roles the memories play (Fig. 2.13). Generally, memories at the bottom of the pyramid (Fig. 2.14) tend to be slow, less costly, and non-volatile; and memories at the top of the pyramid tend to be fast, expensive, and power hungry. Such memory hierarchy has been proven very efficient for decades, but by no means to be the final solution for memory system design. The challenge that current memory system faces is the well-known memory wall issue, meaning that the memory performance accelerates at a much lower rate than that of processors. Therefore, memory usually becomes the bottleneck of the overall computing system. The inefficiency and complexity of data transmitting vertically among different levels of memory hierarchy is the de-facto hinder to bridge the performance gap between memory and processor. For example, the retrieve latency of HDD is several orders longer than that of main memory, and in cases that the data to be executed has not yet been loaded into the main memory, there will be a big performance loss incurred for the processor waiting.

Ideal memory needs to possess the properties of small cell area and great scalability so that low cost can be achieved, non-volatility so that low power will be consumed, and low access latency so that performance can be guaranteed. If such candidate exists, the memory hierarchy can be flattened or even eliminated. All the data can be retained without powering, energy will be consumed only when data is accessed, and required data can be directly fetched for computation almost instantly without processor pausing. Towards this end, great research effort has been done to find the potential universal memory candidate. Fortunately, featured with fast access speed, high density and zero standby power, the emerging non-volatile memories at nano-scale such as spin-transfer torque memory [4], phase-change memory [5], conductive-bridge memory(CBRAM) [7], racetrack memory [9] and memristor [15], have introduced a promising future for the

Figure 2.15: Current status of researches on emerging non-volatile memories towards ideal universal memory

universal memory (Fig. 2.15). Some of them, like racetrack memory and CBRAM, may still be in their infancy and only have small scale demonstration [9, 16], while STT-RAM and phase-change memory are already commercialized and available in the market. For examples, a 45nm 1-gigabit (Gb) phase-change memory for mobile devices has been announced available by Micron [17]. And a 64Mb DDR3 compatible first commercial STT-RAM has been introduced to market by Everspin [18]. To make emerging NVMs widely available in market and replace current memory technologies still requires further advances in their manufacturing technologies.

2.3.1.1 Resistive Random-access Memory

Memristor Dr. Leon Chua in 1976 [19] first predicted the existence of memristor as the fourth fundamental circuit element, as illustrated in Fig.2.16. More than 30 years later in 2008, its first physical realization based on TiO_2 thin-film was demonstrated by HP Labs [15], as shown in Fig.2.17. The name *memristor* is derived from the fact that its resistance is determined by the time integral of the current passed through as if it can remember its history. As illustrated in Fig. 2.18, the memristive effect was achieved by moving the doping front by injected current flow: current from left to right will push the doping front to shift right and vice versa. Without current, doping front retains its previous position. The materials at the different sides of the doping front have different resistivity, which makes its overall resistance the two resistors in series. The bistable states of the device are defined as high resistance state and low resistance state. The write operation is achieved by applying large current, which changes its resistance rapidly.

Figure 2.16: The relations among the fundamental elements and the prediction of the fourth element: memristor

Figure 2.17: STM photo of memristor array from HP Labs [8]

To read the cell, small current is applied to detect its resistance without significantly changing its resistance. Besides the application as high density non-volatile memory [8, 20], memristor is also applied in reconfigurable computing [21, 22, 23], and used as synapses in neuromorphic systems [24].

Conductive Bridging Memory conductive bridging random-access memory (CBRAM), also known as programmable metalization cell (PMC) [25, 26] or NanoBridge [27], is one emerging two-terminal cylinder shape nano-scale device. Each CBRAM cell con-

Figure 2.18: The diagram of TiO_2/TiO_{2-x} based memristor cell structure

Figure 2.19: The diagram of conductive bridging memory cell structure

sists of two metal layers as electrodes, one relatively inert (e.g., tungsten) the other electro-chemically active (e.g., copper), with solid electrolyte between them. Within the solid electrolyte, there exists a vertical conductive filament that grows from the inert electrode. The dynamics of CBRAM can be summarized as the physical re-location of ions between the active electrode and conductive filament. Specifically, when a positive bias-voltage is applied, the active electrode will dissolve and the metal ions will accumulate on the filament, resulting in the vertical growth of filament until the two electrodes are bridged together. This turns the CBRAM into low-resistance state. On the contrary, when a negative bias-voltage is applied, the ions of conductive filament travel back the active electrode, resulting in the vertical shrink of filament, which disconnects the two electrodes and sets the CBRAM in high-resistance state (Fig. 2.19). A variety of electrode and solid electrolyte materials combinations have been reported in the literature [28, 29, 30, 31]. CBRAM-based memory has also been successfully fabricated in chip-level [16].

2.3.1.2 Magnetoresistive Random-access Memory

Giant magnetoresistance (GMR) was discovered in 1988 independently by the groups of Albert Fert and Peter Grnberg [32], for which they were awarded the Nobel Prize in Physics in the year 2007. For two adjacent magnetization ferromagnetic layers, Giant magnetoresistance is observed as significant resistance difference between a parallel and an anti-parallel alignment. The overall resistance in parallel alignment is relatively lower than that in anti-parallel alignment. Motivated by the discovery of GMR, magnetoresistive random-access memory (MRAM) was under great interest since then.

Toggle Mode MRAM Toggle mode MRAM [33, 34, 35] was the first generation of the MRAM, where external magnetic field needs to be introduced to operate the device. The diagram of a toggle mode MRAM cell is shown in Fig. 2.20. Each MRAM cell is composed of three layers: free magnetic plate on the top, fixed magnetic plate at the bottom, and insulator plate sandwiched in between. The fixed layer is strongly magnetized thus its magnetization cannot be easily altered. The stored bit is denoted by the magnetization orientation in free layer, which is made by thin-film ferromagnetic material and its magnetization can be changed by applying external magnetic field, whose strength exceeds original field. To be compatible with electronic system, current induced magnetic field is used. Based on the magnetization orientation in the fixed layout and the high-or-low resistance detected, the polarity of the writable plate can be known for a read operation.

Figure 2.20: The diagram of toggle MRAM cell structure

MRAM has similar performance compared to SRAM but is much less power hungry [33, 34]. IBM has demonstrated MRAM with a fast access times of 2 ns [36]. However,

the MRAM has poor scalability. The current required to generate the external magnetic field increases as the dimensions scale down, which results in high write power consumption. In addition, as the magnetic field for write operation in half-selection scheme cannot be spatially confined for specific target cell, interference between adjacent cells limits cell sizes to around 180 nm or more.

Spin-transfer Torque Magnetic Tunneling Junction The magnetic field switching based MRAM faces severe scaling issues as introduced above. STT-RAM, which eliminates the need of long-range Oersted field to alter free layer magnetization, is introduced to provide better scalability and performance [4, 3].

STT-RAM (Fig. 2.21) is based on the spin-transfer torque effect, which enables current based switching for free layer. Specifically, the magnetization orientation of free layer can be manipulated via spin-polarized current. When electric current flowing through the first strongly magnetized fixed layer, the fixed layer works like a filter and only allows electrons with same spin-polarization to pass, and then spin-polarized current goes into and interacts with the free layer, exerting a spin torque on the magnetization in free layer in a macroscopic perspective. With persistent incoming spin-polarized electrons, the magnetization of free layer is forced to align with incoming spin-polarized electrons through the transfer of angular momentum. This provides a local means of magnetization manipulation, which is favored in electronic application, and its switching current is proportional to the MTJ size as well.

Figure 2.21: The diagram of spin-transfer torque based MRAM cell structure

Racetrack Memory Racetrack memory [9, 10, 37], or domain-wall memory, is considered as the next generation of MRAM after STT-MRAM and also the potential solution as a future universal memory due to extremely high density and high performance. Racetrack memory is a thin ferromagnetic film strip that consists many magnetic domains, where in each domain the stored information is denoted as the magnetization

orientation (Fig. 2.22). The train-like domains can be pushed to move bidirectionally synchronously by applying current along the strip via spin-transfer torque effect. An MTJ-like junction is constructed at certain position of the ferromagnetic strip, and the domain that aligns with the fixed layer serves as the free layer in MTJ. As such, the read and write operations on the racetrack can be achieved through this constructed head, and together with the shift operation the three basic operations of racetrack memory are formed. Racetrack memory is not random access memory as the target cell to be operated needs to be shifted to the read and write head first; instead, it works similarly as shift register.

Based on spin-transfer torque effect, racetrack memory is expected to have similar performance as its precursor STT-MRAM, but provide much higher density due to its highly packed fashion.

Figure 2.22: The diagram of racetrack memory nanowire structure

Topological Insulator Topological insulator (TI) [38, 39] is recently discovered nano-device whose bulk acts as insulator but surface behaves as metal. Materials like Bi_2Se_3 , Bi_2Te_3 and Sb_2Te_3 are experimentally observed as three-dimensional (3D) TI devices [40, 41, 42]. TI has insulating band-gap state in the bulk and gap-less metallic state at surface. The gap-less metallic state at surface in a 3D TI is very robust under perturbations. Due to the strong spin-orbit coupling, electrons in TI move along their surface into two distinguished directions without scattering according to their spins. This works similarly to that vehicles can move in two opposite directions at two sides of a highway without disturbing each other. The scattering, in which electrons deviate from their trajectory resulting in dissipation, is the fundamental reason of power consumption. As state information in a TI device is conducted by ordered spins without scattering, it draws tremendous interest for ultra-low power memory application.

Figure 2.23: (a) The cross-section of phase change memory cell; (b) Temperature profile of chalcogenide material in write operation; (c) mutual phase-change between the amorphous phase and polycrystalline phase

2.3.1.3 Phase Change Random-access Memory

Phase change random-access memory (PC-RAM) device is based on the phase-changing chalcogenide materials such as Ge_2SbTe_5 (GST) [43]. The chalcogenide materials have two different stable states: crystalline and amorphous, shown in Fig. 2.23(c). The resistance of chalcogenide in amorphous state is larger, in several orders of magnitude, than that in crystalline state. In addition, the chalcogenide crystal structure can be thermally switched mutually. Exploiting such unique behaviors of chalcogenide, the phase change memory has been made possible in recent years [6, 44, 45].

A typical mushroom-shaped phase-change memory cell is illustrated in Fig. 2.23(a). A stick of heater is placed right beneath the GST layer with small contact area, and is surrounded by thermally insulating materials. This structure under current will produce heat at higher rate than it can be dissipated, which leads to the temperature surge of bottom GST; and by applying current pulse with different shapes to the heater, the phase changing process can be activated bidirectionally. As illustrated in Fig. 2.23(b), when

a narrow pulse of large current is applied, the temperature of GST will surge above its melting point, leading to the activation of amorphousizing process. On the other hand, when a long pulse of small current is applied, the temperature of GST will increase but stay below its melting point, resulting in the activation of crystallizing process. Similar like previous introduced non volatile memories, the state readout of phase change memory also involves the detection of resistance.

2.3.2 NVM Design Challenges

Although the emerging NVM technologies possess features like faster speed, lower power and higher density, they are still in their infancy and facing some design challenges. The challenge from device level is the lack of design verification platform. For CMOS technology, all devices have well established device models that are used in SPICE simulator for design verification. However, different from conventional devices, the emerging NVM devices possess unique physics and their states are represented by non-electrical variable. Therefore, emerging NVM devices are not compatible with current SPICE simulator if no modification is provided. The work in [46] proposed a NVM simulation theory which is however limited for spintronic NVM technologies. In [47, 48, 49, 50, 51], one universal simulation theory that considers the non-volatile state variables of all NVMs is developed. In Chapter 3, we will discuss how to effectively describe the non-volatile states of all NVM devices.

From circuit level point of view, the current cell structures and operating circuits for the existing memory technologies can not be adopted for the emerging NVM technologies. The fundamental reason is that the working mechanism is no longer electron based. Even for different emerging NVM technologies, their working mechanisms are different from each other. This requires deliberately designed cell structures as well as according peripheral circuits for each emerging NVM technology. In addition, due to immature fabrication process and the high-dimension sensitivity of NVM devices, large device variations will be incurred, which poses even higher requirement for peripheral circuit for robust and reliable operations. In [52, 53, 54, 55, 56], different memory cell structures as well as peripheral circuits are explored for different NVMs. We will discuss the circuit level design challenges in Chapter 4 in details.

Lastly from system level, as the strengths of emerging NVM are different from conventional ones, it brings many possibilities to come up with emerging NVM specific memory hierarchy and architectures to better take advantages of performance improvement. For example, various works on replacing cache and main memory with emerging NVM technologies are attempted in [57, 58, 59, 60]. Besides simply substituting

existing memory with emerging NVM, novel architectures that can exploit the performance potentials of emerging NVM are also studied, such as in-memory architecture [61, 37, 62, 63, 64], where computation can be partially done inside memory instead of fully executed in logic units. We will discuss the system level design exploration in Chapter 5 accordingly.

2.4 Non-volatile In-memory Computing

The memory technology can affect the computing performance from two aspects. The first aspect is from memory technology itself. For example, memory access latency, access energy as well as memory density are important figures of merit of memory that tell how fast and how efficiently data can be stored and retrieved. Beside the memory technology itself, the second aspect is the way how memory is integrated with logic. This will greatly affect how fast and how efficiently the retrieved data can be processed by logic units. In this part, the memory and logic integration issues will be discussed.

2.4.1 Memory-logic-integration Architecture

Current memory and logic integration has hit the memory wall. That is to say, the memory is the bottleneck of the whole system which is not able to provide data at the rate that processor requires. In this case, the processor is not fully operating. Such hardware resource waste is especially severe for data intensive applications. This is because current memory and logic components are separated. The data required by logic components will be readout from memory, and write back the results to memory through I/O after execution is done. In other words, the linking bridge between logic and memory is the limited I/Os.

The memory-logic throughput can be determined by two factors: I/O pin numbers and how fast the I/O can be operated. Speed-wise, the current I/O can be operated ranging from 100MB per second per pin for Flash memory to 10000MB per second per pin for GDDR memory. Although higher I/O operating frequency is desirable, it has fundamental limits such as signal propagation physics (Maxwell's equation) and signal integrity issues (crosstalk, loss and reflection). On the other hand, the I/O number depends on the packaging technology, number of balls bumps for example. Wider I/O requires higher power and cost budget, and current maximum is about 512 bits. Higher number of pins requires packaging and interconnect breakthroughs. As a rule of thumb, a new generation of packaging technology comes at every six years: 1994 lead technology TSOP, 2000 FBGA (0.8mm), 2006 PoP/MCM (0.4mm) and 2012 die stack (40-50 μ m).

As the pitch of interconnects is getting smaller, it is promising to have more I/O for memory chip.

2.4.2 Logic-in-memory Architecture

Instead of improving memory bandwidth, it is also possible to reduce the required data communication traffic between memory and processor. The basic idea behind this is that, instead of feeding processor large volume of raw data, it is beneficial to preprocess the data and provide processor only intermediate result. The key to lower communication traffic is the operands reduction. For example, to perform a sum of ten number, instead of transmitting ten number to processor, in-memory architecture is able to obtain the sum by in-memory logic and transmit only one result thus reduce traffic by 90%. To perform in-memory logic, it is necessary to implement logic inside memory so that preprocessing logic can be done. Such architecture is called logic-in-memory architecture [65, 66, 67, 68, 69]. Figure 2.24 shows logic in memory architecture at memory cell level. The example illustrated here is an in-memory full adder with both *sum* logic and *carry* logic.

The basic circuitry, including access transistor, the word-line and bit-lines, is to ensure memory access. The data is stored in non-volatile memory devices which have either low or high resistance. Redundant data is required for each bit of data for logic purpose. Combinational logic circuit is added inside which the non-volatile devices are equivalent to transistors: considered turned on if at low resistance state or turned off if at high resistance state. In such architecture, the desired result can be obtained immediately without reading operands as if the results are already stored in data array and it can just be “readout”. This is very useful for some specific applications as this architecture is able to preprocess data without loading data to processor with extremely short latency.

As the logic is inserted to one cell or a few cells, it is limited to small size thus cannot be made complex. Usually only simple logic is suitable for such architecture otherwise the overhead would be overwhelming. However, though simple logic in such architecture is able to share the workload of processor, its effect to reduce communication traffic is not obvious due to limited operands reduction. In addition, similar to the operation of memory, for the whole data array only a few logic can be active concurrently at one time. This leads most logic circuit to be idle at most of the time, which is not only a waste of computational resources but also incurs leakage power for CMOS logic. Another disadvantage is that the data needs to be stored in a very strict manner, determined by how the in-memory logic is organized.

An alternative in-memory architecture at block level which is more suitable for traffic reduction is illustrated in Figure 2.25. A memory data is usually organized in H-tree

Figure 2.24: In-memory computing architecture at memory cell level

fashion, and the data block can be the data array or a number of data arrays that belong to same ‘H-tree’ branch. Instead of inserting in-memory logic at memory cell level inside the data array, the architecture in Figure 2.25 pairs each block of data with in-memory logic (accelerators). Different from the cell level in-memory architecture, the accelerators can be made with higher complexity, and the number of accelerators for each data block can also be customized. The data flow of the block level in-memory architecture is to readout data from data block to in-memory logic, which performs particular functionality and then writes back the result. The data also needs to be stored in assigned blocks but it is much more flexible than that of cell level in-memory architecture. The block

Figure 2.25: In-memory computing architecture at memory block level

level in-memory architecture is very effective to reduce communication traffic between memory and processor. This is because high operands reduction can be achieved due to higher accelerator complexity. For example, for face recognition in image processing application, instead of transmitting a whole image to obtain a Bool result, the result can be directly gained through in-memory logic. In other words, the block level in-memory architecture is suitable for big data driven applications where traffic reduction is more important than latency reduction.

Chapter 3

Non-volatile State Identification and NVM SPICE

3.1 SPICE Formulation with New Nano-scale NVM Devices

To analyze a given electric circuit, one needs to formulate differential algebraic equation (DAE) based on Kirchhoffs Current Law (KCL) and Kirchhoffs Voltage Law (KVL), which are two fundamental rules that govern electric systems. The most basic analysis tool SPICE [70], shorted for *Simulation Program with Integrated Circuit Emphasis*, analyzes a given circuit roughly in two steps. In the first step, the SPICE parses circuit description (netlist) and formulates the DAE, based on topology, device models, and parameters for all devices. In the second step, the SPICE solves the equations by LU factorization and Newton iteration. Conventionally, the unknown variables to solve are either branch current or nodal voltage, as all existing electric components are modeled as functions of current and voltage. However, as introduced in Chapter 2, the states of the emerging nano-scale non-volatile memory devices cannot be fully described by electrical states, non-electrical states like doping ratio for memristor or magnetization orientation are also essential. Therefore, the DAE formulated conventionally may not be able to easily consider the new non-volatile devices with non-electrical states.

Due to the lack of non-volatile memory compatible circuit simulator for design verification, circuits including aforementioned NVM devices are currently limited in terms of scale. The challenge of hybrid NVM/CMOS design verification is still left unresolved. With the aid of the SPICE-like simulator for non-volatile memory devices developed in this chapter to describe both electrical and non-electrical states, the hybrid CMOS/NVM co-simulation can be efficiently conducted with high accuracy.

3.1.1 Traditional Modified Nodal Analysis

Assume a circuit with n nodes and b branches, the incidence matrix $E \in (R^{n \times b})$ that represents its topology is defined by

$$e_{i,j} = \begin{cases} 1, & \text{if branch } j \text{ flows into node } i \\ -1, & \text{if branch } j \text{ flows out of node } i \\ 0, & \text{if branch } j \text{ is not included at node } i \end{cases} \quad (3.1)$$

Then we have the Kirchhoffs rules

$$\begin{cases} E j_b = 0, & KCL \\ E^T v_n = v_b, & KVL \end{cases} \quad (3.2)$$

where j_b is the branch current, v_b the branch voltages and v_n the nodal voltages. Ideally, the j_b is dependent on v_n with branch device model known:

$$j_b = \frac{d}{dt} q(E^T v_n, t) + j(E^T v_n, t) \quad (3.3)$$

Equation 3.3 is known as nodal analysis (NA) where all unknown variables are nodal voltages. However, NA falls short when dealing with inductor and voltage source as they become indefinite at dc . The modified nodal analysis (MNA) [71] introduces new state-variables branch inductive current j_l and branch source current j_i , and therefore Equation 3.3 can be rewritten as

$$\begin{cases} \frac{d}{dt} E_c q(E^T v_n, t) + E_g j(E^T v_n, t) + E_l j_l + E_i j_i = 0, \\ L_l \frac{d}{dt} j_l - E_l v_n = 0, \\ E_i^T v_n = 0. \end{cases} \quad (3.4)$$

where $[E_c, E_g, E_l, E_i]$ are the four incidence matrices that describe the topological connections of capacitive, conductive, inductive, and voltage-source elements, respectively. With state variable $x = [v_n, j_l, j_i]^T$, Equation 3.4 in DAE form can be formulated as,

3.1.2 New MNA with Non-volatile State Variables

If additional non-volatile state variables for a given NVM, termed as s_m , are included into MNA, hybrid CMOS/NVM simulation will be then made possible. The work in [47] introduced the theory of hybrid CMOS/NVM simulation. Figure 3.1 shows the tra-

Figure 3.1: New MNA with (a) component symbols and state variables; (b) large signal and (c) small signal KCL

ditional components which can represent CMOS as well as the newly introduced NVM component. The conductance on the added NVM component branch is denoted as memductance (memory conductance), whose value can be determined by v_n and s_m . Note that the complexity to solve DAE is proportional to the number of state variables introduced by all components, Each NVM component normally brings at most three new state variables,¹ which is far less than that of equivalent circuits approach.

In equivalent circuits approach [72, 73, 74], dozens of traditional components (means even more state variables) are used to curve-fit non-linearity of each NVM component, resulting in significant overhead in simulation when there is a large number of NVM components. For example in [74], the SPICE equivalent circuit for STT-MTJ is composed of more than twenty traditional electric devices such as resistors, capacitors, CCVS, CCCS, and VCVS. These devices together will introduce tens of state variables to emulate only one STT-MTJ device.

¹Depending on the connection topology, it will introduce at least one state variable s_m , if two terminal nodes already exist; and at most three: one for s_m and two for terminal v_n , assuming single s_m is adequate.

The new MNA with additional non-volatile state variable s_m then reads,

$$\begin{cases} \frac{d}{dt} E_c q(E_c^T v_n, t) + E_m j(E_m^T v_n, s_m, t) + E_g j(E_g^T v_n, t) + E_l j_l + E_i j_i = 0, \\ L_l \frac{d}{dt} j_l - E_l v_n = 0, \\ E_i^T v_n = 0, \\ f(E_m^T v_n, s_m, t) + \frac{d}{dt} g(E_m^T v_n, s_m, t) = 0. \end{cases} \quad (3.5)$$

where E_m is the incidence matrix for NVM component, and functions f and g are the additional state equations for non-volatile devices with s_m . The term $E_m j(E_m^T v_n, s_m, t)$ is the new NVM branch current.

The DAE form in Equation 3.4 is still valid for the new MNA described in Equation 3.5 above. By calculating the Jacobian matrices, the linearized DAE for new MNA then becomes

$$\begin{bmatrix} G & E_i & E_l & S \\ -E_i & 0 & 0 & 0 \\ -E_l & 0 & 0 & 0 \\ K_v^F & 0 & 0 & K_s^F \end{bmatrix} \begin{bmatrix} \delta v_n \\ \delta j_i \\ \delta j_l \\ \delta s_m \end{bmatrix} + \begin{bmatrix} C & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & L_l & 0 \\ K_v^G & 0 & 0 & K_s^G \end{bmatrix} \begin{bmatrix} \delta \dot{v}_n \\ \delta \dot{j}_i \\ \delta \dot{j}_l \\ \delta \dot{s}_m \end{bmatrix} = -F(X_0, \dot{X}_0, t) \quad (3.6)$$

Jacobian matrices (G, C, L_l, S) are conductance, capacitance, inductance and new memductance, respectively. Jacobian matrices $(K_v^F, K_s^F, K_v^G, K_s^G)$ are the first-order partial derivatives of functions f and g (denoted by superscript F and G), with respect to non-volatile component branch voltage v_b^m and its non-volatile state variable s_m (denoted by subscripts v and s). In the following sections, concrete examples of stamping both ReRAM model and STT-RAM model into the new MNA will be discussed.

3.2 ReRAM Device Model

3.2.1 Memristor

3.2.1.1 Non-volatile State Identification

A compact memristor model for the HP Labs memristor based on TiO_2 thin-film is studied in [49, 51]. A TiO_2 thin-film based memristor is manipulated by applying current with different polarity, which pushes the doping front that separates the doped region and undoped region bidirectionally. Due to the significant resistivity difference in doped

and undoped regions, the variable that determines the external resistance of the memristor can be the ratio (w) of doping region (W) over the whole thickness of memristor film (D). Due to the physical limitation, the resistance of memristor cannot be negative or infinite as applied current prolongs, and therefore the boundary is treated with a modeled slow-down effect. The memristor model introduced in [49] is shown in 3.2.

Figure 3.2: Structure of memristor and nonlinear effects for dynamic model: (a) slow-down effect at boundary, (b) exponential relation between drift velocity and electric field

Figure 3.3: Validated memristor model with Joglekar window function for memristor hysteresis loop in [49]

$$\frac{v(t)}{i(t)} = R_{off} - (R_{off} - R_{on}) w(t), \quad (3.7)$$

$$\frac{dw(t)}{dt} = \frac{\mu E_0}{D} \sinh\left(\frac{v(t)}{DE_0}\right) F(w(t)) \quad (3.8)$$

where $F(w)$ is the Joglekar window function [75]

$$F(w) = 1 - (2w - 1)^{2p} \quad (3.9)$$

The memristor model in equation (3.7) and (3.8) captures the experimental measurement well, as shown in Figure 3.3. Note that the memristor needs time to switch, and the DC analysis simulates the device at one time point so the switch will never happen. The hysteresis loop is obtained in transient analysis with swept input sine signal $v = 0.2 + \sin(200\pi t)$ (V).

With equations (3.9), (3.7) and (3.8), the new MNA (3.6) introduced in last section can be completed with following required Jacobian terms

$$G = \begin{bmatrix} G & -G \\ -G & G \end{bmatrix}; S = \begin{bmatrix} v_n \times \frac{dG}{dw_m} \\ -v_n \times \frac{dG}{dw_m} \end{bmatrix}$$

$$K_v^F = \begin{bmatrix} -c_1 c_2 F(w_m) \cosh(c_2 v_n) & c_1 c_2 F(w_m) \cosh(c_2 v_n) \end{bmatrix};$$

$$K_s^F = -c_1 \sinh(c_2 v_n) \times \frac{dF(w_m)}{dw_m}; K_v^G = 0; K_s^G = 1,$$

where G is the conductance and $\frac{dG}{dw_m}$ is the non-volatile state variable derivative of conductance; $c_1 = \frac{\mu E_0}{D}$, $c_2 = \frac{1}{DE_0}$; and $F(w)$ is the Joglekar window function.

3.2.1.2 Simulation Efficiency Comparison

The current approach of co-simulating memristor components together with conventional electric components replaces NVM devices with their equivalent circuits [72, 73]. Due to the aforementioned non-linearity of memristor model, the equivalent circuit can be quite complicated for curve-fitting, which therefore leads to higher complexity of formulated DAE and longer time to solve. For instance, figure 3.4 depicts an equivalent circuit of memristor presented in the work [72]. Many circuit components such as resistor, capacitor, diode and voltage-controlled-current-source are used with dozens of internal nodes added, which introduces many state variables. On the contrary, the implementation of intrinsic memristor model in the new MNA only introduces one non-volatile state variable.

The transient analysis efficiency comparison between non-volatile state variable based approach and equivalent circuit based approach is shown in Figure 3.5. The equivalent circuit in the work [72] is used for comparison. The testbench circuit is a memristor crossbar [15] with varying number of memristor components. The result showed that

Figure 3.4: An equivalent circuit of memristor model

for small memristor crossbar, i.e. small number of memristor components, the non-volatile state variable based approach can achieve an efficiency improvement of 2X. When the memristor crossbar size is relatively large, 32×32 with 1024 memristors for instance, the non-volatile state variable based approach can achieve a speedup of around 40X. The trend also tells that the improvement can be higher for larger number of memristor. Since the memristor circuit is usually tested with large number of components, for instance for memory array validation, the speedup can be expected to be at least one order of magnitude, which makes the proposed simulation approach significantly useful when simulating circuit with large number of memristors.

Figure 3.5: Simulation time consumption comparison between non-volatile state variable based approach and equivalent circuit based approach for memristor

Figure 3.6: (a) Working mechanism of CBRAM with the shape morphing of conductive filaments in several phases illustrated between ON-state and OFF-state; (b) cross section of CBRAM device with defined geometric variables

3.2.2 Conductive Bridge

3.2.2.1 Non-volatile State Identification

The physical working mechanism of conductive bridging random-access memory (CBRAM) device is illustrated in Figure 3.6(a), which can be summarized as the shape-morphing of the conductive metal filament within ambient resistive solid electrolyte. When a positive bias-voltage is applied, the conductive filament, modeled as a cone with fixed base radius, first grows vertically by accumulating metal ions until the two electrodes are bridged together. Then the cone begins to morph into a cylinder, which will eventually set the CBRAM into low-resistance or *ON-state*. Similarly when a negative bias-voltage is applied, the cylinder-shaped conductive filament dissolves to a cone with same base area, disconnects the two electrodes, shrinks vertically and then turns the CBRAM in high-resistance or *OFF-state*. From the defined geometric variables shown in Figure 3.6(b), it can be observed that the variables h and r are capable to model the shape-morphing process, where detailed equations are given in [76].

We find that the angle θ from the side of the cone to the bottom can determine the shape of filament, when assuming that a cylinder is simply a cone whose apex is at infinity. Thus the non-volatile state variable s or $\tan\theta$, i.e. the height of cone in a projective perspective to base radius, is selected to determine the resistance of one CBRAM device. Hence s becomes a shape-determining state variable in which a large value of s indicates the filament more cylinder shaped while a small value for more cone shaped. As such, the number of state variable for each device can be reduced from two to one, which will greatly improve the simulation speed.

As such, by modifying the equations from [76], the dynamic behavior of CBRAM device based on variable s can be modeled as

$$\frac{ds}{dt} = \begin{cases} \frac{k_1}{R} \cdot \sinh\left(\frac{k_2 \cdot \mathbf{V}}{L + k_3 \cdot R \cdot s}\right) & s \leq \frac{L}{R} \\ \frac{s^2 \cdot k_1}{L} \cdot \sinh(k_4 \cdot \mathbf{V}) & s > \frac{L}{R} \end{cases} \quad (3.10)$$

where k_1 , k_2 , k_3 and k_4 are constants, and their detailed definitions can be found in [76]. Additionally, $\mathbf{V}$ is the applied voltage on CBRAM device, L is the thickness of CBRAM and R is the constant base radius.

The equivalent resistance of CBRAM with respect to variable s can be calculated by

$$\begin{cases} R_{off} = \frac{\rho_{on} \cdot s \cdot R + \rho_{off}(L - s \cdot R)}{\pi \cdot R^2} & s \leq \frac{L}{R} \\ R_{on} = \frac{\rho_{on} \cdot L}{\pi \cdot R \cdot (R - L/s)} & s > \frac{L}{R} \end{cases} \quad (3.11)$$

where ρ_{on} and ρ_{off} are the conductive filament resistivity and non-conductive solid-electrolyte resistivity, respectively.

Two observations from equations (3.10) and (3.11) can be drawn for the fast write-speed of CBRAM. Firstly, there exists an exponential relation between the state changing-speed ds/dt and the applied voltage $\mathbf{V}$. In the literature, CBRAMs with less than 50ns write-latency have been demonstrated under 1.5V bias voltage [16, 77]. Secondly, the morphing steps between *OFF-state* and intermediate state 1 as well as that between *ON-state* and intermediate state 2 are much more time consuming than that between intermediate states 1 and 2. Practically, by properly defining the intermediate states 1 and 2 and operating the devices within this range, the unnecessary time for morphing at two ends can be eliminated thus to significantly improve the write-speed while still to achieve acceptable *off/on* resistance ratio.

In addition, it can be observed that the switching speed indicated by equation (3.10), and also R_{on} and R_{off} indicated by Equation (3.11), are all dependent only on the shape

of internal conductive filament, rather than the external sizes of the upper and bottom contacts. As the filament size is significantly smaller than that of contacts, the CBRAM device exhibits a great potential of scalability. The scalability of CBRAM device has been confirmed by [7], within which it has been demonstrated that the threshold voltage, R_{on} and R_{off} are feature-size independent from μm to nm regions.

Combining equations (3.10) and (3.11) together, we can reach the conclusion that the branch current of the CBRAM device can be decided by the conductive filament shape-determining variable s and also the applied voltage V . As such, the modified nodal analysis (MNA) that takes s as the additional new non-volatile variable can be built into NVM-SPICE [78] and then solved by it.

As discussed previously, to perform transient analysis together with NVM devices, the linearized circuit Equation 3.6 has to be established.

The new state variable vector $X = [v_n, j_i, j_l, s_m]^T$ contains nodal voltage, source current, inductor current, and non-volatile state variable of NVM devices, respectively. $K_v^g, K_s^g, K_v^c, K_s^c$ are the additional Jacobian terms introduced from NVM device equations. As such, for CBRAM memory circuit designs, our approach avoids the use of equivalent circuit components, which is not scaled with dependence on geometries and not scalable for large-scale memory design. Moreover, the reduction of the number of CBRAM state variables from two to only one will also greatly reduce the size of state matrix in (3.6).

Applying equations (3.10) and (3.11) to (3.6), we can obtain the required Jacobian terms as follows

$$G = \begin{bmatrix} g(s) & -g(s) \\ -g(s) & g(s) \end{bmatrix}, S = \begin{bmatrix} V \cdot \frac{dg(s)}{ds} \\ -V \cdot \frac{dg(s)}{ds} \end{bmatrix},$$

$$K_v^g = \begin{bmatrix} -\frac{df(V,s)}{dv} & \frac{df(V,s)}{dv} \end{bmatrix}, K_s^g = -\frac{df(V,s)}{ds},$$

$$K_v^c = 0, K_s^c = 1 \quad (3.12)$$

where $g(s) = \frac{1}{R}$ is the CBRAM conductance, and $f(V, s) = \frac{ds}{dt}$ as indicated in equation (3.10). As such, the new MNA for CBRAM is formulated and can be directly applied in transient analysis in NVM-SPICE.

3.2.2.2 Simulation Results

The CBRAM device model parameters are based on [16]. Besides, 100nm L_{pitch} , 1.8V v_w , 0.9V v_r , and 6.4fF distributed C are utilized as settings. Figure 3.7 shows the val-

validation of CBRAM device model against the published measurement data [16, 26]. It can be seen that the exponential relation between applied voltage and switching time is valid.

Figure 3.7: CBRAM model validation against the published measurement data [16, 26]

The transient analysis for CBRAM switching is conducted in the NVM-SPIICE simulator with results shown in Figure 3.8. The model parameters k_1 , k_2 , k_3 and k_4 are based on [16] with 1.8V voltage supply. $2M\Omega$ and $1G\Omega$ are assumed as thresholds for *on* and *off* states of CBRAM, respectively. A pitch size of $100nm$ is assumed for the crossbar structure, with $50nm \times 50nm$ cross-section area of nano-wire made by copper. Multiple supply voltages are used, where 65nm technology node with 1.2V is assumed for CMOS logic and 1.8V for CBRAM-crossbar operations. The CBRAM is initialized at OFF-state, and 1.8V voltage is applied to set the device (off to on). The non-volatile state variable s grows from 0.2 to around 4, indicating that the conductive filament first grows its height as a cone, reaches the top electrode, and then transforms towards cylinder. This is also confirmed by the change of resistance from $G\Omega$ scale to $M\Omega$ scale. The CBRAM has been successfully set after around $5ns$, then a $-1.8V$ voltage is applied to reset the device, where a reverse process can be observed.

3.2.3 ReRAM Model Extension

As the physics of current ReRAM is still in debate, it is of great importance for the proposed framework to support model extension. Besides the memristor and CBRAM models introduced above, it is also possible to extend NVM-SPIICE to support other ReRAM models, as long as the model is well formulated. Firstly, a well formulated

Figure 3.8: Transient response of CBRAM device Set and Reset when v_w amplitude of 1.8V is applied

model must describe the link between electrical resistance and non-volatile state variable, as illustrated in equations 3.7 and 3.11 for instances. Secondly, a well formulated model must relate the non-volatile state variable with electrical voltage or current, as shown in equations 3.7 and 3.10. In other words, it is necessary to know how the voltage or current on the NVM device will alter its non-volatile state variable. In addition, for piece-wise models, it is required to ensure the first and second derivatives of the model equations are continuous due to numerical stability and convergence issue.

It is also of great importance to extend models to consider process variations. For ReRAM devices, the on-state and off-state resistance values are highly dependent on the geometrical parameters. Therefore, by considering the geometrical parameters in the resistance/non-volatile state variable link, the simulation with process variation can be included.

3.2.4 A Case Study: ReRAM Synapse for Analog Learning

The value-adaptive nature of memristor can lead to potential application in neuromorphic systems. There are many recent researches conducted on implementing memristors in neural network and other biological circuits [79, 80, 81].

In the Pavlov's classic experiment, an unconditional food stimulus can always elicit unconditional salivation response of dogs, and a neutral stimulus of bell sound is not able to elicit salivation at first. However, after repeated sound-food pairing stimuli the salivation can be elicited by only sound stimulus without food. The neutral bell stimulus then becomes a conditional stimulus which is able to elicit conditional response. This conditioning behavior can be strengthened or weakened by more sound-food pairing training or no more food association with sound.

Such a training behavior has been modeled in [82]. Memristors have been utilized as synapses to denote the dynamically changing strong or weak bond between neural cells by its programmable ability. However their model of memristor ideally assumes a threshold for memristor programming, and therefore cannot model the behavior that the conditioning can be dynamically weakened. Here we propose a simple hybrid CMOS memristor circuit, as shown in figure 3.9, which can better model the conditioning behavior.

Figure 3.9: A simple hybrid CMOS-memristor circuit to model the dynamic conditioning behavior

The interaction between unconditional food stimulus and salivation response can be modeled as a resistor R_f with very small resistance, to indicate that the food stimulus signal can easily pass through to generate salivation output. Since the sound stimulus is not able to elicit salivation at first, the memristor is initially programmed in OFF-state to prevent the salivation response. R_o is designed with resistance somewhere between ON-state and OFF-state resistance. The sound and food stimuli can be controlled by CMOS transistors. The unconditional stimulus voltage V_f is set with higher voltage than neutral stimulus V_s . Therefore before training, when unconditional food stimulus is applied, the salivation output will always respond $V_o \approx V_f$. The sound stimulus alone will result in $V_o \approx 0$. The training can be achieved by applying both food and sound stimuli. The resistance relationship will lead to $V_o \approx V_f > V_s$. The negative voltage drop on memristor will slowly program it from OFF-state to ON-state. After training, the sound stimulus alone can also elicit salivation output $V_o \approx V_s$, and hence the conditioning is established. During the conditioning action, there exists a small positive voltage drop on memristor which will eventually program memristor from ON-state to OFF-state. This can be observed as the conditioning getting weakened.

The above hybrid design for classic conditioning behavior is simulated within NVM-SPICE. R_o is set to $2k\Omega$, R_f 100Ω , V_f DC 2V, and V_s DC 1V. The memristor is set

Figure 3.10: The simulation results of the hybrid CMOS-memristor design for simple classic conditioning

with $R_{on} = 100\Omega$, $R_{off} = 40k\Omega$, and other parameters with default values as described in Table A.1. Two ideal transistor switches are controlled by the “food” and “sound” signals. Transient analysis is conducted and the results are shown in figure 3.10.

The simulation results, shown in figure 3.10, are consistent with previous analysis. Before training, the unconditional food stimulus can elicit an unconditional salivation response, while the neutral sound stimulus is unable to do so due to the high OFF-state resistance of memristor which cuts down the signal transmission. During the training phase, the sound stimulus is associated with food stimulus. This is physically achieved by a negative voltage drop on memristor which programs it from OFF-state to ON-state. This can be observed in the fourth subfigure. After training, the neutral sound stimulus then becomes conditional stimulus which can elicit conditional salivation response. The response is then weakened as there is no reward of food. The conditioning then disappears. This behavior is physically achieved because the memristor is programmed from ON-state back to OFF-state. Note that by scaling the settings the training speed and weakening speed can be controlled. For instance, a higher V_f will contribute to a faster training, and a lower V_s will slow down the conditioning weakening process.

3.3 Spintronics Device Model

3.3.1 STT-MTJ

Based on spin-transfer torque effect, STT-RAM is seen as the second phase of magnetization based non-volatile memory technology after the toggle MRAM. Because of

its great scalability thanks to current-induced magnetization switch instead of external magnetic field, it has attracted a lot of attention. In the following, the non-volatile state variables of STT-MTJ are identified based on the magnetization physics.

3.3.1.1 Non-volatile State Identification

The fundamental structure of spin-transfer torque based memory (STT-RAM) is the spin-transfer torque tunneling junction (STT-MTJ) [83, 84]. An STT-MTJ has two major physical effects, giant magnetoresistance (GMR) effect and spin-transfer torque as its name suggests. The GMR effect, as introduced in Chapter 2, determines the resistance of STT-MTJ according to its magnetization alignment in two adjacent magnetic layers [85]. When the magnetization in fixed layer is same to that in free layer, i.e. parallel alignment (P), it exhibits low resistance; and on the other hand, when they are in an anti-parallel alignment (AP), it shows a high resistance instead. As the free layer is where the data bit is stored, in the form of magnetization direction, by detecting the resistance, the state of an STT-MTJ device can be known. Therefore, the resistance of STT-MTJ can be expressed as a function of its magnetization angle [32],

$$\begin{cases} R(\theta) = R_L + \frac{R_H - R_L}{2}(1 - \cos(\theta)) \\ = R_L + \frac{\Delta R_{GMR}}{2}(1 - \cos(\theta)) \end{cases} \quad (3.13)$$

and when $R(\theta = 0)$ and π , it can be easily observed that the resistance equals to R_H for AP state and R_L for P state, respectively.

According to [49], the GMR resistance is also dependent on the voltage across the device,

$$\begin{cases} R_l(V) = \frac{R_{l0}}{1 + c_l V^2} \\ R_h(V) = \frac{R_{h0}}{1 + c_h V^2} \end{cases} \quad (3.14)$$

where c_l and c_h are voltage-dependent coefficients for P state and AP state.

As the GMR effect is exploited for read operation, the spin-transfer torque is used for write. The write operation of STT-MTJ can be interpreted as the magnetization direction switch, or change of θ , between A and AP states by applying write current with different polarity. In order to model the dynamic behavior of magnetization reversal, the magnetization trajectory needs to be studied, which is described by the normalized Landau-Lifshitz-Gilbert equation (LLG) at macro-scale

$$\frac{dm}{d\tau} = -m \times h + \alpha(m \times \frac{dm}{d\tau}), \quad (3.15)$$

and its solution for STT-MTJ is given by [49],

$$\theta = \theta_0 \text{Exp} \left(-\frac{t}{t_0} \right) \cdot \cos(\phi) \quad (3.16)$$

$$\omega = \frac{d\phi}{dt} = k_1 \sqrt{k_2 - (k_3 - k_4 I)^2} \quad (3.17)$$

where θ and ϕ are the modeled magnetization in the spherical coordinates, as shown in Figure 3.11, that denote the magnetization direction. θ_0 is the initial value of θ , slightly tilted from x -axis by thermal noise; I the write current that causes the magnetization reversal; t_0 is reversal time constant, and ω angular speed of ϕ . k_1 to k_4 are magnetic parameters with detailed explanation in Appendix B;

Applying equations (3.13), (3.14), (3.16) and (3.17), we can complete the linearized DAE with following Jacobian terms

$$G = \begin{bmatrix} \frac{dj_b}{dv_n} & -\frac{dj_b}{dv_n} \\ -\frac{dj_b}{dv_n} & \frac{dj_b}{dv_n} \end{bmatrix}; S = \begin{bmatrix} v_n \times \frac{dG}{d\theta_m} & 0 \\ -v_n \times \frac{dG}{d\theta_m} & 0 \end{bmatrix}; C = 0$$

$$K_v^F = \begin{bmatrix} -\frac{df(\phi_m, j_b, t)}{dj_b} G & \frac{df(\phi_m, j_b, t)}{dj_b} G \\ -\frac{d\omega_m}{dj_b} G & \frac{d\omega_m}{dj_b} G \end{bmatrix}$$

$$K_s^F = \begin{bmatrix} 1 & -\frac{df(\phi_m, j_b, t)}{d\phi_m} \\ 0 & 0 \end{bmatrix}; K_v^G = 0; K_s^G = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

where $\frac{dj_b}{dv_n}$ and $\frac{dG}{d\theta_m}$ can be derived from equations Equation 3.13 and 3.14, and $\frac{df(\phi_m, j_b, t)}{dj_b}$, $\frac{df(\phi_m, j_b, t)}{d\phi_m}$ and $\frac{d\omega_m}{dj_b}$ can be derived from Equation 3.16 and 3.17.

Figure 3.11: Spherical coordinates with two magnetization angles: θ and ϕ

Figure 3.12: Equivalent circuit of STT-MTJ model

3.3.1.2 Simulation Efficiency Comparison

The 1T-1MTJ structure STT-RAM array with transient analysis for a write operation is set as the test bench circuit. The circuit netlist is described in two versions which only differ in terms of STT-MTJ model, one using the equivalent circuit (Fig. 3.12) based SPICE macromodel in [74], and one using the intrinsic physics-based model in NVM-SPICE. Necessary modifications are made to port the HSPICE subcircuit netlist provided by [74] compatible with Berkeley SPICE styled NVM-SPICE. The simulation duration is set 20ns with time step of 0.1ns.

Table 3.1: Simulation time comparison for STT-RAM array using different simulation approaches (unit in second)

array size	Macromodel in [74]	NVM-SPICE	speedup
8×8	2.522	0.257	10x
16×16	98.131	1.87	52x
32×32	1119.99	11.533	97x
64×64	22188.8	189	117x

Table 3.1 shows the runtime comparison of the two different simulation approaches for different array sizes. It can be observed that the simulation using NVM-SPICE intrinsic physics-based model is 10x~117x faster than the equivalent circuit approach. The advantage will be even larger when the array size increases. For typical memory array size of hundreds by hundreds, the speedup can be expected more than 100x.

For simulation accuracy, both the equivalent circuit approach and the physics-based model are validated to be accurate in [74, 50] in terms of switching time. However, the magnetization reversal in the equivalent circuit approach is only a smooth transition. In the physics-based model, the oscillation of magnetization can be captured (shown in Fig. A.5), which correlates with the actual magnetization damping observation. The

modeling of such complicated non-linearity is theoretically attainable for equivalent circuit approach, but extremely inefficient in practice as it may introduce hundreds of state variables to emulate even one single STT-MTJ.

3.3.2 Topological Insulator

3.3.2.1 Non-volatile State Identification

A spin-transfer torque magnetic tunneling junction (STT-MTJ) device is found with state depending on magnetization angle. Similarly, a topological insulator (TI) device also has non-conventional electrical states to describe. In this section, the working mechanism of TI is discussed with additional state variable identified as well. Then, similar to the BSIM model for MOSFET, the according device model to stamp a TI device in SPICE is described. The deployed variables and terms are summarized in Table 3.2.

Table 3.2: Notation for topological insulator device modeling

Symbols	Definitions
θ, ϕ	shown in Figure 3.11, azimuthal angles of magnetization orientation in x-z and x-y plane.
α	damping constant
H_{eff}	effective field
H^e, H_k	external applied field and shape anisotropy field
M	magnetization
M_s	saturation magnetization of material
m, h	normalized magnetization and effective magnetic field
$coef f$	coefficient between read-current and produced external magnetic field
σ_H, V_H	quantum Hall conductance and quantum Hall voltage

A typical TI based memory device is built by a two-layer structure with ferromagnetic layer on the top and topological insulator layer on the bottom as shown in Figure 3.13(a). The TI device has four terminals, with two controlling terminals along x -axis and two Hall terminals attached to the lateral sides. As discussed in [86], one bit can be then stored by the perpendicular magnetization of the ferromagnetic layer. Programming a bit requires an external magnetic field whose field strength is exceeding the coercivity of ferromagnetic layer. Under the magnetic field of the ferromagnetic layer, the topological insulator exhibits a quantum Hall conductance as shown in Figure 3.13(b). It can be observed that the sign of quantum Hall conductance is determined by the magnetic field orientation, thus the stored bit can be read out by detecting the sign of quantum

Figure 3.13: (a) Device structure; (b) schematic diagram of quantum Hall conductance; and (c) abstracted equivalent device circuit model

Hall voltage

$$V_H = \frac{I_R}{\sigma_H} \quad (3.18)$$

where I_R is the applied read-current pulse along x axis.

The Hall conductance σ_H can be calculated by [87]

$$\sigma_H = \frac{e^2}{h} \int \frac{d^2 \vec{k}}{(2\pi)^2} (f_c - f_v)(\vec{k}) \Omega_z(\vec{k}), \quad (3.19)$$

where the $\vec{k}$ is the wave vector, Ω_z the Berry curvature, f_c and f_v the Fermi-Dirac distributions of conduction band and valence band, respectively.

It can be seen that the Hall conductance σ_H is a function of band-gap, Fermi Level and temperature. When the band-gap $\Delta \gg k_B T$, Equation 3.19 becomes

$$\sigma_H \approx \frac{e^2}{2h} \text{sgn}(M). \quad (3.20)$$

The h here is the Planck constant, e is the charge of electron, and $\text{sgn}(M)$ is the orientation of magnetization. As such, the quantum Hall conductance is a constant approximately equals to $19.4 \mu S$. Note that the TI device is insensitive to disorder, imperfection, and cell geometry, which ensures a constant readout voltage even in the presence of perturbations.

From Equation 3.20, the quantum Hall voltage can be regarded as a current-controlled voltage source with the coefficient of σ_H , as shown in Figure 3.13(c). Note that this equivalent quantum Hall voltage source has very limited driving ability, and a large internal resistance R_{in} is introduced for an accurate modeling.

More importantly, in order to model the dynamic behavior of the TI device during programming procedure, the magnetization trajectory is also studied by the normalized Landau-Lifshiltz-Gilbert equation (LLG),

$$\frac{dm}{d\tau} = -m \times h + \alpha(m \times \frac{dm}{d\tau}), \quad (3.21)$$

where the normalized effective field h equals to $-\frac{\delta\varepsilon}{\delta m}$.

The ε is the normalized energy density:

$$\varepsilon = \frac{1}{2}h_k(1 - m_z^2) - m \cdot h^e, \quad (3.22)$$

where the two energy density contributions are associated with anisotropy field and the external field, respectively.

Note that the required external magnetic field for device programming is generated by a current, as shown in Figure 3.13(c). There exists a coefficient between the read-current and the produced field,

$$H^e = coeff \cdot I_M. \quad (3.23)$$

We assume the external magnetic field only have a perpendicular component along the easy axis z . Thus, normalized effective field $h = coeff \cdot I_M/M_s + h_k \cdot m_z$ can be obtained.

The solution of Equation 3.21 can be interpreted as the change of the normalized magnetization (m) over time. The normalized magnetization m can be expressed in spherical coordinates with variables θ and ϕ as shown in Figure 3.11. The dynamic behavior described by θ and ϕ can be finally calculated by LLG

$$\theta = \theta_0 \exp\left(-\frac{t}{t_0}\right) \cdot \cos(\phi) \quad (3.24)$$

$$\omega = \frac{d\phi}{dt} = k_c \sqrt{k_d - (\alpha h_k - \alpha h_x^e)^2} \quad (3.25)$$

where θ_0 is the initial value of θ , slightly tilted from the stable z or $-z$ directions; t_0 is precession time constant; and ω is the angular speed of θ ; $k_c = \gamma_0 \cdot M_s$ is product of gyro-magnetic ratio and saturation magnetization; and $k_d \approx \frac{H_k}{M_s}$. Definitions of the remaining variables are also shown in Table 3.2.

The new state variable vector $X = [v_n, j_i, j_l, s_m]^T$ contains nodal voltage, source current, inductor current, and the new state variable s_m (magnetization angles θ and ϕ) for a TI device, respectively. We assume that i) the conductance along x axis shows only weak dependency on the new state variables θ and ϕ ; and ii) the magnetization is only

subject to the external field, and hence $S \approx 0$, $K_v^f \approx 0$ and $K_v^g \approx 0$.

Applying equations (3.24) and (3.25) to (3.6), we can obtain the required Jacobian terms as follows

$$G = \begin{bmatrix} \sigma_x & -\sigma_x \\ -\sigma_x & \sigma_x \end{bmatrix};$$

$$K_s^f = \begin{bmatrix} 1 & -\frac{df(\phi_m, t)}{d\phi_m} \\ 0 & 0 \end{bmatrix}; K_s^g = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$$

where $f(\phi_m, t)$ is the right-hand-side of Equation 3.24, and σ_x is the conductance along the x -axis provided as a model parameter. Due to the non-scattering property of TI surface, an extremely high σ_x can be expected, which will contribute to an ultra-low power consumption.

Besides the magnetization dynamics, the derived MNA also has to model the quantum Hall voltage readout behavior. As discussed in the last subsection, the quantum Hall voltage can be modeled as a current-controlled voltage source, with coefficient as $\frac{1}{\sigma_H}$. As such, applying Equation 3.18, the incidence matrix for the quantum Hall voltage source can be obtained as

$$E_i = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Meanwhile, I_c/σ_H has to be added to the corresponding right-hand-side of Equation 3.6, where I_c is the read-current flowing through x -axis, and σ_H can be calculated by Equation 3.19. As the linearized system Equation 3.6 is complete, hybrid TI/CMOS simulation can be conducted.

3.3.2.2 Memory Cell Circuit Design

Figure 3.14: Cell circuit of topological insulator based memory

Inspired by the toggle MRAM design [33], the cell structure of a TI device for NVM

application is proposed in Figure 3.14. To program a selected TI cell, both word-write-line (WWL) and bit-write-line (BWL) produce half of the required external magnetic field H_e .

In order to achieve addressability, the amplitude of H_e is subject to:

$$H_e/2 \leq H_c \leq H_e \quad (3.26)$$

where the H_c is the coercivity of the magnetic surface. Here the currents along WWL and BWL to produce $H_e/2$ are defined as programming current I_{PW} and I_{PB} , respectively. The upward I_{PB} and leftward I_{PW} are defined as positive. If both I_{PW} and I_{PB} are applied, there exists an external magnetic field of H_e that exceeds the coercive field of ferromagnetic layer to program a cell. If I_{PW} is applied only or I_{PB} is applied only, the cell is exposed to a magnetic field of $H_e/2$, which is insufficient to switch the magnetization. A zero magnetic field, when no current is applied for both WWL and BWL, is also not able to switch the magnetization. Thus, the cell addressability can be achieved under programming operation. Moreover, to read a cell, the cell is first selected by the signal through word-line, and then a read-current pulse I_R is applied through bit-line. The corresponding quantum Hall voltage V_H will be either positive or negative depending on the status of magnetization orientation. Therefore, by comparing V_H with reference voltage, the bit stored by magnetization orientation can be known.

3.3.2.3 Memory Array Circuit Design

Figure 3.15: One 4×4 topological insulator based NVM array

Figure 3.15 shows a 4×4 memory array design based on the TI memory cell. It can

be observed that it is impossible to write 1-bits and 0-bits for a word at one time since they require opposite I_{PW} and I_{PB} directions. Thus to write a word, writing 1-bits or 0-bits is conducted separately. To write 1001 for *word0* for instance, WWL0, BWL0 and BWL3 are applied with positive I_{PW} and I_{PB} while other WWLs and BWLs are applied with no current. As discussed in last subsection, the addressability allows only *bit0* and *bit3* to be programmed into 1, while other bits remain their status. Then WWL0, BWL1 and BWL2 are applied with negative I_{PW} and I_{PB} so that *bit1* and *bit2* are programmed into 0. As such, 1001 is successfully written to *word0* in two separate steps. Because there exists an inversely proportional relationship between current induced magnetic field and distance in space, and also due to the programming threshold for magnetic field, the operations on target cells will not interfere their neighboring cells. To read out a word, its corresponding word-line is selected, and a current pulse I_R is applied for all bit-lines. As a result, the V_H for each bit can be interpreted by sense amplifiers for each bit-line.

A TI based NVM design platform is developed. As CMOS circuits are still required as interfacing part, hybrid CMOS-TI simulation is required. Similar to a BSIM model for MOSFET, the physical model of TI device is implemented into a SPICE-like simulator NGspice [88]. Based on the developed SPICE-like simulator, a number of experiments have been conducted for TI based NVM designs. In the following numerical experiments, we first validate our physical model against the reported measurement data [89]. Then, we design the cell and array memory circuits of TI and verify them by the SPICE-like simulator under both read and write operations. Finally, we compare the performance of read and write power with the other emerging NVM devices. All numerical experiments are implemented in C and are conducted on the same work-station with Intel Core i5 CPU and 8G RAM.

3.3.2.4 Validation of Dynamic Effect with New State Variable of TI Device

In order to validate the dynamic model of magnetization for TI device, we show the device level simulation results based on soft ferromagnet material $Ni_{60}Fe_{40}$ with parameters extracted from measurement in [89]. The lateral dimension of TI device is 0.8 by $1.6 \mu m^2$, with ferromagnet layer thickness of $5nm$. The saturation magnetization is set to $740kA/M$, the shape anisotropy is field $1.72kA/M$, the damping constant is 0.01, and the current-to-magnet coefficient is 10^6 . Same parameters are assumed in the following experiments for consistency.

Figure 3.16 shows the magnetization switching time versus the applied external magnetic field amplitude. We can see the results produced by our simulator fit well with the

measured data reported in [89]. A nearly inversely proportional relationship can be observed between switching time and magnetic field. For example, in order to achieve a faster memory programming speed, a stronger magnetic field is desirable. Moreover, a switching threshold can be observed from Figure 3.16. So only when the external magnetic field exceeds a certain strength, approximate 2mT observed from figure, can the magnetization be switched. This is also consistent with [89], where the magnetic coercivity is reported as 2mT.

Figure 3.16: Validation of the switching time and external magnetic field relationship for magnetization dynamics

As such, the magnetization dynamics has been verified validate for our developed TI device simulator. Note that for TI with other materials as ferromagnetic layer, parameters of saturation magnetization, shape anisotropy field, and damping constant need to be specified for the simulator in a model file.

3.3.2.5 Hybrid Simulation with CMOS for TI-based Memory Cell

In order to investigate the performance of TI device based memory, transient analysis is conducted for one TI memory cell circuit as illustrated in section 3.3.2.2. The technology node is 65nm for CMOS part, and VDD is 1.2V. The aspect ratio of transistors are set to 3. The read-current pulse is set to be 1μA with a pulse width of 5ns. The programming current of both word-write-line I_{PW} and bit-write-line I_{PB} are generated by their respective current sources. As discussed above, strong external magnetic field is desirable for a fast programming speed. Moreover, in order to achieve memory cell addressability, however, the field strength should be subject to Equation 3.26. So in this work, the magnetic coercivity H_c is 2mT, and H_e is designed at 3mT. Parameters for ferromagnetic layer are the same as last section.

Figure 3.17 shows the dynamic response when using the new state variables. Both programming current I_{PW} and I_{PB} are applied at zero second. It can be observed that θ starts to deviate from the original angle once H_e is applied. Its maximum deviation increases exponentially with time before the reversal happens. The fluctuation decays very fast after reversal since the shape anisotropy field alters its direction to strengthen H_e , and then the device enters into the other stable state. The whole process indicates that a write latency, i.e. switching time, of about $10ns$ is achieved under 3mT magnetic field. The continuously increasing ϕ shows a very fast circulating frequency in the x-y plane, which causes θ to fluctuate in the same frequency.

Figure 3.17: Dynamic response of topological insulator new state variables

The simulation result for the cell circuit readout operation is shown in Figure 3.18. Firstly, the word-line signal is set to logic-1 before the readout operation. Then at the time of 12ns, a read-current pulse with amplitude of $1\mu A$ is applied through bit-line. The quantum Hall voltage responds simultaneously, and is fed into the sense amplifier (SA) later. After a time of T_{delay} , which is the combination of bit-line delay and SA sensing delay, the SA outputs the stored logic. Note that the T_{delay} depends on the array size and quantum Hall voltage amplitude in practice, and the read-current pulse width is set to $5ns$ to secure a successful readout operation.

3.3.2.6 Performance Comparison of TI-based Memory Array

Here we further investigate the TI based memory array design as shown in Figure 3.15. The circuit settings are same with section 3.3.2.5. The transient analysis conducted is to write 1001 to *word0* as discussed. The *word0* is initialized 0110 for better illustration, the other bits are initialized all zeros.

Figure 3.19 shows the timing diagram of write and read operations. Note that the positive and negative current are indicated by 1 and -1 in this figure. It can be observed

Figure 3.18: Read operation for topological insulator based memory cell

that the write operation is executed in two phases. First, the first bit and fourth bit of *word0* are written to 1 in the first 10ns. From 10ns to 20ns, the second and third bits are written to 0. To read out *word0*, *word0* is selected through word-line, and bit-lines of all bits are applied with read current pulse. A correct readout can be observed. It has also been verified that all the other bits remain their initial values.

Figure 3.19: Timing diagram for word write and read operations in a 4×4 TI array

Table 3.3: Performance comparison for different memory technologies

Memory Technology	Write Latency (ns)	Read Latency (ns)	Write Energy (J/bit)
SRAM	0.2	0.2	5e-16
DRAM	2-10	2-10	4e-15
PCM	100	12	6e-12
STT-MTJ	35	35	2.5e-12
FeRAM	40	60	3e-14
TI	20	5	1e-17

Table 3.3 shows the comparison of performance for different memory technologies.

Compared with the other emerging NVMs, such as phase change memory (PCM), spin-transfer torque magnetic tunneling junction (STT-MTJ), and ReRAM, the TI device based memory shows both faster read and write latencies. It is also noticed that the TI-based memory exhibits in several orders of magnitude lower write energy. The write energy of TI is calculated by Equation 3.22 with device dimension. Actually, the read energy of TI is also extremely low due to the non-scattering property. Simulation shows a read energy of $1.2e - 17J/bit$. The data for other memory technologies is extracted from ITRS 2011 [90].

3.3.3 Racetrack and Domain-wall

Domain-wall nanowire, also known as racetrack memory [9, 10, 91], is a newly introduced non-volatile memory device in which multiple bits of information are stored in single ferromagnetic nanowire. As shown in Figure 3.20(a), each bit is denoted by the leftward or rightward magnetization direction, and adjacent bits are separated by domain-walls. By applying a current through the shift port at the two ends of the nanowire, all the domain-walls will move left or right at the same velocity while the domain width of each bit remains unchanged, thus the stored information is preserved. Such a tape-like operation will shift all the bits similarly like a shift register.

In order to access the information stored in the domains, a strongly magnetized ferromagnetic layer is placed at desired position of the ferromagnetic nanowire and is separated by an insulator layer. Such a sandwich-like structure forms a magnetic-tunnel-junction (MTJ), through which the stored information can be accessed. In the following, the write, read and shift operations are modeled respectively.

3.3.3.1 Magnetization Reversal

The write access can be modeled as the magnetization reversal of MTJ free layer, i.e. the target domain of the nanowire. Note that the dynamics of magnetization reversal can be described by the precession of normalized magnetization m , or state variables θ and ϕ in spherical coordinates as shown in Figure 3.20(b). The spin-current induced magnetization dynamics described by θ and ϕ is given by

$$\theta = \theta_0 \exp\left(-\frac{t}{t_0}\right) \cdot \cos(\phi) \quad (3.27)$$

$$\omega = \frac{d\phi}{dt} = k_1 \sqrt{k_2 - (k_3 - k_4 I)^2} \quad (3.28)$$

Figure 3.20: (a) Schematic of domain-wall nanowire structure with access port and shift port; (b) magnetization of free-layer in spherical coordinates with defined magnetization angles; and (c) typical R-V curve for MTJ

where θ_0 is the initial value of θ , slightly tilted from the stable x or $-x$ directions; t_0 is precession time constant; ω is the angular speed of ϕ ; k_1 to k_4 are magnetic parameters with detailed explanation in Appendix B; and I is the spin-current that causes the magnetization precession.

3.3.3.2 MTJ Resistance

A typical R-V curve for MTJ is shown in Figure 3.20(c) with two regions: giant magnetoresistance (GMR) region and tunneling region. Depending on the alignment of magnetization directions of the fixed layer and free layer, parallel or anti-parallel, the MTJ exhibits two resistance values R_l and R_h . As such, the general MTJ resistance can be calculated by the giant magnetoresistance (GMR) effect

$$R(\theta_u, \theta_b) = R_{l0} + \frac{R_{h0} - R_{l0}}{2} (1 - \cos(\theta_u - \theta_b)) \quad (3.29)$$

where θ_u and θ_b are the magnetization angles of upper free layer and bottom fixed layer, R_{l0} and R_{h0} are the MTJ resistances when the applied voltage is subtle. When the applied voltage increases, there exists tunneling effect caused voltage-dependent resistance roll-

off,

$$\begin{cases} R_l(V) = \frac{R_{l0}}{1 + c_l V^2} \\ R_h(V) = \frac{R_{h0}}{1 + c_h V^2} \end{cases} \quad (3.30)$$

where c_l and c_h are voltage-dependent coefficients for parallel state and anti-parallel state, respectively.

3.3.3.3 Domain-wall Propagation

Like a shift register, the domain-wall nanowire shifts in a digital manner, thus could be digitalized and modeled in the unit of domains, in which a bit is stored. Note that except the bit in the MTJ, the other bits denoted by the magnetization directions are only affected by their adjacent bits. In other words, the magnetization of each bit is controlled by the magnetization in adjacent domains. Inspired by this, we present a magnetization controlled magnetization (MCM) devices based behavioral model for domain-wall nanowires. Unlike the current-controlled and voltage-controlled devices, the control in MCM device needs to be triggered by rising edge of one SHF-signal, which can be formulated as

$$\begin{aligned} \theta &= f(T_{sl}, \theta_r, T_{sr}, \theta_l, \theta_c) \\ &= T_{sl}\theta_r + T_{sr}\theta_l + \overline{T}_{sl}\overline{T}_{sr}\theta_0. \end{aligned} \quad (3.31)$$

in which T_{sl} and T_{sr} are the shift-left and shift-right commands; θ_r and θ_l are the magnetization angles in right adjacent cell and left adjacent cell respectively; θ_c is the current state before the trigger signal. This describes that the θ -state will change when triggered and will remain state if no shift-signal is issued.

For the bit in MTJ, the applied voltage for spin-based read and write will also determine the θ -state as discussed previously. Therefore we have,

$$\theta = f(T_{sl}, \theta_r, T_{sr}, \theta_l, \theta_0) + g(V_p, V_n, \theta_c) \quad (3.32)$$

where V_p and V_n are the MTJ positive and negative nodal voltages, and $g(V_p, V_n, \theta_0)$ is the additional term that combines Equation 3.27 to 3.30.

In addition, the domain-wall propagation velocity can be mimicked by the SHF-request frequency. The link between the SHF-request frequency and the propagation velocity is experimentally observed by current-velocity relation [92],

$$v = k(J - J_0), \quad (3.33)$$

where J is the injected current density and J_0 is the critical current density.

By combining equations (3.27) to (3.32), with magnetization angles θ and ϕ as non-volatile state variables other than electrical voltages and currents, one can fully describe the behaviors of the domain-wall nanowire device, where each domain is modeled as the proposed MCM device. As such, the modified nodal analysis (MNA) can be built in NVM-SPICE to verify circuit designs by domain-wall nanowire devices.

3.3.4 Spintronics Model Extension

Unlike ReRAM whose physics is still in debate, the physics of spintronics device category is relatively well understood. Similar to the ReRAM model extension in Section 3.2.3, two key links are required to establish well formulated models. The principle for the spintronics device category is magnetization switching, which is governed by the LLG equation with spin-transfer torque term (Eq. 3.21) and GMR effect. LLG equation describes the link between spin-polarized current and non-volatile state variable magnetization orientation. GMR effect depicts the link between resistance and state variable magnetization orientation.

To support process variations, the geometrical parameters need to be included. For resistance side, the elliptical domain size of MTJ will affect the energy required to switch the magnetization. And the tunnel-barrier thickness exponentially determines the tunneling resistance of MTJ. In addition, the geometry and oxide barrier thickness also affect the critical write current of MTJ, thus introducing variations on the switching time and switching probability. Furthermore, the temperature also has great impact on switching time/probability and on/off resistance. For the parameters not implemented in the NVM-SPICE models, their relations with existing parameters can be specified in net-list and Monte-Carlo simulations can be performed accordingly.

Chapter 4

Non-volatile Circuit Design

4.1 Memory and Readout Circuit

4.1.1 Crossbar Resistive Memory

Crossbar memory, also known as cross-point memory, structure has been popular ever since the advent of memristor array in crossbar structure fabricated by HP Labs [8]. Crossbar structure is mostly associated with resistive-ram (RRAM) devices with non-linear I-V characterization, memristor and CBRAM for instance, so that the half selection scheme can be applied. This leads to the most noticeable feature of crossbar, that is, unlike the conventional 1T1C/1T1R structure, access transistors are not required for every cell. In this section, we will introduce the crossbar based memory design.

4.1.1.1 Crossbar Memory Array

Fig. 4.1 illustrates the design for CBRAM-crossbar structure. Compared to the 1T1R structure, the crossbar structure has two major advantages. Firstly, an extremely high integration density can be realized due to the small pitch-size of nano-bars [8, 93]. Secondly, the crossbar structure can be stacked on top of the active transistor layer in a 3D fashion[94, 27, 95], which further reduces the area overhead and improves communication bandwidth. Fig. 4.2(a) shows the approach to fabricate the CBRAM-crossbar structure within the interconnect layers, where the CBRAM devices are deposited at the bottom of the copper vias [94]. Another approach of CBRAM-crossbar fabrication, shown in Fig. 4.2(b), is to stack the crossbar structure on top of the interconnect layer, which incurs the least modifications to the conventional CMOS process [27].

The conventional voltage-divider based readout design is shown in Fig. 4.3, where R_l follows $\frac{R_{off}}{R_l} = \frac{R_l}{R_{on}}$. As such, V_o will be logic-1 when target-cell is in low-resistance-state

Figure 4.1: Crossbar operations and peripheral circuits

(LRS) and be logic-0 when in high-resistance-state (HRS). Note that all the unselected word-lines and bit-lines need to be floated to avoid sensing current branches before reaching R_t . This will incur sneak-path issue. When the target-cell is in HRS and is surrounded by other cells in LRS, there will be a significant leaked current flowing through the neighboring cells, which may lead to misinterpretation of the stored bit. The sneak-path issue can be addressed by adding a selection device for each CBRAM device [96, 97]. The selection device works like bidirectional-diode to ensure no current flowing through paths with more than one CBRAM device.

Instead of applying selection devices, alternative operations are proposed here for the CBRAM-crossbar readout circuit to avoid the sneak-path issue. The *write* and *read* operations for CBRAM-crossbar are shown in Fig. 4.1(a) and 4.1(b), respectively. To write CBRAM-crossbar, the bias-voltage v_w needs to be applied on the designated cell through multiplexed voltage selector. As such, the corresponding word-line and bit-line are applied with $v_w/2$ and $-v_w/2$ voltages, respectively. Such method is called half selection. By controlling the voltage level and polarity, one can determine the write-speed and change the on/off states of the cell. The cells that are half-selected will not change their states. To read CBRAM-crossbar, the bias-voltage v_r (normally $v_r < v_w$) is applied on the corresponding word-line. By measuring the current of designated column, the on/off state of target-cell can be detected. Note that because all unselected word-lines and bit-lines are grounded rather than floated, the voltage-divider based readout scheme

Figure 4.2: Two approaches for 3D stacked CBRAM-crossbar fabrication (a) crossbar structure integrated within interconnect layers, and CBRAM devices fabricated at the bottom of vias (b) crossbar structure fabricated on the top of interconnect layers

Figure 4.3: Conventional crossbar read operation with incurred sneak-path issue

is incapable to function. As such, more sophisticated readout circuit design is required and is discussed later in this section.

Fig. 4.1(c) further shows the peripheral circuit design for one CBRAM-crossbar. During write operation, the corresponding word-line and bit-line are applied with writing voltage through address decoding. The voltage selection is done by the row drivers and column drivers, which switch among different voltage supplies according to the command issued. During read operation, word-lines still need voltage selection while bit-lines are switched to current sensing circuit.

4.1.1.2 Crossbar Readout Circuit

To sense the current of designated column in the crossbar structure, readout circuit illustrated in Fig. 4.4 is proposed. The sensing is done in two steps here. Firstly, a current mirror is deployed to amplify the current determined by the state of the target-cell. The bias current I_{bias} is applied to ensure that transistor M1 works in saturation region. The bias voltage V_{bias} has to be deliberately chosen according to I_{bias} to achieve a virtually grounded node A, i.e. a 0V column voltage required in Fig. 4.1(b). Otherwise, the current to detect will have to branch at node A to other cells at same column, which will weaken the current signal and thus lead to a degradation of sensing accuracy. Secondly, the amplified current signal is converted into voltage signal, and is further compared with the reference voltage to decide and output on/off state denoted by different logic levels.

Figure 4.4: Readout circuit design for CBRAM crossbar structured memory

Another issue in readout operation is the severe device resistance variations. It is reported that $5x \sim 100x R_{off}$ and $2x \sim 10x R_{on}$ variations can be expected based a variety of measurement data [98]. For one crossbar array, there exists a valid reference voltage

in the readout circuit only when the minimum R_{off} of all CBRAM devices is greater than the maximum R_{on} of all devices, i.e.

$$\min(R_{off}) > \max(R_{on}) \quad (4.1)$$

which, however, may not hold true if variation is significant. A crossbar with device resistance variations that cannot meet above condition is called a failure. As such, the sense amplifier is not able to guarantee successful readout for every operation. In the following, we will show the failure rate for different crossbar array sizes, and possible solutions to address such issue when failure rate is high.

Figure 4.5: Monte Carlo simulation of 100×100 crossbar-array with device resistance variations

Monte Carlo simulation with 5000 times is conducted in order to investigate the device resistance variations impact on readout operation, which is shown in Fig. 4.5. Accordingly to [98], the typical on-state resistance variation ranges from 2x to 10x and off-state resistance variation from 5x to 100x. Therefore, in each iteration, a 100×100 crossbar-array is generated, where R_{off} and R_{on} of all CBRAM devices have a normal distribution with μ of $1G\Omega$ and $2M\Omega$, and deviation δ $0.2G\Omega$ for R_{off} and $1M\Omega$ for R_{on} . The $\max(R_{on})$ and $\min(R_{off})$ are calculated and then used as the x-axis and y-axis values for each point. The whole domain is divided into two regions, pass region and fail region, separated by the dashed line $\max(R_{on}) = \min(R_{off})$. It can be observed that there are 9 failed cases, which lead to a 0.18% failure rate for the 100×100 crossbar size. In other words, 9 out of 5000 crossbars in this size do not have the ideal reference

voltage value.

The Monte Carlo simulations for crossbar sizes starting at 50×50 to 600×600 with a step of 50 are conducted with results shown in Table 4.1. It can be observed that the failure rate increases as the crossbar size enlarges. For large crossbar arrays where the failure rate is not negligible, robust readout schemes namely the self-reference [99] and runtime ECC/ECP [100] can be applied to increase the readout reliability. Thus it is favorable to limit the crossbar size within a few hundreds by a few hundreds.

Table 4.1: Device resistance variation caused sense amplifier failure rate for different crossbar-array sizes

Array size	50^2	100^2	150^2	200^2	250^2	300^2
Failure rate	0.05%	0.18%	0.85%	0.95%	2.65%	3.20%
Array size	350^2	400^2	450^2	500^2	550^2	600^2
Failure rate	4.30%	5.50%	5.55%	7.65%	10.85%	13.40%

4.1.1.3 Crossbar Memory Modeling

Based on the circuit design for CBRAM-crossbar, here we propose its system-level delay, power and area models like the CMOS memory design tool CACTI [101]. We further verify such models by the transistor-level SPICE-level simulator developed in the previous section.

Delay Model Generally, the time needed for write operation in CBRAM-crossbar is the signal propagation delay on wires (i.e. word-line and bit-line), denoted by D_{wire}^w , plus the device switching time $D_{switching}$; for read operation, it is composed of wire delay D_{wire}^r and sensing delay $D_{sensing}$. Therefore we have

$$D_{write} = D_{wire}^w + D_{switching} \quad (4.2)$$

$$D_{read} = D_{wire}^r + D_{sensing}. \quad (4.3)$$

Different from DRAM/SRAM cell, the CBRAM device has asymmetrical write/read-delays. Since the write operation requires physical change of CBRAM cell, i.e. the shape-morphing of the conductive filament as illustrated in Fig. 3.6, it usually takes much longer time than simply detecting the resistance of CBRAM cell in the read operation. The CBRAM switching time $D_{switching}$ can be obtained from the proposed CMOS-CBRAM simulator, where high accuracy can be achieved thanks to the use of physical model developed in last chapter.

Figure 4.6: RC-delay model of one CBRAM-crossbar for: (a) read operation; (b) write operation

Due to the existence of leakage current in crossbar structure, CBRAM cells at different positions suffer from different amount of reduced applied voltages introduced by IR-drop along the word-line and bit-line. Since the switching time is very sensitive to even the slightest of voltage deviation from the expected value, CBRAM cells at different positions of the crossbar array exhibit different values of switching time. Therefore, the exponential relation between $D_{switching}$ and the applied voltage v_w , represented as a lookup table, is built into CACTI.

Different from SRAM/DRAM sensing schemes where the subtle voltage/current swing or capacitor driving signal needs to be detected, the current signal in Fig. 4.4 is amplified and driven by power source thus the sensing can be performed really fast. Note that the sensing delay can also be obtained by performing SPICE-level simulation. In the following, we focus on the modeling and calculation of wire delay D_{wire} for read/write operations.

For conventional 1T-1R structure, word-line delay can be calculated by distributed RC-line delay and the bit-line delay can be estimated by Seevinck model [102]. However, although the same approach has been applied to estimate the wire delay of crossbar structure in [103], it lacks accuracy due to the following reasons. Firstly, since there is no transistor in a crossbar, the word-line and bit-line delay are symmetric. In other words, the word-line and bit-line delay have to be modeled in the same manner. Moreover, the leakage current of cells along word-line and bit-line is a phenomenon specific to crossbar structure, which is not considered by conventional RC-line delay model. The leakage current will weaken the driving ability of row and column drivers, and hence a longer delay can be predicted compared to the conventional RC-line delay.

Fig. 4.6(a) and Fig. 4.6(b) illustrate the crossbar delay model for read/write operation, with leakage path of cell i along the word-line and bit-line modeled as parallel R_{li} , whose value is its corresponding CBRAM on/off resistance. For the read operation, the bit-line is virtually grounded as illustrated in Fig. 4.1, thus only the word-line delay for the propagation of v_r contributes to the wire delay. For the write operation, both word-line driving voltage $v_w/2$ and bit-line voltage $-v_w/2$ will propagate to target cell, and the total wire delay is determined by the slower one. Therefore, for a CBRAM-crossbar with m rows and n columns, the worst case wire delay for both read operation and write operation can be calculated respectively by

$$D_{wire}^r = \alpha RCn^2 \quad (4.4)$$

$$D_{wire}^w = \max(\alpha RCn^2, \alpha RCm^2) \quad (4.5)$$

where R and C are parasitic resistor and capacitor in unit length similar as the distributed RC-line model. Note that when the CBRAM device is scaled down, the R and C will be reduced accordingly, which produces a smaller wire delay. In addition, the location-dependent switching speed issue incurred by the IR-drop along the word-line and bit-line will be relieved as well.

Compared to the conventional RC-line delay expression, a fitting parameter α has been added to approximate the expected longer delay due to the CBRAM-crossbar structure, such as the effect introduced by R_{li} in Fig. 4.6. Practically, α can be obtained by fitting with a few samples obtained by simulating entire CBRAM-crossbar in different sizes using the developed SPICE-like simulator.

Fig. 4.7 shows the verification of the proposed crossbar specific delay model against accurate simulation results obtained through the developed SPICE-like simulator. It can be observed that the proposed model with fitting parameter $\alpha = 1.2$ is able to predict the crossbar delay well. As expected in section 4.1.1.3, the crossbar wire delay, calculated by $1.2RCn^2$, is thereby more than twice longer compared to the conventional distributed RC-line delay calculated by $0.5RCn^2$. In other words, an error of more than 50% will be incurred if the conventional distributed RC-line delay model is used instead.

Power Model The energy per write-access for CBRAM-crossbar is composed of two parts: the energy consumed to switch the target cell state, and the energy dissipated along the word-line and bit-line. Consider a CBRAM-crossbar with m rows and n columns, the write-access energy E_{write} can be calculated as:

$$E_{write} = E_{switching} + E_{static}^w + E_{dynamic}^w \quad (4.6)$$

Figure 4.7: Verification of proposed CBRAM-crossbar specific wire delay model against simulation results, with fitting parameter $\alpha = 1.2$

where $E_{switching}$ is the energy for changing the target CBRAM cell state, and needs to be obtained through the developed SPICE-like simulator since its dynamics is hard to be approximated using resistor; $E_{dynamic}^w$ is the energy for charging parasitic capacitors along word-line and bit-line; and E_{static}^w is Joule heat dissipated on the half-selected CBRAM cells along word-line and bit-line, and can be modeled as resistors since their resistance values remain constant during write operation.

The E_{static}^w and $E_{dynamic}^w$ can be calculated by

$$E_{static}^w = \left(k \cdot \frac{v_w^2}{4 \cdot R_{on}} + l \cdot \frac{v_w^2}{4 \cdot R_{off}} \right) \cdot D_{write} \quad (4.7)$$

$$E_{dynamic}^w = \frac{1}{8} C \cdot v_w^2 \cdot (m + n - 2) \quad (4.8)$$

where v_w is the write-voltage, R_{on} and R_{off} are the *on/off* state resistance of CBRAM, D_{write} is the crossbar write-delay, C the distributed unit capacitance of crossbar wire, k and l the numbers of CBRAM cells in ON-state and OFF-state respectively along the path, following $k + l = m + n - 2$.

Similarly, the read-access energy E_{read}^r for crossbar can be calculated by

$$E_{read} = E_{static}^r + E_{dynamic}^r \quad (4.9)$$

Note that for read operation, all the bit-lines are virtually grounded and only cells in the target row consume power as shown in Fig. 4.1, thus we have

$$E_{static}^r = \left(k \cdot \frac{v_r^2}{R_{on}} + l \cdot \frac{v_r^2}{R_{off}} \right) \cdot D_{read} \quad (4.10)$$

$$E_{dynamic}^r = \frac{1}{2} C \cdot v_r^2 \cdot n \quad (4.11)$$

where v_r is the read-voltage, and D_{read} is the crossbar read-delay. Moreover, k and l are the numbers of CBRAM cells in ON-state and OFF-state in the target row respectively, and they satisfy $k + l = n$. Note that the scalability of the CBRAM device is beneficial to dynamic power reduction. When scaled down, the smaller pitch size will reduce the wire capacitance with the power reduction at the word-line and bit-line.

Figure 4.8: Verification of proposed CBRAM-crossbar specific power model for read/write operations against simulation results

The verification of the power model against simulation results is shown in Fig. 4.8. The simulation results are obtained by simulating $n \times n$ square CBRAM-crossbars with different n values, where n is used to denote crossbar size. It can be observed that the proposed model for read/write operations is able to capture the trend with minor error when crossbar size increases.

Area Model The area model consists of two parts: A_c for the pure area crossbar structure and A_p for corresponding CMOS peripheral circuits. Utilizing the 3D integration technique shown in Fig. 4.2, the crossbar is stacked over the active layer where its

peripheral circuits are located. As such, the total area becomes

$$A = \max(A_p, A_c). \quad (4.12)$$

For a CBRAM-crossbar with M rows and N columns, its area can be calculated by

$$A_c = M \cdot N \cdot L_{pitch}^2 \quad (4.13)$$

where L_{pitch} is the nano-bar pitch size, determined by the technology node of the CBRAM device. Therefore, at advanced technology node, extremely small CBRAM-crossbar area can be achieved due to the scalability of the CBRAM device. Besides the reduced wire delay inside the CBRAM-crossbar, the addressing delay outside the CBRAM-crossbar can be greatly reduced as well, which together will lead to significant memory access latency reduction. Note that a similar area model can be developed for the peripheral circuits based on [101].

4.1.1.4 Crossbar Memory Performance Evaluation

By integrating above system-level delay, power and area models into memory performance evaluation tool CACTI [101], the CBRAM performance comparison with other memory technologies can be made. Table 4.2 evaluates and compares the performance of the stacked CBRAM-crossbar memory with the other memory technologies for the same capacity of 16MB in 32nm technology node. The data for SRAM and DRAM are generated by CACTI with default settings, and PCRAM data is extracted from PCRAM-sim [104]. As shown in Table 4.2, mainly due to the fast device-level accessing speed by CBRAM device as well as the high density of crossbar structure, the CBRAM-crossbar performance, especially the accessing latency for read/write operations, is already close to DRAM, which shows its potential for the future application as the main memory. Moreover, compared to PCRAM, the CBRAM-crossbar shows 9x faster write-latency, 1.6x smaller area, 4.5x less write-energy per access, and 1.5x slower read-latency on average. Only a slightly slower read-latency is observed for the CBRAM-crossbar.

4.1.2 1T-1R Spintronic Memory

Thanks to the strongly nonlinear I-V curve of ion migration kinetics, ReRAM devices have threshold effect so that half selection technique can be applied. This enables crossbar memory structure and transistor free feature. This is why crossbar memory architecture is often associated with ReRAM technologies. Spintronic memory, on the other

Table 4.2: Performance comparison of 16MB SRAM, DRAM, PCRAM, and CBRAM memories

Feature	SRAM	DRAM	PCRAM	CBRAM
area (mm ²)	27.4	4.19	3.77	2.33
read latency (ns)	3.43	2.25	2.54	3.90
write latency (ns)	3.43	2.55	RESET 42.54 SET 102.54	8.01
read energy (nJ)	0.83	0.61	0.73	1.8
write energy (nJ)	0.75	0.61	RESET 6.66 SET 2.28	2.0

hand, is similar to conventional SRAM/DRAM technologies which require transistors to control. Thus spintronic memory is often associated with 1T-1R structure, where ‘T’ stands for transistor and ‘R’ denotes one non-volatile device whose state is represented by resistance.

4.1.2.1 STT-RAM Memory

Figure 4.9: Circuit diagrams of 1T-1R STT-RAM memory cell: (a) simplified memory cell for write 1; (b) simplified memory cell for write 0; and (c) 16-bit STT-RAM with 4 bit-lines and 4 word-lines

A typical hybrid STT-RAM cell is shown in Figure 4.9 (a) with one transistor and one STT-MTJ in series connection. The structure is identical to that of DRAM cell except that the capacitor is replaced by STT-MTJ device. The gate of transistor is connected to word-line, which serves to select target cells in same word-line. When enable, two bit-lines (named bit-line and select line to distinguish) can be driven to have V_w or $-V_w$ depending on the desired data to write, 1 or 0. In the ‘write 1’ operation, WL is connected to VDD, and BL and SL are connected to V_{DD} and ground, respectively. In the ‘write 0’ operation the polarities of SL and BL line are interchanged. Readout operation can be performed in similar way by using V_r , and the current loop will eventually be measured by readout circuit, which determines device state by current amplitude.

4.1.2.2 STT-RAM Readout Circuit

Existing STT-RAM readout schemes to avoid disturbance of large STT-MTJ resistance variation usually require several steps, which slows down the read latency. We show that by applying a single-sawtooth pulse and exploiting the resistance roll-off of STT-MTJ, the robust readout can be achieved within one cycle.

Basic STT-RAM Readout Circuit The basic voltage sensing scheme for the popular 1T-1MTJ structure STT-RAM is shown in Figure 4.10 (a). The reference voltage is set to satisfy

$$I_r \cdot (R_{AP} + R_t) > V_{ref} > I_r \cdot (R_P + R_t)$$

where I_r is the applied read current, R_{AP} , R_P and R_t are the MTJ anti-parallel state resistance, parallel state resistance and cell transistor ON-state r_{ds} , respectively. However, in the presence of bit-to-bit MTJ resistance variation, the reference voltage has to fulfill

$$\text{Min}(V_{BL,AP}) > V_{ref} > \text{Max}(V_{BL,P})$$

where a satisfying V_{ref} may not exist when variation is large.

Destructive Self-reference Readout Circuit In order to achieve reliable readout in the presence of large MTJ resistance variation, a self-reference readout is presented in [99], whose diagram is shown in Figure 4.10 (b). The read-operation is done in five phases:

- the read-current I_r is applied and its bit-line voltage is stored in C1;
- the “0” (parallel state) value is written to the target cell;

Figure 4.10: The existing schemes for STT-RAM readout: (a) basic STT-RAM readout (b) destructive self-reference readout in [99] (c) non-destructive self-reference readout in [105]

- the read-current I_r is applied again and its bit-line voltage is stored in C2;
- the sense amplifier is enabled and voltages of C1 and C2 are compared; and the output is “1” (anti-parallel state) if V_{C1} is greater than V_{C2} and “0” otherwise;
- the output value has to be written back to the destructed cell.

Therefore in terms of both speed and power, the overhead brought by write-back may be large for this scheme.

Non-destructive Self-reference Readout Circuit The current-dependent resistance roll-off can be observed for STT-MTJ as shown in Figure 4.11. By exploiting the fact that the roll-off slope of the anti-parallel state is much greater than that of parallel state, a non-destructive self-reference readout is proposed in [105] with diagram shown in Figure 4.10 (c).

The read-operation is done in three phases:

- the read-current I_{r1} is applied to achieve its corresponding resistance R_1 ;
- the read-current I_{r2} is applied to achieve its corresponding resistance R_2 ;
- the sense amplifier is enabled and R_1 and R_2 are compared; The output is “1” if two values are significantly different and “0” otherwise.

As such, the non-destructive self-reference readout can improve the read latency by eliminating the two time-consuming write phases. However, this scheme still has limited performance in read latency and sensing margin as discussed below.

Figure 4.11: The measured R-I sweep curve of a typical MgO based MTJ in [99]

4.1.2.3 Single Sawtooth-pulse based Readout Circuit

Although the variation can be overcome by the two self-reference schemes above, they still involve several phases which slow down the read latency. A single-sawtooth pulse based readout is proposed in Figure 4.12 to reduce the read latency into one cycle. In the following, we will show that within one cycle, by applying a single-sawtooth pulse to bit-line and obtaining the second derivative of corresponding bit-line voltage, the variation disturbance during readout can be totally avoided.

Assume the applied sawtooth pulse to bit-line can be expressed as

$$i(t) = k_s \cdot t \quad (4.14)$$

where the k_s denotes the current rising rate. Also, the R-I curve slope in Figure 4.11 is assumed linear for simplification, thus the current-dependent resistance can be expressed

Figure 4.12: Diagram of proposed single-sawtooth pulse based readout

as

$$\begin{aligned} R_{AP}(i) &= R_H - k_{AP} \cdot i \\ R_P(i) &= R_L - k_P \cdot i \end{aligned} \quad (4.15)$$

where R_H and R_L are the resistances of R_{AP} (anti-parallel) and R_P (parallel) when $i = 0$. Therefore, the bit-line voltage produced by the applied sawtooth pulse can be expressed as

$$\begin{aligned} V_{BL,AP}(t) &= i(t) \cdot R(i) = R_H \cdot k_s \cdot t - k_{AP} \cdot k_s^2 \cdot t^2 \\ V_{BL,P}(t) &= i(t) \cdot R(i) = R_L \cdot k_s \cdot t - k_P \cdot k_s^2 \cdot t^2 \end{aligned} \quad (4.16)$$

It can be observed that the bit-line voltage depends on the R_H and R_L , which will introduce readout errors in the presence of large variations. Nevertheless, the readout dependency on R_H and R_L can be eliminated if the second derivative of bit-line voltage can be obtained

$$\begin{aligned} \frac{d^2 V_{BL,AP}}{dt^2} &= -2 \cdot k_{AP} \cdot k_s^2 \\ \frac{d^2 V_{BL,P}}{dt^2} &= -2 \cdot k_P \cdot k_s^2 \end{aligned} \quad (4.17)$$

Note that the current-dependent resistance roll-off slopes k_{AP} and k_P are not fabrication-process sensitive, and k_P is a close-to-zero while k_{AP} is much larger as indicated in Figure 4.11, the robust readout can be easily achieved under the proposed scheme right after the sawtooth pulse is applied. Thus compared with previous work where several steps are required, the proposed scheme can potentially reduce the read latency into one cycle time

The circuit to implement second-derivative operation can be designed as two differentiators in series, and each differentiator can be implemented as either an OPAMP-based feedback circuit or a RC-based high-pass filter. RC-based high-pass filter provides simple differentiation but has limited gain, while OPAMP-based one can generate out-

Figure 4.13: Transient response of bit-line voltage, first derivative, and second derivative to the applied sawtooth pulse for pure OPAMP, hybrid and pure RC based circuit, respectively

put with significant sensing margin. Three different circuits to implement the second-derivative operations with trade-off between circuit complexity and readout sensing margin as follows:

- Pure OPAMP: two OPAMP-based in series;
- Hybrid: first-stage with OPAMP-based and second-stage with RC-based high-pass filter;
- Pure RC: two RC-based high-pass filters in series.

In this experiment, the hybrid approach is deployed for the single-sawtooth pulse based readout as shown in Figure 4.12.

The proposed single-sawtooth pulse based readout scheme in Figure 4.12 along with the STT-RAM array circuit are simulated together using NVM-SPICE intrinsic STT-MTJ model. The STT-MTJ model parameters are set with $cap = 0.9$, $cp = 0.1$, $rap = 2650$ and $rp = 1230$ with explanations for each parameter in Table A.2; the BSIM4.7 model is used for all the transistors with $L = 90nm$ and $W = 2\mu m$; the $I_w = \pm 900\mu A$ is

used for write operation, and read current I_r rises from 0 to $200\mu A$ within 25ns, which produces a sawtooth pulse with $k_s = 8000A/s$.

Table 4.3: Comparison of different readout schemes for STT-RAMs

Scheme	Read latency	Sense margin (mV)
destructive readout [99]	2 read cycles + 2 write cycles	76.6
non-destructive readout [105]	2 read cycles	12.1
proposed	1 read cycle	15 (hybrid) 100 (OPAMP)

The single-sawtooth pulse based readout with second derivative circuit achieved in all three ways are simulated with results shown in Figure 4.13. It can be observed that for the bit-line response to the applied sawtooth pulse current, the bit-line of AP state STT-MTJ exhibits some non-linearity while that of P state is almost linear. The first derivative of bit-line signals shows larger difference between AP state and P state cases, where the that of P state is almost constant while that of AP state has a considerable slope. The output difference in this stage still cannot avoid the readout fault caused by resistance variation according to Equation 4.16. The second derivation of bit-line voltage, which is variation tolerant according to Equation 4.17, shows separated voltage level for AP state and P state, which is also easy to distinguish by later sensing stage. The pure OPAMP based one can produce voltage levels separated by around 200mV between AP state and P state, and that of hybrid based one is around 30mV, and less than 0.2mV for pure RC based one. Table 4.3 shows the readout performance comparison between the proposed scheme and previous work.

To achieve single cycle readout, the trade-off made is the increased design cost brought by OPAMP. With negative feedback applied in OPAMP to maintain high gain for accurate differentiation, higher power consumption overhead may be incurred. The two RC differentiators based approach, on the other head, has the lowest design cost but the initial output swing is insignificant. As such, a positive feedback based latch can be applied to magnify the small output signal.

4.1.3 Domain-wall Spintronic Memory

Compared with the conventional SRAM or DRAM by CMOS, the domain-wall nanowire based memory (DWM) can demonstrate two major advantages. Firstly, extremely high integration density can be achieved since multiple bits can be packed in one macro-cell. Secondly, zero standby power can be expected as a non-volatile device does not require

to be powered to retain the stored data. In addition, the resistance-detection based read-out does not require bit-line pre-charging, which avoids the sub-threshold leakage of the access transistors. In this section, we will present DWM-based design with macro-cell memory: structure, modeling, and data organization.

4.1.3.1 DWM Memory Cell

Figure 4.14: Macro-cell of DWM with: (a) single access-port; and (b) multiple access-ports

Figure 4.14(a) shows the design of domain-wall nanowire based memory (DWM) macro-cell with access transistors. The access-port lies in the middle of the nanowire, which divides the nanowire into two segments. The left-half segment of nanowire is used for data storage while the right-half segment is reserved for shift-operation in order to avoid information lost. In order to access the left-most bit, the reserved segment has to be at least as long as data segment. In such case, the data utilization rate is only 50%. In order to improve the data utilization rate, a multiple port macro-cell structure is presented in Figure 4.14(b). The access-ports are equally distributed along the nanowire, which divides the nanowire into multiple segments. Except the right-most segment, all other segments are data segments with the bits in one segment form a *group*. In such case, to access arbitrary bit in the nanowire, the shift-offset is always less than the length of one segment, thus the data utilization rate is greatly improved.

Thus, the number of bits in one macro-cell can be calculated by

$$N_{cell-bits} = (N_{rw-ports} + 1)N_{group-bits} \quad (4.18)$$

In which $N_{rw-ports}$ is the number of access ports. Then the macro-cell area can be calculated by

$$A_{nanowire} = N_{cell-bits}L_{bit}W_{nanowire} \quad (4.19)$$

$$A_{cell} = A_{nanowire} + 2A_{shf-nmos} + 2A_{rw-nmos}N_{rw-ports} \quad (4.20)$$

where L_{bit} is the pitch size between two consecutive bits, $W_{nanowire}$ the width of domain-wall nanowire, $A_{shf-nmos}$ and $A_{rw-nmos}$ are the transistor size at shift-port and access-port respectively.

Moreover, the bit-line capacitance is crucial in the calculation of latency and dynamic power. The increased bit-line capacitance due to the multiple access-ports can be obtained by

$$C_{bit-line} = (N_{rw-ports}C_{drain-rw} + C_{drain-shf} + C_{bl-metal}) \times N_{row} \quad (4.21)$$

in which $C_{bl-metal}$ is the capacitance of bit-line metal wire per cell, the $C_{drain-rw}$ and $C_{drain-shf}$ are the access-port and shift-port transistor drain capacitances, respectively. Note that the undesired increase of per-cell capacitance will be suppressed by the reduced number of rows due to higher nanowire utilization rate.

Additionally, the domain-wall nanowire specific behaviors will incur in-cell delay and energy dissipation. The magnetization reversal energy $0.27pJ$ and delay $600ps$ can be obtained through the transient analysis by NVM-SPICE as discussed in chapter 3. The read-energy is in fJ scale thus can be omitted. Also, the read-operation will not contribute in-cell delay. The delay of shift-operation can be calculated by

$$T_{shift} = L_{bit}/v_{prop} \quad (4.22)$$

in which v_{prop} is the domain-wall propagation velocity that can be calculated by equation 3.33. The Joule heat caused by the injected current is calculated as the shift-operation dynamic energy.

Figure 4.15: Sensing circuit design for domain-wall nanowire

4.1.3.2 DWM Readout Circuit

The readout circuit for domain-wall memory is similar to that of STT-RAM, both based on the GMR effect. The basic readout circuit for DWM is shown in Figure 4.15. To obtain the bit information from a specific domain, the domain needs to be first shifted and aligned with the fixed layer so that the readout operation can be performed. Its resistance, determined by GMR effect, is then compared to the reference resistance, and the result that indicates its state will then be output. Note that the readout circuit in Figure 4.15 is the most basic readout circuit; more sophisticated readout techniques like self-reference readout and non-destructive readout circuit described in Section 4.1.2.2 can be also applied.

4.1.3.3 DWM Main Memory

There are two potential problems for above DWM macro-cell. Firstly, there exists variable access latencies for the bits that locate at different positions in the nanowire. Secondly, if the required bits are all stored in the same nanowire, very long access latency will be incurred due to the sequential access.

It is important to note that the data exchange between main memory and cache is always in the unit of a cache-line size of data, i.e. the main memory will be read-accessed when last-level cache miss occurs; and will be write-accessed when a cache-line needs to be evicted. Therefore, instead of the per access latency, the latency of the data block in the size of a cache-line becomes the main concern. Based on such fact, we present a cluster-group based data organization. The idea behind *cluster* is to distribute data in different nanowires thus they can be accessed in parallel to avoid the sequential access; and the idea behind *group* is to discard the within-group addressing, and transfer the $N_{group-bits}$ bits in $N_{group-bits}$ consecutive cycles, to avoid the variable latency. Specif-

ically, a cluster is the bundle of domain-wall nanowires that can be selected together through bit-line multiplexers. The number of nanowires in one cluster equals the I/O bus bandwidth of the memory. Note that the data in one cache-line have consecutive addresses. Thus, by distributing the bits of N consecutive bytes, where N is decided by the I/O bus bandwidth, into different nanowire within a cluster, the required N bytes can be accessed in parallel to avoid the sequential access. In addition, within each nanowire in the cluster, the data will be accessed in the unit of group, i.e. the bits in each group will be accessed in consecutive cycles with a similar fashion as DRAM.

The number of groups per nanowire is thus decided by

$$N_{group-bits} = N_{line-bits} / N_{bus-bits}. \quad (4.23)$$

For example, in system with cache-line size of 64-byte, and memory I/O bus bandwidth of 64-bit, the group size is 8-bit. As such, the DWM with cluster-group based data organization can be operated in the following steps:

- Step1: The group-head initially is aligned with the access-port, thus the distributed first 8 consecutive bytes can be first transferred between memory and cache;
- Step2: Shift the nanowire with 1-bit offset, and transfer the following 8 consecutive bytes. Iterate this step 6 more times until the whole cache-line data is transferred;
- Step3: After the data transfer is completed, the group-head is relocated to the initial position as required in step 1.

As mentioned in Equation 3.33, the current-controlled domain-wall propagation velocity is proportional to the applied shift-current. By applying a larger shift-current, a fast one-cycle cluster head relocation can be achieved. In such a manner, the data-transfer of cache block will be able to achieve a fixed and also lowest possible latency.

The domain-wall nanowire based memory differs from the conventional CMOS-based memory in many aspects, thus CACTI has been extended with the domain-wall nanowire based memory model for DWM based main memory, with accurate device operation energy and delay data obtained from NVM-SPICE. The memory configuration is shown in table 5.5

Table 4.4 shows the 128MB memory-bank comparison between CMOS-based memory (or DRAM) and domain-wall nanowire based memory (or DWM). The number of access ports in main memory is varied for design exploration. The results of DRAM are generated by configuring the original CACTI with 32nm technology node, 64-bit of I/O

bus width with leakage optimized. The results of the DWM are obtained by the modified CACTI with the same configuration.

Table 4.4: Performance comparison of 128MB memory-bank implemented by different structures

Memory structure	area (mm^2)	access energy (nJ)	access time (ns)	leakage (mW)
DRAM	20.5	0.77	3.46	620.2
DWM/1 port	8.9	0.65	1.90	48.4
DWM/2 ports	6.2	0.72	1.71	30.1
DWM/4 ports	6.2	0.89	1.69	24.3
DWM/8 ports	5.7	1.31	1.88	19.0

It can be observed that the memory area is greatly reduced in the DWM designs. Specifically, the DWMs with 1/2/4/8 access ports can achieve the area saving of 57%, 70%, 70% and 72%, respectively. The trend also indicates that the increase of number of access-ports will lead to higher area saving. This is because of the higher nanowire utilization rate, and is consistent with the analysis discussed previously. Note that the area saving in turn results in a smaller access latency, and hence the DWM designs on average provide 1.9x improvement on the access latency. However, the DWM needs one more cycle to perform shift operation, which will cancel out the latency advantage. Overall, the DWM and DRAM have similar speed performance. In terms of power, the DWM designs also exhibit benefit with significantly leakage power reduction. The designs with 1/2/4/8 access ports can achieve 92%, 95%, 96% and 97% leakage power reduction rates, respectively. The advantage mainly comes from the non-volatility of domain-wall nanowire based memory cells. The reduction in area and decoding peripheral circuits can further help leakage power reduction in DWM designs. In addition, the DWM designs have the following trend of access energy when increasing the number of access ports. The designs with 1/2 ports require 16% and 6% less energy, while designs with 4/8 ports incur 15% and 70% higher access energy cost. This is because when the number of ports increases, there are more transistors connected to the bit-line which leads to increased bit-line capacitance.

4.2 Non-volatile Domain-wall Logic Circuit

4.2.1 XOR

The magnetization switching with sub-nanosecond speed and sub-pJ energy have been experimentally demonstrated [106, 107, 108]. As such, the domain-wall nanowire

based logic can be further explored for logic-in-memory based computing. In this section, we show how to further build DWL based XOR-logic, and how it is applied for low-power ALU design for comparison and addition operations.

The GMR-effect can be interpreted as the bitwise-XOR operation of the magnetization directions of two thin magnetic layers, where the output is denoted by high or low resistance. In a GMR-based MTJ structure, however, the XOR-logic will fail as there is only one operand as variable since the magnetization in fixed layer is constant. Nevertheless, this problem can be overcome by the unique domain-wall shift-operation in the domain-wall nanowire device, which enables the possibility of DWL-based XOR-logic for computing.

Figure 4.16: Low power XOR-logic implemented by two domain-wall nanowires

A bitwise-XOR logic implemented by two domain-wall nanowires is shown in Figure 4.16. The proposed bitwise-XOR logic is performed by constructing a new read-only-port, where two free layers and one insulator layer are stacked. The two free layers are in the size of one magnetization domain and are from two respective nanowires. Thus, the two operands, denoted as the magnetization direction in free layer, can both be variables with values assigned through the MTJ of the according nanowire. As such, it can be shifted to the operating port such that the XOR-logic is performed.

For example, the $A \oplus B$ can be executed in the following steps

- The operands A and B are loaded into two nanowires by enabling WL_1 and WL_2 respectively;
- A and B are shifted from their access-ports to the read-only-ports by enabling SHF_1 and SHF_2 respectively;
- By enabling RD , the bitwise-XOR result can be obtained through the GMR-effect.

Note that in the x86 architecture processors, most XOR instructions also need a few cycles to load its operands before the logic is performed, unless the two operands are both in registers. As such, the proposed DWL-based XOR-logic can be a potential substitution of the CMOS-based XOR-logic. Moreover, similar as the DWM macro-cell, zero leakage can be achieved for such XOR-logic.

The transient analysis of the domain-wall nanowire XOR structure has been performed in the SPICE simulator. with both controlling timing diagram and operation details shown in Figure 4.17.

Figure 4.17: The timing diagram of DWL-XOR with SPICE-level simulation for each operation

The current density of $7e10A/m^2$ is utilized for magnetization switching. The θ states of the nanowire that takes A are all initialized at 0, and the one takes B all at π . Only two-bit per nanowire is assumed for both nanowires. The operating-port is implemented as a developed magnetization controlled magnetization (MCM) device, with internal state variables θ and ϕ for both upper layer and bottom layer. In the cycles of *loadA* and *loadB*, the precession switching can be observed for the MTJs of both nanowires. Also, the switching energy and time have been calculated as 0.27pJ and 600ps, which is consistent with the reported devices [106, 107, 108]. In the *shift* cycles, triggered by the *SHF*-control signal, the dynamics θ and ϕ of both upper and bottom layers are updated immediately. In the *operation* cycle, a subtle sensing current is applied to provoke GMR-effect. Subtle magnetization disturbance is also observed in both layers in the MCM device, which validates the read-operation. The θ values that

differ from initial values in the *operation* cycle also validate the successful domain-wall shift.

4.2.2 Adder

Figure 4.18: The carry out logic achieved by domain-wall nanowires

To realize a full adder, one needs both *sum* logic and *carry* logic. As the domain-wall nanowire based XOR logic has been achieved, the sum logic can be readily realized by deploying two units: $Sum = (A \oplus B) \oplus C$. As for carry logic, spintronics based carry operation is proposed in [109], where a pre-charge sensing amplifier (PCSA) is used for resistance comparison. The carry logic by PCSA and two branches of domain-wall nanowires is shown in Figure 4.18 (a). The three operands for carry operation are denoted by resistance of MTJ (low for 0 and high for 1), and belong to respective domain-wall nanowires in the left branch. The right branch is made complementary to the left one. Note that the C_{out} and $\overline{C_{out}}$ will be pre-charged high at first when PCSA EN signal is low. The complementary values can be easily obtained by reversely placing the fixed layers of MTJs in the right branch. When the circuit is enabled, the branch with lower resistance will discharge its output to '0'. For example, when left branch has no or only one MTJ in high resistance, i.e. no carry out, the right branch will have three or two

Figure 4.19: The sum logic achieved by domain-wall nanowires

MTJs in high resistance, such that the C_{out} will be 0. The complete truth table is shown in Figure 4.18 (b), which is able to confirm carry logic by this circuit. The domain-wall nanowire works as the writing circuit for the operands by writing values at one end and shift it to PCSA. The *sum* logic by PCSA is shown in Figure 4.19.

As the domain-wall *carry* logic is symmetric, there are only two possible input scenarios, which are both simulated by NVM-SPICE, and the simulation results are shown in Figure 4.20(a) and 4.20(b). For the scenario in Figure 4.20(a), all three MTJs in the left branch are at anti-parallel states with high resistance, and their complementary MTJs in the right branch are at parallel states with low resistance. Before logic is enabled, both C_{out} and $\bar{C}_{out}$ are logical high, and when the enable signal is asserted, the $\bar{C}_{out}$ which represents the branch with lower resistance is pulled down quickly, as expected. Similarly for scenario in Figure 4.20(b), where one MTJ state different from the other two, the C_{out} that represents the branch with lower resistance is pulled down to logical low after enable signal is asserted. All other input combinations are equivalent to either case therefore the *carry* logic can be validated. For both case, the operation current peaked at $10\mu A$, which is far less than the writing current $100\mu A$, and thus will not accidentally switch input operands and lead to incorrect result. By integrating the current-voltage product within the marked range, the energy consumption in this step can be calculated to be 3.3/3.49fJ.

Figure 4.20: The NVM-SPICE simulation results for carry logic (a) $A = 1$, $B = 1$, and $C_{in} = 1$; (b) $A = 0$, $B = 1$, and $C_{in} = 0$

4.2.3 Domain-wall multiplication

With the full adder implemented by domain-wall nanowires and intrinsic shift ability of domain-wall nanowire, the multiplication operation can be easily achieved by breaking it down to multiple domain-wall shift operations and additions. Operand A with m non-zero bits multiplied by operand B with n non-zero bits ($m > n$) can be decomposed into n shift operations and n additions. For example, multiplication of binary 1011 and 110 can be decomposed into addition of 10110 and 101100, where 10110 and 101100 are obtained by left-shifting 1011 one and two bits in domain-wall nanowire. As such, not only can the complicated domain-wall multiplier circuit be avoided, but also multiplication operation can be handled more efficiently by reusing domain-wall adders in a distributed MapReduce fashion, which will be discussed in Section 4.2.5.

4.2.4 LUT

Figure 4.21(a) shows the structure of cell in the LUT array. The access-port lies in the middle of the nanowire, which divides the nanowire into two segments. The left-half segment of nanowire is used for data storage while the right-half segment is reserved for shift-operation in order to avoid information lost.

Figure 4.21(b) shows the domain-wall nanowire based LUT array. The input of the function implemented by LUT is represented as binary address. The address is fed into word-line decoder and bit-line MUX to find the target domain-wall nanowire cell, where the multiple-bit result is kept. The LUT array size depends on the domain, range and

Figure 4.21: (a) Domain-wall memory cell structure; (b) LUT by domain-wall nanowire array with parallel output and serial output

precision of the function to perform.

Based on the way data is organized, the result can be output in serial manner or parallel manner. In serial output scenario, the binary result is stored in single domain-wall nanowire that is able to hold multiple bits of information. Assume each cell has only one access port and the first bit of result is initially aligned with access port, the way to output result is to iteratively readout and shift one bit until the last bit is output. In parallel output scenario, the multiple-bit result is distributed into different nanowires. Because each cell has their own access port, the multiple bits can be output concurrently. The design complexity of parallel output scheme is that, to find the relative position of the result within the nanowire, a variable access time will be introduced. For example, if luckily the result is stored at first bit of the nanowires, the result can be readout in one cycle; on the contrary if the result is kept at the very last bit of the nanowires, it will take tens of cycles to shift first before the result is output. Therefore, the choice

between serial output and parallel output is the tradeoff between access latency and design complexity.

Figure 4.22: Power characterization for DW-LUT in different sizes

Figure 4.22 shows the power characterization of DW-LUT in different array sizes. To obtain the area, power and speed of DW-LUT, the memory modeling tool CACTI [101] has been extended with domain-wall nanowire model as discussed in chapter 4. In terms of dynamic energy per look-up operation, the parallel output scenario is much more power efficient than serial output scenario, and the gap enlarges when array size increases. This is because more cycles are required to output results in serial than in parallel, therefore more access operations are involved. However, the serial scenario is able to avoid the variable access latency issue, which reduces the design complexity of the controller. For leakage power, the non-volatility leads to extremely low leakage power in nW scale, which is negligible compared with its dynamic power. For volatile SRAM and DRAM, the leakage power may consume as large as half the total power especially in advanced technology node [101].

Once the domain, range, and precision of function are decided, the DW-LUT size can be determined accordingly. Therefore, the power characterization can be used as a quick reference to estimate the power profile of specific function to perform system level design exploration and performance evaluation.

4.2.5 A Case Study: Matrix Multiplication By Domain-wall Logic

As shown in Figure 4.23, the proposed non-volatile memory (NVM) based computing platform is composed of three parts. Firstly, non-volatile domain-wall nanowire based lookup-table (DW-LUT) is utilized for configuring general logic. In this part, multiple

Figure 4.23: System overview of the proposed NVM-based logic-in-memory computing platform

LUTs are configured as different functions according to the program to be executed. Conventionally, program that intends to achieve complex functionality will be decomposed into basic instructions that the ALU can take by compiler. Therefore, it needs multiple clock cycles to be executed due to the decomposition. This compromises efficiency in order to gain better generality. However, in big-data applications, programs are usually intensive with a set of domain-specific functions without generality such that the coarse granularity can be introduced. With the basic functions implemented by LUTs, programs can be executed with greatly augmented execution performance as well as power efficiency. Secondly, the physics of domain-wall nanowire device is exploited for special logic. Although most of the functions are covered by LUTs, some commonly executed functions such as XOR and shift can be economically implemented by domain-wall nanowire device directly. Thirdly, non-volatile memory is deployed for the data storage. Obviously, the three main parts of the proposed computing system are rich of non-volatile devices, which can be significantly helpful to achieve high power efficiency and high bandwidth. In the following, we will explore detailed design by non-volatile domain-wall nanowire device for each part.

4.2.5.1 Map-Reduce Programming Paradigm

Matrix multiplication is one of the essential functions in big-data applications like data mining and web searching. For instance, singular value decomposition (SVD), which

can be used for deep learning in neural networks [110], involves iterations of matrix multiplication. Google PageRank, which intends to provide relative importance of billions of web pages according to searching query, also involves large amount of matrix multiplication operations [111]. In the following, we will show how matrix multiplication can be computed in parallel, and how it can be mapped to the proposed platform.

MapReduce [112] is a parallel programming model to efficiently handle large volume of data. The idea behinds MapReduce is to break down large task into multiple sub-tasks, and each sub-task can be independently processed by different *Mapper* computing units, where intermediate results are emitted. The intermediate results are then merged together to form the global results of the original task by the *Reducer* computing units.

The problem to solve is $x = M \times v$. Suppose M is an $n \times n$ matrix, with element in row i and column j denoted by m_{ij} , and v is a vector with length of n . Hence, the product vector x also has the length of n , and can be calculated by

$$x_i = \sum_{j=1}^n m_{ij}v_j = \sum_{j=1}^n \sum_{k=1}^l b_{ijk}$$

where the multiplication of $m_{ij}v_j$ is decomposed into the sum of b_{ijk} . As such, the matrix multiplication can be purely calculated by addition operations, and thus the domain-wall adder logic can be exploited.

The pseudo-code of matrix multiplication in MapReduce form is demonstrated in Algorithm 1. Matrix M is partitioned into many blocks, and each Mapper function will take the entire vector v and one block of matrix M . For each matrix element m_{ij} it decomposes multiplication of $m_{ij}v_j$ into additions of multiple b_{ijk} and emits the key-

Algorithm 1 Matrix multiplication in MapReduce form

function MAPPER(*partitioned matrix* $p \in M, v$)

for all *elements* $m_{ij} \in p$ **do**

$b_{ijk} \leftarrow \text{decompose}(m_{ij}v_j)$

$\text{emit}(i, b_{ijk})$ to list l_i

end for

end function

function REDUCER(l_q)

if *length of* $l_q > 1$ **then**

$\text{remove } (q, v_1), (q, v_2)$ from list l_i

$\text{emit}(q, v_1 + v_2)$ to list l_i

end if

end function

value pair (i, b_{ijk}) . The sum of all the values with same key i will make up the matrix-vector product element x_i . A reducer function simply has to sum all the values associated with a given key i . The summation process can be executed concurrently by iteratively summing two values and emitting one result until only one key-value pair is left for each key, namely the (i, x_i) .

4.2.5.2 Matrix Multiplication Task Mapping

Figure 4.24: Matrix multiplication mapping to proposed domain-wall nanowire based computing platform

Figure 4.24 shows how the MapReduce based matrix multiplication is mapped into the proposed non-volatile memory based computing platform. The execution starts with a command issued by external processor to the memory. The local controller in the in-memory logic part, a simple state machine for example, then loads data from the data array: the matrix M and vector v . A *map* process follows to decompose the multiplications into multiple values to sum by domain-wall shift operations, and then emit $\langle \text{key}, \text{value} \rangle$ pairs accordingly. All emitted pairs are stored in a separate segment of data array called intermediate results pool.

The <key, value> pairs are further combined in the *reduce* process. Specifically, the controller will fetch elements in the intermediate results pool and dispatch them to available reducers, namely domain-wall adders as introduced previously. Each reducer will take two values with same key, combine the values by addition, and emit a new pair to the intermediate results pool. The reduce process works in an iterative manner, combining two pairs to one pair until the intermediate results can not be further combined.

Table 4.5: Platform comparison

General settings		
Platform	multi-core	proposed
Technology	32nm	
Workload	1M×1M matrix multiplication	
Clock rate	3.4GHz	500MHz
Under 100W power budget		
Platform	multi-core	proposed
# of computing resources	8 cores	58716 LUTs+ 5963 adders
Performance	24.48 GOPS	917.44 GOPS
Area	145 mm ²	1292 mm ²
Under 145 mm ² silicon area budget		
Platform	multi-core	proposed
# of computing resources	8 cores	6591 LUTs+ 669 adders
Performance	24.48 GOPS	102.98 GOPS
Power	100W	11.23W

4.2.5.3 Performance Evaluation and Comparison

Table 4.5 shows the power, area, and performance comparison between proposed platform and conventional multi-core platform. The workload is matrix multiplication in the scale of million by million, and all matrix elements are 8 bit integer numbers. In addition, serial output scenario is adopted, and each nanowire is composed of 32 bits, out of which 16 bits are used for storing results and the other 16 bits are reserved for shift operation, leading to an LUT array size of 64K for one multiplication operation. The comparison focuses on the computation resources and is exclusive of memory components. The 32nm technology node is assumed for both scenarios.

The multi-core based scenario consists of 8 Xeon cores where the Map-Reduce based matrix multiplication is executed. The evaluation flow is in two steps. Firstly, gem5 [113] simulator is employed to take Map-Reduce based matrix multiplication from

Phoenix benchmark suites [114] and to generate the runtime utilization rate of core components. Next, the generated statistics is taken by McPAT [115], which is able to provide core area, power and performance results. For the domain-wall nanowire based computing platform, the matrix multiplication is translated into the task list. The DW-LUT and DW-logic evaluation is gained from previous sections, and combined with the operation count, the performance of proposed platform can be estimated.

The results indicated that, when power constraint is assumed for both systems, the proposed memory based platform exhibits 37x higher throughput, but at the cost of 9x larger silicon area. This is because the non-volatile memory based computing platform has very high power efficiency, thus more computation resources can be afforded to gain better performance. In the case where area constraint is adopted, the proposed system showed 4.2x better performance and 88.77% less power consumption.

Chapter 5

Non-volatile Memory Computing System

5.1 Hybrid Memory System with NVM

3D die-stacking [116] is promising to integrate hybrid memory components with high density and low latency. One can design a hybrid memory system with each tier by different memory technology stacked by through-silicon-vias (TSVs) [116]. As such, advantages of different memory technologies can be leveraged with compact vertical integration. However, as leakage power is the primary concern of a memory system, one needs to have well-designed data-retention scheme such that power gating can be effectively deployed to reduce leakage power yet without degrading performance. Traditionally, the common approach for data retention of SRAM/DRAM [117] is to deploy a small retention voltage for all memory cells in sleep-mode, which still has non-negligible leakage power. Recently, the work in [118] applies the non-volatile PCRAM for *system-level* data retention of DRAM. DRAM layer and PCRAM layer are stacked in 3D fashion and connected by TSVs. Benefited from the increased number of vertical data paths, the bus bandwidth is significantly improved compared to 2D scenario. However, due to the asymmetric performance between DRAM and PCRAM, the data transfer frequency and bandwidth are limited by the low write latency of PCRAM. Also, asynchronous hand-shaking protocol is required that incurs additional overhead. Another recent work in [119] has *bit-level* data retention by embedding one FeRAM device for each SRAM cell with bit-wise data-transfer controllers. Although concurrent bit-level data migration achieves fast speed, the overhead is overwhelmed with additional bit-wise data-transfer controllers in SRAM cells, which degrades the SRAM performance during the normal active mode. In this part, we will introduce a 3D hybrid memory architec-

ture, in which CBRAM-crossbar is used to reduce system leakage power by block-level data retention.

The motivation of data retention by non-volatile memory is illustrated in Figure 5.1. Without any power saving technique, the system will consume both dynamic power while it is executing and the leakage power when it is idle, and this results in a power profile as shown in Figure 5.1(a). This is because even when the system is idle, both memory and logic components of computing system will have leakage power. For memory, data needs to be retained in the memory for future usage, and conventional volatile memory needs to be powered on always. Ideally, a system can be most power efficient by only turning it on while it is executing, and turning it off when idle, which leads to the power profile in Figure 5.1(b).

Figure 5.1: (a) Conventional computing system power profile; (b) ideal instant on/off computing system power profile

Practically, the logic components can be controlled to shut down when system is idle. However, the current volatile memory still has performance advantage and will still prevail, therefore the use of volatile memory will still bring inevitable leakage power. Nevertheless, the memory leakage power can be saving in two ways. The first way, shown in Figure 5.2(a), is to apply the drowsy memory technique and put memory into sleep mode when system is idle. In sleep mode, the memory supply voltage is reduced while it is still able to keep data, so that leakage power is but not completely reduced.

Figure 5.2: Data retention for leakage reduction by (a) drowsy memory; (b) data back-up and recovery

The second way is to deploy data retention scheme. In this scheme, the still can be totally shut down, both for logic and memory, by migrating the memory content into hard-disk drive while it is about to be off, and restoring when it is on. Its power profile is illustrated in 5.2(b). Although a complete shut down can be achieved to avoid leakage power when it is idle, the data migration power overhead will be incurred. Therefore, which of the two ways is better depends on how frequently the system is in idle state and how long idle state will keep. In the following, we will show how the non-volatile memory can assist data retention scheme to reduce data migration overhead, reaching a power profile of 5.3.

5.1.1 Overview of CBRAM based Hybrid Memory System

Figure 5.4 illustrates the overall system architecture of the proposed 3D hybrid memory, which is composed of embedded DRAM, SRAM and CBRAM-crossbar located at three layers (or tiers) connected by TSVs. Similar to the common memory organization [101], the entire CBRAM-crossbar memory is broken into *banks*, where each bank can be accessed independently with dedicated data and address buses. Each bank is further

Figure 5.3: More power efficient instant on/off computing by incremental back-up as well as 3D CBRAM-crossbar memory

Figure 5.4: 3D hybrid memory system with CBRAM-crossbar based data retention

broken into a $M \times N$ mat array, where each mat consists of one $m \times n$ CBRAM-crossbar. In addition, the terms of *sleep mode* and *active mode* are used to denote whether the system is power-gated or not. The active to sleep mode transition is called *hibernating transition*, and term *wakeup transition* is used vice versa. For better presentation, Table 5.1 summarizes the notations used throughout this section.

To reduce the sneak-path power in crossbar structure [93], we propose to distribute the multi-bit data into different mats of the same row concurrently, where each mat (i.e., one CBRAM-crossbar) accepts one bit of the written data at each cycle. To achieve this, the data address is decoded to find the exact same crossing point of each CBRAM-crossbar mat where one bit of the arriving data is to be kept. For read operation, the same address decoding is carried out and the bits from each mat are combined together as output. As such, during one hibernating transition, each SRAM/DRAM bank is associated with one dedicated CBRAM-crossbar bank of the same capacity for data retention.

Table 5.1: Notations

Notation	Descriptions
T_{Hi}/P_{Hi}	Hibernating transition time/power for bank i
T_{Wi}/P_{Wi}	Wakeup transition time/power for bank i
P_b/P_m	Bank/mat power
E_b/E_m	Bank/mat access energy
C_b/C_m	Memory bank/mat capacity
N_b/N_m	Number of memory banks/mats
W_d/W_a	Data/address bus width of each bank
W_D/W_A	Total data/address bus width
M/N	Number of row/column mats of each bank
B_C/B_D	Cache line/DRAM page size

Therefore, power gating is employed at bank-level (i.e., block-level). Once the hibernating transition begins, all the data in the specified SRAM/DRAM bank must be copied or migrated to the corresponding CBRAM-crossbar memory bank through dedicated data and address buses implemented by TSVs in the vertical direction. On the other hand, all data must be migrated from CBRAM-crossbar back to the SRAM/DRAM during the wakeup transition.

However, the primary design challenges to develop the above-mentioned 3D hybrid memory with CBRAM-crossbar are from twofold. Firstly, there is no design platform for CBRAM device and CBRAM-crossbar circuit such that the delay, area and power can be estimated and optimized. Secondly, there is no memory controller developed and verified for the CBRAM-crossbar based data retention, which can perform efficient data migration for SRAM/DRAM. In the following, we show the development of one design platform for CBRAM device and CBRAM-crossbar circuit. Moreover, we show one memory controller design for CBRAM-crossbar using an incremental block-level data retention.

5.1.2 Block-level Incremental Data Retention

In this section, the proposed *block-level* data retention is discussed in details for the 3D hybrid memory architecture with CBRAM-crossbar.

The data is migrated between memory blocks (i.e., banks) sequentially through dedicated 3D TSV buses. Compared to the *bit-level* data retention scheme [119], where each SRAM memory cell is associated with one neighboring FeRAM cell with cell-wise controllers, our block-level approach can achieve much smaller area overhead since the data migration controller is shared by all memory cells of the same block. In addition, our block-level approach will not degrade the SRAM performance since no change is made

inside the SRAM memory cell. Furthermore, [118] has a system-level data retention by updating check-points. Because the use of PCRAM limits the frequency and amount of data that are retained, the system in [118] can only keep system check-points between relatively longer time intervals, which is insufficient when more fine-grained data retention is required. Based on the CBRAM-crossbar structure, this section has introduced one block-level data retention, which includes the block-level memory controller with two operations: dirty-bit set-up and incremental write-back.

5.1.2.1 Dirty Bit Set-up

The target for data retention is to synchronize the data of CBRAM-crossbar with corresponding SRAM/DRAM contents at block-level. For any SRAM/DRAM bank i , the time needed to copy the data to CBRAM-crossbar is decided by

$$T_{Hi} = \frac{M_b}{f_b \cdot W_d} \quad (5.1)$$

where f_b is the write-frequency of CBRAM-crossbar memory limited by its latency, W_d is the bank-level data bandwidth, and M_b is the amount of data to be migrated, which equals to the bank capacity C_b in the brute-force approach.

Clearly, reducing M_b directly reduces the T_{Hi} . Considering the common law of locality [120], the system tends to access relatively local-memory regions during a given period of time. As such, between two successive power gating stages, only part of the content in CBRAM-crossbar and SRAM/DRAM becomes unsynchronized, which is denoted as *dirty data* throughout this part. By only incrementally writing dirty data to CBRAM-crossbar, significant amount of migration data and power can be saved.

As shown in Figure 5.4, to keep the dirty-data status information, an extra CBRAM-crossbar called *dirty-data pool* is embedded to each CBRAM-crossbar bank, where each bit in the pool, referred as *dirty bit*, indicates the dirty status of a few continuous bytes of data in SRAM/DRAM, referred as *dirty-data group*. Empirically, we design the group granularity G_d as the cache line size B_C for SRAM or the page size B_D for DRAM.

The dirty-bit set-up occurs simultaneously each time when the content of memory is changed during active mode. As shown in Figure 5.5, each CBRAM-crossbar bank *listens* to all memory write operations issued to its corresponding SRAM/DRAM bank to update the dirty pool during active mode. Once the SRAM/DRAM write-action is detected, the corresponding data group becomes dirty.

As such, the corresponding bit in the dirty pool needs to be in SET state. The dirty pool size C_p is decided by

Figure 5.5: Circuit diagram for dirty bit set-up at active mode

$$C_p = \frac{C_b}{G_d} = p \cdot q \quad (5.2)$$

where p and q is the CBRAM-crossbar dimension of dirty pool. Therefore, the designated dirty bit position can be located by decoding the first $\log(p)$ and the following $\log(q)$ bits of the physical memory write-address, respectively.

5.1.2.2 Incremental Write-back

The flow to write back dirty SRAM/DRAM data to CBRAM-crossbar during hibernating transition is illustrated in Figure 5.6. Specifically, one address counter is used to check the status of all dirty bits in the dirty-data pool. Once the dirty bit in SET state is detected, the corresponding data group needs to be copied to the CBRAM-crossbar. Due to the limited data-bus bandwidth W_d , the group data of size G_d needs to be written back to CBRAM-crossbar in several cycles. As such, the address counter generates the memory address of the next piece of data to be copied from SRAM/DRAM by adding W_d -offset each cycle, and the read-signal to DRAM and write-signal to CBRAM-crossbar are issued for data migration. Once finished, the corresponding dirty bit is RE-SET. During wakeup transition, all data will be copied from CBRAM-crossbar back to SRAM/DRAM with similar hardware supports.

Here we will discuss how the use of CBRAM-crossbar is able to instinctively facilitate the dirty-data write-back without additional design efforts. As discussed above, we can see that intensive bit operations are required for dirty flags update and read out. For

tal write-back. We also present the comparison with system-level and bit-level data retention respectively.

Figure 5.7: Hibernating power and time reduction by incremental dirty-data write-back

In order to evaluate the dirty-data write-back strategy, a set of benchmark programs are selected from SPEC2006 suite and are run in gem5 simulator [113], where memory-access traces are generated. As the advantage of the dirty-data write-back strategy may depend on the memory access patterns of executed programs, the benchmarks with different memory access characteristics are picked in general. For example, *mcf* and *lbm* have high cache-miss rates while *h264* and *namd* have low cache-miss rates; *perlbench* and *gcc* have intensive store instruction while *astar* and *namd* have low store instruction [121]. For each benchmark, its dirty-data flags are updated according to the memory-access trace. Then, the dirty-data write-back strategy is deployed to evaluate the power saving and speedup during data migration. Figure 5.7 compares the hibernating transition power and time with and without using dirty-data write-back strategy. Averagely, system with dirty-data write-back strategy achieves 5x reduction in power and 1.5x reduction in time during the hibernating transition.

5.1.3 Design Space Exploration and Optimization

5.1.3.1 Design Space Construction

In this part, based on the developed CBRAM-crossbar models, we further show how to optimize the 3D hybrid memory system by performing design space exploration under different optimization objectives.

Given design freedoms of parameters shown in Table 5.1, the total capacity of SRAM, DRAM or CBRAM is calculated by

$$C_M = N_b \cdot C_b = N_b \cdot N_m \cdot C_m = N_b \cdot M \cdot N \cdot m \cdot n \quad (5.3)$$

where $M \cdot N$ is the dimension of mat array for each bank, and $m \cdot n$ is the dimension of crossbar array for each mat.

As indicated by equations (5.1) and (5.3), different combinations of bank number, mat array dimension and crossbar array dimension need to be explored for the optimal performance. For example, based on equation (5.1) it is desired to maximize the data-bus width W_d and CBRAM working frequency f_b . There are several design constraints to be considered as illustrated below one by one:

Data bandwidth constraint: as mentioned in section 5.1.1, since each CBRAM-crossbar supports only one-bit-write for each cycle, we need to design the multi-bit data to be distributed into different crossbars at the same row. Consequently, the data-bus width W_d must equal to N , which is the number of crossbars (i.e., mats) that each row contains.

Transition power constraint: during the hibernating and wakeup transition, the concurrent data migration will induce high transition power

$$\sum P_{bi} = \sum f_b W_{di} E_{mi} \leq P \quad (5.4)$$

where W_{di} and E_{mi} is the bus-width and energy consumed of bank i . The transition power needs to be limited by reliability concerns for hot-spots of thermal and supply-current.

TSV density constraint: since the TSVs for data transmission occupy certain area, diameter in $5\mu m$ for example [90], they lead to larger footprint as well as difficulties in placement and routing. As a result, the average TSV density must be limited which leads to an upper-bound on data bandwidth

$$\frac{W_D + W_A}{A} \leq D \quad (5.5)$$

where A is the chip area.

For our 3D hybrid memory system, one can perform design space exploration based on design parameters such as bank capacity C_b , data-migration bus bandwidth W_d and CBRAM-crossbar write-frequency f_b . Figure 5.1.3.1 shows the performance-objective with different trade-offs bounded by design constraints, which can be summarized as follows:

- For speed optimization, i.e. minimizing hibernating/wakeup transition time, large bandwidth is desirable. As such, small C_b , large W_d and high f_b are preferred. However, it is limited by TSV density and power constraints.

Figure 5.8: Design space exploration of 3D hybrid memory with CBRAM-crossbar

- For power optimization, i.e. minimizing transition power, small C_b and W_d architecture working in a low f_b is favorable, which is mainly limited by TSV density and speed constraints.
- For memory performance optimization, i.e. minimizing memory SRAM/DRAM performance degradation due to high TSV density, small W_d and large C_b help reduce TSV density, which in turn alleviates memory performance degradation, mainly limited by speed constraint.

For example, regions I, II, III in Figure 5.1.3.1 are corresponding to the limiting design constraints of TSV density, power and speed, respectively.

5.1.3.2 Optimization

This part shows the simulation experiment evaluation of the 3D hybrid memory from twofold. Firstly, the evaluation and optimization of the stacked CBRAM-crossbar memory is discussed. Based on the optimized CBRAM-crossbar memory design, the comparison between block-level data retention with the other schemes is then discussed. As the bus width and bank capacity parameters can only be numbers of 2^n , exhaustive search is applied to traverse all combinations in the design space.

TSVs length of $50\mu m$ is assumed. The TSV power and delay models in [122] are integrated into the extended CACTI [101]. Here we adjust the design parameters C_b , W_d and f_b under constraints of the maximal 10W transition power, maximal $60/mm^2$ TSV density and maximal 3ms transition speed. CBRAM memory performance with different capacity can be evaluated by a similar fashion as in Section 4.1.1.3.

Table 5.2: Optimized performance for different design objectives

Objective	Speed	Power	MP
C_b	256KB	256KB	4MB
W_d (bits)	16	8	128
P_H/P_W (W)	2.3/9.0	1.8/1.8	4.3/7.1
T_H/T_W (ms)	0.5/1.5	3/3	1.9/3
TSV density ($/mm^2$)	60	48	26

When employing CBRAM-crossbar for hybrid memory design, one needs to decide the optimal architecture-level parameters for performance objectives under certain constraints. As one example, we show the procedure of transition time (i.e., speed) optimization. Figure 5.9 shows the trend of TSV density and P_H/P_W with C_b/W_d . Among the available combinations, the one with C_b of 256KB and W_d of 16-bits is chosen since it results in largest W_D and power margin when increasing frequency and when satisfying the defined constraints. As such, when working at its maximal allowed frequency, 0.5ms T_W and 1.5ms T_h can be achieved.

Figure 5.9: (a) TSV density and (b) mode transition power under different architecture-level parameters

Table 5.2 demonstrates the optimal results for different performance objectives. An optimal 0.5ms hibernating transition time and 1.5ms wakeup transition time are achieved for the speed optimization. For transition power optimization, 5x less wakeup transition power and 1.6x less hibernating transition power are achieved with 6x wakeup transition time and 2x hibernating transition time penalties. For memory performance (MP in Table 5.2) optimization, the lowest TSV density is obtained as the minimized memory performance degradation in normal active mode. The design exploration leaves designer

the freedom to choose different design parameters based on system requirements. The configuration with speed optimized is used in the following experiments.

5.1.4 Performance Evaluation and Comparison

The system under investigation is composed of one 2MB level-2 SRAM cache, one 64MB embedded DRAM and one 66MB CBRAM for data retention. We compare the performance of data retention and leakage power in sleep mode for the following data retention schemes:

- *STD*: standard SRAM/DRAM without power gating.
- *DPG*: data-retentive power gating (DPG) of both SRAM [123] and DRAM [124] with reduced supply voltage.
- *PCRAM*: system-level data retention by PCRAM [118] which is used to keep the entire SRAM/DRAM contents.
- *FeRAM*: bit-level data retention by FeRAM [119].
- *CBRAM*: our proposed block-level data retention with incremental dirty-data write-back by CBRAM.

Table 5.3: Data-retention performance comparison for different leakage reduction schemes

Scheme	P_s^L/P_d^L (mW)	T_H/T_W (ms)	P_H/P_W (W/MB)
STD	209/220	NA	NA
DPG	21/22	1e-4	0
PCRAM	0/0	3.7/1.3	0.072/0.1
FeRAM	0/22	1	0.75
CBRAM	0/0	0.33/1.5	0.007/0.14

Table 5.3 shows the full comparison results in terms of sleep-mode leakage power for SRAM and DRAM (P_s^L/P_d^L), hibernating and wakeup transition time (T_H/T_W), and hibernating and wakeup transition power (P_H/P_W). The leakage power for SRAM/DRAM under standard scheme is generated by CACTI at 65nm technology node. Leakage power under DPG scheme is calculated by the leakage reduction factors reported in [123, 124] for both SRAM and DRAM. Block-level CBRAM based data-retention performance is derived from our platform, combining the architecture optimization results and power and time reduction results. PCRAM based scheme performance estimation is based on [118], with PCRAM memory performance obtained from PCRAMsim [104]. For

FeRAM-based scheme, bit-to-bit data-retention performance is extracted from [119]. Because all data is migrated concurrently in this scheme, the data retention power performance can be estimated by multiplication and speed performance is the same with bit-to-bit performance.

Due to the use of NVM for data retention, PCRAM, FeRAM and CBRAM based memory systems all outperform the STD and DPG schemes in terms of leakage power reduction. Moreover, our proposed 3D hybrid CBRAM-crossbar memory system allows a shut-down of both SRAM and DRAM during the power gating. As a result, compared to the block-level PCRAM based data retention, we achieve 11x faster hibernating transition time and 10x smaller hibernating transition power with the same number of TSVs. As shown in Table 4.2, the hibernating transition performance improvement comes from the advantageous CBRAM-crossbar memory performance and also the block-level incremental dirty-data write-back strategy. However, the wakeup transition time and power are slightly inferior to PCRAM based scheme. This is mainly because the incremental dirty-data write-back strategy does not apply to the wakeup transition. In addition, the proposed CBRAM system also outperforms the FeRAM by 107x/5.4x hibernating/wakeup transition power saving, and around 3x faster hibernating transition time.

Table 5.4: Cache performance comparison between block-level data retention (CBRAM) and bit-level data retention (FeRAM) in active mode

Cache capacity	Feature	Block-level CBRAM	Bit-level FeRAM
128KB	access time (ns)	1	1.8 (-80%)
	access energy (nJ)	0.13	0.21 (-62%)
	area (mm ²)	0.82	7.6 (-827%)
512KB	access time (ns)	1.5	2.8 (-87%)
	access energy (nJ)	0.33	0.6 (-82%)
	area (mm ²)	3.8	28.6 (-642%)
2MB	access time (ns)	2.7	4.3 (-59%)
	access energy (nJ)	0.6	1.3 (-117%)
	area (mm ²)	14.2	113 (-696%)

Further illustrated in Table 5.4, when compared with the block-level CBRAM based data retention, the bit-level FeRAM based data retention induces the overwhelming cache performance degradation in normal active mode due to bit-wise controllers embedded in the SRAM. For bit-level FeRAM based data retention scheme, the area overhead for retaining one bit data is $6.1 \mu\text{m}^2$, which is only $0.36 \mu\text{m}^2$ in our CBRAM system, estimated by the developed CBRAM-crossbar based CACTI. Due to such a large area overhead, the FeRAM based system shows an 84% SRAM access time degra-

dition, 74% SRAM access energy overhead on average. The performance of bit-level FeRAM based cache is derived from CACTI by replacing SRAM cell with FeRAM-SRAM cell pair. Therefore, our CBRAM-crossbar based data retention outperforms not only the system-level PCRAM based data retention but also the bit-level FeRAM based data retention.

5.2 In-memory Computing System with NVM

The current Von Neumann architecture has well-known memory-wall issue, which makes memory the bottleneck of whole computing system due to slow access latency of memory as well as the limited bandwidth for bus between computing resources and memory elements. The in-memory computing architecture, as shown in Figure 5.10, is promising to be the solution for the memory-wall issue. The in-memory architecture has lots of accelerators integrated inside the memory, so that the data will be pre-processed before they are readout to processor via I/O.

Figure 5.10: The overview of in-memory architecture with distributed memory for data server

Domain-wall nanowire [9, 10, 37], or racetrack memory, is a newly introduced spintronic NVM device. It has not only potential for high density and high performance memory design, but also interesting in-memory computing capability. As such, the word 'in-memory' can emphasize on two aspects: (1) to perform logic by memory devices (logic-by-memory) and (2) to integrate logic inside memory (logic-inside-memory). In the following, we will show how the domain-wall memory is exploited for in-memory computing from three levels. Firstly, a Map-reduce based general purpose processing platform with emphasis on logic-by-memory is presented. Secondly, an application specific (encryption) DWM based design is discussed. Thirdly, a domain specific (machine learning) DWM based in-memory computing platform is illustrated. The latter two emphasize on both logic-by-memory and logic-inside-memory.

5.2.1 General Purpose Big Data Computing

5.2.1.1 System Overview

Figure 5.11: The overview of the general purpose big data computing platform by domain-wall nanowire devices

One general purpose non-volatile in-memory computing platform based on domain-wall nanowire is shown in Figure 5.11. Big-data applications are compiled with Map-Reduce based parallel-computing model to generate scheduled tasks. A memory-based computing system is organized with integrated many-core microprocessor and main memory, which are mainly composed of the non-volatile domain-wall nanowire devices. The many-core microprocessors are further classified into clusters. Each cluster shares an L2-cache and accesses the main memory by shared memory bus. Each core works highly independently for allocated tasks such as Map or Reduce functions.

In this platform, the domain-wall nanowire is intensively utilized towards the ultra-low power big-data processing in both memory and logic, simultaneously. The domain-

wall nanowire based main memory can significantly reduce both the leakage and operating power of the main memory. What is more, large-volume of memory can be integrated with high density for data-driven applications. As such, one can build a hybrid memory system with CMOS-based cache as well as domain-wall nanowire based main memory, whose compositions can be optimized by studying the accessing patterns under big-data applications. More importantly, the domain-wall nanowire is also explored for computing purpose. Specifically, the domain-wall nanowire based XOR-logic for comparison and addition is studied in details based on the following observations. Firstly, at instruction level, the web-searching orientated big-data applications usually involve intensive string operations, namely comparison, where the XOR and Adder logics will be visited more frequently than usual. Moreover, from logic level, the transistors to implement XOR gates in ALU account for more than half of the total number, due to its much higher complexity compared to the NAND, NOR and NOT gates. As such, an optimized design of XOR-logic by new technology such as domain-wall nanowire may provide the largest margin to optimize hardware for big-data processing.

As discussed in Section 4.2, the domain wall based logic can be applied into ALU design in two function units, the XOR and full-adder for comparison and addition operations, respectively. Such two units account for more than half of the total transistors in ALU, and also are the most frequently used units, especially in big-data applications. The N -bit DW-XOR can be realized by employing N -bitwise DW-XOR, which is able to take the highly intensive *comparison* instruction. Although domain-wall logic is multi-cycled, the stalls caused by the slightly longer cycles can be greatly suppressed in the out-of-order super-scalar processor.

5.2.1.2 Performance Evaluation

As evaluated in Section 4.1.3, DWM with 1/2/4/8 access ports can achieve the area saving of 57%, 70%, 70% and 72%, and 92%, 95%, 96% and 97% leakage power reduction rates, respectively. In addition, as the data-orientated applications may have unique memory access patterns, thus in order to study the memory dynamic power under different benchmarks, gem5 [113] simulator is employed to take both SPEC2006 and Phoenix benchmarks [114] and generate memory accessing traces. The SPEC2006 is a CPU-intensive benchmark suite and Phoenix is a memory-intensive benchmark suite, so the selection of both benchmark suites will comprehensively characterize the system under various situations. The runtime dynamic power comparison under different benchmark programs are shown in Figure 5.12(a). With power differing greatly for different benchmark, it can be concluded that the dynamic power is very sensitive to the input

benchmark, and the results of the Phoenix benchmarks (matrix mul, kmeans, pca, word-count, and string match in Fig.5.12(a)) showed no significant difference from those in SPEC2006 (gcc, mcf and bzip2). This is because the dynamic power is effected by both intended memory access frequency and the cache miss rate. Figure 5.12(b) shows the normalized intended memory reference rate, and as expected the data-driven Phoenix benchmarks have several times higher intended memory reference rate. However, both L1 and L2 cache miss rates of Phoenix benchmarks are much lower than SPEC2006, which is due to the very predictable memory access pattern when exhaustively handling data in Phoenix benchmarks. Overall, the low cache miss rates of Phoenix benchmarks cancel out the higher memory reference demands, which leads to a modest dynamic power. Also, the runtime dynamic power contributes much less to the total power consumption compared to leakage power, thus the leakage reduction should be the main design objective when determining the number of access ports.

Table 5.5: System configuration

Processor	
Technology node	65nm
Number of cores	4
Frequency	1GHz
Architecture	x86, O3, issue width - 4, 32 bits
Functional units	Integer ALU - 6 Complex ALU - 1 Floating point unit - 2
Cache	L1: 32KB - 8 way/32KB - 8 way L2: 1MB - 8 way Line size - 64 bytes
Memory	
Technology node	32nm
Memory size	2GB - 128MB per bank
IO bus width	64 bits

In order to evaluate the DW-XOR based ALU design, gem5 simulator [113] is used to take both SPEC2000 and Phoenix benchmarks [114] and generate the instruction traces, which is then analyzed with the statistics of instructions that can be executed on the proposed XOR and adder for logic evaluation. McPAT[115] is then extended with additional power models of DW-XOR logic. As such, by taking the instruction analysis from gem5, the extended McPAT is able to evaluate both accurate dynamic power and leakage power information of DW-logic based ALU.

The system configuration is shown in Table 5.5. The 32-bit 65nm processor is assumed with four cores integrated. In each core, there are six integer ALUs which are

Figure 5.12: (a) The runtime dynamic power of both DRAM and DWM under Phoenix and SPEC2006 ;(b) the normalized intended memory accesses

able to perform XOR, OR, AND, NOT, ADD and SUB operations, and complex integer operations like MUL, DIV are executed in integer MUL. The instruction controlling decoder circuit is also considered during the power evaluation. The leakage power of both designs is calculated at gate level by the McPAT power model.

Figure 5.13: The per core ALU power comparison between CMOS design and DW-logic based design

Figure 5.13 presents the per-core ALU power comparison between the conventional CMOS design and domain-wall nanowire logic based design. Benefited from the use of DW-logic, both the dynamic power and leakage power can be greatly reduced. It can be observed that the set of Phoenix benchmarks consume higher dynamic power compared to those of SPEC2006, which is due to the high parallelism of Map-Reduce framework with high utilization rate of the ALUs. Among each set, the power results exhibit a low sensitivity to the input, which indicates that percentages of instructions executed in

XOR and ADDER of ALU are relatively stable even for different benchmarks. The stable improvement ensures the extension of the proposed DW-logic to other applications. Averagely, a dynamic power reduction of 31% and leakage power reduction of 65% can be achieved for ALU logic based on all the eight benchmarks.

5.2.2 Application Specific Computing: AES Encryption

Due to fast instant-on power-up and ultra-low leakage power, the newly introduced nano-scale NVM has shown great potential for future big data storage. However, the sensitive data will not be lost during reboot or suspension and hence is susceptible to attack. Further, large volumes of data must be encrypted with high throughput and low power. Traditional memory-logic integration based design incurs large overhead when performing encryption by logic through I/Os. Therefore, in-memory encryption would be preferred to achieve high energy efficiency during data protection.

Advanced encryption standard (AES) is the most widely used encryption algorithm, and various CMOS-based hardware implementations for AES have been presented [125, 126]. In scenarios where energy efficiency is critical, CMOS-based ASIC implementations tend to incur significant leakage power in current deep sub-micron regime with limited throughput. In [127], a memristive CMOL implementation by hybrid CMOS and ReRAM design is introduced to facilitate AES application. However, while the ReRAM serves as reconfigurable interconnection, it is not used for in-memory computation based encryption.

As spintronic devices have shown great scalability [128], it is promising to build big-data storage with in-memory logic based computing such as encryption. In this work, we propose a full domain-wall nanowire device based in-memory AES computing, called DW-AES. The non-volatile domain-wall nanowire devices are both used as storage element and deployed for logic computing in AES encryption. For example, *ShiftRow* transformation is facilitated by the unique shift operation of the domain-wall nanowire; *AddRoundKey* and *MixColumns* transformations benefit from the domain-wall nanowire based XOR logics (DW-XOR); and *SubBytes* and *MixColumns* transformations are assisted by the domain-wall nanowire based look-up table (DW-LUT). As such, all four fundamental AES transformation can be fully mapped to the non-volatile domain-wall nanowire based design.

5.2.2.1 Advanced Encryption Standard

The AES algorithm is described by Algorithm 2 and its corresponding flow chart is shown in Figure 5.14. All symbols used are listed in Table 5.6. In AES, the standard

input length is 16 bytes (128 bits), which are internally organized as a two-dimensional four rows by four columns array, called *state matrix* (M_s). During the AES algorithm, a sequence of transformations will be applied to the state matrix, after which the input block is considered encrypted and is then output. In order to study AES from hardware implementation point of view, the hardware complexity is analyzed for each transformation module with dominant resources reported within each module. Note that gate utilization data can be obtained by synthesizing a public domain AES Verilog code from [129]. Each transformation module is briefly described as follows.

- **SubBytes:** each state byte $S_{i,j}$ in the state matrix M_s will be updated by a nonlinear transformation independently, denoted as function f . The nonlinear transformation is often implemented as a substitution table, called S-box. S-box takes one byte as input and then outputs one byte to its original position. The SubBytes module accounts for half of the total gates in AES. Within this block, the dominant hardware resources are registers, used as look-up table elements. Note that the percentage may vary depending on designs. For example as reported in [127], when the non-linear function of S-box is implemented by combinational logics, the dominant resources are XOR gates, which account for more than 70% gates of this transformation block. As such, XOR logics are the essential logics to be optimized in the hardware implementation.
- **ShiftRows:** the n_{th} row of M_s will be cyclically shifted left by n bytes offset. As shown in Figure 5.14, the top row is not shifted; the second row is shifted by one byte position; the third row by two; and the fourth row by three. In the ASIC

Figure 5.14: The flow chart of AES algorithm with gate utilization analysis

Algorithm 2 Advanced Encryption Standard

Require: 128 bits plaintext

```
1: Organize input data as  $4 \times 4$  state matrix  $M_s$ , with each entry  $S_{i,j}$  as one byte
2: for  $r = 1 : N_r$  do
3:   for all  $S_{i,j} \in M_s$  do
4:     SubBytes transformation:  $S'_{i,j} \leftarrow f(S_{i,j})$ 
5:   end for
6:   for all  $S_{n,j} \in M_s$  in  $n_{th}$  row,  $n \in \{1, 2, 3, 4\}$  do
7:      $k \leftarrow (j-n+1) \bmod 4$ 
8:     ShiftRows transformation:  $S'_{n,j} \leftarrow S_{n,k}$ , i.e. each row left shifts circularly by
        $n - 1$  bytes
9:   end for
10:  if  $i \neq N_r$  then
11:    MixColumns transformation:  $M_s \leftarrow M_s \times M_{mc}$ 
12:  end if
13:  for all  $S_{i,j} \in M_s$  do
14:    AddRoundKey transformation:  $S'_{i,j} \leftarrow S_{i,j} \oplus K_{i,j}$ 
15:  end for
16:   $M_s \leftarrow M'_s$ 
17: end for
```

Ensure: 128 bits ciphertext

design, the ShiftRows transformation does not require additional logic gates but are performed on shift registers where the state matrix is held.

- MixColumns: each column of the state matrix M_s is multiplied by the known matrix M_{mc} as shown in Figure 5.14. The multiplication operation is defined as: multiplication by 1 means no change; multiplication by 2 means left shift; and multiplication by 3 means left shift and then XOR with the initial unshifted value. The step serves as an invertible linear transformation that takes four bytes in a column as input and then outputs four bytes to their original position, where each input byte affects all four output bytes. This module accounts for nearly half of the total number of gates, where all are XOR gates again.
- AddRoundKey: M_s is combined with the round keys M_k . The round keys are also 16 bytes organized in a 4×4 array as state matrix M_s . Each entry is denoted as $K_{i,j}$. Each byte $S_{i,j}$ of M_s will be updated by bitwise-XOR with its counterpart byte $K_{i,j}$ in the round key matrix M_k . Therefore, the AddRoundKey module is purely built by XOR gates, which accounts for 3.3% of total gates.

As a conclusion, we can observe that although in different designs the percentages may vary, the basic operations are without exception: XOR, shift, and table look up.

Table 5.6: Notation for domain-wall device based DW-AES implementation

Symbols	Descriptions
θ, ϕ	shown in Figure 3.20(b), azimuthal angles of magnetization orientation in x-z and x-y plane.
m	normalized magnetization
α	damping constant for domain-wall nanowire
θ_0	thermally provoked initial value of θ
R_l, R_h	P and AP state resistance of MTJ under voltage V
c_l, c_h	voltage-dependent coefficients for P and AP states
J_0	critical current density for domain-wall shift operation
v	domain-wall propagation velocity
$M_s, S_{i,j}$	4×4 state matrix that stores 128 bits AES input, with each entry state byte as $S_{i,j}$
$K_{i,j}$	key bytes correspond to $S_{i,j}$
A_i	the i_{th} domain-wall array for physical implementation of M_s
N_r	number of iterative rounds in AES algorithm
M_{mc}	the MixColumns transformation matrix
T	throughput of AES cipher
N_{lut}, N_{xor}	number of look-up table and xor units
t	cycle latency in DW-AES
$\mathcal{S}_R$	domain-wall architecture AES utilized resources set
$\mathcal{R}$	resources sets for each module
M	memory based logic units as element in resource sets

This motivates the utilization of the domain-wall nanowire devices, which is ideally suited for technology mapping for the AES encryption.

5.2.2.2 AES Task Mapping

The in-memory encryption offers two major advantages over the existing approaches. Firstly, all domain-wall based AES ciphers (DW-AES) can be integrated inside the memory, and hence AES encryption is performed directly on the targeted data that is stored in non-volatile domain-wall memory. This is significantly different from the conventional the memory-logic architecture in which the non-volatile storage data to process must be loaded into the volatile main memory, processed by logic, and written back afterwards. Secondly, the DW-AES cipher is implemented purely by the domain-wall nanowire devices, which are identical to the storage elements. This provides a good integration compatibility between DW-AES ciphers and memory elements, as well as the ability to reuse peripheral circuits like decoders and sense amplifiers. In this section, the detailed domain-wall nanowire based in-memory encryption will be discussed.

Figure 5.15: Data organization of state matrix by domain-wall nanowire devices in distributed manner

Data Organization of State Matrix Because in-memory encryption is performed directly on data cells, the data needs to be organized in certain fashion to facilitate the AES algorithm. As domain-wall nanowires only support serial access, that is, one bit of information can be accessed from a domain-wall nanowire at one time. In order to access multiple bits within one cycle, the data needs to be distributed into separate nanowires so that they can be operated concurrently. In AES algorithm, the basic processing unit is each byte in the state matrix. Therefore, the state matrix M_s is split into eight 4×4 arrays A_i ($\text{int } i \in [1, 8]$), as illustrated in Figure 5.15, where each entry of each array A_i becomes one bit instead of one byte. In other words, the n_{th} array stores the n_{th} bits of all $S_{i,j}$ in M_s . By distributing the state bytes $S_{i,j}$ and operating eight arrays together, the byte access requirement in AES algorithm is satisfied. In addition, to facilitate the ShiftRows transformation by exploiting the shift property of domain-wall nanowire, each row of an array needs to be stored within one domain-wall nanowire. In this case, each array is composed of four nanowires, and within each nanowire, the four bits data are kept along with some redundant bits used for efficient circular shift. Details regarding redundant bits will be discussed later in ShiftRows transformation. By organizing each 16 bytes of data in the above manner, the AES algorithm can be applied efficiently.

SubBytes In this step, each byte in the state matrix will undergo an invertible non-linear transformation. This transformation is commonly implemented as a look-up table (LUT), called substitution box (S-box). S-box LUT, essentially a pre-configured memory array, takes 8 bit input as a binary address, finds target cells that contain 8 bit result

through decoders, and finally outputs correspondingly by sense amplifiers. With 2^8 possible input scenarios, and each scenario having 8 bit result, the LUT size can be determined as $2^8 \cdot 8 = 2048$ bits. The LUT is conventionally implemented by SRAM cells, which in this size will incur significant leakage power.

Figure 5.16: SubBytes step with S-box function achieved by domain-wall memory based look-up table

In our proposed DW-AES design, the LUT is implemented by non-volatile domain-wall nanowire devices, i.e. DW-LUT, as shown in Figure 5.16. By distributing the 8 bit results in separated nanowires, the transformation can be done fast. In this parallel output scenario, eight sense amplifiers are required for each DW-LUT. As such, SubBytes transformation can be realized in a non-volatile fashion, that is, both the in-memory state matrix and S-box are made up of non-volatile domain-wall nanowires, which will enable significant leakage reduction. In addition, the memory and DW-LUT can share decoders and sense amplifiers, which leads to further power and area savings. The cycle number consumed in *SubBytes* stage t_{SB} can be represented as

$$t_{SB} = (t_{read} + t_{lut} + t_{write}) \times \frac{16}{N_{lut}}, N_{lut} \in \{1, 2, 4\} \quad (5.6)$$

where t_{read} and t_{write} are the read and write time (in cycle numbers) of state matrix M_s , and t_{lut} is the cycle number for look-up table. As each DW-LUT can handle one single $S_{i,j}$ transformation at one time, deploying more DW-LUT can increase the parallelism. The maximum value of N_{lut} is limited by the number of bytes that can be accessed of state matrix M_s . As each A_i can at most have four sense amplifiers, one for each

nanowire (row), at most four bytes can be accessed simultaneously. In other cases with less parallelism, sense amplifiers can be shared among different nanowires (rows).

ShiftRows The ShiftRows transformation can be efficiently achieved by exploiting the unique shift property of domain-wall nanowire. Due to the distributed data organization, in the ShiftRows transformation, the n_{th} rows of all A_i needs to left shift circularly by n bits ($n \in \{0, 1, 2, 3\}$). In other words, the second row needs to be left shifted cyclically by one bit, the third by two bits, and the fourth row by three bits, while the top row remains unshifted. In order to accomplish the circular shift in an elegant manner, i.e. without writing back the most significant bits to the least significant bits, redundant bits are used to form a virtual circle on the nanowire, as illustrated in Figure 5.17.

Since each row has predetermined shift operation, the number of redundant bits of each row can be readily determined: one redundant bit is required for second row, two bits for third row, and three bits for last row, all attached to the least significant bit from right side. As the shift operation for each nanowire (row) can be performed concurrently, the cycle number consumed in *ShiftRows* stage t_{SR} can be determined by the longest row shift delay,

$$t_{SR} = \max(t_{rs0}, t_{rs1}, t_{rs2}, t_{rs3}) \quad (5.7)$$

in which the t_{rsi} denotes the row shift delay for i_{th} row, $i \in \{0, 1, 2, 3\}$. In order to achieve all shifts in one cycle, shift currents of different amplitude are applied to each row according to the linear current-velocity relationship of shift operation [130]. In other words, the third row and fourth row are applied shift current that is twice and three times the amplitude applied to the second row. Consider the equivalent operations in circular shift with four bits: $LS1 \stackrel{\text{def}}{=} RS3$, $LS2 \stackrel{\text{def}}{=} RS2$, $LS3 \stackrel{\text{def}}{=} RS1$, where LS and RS indicate left and right shift, the number denotes the length to shift. This means in last row, instead of shifting 3 bits leftward, only right shift 1 bit needs to be performed. This helps to reduce the redundant data from 3 bits to 1 bit, as well as reduce the applied shift current to one third of previously required amplitude. The bits in same color indicate that they are synchronized bits. To ensure correct circular shift, the redundant bits need to be synchronized with their counterparts. As a result, during changes in the matrix state the redundant bits must also be updated.

In contrast with conventional computing flow, in which data needs to be moved to computing units for execution and written back to memory afterwards, the ShiftRows transformation is done directly on the stored data by in-memory computing fashion.

Figure 5.17: ShiftRows transformation by domain-wall nanowire shift operations

AddRoundKey In the AddRoundKey step, each byte in the state array will be updated by bit-wise XOR with corresponding key byte. As the dominant operation in this step is XOR, we propose a nanowire based XOR logic (DW-XOR) for leakage free computing. As described in Section 4.2, the GMR-effect can be interpreted as the bitwise-XOR operation of the magnetization directions of two thin magnetic layers, where the output is denoted by high or low resistance. In a GMR-based MTJ structure, however, the XOR-logic will fail as there is only one operand as variable since magnetization in the fixed layer is constant. This problem is overcome by the unique domain-wall shift-operation in the domain-wall nanowire device, which enables DW-XOR for computing.

The AddRoundKey with bitwise-XOR logic implemented by two domain-wall nanowires is shown in Figure 5.18. The proposed bitwise DW-XOR logic is performed by constructing a new read-only-port, where two free layers and one insulator layer are stacked. The two free layers each have the size of one magnetization domain and are from two respective nanowires. Thus, the two operands, representing the magnetization direction in each free layer, can both be variables with values assigned through the MTJs of their own nanowire. These assigned values are then shifted to the operating port such that the XOR can be performed.

Given A and B are 1 bit operands from state and key byte respectively, and 8 identical DW-XORs are used for bit-wise XOR between state byte and key byte, the $state \oplus key$ can be executed in the following steps

- The A and B are loaded into two nanowires by enabling WL_1 and WL_2 respectively;

- A and B are shifted from their access-ports to the read-only-ports by enabling SHF_1 and SHF_2 respectively;
- By enabling RD , the bitwise-XOR result can be obtained through the GMR-effect.

Hence, the cycle number consumed in *AddRoundKey* stage t_{ARK} can be represented as

$$t_{ARK} = (t_{read} + t_{xor} + t_{write}) \times \frac{128}{N_{xor}}, \quad (5.8)$$

$$N_{xor} \in \{1, 2, 4, 8, 16, 32\}$$

Similar to the *SubBytes* transformation, deploying more DW-XOR resources will increase the parallelism and reduce cycle number for this stage. When $N_{xor} = 32$, the *AddRoundKey* stage parallelism reaches its peak, and can handle one column of M_s at each time, which however requires more DW-XOR units and sense amplifiers.

MixColumns The MixColumns transformation can be expressed as the state matrix M_s multiplied by the known matrix M_{mc} shown in Figure 5.14. Specifically for the i^{th} column $i \in \{0, 1, 2, 3\}$, the column after transformation becomes

Figure 5.18: AddRoundKey step with XOR logic achieved by domain-wall nanowire

$$\begin{aligned}
s'_{0,i} &= 2 * s_{0,i} \oplus 3 * s_{1,i} \oplus s_{2,i} \oplus s_{3,i} \\
s'_{1,i} &= s_{0,i} \oplus 2 * s_{1,i} \oplus 3 * s_{2,i} \oplus s_{3,i} \\
s'_{2,i} &= s_{0,i} \oplus s_{1,i} \oplus 2 * s_{2,i} \oplus 3 * s_{3,i} \\
s'_{3,i} &= 3 * s_{0,i} \oplus s_{1,i} \oplus s_{2,i} \oplus 2 * s_{3,i}
\end{aligned}$$

The operations needed are multiplication by two (*xtime-2*), multiplication by three (*xtime-3*), and bit-wise XOR. The *xtime-2* is defined by left shift by 1 bit, and bit-wise XOR with *0x1B* if the most significant bit is 1; The *xtime-3* is defined as *xtime-2* result XOR with its original value. Therefore, there are only two de-facto atomic operations: 1) bit-wise XOR, executed by proposed DW-XOR, and 2) *xtime-2*. Although *xtime-2* can be implemented by in-memory shift together with additional DW-XOR, it is more efficient to use 8-bit input 8-bit output DW-LUT due to its branch operations depending on its most significant bit. As such, the MixColumns transformation can be purely performed by DW-LUT and DW-XOR, as shown in Figure 5.19.

Unlike *SubBytes* and *AddRoundKey* stages where the transformations for each byte $S_{i,j}$ is independent from other bytes, in the *MixColumns* each output byte $S'_{i,j}$ is a combination of four bytes in a column of M_s , therefore the parallelism is fixed and the cycle number consumed in *MixColumns* stage t_{MC} can be calculated as

$$t_{MC} = (t_{read} + t_{lut} + 3 \times t_{xor} + t_{write}) \times 4 \quad (5.9)$$

Due to the multiple-cycle operation of DW-XOR, and intensive utilization of DW-XOR, the *MixColumns* stage can be the most time consuming stage among the four stages.

Figure 5.19: MixColumns transformation by DW-LUT and DW-XOR

5.2.2.3 Pipelined AES by Domain-wall Nanowire

To further improve the performance of the domain-wall nanowire based AES, we further introduce a pipelined AES computing in this section.

Pipelined DW-AES Similar to the CMOS ASIC implementation, the DW-AES can be also implemented in pipelined fashion. AES has four stages, *SubBytes*, *AddRoundKeys*, *ShiftRows*, and *MixColumns*, and they can have different cycle numbers. Figure 5.20(a) shows the DW-AES implementation without pipeline. In Figure 5.20(a), only one of the four stages is active at one time while the rest three are idle waiting for data. The modules in idle state will still consume leakage power and, hence, inefficient use of energy. The benefit of introducing pipeline is to exploit the idle resources and increase their utilization rate, thus lead to higher energy efficiency and throughput. In CMOS ASIC implementation the pipeline can be readily achieved as the combinational circuit for each stage takes single cycle to execute and four stages are separated by registers. In the DW-AES, however, the case is a little different as each stage has different cycle numbers due to the multiple cycle nature of DW logic units. Assume the cycle numbers for the four stages are six, two, eleven, and five, the pipeline is obviously limited by the ShiftRows stage, i.e. the most time consuming stage. It means the AddRoundKey result has to wait for nine cycles before it takes new input data from SubBytes stage and its result enters next ShiftRows stage, and MixColumns stage also has to wait until

Algorithm 3 High Throughput/Energy Efficiency Domain-wall Nanowire based AES Mapping

Require: AES design area constraint A_m or/and power constraint P_m

```

1:  $\mathcal{S}_R \leftarrow \emptyset$ 
2:  $\mathcal{R}_{SR} \leftarrow \{32 \times M_{nw}\}; \mathcal{R}_{MC} \leftarrow \{4 \times M_{lut}, 8 \times M_{4-16dec}\}$ 
3: for  $(N_{xor}, N_{lut})$  combinations,  $N_{xor} \in \{1, 2, 4 \dots 32\}$  and  $N_{lut} \in \{1, 2, 4\}$  do
4:    $\mathcal{R}_{ARK} \leftarrow \{N_{xor} \times M_{xor}, 32 \times M_{sa}\}$ 
5:    $\mathcal{R}_{SB} \leftarrow \{N_{lut} \times M_{lut}, 32 \times M_{sa}, 8 \times M_{4-16dec}\}$ 
6:   calculate  $\{t_{SB}, t_{ARK}, t_{SR}, t_{MC}\} \leftarrow f(\{\mathcal{R}_{SB}, \mathcal{R}_{ARK}, \mathcal{R}_{SR}, \mathcal{R}_{MC}\})$ 
7:    $t'_{cycle} \leftarrow Sum\{t_{SB}, t_{ARK}, t_{SR}, t_{MC}\}$ 
8:    $\mathcal{S}'_R \leftarrow \mathcal{R}_{SB} \cup \mathcal{R}_{ARK} \cup \mathcal{R}_{SR} \cup \mathcal{R}_{MC}$ 
9:   calculate area  $A \leftarrow g(\mathcal{S}'_R)$ , power  $P \leftarrow h(\mathcal{S}'_R)$ , and throughput  $T' \leftarrow j(\mathcal{S}'_R)$ 
10:  if  $A \leq A_m$  and  $P \leq P_m$  and  $t'_{cycle} \leq t_{cycle}$  then
11:     $\mathcal{S}_R \leftarrow \mathcal{S}'_R$ 
12:     $t_{cycle} \leftarrow t'_{cycle}$ 
13:  end if
14: end for

```

Ensure: Domain-wall nanowire based AES architecture resource utilization S_R

Figure 5.20: (a)DW-AES without pipeline;(b) Pipelined DW-AES by inserting DW-FIFO ;(c) Stages balancing by the cycles delay of DW-FIFO

ShiftRows feeds its result.

By exploiting the shift operation of domain-wall nanowire, such cycle delay can be easily accomplished instead of using timers. Figure 5.20(b) shows the timing diagram of pipelined DW-AES with DW-FIFO inserted. The DW-FIFO is illustrated in Figure 5.20(c) which is essentially a nanowire with different lengths. By configuring the length of nanowire and shifting one domain per cycle, any number of cycle delay can be achieved. It can be observed in Figure 5.20(b) that each stage after compensation by DW-FIFO in the pipelined DW-AES has same cycle number (that of MixColumns stage), which ensures the data can be input at regular interval.

In DW-AES without pipeline scenario, the objective of DW-AES design optimization in terms of throughput can be formulated as

$$t_{MinSum} = Min(Sum t_{SB}, t_{ARK}, t_{SR}, t_{MC}) \quad (5.10)$$

Where t_{SB} , t_{ARK} , t_{SR} and t_{MC} are the required execution time of each stage in the unit

of cycle. The complete DW-AES without pipeline mapping algorithm is described in Algorithm 3 with symbols defined in Table 5.6. As such the DW-AES can produce a throughput of

$$T \propto \frac{1}{t_{MinSum}} \quad (5.11)$$

However, for the pipelined scenario, the case is different. As the most timing consuming stage, defined as critical stage, is the limiting factor of throughput, the objective of pipelined DW-AES design optimization in terms of throughput can be formulated as

$$t_{unit} = t_{MinMax} = \text{Min}(Max_{t_{SB}}, t_{ARK}, t_{SR}, t_{MC}) \quad (5.12)$$

with corresponding throughput as

$$T_{pipeline} \propto \frac{1}{t_{unit}} \quad (5.13)$$

where t_{unit} is the smallest possible interval of data input. Ideally if the four stages have the same cycle delay, the pipelined DW-AES will produce four times higher throughput, and in the worst case scenario where one stage is dominant the throughput for both cases will be almost the same. Therefore, the key to optimize pipelined DW-AES is to balance the cycle numbers of four stages. Considering that practically the resources, may it be power budget or area budget, is highly likely to be restricted, it is only possible to relax non-critical stages and improve critical stage by adjusting resources between different stages. Specifically, it needs to relocate computation resources from non-critical stages to critical stage. Given the above objective and constraints, design exploration that finds the pipelined DW-AES configuration can be performed to obtain design with the highest throughput.

Pipelined Multi-issue DW-AES Due to the huge hardware complexity difference of each stage in terms of gate count, the stages may still be unbalanced even after resources adjustment, i.e. one stage needs significant more time than other stages. Therefore, the multi-issue technique that is used in super-scalar processor is adopted for pipelined DW-AES. The DW-AES with pipeline/multi-issue mapping algorithm is illustrated in Algorithm 4 with symbols defined in Table 5.6. To solve the bottleneck of the critical stage, additional stage processing units are introduced to share the workload and improve the throughput calibre of critical stage, as shown in Figure 5.21(b). In Figure 5.21(b), the bottleneck is assumed to be the MixColumns stage therefore additional two units are added. Its timing diagram is shown in Figure 5.21(a) to show how this can improve the

Figure 5.21: Example (a) timing diagram and (b) block diagram of pipelined DW-AES with multi-issue

overall throughput. In Figure 5.21(a), it can be seen that the MixColumns stage needs to consume two times more time than other stages. Before applying multiple issue

Algorithm 4 Pipelined Domain-wall Nanowire based AES Mapping

Require: AES design area constraint A_m or/and power constraint P_m

- 1: $\mathcal{S}_R \leftarrow \emptyset$
- 2: $\mathcal{R}_{SR} \leftarrow \{32 \times M_{nw}\}; \mathcal{R}_{MC} \leftarrow \{4 \times M_{lut}, 8 \times M_{4-16dec}\}$
- 3: **for** (N_{xor}, N_{lut}) combinations, $N_{xor} \in \{1, 2, 4 \dots 32\}$ and $N_{lut} \in \{1, 2, 4\}$ **do**
- 4: $\mathcal{R}_{ARK} \leftarrow \{N_{xor} \times M_{xor}, 32 \times M_{sa}\}$
- 5: $\mathcal{R}_{SB} \leftarrow \{N_{lut} \times M_{lut}, 32 \times M_{sa}, 8 \times M_{4-16dec}\}$
- 6: calculate $\{t_{SB}, t_{ARK}, t_{SR}, t_{MC}\} \leftarrow f(\{\mathcal{R}_{SB}, \mathcal{R}_{ARK}, \mathcal{R}_{SR}, \mathcal{R}_{MC}\})$
- 7: minimize $\text{Max}\{\frac{t_{SB}}{N_{SB}}, \frac{t_{ARK}}{N_{ARK}}, \frac{t_{SR}}{N_{SR}}, \frac{t_{MC}}{N_{MC}}\}$
 $\quad \quad \quad N \in P$
- 8: $\mathcal{S}'_R \leftarrow \{N_{SB} \times \mathcal{R}_{SB}\} \cup \{N_{ARK} \times \mathcal{R}_{ARK}\} \cup \{N_{SR} \times \mathcal{R}_{SR}\} \cup \{N_{MC} \times \mathcal{R}_{MC}\}$
- 9: calculate area $A \leftarrow g(\mathcal{S}'_R)$, power $P \leftarrow h(\mathcal{S}'_R)$, and throughput $T' \leftarrow j(\mathcal{S}'_R)$
- 10: **if** $A \leq A_m$ **and** $P \leq P_m$ **and** $T' \geq T$ **then**
- 11: $\mathcal{S}_R \leftarrow \mathcal{S}'_R$
- 12: **end if**
- 13: **end for**

Ensure: Pipelined DW-AES architecture resource utilization S_R

technique, the throughput of MixColumns stage is only one third of other stages, and its cycle number is then the smallest possible interval to feed data. And by adding two more MixColumns units, the throughput of MixColumns stage is adjusted to be same as other stages, thus the smallest possible interval to feed data can be determined by the identical throughput of all stages, which is one third of previous value. To generalize, the smallest possible interval of data input, namely the unit time t_{unit} shown in Figure 5.21(a), can be obtained by

$$t_{unit} = t_{MinMax} = \text{Min}(\text{Max} \frac{t_{SB}}{N_{SB}}, \frac{t_{ARK}}{N_{ARK}}, \frac{t_{SR}}{N_{SR}}, \frac{t_{MC}}{N_{MC}}) \quad (5.14)$$

Where N_{SB} , N_{ARK} , N_{SR} and N_{MC} are the number of processing units of four stages respectively. And the associated throughput is

$$T_{multiissue} \propto \frac{1}{t_{unit}} \quad (5.15)$$

Deploying more processing units for each stage provides a fine-grained way to balance the stages. Thus compared to the throughput of pipelined DW-AES without multiple issue technique, the multi-issue pipelined DW-AES is able to have more balanced stage cycle numbers, which leads to smaller t_{unit} and higher throughput. In addition, due to the higher utilization rate, i.e. less idleness, of the processing resources, the ratio of leakage power to total power will be lowered, which makes the multi-issue pipelined DW-AES more energy efficient. The multi-issue DW-AES will incur some area overhead due to the increase of computational resources.

5.2.2.4 Performance Evaluation and Comparison

To evaluate DW-AES cipher, the following design platform has been set up. Firstly at device level, the transient simulation of MTJ read and write operations are performed within NVM-SPICE to obtain accurate operation energy and timing for domain-wall nanowire. The shift-operation energy is modeled as the Joule heat dissipated on the nanowire when shift-current is applied. The shift-current density and shift-velocity relationship are based on [130]. The area of one domain-wall nanowire is calculated by its dimension parameters. Specifically, the technology node of 32nm is assumed with width of 32nm, length of 64nm per domain, and thickness of 2.2nm for one domain-wall nanowire; the R_{off} is set at 2600Ω , the R_{on} at 1000Ω , the writing current at $100\mu A$, and the current density at $6 \times 10^8 A/cm^2$ for shift-operation. Secondly at circuit level, the

memory modeling tool CACTI [101] is modified with name as DW-CACTI. It can provide accurate power and area information for domain-wall nanowire memory peripheral circuits such as decoders and sense amplifiers (SAs). Together with the device level performance data, the DW-XOR as well as the DW-LUT can be evaluated at circuit level. The additional sequential controller of DW-AES is described by Verilog HDL, which is synthesized with area and power profiles. Finally at system level, an AES behavioral simulator is developed to emulate the AES cipher, as well as to explore the trade-offs among power, area, and speed.

The proposed DW-AES cipher is compared with both CMOS-based ASIC design [125, 126] and hybrid CMOS/ReRAM (CMOL) design [127]. For these implementations, performance data is extracted from the reported results in [125, 126, 127] with necessary technology scaling. C-code based software implementation that runs on a general purpose processor (GPP) is also compared. Evaluation of the AES software implementation is done in two steps. Firstly, gem5 [113] simulator is employed to take AES binary, compiled from C-code obtained from [131], which generates the runtime utilization rate of core components. Next, the generated statistics are taken by McPAT [115], which provides core power and area model. All hardware implementations run at the clock-rate of 3MHz, while the processor is operated at 2GHz for the software implementation. Table 5.7 compares the different implementations of AES cipher, and the results are discussed as follows.

Table 5.7: AES for 128 bits encryption performance comparisons

Implementation	leakage (μ W)	total power power(μ W)	area (μm^2)	cycles
C code [131] on GPP	1.3e+6	4e+5	2.5e+6	2309
CMOS ASIC [125]	120.54	154.74	953.05	534
memristive CMOL [127]	102.35	119.04	251.5	534
DW-AES	14.60	21.568	78.121	1022
DW-AES Pipelined	15.33	26.07	83.263	2652
DW-AES Multi-issue	21.35	45.86	155.05	1320

- Power: As expected, the DW-AES cipher has the smallest leakage power due to the use of non-volatile domain-wall nanowire devices. The remaining small leakage power is introduced by its CMOS peripheral circuits, i.e. decoders, sense amplifiers, as well as simple sequential controllers. Specifically, DW-AES cipher achieves a leakage power reduction of 88% and 86% compared to the CMOS ASIC and memristive CMOL designs, respectively. The leakage power can be further reduced if the decoders and SAs of the memory can be reused by the

DW-AES ciphers. As the computational resources are more active for pipelined DW-AES and multi-issue DW-AES, the power can be observed to be higher compared to DW-AES. Another interesting observation can be made that, the ratio of leakage power to total power is lowered from 68% for DW-AES to 59% for pipelined DW-AES to 47% for multi-issue DW-AES. This indicates that more energy is consumed for actual operations rather than wasted in idleness, and this helps to increased energy efficiency. The power breakdown for each module of all domain-wall based implementations is illustrated in Figure 5.22(c) and 5.22(d). For dynamic power, the *MixColumns* module that involves intensive xor operations and look-up table operations consume the majority of the total dynamic power. For leakage power, the *MixColumns* and *SubBytes* modules are dominant because they have more volatile components such as decoders for domain-wall memory array, as well as sense amplifiers.

- Area: Benefitting from the high density of domain-wall nanowire devices, the DW-AES cipher shows significant area reduction. In particular, highly area efficient DW-LUTs are deployed in the most resource consuming two modules, namely *SubBytes* and *MixColumns*, which contribute to the substantial area saving. Overall, the DW-AES cipher shows area reductions of 97% and 87% compared to the CMOS ASIC and memristive CMOL designs, respectively. Compared to DW-AES, the pipelined DW-AES incurs slight area overhead by 6.6%,

Figure 5.22: The breakdown of (a) area (b) latency (c) dynamic power (d) leakage power of DW-AES, pipelined DW-AES and multi-issued DW-AES

mainly caused by DW-FIFO for stage balancing and additional state matrices. The area for multi-issue DW-AES is almost twice of DW-AES, and this is due to the added processing units. Nevertheless, it still achieved an area reduction of 83.7 % compared to CMOS ASIC design. The area breakdown of all domain-wall based implementations is illustrated in Figure 5.22(a). Due to the use of DW-LUT, the *SubBytes* and *MixColumns* consume almost all the area.

- **Latency:** The tradeoff made in the DW-AES cipher is a larger number of cycles required compared to other hardware implementations. This is caused by the multiple-cycle operations of DW-XOR and its DW-LUT, where the shift-operation needs to be performed first in order to align the target cell with MTJ to operate. The latency breakdown of all domain-wall based implementations is illustrated in Figure 5.22(b). For the pipelined DW-AES, the stage balancing is achieved by inserting DW-FIFO which incurs some additional cycles. Therefore, it has identical cycle number for each module. For the multi-issue DW-AES, the stage balancing is mainly achieved by adding processing units with light insertion of DW-FIFO, therefore the cycle number is only slightly larger than that of DW-AES. Note that the latency between the raw data in and the encrypted data out is longer in pipelined DW-AES and multi-issue DW-AES, they are able to process four times more data than DW-AES, which means four times higher speed. While small latency is critical in real-time systems, in big-data applications the most significant figures of merit are throughput and energy efficiency.

Figure 5.23: In-memory encryption throughput, power and energy efficiency comparisons between different AES platforms

5.2.2.5 Throughput and Energy Efficiency Comparison

In the following section, the proposed in-memory DW-AES is compared with other implementations at the system level. For each AES computing platform, the number of AES units is maximized subject to a fixed area constraint. All AES units are encrypting input data stream concurrently due to the high data parallelism. With the exception of the proposed in-memory DW-AES, all platforms will incur I/O energy overhead accessing data. Given a 10mm^2 area design budget, the system configurations for different platforms are summarized in Table 5.8. The memory I/O energy overhead is obtained from CACTI.

Table 5.8: System configurations AES computing platforms configurations under 10mm^2 area design budget		
platforms	# of AES ciphers	clock-rate
C code [131] on GPP	4 cores	2GHz
CMOS ASIC [125]	10493	3MHz
Pipelined ASIC [126]	499	
memristive CMOL [127]	39761	
DW-AES	128010	
DW-AES pipelined	12010	
DW-AES multi-issue	6450	
memory configurations		
memory capacity	1GB	
bus width	128 bits	
I/O energy overhead	3.7nJ per access	

Figure 5.23 compares throughput, power, and energy efficiency of different AES computing platforms. All AES hardware implementations have several orders of magnitude throughput and energy efficiency improvement compared to the software implementation on general purpose processor, as expected. Among all the hardware implementations, the proposed DW-AES computing platform provides a throughput of 5.6 GB/s, which is 6.4X higher than that of the CMOS ASIC based platform with a power saving of 29%; 2.5X higher than that of the pipelined CMOS ASIC platform with 30% power reduction; and 1.7X times higher than that of memristive CMOL based platform with 74% power saving. And the pipelined multi-issue DW-AES produces the highest throughput of 8.7 GB/s, which is 10X higher than that of the CMOS ASIC based platform with a power saving of 24%; 3.9X higher than that of the pipelined CMOS ASIC platform with 25% power reduction; and 2.6X times higher than that of memristive CMOL based platform with 72% power saving. Due to the in-memory encryption

computing and non-volatility, the proposed DW-AES computing platform can achieve an energy efficiency of 24pJ/bit, which is 9X, 3.6X, 6.5X times higher than its counterpart: the CMOS ASIC, pipelined CMOS ASIC, and memristive CMOL based platforms, respectively. Furthermore, the pipelined multi-issue has the best energy efficiency of 15pJ/bit, which is 15X, 6X, and 11X times higher than above three counterpart implementations.

5.2.3 Domain Specific Computing: Machine Learning

One exciting feature of future big-data storage system is to find implicit pattern of data and excavate valued behavior behind by big-data analytics such as image feature extraction during image search. Instead of performing the image search by calculating pixel similarity, image search by machine learning is a similar process as human brains. For example, each image feature extraction is performed to obtain the characteristics first, and then is matched by key words. As such, the image search becomes a traditional string matching problem to solve.

However, to handle image data at very large scale, there is memory-wall that has long memory access latency as well as limited memory bandwidth. For the example of the image search in one big-data storage system, there may be billions of images. In order to perform feature extraction for one of images, it will lead to significant congestion at I/Os when migrating data between memory and processor. Note that in-memory computing system [65, 66, 67, 68, 69] is promising as one future big-data solution to relieve the memory-wall issue. For example, domain specific accelerators can be developed within memory for big-data processing such that the data will be pre-processed before they are readout with the minimum number of data migrations.

In this part, the big image data processing algorithm by machine learning is examined within the in-memory computing system architecture. Among numerous machine learning algorithms [132, 133, 134, 135], neural-network based algorithm has shown low complexity with genetic adaptability. In particular, the extreme learning machine (ELM) [136, 137] has one input layer, one hidden layer and one output layer, and hence it has tuning-free feature without expensive iterative training process, which makes it suitable for the low-cost hardware implementation. As such, the in-memory hardware accelerators of ELM is studied here for the big image data processing.

The proposed in-memory ELM computing system is examined by the nano-scale non-volatile memory devices. Domain-wall nanowire or racetrack memory, is one newly introduced spintronic NVM device that has not only potential for high density and high performance memory storage, but also feasible in-memory computing capability. In this

part, we show the feasibility of mapping the ELM to a full domain-wall nanowire based in-memory neural network computing system, called DW-NN. Compared to the scenario that ELM is executed in CMOS based general purpose processor, the proposed DW-NN improves the system throughput by 11.6x and energy efficiency by 92x.

5.2.3.1 Extreme Learning Machine

We first review the basic of the neural-network based ELM algorithm. Among numerous machine learning algorithms [132, 133, 134, 135, 136], support vector machine (SVM) [132, 133] and neural network (NN) [134, 135] are widely discussed. However, both two algorithms have major challenging issues in terms of slow learning speed, trivial human intervene (parameter tuning) and poor computational scalability [136]. Extreme Learning Machine (ELM) was initially proposed [136, 137] for the single-hidden-layer feed-forward neural networks (SLFNs) (Fig. 5.24). Compared with traditional neural networks, ELM eliminates the need of parameter tuning in the training stage and hence reduces the training time significantly. The output function of ELM is formulated as (only one output node is considered)

$$f_L = \sum_{i=1}^L \beta_i h_i(X) = h(X) \beta \quad (5.16)$$

where $\beta = [\beta_1, \beta_2, \dots, \beta_L]^T$ is the output weight vector storing the output weights between the hidden layer and output node. $h(X) = [h_1(X), h_2(X), \dots, h_L(X)]^T$ is the hidden layer output matrix given input vector X and performs the transformation of input vector into L -dimensional feature space. The training process of ELM aims to obtain output weight vector β and minimize the training error as well as the norm of output weight

$$\text{Minimize} : \|H\beta - T\| \text{ and } \|\beta\| \quad (5.17)$$

$$\beta = H^\dagger T \quad (5.18)$$

where $H^\dagger$ is the Moore-Penrose generalized inverse of matrix H .

The application of ELM for image processing to be discussed in this section is an ELM based image super-resolution (SR) algorithm [138], which learns the image features of a specific category of images and improves low-resolution figures by applying learned knowledge. Note that ELM-SR is commonly used as pre-processing stage to improve image quality before applying other image algorithms. It involves intensive

matrix operation, such as matrix addition, matrix multiplication as well as exponentiation on each element of a matrix. Figure 5.24 illustrates the computation flow for ELM-SR, where input vector obtained from input image is multiplied by input weight matrix. The result is then added with bias vector b to generate input of sigmoid function. Lastly sigmoid function outputs are multiplied with output weight matrix to produce final results. In the following, we will demonstrate how to map the fundamental addition, multiplication, and sigmoid function to domain-wall nanowires.

Figure 5.24: The working flow of extreme learning machine

5.2.3.2 In-memory Map-Reduce ELM architecture

Conventionally, all the data is maintained within a memory that is separated from the processor but connected with I/Os. Therefore, during the execution, all data needs to be migrated to the processor and written back afterwards. In the data-oriented applications, however, this will incur significant I/O congestion and hence greatly degrade the overall performance. In addition, significant standby power will be consumed in order to hold the large volume of data.

To overcome the above two issues, the in-memory non-volatile computing architecture is introduced. The overall architecture of domain-wall memory based in-memory

Figure 5.25: The overview of the in-memory computing architecture

computing platform is illustrated in Figure 5.25. In particular, domain specific in-memory accelerators are integrated locally together with the stored data in distributed manner such that frequent operations can be performed without much communication with external processor. In addition, the distributed local accelerators can also provide great thread-level parallelism thus the throughput can be improved.

Figure 5.26 shows how the in-memory distributed Map-Reduce [112] data processing is performed locally between one data array and local logic pair. Firstly, the external processor will issue commands to specific pair to perform in-memory logic computing. The commands will be received and interpreted by a controller in accelerator. Secondly, the controller will request related data to the data array with a read operation. As a result, the neural network based processing in ELM, mainly including matrix multiplication and sigmoid function, can be performed in a Map-Reduce fashion. Lastly, the results are written back to the data array.

5.2.3.3 ELM Task Mapping

Vector Inner Product by Domain-wall Adder and Multiplier The GMR-effect can be interpreted as the bit-wise XOR operation of the magnetization direction of two thin magnetic layers, where the output is denoted by high or low resistance. In a GMR-

Figure 5.26: Detailed in-memory domain-wall nanowire based machine learning platform in Map-Reduce fashion

based MTJ structure, however, the XOR-logic will fail as there is only one operand as variable since the magnetization in fixed layer is constant. Nevertheless, this problem can be overcome by the unique domain-wall shift-operation in the domain-wall nanowire device, which enables the possibility of DW-based XOR logic for computing.

A bitwise-XOR logic implemented by two domain-wall nanowires is shown in Figure 5.27. The bitwise-XOR logic is performed by constructing a new read-only-port, where two free layers and one insulator layer are stacked. The two free layers are in the size of one magnetization domain and are from two respective nanowires. Thus, the two operands, denoted as the magnetization direction in free layer, can both be variables with values assigned through the MTJ of the according nanowire. As such, it can be shifted to the operating port such that the XOR-logic is performed.

In addition, to realize a full adder, the carry operation is also needed. Spintronics based carry operation is proposed in [109], where a pre-charge sensing amplifier (PCSA) is used for resistance comparison. The carry logic by PCSA and two branches of domain-wall nanowires is shown in Figure 5.27. The three operands for carry opera-

Figure 5.27: Domain-wall nanowire based full adder with sum operation by DW-XOR logic and carry operation by resistor comparator

tion are denoted by resistance of MTJ (low for 0 and high for 1), and belong to respective domain-wall nanowires in the left branch. The right branch is made complementary to the left one. Note that the C_{out} and $\overline{C_{out}}$ will be pre-charged high at first when PCSA EN signal is low. When the circuit is enabled, the branch with lower resistance will discharge its output to "0". For example, when left branch has no or only one MTJ in high resistance, i.e. no carry out, the right branch will have three or two MTJs in high resistance, such that the C_{out} will be 0. The complete truth table will confirm carry logic by this circuit. The domain-wall nanowire works as the writing circuit for the operands by writing values at one end and shift it to PCSA. The multiplication is done by addition of many shifted values in a MapReduce fashion as discussed in Section 4.2.5.

Sigmoid Function by Domain-Wall Lookup Table Sigmoid function includes exponentiation, division, and addition, which is a computing intensive operation in ELM application. In particular, the exponentiation will take many cycles to execute in the conventional processor due to the lack of corresponding accelerator. Therefore, it is ex-

Table 5.9: Area, Power, Throughput and Energy Efficiency Comparison between In-Memory Architecture and Conventional Architecture for ELM-SR

Platform	Proposed	GPP (on-chip memory)	GPP (off-chip memory)
# of logic units	1 × processor 7714 × DW-ADDER 551 × DW-LUT 1 × controller	1 × processor	1 × processor
Logic Area (mm^2)	18 (processor) + 0.5 (accelerators)	18	18
Logic Power (Watt)	10.1	12.5	12.5
Throughput (MBytes/s)	108	9.3	9.3
EPB (nJ)	7	total: 394 I/O: 364 (92%) logic: 30 (8%)	total: 4127 I/O: 4097 (99%) logic: 30 (1%)

tremely economic to perform exponentiation by look-up table. Look-up table (LUT), essentially a pre-configured memory array, takes a binary address as input, finds target cells that contain result through decoders, and finally outputs correspondingly by sense amplifiers.

A domain-wall nanowire based LUT (DW-LUT) is illustrated in Figure 5.28(a). Compared with the conventional SRAM or DRAM by CMOS, the DW-LUT can demonstrate two major advantages. Firstly, extremely high integration density can be achieved since multiple bits can be packed in one nanowire. Secondly, zero standby power can be expected as a non-volatile device does not require to be powered to retain the stored data. By distributing the multiple bits of results in separated nanowires, the serial operation of nanowire can be avoided and the function can be done fast.

Note that the LUT size is determined by the input domain, the output range, and the required precision for the floating point numbers. Figure 5.28 shows the ideal logistic curve and approximated curves by LUTs. It can be observed that the output range is bounded between 0 and 1, and although the input domain is infinite, it is only informative in the center around 0. The LUT visually is the digitalized logistic curve and the granularity, i.e. precision, depends on the LUT size. For machine learning application, the precision is not as sensitive as scientific computations. As a result, the LUT size for sigmoid function can be greatly optimized and leads to high energy efficiency for sigmoid function execution.

To compare proposed in-memory DW-NN platform and conventional general pur-

Figure 5.28: (a) Sigmoid function implemented by domain-wall nanowire based look-up table (DW-LUT); (b) DW-LUT size effect on the precision of the sigmoid function

pose processor (GPP) based platform, ELM based super resolution (ELM-SR) application is executed as the workload. The evaluation of ELM-SR in GPP platform is based on gem5 [113] and McPAT [115] for core power and area model. DW-NN is evaluated in our developed self-consistent simulation platform based on NVMSPIICE, DW-CACTI, and DW-NN behavioral simulator. The processor runs at 3 GHz while the accelerators run at 500 MHz. System memory capacity is set as 1 GB, and bus width is set as 128 bits. Based on most recent on-chip interconnect and PCB interconnect studies in [139, 140], 40 fJ/bit/mm for on-chip interconnect and 30 pJ/bit/cm for PCB interconnect are used as I/O overhead. For core-memory distance, 10 mm is assumed for on-chip case and 10 cm is assumed for PCB trace length, both according to [139, 140].

Table 5.9 compares ELM-SR in both proposed in-memory computing platform and GPP platforms. Due to the deployment of in-memory accelerators and high data parallelism, the throughput of proposed in-memory computing platform improves by 11.6x compared to GPP platform. In terms of area used by computational resources, proposed in-memory computing platform is 2.7% higher than that of GPP platform. Additional

Figure 5.29: (a) Original image before ELM-SR algorithm (SSIM value is 0.91); (b) Image quality improved after ELM-SR algorithm by DW-NN hardware implementation (SSIM value is 0.94); (c) Image quality improved by GPP platform (SSIM value is 0.97)

0.5 mm^2 is used to deploy the domain-wall nanowire based accelerators. Thanks to the high integration density of domain-wall nanowires, the numerous accelerators are brought with only slight area overhead. In proposed in-memory computing platform, the additional power consumed by accelerators is compensated by the saved dynamic power of processor, since the computation is mostly performed by the in-memory logic. Overall, proposed in-memory computing platform achieves a power reduction of 19%. The most noticeable advantage of proposed in-memory computing platform is its much higher energy efficiency, energy-per-bit (EPB) as metrics, compared to GPP. Specifically, it is 56x and 590x better than that of GPP with on-chip and off-chip memory respectively. The advantage comes from three aspects: (a) in-memory computing architecture that saves I/O overhead; (b) non-volatile domain-wall nanowire devices that are leakage free; and (c) application specific accelerators. Specifically, the use of domain-wall logic/accelerators contributes to 4x improvement, while the in-memory architecture contributes to the rest (save of I/O overhead).

Figure 5.29 shows the image quality comparison between the proposed in-memory DW-NN hardware implementation and the conventional GPP software implementation. To measure the performance quantitatively, structural similarity (SSIM) [141] is used to measure image quality after ELM-SR algorithm. It can be observed that the images after ELM-SR algorithm in both platforms have higher image quality than the original low-resolution image. However, due to the use of LUT, which trades off precision against the hardware complexity, the image quality in DW-NN is slightly lower than that in GPP. Specifically, the SSIM is 0.94 for DW-NN, 3% lower than 0.97 for GPP.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

To conclude, this thesis has shown a thorough study on non-volatile memory design towards in-memory computing from device level, to circuit level, and all the way to system level. For the device level, we proposed a new modified nodal analysis for non-volatile memory devices to identify non-electrical state variables and capture its dynamic behavior. This method will ensure hybrid CMOS/NVM simulation to be performed with high accuracy and efficiency. In details, a compact SPICE-like implementation NVM-SPICE has been developed and released online for the new NVM devices, which exhibits 40x faster simulation speed for ReRAM and 117x faster simulation speed for spintronic devices compared to equivalent circuits based approaches.

For the circuit level, we have explored different kinds of memory circuit structures and structure-corresponding read and write circuits for new generation of universal memory, and the NVM based logic circuits such as XOR and adder are explored towards NVM computing. Specifically, the crossbar structure for ReRAM, 1T-1R structure for STT-RAM and multi-port domain-wall memory with agreeing sensing circuits are studied and validated in NVM-SPICE. A saw-tooth pulse based sensing circuit is designed for spintronic memories with 2X faster read speed with similar sensing margin or 8X larger sensing margin at similar speed. A combined system with NVM as main memory and NVM logic as accelerators is studied, and it can reduce 92% leakage power and 16% dynamic power from NVM side; and can also reduce 31% dynamic and 65% leakage power from NVM logic side.

For the system and architecture, low power hybrid memory and the non-volatile memory based in-memory architecture is proposed and well examined. Both data retention, encryption and machine learning applications are studied. For data retention, the

CBRAM-crossbar based data retention achieves 11x faster data-migration speed and 10x less data-migration power, when compared to phase-change random-access-memory (PCRAM) based approach. When compared to ferroelectric random-access-memory (FeRAM) based bit-level data retention, it achieves 17x smaller area and 56x smaller power under the same data-migration speed. For encryption, the AES algorithm is fully mapped into the in-memory architecture where all atomic operations are performed by NVM logic within memory, and it is 9X and 6.5X times better than CMOS ASIC and memristive CMOL implementations, respectively. And it has 6.4X higher throughput and 29% power saving compared to a CMOS ASIC implementation. For in-memory machine learning neural network, called DW-NN, all the atomic operations are mapped to in-memory NVM logic as well and performed within memory, and results show that the I/O load in the proposed DW-NN is greatly alleviated with an energy efficiency improvement by 56x and throughput improvement by 11.6x compared to the conventional image processing system by general purpose processor.

As the domain-wall based logic is multi-cycled, and the logic propagation timing needs to be controlled by CMOS controller, which becomes the main overhead. The CMOS controllers, essentially counters, enable specific part of domain-wall logic at particular cycle according to the functionality to implement. In addition, under current manufacturing technique, the domain-wall nanowire devices may exhibit large process variations. Such variations may lead to false write, read and inaccurate shift operations, thus may introduce errors in such NVM based logic architecture. The next step realization of the proposed architecture requires both the advance of NVM manufacture technology as well as more matured variation-tolerant NVM operating circuits.

6.2 Recommendations for Further Work

Based on above works, there are a few recommended future works for this thesis.

The first recommended work is to explore the in-memory computing logic by ReRAM crossbar structure. The ReRAM crossbar can be potentially configured into any possible functions by constructing a neural network based on ReRAM crossbar. A ReRAM crossbar has strong structural similarities to neural network, voltage levels as inputs, ReRAM devices resistance as weights, resulting currents merge as sum operation, and ReRAM device threshold as transfer function. Based on this, the in-memory computing can have more options in terms of accelerators, and therefore can be harnessed for a wide range of big-data applications. Especially, 1-bit compressive sensing signal processing technique together with such ReRAM digital neural network in in-memory computing fashion will provide significant power reduction and reconfigurability, and therefore is very promising

to be a universal solution for low power big-data processing architecture.

The second recommended work is to study the compressive domain data storage policy on non-volatile memory. The image data, biomedical measurement (EEG), etc. stored in compressive domain for big data system will be very helpful to lower processing power and reduce storage volume. However, due to the immature fabrication process and small readout voltage margin for the emerging NVM technologies, the data stored in the NVM may be corrupted or misread. Without correction, this will result in false recovered signal. Therefore, robust data reconstruction is needed to ensure the whole system is working without errors. It is possible to study the impact of all kinds of operation parameters, such as current pulse length/amplitude, device dimensions (affect write energy), on successful device operation rate (switching possibility, correct read possibility), so that best parameters can be chosen to minimize power consumption which still can ensure successful reconstruction even if errors are incurred.

Appendix A

NVM-SPICE Design Examples

A.1 Memristor Model Card in NVM-SPICE

NVM-SPICE is developed with NVM nonlinear dynamic models added by extending NGspice. The syntax generally follows NGspice style. One slight difference is one more identifier for NVM device type is required. For example, the general form of model card of a memristor element is

$$n < name > \text{ memristor } < +node > < -node > < model > < params > .$$

The first letter of the element name specifies the element type. Here the memristor has been assigned a letter starting with *n*. The *<name>* column is the name of device, which can be arbitrary alphanumeric strings. The following “memristor” is used to specify the type of NVM device to be the memristor. The following two columns are used to indicate positive and negative nodes for its topological connection in circuit. The *<model>* column is to apply a predefined model (a set of specified model parameters) to the device. The parameters for memristor device instances can be specified in the *<params>* column.

Table A.1: The list of supported parameters for nonlinear dynamic memristor model

Name	Model parameter	Units	Default	Example
ron	resistance of memristor for conducting state	Ω	100	50
roff	resistance of memristor for non-conducting state	Ω	16k	100k
height	thickness of memristor film	m	50n	50n
mu	mobility at small electric field	$m^2/(V \cdot s)$	0.01f	0.01f
e0	characteristic field for a particular mobile atom in the crystal	V/m	100meg	100meg
wf	the type of window function	-	1	2
p	the slow down effect parameter in window function	-	2	5
rhoon	the electrical resistivity of conducting part of memristor	$\Omega \cdot m$	not specified	10u
rhooff	the electrical resistivity of non-conducting part of memristor	$\Omega \cdot m$	not specified	20m
length	the length of cross-section area of memristor	m	not specified	50n
width	the width of cross-section area of memristor	m	not specified	50n
Name	Instance parameter	Units	Default	Example
rinit	initial resistance of memristor	Ω	100	50k

Table A.1 shows the full list of parameters for the implemented nonlinear dynamic memristor model. For not specified parameters, the default values will be assigned. Note that when all the four parameters ρ_{on} , ρ_{off} , w and l are specified, the r_{on} and r_{off} will be calculated by

$$R_{on} = \frac{\rho_{on} \cdot d}{w \cdot l}, \quad R_{off} = \frac{\rho_{off} \cdot d}{w \cdot l}. \quad (A.1)$$

and the specifications of r_{on} and r_{off} will be ignored. The Joglekar window function is applied by default with $wf = 1$. Alternatively, the Biolek window function can be applied by setting $wf = 2$ or no window function $wf = 0$.

Some examples for memristor element description are shown below:

```
.model nvm_mem_model1 memristor ron=0.1k roff=14k wf=1 p=5
n1 memristor 2 0
nref memristor 7 3 rinit=0.1k
nr5c5 memristor 16 2 nvm_mem_model1
```

A.2 Transient CMOS/Memristor Co-simulation Examples

Here, we will illustrate how to use NVM-SPICE for hybrid CMOS/NVM design co-simulation with simple circuits for memristor, shown in figure A.1. This toy example circuit intends to study the SET operation of a memristor device.

Figure A.1: 1T1R structure for memristor device based memory cell

The corresponding netlist to describe the circuit in figure A.1 can be written below:

```
* memristor SET operation study
.model nvmmmod memristor ron=1k roff=16k
```



```
.model nmos nmos level=54 version=4.7.0
vdd nvdd 0 3.3v
n1 memristor nvdd d nvmmmod rinit=15.9k
vcontrol g 0 pwl(0 0 10us 0 11us 3.3 90us 3.3 91us 0 100us 0)
m1 d g s 0 nmos l=90n w=2u
.tran 10n 100us
.end
```

After running transient analysis of above netlist, the following **PLOT** command can be used to investigate the change of internal state doping ratio for memristor:

```
plot n1#doping
```

Figure A.2: Dynamics of doping ratio in memristor set operation under transient analysis

Then we obtain the wave shown in figure A.2. It can be seen that the doping ratio changes from 0 (at 11 μ s) to 1 (at around 82 μ s) which indicates its resistance is switched from 15.9k Ω to 1k Ω within 71 μ s under 3.3V programming voltage. To further verify this, we can plot the actual resistance of memristor by

```
plot (v(nvdd)-v(d))/i(vdd)
```

We can get figure A.3, from which it is clear that the resistance does change from 15.9k Ω to 1k Ω . The internal state variables of NVM devices are usually associated with the external resistance, thus knowing the internal states, the way to obtain external resistance is to calculate it referring to its model equations.

Figure A.3: Plot of time-varying resistance of memristor for verification

A.3 STT-MTJ Model Card in NVM-SPICE

Similar to memristor, the general form of model card of a STT-MTJ element is

$$n < name > sttmjtj < +node > < -node > < model > < params > .$$

Table A.2 shows the full list of parameters for the implemented STT-MTJ model. Some examples for STT-MTJ element description are shown below:

```
.model nvm_sttmjtj_model1 sttmjtj rl=2k rh=4k ms=30k ms=700k ka=0.2 kb=0.5
n1 sttmjtj 2 0
nref sttmjtj 7 3 r0=550k
nr5c5 sttmjtj 16 2 nvm_sttmjtj_model1
```

Table A.2: The list of supported parameters for STT-MTJ model

Name	Model parameter	Units	Default	Example
vcp	voltage-dependent coefficient for parallel state	-	0.01	0.1
vcap	voltage-dependent coefficient for anti-parallel state	-	0.9	0.65
p	pre-factor of the spin transfer term and driving current ratio	-	6.37	6.37
gamma	electron gyromagnetic ratio in Landau-Lifshitz-Gilbert equation	$(sA/m)^{-1}$	221k	221k
ms	saturation magnetization of material	kA/m	800k	800k
hk	effective anisotropy field	kA/m	29.05k	29.05k
rp	resistance value of parallel state	Ω	1230	1k
rap	resistance value of anti-parallel state	Ω	2650	5k
damping	damping constant in Landau-Lifshitz-Gilbert equation	-	0.01	0.005
Name	Instance parameter	Units	Default	Example
phi0	initial radian for internal state variable ϕ	rad	1	1
theta0	initial radian for internal state variable θ	rad	0.001	0.005

A.4 Transient CMOS/STT-MTJ Co-simulation Examples

Figure A.4: (a) STT-MTJ writing circuit in 1T1R memory cell structure and (b) STT-MTJ sensing circuit by resistance comparator

We could write the netlist for STT-MTJ SET operation circuit depicted in figure A.4(a) as following

```
* STT-MTJ SET operation study
.model nvmmmod2 sttmjtj vcp=0 vcap=0 rap=1000 rp=500
.model nmos nmos level=54 version=4.7.0
v1 nvdd 0 pwl(0 0 5ns 0 6ns 1.2v)
vcontrol g 0 pwl(0 0 4ns 0 5ns 1.2v)
m1 d g 0 0 nmos l=90n w=2u
n1 sttmjtj nvdd d nvmmmod2 theta0=0.01
.tran 0.01n 30ns
.end
```

Similarly, we run the commands to plot internal state and external resistance, and results are shown in Figure A.5 and A.6.

```
plot n1#theta
plot (v(nvdd)-v(d))/i(v1)
```

Below is another example of STT-MTJ switch study under repeated write cycles. The netlist is shown as below,

Figure A.5: Plot of time-varying internal state theta of STT-MTJ

Figure A.6: Plot of time-varying resistance of STT-MTJ

** STT-MTJ switch study under pulse signal*

.model nvmmmod2 sttmjtj vcp=0.1 vcap=0.65 rl=1230 rh=2650

.param period=40ns delay=0ns pulse_width=19.5ns vdd=1.1 vss=-1.1

v1 1 0 pulse(1.5 -1.5 0 0.5ns 0.5ns pulse_width period)

n1 sttmjtj 1 0 nvmmmod2 theta0 = 0.001

.tran 0.01ns 300ns

.end

Its simulation results are shown in Figure A.7. The blue line is the pulse writing voltage signal, and the read line is the internal state θ of STT-MTJ device $n1$. The oscillation before switch and damping after successful switch of magnetization in each cycle can be easily observed.

Figure A.7: Switch of STT-MTJ under pulse signal

The netlist of STT-MTJ sensing circuit in Figure A.4(b) is shown below. * *STT-MTJ sensing circuit study*

```
.model nvmmmod2 sttmjtj vcp=0 vcap=0 rh=3000 rl=1000
.model nmos nmos level=54 version=4.7.0
.model pmos pmos level=54 version=4.7.0
.param pwidth=450n nwidth=150n length=45n theta_init=3.04 vdd=1.2
v1 nvdd 0 vdd
v_en nven 0 pwl(0 0 5ns 0 6ns vdd)
* left-top, right-top, and bottom enable xtors
mlen ncout nven nvdd nvdd pmos l=length w=pwidth
mren ncoutb nven nvdd nvdd pmos l=length w=pwidth
mben nben nven 0 0 nmos l=length w=nwidth
* latch for resistance comparison
mlp ncout ncoutb nvdd nvdd pmos l=length w=pwidth
mln ncout ncoutb nlm 0 nmos l=length w=nwidth
mrp ncoutb ncout nvdd nvdd pmos l=length w=pwidth
mrn ncoutb ncout nrm 0 nmos l=length w=nwidth
* STT-MTJ device and reference resistor:
* STT-MTJ resistance is determined by theta0:
* pi for high res, and 0 for low res
n1 sttmjtj nlm nben nvmmmod2 theta0=theta_init
r1 nrm nben 2000
.tran 0.01n 30ns
```

.end

Its simulation results are shown in Figure A.8. The left figure corresponds to the initial condition that $\theta_0 = \pi$, i.e. high resistance state, and the right figure $\theta_0 = 0$, i.e. low resistance state. The EN enable signal is asserted at $5ns$ and it can be observed that the output will stay high for the branch with higher resistance.

Figure A.8: Simulation results for STT-MTJ sensing circuit as shown in Figure A.4(b)

Appendix B

Magnetization Physics

A large portion of the emerging non-volatile memory devices are magnetization based. The MRAM usually is formed by one insulator in the middle, sandwiched by two ferromagnetic layers, namely fixed layer that is strongly magnetized and free layer that can be easily changed. Differed by the approaches for writing, there are several phases of MRAM technology. The first generation MRAM needs external magnetic field to switch the free layer magnetization. The second generation STT-RAM is introduced and the free layer magnetization can be altered by polarized current, which brought significant advantages such as easy integration with current CMOS technology and high density, high reliability etc. Recently, the third generation domain-wall racetrack is introduced, with a series of magnetization domains in one ferromagnetic thin-film nanowire, and additional shift ability. The shift is also current-induced operation. In this section, the magnetization dynamics under external field and spin-current in nanosecond regime will be introduced.

B.1 Basic Magnetization Process

As an intrinsic property, electrons spin about its axis and produce magnetic field like current carrying wire loop. From macrospin point of view, the relation between magnetization M and angular momentum associated with electron spin S , can be expressed as

$$M = -\gamma S \quad (\text{B.1})$$

where $\gamma = 2.21 \times 10^5 \text{ m A}^{-1} \text{ s}^{-1}$ is the gyromagnetic ratio. A uniform magnetic field exerts no net force on a current loop but it does exert a net torque, and the torque T , on the current carrying loop under applied magnetic field H , can be expressed as

$$T = M \times H \quad (\text{B.2})$$

By definition, the time derivative of angular momentum is called torque. The relation between the angular momentum L , and torque T , reads

$$\frac{dL}{dt} = T \quad (\text{B.3})$$

The quantum form of Equation B.3 still remains valid, and then we have

$$\frac{dS}{dt} = T \quad (\text{B.4})$$

By combining Equations B.1, B.2 and B.4, we can obtain the motion equation of magnetization under applied magnetic field

$$\frac{dM}{dt} = -\gamma M \times H \quad (\text{B.5})$$

The precessional motion described by equation B.5 indicates that the magnitude of magnetization will not change, and also the angle between H and M will not change, which is depicted in Figure B.1. This is based on that no energy loss is assumed during this process.

B.2 Magnetization Damping

In real systems, however, energy is dissipated through various means and the magnetization motion is damped until an equilibrium is reached. Energy dissipation can occur through the spin-spin, spin-photon, and spin-electron interactions through which the

Figure B.1: The magnetization precession.

Figure B.2: The magnetization precession with damping.

spin energy is transferred. The approach followed by Landau and Lifshitz is to introduce dissipation in a phenomenological way. In fact, an additional torque term that pushes magnetization in the direction of the effective field is introduced. Landau-Lifshitz equation in the Gilbert form, or LLG equation, then reads

$$\frac{d\mathbf{M}}{dt} = -\gamma \mathbf{M} \times \mathbf{H} + \frac{\alpha}{M_s} \mathbf{M} \times \frac{d\mathbf{M}}{dt} \quad (\text{B.6})$$

The magnetization dynamics described by Equation B.6 is sketched in Figure B.2.

B.3 Spin-Transfer Torque

In 1996, Berger [142] and Slonczewski [143] predicted, which later has been confirmed experimentally [144, 145, 146], that electrons that carry enough angular momentum are able to cause magnetization precession and switching by spin-transfer torque effect. When a current passes through a strongly magnetized ferromagnetic layer, the spins of electrons are polarized to align with magnetization direction and hence carry angular momentum. And the spin-polarized current, when enters another thin ferromagnetic layer, exerts a spin torque on the local magnetic moment and causes the magnetization precession and switching when the current is large enough (Fig. B.3).

Therefore, the dynamics of the free layer magnetization can be determined by the LLG equation in conjunction with an additional term for spin-transfer torque,

$$\frac{d\mathbf{M}}{dt} = -\gamma \mathbf{M} \times \mathbf{H} + \frac{\alpha}{M_s} \mathbf{M} \times \frac{d\mathbf{M}}{dt} - \frac{a_J}{M_s} [\mathbf{M} \times (\mathbf{M} \times \mathbf{P})] \quad (\text{B.7})$$

where $\mathbf{P}$ the magnetization of fixed layer, M_s the saturation magnetization, a_J a factor related to the interfacial interaction between magnetic moment and spin-polarized elec-

Figure B.3: The spin-transfer torque effect

trons, which is proportional to the current density, and the sign of a_J depends on the direction of current. When applied properly, the current is able to cancel the damping and switch the magnetization of free layer by spin transfer torque.

For the current induced magnetization precession, there exists a threshold current density J_{c0} , and by applying current larger than J_{c0} the magnetization can be switched back and forth.

$$J_{c0} = \frac{2e\alpha M_s t_F (H_K + H_{ext} + 2\pi M_s)}{\hbar\eta} \quad (\text{B.8})$$

where α the damping constant, M_s the saturation magnetization, t_F the thickness of free layer, η the spin-transfer efficiency, H_K the anisotropy field, and H_{ext} the external applied magnetic field.

There are three modes for the current-driven magnetization switching: thermal activation associated with switching time longer than 10 nanoseconds, precessional switching associated with switching time less than a few nanoseconds, and dynamic reversal as a compound process of both. The above three modes reveal the switching time and current density relationship.

For the fast precessional switching, the switching time is reversely proportional to the applied current,

$$\tau_p \propto \frac{1}{(J - J_{c0})} \ln\left(\frac{\pi}{2\theta_0}\right) \quad (\text{B.9})$$

where θ_0 is the initial magnetization angle deviated from the easy axis. At finite tem-

perature, θ_0 is determined by thermal distribution. For the fast processional switching in the regime of nanosecond, it usually takes a current density that is several times greater than J_{c0} .

In the region of slow thermal activated switch, the switching current is dependent on the thermal stability factor $\Delta = K_u V / k_B T$ and the current pulse width. Interestingly, the current density can be smaller than the critical density, and therefore is useful for current reduction. In this case, the standard thermal agitation will be assisted by spin-current, which introduces extra energy to reach enough energy for the magnetization switching. The relation reads

$$J(\tau) = J_{c0} [1 - \frac{K_u V}{k_B T} \ln(\frac{\tau}{\tau_0})] \quad (\text{B.10})$$

where $\tau_0 \propto 1ns$ is the attempt frequency inverse, $K_u V$ the anisotropy energy.

As the fast processional switching requires large current density which reduces the robustness and cause undesired switching, and the slow thermal activation process takes too long time, the dynamic reversal mode with intermediate current pulses corresponds to the operating speed of practical STT-RAM. However, its explicit formula is hard to be derived due to its complicated process. Therefore, the dynamic reversal is usually studied as a combination of processional and thermally activated switching.

B.4 Magnetization Dynamics

Magnetic domains are formed by the competition among the various energy terms involved in a magnetic system. The energy of a magnetic structure is the sum of the exchange energy, the anisotropy energy, Zeeman energy and the demagnetization energy. The system naturally reaches equilibrium when its overall free energy is minimized. Since the magnitude of the magnetization cannot change, the way to minimize the energy is to vary the direction of the magnetization. The exchange energy seeks to align the spins with each other, the anisotropy energy seeks to align the spins with an axis determined by the crystal structure, the Zeeman energy aligns the spins with an external field. When the magneto static dipole-dipole interaction is also taken into account, known as the demagnetization energy, a nonuniform magnetization will generally be found as the lowest compromise of overall energy. Short range exchange energy will prevail a configuration with the spins aligned, large range dipole-dipole interaction will however prevail a magnetic state with minimal net magnetization. In the macro-spin model of magnetization dynamics study, the short-range exchange energy can be ignored. The

energy associated with anisotropy field can be written as

$$\varepsilon = K(1 - m_x^2) \quad (\text{B.11})$$

where K is the anisotropy constant, m_x the normalized magnetization in x-direction, defined as the in-plane easy axis.

For Zeeman energy by external applied field, we have

$$\varepsilon = -\mu_0 M H_{ext} \quad (\text{B.12})$$

in which μ_0 is called the vacuum permeability.

Demagnetization field represents the work necessary to assemble magnetic poles in a given geometrical configuration. Since the thickness is much smaller than the in-plane dimensions, the dominant term is dominated by demagnetizing field in a uniformly magnetized thin film, that is $H_D = [0, 0, -M_s m_z]$. Its associated energy density is

$$\varepsilon = -\frac{1}{2} \mu_0 M H_D = -\frac{1}{2} \mu_0 M_s^2 H_D^2 \quad (\text{B.13})$$

Combining all three terms together, we have the overall energy,

$$\varepsilon = K(1 - m_x^2) + \frac{1}{2} \mu_0 M_s^2 m_z^2 - \mu_0 M \cdot H_{ext} \quad (\text{B.14})$$

The equivalent effective field H_{eff} in the LLG Equation B.7 therefore becomes

$$H_{eff} = -\frac{1}{\mu_0 M_s} \frac{\delta \varepsilon}{\delta m} = [H_x^{ext} + H_K m_x, 0, -M_s m_z] \quad (\text{B.15})$$

where $H_K = 2K/(\mu_0 M_s)$ is the anisotropy field.

In dimensionless form, we have

$$\omega = \frac{1}{2} Q(1 - m_x^2) + \frac{1}{2} m_z^2 - m \cdot h^{ext} \quad (\text{B.16})$$

and

$$h_{eff} = -\frac{\delta \omega}{\delta m} = [h_x^{ext} + Q m_x, 0, -m_z] \quad (\text{B.17})$$

with $Q = 2K/(\mu_0 M_s^2) = H_K/M_s$. Since the magnitude of m does not change, which suggests that it can be rewritten in spherical coordinates,

$$\frac{d\theta}{d\tau} = h_\phi - \alpha \sin\theta \frac{d\phi}{d\tau} \quad (\text{B.18})$$

$$\sin\theta \frac{d\phi}{d\tau} = -h_\theta + \alpha \frac{deh_\theta}{d\tau} \quad (\text{B.19})$$

by multiplying α to Equation B.18, and add Equation B.19, and multiplying α to Equation B.19, and minus Equation B.19, we could obtain a set of first order differential equations

$$(1 + \alpha^2) \frac{d\theta}{d\tau} = h_\phi - \alpha h_\theta \quad (\text{B.20})$$

$$(1 + \alpha^2) \sin\theta \frac{d\phi}{d\tau} = -h_\theta + \alpha h_\phi \quad (\text{B.21})$$

Generally, the damping constant $\alpha \ll 1$, therefore we can have approximation $1 + \alpha^2 \cong 1$. In addition, when focus on trajectories that is close to equilibrium, it can be approximated $\theta = \pi/2 + \xi$ where ξ and ϕ are small perturbation around equilibrium. Then we have,

$$\frac{d\xi}{d\tau} = h_\phi - \alpha h_\theta \quad (\text{B.22})$$

$$\frac{d\phi}{d\tau} = -h_\theta + \alpha h_\phi \quad (\text{B.23})$$

with

$$h_\theta = -(1 + Q + h_x^{ext})\xi - \chi\theta \quad (\text{B.24})$$

$$h_\phi = +\chi\xi - (Q + h_x^{ext})\phi \quad (\text{B.25})$$

Let $u = Q + h_x^{ext}$, and λ be the first derivative operator $d/d\tau$, the first order differential equation then reads,

$$\lambda^2 + [\alpha + 2(\alpha u - \chi)]\lambda + u(1 + u) = 0 \quad (\text{B.26})$$

solution precession

$$\theta = e^{-\frac{t}{t_0}} \cos(\omega t + \Phi_0) \quad (\text{B.27})$$

where

$$t_0 = \frac{1}{\gamma_0 M_s (\chi_{crit} - \chi)} \quad (\text{B.28})$$

$$\omega = \gamma_0 M_s \sqrt{u(1 + u) - (\chi_{crit} - \chi)^2} \quad (\text{B.29})$$

with

$$\chi_{crit} = \alpha \left(\frac{1}{2} + Q + h_x^{ext} \right) \cong \frac{\alpha}{2} \quad (\text{B.30})$$

if $Q, h_x^{ext} \ll 1$.

B.5 Domain-wall Propagation

Recalling spin-transfer torque effect that when a current is passed through a ferromagnetic material, electrons will polarize, that is, the spin of the conduction electron will align with the spin of the local electrons carrying the magnetic moment of the material. When the conduction electrons subsequently enter a region of opposite magnetization they will eventually become polarized again, thereby transferring their spin momentum to the local magnetic moment, as required by the law of conservation of momentum. Therefore, when many electrons are traversing a domain-wall(DW), magnetization from one side of the DW will be transferred to the other side. Effectively the electrons are able to push the DW in the direction of the electron flow i .

The influence of current on DW dynamics is often treated by adding two spin torque terms in the LLG equation, Equation B.6. When the current, with current density J , is flowing in one direction, the x-direction the LLG equation including the spin torque terms can be written as

$$\frac{\partial M}{\partial t} = -\gamma M \times H + \frac{\alpha}{M_s} M \times \frac{\partial M}{\partial t} - \eta J \frac{\partial M}{\partial x} + \beta \eta J M \times \frac{\partial M}{\partial x} \quad (\text{B.31})$$

where last two terms are added to the regular LLG equation to describe the effect of current on the magnetization dynamics. The first of these terms expresses the adiabatic spin transfer torque as exerted by a current on magnetic DWs with η the strength of the effect. The second STT term in the equation describes the non-adiabatic current induced effect which relative strength is parameterized by β . The strength of the adiabatic spin torque, η , is widely agreed on [147, 148, 149, 150] and given by:

$$\eta = \frac{g\mu_B P}{2eM_s} \quad (\text{B.32})$$

where g is the Land factor, μ_B the Bohr magneton, P the electron polarization and M_s the saturation magnetization, all of which the values are very well know except for the electron polarization. Estimates for P range from $P = 0.4$ to $P = 0.7$ [151].

Appendix C

Publication list

C.1 Book

1. Hao Yu, and *Yuhao Wang*, Design Exploration of Nano-scale Non-volatile Memory Devices, *Springer Publishing*, 2014.

C.2 Tool Developed

1. The main author of *NVM-SPICE*: <http://www.nvmspice.org/>.

C.3 Journal

1. *Yuhao Wang*, Hao Yu, and Wei Zhang, Nonvolatile CBRAM-Crossbar-Based 3-D-Integrated Hybrid Memory for Data Retention, *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)*, vol.22, no.5, pp.957,970, May 2014.
2. *Yuhao Wang*, Hao Yu, and Guangbin Huang, An Energy-efficient Nonvolatile In-memory Computing Architecture for Extreme Learning Machine by Domain-wall Nanowire Devices, *IEEE Transactions on Nanotechnology (TNANO)*, 2014. (Submitted)
3. *Yuhao Wang* and Hao Yu, An Energy Efficient In-Memory AES Encryption by Nonvolatile Domain-wall Nanowire, *IEEE Transactions on COMPUTER-AIDED DESIGN of Integrated Circuits and Systems (TCAD)*, 2014. (Submitted)

C.4 Conference

1. Yuhao Wang, Hao Yu, and Dennis Sylvester, Energy Efficient In-Memory AES Encryption Based on Nonvolatile Domain-wall Nanowire, *ACM/IEEE Design Automation and Test Conference in Europe (DATE)*, March 2014.
2. Hao Yu, Yuhao Wang, Shuai Chen, Wei Fei, Chuliang Weng, Junfeng Zhao, and Zhulin Wei, Energy Efficient In-memory Machine Learning for Data Intensive Image-processing by Non-volatile Domain-Wall Memory, *IEEE/ACM Asia and South Pacific Design Automation Conference (ASP-DAC) (Special Session)*, January 2014.
3. Yuhao Wang and Hao Yu, An Ultralow-power Memory-based Big-data Computing Platform by Nonvolatile Domain-wall Nanowire Devices, *ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED)*, September 2013.
4. Yuhao Wang, Pingfan Kong, and Hao Yu, Logic-in-memory based Big-data Computing by Nonvolatile Domain-wall Nanowire Devices, *IEEE Non-Volatile Memory Technology Symposium (NVMTS)*, August 2013.
5. Yuhao Wang, Yang Shang, and Hao Yu, Design of Single Saw-tooth-pulse based STT-RAM Readout Circuit by NVM SPICE, *IEEE Non-Volatile Memory Technology Symposium (NVMTS)*, October 2012.
6. Yuhao Wang, Wei Fei, and Hao Yu, SPICE Simulator for Hybrid CMOS Memristor Circuit and System, *International Workshop on Cellular Nanoscale Networks and their Applications & Memristor and Memristive System Symposium*, August 2012 (Invited).
7. Yuhao Wang, Chun Zhang, Hao Yu, and Wei Zhang, Design of Low Power 3D Hybrid Memory by Non-volatile CBRAM-Crossbar with Block-level Data-retention, *ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED)*, August 2012.
8. Yuhao Wang and Hao Yu, Design Exploration of Ultra-Low Power Non-volatile Memory based on Topological Insulator, *ACM/IEEE International Symposium on Nanoscale Architectures (NANOARCH)*, July 2012.
9. Yuhao Wang, Chun Zhang, Revanth Nadipalli, Hao Yu, and Roshan Weerasekera, Design Exploration of 3D Stacked Non-Volatile Memory by Conductive Bridge

based Crossbar, *IEEE International 3D System Integration Conference (3DIC)*,
January 2012.

Bibliography

- [1] What is big data? bringing big data to the enterprise. Accessed: 2014-07-21. [Online]. Available: <http://www.ibm.com/big-data/us/en/>
- [2] M. Hilbert and P. López, “The worlds technological capacity to store, communicate, and compute information,” *Science*, vol. 332, no. 6025, pp. 60–65, 2011.
- [3] 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,” in *Electron Devices Meeting, 2005. IEDM Technical Digest. IEEE International*. IEEE, 2005, pp. 459–462.
- [4] T. Kawahara, R. Takemura, K. Miura, J. Hayakawa, S. Ikeda, Y. Lee, R. Sasaki, Y. Goto, K. Ito, I. Meguro *et al.*, “2mb spin-transfer torque ram (sparam) with bit-by-bit bidirectional current write and parallelizing-direction current read,” in *Solid-State Circuits Conference, 2007. ISSCC 2007. Digest of Technical Papers. IEEE International*. IEEE, 2007, pp. 480–617.
- [5] H. P. Wong, S. Raoux, S. Kim, J. Liang, J. P. Reifenberg, B. Rajendran, M. Asheghi, and K. E. Goodson, “Phase change memory,” *Proceedings of the IEEE*, vol. 98, no. 12, pp. 2201–2227, 2010.
- [6] F. Bedeschi, R. Bez, C. Boffino, E. Bonizzoni, E. Buda, G. Casagrande, L. Costa, M. Ferraro, R. Gastaldi, O. Khouri *et al.*, “4-mb mosfet-selected phase-change memory experimental chip,” in *Solid-State Circuits Conference, 2004. ESSCIRC 2004. Proceeding of the 30th European*. IEEE, 2004, pp. 207–210.
- [7] M. Kund, G. Beitel, C.-U. Pinnow, T. Rohr, J. Schumann, R. Symanczyk, K.-D. Ufert, and G. Muller, “Conductive bridging ram (cbram): An emerging non-volatile memory technology scalable to sub 20nm,” in *Electron Devices Meeting, 2005. IEDM Technical Digest. IEEE International*. IEEE, 2005, pp. 754–757.

- [8] R. Williams, “How we found the missing memristor,” *Spectrum, IEEE*, vol. 45, no. 12, pp. 28–35, 2008.
- [9] S. S. Parkin, M. Hayashi, and L. Thomas, “Magnetic domain-wall racetrack memory,” *Science*, vol. 320, no. 5873, pp. 190–194, 2008.
- [10] 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.
- [11] B. Jacob, S. Ng, and D. Wang, *Memory systems: cache, DRAM, disk*. Morgan Kaufmann, 2010.
- [12] Y. Song, H. Yu, S. M. Pudukotai Dinakarrao, and G. Shi, “Sram dynamic stability verification by reachability analysis with consideration of threshold voltage variation,” in *Proceedings of the 2013 ACM international symposium on International symposium on physical design*. ACM, 2013, pp. 43–49.
- [13] A. Lacey, “Thermal runaway in a non-local problem modelling ohmic heating: Part 1: Model derivation and some special cases,” *European Journal of Applied Mathematics*, vol. 6, no. 2, pp. 127–144, 1995.
- [14] R. Schatz and C. Bethea, “Steady state model for facet heating leading to thermal runaway in semiconductor lasers,” *Journal of applied physics*, vol. 76, no. 4, pp. 2509–2521, 1994.
- [15] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, “The missing memristor found,” *Nature*, vol. 453, no. 7191, pp. 80–83, 2008.
- [16] S. Dietrich, M. Angerbauer, M. Ivanov, D. Gogl, H. Hoenigschmid, M. Kund, C. Liaw, M. Markert, R. Symanczyk, L. Altimime *et al.*, “A nonvolatile 2-mbit cbram memory core featuring advanced read and program control,” *Solid-State Circuits, IEEE Journal of*, vol. 42, no. 4, pp. 839–845, 2007.
- [17] “Micron announces availability of phase change memory for mobile devices,” <http://investors.micron.com/releasedetail.cfm?ReleaseID=692563>, accessed: 2014-12-18.
- [18] “Second generation mram: Spin torque technology,” <http://www.everspin.com/products/second-generation-st-mram.html>, accessed: 2014-12-18.

- [19] L. Chua, “Memristor-the missing circuit element,” *Circuit Theory, IEEE Transactions on*, vol. 18, no. 5, pp. 507–519, 1971.
- [20] S. H. Jo, K.-H. Kim, and W. Lu, “High-density crossbar arrays based on a si memristive system,” *Nano letters*, vol. 9, no. 2, pp. 870–874, 2009.
- [21] Q. Xia, W. Robinett, M. W. Cumbie, N. Banerjee, T. J. Cardinali, J. J. Yang, W. Wu, X. Li, W. M. Tong, D. B. Strukov *et al.*, “Memristor- cmos hybrid integrated circuits for reconfigurable logic,” *Nano letters*, vol. 9, no. 10, pp. 3640–3645, 2009.
- [22] B. Li, Y. Shan, M. Hu, Y. Wang, Y. Chen, and H. Yang, “Memristor-based approximated computation,” in *Low Power Electronics and Design (ISLPED), 2013 IEEE International Symposium on*. IEEE, 2013, pp. 242–247.
- [23] M. Hu, H. Li, Q. Wu, and G. S. Rose, “Hardware realization of bsb recall function using memristor crossbar arrays,” in *Proceedings of the 49th Annual Design Automation Conference*. ACM, 2012, pp. 498–503.
- [24] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, “Nanoscale memristor device as synapse in neuromorphic systems,” *Nano letters*, vol. 10, no. 4, pp. 1297–1301, 2010.
- [25] M. N. Kozicki, M. Balakrishnan, C. Gopalan, C. Ratnakumar, and M. Mitkova, “Programmable metallization cell memory based on ag-ge-s and cu-ge-s solid electrolytes,” in *Non-Volatile Memory Technology Symposium, 2005*. IEEE, 2005, pp. 7–pp.
- [26] U. Russo, D. Kamalanathan, D. Ielmini, A. L. Lacaita, and M. N. Kozicki, “Study of multilevel programming in programmable metallization cell (pmc) memory,” *Electron Devices, IEEE Transactions on*, vol. 56, no. 5, pp. 1040–1047, 2009.
- [27] M. Tada, T. Sakamoto, N. Banno, M. Aono, H. Hada, and N. Kasai, “Nonvolatile crossbar switch using tio_x/tasioy solid electrolyte,” *IEEE transactions on electron devices*, vol. 57, no. 8, pp. 1987–1995, 2010.
- [28] T. Sakamoto, K. Lister, N. Banno, T. Hasegawa, K. Terabe, and M. Aono, “Electronic transport in ta₂o₅ resistive switch,” *Applied Physics Letters*, vol. 91, no. 9, pp. 092 110–092 110, 2007.

- [29] C. Schindler, S. P. Thermadam, R. Waser, and M. N. Kozicki, "Bipolar and unipolar resistive switching in cu-doped sio₂," *Electron Devices, IEEE Transactions on*, vol. 54, no. 10, pp. 2762–2768, 2007.
- [30] M. Haemori, T. Nagata, and T. Chikyow, "Impact of cu electrode on switching behavior in a cu/hfo₂/pt structure and resultant cu ion diffusion," *Applied Physics Express*, vol. 2, no. 6, p. 1401, 2009.
- [31] W. Guan, S. Long, Q. Liu, M. Liu, and W. Wang, "Nonpolar nonvolatile resistive switching in cu doped zro₂," *Electron Device Letters, IEEE*, vol. 29, no. 5, pp. 434–437, 2008.
- [32] M. N. Baibich, J. Broto, A. Fert, F. N. Van Dau, F. Petroff, P. Etienne, G. Creuzet, A. Friederich, and J. Chazelas, "Giant magnetoresistance of (001) fe/(001) cr magnetic superlattices," *Physical Review Letters*, vol. 61, no. 21, p. 2472, 1988.
- [33] B. Engel, J. Akerman, B. Butcher, R. Dave, M. DeHerrera, M. Durlam, G. Grynkewich, J. Janesky, S. Pietambaram, N. Rizzo *et al.*, "A 4-mb toggle mram based on a novel bit and switching method," *Magnetics, IEEE Transactions on*, vol. 41, no. 1, pp. 132–136, 2005.
- [34] D. Gogl, C. Arndt, J. C. Barwin, A. Bette, J. DeBrosse, E. Gow, H. Hoenigschmid, S. Lammers, M. Lamorey, Y. Lu *et al.*, "A 16-mb mram featuring bootstrapped write drivers," *Solid-State Circuits, IEEE Journal of*, vol. 40, no. 4, pp. 902–908, 2005.
- [35] T. W. Andre, J. J. Nahas, C. K. Subramanian, B. J. Garni, H. S. Lin, A. Omair, and W. L. Martino Jr, "A 4-mb 0.18- μ m 1t1mtj toggle mram with balanced three input sensing scheme and locally mirrored unidirectional write drivers," *Solid-State Circuits, IEEE Journal of*, vol. 40, no. 1, pp. 301–309, 2005.
- [36] W. J. Gallagher and S. S. Parkin, "Development of the magnetic tunnel junction mram at ibm: From first junctions to a 16-mb mram demonstrator chip," *IBM Journal of Research and Development*, vol. 50, no. 1, pp. 5–23, 2006.
- [37] Y. Wang and H. Yu, "An ultralow-power memory-based big-data computing platform by nonvolatile domain-wall nanowire devices," in *Low Power Electronics and Design (ISLPED), 2013 IEEE International Symposium on*. IEEE, 2013, pp. 329–334.
- [38] J. E. Moore, "The birth of topological insulators," *Nature*, vol. 464, no. 7286, pp. 194–198, 2010.

- [39] D. Hsieh, D. Qian, L. Wray, Y. Xia, Y. S. Hor, R. Cava, and M. Z. Hasan, “A topological dirac insulator in a quantum spin hall phase,” *Nature*, vol. 452, no. 7190, pp. 970–974, 2008.
- [40] Y. Xia, D. Qian, D. Hsieh, L. Wray, A. Pal, H. Lin, A. Bansil, D. Grauer, Y. Hor, R. Cava *et al.*, “Observation of a large-gap topological-insulator class with a single dirac cone on the surface,” *Nature Physics*, vol. 5, no. 6, pp. 398–402, 2009.
- [41] D. Hsieh, Y. Xia, D. Qian, L. Wray, J. Dil, F. Meier, J. Osterwalder, L. Patthey, J. Checkelsky, N. Ong *et al.*, “A tunable topological insulator in the spin helical dirac transport regime,” *Nature*, vol. 460, no. 7259, pp. 1101–1105, 2009.
- [42] Y. Chen, J. Analytis, J.-H. Chu, Z. Liu, S.-K. Mo, X.-L. Qi, H. Zhang, D. Lu, X. Dai, Z. Fang *et al.*, “Experimental realization of a three-dimensional topological insulator, Bi_2Te_3 ,” *Science*, vol. 325, no. 5937, pp. 178–181, 2009.
- [43] H. Horii, J. Yi, J. Park, Y. Ha, I. Baek, S. Park, Y. Hwang, S. Lee, Y. Kim, K. Lee *et al.*, “A novel cell technology using n-doped GeSbTe films for phase change ram,” in *VLSI Technology, 2003. Digest of Technical Papers. 2003 Symposium on*. IEEE, 2003, pp. 177–178.
- [44] H.-R. Oh, B.-h. Cho, W. Y. Cho, S. Kang, B.-g. Choi, H.-j. Kim, K.-s. Kim, D.-e. Kim, C.-k. Kwak, H.-g. Byun *et al.*, “Enhanced write performance of a 64-mb phase-change random access memory,” *Solid-State Circuits, IEEE Journal of*, vol. 41, no. 1, pp. 122–126, 2006.
- [45] S. Kang, W. Y. Cho, B.-H. Cho, K.-J. Lee, C.-S. Lee, H.-R. Oh, B.-G. Choi, Q. Wang, H.-J. Kim, M.-H. Park *et al.*, “A 0.1- μm 1.8-v 256-mb phase-change random access memory (pram) with 66-mhz synchronous burst-read operation,” *Solid-State Circuits, IEEE Journal of*, vol. 42, no. 1, pp. 210–218, 2007.
- [46] S. Manipatruni, D. E. Nikonov, and I. A. Young, “Modeling and design of spintronic integrated circuits,” *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 59, no. 12, pp. 2801–2814, 2012.
- [47] W. Fei, H. Yu, W. Zhang, and K. S. Yeo, “Design exploration of hybrid cmos and memristor circuit by new modified nodal analysis,” *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 20, no. 6, pp. 1012–1025, 2012.
- [48] H. Yu and W. Fei, “A new modified nodal analysis for nano-scale memristor circuit simulation,” in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*. IEEE, 2010, pp. 3148–3151.

- [49] Y. Shang, W. Fei, and H. Yu, "Analysis and modeling of internal state variables for dynamic effects of nonvolatile memory devices," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 59, no. 9, pp. 1906–1918, 2012.
- [50] —, "Fast simulation of hybrid cmos and stt-mtj circuits with identified internal state variables," in *Design Automation Conference (ASP-DAC), 2012 17th Asia and South Pacific*. IEEE, 2012, pp. 529–534.
- [51] Y. Wang, W. Fei, and H. Yu, "Spice simulator for hybrid cmos memristor circuit and system," in *Cellular Nanoscale Networks and Their Applications (CNNA), 2012 13th International Workshop on*. IEEE, 2012, pp. 1–6.
- [52] Y. Wang, C. Zhang, R. Nadipalli, H. Yu, and R. Weerasekera, "Design exploration of 3d stacked non-volatile memory by conductive bridge based crossbar," in *3D Systems Integration Conference (3DIC), 2011 IEEE International*. IEEE, 2012, pp. 1–6.
- [53] Y. Wang, Y. Shang, and H. Yu, "Design of non-destructive single-sawtooth pulse based readout for stt-ram by nvm-spice," in *Non-Volatile Memory Technology Symposium (NVMTS), 2012 12th Annual*. IEEE, 2012, pp. 68–72.
- [54] Y. Wang and H. Yu, "Design exploration of ultra-low power non-volatile memory based on topological insulator," in *Nanoscale Architectures (NANOARCH), 2012 IEEE/ACM International Symposium on*. IEEE, 2012, pp. 30–35.
- [55] Y. Wang, H. Yu, and W. Zhang, "Nonvolatile cbram-crossbar-based 3-d-integrated hybrid memory for data retention," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2013.
- [56] Y. Wang, C. Zhang, H. Yu, and W. Zhang, "Design of low power 3d hybrid memory by non-volatile cbram-crossbar with block-level data-retention," in *Proceedings of the 2012 ACM/IEEE international symposium on Low power electronics and design*. ACM, 2012, pp. 197–202.
- [57] C. W. Smullen, V. Mohan, A. Nigam, S. Gurumurthi, and M. R. Stan, "Relaxing non-volatility for fast and energy-efficient stt-ram caches," in *High Performance Computer Architecture (HPCA), 2011 IEEE 17th International Symposium on*. IEEE, 2011, pp. 50–61.
- [58] W. Xu, H. Sun, X. Wang, Y. Chen, and T. Zhang, "Design of last-level on-chip cache using spin-torque transfer ram (stt ram)," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 19, no. 3, pp. 483–493, 2011.

- [59] M. K. Qureshi, V. Srinivasan, and J. A. Rivers, “Scalable high performance main memory system using phase-change memory technology,” *ACM SIGARCH Computer Architecture News*, vol. 37, no. 3, pp. 24–33, 2009.
- [60] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger, “Architecting phase change memory as a scalable dram alternative,” *ACM SIGARCH Computer Architecture News*, vol. 37, no. 3, pp. 2–13, 2009.
- [61] H. Yu, Y. Wang, S. Chen, W. Fei, C. Weng, J. Zhao, and Z. Wei, “Energy efficient in-memory machine learning for data intensive image-processing by non-volatile domain-wall memory,” in *Design Automation Conference (ASP-DAC), 2014 19th Asia and South Pacific*. IEEE, 2014.
- [62] Y. Wang, P. Kong, and H. Yu, “Logic-in-memory based mapreduce computing by nonvolatile domain-wall nanowire devices,” in *Non-Volatile Memory Technology Symposium (NVMTS), 2013 13th Annual*. IEEE, 2013.
- [63] Y. Wang, P. Kong, H. Yu, and D. Sylvester, “Energy efficient in-memory aes encryption based on nonvolatile domain-wall nanowire,” in *Design Automation and Test Conference in Europe (DATE)*. IEEE, 2014.
- [64] Y. Wang, H. Huang, L. Ni, H. Yu, M. Yan, C. Weng, W. Yang, and J. Zhao, “An energy-efficient non-volatile in-memory dataanalytic for sparse-represented face recognition,” in *Design Automation and Test Conference in Europe (DATE)*. IEEE, 2015 (to appear).
- [65] S. Matsunaga, J. Hayakawa, S. Ikeda, K. Miura, H. Hasegawa, T. Endoh, H. Ohno, and T. Hanyu, “Fabrication of a nonvolatile full adder based on logic-in-memory architecture using magnetic tunnel junctions,” *Applied Physics Express*, vol. 1, no. 9, p. 1301, 2008.
- [66] S. Matsunaga, J. Hayakawa, S. Ikeda, K. Miura, T. Endoh, H. Ohno, and T. Hanyu, “Mtg-based nonvolatile logic-in-memory circuit, future prospects and issues,” in *Proceedings of the Conference on Design, Automation and Test in Europe*. European Design and Automation Association, 2009, pp. 433–435.
- [67] W. H. Kautz, “Cellular logic-in-memory arrays,” *Computers, IEEE Transactions on*, vol. 100, no. 8, pp. 719–727, 1969.
- [68] H. Kimura, T. Hanyu, M. Kameyama, Y. Fujimori, T. Nakamura, and H. Takasu, “Complementary ferroelectric-capacitor logic for low-power logic-in-memory vlsi,” *Solid-State Circuits, IEEE Journal of*, vol. 39, no. 6, pp. 919–926, 2004.

- [69] T. Hanyu, K. Teranishi, and M. Kameyama, “Multiple-valued logic-in-memory vlsi based on a floating-gate-mos pass-transistor network,” in *Solid-State Circuits Conference, 1998. Digest of Technical Papers. 1998 IEEE International*. IEEE, 1998, pp. 194–195.
- [70] L. W. Nagel and D. O. Pederson, *SPICE: Simulation program with integrated circuit emphasis*. Electronics Research Laboratory, College of Engineering, University of California, 1973.
- [71] C.-W. Ho, A. Ruehli, and P. Brennan, “The modified nodal approach to network analysis,” *Circuits and Systems, IEEE Transactions on*, vol. 22, no. 6, pp. 504–509, 1975.
- [72] S. Shin, K. Kim, and S.-M. Kang, “Compact models for memristors based on charge-flux constitutive relationships,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 29, no. 4, pp. 590–598, 2010.
- [73] Y. Ho, G. M. Huang, and P. Li, “Dynamical properties and design analysis for nonvolatile memristor memories,” *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 58, no. 4, pp. 724–736, 2011.
- [74] J. D. Harms, F. Ebrahimi, X. Yao, and J.-P. Wang, “Spice macromodel of spin-torque-transfer-operated magnetic tunnel junctions,” *Electron Devices, IEEE Transactions on*, vol. 57, no. 6, pp. 1425–1430, 2010.
- [75] Y. N. Joglekar and S. J. Wolf, “The elusive memristor: properties of basic electrical circuits,” *European Journal of Physics*, vol. 30, no. 4, p. 661, 2009.
- [76] S. Yu and H.-S. Wong, “Compact modeling of conducting-bridge random-access memory (cbram),” *Electron Devices, IEEE Transactions on*, vol. 58, no. 5, pp. 1352–1360, 2011.
- [77] C. Gopalan, Y. Ma, T. Gallo, J. Wang, E. Runnion, J. Saenz, F. Koushan, P. Blanchard, and S. Hollmer, “Demonstration of conductive bridging random access memory (cbram) in logic cmos process,” *Solid-State Electronics*, vol. 58, no. 1, pp. 54–61, 2011.
- [78] H. Yu and Y. Wang, “Nonvolatile state identification and nvm spice,” in *Design Exploration of Emerging Nano-scale Non-volatile Memory*. Springer, 2014, pp. 45–83.

- [79] A. Afifi, A. Ayatollahi, and F. Raissi, "Implementation of biologically plausible spiking neural network models on the memristor crossbar-based cmos/nano circuits," in *Circuit Theory and Design, 2009. ECCTD 2009. European Conference on*. IEEE, 2009, pp. 563–566.
- [80] G. Snider, "Self-organized computation with unreliable, memristive nanodevices," *Nanotechnology*, vol. 18, no. 36, p. 365202, 2007.
- [81] Y. V. Pershin, S. La Fontaine, and M. Di Ventra, "Memristive model of amoeba learning," *Physical Review E*, vol. 80, no. 2, p. 021926, 2009.
- [82] Y. V. Pershin and M. Di Ventra, "Experimental demonstration of associative memory with memristive neural networks," *Neural Networks*, vol. 23, no. 7, pp. 881–886, 2010.
- [83] W. Brinkman, R. Dynes, and J. Rowell, "Tunneling conductance of asymmetrical barriers," *Journal of applied physics*, vol. 41, no. 5, pp. 1915–1921, 1970.
- [84] X. Wang, W. Zhu, M. Siegert, and D. Dimitrov, "Spin torque induced magnetization switching variations," *Magnetics, IEEE Transactions on*, vol. 45, no. 4, pp. 2038–2041, 2009.
- [85] P. Grünberg, R. Schreiber, Y. Pang, M. Brodsky, and H. Sowers, "Layered magnetic structures: Evidence for antiferromagnetic coupling of fe layers across cr interlayers," *Physical Review Letters*, vol. 57, no. 19, p. 2442, 1986.
- [86] T. Fujita, M. B. A. Jalil, and S. G. Tan, "Topological insulator cell for memory and magnetic sensor applications," *Applied Physics Express*, vol. 4, no. 9, p. 094201, 2011. [Online]. Available: <http://apex.jsap.jp/link?APEX/4/094201/>
- [87] X.-L. Qi, Y.-S. Wu, and S.-C. Zhang, "Topological quantization of the spin hall effect in two-dimensional paramagnetic semiconductors," *Physical Review B*, vol. 74, no. 8, p. 085308, 2006.
- [88] P. Nenzi and V. Holger. (2010) Ngspice users manual. [Online]. Available: <http://ngspice.sourceforge.net>
- [89] R. Koch, J. Deak, D. Abraham, P. Trouilloud, R. Altman, Y. Lu, W. Gallagher, R. Scheuerlein, K. Roche, and S. Parkin, "Magnetization reversal in micron-sized magnetic thin films," *Physical review letters*, vol. 81, no. 20, p. 4512, 1998.

- [90] ITRS. (2010) International technology roadmap of semiconductor. [Online]. Available: <http://www.itrs.net>
- [91] R. Venkatesan, V. Kozhikkottu, C. Augustine, A. Raychowdhury, K. Roy, and A. Raghunathan, "Tapecache: a high density, energy efficient cache based on domain wall memory," in *Proceedings of the 2012 ACM/IEEE international symposium on Low power electronics and design*. ACM, 2012, pp. 185–190.
- [92] D. Chiba, G. Yamada, T. Koyama, K. Ueda, H. Tanigawa, S. Fukami, T. Suzuki, N. Ohshima, N. Ishiwata, Y. Nakatani *et al.*, "Control of multiple magnetic domain walls by current in a co/ni nano-wire," *Applied Physics Express*, vol. 3, no. 7, p. 3004, 2010.
- [93] K.-H. Kim, S. Gaba, D. Wheeler, J. M. Cruz-Albrecht, T. Hussain, N. Srinivasa, and W. Lu, "A functional hybrid memristor crossbar-array/cmos system for data storage and neuromorphic applications," *Nano letters*, vol. 12, no. 1, pp. 389–395, 2011.
- [94] S. Kaeriyama, T. Sakamoto, H. Sunamura, M. Mizuno, H. Kawaura, T. Hasegawa, K. Terabe, T. Nakayama, and M. Aono, "A nonvolatile programmable solid-electrolyte nanometer switch," *Solid-State Circuits, IEEE Journal of*, vol. 40, no. 1, pp. 168–176, 2005.
- [95] M. H. Ben-Jamaa, P.-E. Gaillardon, F. Clermidy, I. O'Connor, D. Sacchetto, G. De Micheli, and Y. Leblebici, "Silicon nanowire arrays and crossbars: Top-down fabrication techniques and circuit applications," *Science of Advanced Materials*, vol. 3, no. 3, pp. 466–476, 2011.
- [96] W. Y. Park, G. H. Kim, J. Y. Seok, K. M. Kim, S. J. Song, M. H. Lee, and C. S. Hwang, "A pt/tio₂/ti schottky-type selection diode for alleviating the sneak current in resistance switching memory arrays," *Nanotechnology*, vol. 21, no. 19, p. 195201, 2010.
- [97] K. Gopalakrishnan, R. Shenoy, C. Rettner, K. Virwani, D. Bethune, R. Shelby, G. Burr, A. Kellock, R. King, K. Nguyen *et al.*, "Highly-scalable novel access device based on mixed ionic electronic conduction (miec) materials for high density phase change memory (pcm) arrays," in *VLSI Technology (VLSIT), 2010 Symposium on*. IEEE, 2010, pp. 205–206.

- [98] A. Chen and M.-R. Lin, “Variability of resistive switching memories and its impact on crossbar array performance,” in *Reliability Physics Symposium (IRPS), 2011 IEEE International*. IEEE, 2011, pp. MY–7.
- [99] G. Jeong, W. Cho, S. Ahn, H. Jeong, G. Koh, Y. Hwang, and K. Kim, “A 0.24- μ m 2.0-v 1t1mtj 16-kb nonvolatile magnetoresistance ram with self-reference sensing scheme,” *Solid-State Circuits, IEEE Journal of*, vol. 38, no. 11, pp. 1906–1910, 2003.
- [100] S. Schechter, G. H. Loh, K. Straus, and D. Burger, “Use ecp, not ecc, for hard failures in resistive memories,” in *ACM SIGARCH Computer Architecture News*, vol. 38, no. 3. ACM, 2010, pp. 141–152.
- [101] S. Thoziyoor, N. Muralimanohar, J. H. Ahn, and N. P. Jouppi, “Cacti 5.1,” *HP Laboratories, April*, vol. 2, 2008.
- [102] E. Seevinck, P. J. van Beers, and H. Ontrop, “Current-mode techniques for high-speed vlsi circuits with application to current sense amplifier for cmos sram’s,” *Solid-State Circuits, IEEE Journal of*, vol. 26, no. 4, pp. 525–536, 1991.
- [103] C. Xu, X. Dong, N. P. Jouppi, and Y. Xie, “Design implications of memristor-based ram cross-point structures,” in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011*. IEEE, 2011, pp. 1–6.
- [104] X. Dong, N. P. Jouppi, and Y. Xie, “Pcrsim: System-level performance, energy, and area modeling for phase-change ram,” in *Proceedings of the 2009 International Conference on Computer-Aided Design*. ACM, 2009, pp. 269–275.
- [105] Y. Chen, H. Li, X. Wang, W. Zhu, W. Xu, and T. Zhang, “A nondestructive self-reference scheme for spin-transfer torque random access memory (stt-ram),” in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010*. IEEE, 2010, pp. 148–153.
- [106] H. Zhao, B. Glass, P. K. Amiri, A. Lyle, Y. Zhang, Y.-J. Chen, G. Rowlands, P. Upadhyaya, Z. Zeng, J. Katine *et al.*, “Sub-200 ps spin transfer torque switching in in-plane magnetic tunnel junctions with interface perpendicular anisotropy,” *Journal of Physics D: Applied Physics*, vol. 45, no. 2, p. 025001, 2012.
- [107] G. Rowlands, T. Rahman, J. Katine, J. Langer, A. Lyle, H. Zhao, J. Alzate, A. Kovalev, Y. Tserkovnyak, Z. Zeng *et al.*, “Deep subnanosecond spin torque switch-

- ing in magnetic tunnel junctions with combined in-plane and perpendicular polarizers,” *Applied Physics Letters*, vol. 98, no. 10, pp. 102 509–102 509, 2011.
- [108] H. Zhao, A. Lyle, Y. Zhang, P. Amiri, G. Rowlands, Z. Zeng, J. Katine, H. Jiang, K. Galatsis, K. Wang *et al.*, “Low writing energy and sub nanosecond spin torque transfer switching of in-plane magnetic tunnel junction for spin torque transfer random access memory,” *Journal of Applied Physics*, vol. 109, no. 7, pp. 07C720–07C720, 2011.
- [109] H.-P. Trinh, W. Zhao, J.-O. Klein, Y. Zhang, D. Ravelsona, and C. Chappert, “Domain wall motion based magnetic adder,” *Electronics letters*, vol. 48, no. 17, pp. 1049–1051, 2012.
- [110] S.-J. Lee and C.-S. Ouyang, “A neuro-fuzzy system modeling with self-constructing rule generation and hybrid svd-based learning,” *Fuzzy Systems, IEEE Transactions on*, vol. 11, no. 3, pp. 341–353, 2003.
- [111] J. Lin and C. Dyer, “Data-intensive text processing with mapreduce,” *Synthesis Lectures on Human Language Technologies*, vol. 3, no. 1, pp. 1–177, 2010.
- [112] J. Dean and S. Ghemawat, “Mapreduce: simplified data processing on large clusters,” *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.
- [113] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hennes, D. R. Hower, T. Krishna, S. Sardashti *et al.*, “The gem5 simulator,” *ACM SIGARCH Computer Architecture News*, vol. 39, no. 2, pp. 1–7, 2011.
- [114] J. Talbot, R. M. Yoo, and C. Kozyrakis, “Phoenix++: modular mapreduce for shared-memory systems,” in *Proceedings of the second international workshop on MapReduce and its applications*. ACM, 2011, pp. 9–16.
- [115] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, “Mcpat: an integrated power, area, and timing modeling framework for multi-core and manycore architectures,” in *Microarchitecture, 2009. MICRO-42. 42nd Annual IEEE/ACM International Symposium on*. IEEE, 2009, pp. 469–480.
- [116] G. H. Loh, “3d-stacked memory architectures for multi-core processors,” in *ACM SIGARCH Computer Architecture News*, vol. 36, no. 3. IEEE Computer Society, 2008, pp. 453–464.
- [117] C.-H. Hua, T.-S. Cheng, and W. Hwang, “Distributed data-retention power gating techniques for column and row co-controlled embedded sram,” in *Memory*

Technology, Design, and Testing, 2005. MTDT 2005. 2005 IEEE International Workshop on. IEEE, 2005, pp. 129–134.

- [118] J. Xie, X. Dong, and Y. Xie, “3d memory stacking for fast checkpointing/restore applications,” in *3D Systems Integration Conference (3DIC), 2010 IEEE International.* IEEE, 2010, pp. 1–6.
- [119] M. Koga, M. Iida, M. Amagasaki, Y. Ichida, M. Saji, J. Iida, and T. Sueyoshi, “First prototype of a genuine power-gatable reconfigurable logic chip with feram cells,” in *Field Programmable Logic and Applications (FPL), 2010 International Conference on.* IEEE, 2010, pp. 298–303.
- [120] J. L. Hennessy and D. A. Patterson, *Computer architecture: a quantitative approach.* Elsevier, 2012.
- [121] S. Bird, A. Phansalkar, L. K. John, A. Mericas, and R. Indukuru, “Performance characterization of spec cpu benchmarks on intels core microarchitecture based processor,” in *SPEC Benchmark Workshop, 2007.*
- [122] Y. Shang, C. Zhang, H. Yu, C. S. Tan, X. Zhao, and S. K. Lim, “Thermal-reliable 3d clock-tree synthesis considering nonlinear electrical-thermal-coupled tsv model,” in *ASP-DAC, 2013,* pp. 693–698.
- [123] H. Qin, Y. Cao, D. Markovic, A. Vladimirescu, and J. Rabaey, “Sram leakage suppression by minimizing standby supply voltage,” in *Quality Electronic Design, 2004. Proceedings. 5th International Symposium on.* IEEE, 2004, pp. 55–60.
- [124] T. Nagai, M. Wada, H. Iwai, M. Kaku, A. Suzuki, T. Takai, N. Itoga, T. Miyazaki, H. Takenaka, T. Hojo *et al.*, “A 65nm low-power embedded dram with extended data-retention sleep mode,” in *Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers. IEEE International.* IEEE, 2006, pp. 567–576.
- [125] J.-P. Kaps and B. Sunar, “Energy comparison of aes and sha-1 for ubiquitous computing,” in *Emerging Directions in Embedded and Ubiquitous Computing.* Springer, 2006, pp. 372–381.
- [126] S.-Y. Lin and C.-T. Huang, “A high-throughput low-power aes cipher for network applications,” in *Proceedings of the 2007 Asia and South Pacific Design Automation Conference.* IEEE Computer Society, 2007, pp. 595–600.

- [127] Z. Abid, A. Alma'Aitah, M. Barua, and W. Wang, "Efficient cmol gate designs for cryptography applications," *Nanotechnology, IEEE Transactions on*, vol. 8, no. 3, pp. 315–321, 2009.
- [128] X. Wang, Y. Chen, H. Li, D. Dimitrov, and H. Liu, "Spin torque random access memory down to 22 nm technology," *Magnetics, IEEE Transactions on*, vol. 44, no. 11, pp. 2479–2482, 2008.
- [129] R. Usselmann. (2002) Advanced encryption standard / rijndael ip core. [Online]. Available: http://opencores.org/project,aes_core
- [130] C. Augustine, A. Raychowdhury, B. Behin-Aein, S. Srinivasan, J. Tschanz, V. K. De, and K. Roy, "Numerical analysis of domain wall propagation for dense memory arrays," in *Electron Devices Meeting (IEDM), 2011 IEEE International*. IEEE, 2011, pp. 17–6.
- [131] K. Malbrain. (2009) Byte-oriented-aes: A public domain byte-oriented implementation of aes in c. [Online]. Available: <https://code.google.com/p/byte-oriented-aes/>
- [132] T. S. Furey, N. Cristianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler, "Support vector machine classification and validation of cancer tissue samples using microarray expression data," *Bioinformatics*, vol. 16, no. 10, pp. 906–914, 2000.
- [133] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural processing letters*, vol. 9, no. 3, pp. 293–300, 1999.
- [134] B. Yegnanarayana, *Artificial neural networks*. PHI Learning Pvt. Ltd., 2004.
- [135] M. T. Hagan, H. B. Demuth, M. H. Beale *et al.*, *Neural network design*. Pws Pub. Boston, 1996.
- [136] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006.
- [137] —, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, vol. 2. IEEE, 2004, pp. 985–990.
- [138] L. An and B. Bhanu, "Image super-resolution by extreme learning machine," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*. IEEE, 2012, pp. 2209–2212.

- [139] S. Park, M. Qazi, L.-S. Peh, and A. P. Chandrakasan, “40.4 fj/bit/mm low-swing on-chip signaling with self-resetting logic repeaters embedded within a mesh noc in 45nm soi cmos,” in *Proceedings of the Conference on Design, Automation and Test in Europe*. EDA Consortium, 2013, pp. 1637–1642.
- [140] V. Kumar, R. Sharma, E. Uzunlar, L. Zheng, R. Bashirullah, P. Kohl, M. S. Bakir, and A. Naeemi, “Airgap interconnects: Modeling, optimization, and benchmarking for backplane, pcb, and interposer applications,” *Components, Packaging and Manufacturing Technology, IEEE Transactions on*, vol. PP, no. 99, pp. 1–1, 2014.
- [141] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *Image Processing, IEEE Transactions on*, vol. 13, no. 4, pp. 600–612, 2004.
- [142] L. Berger, “Emission of spin waves by a magnetic multilayer traversed by a current,” *Physical Review B*, vol. 54, no. 13, p. 9353, 1996.
- [143] J. C. Slonczewski, “Current-driven excitation of magnetic multilayers,” *Journal of Magnetism and Magnetic Materials*, vol. 159, no. 1, pp. L1–L7, 1996.
- [144] M. Tsoi, A. Jansen, J. Bass, W.-C. Chiang, M. Seck, V. Tsoi, and P. Wyder, “Excitation of a magnetic multilayer by an electric current,” *Physical Review Letters*, vol. 80, no. 19, p. 4281, 1998.
- [145] J. Sun, “Current-driven magnetic switching in manganite trilayer junctions,” *Journal of magnetism and magnetic materials*, vol. 202, no. 1, pp. 157–162, 1999.
- [146] J. Katine, F. Albert, R. Buhrman, E. Myers, and D. Ralph, “Current-driven magnetization reversal and spin-wave excitations in co/cu/co pillars,” *Physical Review Letters*, vol. 84, no. 14, p. 3149, 2000.
- [147] L. Berger, “Low-field magnetoresistance and domain drag in ferromagnets,” *Journal of Applied Physics*, vol. 49, no. 3, pp. 2156–2161, 1978.
- [148] 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.
- [149] Z. Li and S. Zhang, “Domain-wall dynamics and spin-wave excitations with spin-transfer torques,” *Physical review letters*, vol. 92, no. 20, p. 207203, 2004.

- [150] A. Thiaville, Y. Nakatani, J. Miltat, and Y. Suzuki, “Micromagnetic understanding of current-driven domain wall motion in patterned nanowires,” *EPL (Europhysics Letters)*, vol. 69, no. 6, p. 990, 2005.
- [151] G. Beach, M. Tsoi, and J. Erskine, “Current-induced domain wall motion,” *Journal of magnetism and magnetic materials*, vol. 320, no. 7, pp. 1272–1281, 2008.