Efficient storage codes for non-volatile memories

Chua, Melissa Wan Jun

2016

Chua, M. W. J. (2016). Efficient storage codes for non-volatile memories. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/68782>

<https://doi.org/10.32657/10356/68782>

**Efficient Storage Codes for
Non-volatile Memories**

CHUA WAN JUN MELISSA

Chua Wan Jun Melissa

2015

School of Electrical & Electronic Engineering

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

2015

ACKNOWLEDGMENTS

I would like to express my special appreciation and thanks to my supervisors, Associate Professor Goh Wang Ling and Assistant Professor (Adjunct) Cai Kui for their tremendous guidance and help rendered along my way to my Ph.D. Being able to work and receive advice on my work with one the best researchers has been an absolute pleasure and honor throughout my past four years.

I wish to express my deepest gratitude to Assoc. Prof. Goh Wang Ling from Nanyang Technological University, Singapore. I am truly thankful to her for the high quality supervision and enthusiastic help provided to me through my Ph.D. Her time in polishing my writing and providing constructive criticism has carved me into who I am today.

I would also like to express my heartfelt gratefulness to Asst. Prof. (Adj) Cai Kui from Data Storage Institute, A*Star, Singapore. She has wholeheartedly dedicated her quality time in guiding me throughout my journey. She provided me with new ideas and critical comments that guided my research into new and fascinating directions. Without her help and supervision, my work will not be in its current form now. I am extremely indebted to her.

I would like to thank Data Storage Institute for allowing me to work on my thesis and to the people there who have helped me in one way or another. I am also appreciative that during my stay in Data Storage Institute that I had the privilege to discuss my Ph.D. research work with Associate Professor Andrew Jiang from Texas A&M University, USA, one of the best researchers in the research area of coding for non-volatile memories. He shared with me many invaluable suggestions for my research and I am thankful for his insights.

Last but not the least, I would like to give special thanks to my parents John and Shirleen, siblings Charmaine and Bernard, friends and Richelle. I am grateful for the love, endless support and the sacrifice made. This thesis is dedicated to them.

CONTENTS

Acknowledgments	ii
Contents	iii
Abstract	vi
List of Abbreviations	ix
List of Figures	x
List of Tables	xii
Chapter 1. Introduction	1
1.1 Non-volatile Memory Technologies	1
1.1.1 Flash Memory	3
1.1.2 Phase Change Memory	5
1.1.3 Other Emerging Technologies	7
1.2 Major Technical Challenges	10
1.2.1 Cell Errors Due to Increased Density	10
1.2.2 Endurance	11
1.2.3 Latency	12
1.3 Motivation and Scope of Current Work	13
1.4 Contributions of the Thesis	15
1.5 Organization of the Thesis	18
Chapter 2. Channel Modeling of Phase Change Memory (PCM)	20
2.1 Fundamentals of PCM	20
2.1.1 Device Operation	21
2.1.2 Threshold Switching	22
2.1.3 Reliability of Scaled PCM Cells	25
2.2 Channel Modeling of PCM	26
2.2.1 Cycling Endurance	26
2.2.2 Resistance Drift	28
2.2.3 Analytical Model	32
2.3 Conclusion	35

Chapter 3.	Binary Write-Once Memory (WOM) Codes	37
3.1	The State-of-the-art WOM-Codes	38
3.2	Basic Structure of a Single-level WOM	40
3.3	Two-Write WOM-Codes	44
3.3.1	A Two-write WOM-Code Construction	44
3.3.2	Maximizing the WOM-Rate	52
3.4	Newly Designed Two-write WOM-Codes	56
3.4.1	Restricted WOM-Codes	57
3.4.2	Unrestricted WOM-Codes	58
3.5	WOM-Rate Analysis	60
3.6	Conclusion	61
Chapter 4.	Non-binary WOM-Codes	63
4.1	Basic Structure of a Multi-level WOM	64
4.2	Code Construction for Non-binary WOM-Codes	66
4.2.1	Code Construction A	67
4.2.2	Code Construction B	70
4.3	Code Construction Comparison	74
4.3.1	WOM-codes Designed from Construction A	74
4.3.2	WOM-codes Designed from Construction B	76
4.4	Conclusion	77
Chapter 5.	Enumerative Coding for Rank Modulation (RM) Codes	79
5.1	Prior-art Codes Based on RM Schemes	80
5.1.1	Full RM Codes	80
5.1.2	Partial RM Codes	81
5.2	Simplified Enumerative Coding Scheme	82
5.2.1	Simplified Enumerative Coding for RM	82
5.2.2	Encoding Scheme	85
5.2.3	Decoding Scheme	88
5.2.4	Algorithm Implementation	89
5.3	A Novel Enumerative Coding Scheme	91
5.3.1	Preliminaries	92
5.3.2	Encoding Scheme	95
5.3.3	Decoding Scheme	99
5.4	Complexity Comparison	102
5.4.1	Encoder Complexity Comparison	103
5.4.2	Decoder Complexity Comparison	104

Abstract	v
<hr/>	
5.5 Conclusion	105
Chapter 6. RM Codes for Resistance Drift	107
6.1 Brief Summary of RM Codes	108
6.2 Hybrid RM Codes to Combat Resistance Drift	110
6.2.1 Hybrid RM Codes	110
6.2.2 Code Rate Analysis	113
6.3 Performance Evaluation of Various RM Codes	117
6.3.1 Simulation Setup	117
6.3.2 Investigation of Varying Parameter ρ	125
6.3.3 Investigation of Varying Codeword Length	128
6.3.4 Investigation of Varying RM Codes	130
6.4 Conclusion	132
Chapter 7. Conclusion of Thesis	133
7.1 Conclusions	133
7.2 Suggestions for Future Work	137
Publications Originated from This Thesis	140
Bibliography	141

ABSTRACT

The onset of the mobile age and the rapid growth of the mobile technology have initiated a tremendous demand for high density storage and retrieval of huge amount of data. The conventional magnetic recording systems are expected to reach storage density limits. In the recent years, the non-volatile memory (NVM) systems have shown a high potential for future ultra-high density data storage systems. Compared with magnetic recording systems, the NVM systems have numerous attractive features, which include lower power consumption, faster read access time, better mechanical reliability, compactness and shock resistance. Flash memory is currently the most mature storage medium in portable electronic devices and is rapidly being introduced into mobile gadgets and data-intensive computer systems. Current semiconductor market researches has revealed that non-volatile memory devices based in flash technology will reach sales in the order of \$26 billion by end of 2010 and \$51.2 billion in 2015 [1]. Due to the maturity of Flash technology, it is proposed to set the benchmark for other emerging NVM technologies. The trend of consumer electronics indicates that constant growing demand for an increase in data storage capacity. Examples are the megapixel race in consumer digital cameras. As an example, the Apple QuickTake 100 launched by Apple in 1994 contained an image sensor with only 0.3 megapixels [2], nowadays cameras like Sony, possess a sensor with a resolution 24.6 megapixels [3]. In the same trend, digital video systems have increased resolution due to the new high definition HD video standards. All these new technologies require a higher amount of storage capacity in order to accommodate their media content, as a result, an

increment in the demand for higher capacity non-volatile memories is highly expected. The increasing market demand for high capacities non-volatile memory devices serves as a driving force to incentivize recent outgoing research in the non-volatile field.

Nevertheless, the physical constraints of Flash makes additional scaling a tremendously expensive task, therefore we exploit alternative technology to break through what the Flash technology has to offer. The phase-change random access memory (PCRAM) is one of the most promising alternative technologies for NVM and is one of the promising candidates to replace Flash. It is envisage that Phase Change Memory (PCM) technology will eventually become a mainstream NVM technology. Hence development of coding techniques for PCM is focused in this work.

While the technological innovations in the design and development of storage media and system are key to achieving high capacity storage systems, the role of advanced coding techniques is increasingly becoming crucial as a cost-effective means of improving density. Despite the favoured position of NVM in the mobile sector, there are poor reliability concerns related to such memories. These poor reliability issues were contributed from resistance drift over time, cells that are stuck in a particular state and inter-cell coupling. This dissertation applies variants of data storage codes to combat some of these reliability issues. Specifically, the core of the work is focused on two sets of issues – asymmetric errors caused by a shift in the cell's resistance over time and the failure contributed from the limitation of the maximum number of rewriting operations of a PCM cell.

Write-once Memory (WOM)-codes were proposed as an efficient storage codes in extending the lifetime of PCM cells, by minimizing the number of memory

degrading reset cycles. An efficient code construction method is briefly described before a critical criterion is proposed to design WOM-codes with high rates. From the WOM-rate analysis, the WOM-codes designed have an improvement of 2% for the unrestricted codes and 1% for the restricted codes. Subsequently, a novel construction method is proposed to achieve high rate non-binary WOM-codes that have a shorter codeword length compared to the best known WOM-codes. Achieving high WOM-rates with shorter codeword length is beneficial as coding efficiency is maximized with less complex encoding and decoding operations. A newly designed two-write 4-ary WOM-code, which is proposed in this work achieves an improvement in coding efficiency of 0.2%.

Next, the work is focused on proposing solutions to combat asymmetric errors. A novel scheme known as rank modulation (RM) was recently proposed for this purpose. As these codes are currently encoded and decoded based on a look-up table (LUT), a novel enumerative coding scheme is proposed. The enumerative coding scheme proposes to compute the lexicographic index analytically rather than adopting a brute-force enumeration technique. This results in a desirable reduction in encoding and decoding complexity. Additionally, a hybrid RM scheme is proposed with the motivation of providing a method of constructing WOM-codes without having to sacrifice as much code rate as the state-of-the-art RM codes, and yet able to provide better performance.

LIST OF ABBREVIATIONS

- BER:** Bit Error Rate
- ECC:**Error Correcting Codes
- FRAM:**Ferroelectric Random Access Memory
- FRM:** Full Rank Modulation
- GST:**Germanium-Antimony-Tellurium
- HRM:** Hybrid Rank Modulation
- LUT:** Look-up Table
- MLC:**Multi-level Cell
- MRAM:**Magnetic Random Access Memory
- MTJ:**Magnetic Tunnel Junctions
- NVM:**Non-volatile Memory
- PCM:**Phase Change Memory
- PCRAM:**Phase Change Random Access Memory
- PRM:**Partial Rank Modulation
- RM:**Rank Modulation
- RRAM:**Resistive Random Access Memory
- SLC:**Single-level Cell
- SR:**Structural Relaxation
- STT:**Spin-Transfer Torque
- WOM:**Write-once Memory

LIST OF FIGURES

1.1	A floating gate transistor.	4
1.2	The semi-circle represents the proportion of crystallized material in a 4-level PCM cell.	6
1.3	Electrical pulses during crystallization and amorphization of GST.	7
1.4	Configurations of parallel and anti-parallel MTJs.	8
1.5	A basic RRAM device structure.	9
1.6	Basic working principle of RRAM.	10
2.1	Temperature profile required for phase change material.	22
2.2	Reset state shows switching behavior at V_{th} [49].	24
2.3	Decomposition of PCM material due to electromigration.	28
2.4	The resistance distribution of multi-level PCM cell with 4 distinct levels. (a) First time the resistance level of PCM cell is read, $t = 0$ sec and (b) after some elapsed time, $t = T$ sec.	32
2.5	Resistance drift phenomenon seen in PCM cells programmed at different initial resistance levels.	33
2.6	Simulated resistance drift based on analytical model.	34
3.1	Block diagram of the i -th write encoder $\mathcal{E}_i$ and decoder $\mathcal{D}_i$.	42
4.1	Block diagram of the first write encoder and decoder.	65
4.2	Block diagram of the second write encoder and decoder.	65
5.1	Size of LUT as a function of codeword length n .	82
6.1	Cell states of 8 PCM cells with $q = 8$ distinct levels. The shaded boxes represent the current resistance state of each cell (e.g. cell 1 is at state “1” and cell 8 is at state 6).	112
6.2	Noise parameter β over the first measured logarithmic resistance R_{rv} .	119
6.3	Resistance level distribution of two distinct PCM cell states. For time (a) $t = 0$ sec and (b) $t = T$ sec.	123
6.4	Error rate for two cells x_1 and x_2 as a function of σ for fixed	124

List of Tables	xi
----------------	----

threshold detection against adaptive threshold detection, where $\xi_1 = 1/2$.	
6.5 Block diagram of RM coded PCM communication channel.	124
6.6 Simulation result from varying parameter ρ .	126
6.7 Code rate analysis of FRM, PRM and HRM with length $n = 16$.	128
6.8 Simulation result from varying parameter n .	129
6.9 BER channel performance of FRM, PRM and HRM keeping their code rate approximately consistent.	131

LIST OF TABLES

1.1	Latency of the read, program and erase operations of the DRAM, Flash and PCM technologies.	12
3.1	Data bits and the corresponding codewords of a two-write WOM-code.	39
3.2	Binary BCH codes parameters and their corresponding WOM-rates.	56
3.3	Breakdown of number of vectors for WOM-code with [15, 11]-BCH code being used as the base code.	58
3.4	Breakdown of number of vectors for WOM-code with [31, 21]-BCH code being used as the base code.	59
3.5	WOM-rate and coding efficiency comparison for unrestricted WOM-codes.	61
3.6	WOM-rate and coding efficiency comparison for restricted WOM-codes.	61
4.1	Mapping of Rivest and Shamir $[3,2:4,4]_2$ WOM-code.	72
4.2	First and second write mapping between information bits and codeword of C_4 .	73
4.3	WOM-rate comparison benchmarked against 4-ary WOM-codes constructed with code constructions [71].	75
4.4	WOM-rates of the WOM-codes constructed from BCH codes.	76
4.5	Comparison of WOM-rates between state-of-the-art WOM-code and the achieved WOM-codes from construction A and B.	77
5.1	Lexicographic ranking of RM codeword c and its corresponding entries for $v = (v_1, v_2, v_3)$.	94
5.2	Mapping of the RM codeword to its corresponding v and $i_c(c)$.	97
6.1	Code rate of FRM codes for $q = 4, 8, 16, 32$.	114
6.2	Code rate of PRM codes for $q = 4, 8, 16, 32$ and $\rho = n/2$.	115
6.3	Code rate of HRM codes for $q = 4, 8, 16, 32$ and $\rho = n/2$.	117
6.4	Simulation settings for PCM analytical channel model.	119
6.5	Improvement comparison between RM codes with varying ρ .	127

List of Tables

xiii

6.6	Improvement comparison between RM codes with varying n .	130
6.7	Performance improvement of RM codes benchmarked against the uncoded system.	131
7.1	Comparison of code rate improvement for the HRM against the PRM codes.	136
7.2	Performance evaluation for various RM codes benchmarking against the uncoded PCM system.	137
7.3	Efficiency comparison for best known WOM-codes with more than two writes.	138

CHAPTER 1

INTRODUCTION

The motivation of this thesis is focused on developing storage codes to extend the lifetime of data storage systems known as non-volatile memories. It is commonly known that NVMs have lifetime issues after a number of block erasures (for Flash memories) or number of reset operations (for phase change memories). These operations are deemed as destructive on the storage medium. Storage coding has the ability to maximize the number of times that information can be written prior to incurring a destructive operation. The thesis begins with a brief introduction to the non-volatile memories and the major technical challenges that are faced. Thereafter, the motivation and scope of this thesis are presented. Finally, the contributions and organization of the thesis are conferred in the last part of this chapter.

1.1 Non-volatile Memory Technologies

Non-volatile memory (NVM) is defined as a class of computer memory that has the ability to retain digital information after the power is removed. An example of a NVM is the punch card, whereby no power is required for it to maintain information. The predecessor of Flash memory is the EPROM, which is a three-layered polysilicon technology. The first

polysilicon is used as an erase gate, the second polysilicon is used as a floating gate, and the last polysilicon is used as a control gate. Programming can be performed by hot electron injection from the channel, similar to that of the ultraviolet erasure-type EPROM. The erasure is carried out by extracting the electrons from the floating gate to the erase gate for all cells at the same time. This EPROM structure is known as the Flash EEPROM because the complete memory could be erased very quickly. The EEPROM has a shorter erase time as compared to the EPROM. Due to various advantages of EEPROM over EPROM, industry has shifted development of EPROM to EEPROM.

In 1985, Mukherjee et al. proposed a source-erase type Flash memory cell, called ETOX (EPROM with tunnel oxide) cell. The structure of this cell is the same as that of EPROM. The cell is programmed to a high threshold state by means of channel hot electron injection, with the control gate and the drain connected to a high voltage. Erasing the cell is performed by Fowler-Nordheim tunneling of electrons from the floating gate to the source diffusion layer. Since then the performance of ETOX has been improved by several companies. In 1987, NAND structured cell was proposed by Masuoka et al. and the proposed structure reduces the cell size without scaling of the device dimensions. The NAND cell can realize a smaller cell area per bit than previous discussed architectures [4].

NVM devices are of paramount importance due to the booming market of the portable electronic systems such as the cell phones, digital cameras and tablets. In order for practical implementation into these portable electronic devices, numerous researches revolve around miniaturizing the NVM devices with reliability constraints satisfied. The Flash memory is currently the most popular NVM technology because the memory can be easily programmed and erased electrically. In order to further improve the recording density, Flash cells are shrunk to

tremendously small size. If the tunnel oxide were to be scaled below 2 nm, the operation voltage could be reduced from more than 10 V to below 4 V [5]. Unfortunately, the retention time would also be reduced, from 10 years to several seconds. This physical damage to the tunnel oxide during the cycling process causes data retention problems, program disturbance, read disturbance, and erratic characteristic behavior of the floating gate memory cell. Such problems severely limit the reliability and multilevel cell operation. This basic limitation of the tunnel oxide thickness becomes increasingly important with scaling [6]. Therefore, it is expedient to search for its successor. The emerging candidates that are able to further improve on storage capacity are the phase change memory (PCM), the magnetic random access memory (MRAM), and the resistive random access memory (RRAM).

These electrically accessed NVM can be categorized according to their programming mechanism. These programming mechanisms allow data to be stored in the storage region of the device, without having a constant power source attached to it. The different NVM candidates' architectures are briefly described below.

1.1.1 Flash Memory

Flash memory was first introduced in 1984 and was named after its capability to be programmed and erased at high speed [7]. A Flash memory is basically made up of an array of memory cells, whereby each cell is a floating gate transistor, illustrated in Figure 1.1. The floating gate structure consists of a source and drain, where voltages are applied during the reading and writing operations.

Figure 1.1: A floating gate transistor.

Due to the existence of the insulating layer between the floating gate and substrate, a high electric field is essential to force electrons to pass through the silicon-oxide insulation layer and get trapped in the floating gate. The process of pushing charges through an insulating layer is commonly known as FN Tunneling [17]. The same process is performed to remove charges from the floating gate during erasure. The presence of trapped charges on the floating gate alters the electric potential across the source and drain. So by applying specific voltages across the source and drain and by measuring the flow of current across, we can detect the state of the memory cell.

Since charges have to tunnel through an insulating layer to reach the floating gate, FN Tunneling operation requires a large voltage to be applied. The application of high voltage results in disturbance to nearby floating gate transistors, resulting in a possible change in state of neighboring cells. Additionally, due to the large voltage applied, tunneling puts a significant stress on the floating gate transistor. To reduce the impact of tunneling on each transistor, the erasure operation is performed at unit size of a block, which typically contains 2^{20} cells [10]. The transistors are arranged

such that they share a single p-substrate and all the transistors in the block can be tunneled at the same time during erasure. Due to this architecture feature, the stress on each transistor is significantly reduced.

To further improve storage efficiencies, multi-level cells (MLC) Flash cells with more than two distinct voltage levels are designed. For every q distinct level in each cell, $\log_2 q$ bits can be stored in a single cell. However different cell states corresponding to different information bits, careful programming operation have to be performed to ensure that no excess charge is tunneled into the floating gate. Whenever a cell is over-programmed, a memory degrading block erasure operation is required. The characteristics of MLC Flash memory result in a higher read and write operation latency, and the lifetime of each MLC to be less than SLC.

1.1.2 Phase Change Memory

Although Flash memory is the most commonly found memory device used in numerous electronic applications today, it is accepted by chip manufacturers that the continual reduction in size of a Flash memory cell has become significantly difficult. For this reason, manufacturers has been exploring for emerging technologies that can overcome the physical limit of Flash memories. It is notable that PCM is better suited to continue the trend of shrinking cost and speed compared to Flash and still holds the ability to store information even when power is removed. This emerging technology exploits the fact that PCM has the ability to reversibly switch between two stable states with a large resistance difference. By appropriately controlling the electrical signal supplied to the memory cell, the phase change

material can be efficiently switched between the amorphous (high resistance) and crystalline (low resistance) state, which have a huge resistance change. For amorphization the phase change material must be heated above its melting point (*e.g.* 635°C for germanium-antimony-tellurium (GST)), while crystallization is performed at a lower temperature (*e.g.* 150°C for GST).

A MLC PCM cell is illustrated in Figure 1.2, where an electrode supplying the electric current is attached to the phase change material. The semi-circle regions represent the lower resistance crystallized material, while the remaining region of the rectangle phase change material represents the higher resistance amorphous material. Since the crystallized material has a lower resistance and cell state c_1 has a largest proportion of crystallized material, then the resistances of the cell states are ranked as $c_1 < c_2 < c_3 < c_4$.

Figure 1.2: The semi-circle represents the proportion of crystallized material in a 4-level PCM cell.

For crystallization of the amorphous GST material, a small but long electrical signal with duration t_c and amplitude a_c is applied. The amount of crystallized region obtained after crystallization depends greatly on the amplitude of the applied signal. From Figure 1.3, we illustrate the amplitude required for each of

the distinct cell state c_1, c_2, c_3 and c_4 . In contrast, applying an electrical signal with a short duration of t_a and large amplitude of a_a avoids the crystallization process and allows the GST material to be stabilized at the highly resistive amorphous state.

Figure 1.3: Electrical pulses during crystallization and amorphization of GST.

The crystallization temperature of the GST material is much lower than the amorphization temperature, numerous research works are focused around crystallization-based PCM cells [14]. In other words, the amplitude a_c can be adjusted to obtain intermediate resistance levels corresponding to different crystallized regions. However, when the GST material is completely crystallized, a reset operation is performed to completely change the GST material into its amorphous phase. Similar to the erasure operation in Flash, the reset operation degrades the lifetime of PCM as a large electrical signal is required.

1.1.3 Other Emerging Technologies

Another emerging technology known as STT-MRAM stores data in magnetic storage elements commonly known as magnetic tunnel junctions (MTJ). The MTJs consists of two ferromagnetic plates, whereby one of them is a permanent magnet set to a fixed polarity. The other plate's field can be easily switched reversibly to store data. These two ferromagnetic plates are separated by an insulating layer, illustrated in Figure 1.4. The parallel configuration represents the low resistive state, while the anti-parallel state represents the high resistive state.

Figure 1.4: Configurations of parallel and anti-parallel MTJs.

In recent years, the spin-torque transfer (STT) effect was analytically developed in [16] and has evolved into a new generation of MRAMs, known simply as STT-MRAM. Previously without STT, the write currents are increased exponentially as the size of MRAM is scaled down. This undesirable effect results in electromigration and insufficient power limitations when the technology node of MRAM is reduced below 90nm, which is subsequently overcome by STT-MRAM. However a critical issue of STT-MRAM is the lack of an accurate macro model that incorporates the temperature and bias voltage effects. Without the model, the

behavior of the device with scaling is unpredictable. Therefore, more work has to be done to develop a more comprehensive model of STT-MRAM before evaluating its potential to replace the Flash memory.

The last emerging technology that is being discussed in this chapter is the RRAM. A basic memory cell is presented in Figure 1.5. The device structure consists of a metallic top and bottom electrode wedged between an insulating material. It utilizes an electric field that is applied to either electrode to induce resistance change in its storage material. Akin to the PCM, the resistance of the insulating material is varied as a voltage is administered. The basic working principle of RRAM demonstrated in Figure 1.6 illustrates the switching between a high resistance state and a low resistance state using a write or erase voltage.

Figure 1.5: A basic RRAM device structure.

Although the STT-MRAM and RRAM are interesting NVM technologies that could prove to be possible alternatives to Flash memories, they still require further testing and development to establish their feasibility. To successfully replace

an existing technology, the emerging one has to be better in terms of reliability and endurance expectations. Based on the analysis of the various NVM technologies, many manufacturers envisage that the PCM technology will eventually become a mainstream NVM technology to displace Flash. Therefore, the thesis is focused on the development of coding techniques for crystallization-based PCM.

Figure 1.6: Basic working principle of RRAM.

1.2 Major Technical Challenges

Emerging NVM technologies will revolutionize the way data can be stored. These memories have aroused huge attention to having numerous advantages. However, NVM technologies still has several issues to overcome in order to be as competitive as the conventional hard disk. The pitfalls of NVM technologies are listed below.

1.2.1 Cell errors Due to Increased Density

Multi-level cell (MLC) storage is the most efficient way of maximizing the storage density of a memory. As PCM has large resistance contrast between its two stable states – amorphous and crystalline phase, multiple cell levels can be divided in each cell, which allows more than one bit of digital data to be encoded.

Resistance drift is a common phenomenon exhibit by chalcogenide materials, whereby the resistance of amorphous state of the phase change material increases stochastically over time. Such an effect is particularly detrimental in MLC data storage, because stochastic fluctuations of the desired programmed resistance of closely divided levels might result in overlapping, and therefore errors during decoding.

1.2.2 Endurance

NVM technologies have a specified number of program/erase cycles, for example a Flash memory has 10^5 program/erase cycles before the wear begins to affect the integrity of the data being stored. A limitation of Flash memories is their block erasure operation. Cells in a Flash memory are arranged into blocks, whereby each block contains 2^{20} memory cells. Although the state of a cell can be increased individually, decreasing the state of a cell requires the entire block to be erased – implying that states of all cells in the block are reduced to the minimum value, before the cells are reprogrammed to their desired state. Such a block erasure operation is known to reduce the lifetime of the Flash memory; hence it is essential to minimize such operations for applications that require data to be modified frequently.

From Figure 1.3 the current required for amorphization of the phase change material is higher than the current required for crystallization. Therefore, the operation of converting the crystalline state to amorphous state exerts a greater strain on the memory cell. This operation is also known as the reset operation. When a

large current is repeatedly injected into the phase change material, the thermal expansion and contractions degrades the contact between the material and the heating element. Due to the degradation, the programming current is not reliably administered into the phase change material. As the material resistivity is directly dependent on the current injected, the variability in resistance is a result from the variability in current provided to the material, which leads to cells being deemed as unreliable, hence affecting the endurance of the memory cell. Typically, the number of writes before a PCM cell is deemed unreliable is between 10^7 to 10^8 [10].

1.2.3 Latency

In recent years, mobile electronic devices have become computationally intensive due to demand in high definition multimedia experience on the user interface. To meet expectations, high speed memories are required to cache information to be sent to the CPU. However, from Table 1.1, the proposed emerging NVM technologies, like Flash memories and PCM, still faced limitations in read, program/erase speed. Thus, for NVM technologies to meet future demands in the portable device market, much higher speed is required.

Table 1.1: Latency of the read, program and erase operations of the DRAM, Flash and PCM technologies.

	DRAM	Flash	PCM
Read Time	< 10ns	10 ns – 50 ns	60 ns
Program Time	< 10 ns	1000 ns	50 ns
Erase Time	< 10 ns	0.1 ms	120 ns

1.3 Motivation and Scope of Current Work

The development of mobile age and the rapid growth of mobile technologies have initiated humongous demand for high density storage and rapid retrieval of huge amount of data. The conventional magnetic recording systems are expected to reach storage density limits, opening up opportunities for NVM applications to be applied to future ultra-high density data storage systems. Compared with magnetic recording systems, the NVM systems have numerous attractive features, which include lower power consumption, faster read access time, better mechanical reliability, compactness and shock resistance. Although currently the most mature storage medium in mobile devices is Flash, physical constraints make additional scaling an extremely expensive task. When Flash memory cells are scaled down to the sub 20nm technological nodes, the number of electrons stored in the floating gate decreases. It has been shown in [10] that the critical number of electrons stored in the floating gate is only in the order of a few tens for technological nodes beyond 20nm. Additionally, the cell-to-cell interference is further amplified for these Flash cells. This results in Flash memory cells being unreliable when technological nodes are scaled below the sub 20nm levels. Therefore, PCM has been suggested as one of the most promising emerging technology to replace Flash. However, before PCM can successfully displace Flash memory, the search for suitable and efficient solutions to overcome PCM's limited lifetime and resistance drift problems has become an important task.

There are numerous research activities evolving around improving the performance of PCM through coding techniques. WOM-codes and RM codes were

reported to be viable solutions to PCM's limited lifetime issue and resistance drift phenomenon, respectively. WOM-codes aim to rewrite information without reducing any cell state values. WOM-code implementation allows minimal usage of memory degrading operations to be performed in the lifetime of a PCM device. The very first WOM-codes presented by Rivest and Shamir proved that WOM could be rewritten to a surprising degree [19]. This work was extensively cited by many and spurred many more research work to effectively maximize the number of rewrites on a WOM cell. Various code construction methods for WOM-codes have been reported in [20, 21]. The theoretical capacity region with zero error and the maximum number of information bits that can be stored in a WOM with a fixed number of successive rewrites is discussed in [22].

To combat the memory sensing errors due to shifts in the resistance value of each PCM cell, RM codes are recently proposed by [26]. By programming and ranking the cell levels to their relative values rather than the accurate absolute values, the RM codes have shown potential to avoid overshooting during programming and reduce the read errors due to the drift of threshold voltage or resistance values. Our study encompasses of the following steps.

- WOM-codes: Propose a criterion to efficiently search for binary WOM-codes with high WOM-rate. The high rate WOM-codes can subsequently be used to construct high rate WOM-codes with larger alphabet size.
- Enumerative coding scheme: A generalized enumerative coding scheme is customized for RM codes to tackle memory limitation issue arising from using a

LUT during encoding and decoding. Next a novel enumerative scheme is proposed and provides an advantage in computation complexity during both encoding and decoding.

- Hybrid RM codes: To provide a mean of constructing a RM code with less redundancy as compared to the partial RM code and has better performance than the full and partial RM codes.
- Performance evaluation: The various RM codes' performance is accessed using the PCM analytical model and the bit error rate (BER) to elapsed time is provided.

We remark all the novel coding schemes proposed in this thesis can be combined with prior-art error correction codes (ECC).

1.4 Contributions of the Thesis

This thesis makes contributions and reports original work on storage codes targeted at PCM system. The main contributions cover five parts.

- The first part of the work focuses on the application of the WOM codes as a rewriting code for extending the lifetime of NVM devices, specifically the PCM. Since having a very high rewriting ability would result in a very large code rate lost, which implies a large decrease in memory storage efficiency, the research work here considers only the WOM code with two writes. The novelty of this work is that a critical criterion is proposed to design binary

WOM-codes with coding efficiencies higher than the state-of-the-art codes [23]. WOM codes are categorized into unrestricted and restricted WOM codes. For every two-write WOM-code, the WOM-rate is the sum of the code rate for the first and second write, where code rate is defined as the amount of information bits that can be stored per cell. For the unrestricted two-write WOM-code, the highest WOM rate achieved from literature is 1.4632, while for the restricted two-write WOM-codes, the highest WOM rate achieved is 1.3750 [23]. Based on the proposed criterion, the highest achievable WOM rate for the unrestricted and restricted two-write WOM codes are increased to 1.4962 and 1.3909 respectively. From [23], the best known WOM-rates for the two-write WOM-codes are achieved from computer search without a practical code (e.g. encoder and decoder) design. However, based on the analysis of the code construction method, a linear code such as the binary BCH codes should be selected as the base code for code construction due to its large hamming distance property.

- The second contribution extends the work on the construction of binary WOM-codes to non-binary WOM-codes. The analysis here is solely for WOM-codes with two writes to minimize the amount of WOM-rate lost, where WOM-rate is the measure of the number of information bits that is stored in each memory storage cell. A code construction method to generate non-binary WOM-code that achieves WOM-rates comparable to the best known non-binary WOM-codes, but with a smaller codeword length and dimension

is proposed. This reduction is translated to a less complex encoding and decoding operation, which is a desirable effect.

- The third contribution focuses on eliminating the memory limitation issue arising from the use of a table look-up to map rank modulation codes to its corresponding data bits. The general enumerative coding algorithm proposed in [29] is customized for codes based on rank modulation. Furthermore, a novel enumerative coding algorithm that provides an advantage in computation complexity during both the encoding and decoding processes of long codewords is proposed. The novel coding scheme provided in this work operates in $O(n^2)$ time for both the encoder and decoder, which is proven later in the thesis to outperform the general enumerative coding scheme provided in [29].
- The fourth contribution is the proposal of a hybrid rank modulation code, whereby the rank modulation scheme is applied to some of the n cells. Akin to the partial rank modulation code, a parameter ρ such that only the highest ρ cells are ranked is defined. The lowest $n - \rho$ cells stores information bits similar to the scenario when no rank modulation is implemented. The motivation for a hybrid rank modulation code is to provide a means of constructing a rank modulation code with lesser redundancy as compared to the partial rank modulation code, and have a better system performance as compared to the full rank modulation code.

- The fifth contribution made in this research work is the performance evaluation of three types of rank modulation codes based on a phase change memory analytical model. This analytical channel model characterizes the stochastic fluctuations in the PCM cells over time [31].

1.5 Organization of the Thesis

The rest of this thesis is arranged in the following manner. Chapter 2 presents the device physics for PCM. The first part of the chapter reviews the fundamental device operations and threshold switching phenomenon. In the second part, the reliability issues faced by PCM and some open problems are discussed.

In Chapter 3, the binary WOM-codes are analyzed. The first part of the chapter introduces the primeval WOM-codes and subsequently a formal definition of WOM-codes is stated. Thereafter, a critical criterion is presented to design binary WOM-codes with coding efficiency 2% higher than prior-art WOM-codes with the same parameters. The chapter is concluded with WOM-rate analysis of two-write binary WOM-codes.

The investigation of WOM-codes is extended in Chapter 4, where non-binary WOM-codes are constructed. Akin to the binary WOM-codes, the study focuses on two-write WOM-codes since memory storage efficiency is of a higher priority in this work. A new construction method is proposed to achieve WOM-rates approximately equal to the best known WOM-rates, but with a smaller codeword length and dimension. The reduction in codeword length and dimension is welcome as it implies that the encoder and decoder operations are less complex. Using the

proposed criterion with an efficient non-binary code construction method described in [71], non-binary WOM-codes achieved a 1% improvement in coding efficiency.

RM codes have been reported to effectively combat asymmetric errors occurring in Flash [26, 30]. In the context of PCM, asymmetric errors arise from a commonly known phenomenon known as resistance drift. Therefore, the second part of this thesis has its attention on the RM codes. A novel enumerative coding scheme is proposed in Chapter 5 to resolve memory limitation due to the use of a table look-up during encoding and decoding for codes based on RM. The proposed enumerative scheme computes the lexicographic index of the RM codes analytically rather than obtaining it through a brute-force enumeration, which simplifies the encoding and decoding complexity by two orders of magnitude.

A hybrid RM scheme is then in Chapter 6. The motivation of the hybrid scheme is to provide a method of constructing a WOM-code without having to sacrifice as much code rate as the partial RM code, and yet able to provide better performance as compare to the full RM code. The effects of implementing various RM schemes to combat the effects of resistance drift is evaluated in the last part of this chapter.

CHAPTER 2

CHANNEL MODELING OF PHASE CHANGE MEMORY (PCM)

The Phase Change Memory (PCM) is an emerging non-volatile memory technology based on operations that are electrically initiated and reversible. High endurance, low programming power and excellent scalability potential are advantages that make PCM a promising candidate to replace Flash memory in future mobile applications. The fundamental processes and reliability issues faced by PCM will be reviewed in Chapter 2.

2.1 Fundamentals of PCM

A PCM device consists of an array of memory cells, and every cell usually comprises of two electrodes connected to two sides of a phase change material region [118]. The phase change material is more commonly known as GST chalcogenide glass. Phase change material exhibits two relatively stable phases - crystalline and amorphous. The two phases are individualized by their contrastive

electrical properties. More accurately, the amorphous phase has resistivity that can be several orders of magnitude higher than that of the crystalline phase.

Figure 1.2 illustrates a conventional structure for the PCM cell. The bottom electrode provides an electrical signal for supplying the necessary temperature to the GST, for switching between amorphous-crystalline phases. As a result of the huge resistance margin, a multitude of intermediate states between fully amorphous and fully crystalline phase can be realized with proper controlling of the electrical signal applied to the GST. Most frequently, PCM channel models represent each intermediate state by an amorphous fraction parameter [4-6].

2.1.1 Device Operation

The set and reset operations applied to the GST allows reversible switching between the amorphous and crystalline phase. Switching from the crystalline (low resistance) phase to the amorphous (high resistance) phase, the GST is first melted and then quenched rapidly by applying a large electrical signal for a short period of time. This operation is defined as the reset operation. After the reset operation, a region of amorphous material is developed. The set operation is performed to convert the amorphous material to crystalline. In contrary to the reset operation, a set operation uses a medium electrical signal for a sufficiently long duration so that the material will be crystallized. Comparatively, the set duration is longer than the reset operation. As shown in Figure 2.1, the PCM cells are programmed to the amorphous or crystalline phase with the appropriate electrical pulses. Information stored in a PCM cell can be easily deciphered by passing a small electrical signal, to

measure the resistance of the cell.

Figure 2.1: Temperature profile required for phase change material.

Typically, the resistance range of a PCM cell is between $10^4\Omega$ and $10^6\Omega$. Fine resistance control of the PCM cells proves to be tremendously challenging. Therefore, it is more meaningful to represent the resistance in the logarithmic domain. The large resistance contrast between the amorphous and crystalline phase allows multiple bits to be stored in a single cell. This is realized by having intermediate levels that are achieved through incomplete phase transition. Achieving multiple bits per cell is highly entrancing, since storage density can be greatly proliferated this way.

2.1.2 Threshold Switching

The reset state is defined as the high resistance state that is below the threshold voltage V_{th} (sub-threshold region) and displays threshold switching characteristic at V_{th} . The reset state is a reversible process if the voltage injected is rapidly withdrawn. However if the voltage applied is introduced for a time longer than what is required for crystallization, memory switching occurs and the targeted

cell is in the low resistance state. This state is also known as the set state. Figure 2.2 shows current-voltage (I-V) curves of the SET and RESET states [49]. The set programming operation relies on the threshold switching phenomenon found in the phase change material. When the value of the electric field across the amorphous material coincides with the threshold voltage V_{th} , the amorphous material resistance reduces to a lower state, which has resistivity corresponding to the crystalline phase. When the PCM is in the reset state, the cell's high resistance forms a barrier for Joule heating to crystallize the phase change material. The threshold switching behavior causes in a drop in resistance of the reset state and enables the set programming.

Threshold voltage of each PCM cell needs to be carefully defined such that the PCM cells are operating in their respective states perfectly. Firstly, read voltages should not exceed V_{th} to prevent any disturbance of the current state of the PCM cell, while having sufficient margin to separate the set and reset state. If the selected V_{th} is too small, it can lead to inaccurate read operation. Furthermore, programming pulse needs to administer voltage that is larger than V_{th} , such that the set operation of the PCM cell can be successfully executed from the reset state. Parameter V_{th} is not mandatory to be smaller than set and reset programming voltages, the reason being V_{th} is determined by the amorphous material size [50-52]. On the other hand, the programming voltages of the set and reset operations are due to the thermal efficiency of the phase change material. Therefore, too large V_{th} can raise the maximum voltage needed by PCM device.

Figure 2.2: Reset state shows switching behavior at V_{th} [49].

Despite the importance of threshold switching in PCM operations, the theories backing the phenomenon of threshold switching is not completely understood and several theoretical models have been proposed. The thermal instability model describes threshold switching induced by Joule heating [53]. This model is established by the consideration that current passing through the phase change material rises exponentially due to temperature-dependent conduction of the material with respect to an increase in temperature. In consideration of threshold voltage switching having higher speeds, electronic mechanisms are preferred over the purely thermal mechanisms [54]. An electric model was presented that features threshold switching in contrast to strong carrier generation that is a consequence of high electric field and large carrier density [55-57]. Another electronic model displayed energy gain of electrons in a high electric field, which leads to voltage-current instability [58].

Additionally, there is a crystallization model that refers threshold switching

to the actual crystallization, which is built on a nucleation model that is facilitated by electric field [59]. Reversible trait of threshold switching is related to the separation of premature crystalline embryos upon dislodging the electric field. The internal parameters of the models cannot be precisely determined due to a lack in detailed experimental affirmation. The main threshold switching phenomenon can vary for different phase change materials as hybrid models might be mandatory to validate all considerations.

2.1.3 Reliability of Scaled PCM Cells

As PCM devices calibrate into smaller technology nodes, scalability can be anticipated to be a relevant consideration. Studies performed on GeTe have displayed that nano-particles as minuscule as 1.8nm can switch between the amorphous and crystalline phases [65]. This proves that phase changing potential is not diminished when technology nodes are scaled down. However there are numerous other criteria for fabricating PCM devices from scaled technology node. In the nanometer range, boundary constraint becomes the leading contribution in deciding the overall general system configurations.

As the programmed region size is being scaled down, essential characteristic namely the melting and crystallization temperature varies. Studies have validated the change in material properties as size of the programming region is below 10nm [60]. The crystallization duration, temperatures and activation energies are decreased for scaled down programming materials [62, 63].

The electrical feature of PCM cells is calculated on the size of the phase change material, which has been validated that threshold switching phenomenon happens when the switching field is adhered [51]. Consequently, parameter V_{th} is decreased as PCM device is scaled down. This would complicate matters as a smaller V_{th} would imply unstable cells and slower reading operation. It has been proposed that set and reset states exhibits different scaling characteristics [64].

2.2 Channel Modeling of PCM

In order to pack more bits in a single cell, more distinguishable intermediate levels have to exist in a PCM cell. With the maximum and minimum resistance values of the cells kept constant, having more distinguishable intermediate levels would imply that the intervals between two consecutive levels are smaller. As the interval gets smaller, the reliability of the PCM cells becomes debatable. The two major challenges for PCM cells, namely cycling endurance and resistance drift, are hence analyzed in Section 2.2.1 and 2.2.2, respectively. In order to predict the behavior of both the cycling endurance and the resistance drift phenomenon, the analytical channel model is also provided at the last part of this section.

2.2.1 Cycling Endurance

Cycling endurance is the number of re-writes that can be applied to a cell before the cell is deemed as being unreliable. It has been reported in publications that the endurance of a PCM cells is approximated to be 10^9 set/reset cycles [10]. Thorough analysis of the failed cells had proven much difference in their electrical

characteristics [18]. The failures of cells are either due to void formation in the GST material (“stuck reset”) or the deterioration of the heating element of the cell (“stuck set”).

The electrical characteristic of a cell in “stuck reset” state has problems generating sufficient Joule heat due to the intermixing of elements from the bottom electrode and the GST material. The cell is thus unable to overcome its designed threshold voltage. These “stuck reset” cells are unable to efficiently switch between the desired amorphous and crystalline phase, resulting in the cell being stuck in the reset state.

Contrastively, cells in the “stuck set” state exhibits a reduction in its threshold voltage [18]. This is due to the GST redistribution caused by electromigration as illustrated in Figure 2.4. There are two main forces contributing to the redistribution of the GST material; the electrostatic force from the molten state and the hole wind-force from the crystalline state. Due to the valence difference for each atom in the molten state, represented by the area closer to the bottom cathode, the electrons are redistributed. The Te element attracts electrons and therefore becomes an anion and moves towards the anode. While the Ge and Sb elements lose electrons to become cations, causing them to move towards the cathode. On the other hand, the hole wind-force phenomenon is found in the crystalline material, represented by the region closer to the top anode. The conventional hole wind-force migrates all elements in the same lattice structure to move in the same direction as the electron flow. However, it has been reported that

for p-type poly-crystalline silicon, migration can be in the opposite direction. Since the crystalline GST material has p-type conductivity, the wind-force moves all elements towards the cathode.

Figure 2.3: Decomposition of PCM material due to electromigration.

The decomposition of the GST material due to electromigration causes a Sb and Ge rich region and another Te rich region, as illustrated in Figure 2.3. The decomposed GST material exhibits lower crystallization temperature and the quenched region is able to re-crystallize itself. This makes the production of amorphous GST a more difficult task. When the amorphous material is able to completely re-crystallize by itself, the cell is defined as a “stuck set”.

2.2.2 Resistance Drift

The dependency on reversible switching between amorphous and crystalline phases entails another reliability issue, known as resistance drift. In any

chalcogenide material, it is established that the amorphous material displays structural relaxation (SR) due to thermally accelerated atomic rearrangement in irregular configuration [2]. During SR, PCM cells' resistances increase due to defects annihilation, which is a reduction in the concentration of point defects of the amorphous volume [7-9, 13]. Since carrier transport in amorphous phase occurs by thermally activated hopping, it leads to an enhancement in resistance in the PCM cells. Therefore, the anticipated resistance at time t , $R(t)$ obeys the power-law [14-15], which is defined as

$$R(t) = R_0 \left(\frac{t}{t_0} \right)^\nu, \quad (2.1)$$

where R_0 is the resistance at the reference time t_0 and ν is the power-law exponent, which is a random variable proportional to the annealing temperature T . The power-law exponent is proportional to the annealing temperature because of thermal activation of the SR process. The SR time obeys the Arrhenius law [16] that is expressed as

$$\tau_{SR} = \tau_0 \exp \left(\frac{E_A}{k \times T} \right), \quad (2.2)$$

where τ_{SR} is the time required to attain a resistance value $\hat{R} \in [5M\Omega, 100M\Omega]$, τ_0 is the pre-exponential time constant, E_A is the activation energy, and k is the Boltzmann constant. The compensation (Meyer-Neldel) rule [17] further shows that τ_0 can be represented by

$$\tau_0 = \tau_{00} \exp\left(\frac{-E_A}{k \times T_{MN}}\right), \quad (2.3)$$

where τ_{00} is a pre-exponential time constant, $T_{MN} = 760K$ and $\tau_{00} = 4.2 \times 10^{-6}$. The compensation rule expressed in (2.3) has been validated from experimental results presented in [7]. From the compensation rule, the power law exponent ν of the resistance drift can be derived from combining equations (2.1) – (2.3)

$$\tau_{SR} = \tau_{00} \exp\left[\frac{E_A}{k} \left(\frac{1}{T} - \frac{1}{T_{MN}}\right)\right]. \quad (2.4)$$

Setting ν as the subject,

$$\begin{aligned} \nu &= \frac{kT \log(\hat{R}/R_0)}{E_A (1 - T/T_{MN})} \\ &= \alpha \frac{T}{1 - T/T_{MN}} \\ &= \frac{\alpha}{1/T - 1/T_{MN}}. \end{aligned} \quad (2.5)$$

Seen in [de7], it is evaluated that $\alpha = \frac{k \log(\hat{R}/R_0)}{E_A} = 2.5 \times 10^{-4} k^{-1}$. Therefore, from (2.5), the power-law exponent ν is directly proportional to the annealing temperature T .

PCM cells constitutes of chalcogenide alloy composition of Ge, Sb and Te that can either occur in low resistance or high resistance states. As the two extreme states have vastly different resistance, hybrid states can be constructed. These states have resistance that is between the low and high resistance states. Due to the

existence of the hybrid states, each cell can store n bits with each MLC cell having 2^n distinct states. An important observation to note is that as the number of distinct levels in a cell increases, the range between each pair of adjacent state becomes smaller.

Each MLC state has an upper and lower boundary resistance threshold limits. If the resistance of a cell is between the threshold limits, the cell is said to be represented by the corresponding state. The effect of mechanical relaxation in the amorphous state causes the resistance of the cell to increase with respect to time. This effect is commonly known as resistance drift. The resistance drift phenomenon is described by Figure 2.4. The top graph showed the probability density function (p.d.f) of the four different cell state levels during the first occurrence the PCM cell resistance level is read, denoted by $t = 0$ seconds. The bottom graph illustrates the probability density function of these four cell states after some elapsed time $t = T$ seconds. Detailed observation of the resistance drift phenomenon given in Figure 2.4 showed that some cells, which represent the information bits “10” at $t = 0$, but due to drift after T seconds, are categorized with the information bits “11” instead. This shift results in a memory sensing error. Another important observation to note from Figure 2.4 is that the distribution of the higher resistance cells experiences a greater effect from drifting. Therefore they have a higher probability of error. These drift errors are expected to be the main contributor for errors occurring in multi-level PCM cells [78].

Figure 2.4: The resistance distribution of multi-level PCM cell with 4 distinct levels. (a) First time the resistance level of PCM cell is read, $t = 0$ sec and (b) after some elapsed time, $t = T$ sec.

2.2.3 Analytical Model

The analytical channel model defined focuses on the resistance drift, which is a dominant source of ambiguity in the PCM cell. Akin to magnetic recording channels, the PCM communication channel has a sender which transmits data by storing information on the PCM medium, and a receiver who reads the information after an arbitrary amount of time. The typical behaviors of a PCM cell programmed at different initial resistance level are show in Figure 2.5.

Figure 2.5: Resistance drift phenomenon seen in PCM cells programmed at different initial resistance levels.

The phenomenon described is most commonly known as a resistance drift effect, whereby the logarithmic resistance has a general upward trend across a time period. However, it is imperative to observe that the increment can be either positive or negative. Most physical channel models proposed usually consider only a positive increment, but [31] presented a simple model that accounts for both positive and negative increments. A simple model describing the resistance drift behavior is expressed with

$$\log R(t) = \log R(t_{rv}) + \nu \log\left(\frac{t-t_0}{t_{rv}-t_0}\right) + \beta W\left[\log\left(\frac{t-t_0}{t_{rv}-t_0}\right)\right], \quad (2.6)$$

where ν is the drift parameter for the PCM cell, t_0 and t_{rv} are the times for which the cell was quenched and the first time after programming at which the cell is read and verified, respectively. Parameter W is a Wiener process with unit variance per

unit time, β is a noise parameter and $t \geq t_{rv}$. Experimental studies [31] demonstrated that both ν and β are dependent on the resistance value $R(t_{rv})$. In particular,

$$\nu(\log R(t_{rv})) = \nu_{max} \frac{\log R(t_{rv}) - \log R_{min}}{\log R_{max} - \log R_{min}} \tag{2.7}$$

$$\beta(\log R(t_{rv})) = a(\log R(t_{rv}))^2 + b\log R(t_{rv}) + c \tag{2.8}$$

where $\nu_{max} = 0.11$ is proven in experimental studies [31] and a, b, c are constants. Figure 2.6 illustrates the characterized channel model behavior, simulated with four equally spaced log-resistance levels.

Figure 2.6: Simulated resistance drift based on analytical model

2.3 Conclusion

PCM has made vast improvements over the past decade. A decade or so ago, this abandon technology helped point the way to an efficient crystallizing memory device. At that period, any apprehension about the future of Flash technology was secured by the assurance of Ferroelectric RAM (FRAM) and Magnetic RAM (MRAM). However, both memories had exhibited less scalability than what they were envisioned to be [43]. It is known that the initial MRAM technology has been substituted with a more encouraging STT-MRAM [44] technology. Moreover, extensive research work has managed to delay its imminent downfall and successfully reduced the technology nodes.

Nonetheless, there are persistent concerns about ultra-scaled Flash devices that are driven by consumer-oriented devices such as mobile phones and tablets. These concerns evolve around the rapid increase in storage capacity of the Flash memories available in the market. Such favorable circumstances gave PCM another chance for revival. PCM took the opportunity by demonstrating higher endurance, and switching speeds that is comparable to DRAM [45]. Subsequently, even CMOS-compatible integration [46] and scalability to future technology nodes [47] is investigated. An important advancement is the invention for multi-level cell (MLC) architecture [48].

In spite of the numerous improvements PCM has made, there exist critical obstacles hindering the success of PCM in the NVM market. The requirement of

high temperature forces the transistor in the memory cell to be used as a connection agent to provide a large enough current for the writing operation. This influenced many designs to be constructed around sub-lithographic characteristic. Subsequently, the need to provide such microscopic features with high yield results in precise fabrication procedure that is challenging to perfect. MLC devices are impaired by resistance drift over time, in the amorphous phase. The retention issue evident in PCM is contributed by the premature failure of the unfortunate faulty cells. The cycling endurance is affected by gradual separation of the constituent elements, whereby some cells will be in “stuck-reset” while the rest of the faulty cells are deemed to be in the “stuck-set” phase.

Continual work has to be done to successfully maneuver through these hurdles, permitting PCM technology to reach its potential and stride into products in the market today. With the capability of PCM, one can assume that PCM can succeed in the long run. A failure in PCM would boil down to the fabrication cost, in other words, the processes have been convinced to be too complex to be implemented and consumers would be unwilling to pay a hefty amount for a superior characteristic of PCM. However, if researchers are able to address all of the concerns related to MLC devices, it would poise PCM in an advantageous position to compete directly with Flash.

CHAPTER 3

BINARY WRITE-ONCE MEMORY (WOM) CODES

The binary write-once memory (WOM) code is proposed to ameliorate the endurance issues faced by the PCM. Figure 2.2 illustrates that the prerequisite for a reset operation is to have the bottom electrode supplying a large electrical signal while a set operation only requires a moderate electrical signal. Applying a large electrical signal consistently over time to the GST material would catalyze the heating element to cause it to wear out. The ultimate aim is therefore to maximize the number of set operations and minimize the number of reset operations in the lifetime of a PCM cell. To investigate this problem, the PCM is modeled as a WOM. The objective is to maximize the number of times information can be rewritten exclusively with set operations. In WOM, each cell has two states – if the GST material is in its amorphous phase, the cell is in state 0; if it is in the crystalline phase, and the cell is in state 1.

The outline of this chapter is briefly explained as follows. The primeval WOM-codes available in literature are first introduced in Section 3.1. Section 3.2

explains in detail the basic structure of a single level WOM and a formal definition is subsequently given. Thereafter, a critical criterion is proposed to design high-rates binary WOM-codes in Section 3.3. Section 3.4 presents the newly designed two-write WOM-codes that improve upon the best known restricted and unrestricted codes. The chapter is concluded with the WOM-rate analysis in Section 3.5.

3.1 The State-of-the-art WOM-codes

Rivest and Shamir founded the field of WOM-codes publishing their findings in the paper, “How to reuse a write-once memory” in 1982 [19], proving that a WOM could be efficiently rewritten to a surprising degree. From [19], wits are defined as WOM cells with two states; particularly bit ‘0’ and ‘1’. Examples of WOM back then were punch cards and digital optical disk. As one of the well known examples, it was shown that with three wits, it is possible to alter any two bit information vector to a different data bits. This groundbreaking presentation had been extensively cited by many and incited a high number of research work to efficiently maximize the number of rewrites on WOM.

In 1986, Gerard D. Cohen, Philippe Godlewski and Frans Merks [20] developed an application of the linear binary error correcting codes to the WOM codes defined earlier by Rivest and Shamir. The code construction presented was motivated by coset decoding. Zemor and Cohen later derived a construction method for error-correcting WOM codes [21]. F. Fu and A. J. Han Vinck theoretically determined the capacity region with zero error and the maximum number of information bits that can be stored in a WOM with a fixed number of successive

rewrites [22].

Here is a simple illustration enables a single rewriting of two information bits in three memory cells. The example shows that during the first write, a 2-bit information vector is encoded into a 3-bit codeword. Each of these three bits in the codeword represents the three memory cells being used. It is straightforward that if the second write 2-bit information word is different from the first write, the second write 3-bit codeword does not decrease any bit 1 into a bit 0. This code ensures that the WOM cell constraint is not violated. The data bits and their corresponding codewords are tabulated in Table 3.1 shown below.

Table 3.1: Data bits and the corresponding codewords of a two-write WOM-code [19].

Information bits	First write	Second write
00	000	111
10	100	011
01	010	101
11	001	110

The most essential obstacle faced by the WOM-code is to maximize the total number of distinct messages that can be written into n WOM cells in t writes, while conserving the constraint that no cell states can be reduced (*e.g.* cell states can only change from zero state to ones state). Rivest and Shamir first showed that WOM can be efficiently rewritten to a surprising degree [19], by storing two bits on three cells with two rewriting cycles as described by Table 3.1. This achievement had

been extensively cited by many and incited a high number of research work to efficiently maximize the number of rewrites on WOM. In the same work, Rivest and Shamir also described more code constructions for WOM-code, which consist of tabular WOM-codes and linear WOM-codes. Additionally, Merkle designed WOM-codes based on projective geometry [12]. In [13], Cohen et al. presented a coding technique motivated by coset decoding to construct WOM-codes from linear binary codes. Further work was done by Godlewski [15] to improve one of the constructions presented in [13]. A. Fiat and A. Shamir [27] considered the generalized WOM, in which the basic storage unit state transitions are described by a directed acyclic graph. Zemor and Cohen [39] derived a construction method for error-correcting WOM codes. F. Fu and A. J. Han Vinck [22] determined the capacity region from an information theory point of view. They studied the capacity region with zero error and the maximum number of information bits that can be stored in a WOM with t -successive rewriting cycles for the scenario where encoder knows the previous state of the memory while the decoder has no such information. Issues related to the WOM-codes were also addressed from its information-theoretic approach in [36]. In 2007, the connection between WOM-codes and Flash memories was first revealed [30,81]. The generalization of WOM-codes, including floating codes, buffer codes, and error-correcting floating codes were studied. There have been numerous papers on WOM-codes published subsequently [23,25,35,36].

3.2 Basic Structure of a Single-level WOM

The definitions and notations that are used throughout this chapter are introduced in the following paragraphs. Each WOM element is defined as a cell that can either be a zero or a one state. The initial states of all the cells in a WOM device are assumed to be set to zero. Every WOM memory cell is subjected to the constraint whereby they can only be switched from a zero to one state, irreversibly. The cell state vectors are binary vectors defined by the set $\{0,1\}^n$. For any $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$, vectors $\mathbf{x} \leq \mathbf{y}$ if and only if $x_i \leq y_i$ for all $1 \leq i \leq n$. In other words, vector $\mathbf{y}$ covers $\mathbf{x}$.

Definition 3.2.1 An $[n, M_1, M_2]_q$ two-write q -ary WOM-code C is a coding scheme that stores each information word in n memory cells. The encoding and decoding rules for the first and second write are denoted by $\mathcal{E}_i$ and $\mathcal{D}_i$, where $i=1, 2$.

1. $\mathcal{E}_1: \{1, 2, \dots, M_1\} \rightarrow \{0, 1, \dots, q-1\}^n$
2. For the second write, let the first write cell state be $\mathbf{s} = (s_1, \dots, s_n)$ and l be the second write information word. The second write encoding rule should satisfy $\mathcal{E}_2(l, \mathbf{s}) \geq \mathbf{s}$, where given two vectors $\mathbf{x} = (x_1, \dots, x_n)$ and $\mathbf{y} = (y_1, \dots, y_n)$, we say $\mathbf{x} < \mathbf{y}$ if for all $1 \leq j \leq n, x_j \leq y_j$.
3. For decoding,

$$\mathcal{D}_i: \{0, 1, \dots, q-1\}^n \rightarrow \{1, 2, \dots, M_i\}$$

such that $\mathcal{D}_1(\mathcal{E}_1(l)) = l$ and $\mathcal{D}_2(\mathcal{E}_2(l, \mathbf{s})) = l$, where $\mathbf{s}$ is the first write cell state and l is the second write information word.

Figure 3.1: Block diagram of the i -th write encoder $\mathcal{E}_i$ and decoder $\mathcal{D}_i$.

The input and output parameters of the encoding and decoding pairs are illustrated in Figure 3.1. Let $u_i \in \{0,1\}^k$ and $v_i \in \{0,1\}^n$ be the information vector and the cell state vector of the i -th write respectively. Before an encoding operation, a read operation is imperative to acquire the current cell state vector. With both the current cell state vector and new information vector, a new cell state vector is updated. According to the WOM constraint, cell levels only increase irreversibly, so the encoding function for the i -th write, where $2 \leq i \leq t$, takes the current value of the cell state vector v_{i-1} together with the new information vector u_i and returns a updated cell state vector v_i . In the case of the first write, the current state vector is the zero row vectors.

Let M_i represent the number of distinct messages that can be stored on the i -th write. Given n WOM cells can be rewritten in a total of t writes, then this WOM-code is known as a t -write WOM-code C_t with a WOM-rate of

$$\mathcal{R}(C_t) = \frac{\log_2(\prod_{i=1}^t M_i)}{n}. \quad (3.1)$$

Conventionally, the code rate of a code C of length n is defined to be $\frac{\log_2 |C|}{n}$, where $|C|$ is the

size of code C . In other words, the WOM-rate of a two-write WOM code is the sum of the code rates for the first and second write.

Let the write number during every write be known to the encoder and decoder. This additional information does not influence the WOM-rate. First assume that there exists an t -write WOM-code C and the write number is known to the encoder. The WOM-rate for the specified WOM-code is assumed to be defined by (3.1). There exists another WOM-code $[Nn + t, t; M_1^N, \dots, M_t^N]$ WOM-code $\hat{C}$ that is constructed by combining X blocks of C and t more cells that are used to represent the write number. The WOM-rate of $\hat{C}$ is

$$\begin{aligned}
 \mathcal{R}(\hat{C}) &= \frac{\log_2(\prod M_i^X)}{nX + t} \\
 &= \frac{X \cdot \log_2(\prod M_i)}{nX + t} \\
 &= \frac{\log_2(\prod M_i)}{n} \cdot \frac{nX}{nX + t} \\
 &= \mathcal{R}(C) \cdot \frac{1}{1 + t/nX}
 \end{aligned} \tag{3.2}$$

As $X \rightarrow \infty$, the WOM-rate of $\hat{C}$ converges to the WOM-rate of C . It is assumed that the write number is known to all encoders.

Since the main focus on the work here is targeted for storage purpose in NVM devices such as the Flash memory and PCM, our emphasis is placed on the construction of the two-write binary WOM-codes and show the improvement on the best WOM-codes that were previously known. The reason is because $t > 2$ would result in larger code rate lost [34-36], which directly leads to a larger decrease of the memory storage efficiency. There are two disjointed sets of WOM-codes – a

restricted WOM-code is a code with the same number of messages that can be written for both the first and the second write, namely $M_1 = M_2$, and the unrestricted WOM-code, namely M_1 and M_2 do not have to be equal.

3.3 Two-write WOM-codes

In this section, the theories for the construction of the two-write WOM-codes from binary linear codes are presented [23]. A criterion is subsequently proposed to select the binary linear codes such that the corresponding WOM-code constructed has a high WOM-rate. The construction is motivated by the coset coding used in conventional linear block codes. Some WOM-codes have coset coding scheme used on both first and second write operation [13, 15], while others adopts coset coding only on the second write [41]. The code construction presented in Section 3.3.1 demonstrates the construction of a two-write WOM-code from any linear code [23]. Similar to [41], the coset coding scheme is implemented on the second write of the construction presented below.

3.3.1 A Two-write WOM-code Construction

The preliminaries for the code construction of any two-write binary WOM-codes are given prior to the construction itself. Let $C[n, k]$ be a linear code with parity check matrix H . A set ϕ is defined as

$$\phi = \{c \in \{0,1\}^n \mid \text{rank}(H_c) = n - k\}. \quad (3.3)$$

For every codeword $c \in \{0,1\}^n$, a matrix H_c is constructed from the parity check

matrix of the linear code H . Let C be a linear code with parity matrix H . For every codeword $\mathbf{c} = (c_1, \dots, c_n) \in \{0,1\}^n$, a matrix H_c can be constructed from H . The i -th column of H_c , $1 \leq i \leq n$, is the i -th column of H if $c_i = 0$. Otherwise, the i -th column of H_c is the all-zero column. Finally, matrix H_c is updated by removing all the null columns.

Theorem 3.3.1. *Given a matrix M , with dimension m by n , the row rank of M is equal to the column rank of M .*

Proof. Let M be a $m \times n$ matrix with column rank of matrix M , column $\text{rank}(M) = s$. Then there exists s column vectors, $c_1, c_2, \dots, c_s$ that forms the basis of the column space of M , where

$$c_i = \begin{bmatrix} c_{1i} \\ c_{2i} \\ \vdots \\ c_{mi} \end{bmatrix}, \forall 1 \leq i \leq s. \quad (3.4)$$

The s column vectors $c_1, c_2, \dots, c_s$ forms an $m \times s$ matrix

$$\begin{aligned} C &= \begin{bmatrix} c_{11} & c_{12} & c_{13} & \cdots & c_{1s} \\ c_{21} & c_{22} & c_{23} & \cdots & c_{2s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & c_{m3} & \cdots & c_{ms} \end{bmatrix} \\ &= [\mathbf{c}_1 \quad \mathbf{c}_2 \quad \mathbf{c}_3 \quad \cdots \quad \mathbf{c}_s] \end{aligned} \quad (3.5)$$

Then by matrix multiplication, there exists a $s \times n$ matrix R , such that $M = C \cdot R$.

$$\begin{aligned}
 CR &= \begin{bmatrix} r_{11} & r_{12} & r_{13} & \cdots & r_{1n} \\ r_{21} & r_{22} & r_{23} & \cdots & r_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{s1} & r_{s2} & r_{s3} & \cdots & r_{sn} \end{bmatrix} \\
 &= [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3 \quad \cdots \quad \mathbf{r}_s]^T
 \end{aligned} \tag{3.6}$$

where $\mathbf{r}_i$ is the i -th row of the matrix R , for $1 \leq i \leq s$. Therefore, matrix M is represented as

$$\begin{aligned}
 M &= C \cdot R \\
 &= \begin{bmatrix} c_{11}r_{11} + \cdots + c_{1s}r_{s1} & \cdots & c_{11}r_{1n} + \cdots + c_{1s}r_{sn} \\ c_{21}r_{11} + \cdots + c_{2s}r_{s1} & \cdots & c_{21}r_{1n} + \cdots + c_{2s}r_{sn} \\ \vdots & & \vdots \\ c_{m1}r_{11} + \cdots + c_{ms}r_{s1} & \cdots & c_{m1}r_{1n} + \cdots + c_{ms}r_{sn} \end{bmatrix} \\
 &= [r_{11}\mathbf{c}_1 + \cdots + r_{s1}\mathbf{c}_s \quad \cdots \quad r_{1n}\mathbf{c}_1 + \cdots + r_{sn}\mathbf{c}_s]
 \end{aligned} \tag{3.7}$$

The (i, j) -th element of R is the coefficient of the column vector c_i , shown above, when the j -th column of matrix M is expressed as a linear combination of the s columns of C . Also, matrix M can be rewritten in the following form.

$$M = \begin{bmatrix} c_{11}\mathbf{r}_1 + c_{12}\mathbf{r}_2 \cdots + c_{1s}\mathbf{r}_s \\ c_{21}\mathbf{r}_1 + c_{22}\mathbf{r}_2 \cdots + c_{2s}\mathbf{r}_s \\ \vdots \\ c_{m1}\mathbf{r}_1 + c_{m2}\mathbf{r}_2 \cdots + c_{ms}\mathbf{r}_s \end{bmatrix} \tag{3.8}$$

Then the (i, j) -th element of C is the coefficient of r_i when the i -th row of M is expressed as a linear combination of the s rows of R . Therefore, the s rows of R span the rows of M . Then we have,

$$\begin{aligned}
\text{row rank}(M) &\leq \text{row rank}(R) \\
&= r \\
&= \text{column rank}(M).
\end{aligned} \tag{3.9}$$

For any $m \times n$ matrix M , the $\text{row rank}(M) \leq \text{column rank}(M)$. In order to prove that $\text{row rank}(M) = \text{column rank}(M)$, the inverse inequality, $\text{row rank}(M) \geq \text{column rank}(M)$ has to be satisfied.

$$\begin{aligned}
\text{column rank}(M) &= \text{row rank}(M^T) \\
&\leq \text{column rank}(M^T) \\
&= \text{row rank}(M)
\end{aligned} \tag{3.10}$$

Therefore, $\text{row rank}(M) = \text{column rank}(M)$.

□

Theorem 3.3.2. *Let C be a $[n, k]$ binary linear code and let H be a parity check matrix for C . For any $\mathbf{c} \in C$, if $\mathbf{c} \in \phi$, its weight is at most k .*

Proof. A generator matrix of any linear code is defined as a matrix whose rows form a basis of the linear code. For the linear code C , the parity check matrix H is also the generator matrix of the dual code $C^\perp$. By definition of a generator matrix, there are $n - k$ rows of H that are linearly independent. Hence, the $\text{row rank}(H) = n - k$. For any linear binary code, the $\text{column rank}(H)$ is equal to the $\text{row rank}(H)$. Therefore, $n - k$ columns of H are also linear independent. We then assume that

there exists a vector $\mathbf{c}$ with weight more than k , represented as $wt(\mathbf{c}) > k$. From (4), matrix H_c is obtained by removing at least $k + 1$ columns from H , or at most $n - k - 1$ columns of H are not removed. Since there are $n - k - 1$ columns of H that are linearly independent, the $\text{rank}(H_c) \leq n - k - 1 < n - k$. A contradiction is formed as $\text{rank}(H_c)$ must be equal to $n - k$ defined in (3). Our assumption that there exists a vector $\mathbf{c}$ with weight more than k is hence invalid. Therefore, we conclude that the weight of $\mathbf{c}$ must be less than or equal to k .

□

The above theorem shows that for any $\mathbf{c} \in \phi$, the rank of H_c must be at least $n - k$. However, for the code construction proposed in [23] the rank of H_c is strictly $n - k$.

Definition 3.3.1. *The support of a binary vector γ , denoted by $\text{supp}(\gamma)$, is the set $\{i \mid \gamma_i = 1\}$. Then a binary vector γ is said to be covering vector ϕ if and only if $\text{supp}(\phi) \subseteq \text{supp}(\gamma)$.*

Example 3.3.1. Let three binary vectors $\gamma, \phi_1, \phi_2 \in \{0,1\}^4$ be

$$\begin{aligned}\gamma &= 1001 \\ \phi_1 &= 1011. \\ \phi_2 &= 0001\end{aligned}\tag{3.11}$$

It is evaluated that the support for each binary vector is then

$$\begin{aligned}
\text{supp}(\gamma) &= \{1,4\} \\
\text{supp}(\varphi_1) &= \{1,3,4\}. \\
\text{supp}(\varphi_2) &= \{4\}
\end{aligned} \tag{3.12}$$

From the above illustration, the binary vector φ_1 is said to cover vector γ , while the binary vector φ_2 does not.

Lemma 3.3.3. Let C be an $[n, k]$ linear code over $\{0,1\}^n$ and let H be a parity check matrix for C . For every vector $c \in \{0,1\}^n$, $\text{rank}(H_c) = n - k$ if and only if vector c does not cover any non-zero codeword in $C^\perp$.

The proof of Lemma 3.3.3 is provided in [20]. However, the intuitive idea of having every vector $c \in \phi$ not covering any non-zero codeword in $C^\perp$ is explained as follows.

Assume $x \in \phi$ which covers a non-zero codeword $y \in C^\perp$, for example $\text{supp}(y) \subseteq \text{supp}(x)$, and that the cell state for the first and second write is x, y , respectively. Let

$$I = \text{supp}(x) = \{i \mid x_i = 1\} \tag{3.13}$$

where x_i is the i -th entry of vector x . Consider the case when $\text{supp}(y) = \text{supp}(x)$, in other words, the cell state of the first write is exactly the second write. In this scenario the prior information that was being stored in the WOM memory cells is unchanged. Therefore, a rewriting operation is not required to update vector x to

vector $\mathbf{y}$.

Next, we consider the case when $\text{supp}(\mathbf{y}) \subset \text{supp}(\mathbf{x})$. It is obvious that there exists at least one bit position in the first write that has been converted to state 1, but the particular bit position is state 0 for the second write. Due to the constraints of a WOM code, it is impossible to change any bit position from state 1 to state 0. As a result, vector $\mathbf{x}$ cannot be rewritten into vector $\mathbf{y}$. Based on the assumption that there exists a vector $\mathbf{x} \in \phi$ which covers a non-zero codeword $\mathbf{y} \in C^\perp$ (e.g. $\text{supp}(\mathbf{y}) \subseteq \text{supp}(\mathbf{x})$), it is not possible to rewrite the vector $\mathbf{x}$ into $\mathbf{y}$. From the argument above, every vector $\mathbf{x} \in \phi$ does not cover any non-zero codeword $\mathbf{y} \in C^\perp$.

The next theorem states the code construction method to design a two-write WOM-code from a linear block code.

Theorem 3.3.4. *Let C be an $[n, k]$ linear code over $\{0, 1\}^n$, H be a parity check matrix for C and let ϕ be the set defined in (3.3). Then there exists a $C_2[n, 2: \phi, 2^{n-k}]_2$ two-write WOM-code constructed from linear code C with WOM-rate*

$$\mathcal{R}(C_2) = \frac{\log_2(|\phi|) + (n - k)}{n} \quad (3.14)$$

In order to prove that a two-write WOM-code exists for this construction defined by Theorem 3.3.4, the encoding and decoding rules for both writes must be present. The encoding and decoding operations for the two-write WOM-code are illustrated

in Example 3.3.2.

Example 3.3.2. Assume we have a linear code C with parity check matrix H , given by

$$H = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}. \quad (3.15)$$

During the first write, we want to encode the information word $s_1 = 11$, and then in the second write we want to encode the information word $s_2 = 10$. Let c_1 and c_2 denote the codeword of the first and second write respectively. The encoding rule is motivated by syndrome decoding. For the first write, the encoding operation denoted by $\mathcal{E}_1(s_1) = c_1$, is defined as $c_1 \cdot H^T = s_1 = 11$, therefore $c_1 = 100$. The decoding of c_1 to achieve s_1 is trivial. During the second write, the reduced form of H is represented by H_{c_1} . We then obtain

$$H_{c_1} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \quad (3.16)$$

We defined the second write encoding operation by $\mathcal{E}_2(s_1 + s_2) = c_2$. For the second write, the information word s_2 is not stored as it is. Instead, the modulo-2 sum of s_1 and s_2 ,

$$s_1 + s_2 = 11 + 10 = 01 \pmod{2}. \quad (3.17)$$

is stored during the second write. This is to ensure that s_2 can be successfully decoded. This motivation will further be explained in the next paragraph. The codeword c_2 , representing the updated information word, is calculated by the

following procedures. First, we evaluate the reduced vector of c_2 , and is represented by the variable c_2' . From Equation (3.18) shown below,

$$c_2' \cdot H_{c_1}^T = 01 \quad (3.18)$$

it is trivial to observe that $c_2' = 10$. Next, we add “0”s back to the column entries of c_2' , which was removed earlier from H to derive H_{c_1} . In this example, the first column was removed from H to produce H_{c_1} . Therefore, a 0 is added to the first column of the codeword c_2' , leading to $c_2 = 010$. Therefore the codeword after the second write will be $c_1 + c_2 = 110$.

During the re-writing operation, the modulo-2 sum of s_1 (first write information word) and s_2 (second write information word) is being encoded. Given that the current WOM cell state after re-writing is $c_1 + c_2$, after multiplying the current WOM cell state to the transpose of the parity check matrix of the linear code H , we have

$$\begin{aligned} (c_1 + c_2) \cdot H^T &= c_1 \cdot H^T + c_2 \cdot H^T \\ &= s_1 + (s_1 + s_2) \\ &= s_2 \pmod{2}. \end{aligned} \quad (3.19)$$

From the above, we verify that the modulo-2 sum of s_1 and s_2 , during the second write encoding, is essential to correctly decode the second write information word s_2 .

3.3.2 Maximizing the WOM-rate

Storage memories aim to maximize the areal density, and thus a higher rate code is more desirable. Careful selection of the linear code C is crucial, so that the overall code rate can be maximized. From Section 3.3.1, we concluded that a two-write WOM-code can always be constructed from any linear code C . In this section, we will analyze the selection of a linear code C for the construction of a WOM-code C_2 , such that its corresponding WOM-rate $\mathcal{R}(C_2)$ is maximized.

For a given linear code $C[n, k]$ over $\{0,1\}^n$ with a parity check matrix H , the WOM-rate is maximized when $|\phi|$ is maximized. From (3.14), WOM-rate is proportional to the size of ϕ . In order to analyze how one can maximize the size of set ϕ , we model this as a maximization problem. The objective function is to maximize the size of ϕ , subjected to two constraints. As stated by Theorem 3.3.2, one of the constraints is that the weight of vector c must be less than or equal to k , for all $c \in \phi$. The second constraint is $c \in \phi$ if and only if the rank of H_c is equal to $n - k$, shown in equation (3.3). By Lemma 3.3.3, the above statement is equivalent to $c \in \phi$ if and only if vector c does not cover any non-zero codeword in $C^\perp$. To illustrate Lemma 3.3.3, we consider three scenarios, with the first case assuming

$$C^\perp = \{0100\}. \quad (3.20)$$

For this case, the length of the codeword $n = 4$ and $\text{supp}(0100) = \{2\}$. Every vector $c \in \phi$ cannot cover the vector 0100. Therefore, $\text{supp}(c)$ can take the value

of any entry in the set

$$\{\{1\}, \{3\}, \{4\}, \{1,3\}, \{1,4\}, \{3,4\}, \{1,3,4\}\}. \quad (3.21)$$

Following that, it is clear that $|\phi| = 7$.

Next, assume if the dual code is

$$C^\perp = \{0101\}. \quad (3.22)$$

We have $\text{supp}(0101) = \{2,4\}$, and can at most be the set

$$\{\{1\}, \{2\}, \{3\}, \{4\}, \{2,4\}, \{1,3\}, \{1,4\}, \{3,4\}, \{1,2,3\}, \{1,3,4\}\}. \quad (3.23)$$

implying that $|\phi| = 11$. This comparison shows that size of ϕ increases with increasing weight of c . Lastly if the dual code contains the codewords from the previous two scenarios,

$$C^\perp = \{0100, 0101\}. \quad (3.24)$$

The support of the dual code becomes $\text{supp}(C^\perp) = \{\{2\}, \{2,4\}\}$. From the previous comparisons, it is shown that $\text{supp}(0100) \subset \text{supp}(0101)$. From the analogy above, if there exists a support of a vector x that is a subset of another vector y , it is only necessary to consider the vector x when computing the size of ϕ .

It is proven that for any linear code C , the minimum Hamming weight of C is equal to the minimum Hamming distance between the codewords of C . [37, 38]. Therefore, in order for $|\phi|$ to be maximized the Hamming weight of $C^\perp$ should be maximized. Since $C^\perp$ is a linear code, then maximizing the Hamming weight of $C^\perp$ is equivalent

to maximizing the minimum Hamming distance of $C^\perp$. The error correcting capability of a code is directly proportional to the minimum distance $d_{\min}$, of the linear code; consequently, the BCH codes have high error correcting capabilities due to their large minimum distances. For our simulations, the binary BCH codes are used as the dual code of the linear code C . For binary BCH codes with a fixed length n , there exist multiple BCH codes with different dimensions and minimum Hamming distance. Letting the dual code be a particular BCH code $C^\perp[n, n - k]$, the linear code can easily be fulfilled by taking the generator matrix of C as the parity check matrix of C , and the parity check matrix of $C^\perp$ as the generator matrix of $C^\perp$. Thus, C has length n and dimension k . Let the base code be the dual code $C^\perp[n, n - k]$ that is a binary BCH code, such that WOM-rate is maximized for a fixed n . For a fixed length n , we need to find a dual code that maximizes the WOM-rate. The following criterion is proposed to select the optimal binary BCH code.

Criterion 3.3.1. For a fixed n , the dual code is initialized to be the binary BCH code with the minimum k , in order to maximize the minimum Hamming distance. Next k is increased iteratively until the WOM-rate for the two-write WOM-code is maximized.

We remark that in general, the above criterion proposed for the binary BCH code applies to other linear block codes as well.

The following example is to illustrate how the base code is selected, when we fixed the length of the code to be $n = 31$.

Example 3.3.3. Let C_i be a $[n, k_i, d]$ linear code and the dual code be $C_i^\perp[n, n - k_i, d]$, where $i = 1, 2, 3$. In the table below, we list the dimensions of the binary BCH codes of length, $n = 31$. The WOM-rate $\mathcal{R}(C_2)$ is calculated by (3.14). From Table 3.2, it is clear that the dual code with the largest Hamming distance has the largest value of $|\phi|$, which is consistent with our analysis of having a large hamming distance as the base code to maximize the size of set ϕ . For a fixed length $n = 31$, the BCH code with the largest Hamming distance is the code with the smallest value of $n - k$, which is proportional to the number of codewords available for the second write M_2 . Therefore, even when the size of $|\phi|$ increases, it does not necessarily imply that the WOM-rate will increase. Hence, the BCH code that is to be used in the proposed code construction has to be carefully chosen so that the WOM-rate is maximized. In this case when the length of the code is $n = 31$, the maximum WOM-rate is achieved, if $n - k = 21$. Therefore, the base code for $n = 31$ is the BCH code $[31, 21, 2]$.

Table 3.2: Binary BCH codes parameters and their corresponding WOM-rates.

$C_i^\perp$	n	$n - k_i$	d_{min}	$ \phi $	$\mathcal{R}(C_2^{(i)})$
$C_1^\perp$	31	26	1	206368	1.4059
$C_2^\perp$	31	21	2	43721409	1.4962
$C_3^\perp$	31	16	3	539069042	1.4518

3.4 Newly Designed Two-write WOM-codes

We define a restricted WOM-code as a code with the same number of messages that can be written for both the first and the second write, namely $M_1 = M_2$, and the unrestricted WOM-code, namely M_1 and M_2 do not need to be equal. It is proven in Section 3.3.4 that in order to achieve a higher WOM-rate, we need to maximize ϕ by selecting the dual of the linear code with large Hamming distance. Therefore, we ran our simulations using the binary BCH codes as the dual code. The results of the best two-write binary WOM-codes are introduced in the subsequent sections.

3.4.1 Restricted WOM-codes

For restricted WOM-codes, since there is an additional constraint that the total number of possible messages in the first and second write must be the same. Equation (3.14) is reduced to

$$\mathcal{R}(C) = \frac{2 \cdot \hat{R}}{n} \quad (3.25)$$

where $\hat{R}$ is the minimum value between $\log_2(|\phi|)$ and $(n - k)$.

From literature it is exhibited that the best known WOM-rates can be improved when WOM-codes are constructed with the construction presented in Section 3.3.1 [23]. The best known restricted WOM-rate was increased from 1.343 to 1.4546 by performing a computer search and 1.375 by taking the linear code as

the first order Reed-Muller code $C[16, 5, 8]$. However consideration must be taken for WOM-codes achieved by computer search, as these codes do not have a practical code construction. This might lead to expensive and complicated encoding and decoding operations.

The work presented in this section are based on selecting different binary BCH codes – which have a practical code construction, to construct various two-write binary WOM-codes. In our simulation, the $[15, 11]$ -BCH code is used as the dual code $C^\perp$ and the WOM-rate is calculated. From Theorem 3.3.2, the set $|\phi|$ for this BCH code consists of vectors c with weight at most the dimension of the linear code. Since the dual code has dimension 11, the dimension of the linear code is 4. The breakdown of the total number of vectors for each hamming weight is tabulated in Table 3.3. From the simulation results, we compute the size of set ϕ to be 1381. The WOM-rate is calculated with (3.14), and is therefore 1.3909. This is the best WOM-rate for the restricted WOM-code with two writes.

Table 3.3: Breakdown of number of vectors for WOM-code with $[15, 11]$ -BCH code being used as the base code.

Hamming weight	Number of vectors with weight
0	1
1	15
2	105
3	420
4	840

3.4.2 Unrestricted WOM-codes

The best WOM-rate for unrestricted two-write WOM-code is discussed in this section. From [23], it was shown that the WOM-rate for unrestricted WOM-code can be increased from 1.3707 to 1.4928 by computer search, and 1.4632 by using the [23,12,7]-Golay code. In another simulation set, the [31,21]-BCH code is used as the dual code $C^\perp$. This suggests that the linear code has length and dimension, $n = 31$ and $k = 10$, respectively. The set ϕ consists of vectors with at most weight 10. The breakdown of the total number of vectors for each hamming weight is tabulated in Table 3.4. From the simulation results, we compute the size of set ϕ to be 43721409. The WOM-rate is calculated with (3.14), and is therefore 1.4962. When the [31,21]-BCH code is selected as the dual code, the WOM-rate is higher than the code rate achieve by computer search as well as the best constructed two-write WOM-code, presented in [23].

Table 3.4: Breakdown of number of vectors for WOM-code with [31, 21]-BCH code being used as the base code.

Hamming weight	Number of vectors with weight
0	1
1	31
2	465
3	4495
4	31465
5	169725
6	730639

7	2546340
8	7097760
9	14819240
10	18321284

However, there exists a trade-off between increasing the minimum Hamming distance of the BCH code and the WOM-rate. It is known that for BCH codes with a fixed length n , BCH codes with larger k have smaller Hamming distance. Hence, for BCH code of a given n with larger k , the size of is smaller but the size of messages for second write is larger. It is emphasized that in order to maximize the WOM-rate, the BCH code must be carefully selected.

3.5 WOM-rate Analysis

To measure the improvement in WOM-rate attained in this current work, and evaluate its merits to what is available in literature, we define a measure known as code efficiency and state the capacity for the two-write WOM-code. The capacity for any binary WOM with two writes was proven in [22] and [24] to be

$$\Gamma_2 = \{(R_1, R_2) \mid \exists p \in [0,0.5], R_1 \leq h(p), R_2 \leq 1 - p\}.$$
 (3.26)

The WOM-rate for a two-write binary WOM-code is maximized for $p = \frac{1}{3}$ and $\Gamma_2 = \log_2 3 \approx 1.585$.

Definition 3.5.1. The code efficiency of a binary WOM-code C_2 , denoted by η_2 , is

defined as

$$\eta_2 = \frac{\mathcal{R}(C_2)}{\Gamma_2}, \tag{3.27}$$

where $\mathcal{R}(C_2)$ is the WOM-rate and Γ_2 is defined in (3.26).

The comparison of the WOM-codes designed by this work with those in literature is tabulated in Table 3.5 and Table 3.6. For the unrestricted WOM-code, the WOM-rate is increased from 1.4632 to 1.4962, which leads to an approximate increase of 2% in code efficiency. As for the restricted WOM-code, the WOM-rate is increased from 1.3750 to 1.3909 that is approximately 1% increase in code efficiency. We remark that the maximum restricted WOM-rate to-date is 1.4546 [23]. However, this is achieved from computer search, without having a practical code (e.g. encoder and decoder) design.

Table 3.5: WOM-rate and coding efficiency comparison for unrestricted WOM-codes.

	Before [23]	Prior-art [23]	Work done
$\mathcal{R}(C)$	1.3707	1.4632	1.4962
η	86.48%	92.32%	94.40%

Table 3.6: WOM-rate and coding efficiency comparison for restricted WOM-codes.

	Before [23]	Prior-art [23]	Work done
$\mathcal{R}(C)$	1.3430	1.3750	1.3909
η	84.73%	86.75%	87.76%

3.6 Conclusion

A high emphasis is placed on WOM-codes for PCM as they allow the increment on the lifetime of the memory cells. Since the research here is targeted for storage purpose, the attention is placed on two-write WOM-codes. This is because higher re-writability will greatly penalize both the WOM-rate and the memory storage efficiency. The code construction [23] is introduced and investigation work is done on the selection procedure for the best dual code to maximize the size of ϕ . Further research has shown that in order to maximize the size of ϕ , the minimum Hamming distance of the dual code should be maximized. In Section 3.3.2, the BCH code is proposed to be used as the base code, considering that these codes have large Hamming distances. However, there is a trade-off between increasing the minimum Hamming distance and the number of messages available for the second write operation. Additional analysis revealed that for a fixed n , in order to maximize the overall WOM-rate, the search should be initialized with a BCH code with the smallest k , such that the hamming distance is maximized. The parameter k is iteratively increased till the WOM-rate is maximized. Using the proposed code

design criterion, we managed to obtain efficient two-write WOM-codes, whose rates are higher than those available in literature, for both the unrestricted and restricted WOM-codes. In the context of NVM, our interest lies in the restricted WOM-codes, for a simple reason that it is more practical for both writes to have the same number of messages.

CHAPTER 4

NON-BINARY WOM-CODES FOR ENDURANCE

Spurred by the interest to improve storage capacity, development in NVM technologies has brought about multilevel cells (MLC), which are able to store more than a single bit per cell [35, 69]. Recently, a generalized WOM model for multilevel cells and information theory limits were examined [27]. If WOM-codes are to be implemented as a type of storage code, emphasis has to be placed on maximizing the number of bits that can be written to a given n cells in two writes. The constraint that no cell states can be decreased during each write must not be violated as well. Similar to the binary WOM-codes, the non-binary WOM-codes designed in this thesis contains two writes. The non-binary WOM-codes are thoroughly examined in this chapter as follows.

In Section 4.1 the basic model of a MLC WOM cell is explained together with definitions and notations used. Section 4.2 examines two code construction methods that are used to design non-binary WOM-codes. In Section 4.3, both the code constructions are compared. To further the research on the non-binary WOM-

codes, novel high-rate WOM-codes of higher alphabet size are designed, based on the efficient binary WOM-codes developed in Chapter 3. The chapter is finally concluded in Section 4.4.

4.1 Basic Structure of a Multi-level WOM

A basic structure of a MLC-WOM cell is that the cells have q distinct states: $\{0, 1, \dots, q-1\}$, with initial state of all cells to be zero. The unique property of WOM cells is that none of the cells are able to decrease their cell state. The definition of a two-write binary WOM-code is extended for the non-binary codes.

Definition 4.1.1. Let M_i be the number of distinct messages that can be written on the i -th write. A $[n, 2: M_1, M_2]_q$ two-write non-binary WOM-code C_q is a coding scheme on n q -ary cells and two pairs of encoding and decoding rules. The encoding and decoding rules, denoted by $\mathcal{E}_i$ and $\mathcal{D}_i$, are defined as follows.

For each of the i -th write,

1. $\mathcal{E}_1: \{1, 2, \dots, M_1\} \rightarrow \{0, 1, \dots, q-1\}^n$
2. For the second write, let the first write cell state be represented by $\mathbf{s} = (s_1, \dots, s_n)$ and m be the second write information word. For $i = 1, 2$, the second write encoding rule should satisfy the following rule.

$$\mathcal{E}_2(m, \mathbf{s}) \geq \mathbf{s}$$

3. Let the first and second write information word be represented by $m_1 \in$

$\{1, 2, \dots, M_1\}$ and $m_2 \in \{1, 2, \dots, M_2\}$ respectively. The decoding rule for each write is defined by the mapping rule

$$\mathcal{D}_i: \{0, 1, \dots, q-1\}^n \rightarrow \{1, 2, \dots, M_i\}$$

such that $\mathcal{D}_1(\mathcal{E}_1(m_1)) = m_1$ and $\mathcal{D}_2(\mathcal{E}_2(m_2, s)) = m_2$.

The inputs and outputs of the two pairs of encoding and decoding rules are illustrated in Figure 4.1 and Figure 4.2, respectively. Let $u_1 \in \{1, \dots, M_1\}$ and $u_2 \in \{1, \dots, M_2\}$ represent the information vector of the first and second write. While $v_1, v_2 \in \{0, 1, \dots, q-1\}^n$ are the cell state vectors after the first and second write. For the first write encoding operation, the information vector u_1 is mapped to a unique cell state vector v_1 . While for the second write encoding operation, with the information of the current cell state vector after the first write provided, the information vector u_2 is mapped to a unique cell state vector v_2 . The decoding procedure for both writes maps the cell state back to the corresponding information vector.

Figure 4.1: Block diagram of the first write encoder and decoder.

Figure 4.2: Block diagram of the second write encoder and decoder.

Definition 4.1.2. *The WOM-rate of a two-write q -ary WOM-code C_q on the i -th write is defined to be*

$$\mathcal{R}_i(C_q) = \frac{\log_2 M_i}{\log_2 q^n} = \frac{\log_q M_i}{n}. \quad (4.1)$$

Therefore, the WOM-rate of C_q is the sum of the first and second WOM-rate

$$\mathcal{R}(C_q) = \mathcal{R}_1(C_q) + \mathcal{R}_2(C_q). \quad (4.2)$$

4.2 Code Construction for Non-binary WOM-codes

In this section, two code constructions for the two-write non-binary WOM-code are introduced. We first propose a code construction that is able to achieve WOM-rates considerably close to the best known WOM-codes with a smaller codeword length. The smaller codeword length would provide less complexity during encoding and decoding operations without sacrificing much storage capacity. A second construction presented in [71] has shown to outperformed other similar code constructions [96] on numerous numbers of occurrences. Furthermore, after conducting a search based on the critical criterion proposed in Chapter 3, we show that the newly designed WOM-codes achieve rates higher than the state-of-the-art codes.

4.2.1 Code Construction A

Generalizing from the code construction of the binary WOM-codes, assume first that each cell has q levels. Let $C[n, k]$ be a linear code with parity check matrix H . The set defined in (3.3) is redefined as

$$\phi_q = \{\mathbf{c} \in \{0, \dots, q-1\}^n \mid \hat{S}, \forall x_i \in \{0, \dots, q-1-c_i\}\}. \quad (4.3)$$

where c_i and h_i represents the i -th entry of the cell state vector and i -th column of the parity check matrix of a linear code, respectively. Let the non-empty set $V = \{0, \dots, q-1\}^{n-k}$, the set $\hat{S}$ is defined as

$$\hat{S} = \{\sum x_i h_i \text{ spans } V\}. \quad (4.4)$$

In the code construction for non-binary codes, the matrix H_c is calculated from the rule defined in (4.5).

$$H_c^{(i)} = \begin{cases} H^{(i)} & , \text{ if } c_i \neq q-1 \\ \text{replace } i\text{-th column with zero vector} & , \text{ if } c_i = q-1 \end{cases} \quad (4.5)$$

where c_i is the i -th entry of the vector $\mathbf{c}$.

Theorem 4.2.1. *Let $C[n, k]$ be a linear code with parity check matrix H and let ϕ_q be the set defined in (4.3). Then there exists an $[n, 2 : |\phi|, q^{n-k}]$ two-write q -ary WOM-code with WOM-rate*

$$\mathcal{R}(C_q) = \frac{\log_2 |\phi| + \log_2 q^{n-k}}{\log_2 q^n} \quad (4.6)$$

Proof. It is necessary to shown the existence of the encoding and decoding rules on both writes. Let $\{c_1, c_2, \dots, c_{|\phi_q|}\}$ be an ordering of set ϕ_q . The encoding rules of both writes are defined as follows.

During the first write, a symbol $x \in \{1, 2, \dots, |\phi_q|\}$ is written in the WOM cells. Let $\mathcal{E}_1, \mathcal{D}_1$ be the encoding and decoding rule for the first write. For every $x \in \{1, 2, \dots, |\phi_q|\}$, the encoding and decoding rule maps the symbol x to a cell state c_x and vice versa, for example $\mathcal{E}_1(x) = c_x$ and the inverse $\mathcal{D}_1(c_x) = c_x \cdot H^T = x$. On the second write, a different information vector y of $n - k$ bits is rewritten on WOM cells, without violating the WOM constraint. Let c_x be the cell state vector after the first write and $x = c_x \cdot H^T$. The second write encoding rule is defined as

$$\mathcal{E}_2(y, c_x) = (q - 1) \cdot c_x + c_y, \quad (4.7)$$

where $c_y \cdot H_{c_x}^T = (q - 1) \cdot x + y$ must be satisfied. For the decoding rule, if $\mathbf{c}'$ is the cell state after two writes, the decoded $n - k$ -bits information vector in the second write is

$$\begin{aligned} \mathcal{D}_2(\mathbf{c}') &= \mathbf{c}' \cdot H^T \\ &= (c_x + c_y) \times H^T \\ &= x + (q - 1) \times x + y \\ &= y \pmod{q}. \end{aligned} \quad (4.8)$$

The second write decoding rule is always preserved when every $x \in \phi_q$ results in $\text{rank}(H_x) = n - k$.

□

Example 4.2.1. Let C_3 be a two-write 3-ary WOM-code and C be a linear binary code with parity check matrix

$$H = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}. \quad (4.9)$$

To simplify the illustration, we assume the first write encoding map to be

$$\mathcal{E}_1(x) = c_1 \quad (4.10)$$

such that $c_1 \cdot H^T = x$ is satisfied. During the first write, the information vector $x = (20)$ is written in the WOM cells with corresponding cell state $c_1 = (200)$, derived by the first write encoding map. The process of decoding the first write the cell state vector c_1 to obtain the information word x is straightforward and is illustrated below.

$$D_2(c_1) = c_1 \cdot H^T = (20) \quad (4.11)$$

On the second write, the user rewrites another information vector $y = (10)$ on the WOM cells. The final cell state vector after the second write is calculated as shown below.

1. The parity check matrix is altered to

$$H_x = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}. \quad (4.12)$$

2. The equation $c_y \cdot H_{c_x}^T = (q-1) \cdot x + y$ must be satisfied, as mentioned in the proof of Theorem 4.2.1.

$$\begin{aligned}
c_y \cdot H_{c_x}^T &= (q - 1) \times x + y \\
&= 2(20) + (10) \\
&= 20 \pmod{3}
\end{aligned} \tag{4.13}$$

3. After the above manipulation, the cell state vector of the second write is $c_y = (012)$ and the cell state vector after both writes is $\mathbf{c}' = (212)$. The decoding rule for the second write is as follows.

$$\begin{aligned}
\mathcal{D}_2(\mathbf{c}') &= (212) \times H^T \\
&= (10) \pmod{3}
\end{aligned} \tag{4.14}$$

Since there were no requirements specified in Theorem 4.2.1 for the chosen linear code, the user is free to select any available linear codes. Since the WOM-codes discussed in this section are targeted for data storage applications, investigation is done on specifically WOM-codes with high WOM-rates, which is further explored in Section 4.3.

4.2.2 Code Construction B

Prior to describing the second code construction, we briefly outline the conventional encoding and decoding of decimal numbers, where each decimal digit is mapped to a unique vector $\mathbf{v} \in \{0, 1, \dots, q - 1\}^n$. The mapping rule is defined below.

$$\delta_{q,n}(\mathbf{v}) = \sum_{k=1}^n q^{n-k} v_k. \quad (4.15)$$

This mapping is easily verified to be one-to-one and that the inverse exists. The procedure of performing the encoding of decimal digit $\delta_{q,n}(\mathbf{v})$ to obtain the encoded vector $\mathbf{v} = (v_1, v_2, \dots, v_n)$ is

$$v_k = \left\lfloor \frac{\delta_{q,n}(\mathbf{v})}{q^{n-k}} \right\rfloor. \quad (4.16)$$

and after each computation of v_k , decimal digit $\delta_{q,n}(\mathbf{v})$ is updated to

$$\delta_{q,n}(\mathbf{v}) = \delta_{q,n}(\mathbf{v}) - v_k \times q^{n-k}. \quad (4.17)$$

On the other hand, the procedure of decoding the vector $\mathbf{v}$ to obtain its corresponding decimal digit is calculated from (4.15). It has been proven in [71] that for all $m, q \geq 2$, an $C_{q^m}[n, 2: M_1^m, M_2^m]_{q^m}$ WOM-code exists if and only if an $C_q[n, 2: M_1, M_2]_q$ WOM-code exists.

Definition 4.2.1. If $\mathcal{R}(C_q)$ is the WOM-rate of a WOM-code $C_q[n, 2: M_1, M_2]_q$, then the WOM-rate of higher alphabet size WOM-codes constructed from C_q , denoted as $C_{q^m}[n, 2: M_1^m, M_2^m]_{q^m}$ is

$$\mathcal{R}(C_{q^m}) = m \times \mathcal{R}(C_q). \quad (4.18)$$

We remark that the WOM-code which is used to constructed higher alphabet size WOM-codes is referred to as a base code in this chapter.

Furthermore, it has also been shown in [71] that maximizing the WOM-rate of the base code used in the non-binary WOM-code construction, would also maximize the WOM-rate for the higher alphabet size WOM-code. Therefore, we proposed to leverage on the highest rate binary WOM-code discovered in Section 3.4 to construct higher alphabet size WOM-codes (e.g. $q = 2^m$).

A drawback of the best known WOM-codes presented in [71] is that they lack of a practical code design, which implies that the codes are not physically available. The novelty of the work on the non-binary WOM-code is that the codes designed are able to achieve the highest coding efficiency for the parameters specified and also have a practical code construction. Table 4.1 provides the mapping of how information bits are mapped to their corresponding codewords for the first and second-write of the popular two-write binary WOM-code found by Rivest and Shamir, denoted as the $[3,2:4,4]_2$ WOM-code. In Example 4.2.2, the non-binary WOM-code constructed from the $[3,2:4,4]_2$ WOM-code is presented.

Table 4.1: Mapping of Rivest and Shamir $[3, 2: 4, 4]_2$ WOM-code.

Information word	Encoded word	
	First write	Second write
00	000	111
01	010	101
10	100	011
11	001	110

Example 4.2.2. Let Rivest and Shamir’s binary WOM-code be denoted as $C_2[3,2:4,4]_2$. It is proven in [71] that another 4-ary WOM-code can be constructed from C_2 and is denoted by $C_4[3,2:4^2,4^2]_{2^2}$. Both the first and second write operation encode four information bits. The first step of the encoding operation is done by using C_2 , where information vectors $(11,01)$ is mapped to codewords $(001,010)$. We label 001 and 010 as the first and second triplet respectively. The next step is to extract the first entry of each triplet and append them to obtain $(v_1, v_2, v_3) = (00,01,10)$. Finally using mapping $\delta_{q,n}(v_i)$, where $i = 1, 2, 3$, the encoded word for the first write is $(0, 1, 2)$. The second write codeword can be calculated with the same procedure and is summarized in the table below.

Table 4.2: First and second write mapping between information bits and codeword of C_4 .

Write number	Information word	Codeword
1	$(11,01)$	$(0, 1, 2)$
2	$(01,10)$	$(1, 2, 3)$

For the code C_4 , let the first write codeword be $c_1 = (c_{11}, c_{12}, c_{13})$ and the second write codeword be $c_2 = (c_{21}, c_{22}, c_{23})$. The constraint of the WOM is not violated as $c_{2k} \geq c_{1k}$ for all values of k . Therefore, the constructed non-binary code is a valid WOM-code. Based on Definition 4.2.1, the WOM-rate of the constructed 4-ary

WOM-code is $4 \times 2/3 = 4/3$. If an 8-ary WOM-code was constructed the WOM-rate would be $8 \times 2/3 = 16/3$ instead.

4.3 Code Construction Comparison

4.3.1 WOM-codes Designed from Construction A

The non-binary WOM-codes with the highest WOM-rate, designed by construction A, is presented in this section. Akin to the binary WOM-code, we design the non-binary WOM-codes with BCH codes as the base code. The best possible simulation is conducted using the BCH code with parameters $n = 7$, $k = 4$ and $q = 4$. Higher parameters would result in insufficient memory. The parity check matrix of the mentioned BCH code is given as follows.

$$H_1 = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}_{3 \times 7} \quad (4.19)$$

Next, we present another parity check matrix H_2 , which has a slightly bigger dimension as compared to H_1 .

$$H_2 = \begin{bmatrix} 1 & 0 & 1 & 3 & 2 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 3 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 3 & 2 & 1 \end{bmatrix}_{3 \times 8} \quad (4.20)$$

In order to evaluate its merits benchmarking with what is available on literature, we define a measure known as efficiency which is the ratio of its WOM-rate and the capacity. The capacity for any 4-ary WOM-code with two writes was proven in [22, 24] to be $\Gamma_4 = 3.3219$ and the efficiency of the WOM-code is calculated by

$$\eta_4 = \frac{\mathcal{R}(C_4)}{F_4}. \tag{4.21}$$

The WOM-rate and coding efficiency when both H_1 and H_2 are used as the parity check matrix of the base code is tabulated in Table 4.3.

Table 4.3: WOM-rate comparison benchmarked against 4-ary WOM-codes constructed with code constructions [71].

	Best known 4-ary WOM-codes	Achieved by H_1	Achieved by H_2
(n, k)	(33,22)	(8,3)	(31,21)
$\mathcal{R}(C_4)$	2.9856	2.3407	2.9572
η_4	89.88%	70.46%	89.02%

The 4-ary WOM-code that achieves WOM-rate of 2.9856 has parameters of $n = 33$ and $k = 22$ [23]. Although the WOM-rate achieved by H_2 has efficiency reduced by approximately 0.8%, it only requires parameters of $n = 8$ and $k = 4$. With a reduction of 75.7% in the codeword length n and 81.8% reduction in dimension k , the complexity at the encoder and decoder is greatly reduced. We conclude that by using construction A, a small efficiency can be sacrificed to design WOM-codes with a significant reduction in complexity at the encoder and decoder. The WOM-code designed achieves a close enough WOM-rate to the best known WOM-rate. It can be one of the future works after this PhD to explore an analytical approach to efficiently calculate the WOM-rate given the designed code's parity check matrix. Currently, the WOM-rates are calculate by running simulations, which has limitations to how large parameters n, k and q can be.

4.3.2 WOM-codes Designed from Construction B

Applying the best known two-write binary WOM-codes to the code construction B, WOM-codes with larger alphabet size are designed. The WOM-rates achieved were higher than what was achieved in [71]. The best non-binary WOM-codes are presented as follows.

Section 3.3 proposed that a two-write binary WOM-code can be efficiently constructed from any linear block code C using a critical criterion to maximize the WOM-rate. The search criterion was carried out for binary BCH codes with $n = 31$ and the results are tabulated in Table 4.4. From the second column of Table 4.4, the highest WOM-rate achieved is 1.4962, constructed with the binary BCH [31,21]. This binary WOM-code is applied as a base code to construct higher alphabet size (e.g. $q = 4, 8$) WOM-codes and the results are summarized in the third and fourth column of Table 4.4.

Table 4.4: WOM-rates of the WOM-codes constructed from BCH codes.

Base Code	$\mathcal{R}(C_q)$		
	$q = 2$	$q = 4$	$q = 8$
BCH [31,26]	1.4059	2.8118	4.2177
BCH [31,21]	1.4962	2.9924	4.4886
BCH [31,16]	1.4518	2.9036	4.3554
*Best prior-art WOM-code [33,22]	1.4928	2.9856	4.4784

* The best prior-art WOM-codes is extracted from [23,71]

Similar to the binary case, the efficiency of the newly designed binary WOM-code is calculated. For a fair comparison, the non-binary case $q = 4$ is

considered. Although WOM-codes designed by construction A allows the codeword length and dimension to be reduced by a significant amount, the coding efficiency is 0.8% lesser than the best prior-art WOM-code. However, the best WOM-code designed by construction B is able to achieve an increase of 1% increase in efficiency. It is reiterated again that the best known WOM-code from literature is obtain from a computer search and therefore lacks a practical code design, which might lead to a complicated encoding and decoding process. The result of the comparison between construction A and B benchmarking against the best known 4-ary WOM-code is compared in Table 4.5.

Table 4.5: Comparison of WOM-rates between state-of-the-art WOM-code and the achieved WOM-codes from construction A and B.

	WOM-codes found in literature	Achieved by construction A	Achieved by construction B
$\mathcal{R}(C_4)$	2.9856	2.9572	2.9924
η_4	89.88%	89.02%	90.08%

4.4 Conclusion

In this chapter, two code constructions were presented for non-binary WOM-codes. The WOM-codes designed in this chapter focuses on two writes, because higher re-writability will greatly penalize both the WOM-rates and the memory storage efficiency, which is undesirable for data storage applications. Code construction A has shown its capability of producing high WOM-rate codes without

requiring large values of n and k . Sacrificing 0.8% of coding efficiency can result in the codeword length n to be reduced by approximately 75.7% and dimension k to be reduced by approximately 81.8%. Another efficient code construction method proposed in [71] is briefly described and using the proposed code design criterion, we managed to design efficient two-write WOM-codes, whose efficiency is 1% higher than those available in literature.

An analytical approach to efficiently calculate the WOM-rate given the designed base code's parity check matrix remains open for future research. Currently, the WOM-rates are calculate by running simulations, which has limitations to how large parameters n , k and q can be. Using the method proposed by code construction B, we discovered non-binary WOM-codes that had better rates than all previously known codes.

CHAPTER 5

ENUMERATIVE CODING FOR RANK MODULATION (RM) CODES

Codes based on RM are a new information representation scheme and was proposed as a novel information representation scheme for multilevel NVM [26]. The conventional programming scheme used in Flash memories can easily result in overshooting and block eraser, which seriously affect the lifetime of Flash [28]. In addition, the drift of the threshold voltage in aging devices also leads to read errors. Similar to Flash memories, the multilevel phase-change memory (PCM) has an impairment of resistance drift [32], which may affect the memory sensing accuracy significantly. By programming and ranking the cell levels to their relative values rather than the accurate absolute values, the RM codes have shown potential to avoid overshooting during programming and reduce the read errors due to the drift of threshold voltage or resistance values.

RM codes are currently encoded and decoded based on the table look-up, which may occupy large memory space and can easily become an engineering impossibility with long codeword lengths. In this work, approaches are proposed to encode and decode RM codes based on a lexicographic ordering based on the

principle of a conventional enumerative coding scheme. The lexicographic ordering of each RM code is proposed to be calculated analytically instead of deriving it through brute-force enumerative methods. Thereafter, a novel enumerative coding scheme is proposed that analytically maps the RM codewords with its rank and vice versa. Further analysis showed that the computational complexity of the encoder and decoder is reduced.

5.1 Prior-art Codes Based on RM Schemes

5.1.1 Full RM Codes

The RM scheme was initially proposed by [77] to combat problems of overshooting and memory endurance in aging Flash memory devices. They are most commonly referred to as RM codes. For differentiation purpose in this work, they are labeled as the full RM (FRM) codes.

In the RM scheme of n memory cells, information is stored as the rank of cells, in descending order. Let $s = (s_1, s_2, \dots, s_n)$ denote the absolute value of n cells, whereby $s_j \neq s_l$, for all $j \neq l$. Induced by the absolute values of the n cells, information about the relative values of these cells is represented in the form of a vector. For example, given $n = 5$ and the absolute values of the cell states are

$$s = (1.2, 3.4, 9.2, 6.8, 4.5). \quad (5.1)$$

The induced permutation codeword, denoted by c_R , is given by

$$c_R = (3, 4, 5, 2, 1). \quad (5.2)$$

5.1.2 Partial RM Codes

There has been a significant number of works evolving around RM codes [72, 73]. One drawback of the RM scheme is that a large number of comparisons are needed to determine the relative order of the memory cells in the RM codeword during decoding. In order to lower the number of comparisons required, partial rank modulation code (PRM) [74] was proposed, where only the highest cell values are considered for information representation. The PRM scheme is denoted with (n, ρ) -PRM. That is, for an $(4, 2)$ -PRM code, given that the absolute values are $s = (1.2, 3.4, 6.8, 4.5)$, the induced PRM code is represented as $\mathbf{c}_p = (3, 4 \mid 2, 1)$ to emphasize that information is represented by the first two entries. Since only the relative order of the top 2 cells contains information, the shortened PRM codeword is represented as $\mathbf{c} = (3, 2)$. Throughout this chapter, the shortened codeword $\mathbf{c}$ is used during encoding and decoding.

Currently both encoding and decoding of the RM codes and PRM codes are based on memory consuming LUT. It is easy to verify that sizes of the LUT for RM and PRM scheme are $n!$ and $\binom{n}{\rho} \rho!$, respectively. Plotting these two functions in Figure 5.1, we observe that the size of the LUT can be reduced through PRM by decreasing the number of effective channel bits used to represent the information bits (*i.e.* reducing value of parameter ρ). However, even in the scenario of the PRM code, the increase in the size of the LUT with respect to n is still exponential. Thus, the memory space will quickly become insufficient for RM or PRM codes with long codeword lengths (*e.g.* $n > 24$).

Figure 5.1: Size of LUT as a function of codeword length n .

5.2 Simplified Enumerative Coding Scheme

In this section we proposed a novel enumerative coding scheme, which provides an analytical approach to compute the lexicographic index $i_c(c)$, which is specifically designed for RM/PRM codes. As the simplified approach, proposed in Section II, is computationally expensive for long codeword length n , the novel approach aims to provide a more efficient method to compute $i_c(c)$ for large n . Before introducing the novel enumerative coding, we propose the definition of two crucial vectors used during the encoding and decoding operations – the position vector $\mathbf{v}$ and the weight vector $\mathbf{w}$.

5.2.1 Simplified Enumerative Coding for RM

A conventional enumerative scheme was first described by Cover in [75]. The scheme provides a unique ordering that maps information bits into codewords,

using an intuitive and straightforward method. The method first list out all codewords before the exhaustive search is performed to retrieve the location of a particular codeword. In this section, we give a brief review of the conventional scheme suggested by Cover. We define C to represent the set of codewords, with length n and alphabet size q , that corresponds to the set of distinct information bits. Let $\mathbf{c} = (c_1, c_2, \dots, c_n)$ be an element of a set $C = \{1, 2, \dots, q\}^n$. For any $k \leq n$, the number of codewords with the first k coordinates given by $(c_1, c_2, \dots, c_k) \in C$ is represented by $n_c(c_1, c_2, \dots, c_k)$. It has also been proven in [75] that $\mathbf{x}$ have a lower rank than $\mathbf{y}$ if and only if the lexicographic index $i_c(\mathbf{x}) < i_c(\mathbf{y})$ and the lexicographic index is defined as

$$i_c(\mathbf{c}) = \sum_{j=1}^n \sum_{m=1}^{c_j-1} n_c(c_1, c_2, \dots, c_{j-1}, m). \quad (5.3)$$

The conventional enumerative scheme discussed was not customized for codes based on RM, which have an added constraint that no two entries of the codeword are the same. It also includes an exhaustive search required during encoding of $\mathbf{c}$ to calculate its corresponding rank $i_c(\mathbf{c})$. Therefore, we present a simplified enumerative coding scheme that is customized for a PRM code, denoted by (n, ρ) -PRM.

Definition 5.2.1. Let C be an (n, ρ) -PRM code. A codeword $\mathbf{c} = (c_1, c_2, \dots, c_\rho) \in C$ if the following conditions are satisfied.

- $c_i \in \{1, 2, \dots, n\}$, for $1 \leq i \leq \rho$
- $c_i = c_j$ if and only if $i = j$

A closer look at Definition 5.2.1 reveals that the RM code is a specific type of PRM code when $\rho = n - 1$. If C represents the codewords in the (n, ρ) -PRM code, the calculation of n_c is defined in the theorem that follows.

Theorem 5.2.1. *Let C be an (n, ρ) -PRM code. The number of codewords in C with the first k coordinates of $\mathbf{c}$ as $(c_1, c_2, \dots, c_k)$ is*

$$n_c(c_1, \dots, c_k) = \begin{cases} \frac{(n-k)!}{(n-\rho)!}, & \text{if } c_i \neq c_j, \text{ for all } i \neq j. \\ 0, & \text{otherwise} \end{cases} \quad (5.4)$$

Proof. Let $C \subseteq \{1, 2, \dots, n\}^\rho$ denote the set of codewords in (n, ρ) -PRM code and let $\mathbf{c} = (c_1, c_2, \dots, c_\rho)$. To satisfy conditions stated in Definition 1, one can easily deduce that $n_c(c_1, \dots, c_k) = 0$ if there exists a $c_i = c_j$, for some $i \neq j$ and $k \leq \rho$. For the remaining part of the proof, it suffices to derive an expression for $n_c(\mathbf{c})$ that corresponds to the remaining combinations of $(c_1, \dots, c_k)$. This is equivalent to counting the number of codewords with prefix $(c_1, \dots, c_k)$, which is

$$\begin{aligned} & (n-k)[n-(k+1)] \cdots [n-(k+(\rho-k)-1)] \\ &= (n-k)(n-k-1) \cdots (n-\rho-1) \\ &= \frac{(n-k)!}{(n-\rho)!} \end{aligned} \quad (5.5)$$

□

The following example is provided to show the analytical computation of the lexicographic rank $i_c(\mathbf{c})$.

Example 5.2.1. Let C be an $(4,3)$ -PRM code. Consider a codeword $\mathbf{c} = (3,1,2) \in C$, the lexicographic index is

$$\begin{aligned}
 i_c(3,1,2) &= \sum_{j=1}^3 \sum_{m=1}^{c_j-1} n_c(c_1, c_2, \dots, c_{j-1}, m) \\
 &= n_c(1) + n_c(2) + n_c(3,1,1) \\
 &= 2 \times \frac{(4-1)!}{(4-3)!} \\
 &= 12.
 \end{aligned} \tag{5.6}$$

Theorem 5.2.1 provides a straightforward method of calculating the corresponding value of $n_c(\mathbf{c})$ for all codes based on any RM scheme. From the ranking based on (5.3), there exists a one-to-one lexicographic mapping for each decimal digit to a unique RM codeword $\mathbf{c} \in C$. In the section that follows, we illustrate the encoding and decoding operations.

5.2.2 Encoding Scheme

Let C be an (n, ρ) -PRM code. The encoder takes a decimal digit $i_c(\mathbf{c})$ as its input, and computes the unique codeword $\mathbf{c} = (c_1, c_2, \dots, c_\rho) \in C$. Due to the characteristic of codes based on RM, we define a set

$$\mathcal{J}_i = \begin{cases} \{1, \dots, n\}, & \text{if } i = 1 \\ \{1, \dots, n\} \setminus \{c_1 \dots c_{i-1}\}, & \text{if } i > 1 \end{cases} \tag{5.7}$$

The prior value of c_i 's are omitted to ensure that the constraint $c_i = c_j$ if and only if $i = j$ is not violated. The procedure of obtaining the codeword c is explained as follows.

1. Let $J_i[j]$ be the j -th entry of set J_i . If $n_c(J_1[j]) < i_c(c) < n_c(J_1[j+1])$, the first entry of c is $c_1 = J_1[j+1]$ and update $i_c(c) = i_c(c) - n_c(J_1[j])$. Otherwise, $c_1 = J_1[j]$. The set J_2 representing the set of possible entries of c_2 is updated after the value of c_1 is finalized.

$$J_2 = \{1, 2, \dots, n\} \setminus c_1 \quad (5.8)$$

2. For the remaining i -th entries of the codeword c (e.g. $i = 2, \dots, \rho$), if $n_c(c_1, \dots, c_{i-1}, J_i[j]) < i_c(c) < n_c(c_1, \dots, c_{i-1}, J_i[j+1])$, we set $c_i = J_i[j+1]$ and update $i_c(c) = i_c(c) - n_c(c_1, \dots, c_{i-1}, J_i[j])$. Otherwise $c_i = J_i[j]$. The set of possible entries of c_{i+1} is

$$J_{i+1} = \{1, 2, \dots, n\} \setminus \{c_1, \dots, c_i\}. \quad (5.9)$$

The following example illustrates the proposed encoding scheme.

Example 5.2.2. Let C be an $(8, 4)$ -PRM code. For a lexicographic index $i_c(c) = 49$, we wish to derive the corresponding codeword $c = (c_1, c_2, c_3, c_4)$. Equation (5.3) is rewritten as

$$49 = \sum_{j=1}^4 \sum_{m=1}^{c_j-1} n_c(c_1, c_2, \dots, c_{j-1}, m) \quad (5.10)$$

Iteration 1: $c_1 \in J_1 = \{1, 2, 3, 4, 5, 6, 7, 8\}$

- $c_1 = 1: \sum_{m=1}^0 n_c(m) = 0 < 49$

5.2 Simplified Enumerative Coding Scheme

88

- $c_1 = 2: \sum_{m=1}^1 n_c(m) = 210 > 49$
- From the above computation, the first entry of the codeword is $c_1 = 1$ and we update $i_c(c) = 49 - 0 = 49$.

Iteration 2: $c_2 \in \mathcal{I}_2 = \{2, 3, 4, 5, 6, 7, 8\}$

- $c_2 = 2: \sum_{m=1}^1 n_c(1, m) = 0 < 49$
- $c_2 = 3: \sum_{m=1}^2 n_c(1, m) = 30 < 49$
- $c_2 = 4: \sum_{m=1}^3 n_c(1, m) = 60 > 49$
- From the above computation, the second entry of the codeword is $c_2 = 3$ and we update $i_c(c) = 49 - 30 = 19$.

Iteration 3: $c_3 \in \mathcal{I}_3 = \{2, 4, 5, 6, 7, 8\}$

- $c_3 = 2: \sum_{m=1}^1 n_c(1, 3, m) = 0 < 19$
- $c_3 = 4: \sum_{m=1}^3 n_c(1, 3, m) = 5 < 19$
- $c_3 = 5: \sum_{m=1}^4 n_c(1, 3, m) = 10 < 19$
- $c_3 = 6: \sum_{m=1}^5 n_c(1, 3, m) = 15 < 19$
- $c_3 = 7: \sum_{m=1}^6 n_c(1, 3, m) = 20 > 19$
- From the above computation, the third entry of the codeword is $c_3 = 6$ and we update $i_c(c) = 19 - 15 = 4$.

Iteration 4: $c_4 \in \mathcal{I}_4 = \{2, 4, 5, 7, 8\}$

- $c_4 = 2: \sum_{m=1}^1 n_c(1, 3, 6, m) = 0 < 4$
- $c_4 = 4: \sum_{m=1}^3 n_c(1, 3, 6, m) = 1 < 4$
- $c_4 = 5: \sum_{m=1}^4 n_c(1, 3, 6, m) = 2 < 4$
- $c_4 = 7: \sum_{m=1}^6 n_c(1, 3, 6, m) = 3 < 4$

- $c_4 = 8: \sum_{m=1}^7 n_c(1,3,6,m) = 4 = i_c$
- From the above computation, the first entry of the codeword is $c_4 = 8$ and we update $i_c(\mathbf{c}) = 4 - 4 = 0$.

When $i_c(\mathbf{c})$ is reduced to zero, we terminate the algorithm and obtain the RM code $\mathbf{c} = (1, 3, 6, 8)$.

We remark that for the iterations in Example 5.2.2, the values of the prior c_i 's are omitted to ensure that the condition, $c_i = c_j$ if and only if $i = j$, is not violated. From Example 5.2.2, the possible values of c_2 are $\{2, 3, 4, 5, 6, 7, 8\}$, because we omit the prior value $c_1 = 1$.

5.2.3 Decoding Scheme

On the other hand, the decoder provides the inverse operation to achieve the same mapping. Unlike the encoder which performs an exhaustive search, the decoder performs a series of arithmetic operations shown in (5.3) to evaluate the lexicographic index $i_c(\mathbf{c})$ of the corresponding codeword $\mathbf{c} \in \mathcal{C}$. The procedure of obtaining the lexicographic index is illustrated below.

Example 5.2.3. Let $\mathcal{C}$ be an $(8, 4)$ -PRM code. The decoded lexicographic index of codeword $\mathbf{c} = (1, 3, 6, 8)$ is

$$\begin{aligned}
i_c(1, 3, 6, 8) &= \sum_{j=1}^4 \sum_{m=1}^{c_j-1} n_c(c_1, c_2, \dots, c_{j-1}, m) \\
&= \sum_{m=1}^2 n_c(1, m) + \sum_{m=1}^5 n_c(1, 3, m) \\
&\quad + \sum_{m=1}^7 n_c(1, 3, 6, m) \\
&= \frac{6!}{4!} + 3 \times \frac{5!}{4!} + 4 \times \frac{4!}{4!} \\
&= 49.
\end{aligned} \tag{5.11}$$

5.2.4 Algorithm Implementation

The algorithms related to the encoding and decoding operation for the simplified enumerative coding scheme is discussed in this section. We consider three algorithms, the first is function **CALCNC** which is implemented to compute the value of $n_c(c_1, \dots, c_k)$. This function is used in both the encoder and decoder implementation, represented by functions **SIMPLEENCODER** and **SIMPLEDECODER** respectively.

1. **CALCNC**: To calculate the number of codewords in C such that the first k entries is given by $(c_1, \dots, c_k)$. The input parameters required are the first k coordinates and the parameters defining the PRM code – n and ρ .

Algorithm 1 Calculate the number of codewords in $\mathcal{C}$ for which the first k coordinates are given by $(c_1, \dots, c_k)$

```

1: procedure CALCNC( $(c_1, \dots, c_k), n, \rho$ )
2:    $k = \text{length of } (c_1, \dots, c_k)$ 
3:    $count = 1$ 
4:   for  $i \leftarrow 0$  to  $k - 1$  do
5:      $count = 1$ 
6:     for  $j \leftarrow 0$  to  $k - 1$  do
7:       if  $j \neq i$  and  $c[j] == c[i]$  then
8:          $count = count + 1$ 
9:       end if
10:    end for
11:    if  $count > 1$  then
12:      return 0
13:    end if
14:  end for
15:  return  $\frac{(n-k)!}{(n-\rho)!}$ 
16: end procedure

```

2. **SIMPLEENCODER**: The function is an implementation to achieve the desired one-to-one mapping between $i_c(\mathbf{c}) \rightarrow \mathbf{c}$, described in Section 5.2.1. The function **ENCODER** requires input parameters $i_c(\mathbf{c})$, n and ρ to return the corresponding encoded codeword $\mathbf{c}$. The algorithm first initialized a set $\mathcal{J}_i$ as the set of possible values for c_i 's, and next evaluate the value of each c_i iteratively. The integer value of c_i is increased until $\sum n_c \geq i_c(\mathbf{c})$. When this criterion is met, the finalized value of c_i is removed from $\mathcal{J}_i$. Algorithm 2 presents the details of the **SIMPLEENCODER** function.

Algorithm 2 Simplified Encoding Algorithm

```

1: procedure SIMPLEENCODER( $i_c, n, \rho$ )
2:    $\mathcal{I} = [1, 2, \dots, n]$ 
3:    $\mathbf{c} = \text{zeroes}(\rho)$ 
4:   for  $i \leftarrow 0$  to  $\rho - 1$  do
5:      $\mathbf{x} = \text{zeroes}(i + 1)$ 
6:     for  $j \leftarrow 0$  to  $i$  do
7:        $\mathbf{x}[j] = \mathbf{c}[j]$ 
8:       if  $j == i$  then
9:          $\mathbf{x}[j] = \mathcal{I}[0]$ 
10:      end if
11:    end for
12:     $\text{index} = 0$ 
13:    while  $i_c > 0$  do
14:       $i_{c1} = \text{CALCNC}(\mathbf{x}, n, \rho)$ 
15:      if  $i_{c1} > i_c$  then
16:        break
17:      else
18:         $i_c = i_c - i_{c1}$ 
19:         $\text{index} = \text{index} + 1$ 
20:         $\mathbf{x}[i] = \mathcal{I}[\text{index}]$ 
21:      end if
22:    end while
23:     $\mathbf{c}[i] = \mathbf{x}[i]$ 
24:     $\mathcal{I} \leftarrow \text{remove } \mathbf{x}[i] \text{ from } \mathcal{I}$ 
25:  end for
26:  return  $\mathbf{c}$ 
27: end procedure

```

3. **SIMPLEDECODER**: On the other hand, the decoder provides the inverse operation to achieve the one-to-one mapping $\mathbf{c} \rightarrow i_c(\mathbf{c})$. Unlike the encoder,

which performs some form of coarse-to-fine search, the decoder performs a series of arithmetic operations, in order to evaluate the value of $i_c(\mathbf{c})$. Algorithm 3 presents the details of the **SIMPLEDECODER** function.

Algorithm 3 Simplified Decoding Algorithm

```

1: procedure SIMPLEDECODER( $\mathbf{c}, n, \rho$ )
2:    $i_c = 0$ 
3:   for  $i \leftarrow 0$  to  $\rho - 1$  do
4:      $x = \text{zeroes}(i + 1)$ 
5:     if  $i \neq 0$  then
6:       for  $j \leftarrow 0$  to  $i - 1$  do
7:          $x[j] = \mathbf{c}[j]$ 
8:       end for
9:     end if
10:    for  $j \leftarrow 0$  to  $\mathbf{c}[i] - 2$  do
11:       $x[i] = x[i] + 1$ 
12:       $i_c = i_c + \text{CALCNC}(x, n, \rho)$ 
13:    end for
14:  end for
15:  return  $i_c$ 
16: end procedure

```

5.3 A Novel Enumerative Coding Scheme

In this section, a novel enumerative coding scheme is proposed. The novel coding scheme provides an analytical approach to compute the lexicographic index $i_c(\mathbf{c})$ that is customized for codes based on RM schemes. As the simplified approach proposed in Section 5.2 is computationally expensive for long codeword length n , the novel approach aims to provide a more efficient method to compute $i_c(\mathbf{c})$ for larger n . Before introducing the novel enumerative coding, the definition of two crucial vectors used during the encoding and decoding operations – the position vector $\mathbf{v}$ and the weight vector $\mathbf{w}$, is proposed.

5.3.1 Preliminaries

The definition of position vector $\mathbf{v}$ is first presented and the details of obtaining $\mathbf{v}$ from a codeword $\mathbf{c}$ of an (n, ρ) -PRM code is explained in Example 5.3.1.

Definition 5.3.1. Let C be an (n, ρ) -PRM code and $\mathbf{c} = (c_1, c_2, \dots, c_\rho) \in C$. A position vector $\mathbf{v} = (v_1, v_2, \dots, v_\rho)$ is defined as follows. Each $v_i \in \{0, 1, \dots, |\mathcal{J}_i| - 1\}$, where set $\mathcal{J}_i$ is defined by (5.7) and $c_i \in \mathcal{J}_i$. The possible values of c_i are tabulated in the first row of the table below.

c_i	1	2	...	n
v_i	0	1	...	$ \mathcal{J}_i - 1$

The corresponding value of v_i can be extracted from the table above.

The following example illustrates the procedure of obtaining $\mathbf{v}$ from a particular RM codeword $\mathbf{c}$.

Example 5.3.1. For a $\mathbf{c} = (2, 5, 3)$, we aim to deduce its corresponding position vector $\mathbf{v} = (v_1, v_2, v_3)$ of a $(5, 3)$ -PRM code.

1. For $c_1 = 2$, we have set $\mathcal{J}_1 = \{1, 2, 3, 4, 5\}$.

c_i	1	2	3	4	5
v_i	0	1	2	3	4

$$\Rightarrow v_1 = 1.$$

2. For $c_2 = 5$, we have set $\mathcal{J}_2 = \{1, 3, 4, 5\}$.

c_i	1	3	4	5
v_i	0	1	2	3

$$\Rightarrow v_2 = 3.$$

3. For $c_3 = 3$, we have set $\mathcal{I}_3 = \{1, 3, 4\}$.

c_i	1	3	4
v_i	0	1	2

$$\Rightarrow v_3 = 1.$$

Therefore the unique position vector $\mathbf{c} = (2, 5, 3)$ corresponding to is $\mathbf{v} = (1, 3, 1)$.

Unlike the position vector $\mathbf{v}$ which is unique to each codeword, the weight vector $\mathbf{w}$, introduced in Definition 5.3.2 is unique to each (n, ρ) -PRM code.

Definition 5.3.2. Let C be an (n, ρ) -PRM code. A weight vector is defined as $\mathbf{w} = (w_1, w_2, \dots, w_\rho)$ and w_i denotes the number of codewords in C with $v_i = 0$, where v_i represents the i -th entry in the corresponding position vector $\mathbf{v}$.

Given an $(5, 3)$ -PRM code is denoted as C . The codewords $\mathbf{c} \in C$ and its corresponding position vector $\mathbf{v}$ are listed down in Table 5.1. The first entry of the weight vector $\mathbf{w}$ represents the number of codewords with the first entry of their corresponding position vector as $v_1 = 0$ (e.g. total number of codewords with the general form $\mathbf{c} = (1, c_2, c_3)$). The total number of combinations of codewords with such a configuration is $w_1 = 4 \times 3 = 4!/2!$. The rest of the weight vector entries are calculated similarly. Hence, we obtain the weight vector $\mathbf{w} = (12, 3, 1)$ for PRM-code C .

Table 5.1: Lexicographic ranking of RM codeword c and its corresponding entries for $v = (v_1, v_2, v_3)$.

c	v_1	v_2	v_3
(1,2,3)	0	0	0
(1,2,4)	0	0	1
(1,2,5)	0	0	2
(1,3,2)	0	1	0
(1,3,4)	0	1	1
(1,3,5)	0	1	2
$\vdots$	$\vdots$	$\vdots$	$\vdots$
(5,4,3)	4	3	2

However if n gets too large, it is impractical to list down all the codewords and count the number of codewords to obtain the values of w_i . Theorem 5.3.1 provides a more efficient method to compute the weight vector w for a particular (n, ρ) -PRM code.

Theorem 5.3.1. Let C be an (n, ρ) -PRM code and a weight vector be $w = (w_1, w_2, \dots, w_\rho)$. For $1 \leq i \leq \rho$,

$$w_i = \frac{(n-i)!}{(n-\rho)!} \quad (5.12)$$

Proof. Let $c = (c_1, c_2, \dots, c_\rho) \in C$. From Definition 5.3.2, it is known that w_i represents the number of combinations of $(c_1, c_2, \dots, c_\rho)$ such that $v_i = 0$. This is equivalent to finding the total number of combinations of codewords with such a configuration

$$(1, 2, \dots, i, c_{i+1}, \dots, c_\rho), \quad (5.13)$$

where $c_k \in \{k, k+1, \dots, n\}$, for all $i < k < n$. The total number of combinations is

$$\begin{aligned} \prod_{m=k}^n |c_m| &= (n-i)(n-(i+1)) \cdots [n-(i+\rho-i-1)] \\ &= \frac{(n-i)!}{(n-\rho)!}. \end{aligned} \quad (5.14)$$

□

Example 5.3.2 illustrates the details of computing the weight vector w corresponding to a (5,3)-PRM code.

Example 5.3.2. For a (5,3)-PRM code the weight vector is calculated as shown below.

$$w = \left(\frac{(5-1)!}{(5-3)!}, \frac{(5-2)!}{(5-3)!}, \frac{(5-3)!}{(5-3)!} \right) = (12, 3, 1) \quad (5.15)$$

5.3.2 Encoding Scheme

In this section, the encoding scheme of the novel enumerative coding is discussed. The simplified encoding scheme requires an exhaustive search that returns the location of a specified codeword. The novel encoding scheme replaces the exhaustive search with an analytical approach to efficiently return the location of the codeword. Before presenting the encoder, we prove that this proposed enumerative coding scheme achieves a one-to-one lexicographic ordering.

Theorem 5.3.2. Let C be an (n, ρ) -PRM code. The unique lexicographic index $i_c(c)$ of a codeword $c \in C$ is

$$i_c(c) = v \cdot w \quad (5.16)$$

where $\mathbf{v}$, $\mathbf{w}$ are the position and weight vector respectively.

Proof. From Definition 3, each w_i represents the number of codewords in $\mathcal{C}$ such that $v_i = 0$. If $\rho = 1$, we let $\mathbf{v} = \mathbf{v}_1 = 0$. The position vector $\mathbf{w}$ can be reduced to just a single entry w_1 , where $w_1 = 1$ is easily verified. For the specific case, the index $i_c(\mathbf{c})$ is reduced to

$$i_c(\mathbf{c}) = v_1 \times w_1 = v_1. \quad (5.17)$$

Since $\mathbf{v}$ is unique to a different codeword and $i_c(\mathbf{c}) = v_1$, it is easy to verify that there exists a 1-to-1 lexicographic ordering.

For $\rho > 1$, the position vector and weight vector are defined as $\mathbf{v} = (v_1, v_2, \dots, v_\rho)$ and $\mathbf{w} = (w_1, w_2, \dots, w_\rho)$ respectively. For codewords with corresponding position vectors of the general form $\mathbf{v} = (0, \dots, v_{\rho-1}, v_\rho)$, the lexicographic index is reduced to

$$i_c(\mathbf{c}) = v_{\rho-1} \times w_{\rho-1} + v_\rho. \quad (5.18)$$

Established from the above argument, the lexicographic mapping of codewords $\mathbf{c}$ is provided in Table 5.2. Eventually all the codewords in the RM code $\mathcal{C}$ are accounted for in this lexicographic ordering.

□

Table 5.2: Mapping of the RM codeword to its corresponding $\mathbf{v}$ and $i_c(\mathbf{c})$.

Codeword $\mathbf{c}$	Position vector $\mathbf{v}$	Lexicographic index $i_c(\mathbf{c}) = v_{\rho-1}w_{\rho-1} + v_\rho$
$(1, 2, \dots, \rho-1, \rho)$	$(0, \dots, 0, 0)$	0
$(1, 2, \dots, \rho-1, \rho+1)$	$(0, \dots, 0, 1)$	1
$(1, 2, \dots, \rho-1, \rho+2)$	$(0, \dots, 0, 2)$	2
$\vdots$	$\vdots$	$\vdots$
$(1, 2, \dots, \rho-1, n)$	$(0, \dots, 0, w_{\rho-1}-1)$	$w_{\rho-1}-1$
$(1, 2, \dots, \rho, \rho-1)$	$(0, \dots, 1, 0)$	$w_{\rho-1}$
$(1, 2, \dots, \rho, \rho+1)$	$(0, \dots, 1, 1)$	$w_{\rho-1}+1$
$(1, 2, \dots, \rho, \rho+2)$	$(0, \dots, 1, 2)$	$w_{\rho-1}+2$
$\vdots$	$\vdots$	$\vdots$
$(1, 2, \dots, \rho, n)$	$(0, \dots, 1, w_{\rho-1}-1)$	$w_{\rho-1} + w_{\rho-1} - 1$
$(1, 2, \dots, \rho+1, \rho-1)$	$(0, \dots, 2, 0)$	$2 \times w_{\rho-1}$
$\vdots$	$\vdots$	$\vdots$

Given an (n, ρ) -PRM code C and lexicographic index $i_c(\mathbf{c})$, the encoding operation is described below.

- 1) The weight vector $\mathbf{w} = (w_1, w_2, \dots, w_\rho)$ of C is calculated by Equation (5.12).
- 2) The position vector $\mathbf{v} = (v_1, v_2, \dots, v_\rho)$ is calculated.

For all $j = 1, 2, \dots, \rho$:

- The j entry of $\mathbf{v}$ is $v_j = \left\lfloor \frac{i_c(\mathbf{c})}{w_j} \right\rfloor$.
 - Next, the index $i_c(\mathbf{c})$ is updated to $i_c(\mathbf{c}) \pmod{w_j}$.
- 3) Lastly, the RM code $\mathbf{c} = (c_1, c_2, \dots, c_\rho)$ is derived from $\mathbf{v}$. The first entry $c_1 = v_1 + 1$. For $j = 2, \dots, \rho$, the j -th entry of $\mathbf{c}$ is extracted from the table below, where the second row of the table is the possible entries of c_j .

v_j	0	1	...	$ I_j - 1$
c_j	1	2	...	$n - j + 1$

From the outline of the encoding scheme described above, we provide the detailed routine for the computation of codeword $\mathbf{c}$.

Algorithm 4 Novel Encoding Scheme

```

1: procedure ENCODER( $i_c(\mathbf{c}), n, \rho$ )
2:    $\mathbf{w} = \text{zeroes}(\rho)$ 
3:   for  $i \leftarrow 1$  to  $\rho$  do
4:      $w[i - 1] = \frac{(n-i)!}{(n-\rho)!}$ 
5:   end for
6:    $\mathbf{v} = \text{zeroes}(\rho)$ 
7:   for  $i \leftarrow 0$  to  $\rho - 1$  do
8:      $\mathbf{v}[i] = \lfloor \frac{i_c(\mathbf{c})}{\mathbf{w}[i]} \rfloor$ 
9:      $i_c(\mathbf{c}) = i_c(\mathbf{c}) \pmod{\mathbf{w}[i]}$ 
10:  end for
11:   $\mathcal{I} = [1, 2, \dots, n]$ 
12:   $\mathbf{c} = \text{zeroes}(\rho)$ 
13:  for  $i \leftarrow 0$  to  $\rho - 1$  do
14:     $j = \mathbf{v}[i]$ 
15:     $\mathbf{c}[i] = \mathcal{I}[j]$ 
16:     $\mathcal{I} \leftarrow \text{remove } \mathbf{v}[i] \text{ from } \mathcal{I}$ 
17:  end for
18:  return  $\mathbf{c}$ 
19: end procedure

```

An example is illustrated to observe the encoding operation of the novel enumerative scheme.

Example 5.3.3. Let $\mathcal{C}$ be the set containing codewords from an $(8, 4)$ -PRM code.

We encode the following index $i_c(\mathbf{c}) = 49$ to derive its corresponding codeword $\mathbf{c} = (c_1, c_2, c_3, c_4)$.

Step 1: Calculate weight vector from (5.12).

$$\mathbf{w} = \left[\frac{7!}{4!}, \frac{6!}{4!}, \frac{5!}{4!}, \frac{4!}{4!} \right] = [120, 30, 5, 1]$$

Step 2: Calculate $\mathbf{v}$ such that $i_c(\mathbf{c}) = \mathbf{v} \cdot \mathbf{w}$. From the above conditions,

$$49 = 120v_1 + 30v_2 + 5v_3 + v_4.$$

To satisfy the equation, $\mathbf{v} = (0, 1, 3, 4)$.

Step 3: Derive $\mathbf{c}$ from $\mathbf{v} = (0, 1, 3, 4)$. The first entry of $\mathbf{c}$ is $c_1 = v_1 + 1 = 1$.

Excluding $c_1 = 1$ from the set of possible values of c_2 .

v_2	0	1	2	3	4	5	6
c_2	2	3	4	5	6	7	8

Therefore $v_2 = 1 \rightarrow c_2 = 3$. Excluding $c_1 = 1, c_2 = 3$ from the set of possible values of c_3 .

v_3	0	1	2	3	4	5
c_3	2	4	5	6	7	8

Therefore $v_3 = 3 \rightarrow c_3 = 6$. Excluding $c_1 = 1, c_2 = 3, c_3 = 6$ from the set of possible values of c_4 .

v_4	0	1	2	3	4
c_4	2	4	5	7	8

Finally, we have $v_4 = 4 \rightarrow c_4 = 8$. Therefore, the codeword corresponding to lexicographic index $i_c(\mathbf{c}) = 49$ is $\mathbf{c} = (1, 3, 6, 8)$.

5.3.3 Decoding Scheme

The decoding scheme of the novel enumerative coding is discussed in this section. The simplified decoding scheme performs a series of arithmetic operations which can be computationally expensive for large n . We proposed a novel decoding scheme that provides an efficient analytical approach to return the lexicographic index. The decoding operation is described below.

- 1) Similar to the encoder, the weight vector $\mathbf{w} = (w_1, w_2, \dots, w_\rho)$ of C is calculated by

$$w_i = \frac{(n - i)!}{(n - \rho)!}.$$

- 2) The position vector $\mathbf{v} = (v_1, v_2, \dots, v_\rho)$ is calculated from $\mathbf{c}$. The first entry of $\mathbf{v}$ is $v_1 = c_1 - 1$. For $j = 2, \dots, \rho$, the j -th entry of $\mathbf{v}$ is derived from the table below, where the first row of the table is the possible entries of c_j .

c_j	1	2	...	$n - j + 1$
v_j	0	1	...	$ J_j - 1$

- 3) The lexicographic index is $i_c(\mathbf{c}) = \mathbf{v} \cdot \mathbf{w}$.

The detailed routine for computing the lexicographic index $i_c(\mathbf{c})$ of the corresponding codeword $\mathbf{c}$ is outlined in Algorithm 5. Example 5.3.4 illustrates the decoding scheme of the novel enumerative scheme.

Algorithm 5 Novel Decoding Scheme

```

1: procedure DECODER( $\mathbf{c}, n, \rho$ )
2:    $\mathbf{w} = \text{zeroes}(\rho)$ 
3:   for  $i \leftarrow 1$  to  $\rho$  do
4:      $w[i-1] = \frac{(n-i)!}{(n-\rho)!}$ 
5:   end for
6:    $\mathbf{v} = \text{zeroes}(\rho)$ 
7:    $\mathcal{I} = \{1, 2, \dots, n\}$ 
8:   for  $i \leftarrow 0$  to  $\rho-1$  do
9:      $\mathbf{v}[i] \leftarrow$  index of the entry of  $\mathcal{I}$  that is equal to  $c_i$ 
10:     $\mathcal{I} \leftarrow$  remove  $\mathbf{v}[i]$  from  $\mathcal{I}$ 
11:   end for
12:    $\mathcal{I} = \{1, 2, \dots, n\}$ 
13:    $i_c(\mathbf{c}) = 0$ 
14:   for  $i \leftarrow 0$  to  $\rho-1$  do
15:      $i_c(\mathbf{c}) = i_c(\mathbf{c}) + \mathbf{w}[i] * \mathbf{v}[i]$ 
16:   end for
17:   return  $i_c(\mathbf{c})$ 
18: end procedure

```

Example 5.3.4. Let C be the set containing codewords from an $(8, 4)$ -PRM code. We decode the codeword $\mathbf{c} = (1, 3, 6, 8)$ to derive its corresponding lexicographic index $i_c(\mathbf{c})$.

Step 1: Calculate weight vector from (5.12).

$$\mathbf{w} = \left[\frac{7!}{4!}, \frac{6!}{4!}, \frac{5!}{4!}, \frac{4!}{4!} \right] = [120, 30, 5, 1]$$

Step 2: Calculate $\mathbf{v}$ from $\mathbf{c} = (1, 3, 6, 8)$.

The first entry of $\mathbf{v}$ is $v_1 = c_1 + 1 = 1$. Excluding $c_1 = 1$ from the set of possible values of c_2 .

c_2	2	3	4	5	6	7	8
v_2	0	1	2	3	4	5	6

Therefore $c_2 = 3 \rightarrow v_2 = 1$. Excluding $c_1 = 1, c_2 = 3$ from the set of possible values of c_3 .

c_3	2	4	5	6	7	8
v_3	0	1	2	3	4	5

Therefore $c_3 = 6 \rightarrow v_3 = 3$. Excluding $c_1 = 1, c_2 = 3, c_3 = 6$ from the set of possible values of c_4 .

c_4	2	4	5	7	8
v_4	0	1	2	3	4

The position vector is $\mathbf{v} = (0,1,3,4)$.

Step 3: The lexicographic index $i_c(\mathbf{c}) = \mathbf{v} \cdot \mathbf{w} = 49$.

5.4 Complexity Comparison

Complexity studies the time an algorithm requires as a function of the size of the input function. This approach is commonly used to compare different algorithms that are used to solve the same problem. In this section, the aim is to find an upper bound to the number of operations required to perform the encoding and decoding procedures described in the previous sections. For all the implemented algorithms presented in this work, majority of the key operations performed are assignment based. Hence, the complexity of these algorithms is contributed from the loops and nested loops invoked.

5.4.1 Encoder Complexity Comparison

The encoder of the simplified enumerative scheme performs an exhaustive search which determines the location of the codeword $\mathbf{c} = (c_1, c_2, \dots, c_\rho)$ of the (n, ρ) -PRM code C . Prior to analyzing the complexity of the simplified encoder, we first examine the complexity of calculating the value of $n_c(c_1, c_2, \dots, c_i)$, for some $i \leq \rho$. The calculation of $n_c(c_1, c_2, \dots, c_i)$ is widely used in the encoder. The value of $n_c(c_1, c_2, \dots, c_i)$ represents the total number of codewords with the first i entries containing $(c_1, c_2, \dots, c_i)$. From (3), it is observed that the complexity in computing $n_c(c_1, c_2, \dots, c_i)$ comes from verifying if every entry in $(c_1, c_2, \dots, c_i)$ is unique. This investigation would require two for-loops, which translates to a complexity of $\mathcal{O}(i^2) \leq \mathcal{O}(\rho^2)$.

From the outline of the novel enumerative encoding scheme, the set $\mathcal{J}_i$ represents the possible entries of c_i and $\mathcal{J}_i[j]$ denotes the j -th entry of set $\mathcal{J}_i$. The value of c_i is increased iteratively, until

$$n_c(c_1, c_2, \dots, c_{i-1}, \mathcal{J}_i[j]) \leq i_c(\mathbf{c}) \leq n_c(c_1, c_2, \dots, c_{i-1}, \mathcal{J}_i[j+1]) \quad (5.19)$$

is satisfied. Since the maximum possible choices of c_i is $|\mathcal{J}_i| \leq n$, the complexity of deriving each c_i is $\mathcal{O}(n) \times \mathcal{O}(\rho^2)$. Since there are ρ entries for each codeword $\mathbf{c} \in C$, the total complexity for the encoder is

$$\mathcal{O}(n\rho^2) \times \mathcal{O}(\rho) \leq \mathcal{O}(n^4). \quad (5.20)$$

Next, we evaluate the computational complexity for the novel encoder proposed in this work. The computation of the weight and position vectors has a

complexity $\mathcal{O}(\rho)$. From line 2 to 10 for Algorithm 4, the complexity is bounded at $\mathcal{O}(\rho)$. It remains to derive an upper bound on the number of operations required for the rest of the encoder. The *for-loop* in line 13 consists of an operation to locate and remove a value v_i , which has an upper bound of ρ operations. Since this particular locate-and-remove operation is nested in a loop, we conclude the upper bound to be $\mathcal{O}(n^2) \leq \mathcal{O}(\rho^2)$.

By replacing the exhaustive search with an analytical method to calculate the codeword from a lexicographic ordering, we are able to improve the complexity by two orders.

5.4.2 Decoder Complexity Comparison

The improvement in complexity for the novel decoder is evaluated in this section. The simplified decoder performs a series of arithmetic operations, as shown in (5.3). From the analysis above, the operation of calculating $n_c(c_1, c_2, \dots, c_i)$, for some positive $i \leq \rho$, has complexity $\mathcal{O}(\rho^2)$. Since (5.3) consist of two summations, the complexity of the simplified decoder is $\mathcal{O}(\rho^4) \leq \mathcal{O}(n^4)$.

Algorithm 5 reflects the implementation for the novel decoder proposed in this work. Similar to its encoder, the decoder invokes the $\mathcal{O}(\rho)$ algorithm for computing the weight vector. However, a different approach is used to obtain the position vector $\mathbf{v}$. The routine of obtaining $\mathbf{v}$ by the decoder is defined by lines 6 to 11 in Algorithm 5. The computation of $\mathbf{v} = (v_1, v_2, \dots, v_\rho)$ consists of a search operation to locate and remove v_i . The complexity for the derivation of $\mathbf{v}$ at the decoder is $\mathcal{O}(\rho^2)$. Finally, the index $i_c(\mathbf{c})$ is calculated by performing the dot

product of $\mathbf{w}$ and $\mathbf{v}$, which takes an upper bound of $\mathcal{O}(\rho)$. As the process of computing the weight vector $\mathbf{w}$, the position vector $\mathbf{v}$ and $l_c(\mathbf{c})$, are independent of each other, the complexity of the novel decoding scheme is.

$$\mathcal{O}(\rho^2) \leq \mathcal{O}(n^2). \quad (5.21)$$

By replacing the series of arithmetic operations with an analytical approach to calculate the lexicographic ordering from a particular codeword, we are able to reduce the complexity by two orders.

5.5 Conclusion

Enumerative coding schemes are essential for the encoding and decoding of RM and PRM codes, as they provide a significant reduction of memory storage space by avoiding the usage of a LUT. Even with the conventional enumerative encoding scheme, where every codeword is assigned a unique lexicographic rank, the encoding and decoding operations are computationally expensive. The encoder requires an exhaustive search while the decoder performs a series of summations to successfully obtain the rank. As the codeword length increases, the conventional enumerative scheme also requires a large computational time.

In this paper, we customized the conventional enumerative scheme and proposed approaches to encode and decode the RM codes based on their lexicographic ordering. An analytical approach is proposed to replace the conventional brute-force enumeration method to compute the rank of each codeword.

Furthermore, we proposed a novel enumerative coding scheme that further reduces the computational complexity over the simplified enumerative coding by two orders of magnitude for both encoder and decoder.

CHAPTER 6

RM CODES FOR RESISTANCE DRIFT

In this chapter, the analysis on the performance of the RM coded PCM channel model is investigated. The novel RM scheme was initially proposed in [76] for the use in band-limited digital communication. Jiang subsequently proposed the RM scheme to combat asymmetric errors caused by charge leakage apparent in Flash memories [77]. PCM cells experience a phenomenon known as resistance drift and results in asymmetric errors over time. The work presented in this chapter extends the use of RM codes to combat the resistance drift in PCM. A novel hybrid RM (HRM) code is proposed to combat resistance drift in PCM with coding efficiency higher than FRM and PRM. If n cells are used for storing a single information word, the HRM splits up these n cells into two subgroups. The ρ highest resistance level cells forms a subgroup and these cells stores data based on their relative orders. In other words, the $(\rho, \rho-1)$ -FRM code is employed on these ρ PCM cells. The remaining $n-\rho$ cells represent an information word based on the uncoded scheme. The detailed explanation is provided later in Section 6.2. Lastly, the performance of the three types of RM codes is benchmarked against the uncoded PCM channel model in Section 6.3. The conclusion is drawn in Section 6.4.

As RM codes are a type of permutation codes, the size of the code increases

exponentially as described in Chapter 5. In the simulations executed to evaluate the performance of codes based on RM, the novel enumerative coding scheme proposed in Chapter 5 is used to analytically map each information word to its corresponding codeword.

6.1 Brief Summary of RM Codes

Since the detailed analysis of the RM scheme was given in Chapter 5, this section gives a brief review of the prior-art RM coding schemes as a prelude to the proposal of a HRM code. Consider n Flash memory cells, for $i = 1, 2, \dots, n$, the charge level of the i -th cell, which is commonly known as a cell level is denoted by c_i . For RM codes employed in Flash memories, the cell state levels are ranked in an ascending order, while it is suggested in this work that RM codes employed in PCM should be ranked in descending order. The reason will be explained after the definition of a “push-to-bottom” mechanism is presented. The ranks of the n cells are a permutation of $\{1, 2, \dots, n\}$. If the permutation is $[a_1, a_2, \dots, a_n]$ then the cell state levels are ranked according to the inequality in (6.1).

$$c_{a_1} < c_{a_2} < \dots < c_{a_n} \quad (6.1)$$

Cell a_1 is assumed to have the lowest resistance and cell a_n the highest resistance value. For any codes based on RM, no two cells can have the same resistance, which is always true in practical situations. In order to modify the relative order of the group of n cells to obtain a new permutation, “push-to-bottom” operations are performed on these PCM cells. The definition of a “push-to-bottom” mechanism is

presented in Definition 6.1.1. In general, the “push-to-bottom” operation sets the resistance value of a single cell to the lowest among the n cells.

Definition 6.1.1. Let set T be the set of “push-to-bottom” operations for n cells. Each “push-to-bottom” operation is defined by

$$t_i([a_1, \dots, a_{i-1}, a_i, a_{i+1}, \dots, a_n]) = [a_i, a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_n]. \quad (6.2)$$

Example 6.1.1. An example of a (3,2)-PRM, defined in Definition 5.2.1, code is given and the initial cell level state is $[1,2,3]$. The total set of transitions functions are listed below.

$$\begin{aligned} t_2[1,2,3] &= [2,1,3] \\ t_3[1,2,3] &= [3,1,2] \end{aligned} \quad (6.3)$$

A RM scheme uses the ranks induced by the n cell levels to represent data. Let S_n denote the set of permutations of $\{1,2,\dots,n\}$, and let q denote the size of an alphabet $\mathcal{Q} = \{0,1,\dots,q-1\}$. When the n cells store the data of alphabet size q , we have a decoding function

$$\mathcal{D}: S_n \rightarrow \mathcal{Q} \quad (6.4)$$

that maps every permutation to some value in the alphabet $\mathcal{Q}$.

The key to a successful “push-to-bottom” operation for PCM is the ability to reduce the value of the cell state such that it is just below the resistance level of all cells. As the name suggest, in order to push the cell state to the bottom of the n cells, a larger proportion of the PCM material has to be converted to the crystalline phase, achieved by performing set operations. Compared to the reset operation, the set

operation is less memory degrading and is preferred over the reset operation. In contrast, if a “push-to-top” operation was employed to enable intermediate resistance levels to different amorphous region, the memory degrading reset operation would be required whenever new information is stored.

Before proceeding to propose the new HRM code, the distinction between the work here and prior-art work is summarized. The first RM code suggested for Flash was proposed in [113], which focuses on the motivation and code construction. Other works investigated efficient data movement in Flash memory that minimizes the number of block erasures [114] and fast cell programming algorithms for Flash [115]. The above mentioned theoretical works did not consider a practical code design, where memory cells usually have a maximum number of intermediate levels due to manufacturing limitations. Therefore, a q -level PCM cell can only have RM codes with a maximum codeword length $n \leq q$; such that every cell are distinguishable (e.g. have a different cell state). An important measure missing from the related studies is the coding efficiency. Since the thesis focus is on data storage, codes designed must have a high coding efficiency (e.g. $\geq 90\%$).

6.2 Hybrid RM Codes to Combat Resistance Drift

6.2.1 Hybrid RM Codes

In this section, the novel HRM code is proposed to combat resistance drift in PCM with coding efficiency higher than FRM and PRM. Assume that n PCM cells are selected for storing a single information word, the HRM scheme splits up these

n cells into two subgroups. The ρ highest resistance level cells forms the first subgroup and these cells stores an information word m_1 according to their relative orders. In other words, the $(\rho, \rho-1)$ -FRM code is employed on these ρ PCM cells. The remaining $n-\rho$ cells represent another information word m_2 based on the uncoded scheme. Appending both the information words m_1 and m_2 together, the information word stored on the n PCM cells is illustrated by $m = m_1 m_2$. For easy reference throughout this thesis, this HRM code is denoted by (n, ρ) -HRM.

The PRM code on the other hand, only considers ρ out of n cells for data storage, while the remaining $n-\rho$ cells are ignored. Due to this characteristic of PRM codes, a significant amount of code rate is lost, which translates to low coding efficiency. In order to reduce redundancy, the HRM code leverages on the $n-\rho$ cells to store uncoded data. The full participation of all n cells for data storage results in a higher code rate and better storage efficiency. The improvement in storage efficiency over the FRM and PRM codes is provided in Section 6.2.2. Thereafter, the performance evaluation of the FRM, PRM and HRM codes are summarized in Section 6.3.

A scenario is provided to further illustrate the newly designed HRM code. For instance, a user has a total of eight PCM cells (*e.g.* $n = 8$) and the parameter ρ is assumed to take on the value 4. The details of the HRM codes are further explained using the eight PCM cells illustrated in Figure 6.1. From the figure, the four highest resistance PCM cells are represented by cell 5, 6, 7 and 8, and these four cells will participate in RM. Specifically, we have the cell state vector

$$c = (c_5, c_6, c_7, c_8) = (5, 7, 8, 6) \tag{6.5}$$

whereby the ranking of the cell states corresponds to the vector $\mathbf{r}' = (5, 8, 6, 7)$. Subsequently, vector $\mathbf{r}'$ is normalized by reducing it with the least cell index value (e.g. cell 5) to obtain the RM codeword $\mathbf{r} = (1, 4, 2, 3)$. Using the simplified enumerative coding scheme proposed in Section 5.2, the RM codeword $\mathbf{r}$ corresponds to the decimal digit 4 and information vector $i_r = (0, 1, 0, 0)$.

Figure 6.1: Cell states of 8 PCM cells with $q = 8$ distinct levels. The shaded boxes represent the current resistance state of each cell (e.g. cell 1 is at state “1” and cell 8 is at state 6).

For the remaining lowest $n - \rho = 4$ cells, there are four distinct resistance states for programming. Hence, by the conventional method of rewriting data without RM, two bits can be stored on each of these four cells (e.g. state 1 is mapped to “00”, state 2 is mapped to “01”, state 3 is mapped to “10” and state 4 is mapped to “11”). The lowest four cell states are represented by cell 1, 2, 3 and 4 and they have

cell state vector

$$\mathbf{c}' = (c_1, c_2, c_3, c_4) = (1, 2, 3, 4). \quad (6.6)$$

Without the implementation of any RM codes, the cell state vector corresponds to an information vector $i_c = (0, 0, 0, 1, 1, 0, 1, 1)$. Vectors i_c and i_r are then combined, yielding the corresponding information vector $i = (0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0)$.

6.2.2 Code Rate Analysis

In this section, a thorough code rate comparison is performed for the available variants of the RM codes, namely the FRM, PRM and HRM. The analysis is an important investigation to deduce the redundancy caused by implementation of these RM codes into practical PCM systems. Redundancy should be kept to its minimum in order to maximize the storage efficiency and the code rate achieved.

A. FRM Code Rate

The code rate analysis begins with the FRM. Let Q be represented with the set $\{1, 2, \dots, q\}$ and a multi-level PCM cell with q levels. A RM code C with length $n = q$ would have a codeword $c = (c_1, c_2, \dots, c_q)$, with the condition that each $c_i \in Q$ and $c_i \neq c_j$ for all $i \neq j$. For any RM code with length n , it can be easily validated that the total number of distinct messages is $n!$. Therefore the maximum number of bits stored in the n cells is the floor function of

$$M_c = \log_2(n!) \quad (6.7)$$

Without RM implementation, each q -level PCM cell is able to store $\log_2 q$ bits.

Therefore, the uncoded n PCM cells are able to store $n \times \log_2 q$ bits. The code rate for the full RM is

$$\mathcal{R}(C) = \frac{M_c}{n \times \log_2 q}.$$

(6.8)

We wish to add on that for a multi-level PCM cell with q levels, the maximum codeword length is q . Table 6.1 shows the code rate of the full RM codes with parameters $n = q$, where q is the number of distinct cell state levels in a MLC PCM cell, and $\rho = n - 1$. We remark that as the number of levels in each cell increases, the code rate improves as well. In other words, there is less redundancy for cells with more levels, which translates to better storage efficiency.

Table 6.1: Code rate of FRM codes for $q = 4, 8, 16, 32$.

No. of Levels in Each Cell q	$(n, n - 1)$ -PRM	$\mathcal{R}(C)$
4	(4,3)	0.5731
8	(8,7)	0.6375
16	(16,15)	0.6914
32	(32,31)	0.7354

B. PRM Code Rate

For an (n, ρ) -PRM code, the maximum number of bits stored is calculated by taking the floor function of

$$M_p = \log_2 \left[\binom{n}{\rho} \rho! \right].$$

(6.9)

The code rate computation for PRM code is similar to the case for the FRM.

$$\mathcal{R}(C) = \frac{M_p}{n \times \log_2 q} \quad (6.10)$$

Table 6.2 shows the code rate of multi-level PCM cells for the PRM codes with parameters $n = q = 4, 8, 16, 32$ and $\rho = n/2$. Akin to the FRM, as the number of levels in each cell increases, the code rate improves as well. In other words, there is less redundancy for cells with more levels, which translates to better storage efficiency. For all q , we observed that the code rate for the (n, ρ) -PRM code is less than the $(n, n-1)$ -PRM, which is commonly known as the FRM code. Although the PRM code provides a less complex decoder contributed from the reduction in the number of sorting operations required during decoding, the storage efficiency is greatly reduced.

Table 6.2: Code rate of PRM codes for $q = 4, 8, 16, 32$ and $\rho = n/2$.

No. of Levels in Each Cell q	$(n, n-1)$ -PRM	$\mathcal{R}(C)$
4	(4,2)	0.4481
8	(8,4)	0.4464
16	(16,8)	0.4524
32	(32,16)	0.4588

C. HRM Code Rate

Finally, we present the code rate for the hybrid RM. For an (n, ρ) -HRM code, the maximum number of bits stored is calculated by taking the floor function of

$$M_h = \log_2(\rho!) + (n - \rho) \cdot \log_2(n - \rho). \quad (6.11)$$

The code rate for the (n, ρ) -HRM code is determined by Equation (6.12). As the prior-art RM codes were designed without consideration for a practical code construction [74, 77], the derivation of the code rate for the FRM and PRM is an overestimate. For example, considering an $(8,4)$ -PRM code, the maximum number of bits allowed to be stored on the 8 cells are $\lfloor M_p \rfloor = \lfloor \log_2 \binom{8}{4} 4! \rfloor = 30$. For the FRM and PRM codes, this floor function is not accounted in the code rate computation provided in (6.8) and (6.10). For the proposed HRM codes, practical code designs have been considered and hence the code rate for an (n, ρ) -HRM is

$$\mathcal{R}(C) = \frac{\lfloor M_h \rfloor}{n \times \log_2 q}. \quad (6.12)$$

Table 6.3 shows the code rate of multi-level PCM cells for the HRM codes with parameters $n = q = 4, 8, 16, 32$ and $\rho = n - 2$. Similar to the prior-art RM codes (e.g. FRM and PRM), the number of levels in each cell is proportional to the code rate. Therefore the cells with more levels achieve better storage efficiency.

It is consistent from the code rate analysis for the RM codes that an increase in the number of levels available in the PCM cell will translate to a higher code rate to yield better storage efficiency. On the other hand, the number of levels cannot be increased infinitely due to practical considerations. For the PCM channel performance evaluation, the optimal number of levels available in each PCM cell is selected to be 16.

Table 6.3: Code rate of HRM codes for $q = 4, 8, 16, 32$ and $\rho = n/2$.

No. of Levels in Each Cell q	(n, ρ) -PRM	$\mathcal{R}(C)$
4	(4,2)	0.3750
8	(8,4)	0.5000
16	(16,8)	0.6094
32	(32,16)	0.6750

6.3 Performance Evaluation of Various RM Codes

Before analyzing the RM performance, it is necessary to state the assumptions and parameters that define the characteristics of the PCM cell’s behavior. For all channel simulations carried out in this work, the adaptive threshold detection method is employed to verify the resistance cell state of the PCM cells. The motivation for employing the adaptive threshold detector has been provided in this section.

6.3.1 Simulation Setup

From the PCM channel model discussed in Chapter 2, equation (2.7) and (2.8) showed that the PCM resistance drift characteristics is determined by the noise parameters v and β . The GST material was evaluated by [31] and through experimental studies, it was confirmed that $v_{max} = 0.11$. Hence, throughout all the simulations carried out in this work, the parameter v_{max} is kept as 0.11 and the value of β is calculated from Figure 6.2, whereby the data points are obtained from [31]. The value of the noise parameter β for PCM cells was accessed in [31] using

the Blahut-Arimoto algorithm presented in [33]. Figure 6.2 shows the noise parameter β as a function of the first measured logarithmic resistance R_{rv} . The triangle data points are experimental data extracted from [31], while the dotted line is the best fit line of the general quadratic function $y = c_1x^2 + c_2x + c_3$, where variable $x = \log_{10} R_{rv}$ and variable $y = \beta$. Using the fit function available on gnuplot [40], the best fit line is expressed with Equation (6.13).

$$y = -0.0004x^2 + 0.0045x - 0.0121. \quad (6.13)$$

From the code rate analysis, it was concluded that channel simulations were carried out with PCM cells that contains 16 distinct levels. The minimum and maximum resistances for a PCM cell is assumed to be $10^4 \Omega$ and $10^8 \Omega$ respectively, and is divided into 16 equally spaced intervals that represent a unique cell state level. The cells are initially programmed with its desired resistance level with a 3% variance. It is because the cost to ensure that the resistance level of every cell in each state has exactly the same initial resistance is too high; hence the inclusion of a variance in programming the initial resistance of a PCM cell is meant to provide a more realistic channel model. The initial resistance is represented by R_0 and the first measured resistance is denoted by R_{rv} for each cell state is tabulated in Table 6.4. From the analysis of resistance drift in Section 2.2.3, cells with higher resistance have a larger proportion of amorphous material and hence, will result in greater resistance drift over time. Therefore, the increment from R_0 to R_{rv} is the least in set 1 and the most in set 16. The simulation settings are further summarized in Table 6.4.

Figure 6.2: Noise parameter β over the first measured logarithmic resistance R_{rv} .

Table 6.4: Simulation settings for PCM analytical channel model.

Cell State	R_0	R_{rv}	v	β
1	13335	13340	0.003437	-0.00034
2	23714	23723	0.010313	-6.9E-05
3	42170	42182	0.017188	0.000156
4	74989	75007	0.024062	0.000331
5	133352	133387	0.030937	0.000456
6	237137	237195	0.037812	0.000531
7	421697	421784	0.044688	0.000556

8	749894	750014	0.051562	0.000531
9	1333521	1333688	0.058437	0.000456
10	2371374	2371599	0.065313	0.000331
11	4216965	4217256	0.072187	0.000156
12	7498942	7499305	0.079062	-6.9E-05
13	13335214	13335665	0.085937	-0.00034
14	23713737	23714281	0.092812	-0.00067
15	42169650	42170302	0.099687	-0.00104
16	74989421	74990231	0.106563	-0.00147

These simulations provide further insights on how the RM code impacts the performance of the PCM. Investigations are carried out varying the length of the code and the number of cells used for the RM coding scheme. These simulations are benchmarked against the PCM channel model without implementing the RM codes.

For an accurate BER performance evaluation for the RM codes, the adaptive threshold detection scheme is employed in the PCM channel simulations. The following paragraphs compare the advantage of an adaptive threshold detection scheme over a fixed threshold detector. Theoretical works [105] have also validated that the adaptive thresholds detection schemes have outperformed the fixed thresholds methods for both binary and non-binary scenarios. An illustration is provided in Figure 6.3, where a single-level cell contains two distinguishable states – bit “0” and “1”. Initially, both cell states have equivalent variance, but after a period of time T seconds, the mean of bit “0” is shifted to the right and its variance is increased. With the shift in mean resistance of the cell states and increased variance, a higher probability of error is accounted for when the fixed threshold ξ_1 is applied after T seconds has elapsed from the first time the cell state is verified. By employing the adaptive threshold detection scheme, the fixed threshold ξ_1 can be

adjusted to a new threshold level ξ_2 to minimize the probability of error. This statement is proven in the following paragraphs.

The theoretical analysis for the adaptive threshold detection is compared with the fixed threshold detection for a single-level PCM cell (e.g. $q = 2$). Assume the fixed threshold level is ξ_1 and the cell state of each distinct level is denoted by x_1 and x_2 . Initially the cell states have a Gaussian distribution that are $x_1 \sim N(\mu_1, \sigma^2)$ and $x_2 \sim N(\mu_2, \sigma^2)$. From Figure 6.2, we observed that an error occurs at the overlapping regions of the Gaussian distributed curves. Therefore, the probability of error, after a period of t seconds, is calculated as follows.

$$\begin{aligned}
 \Pr(\text{error}|\xi_1) &= \Pr(x_1 > \mu_1 + \xi_1 \cup x_2 \leq \mu_2 - \xi_1) \\
 &= \Pr(x_1 > \mu_1 + \xi_1) + \Pr(x_2 \leq \mu_2 - \xi_1) \\
 &= \Pr\left(Z > \frac{\xi_1}{\sigma}\right) + \Pr\left(Z \leq -\frac{\xi_1}{\sigma}\right) \\
 &= 2 \left[1 - \Phi\left(\frac{\xi_1}{\sigma}\right)\right]
 \end{aligned} \tag{6.13}$$

where the function $\Phi(x)$ is the c.d.f. for a zero-mean, unit-variance Gaussian distribution.

Let the error probability be $\Pr(\text{error}|\xi_2)$ when the adaptive threshold detector is applied. Whenever $x_1 < x_2$, the threshold is set to be between x_1 and x_2 . Thus no errors are contributed in this case. However errors are observed for all $x_1 \geq x_2$. The probability of error for the adaptive threshold detector is summarized in the equation below.

$$\begin{aligned}
 \Pr(\text{error} | \xi_2) &= \Pr(x_1 \geq x_2) \\
 &= \int_{-\infty}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(z - (\mu - \xi_2))^2}{2\sigma^2}\right] \Phi\left[\frac{z - (\mu - \xi_2)}{\sigma}\right] dz \quad (6.14)
 \end{aligned}$$

It is observed from Figure 6.4 that the error rate for the adaptive threshold detection is significantly smaller than the fixed threshold detection. The previous derivation of error probabilities explains the advantage of adaptive thresholds. When using fixed thresholds, a cell can be in error simply by having its charge level deviate enough to leave its initial threshold interval. With adaptive thresholds, cells have to deviate relative to each other to cause an error. In other words, two cells' voltage levels must cross over before an error can take place, which has a lower probability. Since the adaptive threshold detector provides a higher memory sensing accuracy as compared to the fixed threshold detector, the PCM channel simulations carried out in this thesis employed the adaptive threshold detector.

Figure 6.3: Resistance level distribution of two distinct PCM cell states. For time (a) $t = 0$ sec and (b) $t = T$ sec.

In Chapter 5, a novel enumerative coding scheme is proposed in order to compute the lexicographic index of the RM code analytically rather than obtaining it through a brute-force enumeration and hence simplifying the complexity of the encoder and decoder. Due to the availability of an analytical computation method, the need for a memory-consuming table look-up is not required. In this study, the enumerative scheme is included in the channel simulations and the BER of the PCM system is investigated.

Figure 6.4: Error rate for two cells x_1 and x_2 as a function of σ for fixed threshold detection against adaptive threshold detection, where $\xi_1 = 1/2$.

A block diagram of the PCM channel simulation is shown in Figure 6.5 and is used for analyzing the various RM codes performance. The study mainly focuses on three variants of RM codes; namely FRM, PRM and HRM codes. To keep the BER analysis of the RM coded systems clear and concise, thermal crosstalk is not considered in the study.

Figure 6.5: Block diagram of RM coded PCM communication channel.

In the following subsection, the evaluation begins with the search for a HRM coded system with the best channel performance. Thereafter, a comparison is carried

out, whereby the HRM system is compared with prior-art FRM and PRM codes. For fair comparison, the code rate is fixed for all RM codes.

6.3.2 Investigation of Varying Parameter ρ

First, the codeword length of the HRM code is fixed at $n = 16$ while varying the parameter ρ that represents the number of PCM cells used for the RM scheme. As mentioned during the simulation setup, each PCM cell has 16 distinct levels. When the number of levels in the cell and the codeword length are the same (e.g. $n = q = 16$), the RM code loses its ability to rewrite new information, through the “push-to-bottom” operation. As a RM code requires all cells to be at a unique state, each of the cell states available on the 16-level PCM cell corresponds to either one of the n cells. If a “push-to-bottom” operation is executed, there is no unique cell state available to convert either of the n cells such that all of them will be unique. However, sacrificing the ability to rewrite information is suggested in this thesis so that code rate loss can be kept to the minimum. Although the capability of rewriting is impacted, the ability to combat against resistance drift is still intact, which is the goal for implementing HRM codes in this work.

The evaluation of the proposed HRM is initially investigated with the highest number of cells begin used for ranking, which is $\rho = 14$ (e.g. by setting $\rho = 15$ is essentially the FRM code). In Figure 6.6, curve 6 depicts this particular HRM code (e.g. (16,14)-HRM code). Channel simulations were subsequently carried out with decreasing values of $\rho = 10, 12$ (depicted by curves 4 and 5 in Figure 6.6) and an improvement in the PCM channel performance is observed. However, further

decrease of parameter $\rho = 4, 6, 8$ resulted in a decrease in channel performance (depicted by curves 1 to 3 in Figure 6.6). Therefore, with a given amount of drift experienced by the PCM cells, an optimal number of cells should be selected for ranking.

Figure 6.6: Simulation result from varying parameter ρ .

The code rate of each HRM code is examined to fully understand the trend observed in the PCM channel simulations for varying ρ . We plot the code rate functions of the FRM, PRM and HRM for codeword length $n = 16$ in Figure 6.7 and observed the following trend. From Table 6.4, as ρ decreases from 14 to 10 the code rate is reduced as well. The increased code rate loss implies that redundancy is increased. Further decrease in the value of ρ results in an increase in code rate instead. This is in contrast to the prior trend for the higher values of ρ . Due to this trend in the code rates of the HRM codes, a similar trend is observed in the channel

performance of the HRM coded system. The channel performance improvement is compared with the uncoded system and is summarized in Table 6.5. The improvement is calculated with a reference point on the x-axis, where the uncoded system intersects the x-axis at $x = 5.75$. For a 16 level PCM cell, the optimal number of cells used for ranking is 10.

Table 6.5: Improvement comparison between RM codes with varying ρ .

Curve	n	ρ	Code rate	PCM Channel Improvement
1	16	4	0.6250	40.0%
2	16	6	0.6094	43.5%
3	16	8	0.6094	47.8%
4	16	10	0.5156	86.1%
5	16	12	0.5625	78.3%
6	16	14	0.5938	69.6%

From the investigation above, it is concluded that with a given amount of resistance drift determined by the PCM cell characteristics, there is an optimal number of cells required for ranking to obtain the best PCM channel performance. This trend is in sync with the code rate loss, where the best performance channel has the greatest code rate loss (curve 4) and the worse performance channel has the least code rate loss (curve 1). The code rates achieved by the FRM, PRM and HRM with codeword length $n = 16$ are illustrated in Figure 6.7. The dotted lines are prior-art RM codes, while the solid lines are the code rate achieved by the HRM proposed in this work. From the graph, code rates of PRM codes decreases as n increases, while the HRM codes, similar to the FRM, has its code rates increasing together with an

increase in codeword length n . For $\rho = 4$, the larger codeword length (e.g. $n \geq 13$) is able to achieve code rates higher than any FRM or PRM codes with $n = 16$.

Figure 6.7: Code rate analysis of FRM, PRM and HRM with length $n = 16$.

6.3.3 Investigation of Varying Codeword Length

After obtaining the optimal ρ for a 16 level PCM cell, the next step is to optimize the length of the codeword n . Figure 6.8 presents the result of investigating

the impact of varying the length n of a RM code. Each of the curves depicts the transition from an uncoded PCM communication channel to a RM coded PCM system with decreasing code rate. The performance for the uncoded system has shown to be significantly poorer than the RM coded system. From the simulation results, it is noted that as the length of the codeword increases, the PCM channel performance improves. The improvement is summarized in Table 6.6. From the investigation above, the HRM code that achieves the best channel performance is the (16,10)-HRM code. An inference can hence be drawn, where given an amount of resistance drift controlled by the PCM cell characteristics; an optimal value of ρ can be deduced such that the channel performance is maximized.

Figure 6.8: Simulation result from varying parameter n .

Table 6.6: Improvement comparison between RM codes with varying n .

Curve	n	ρ	PCM Channel Improvement
1	16	10	86.1%
2	15	10	79.1%
3	14	10	72.2%
4	13	10	68.7%

6.3.4 Investigation of Varying RM Codes

In this section, the BER obtained from the various RM coded channel simulations are analyzed. The proposed (16,10)-HRM is compared with prior-art FRM and PRM codes. For a fair comparison, the code rates of these various RM codes are kept approximately equal. The uncoded system is used as a benchmark for the BER improvement of the RM coded system. The BER result for the investigation of the various RM codes is illustrated in Figure 6.9. Curve 3 depicts the FRM scheme with $n = 13$. Taking the reference point from the x-axis (e.g. $BER = 0.0001$), the uncoded system intersects the x-axis at $x = 5.75$ while the full RM system intersects at approximately $x = 9.90$. The performance improvement for the FRM is around 72.2%. For code rates to be approximately equal, the (16,9)-PRM code is selected. Curve 2 intersects the x-axis at $x = 10.50$ resulting in an improvement of 82.6%. Lastly, the (16,10)-HRM code is illustrated by curve 1 and an improvement of 86.1% is achieved. This result is summarized in Table 6.7 for easy reference. The comparison has concluded that with approximately the same code rate sacrificed, the HRM is able to provide a better channel performance gain as compared to its counterparts (e.g. FRM and PRM).

Figure 6.9: BER channel performance of FRM, PRM and HRM keeping their code rate approximately consistent.

Table 6.7: Performance improvement of RM codes benchmarked against the uncoded system.

Type of RM codes	$\mathcal{R}(C)$	Performance Improvement
FRM	0.508	72.2%
PRM	0.543	82.6%
HRM	0.516	86.1%

6.4 Conclusion

In this chapter, we have analyzed the bit error rates of three different types of RM schemes – FRM, PRM and HRM. From the performance evaluation of RM coded PCM communication channel, for a given amount of drift experienced by a PCM cell, there is an optimal number of cells that are required to be ranked (*e.g.* search for the optimal value of ρ). Upon obtaining the value of ρ , channel simulations varying the codeword length n are executed to search for the optimized (n, ρ) -HRM code that results in the best channel performance for a given PCM resistance drift characteristics. To ensure that a fair comparison is carried out, the code rates of FRM, PRM and HRM are kept approximately the same. With the same amount of redundancy, it is observed that the HRM achieved an improvement of 86.1% over the uncoded systems, while the FRM and PRM each achieved an improvement of 72.2% and 82.6% respectively. Therefore, the HRM code is able to improve the PCM channel performance of the FRM coded system by 13.9% and the PRM coded system by 3.5%.

The RM scheme was suggested in [77] as a scheme to effectively combat resistance drift phenomenon present in PCM systems by leveraging on the relative order of the PCM cells for data storage. However, it is essential to note that errors will still appear if the relative ranks of the cells change due to non-uniform resistance drift. When such an error occurs, the only method is to employ the ECC to help correct the erroneous data bits to the corrected data bits. The work in this thesis can serve as a starting point for future research of ECC in PCM systems.

CHAPTER 7

CONCLUSION OF THESIS

7.1 Conclusion

In this thesis, coding strategies targeted to improve the lifetime and memory sensing accuracy of non-volatile memory (NVM) devices are presented. The write-once memory (WOM) codes were targeted for lifetime enhancement in NVM while the rank modulation (RM) codes were a focus on improving memory sensing accuracy especially in multi-level NVM devices. First, two-write WOM-codes that were proposed in this thesis have achieved WOM-rates higher than the state-of-the-art WOM-codes, resulting in better storage efficiency. This is a particularly difficult task as prior-art codes have achieved coding efficiency of 92.32% and 86.75% for the unrestricted and restricted WOM-codes, respectively. Conducting brute-force search operations would be a futile method to achieve near capacity improvements in the WOM-rates. Instead, from the analysis of the code construction, this thesis has proposed a criteria criterion to design capacity approaching WOM-codes and further improve the code rate of the unrestricted and restricted WOM-rate by 2% and 1%, respectively.

For the research on improving the memory sensing accuracy, a new variant of the RM code was proposed to combat the asymmetric errors due to resistance drift in phase change memory (PCM) cells. PCM channel simulations were carried out where different variants of RM codes were evaluated. Benchmarking the proposed RM code against existing RM codes have also been conducted. To our knowledge, the work done in this thesis is the first thorough and comprehensive study of efficient storage codes for PCM devices.

The merits and the contributions of this thesis are summarized below.

1. Proposed novel criterion to design WOM-codes which achieves storage efficiencies higher than the state-of-the-art codes (Chapter 3 and 4).
 - From the analysis of the code construction method, the WOM-rate is maximized when the selected base code has a large hamming distance. Based on this rationale, binary BCH codes were selected as the base code to construct binary WOM-codes.
 - The work is further extended to cover two-write non-binary WOM-codes and subsequently showed that high-rate non-binary WOM-codes can also be constructed from these efficient binary WOM-codes.
 - The best known WOM-codes from literature were constructed by means of a computer search that does not have a practical encoder/decoder. On the other hand, WOM-codes designed using the proposed criterion have not only achieved an efficiency improvement

of 2% for the binary codes and 1% for the non-binary codes, they also have a practical encoder/ decoder available in literature.

2. Development of a novel enumerative coding scheme that computes the lexicographic index of the RM codewords analytically (Chapter 5).

- Instead of the brute-force enumeration, the novel enumerative coding scheme was proposed to replace the memory consuming table look-up and analytically compute the lexicographic index of a RM codeword.
- Investigation work on the computational complexity of the conventional enumerative scheme and the proposed novel enumerative scheme customized for RM codes was carried out. Analysis showed that the proposed novel scheme is able to reduce both the encoder and decoder's computational complexity by two orders of magnitude each.

3. Designed a novel hybrid RM (HRM) code to combat asymmetric errors contributed by resistance drift from PCM cells (Chapter 6).

- With a given amount of resistance drift that is determined by the PCM cell properties, an optimal number of cells should be selected for ranking such that the storage efficiency and channel performance are maximized.

- From the code rate analysis, the proposed HRM codes demonstrated higher code rates as compared to the partial RM (PRM) codes that have the same parameters, which then translate to better storage efficiency. Details on the code rate improvement for different multi-level PCM cells are given in the Table 7.1. For PCM cells with four distinct levels, the (4,2)-HRM code has exhibit a lower achieved code rate as compared to the (4,2)-PRM. This is because the prior-art PRM codes were designed without consideration for practical code designs. Therefore, the code rate computation for the PRM codes is an overestimate, while the code rate computation for the HRM codes has considerations for practical code designs.

Table 7.1: Comparison of code rate improvement for the HRM against the PRM codes.

# of distinct levels	(n, ρ)	Code rate		Code rate improvement
		Partial RM	Hybrid RM	
4	(4,2)	0.4481	0.3750	-16.3%
8	(8,4)	0.4464	0.5000	12.0%
16	(16,8)	0.4524	0.6094	34.7%
32	(32,16)	0.4588	0.6750	47.1%

4. Performance analysis of bit error rates (BER) of various codes based on RM with PCM channel model (Chapter 6).
- Computer simulations validate the performance of the three types of RM schemes, with the HRM being the best performing among the

three. The performance improvement of the three various RM codes are summarized in Table 7.2.

Table 7.2: Performance evaluation for various RM codes benchmarking against the uncoded PCM system.

Type of RM codes	$\mathcal{R}(C)$	Performance Improvement
FRM	0.508	72.2%
PRM	0.543	82.6%
HRM	0.516	86.1%

7.2 Suggestions for Future Work

The work in this thesis shows that efficient storage coding techniques can overcome the various key issues faced by PCM devices. In particular, the WOM-codes were targeted to extend the lifetime of these memory cells whereas the RM codes focus on improving memory sensing accuracy in PCM. The latter is hindered by the undesirable resistance drift phenomenon. The two families of storage codes can certainly do with more explorations such as the following.

- 1. Development of efficient WOM-codes with more than two-writes capabilities.

WOM-codes with higher rewritability relates to a longer lifetime for PCM devices. In this thesis, two-write WOM-codes were studied as higher write WOM-codes will result in larger code rate lost, which cause degradation of memory storage efficiency. The best known rates for WOM-codes with more than two writes are summarized in Table 7.3. For these WOM-codes with

greater rewritability, there is a 20% to 40% redundancy while the two-write WOM-codes achieved in this work have approximately 5% redundancy. An efficient code construction method for the high-rates multiple write WOM-codes is able to achieve an efficiency of approximately 95% or more is certainly desirable for data storage.

Table 7.3: Efficiency comparison for best known WOM-codes with more than two writes.

No. of Writes	Best Known WOM-rates [109]	Capacity	Efficiency (%)
3	1.6102	2.0000	80.51
4	1.8566	2.3219	79.96
5	1.9689	2.5850	76.17
6	2.1331	2.8074	75.98
7	2.1723	3.0000	72.41
8	2.2544	3.1699	70.98
9	2.2918	3.3219	68.99
10	2.3466	3.4594	67.83

2. Combination of prior-art ECC with WOM and RM codes.

The RM scheme was suggested as a scheme to efficiently combat resistance drift phenomenon present in PCM systems by leveraging on the relative order of the PCM cells for data storage. However, it is important to note that errors will still appear if the relative ranks of the cells change due to non-uniform resistance drift. When such error occurs, deployment of the ECC is the only approach to help correct the erroneous data bits. The newly designed storage codes proposed in this thesis can be combined with prior-art ECC to guarantee lifetime enhancement while providing error correction capabilities. However the key to successful ECC for codes based on RM

requires a well defined function to determine the distance between two distinct RM codewords, and must be studied as a future work. This research work will be a starting point to further develop coding schemes to provide ECC capabilities with lifetime enhancement and accurate memory sensing advantage.

3. Development of a channel model for thermal crosstalk [110]

For improvement in storage densities, the size of PCM cells continues to scale down, which brings about new challenges. Therefore, the development of coding techniques to overcome these problems is essential for the success of PCM technologies in future practical applications. A critical issue known as thermal crosstalk that is contributed from intercellular disturbance has been widely recognized. There has been no appropriate analytical channel model proposed in literature up till now. To ensure a more realistic analytical mathematical channel model, the thermal crosstalk phenomenon should be taken into consideration during modeling.

4. Design of coding and detection techniques to combat both resistance drift and thermal crosstalk in PCM.

There has not yet been any report on an appropriate channel coding or detection schemes to tackle both the resistance drift and thermal crosstalk issues in PCM. In this thesis, the HRM scheme was proposed as an efficient detection technique to combat resistance drift. An efficient finite-state encoding methods to design capacity-approaching constrained codes to

combat thermal crosstalk has been recently reported [111]. A possible future work would be the analysis of merging both coding techniques and subsequently propose a single novel scheme that has the capability to combat issues arising from the resistance drift and thermal crosstalk.

PUBLICATIONS ORIGINATED FROM THIS THESIS

Based on the work reported in this thesis, the author has contributed the following publications.

1. M. Chua, M. R. Elidrissi, K. Eason, S. H. Zhang, Z. L. Qin, K. S. Chai, K. P. Tan, K. Cai, K. S. Chan, W. L. Goh, Y. Dong and R. H. Victora, "Comparing Analytical, Micromagnetic and Statistical Channel Models at 4 Tcbpsi Patterned Media Recording," IEEE Transactions on Magnetics, vol. 48, no. 5, pp. 1826-1832, May 2012.
2. Maria Yu Lin, Kheong Sann Chan, Melissa Chua, Songhua Zhang, Cai Kui, and Moulay Rachid Elidrissi, "Modeling for write synchronization in bit patterned media recording," Journal of Applied Physics, vol. 111, issue 7, pp. 07B918 - 07B918-3, Mar 2012.
3. M. W. J. Chua, K. Cai and W. L. Goh, "Two-write WOM-codes for Non-volatile Memories," International Symposium on Intelligent Signal Processing and Communications Systems (ISPACS), 2012, vol., no., pp.710-715, 4-7 Nov 2012.
4. M. Y. Lin, M. R. Elidrissi, K. S. Chan; K. Eason, M. Chua, M. Asbahi, J. K. W. Yang, N. Thiagarajah and V. Ng, "Channel Characterization and Performance Evaluation of Bit-Patterned Media," IEEE Transactions on Magnetics, vol.49, no.2, pp.723-729, Feb 2013.
5. M. W. J. Chua, K. Cai and W. L. Goh, "Enumerative Coding for Rank Modulation," submitted to IEEE Transactions on Information Theory.
6. M. W. J. Chua, K. Cai and W. L. Goh, "Efficient Two-write WOM-codes for Non-volatile Memories," submitted to IEEE Communications Letters.
7. M. W. J. Chua, K. Cai and W. L. Goh, "Hybrid Rank Modulation for Phase Change Memories," to be submitted to Transactions on Information Theory.

BIBLIOGRAPHY

- [1] Databeans Inc., "Flash Memory Market to Reach \$51.2 Billion by 2015," Market Research Report 2010.
- [2] Apple Inc. (1994) Apple QuickTake100. User's Guide.
- [3] Sony Electronics Inc. (August, 2010) DSLR-A900. Marketing Specifications.
- [4] Aritome, S.; Shirota, R.; Hemink, G.; Endoh, T.; Masuoka, F., "Reliability issues of flash memory cells," in the Proceedings of the IEEE, vol. 81, no. 5, pp. 776-788, May 1993.
- [5] Brewer JE, Gill M. Nonvolatile Memory Technologies with Emphasis on Flash: A Comprehensive Guide to Understanding and Using Flash Memory Devices. Hoboken: Wiley; 2008. p. 750.
- [6] Meena, Jagan Singh et al. "Overview of Emerging Nonvolatile Memory Technologies." Nanoscale Research Letters 9.1 (2014): 526. PMC. Web. 27 Sept. 2015.
- [7] Daniele Ielmini and Mattia Boniardi, "Common signature of many -body thermal excitation in structural relaxation and crystallization of chalcogenide glasses", Applied Physics Letters, vol. 94, no. 9, Mar 2009.
- [8] Wei Xu and Tong Zhang, "Using time-aware memory sensing to address resistance drift issue in multi-level phase change memory," Quality Electronic Design (ISQED), 2010 11th International Symposium on, pp.356-361, 22-24 Mar 2010.
- [9] Boniardi Mattia and Ielmini Daniele, "Physical origin of the resistance drift exponent in amorphous phase change materials," Applied Physics Letters,

- hr/>
- vol.98, no.24, pp.243506-243506-3, Jun 2011.
- [10] R. Bez, E. Camerlanghi, A. Modelli, and A. Visconti, "Introduction to flash memory," *Proc. of the IEEE*, vol. 91, no. 4, pp. 489–502, Apr. 2003.
- [11] Pirovano, A. L. Lacaita, F. Pellizzer, S. A. Kostylev, A. Benvenuti, and R. Bez, "Low -field amorphous state resistance and threshold voltage drift in chalcogenide materials," *IEEE Transactions on Electron Devices*, vol. 51, pp. 714– 719, May 2004.
- [12] A. Jiang, "On the generalization of error-correcting WOM codes," *Proc. IEEE Int. Symp. Inform. Theory*, pp. 1391–1395, Nice, France, June 2007.
- [13] T. Berger, F. Jelinek, and J.K. Wolf, "Permutation codes for sources," *IEEE Trans. Inform. Theory*, vol. 18, no. 1, pp. 160–168, January 1972.
- [14] You Yin and Sumio Hosaka, "Controllable Crystallization in Phase-change Memory for Low-Power Multilevel Storage," *Japanese Journal of Applied Physics*, vol. 51, no. 6, pp. 064101-1–064101-4, Jun 2012.
- [15] J.H. Conway and N.J. Sloane, "Orbit and coset analysis of the Golay and related codes," *IEEE Trans. Inform. Theory*, vol. 36, no. 5, pp. 1038–1050, September 1990.
- [16] J. C. Slonczewski, "Current-Driven Excitation of Magnetic Multilayers," *Journal of Magnetism and Magnetic Materials*, vol. 159, no. 1-2, pp. L1–L7, 1996.
- [17] M. Lenzlinger and E.H. Snow. "Fowler-Nordheim tunneling into thermally grown SiO₂," *Electron Devices, IEEE Transactions on*, 15(9):686-686, Sep 1968.

-
- [18] B. Gleixner, F. Pellizzer, R. Bez, "Reliability characterization of Phase Change Memory," Non-Volatile Memory Technology Symposium (NVMTS), 2009 10th Annual, pp.7-11, Oct 2009.
 - [19] R. L. Rivest and A. Shamir, "How to reuse a write -once memory," Information and Control, vol. 55, no. 1-3, pp. 1– 19, Dec 1982.
 - [20] G. Cohen, P. Godlewski, F. Merks, "Linear binary code for write-once memories (Corresp.)," IEEE Transactions on Information Theory, vol.32, no.5, pp. 697- 700, Sep 1986.
 - [21] G. Zemor and G. D. Cohen, "Error -correcting wom-codes," IEEE Trans. Information Theory, vol. 37, pp. 730-734, May 1991.
 - [22] F. Fu and A. J. H. Vinck, "On the capacity of generalized write-once memory with state transitions described by an arbitrary directed acyclic graph," IEEE Trans. Information Theory , vol. 45, no. 1, pp. 308-313, Sep 1999.
 - [23] E. Yaakobi, S. Kayser, P. H. Siegel, A. Vardy, and J. K. Wolf, "Efficient two -write wom-codes," IEEE Information Theory Workshop (ITW), pp. 1-5, Aug 2010.
 - [24] C. Heegard, "On the capacity of permanent memory," IEEE Trans. Information Theory, vol. 30, pp. 470-480, Sep 1984.
 - [25] R. Gabrys, E. Yaakobi, L. Dolecek, P. H. Siegel, A. Vardy, J. . Wolf, "Non - binary WOM-codes for multilevel flash memories," IEEE Trans. Information Theory Workshop (ITW) , pp. 40-44, Oct 2011.
 - [26] A. Jiang, R. Mateescu, M. Schwartz, J. Bruck, "Rank Modulation for Flash

- Memories,” IEEE Int. Symp. Information Theory, Toronto, Canda, pp. 1166–1170, 2008.
- [27] A. Fiat and A. Shamir, “Generalized write -once memories,” IEEE Trans. Information Theory, vol. 30, pp. 470–480, Sep 1984.
- [28] M. Grossi, M. Lanzoni, and B. Ricco, “Programming schemes for multilevel flash memories,” IEEE Proceedings, vol. 91, no. 4, pp. 594–601, Apr 2003.
- [29] T. M. Cover, “Enumerative source encoding,” IEEE Transactions on Information Theory, vol. 19, no. 1, pp. 73–77, Jan. 1973.
- [30] A. Jiang and J. Bruck, “Joint coding for flash memory storage,” Proc. IEEE International Symposium on Inform. Theory, pp. 1741–1745, Jul 2008.
- [31] Franceschini M., Lastras-Montano L. A., Jagmohan A., Sharma M., Cheek R., Lee M.-H., “A Communication-Theoretic Approach to Phase Change Storage,” IEEE International Conference on Communications (ICC) 2010, pp.1-6, 23-27 May 2010.
- [32] N. Papandreou, H. Pozidis, T. Mittelholzer, G. F. Close, M. Breitwisch, C. Lam, and E. Eleftheriou, “Drift -tolerant multilevel phase-change memory,” IEEE International Memory Workshop, pp. 1–4, May 2011.
- [33] R. Blahut, “Computation of channel capacity and rate-distortion functions,” IEEE Transactions on Information Theory, vol. 18, no. 4, pp. 460–473, 1972.
- [34] S. Kayser, E. Yaakobi, P. H. Siegel, A. Vardy, and J. . Wolf, “Multiple write wom-codes,” 48th Annual Allerton Conference on Communication, Control, and Computing, pp. 1062–1068, Feb 2011.
- [35] E. Yaakobi, P. H. Siegel, A. Vardy, and J. . Wolf, “Multiple error correcting

- wom-codes,” IEEE Trans. Information Theory, vol. 58, no. 4, pp. 2220–2230, Apr 2012.
- [36] S. Kayser, E. Yaakobi, P. H. Siegel, A. Vardy, and J. K. Wolf, “Multiple-write WOM-codes,” Proc. 48-th Annual Allerton Conference on Communication, Control and Computing, Monticello, IL, September 2010.
- [37] S. Lin and D. J. Costello, “Error control coding,” Prentice Hall, May 2004.
- [38] S. Ling and C. Xing, “Coding theory: A first course,” Cambridge University Press, Mar 2004.
- [39] G. Zemor and G. D. Cohen, “Error-correcting wom-codes,” IEEE Trans. Information Theory, vol. 37, no. 3, pp. 730–734, May 1991.
- [40] Gnuplot homepage [internet] [updated Jan 2015; cited Jan 2015].
Available from: <http://www.gnuplot.info/>
- [41] D. E. Knuth, “Efficient balanced codes,” IEEE Trans. Inform. Theory, vol. 32, no. 1, pp. 51–53, January 1986.
- [42] Panasonic. The world’s first mass production of FeRAM-embedded system-on-a-chip (SoC) using a 0.18 μ m processing technology. Technical report, Matsushita Electric Industrial Co., Ltd., Jul 2003.
- [43] G. W. Burr, B. N. Kurdi, J. C. Scott, C. H. Lam, K. Gopalakrishnan, and R. S. Shenoy. Overview of candidate device technologies for storage class memory. IBM Journal of Research and Development, 52(4/5):449-464, 2008.
- [44] M. Hosomi, H. Yamagishi, T. Yamamoto, K. Bessho, Y. Higo, K. Yamane, H. Yamada, M. Shoji, H. Hachino, C. Fukumoto, H. Nagao, and H. Kano. A novel non-volatile memory with spin torque transfer magnetization switching:

- spin-ram. In IEDM Technical Digest, pages 459-462, 2005.
- [45] S. Lai and T. Lowrey. OUM – a 180 nm nonvolatile memory cell element technology for stand alone and embedded applications. In IEDM Technical Digest, pages 3651-3654. IEEE, 2001.
- [46] Y. N. Hwang, J. S. Hong, S. H. Lee, S. J. Ahn, G. T. Jeong, G. H. Koh, J. H. Oh, H. J. Kim, W. C. Jeong, S. Y. Lee, J. H. Park, K. C. Ryoo, H. Horii, Y. H. Ha, and J. H. Yi. Full integration and reliability evaluation of phase-change ram based on 0. 24μm — cmos technologies. In Symposium on VLSI Technology, pages 173-174, 2003.
- [47] Y. C. Chen, C. T. Rettner, S. Raoux, G. W. Burr, S. H. Chen, R. M. Shelby, M. Salinga, W. P. Risk, T. D. Happ, G. M. McClelland, M. Breitwisch, A. Schrott, J. B. Philipp, M. H. Lee, R. Cheek, T. Nirschl, M. Lamorey, C. F. Chen, E. Joseph, S. Zaidi, B. Yee, H. L. Lung, R. Bergmann, and C. Lam. Ultra-thin phase-change bridge memory device using gesb. In IEDM Technical Digest, page S30P3, 2006.
- [48] T. Nirschl, J. B. Philipp, T. D. Happ, G. W. Burr, B. Rajendran, M.-H. Lee, A. Schrott, M. Yang, M. Breitwisch, C.-F. Chen, E. Joseph, M. Lamorey, R. Cheek, S.-H. Chen, S. Zaidi, S. Raoux, Y. C. Chen, Y. Zhu, R. Bergmann, H.-L. Lung, and C. Lam. Write strategies for 2 and 4-bit multi-level phase-change memory. In IEDM Technical Digest, page 17.5, 2007.
- [49] A. Pirovano, A. L. Lacaita, F. Pellizzer, S. A. Kostylev, A. Benvenuti, and R. Bez, "Low-field amorphous state resistance and threshold voltage drift in chalcogenide materials," IEEE Trans. Electron Devices, vol. 51, pp. 714-719,

2004.

- [50] A. L. Lacaita, A. Redaelli, D. Ielmini, F. Pellizzer, A. Pirovano, A. Benvenuti, and R. Bez, "Electrothermal and phase-change dynamics in chalcogenide-based memories," in IEDM Tech. Dig., 2004, pp. 911-914.
- [51] D. Krebs, S. Raoux, C. T. Rettner, G. W. Burr, M. Salinga, and M. Wuttig, "Threshold field of phase change memory materials measured using phase change bridge devices," Appl. Phys. Lett., vol. 95, p. 082101, Aug 2009.
- [52] D. Yu, S. Brittman, J. S. Lee, A. L. Falk, and H. Park, "Minimum voltage for threshold switching in nanoscale phase-change memory," Nano Lett., vol. 8, pp. 3429-3433, Oct 2008.
- [53] A. E. Owen, J. M. Robertson, and C. Main, "The threshold characteristics of chalcogenide-glass memory switches," J. Non-Cryst. Solid, vol. 32, pp. 29-52, 1979.
- [54] D. Adler, H. K. Henisch, and S. D. Mott, "The mechanism of threshold switching in amorphous alloys," Rev. Mod. Phys., vol. 50, pp. 209-220, 1978.
- [55] A. Pirovano, A. L. Lacaita, A. Benvenuti, F. Pellizzer, and R. Bez, "Electronic switching in phase-change memories," IEEE Trans. Electron Devices, vol. 51, pp. 452-459, 2004.
- [56] D. Adler, M. S. Shur, M. Silver, and S. R. Ovshinsky, "Threshold switching in chalcogenide glass thin films," J. Appl. Phys., vol. 51, pp. 3289-3309, 1980.
- [57] D. Adler, M. S. Shur, M. Silver, and S. R. Ovshinsky, "Reply to Comment

- on "Threshold switching in chalcogenide glass thin films", J. Appl. Phys., vol. 56, pp. 579-580, 1984.
- [58] D. Ielmini, "Threshold switching mechanism by high-field energy gain in the hopping transport of chalcogenide glasses," Phys. Rev. B, vol. 78, p. 035308, 2008.
- [59] V. G. Karpov, Y. A. Kryukov, S. D. Savransky, and I. V. Karpov, "Nucleation switching in phase change memory," Appl. Phys. Lett., vol. 90, pp. 123504-123504-3, 2007.
- [60] S. Raoux, H.-Y. Cheng, J. L. Jordan-Sweet, B. Munoz, and M. Hitzbleck, "Influence of interfaces and doping on the crystallization temperature of Ge-Sb," Appl. Phys. Lett., vol. 94, pp. 183114-3, 2009.
- [61] S. Raoux, J. L. Jordan-Sweet, and A. J. Kellock, "Crystallization properties of ultrathin phase change films," J. Appl. Phys., vol. 103, pp. 114310-114310-7, 2008.
- [62] S.-H. Lee, Y. Jung, and R. Agarwal, "Highly scalable non-volatile and ultra-low-power phase-change nanowire memory," Nat. Nanotechnol., vol. 2, pp. 626-630, 2007.
- [63] S.-H. Lee, Y. Jung, and R. Agarwal, "Size-dependent surface-induced heterogeneous nucleation driven phase-change in Ge₂Sb₂Te₅ nanowires," Nano Lett., vol. 8, pp. 3303-3309, 2008.
- [64] D. Fugazza, D. Ielmini, S. Lavizzari, and A. L. Lacaita, "Distributed-Poole-Frenkel modeling of anomalous resistance scaling and fluctuations in phase-change memory (PCM) devices," in IEDM Tech. Dig., 2009, pp. 617-620.

-
- [65] M. A. Caldwell, S. Raoux, R. Y. Wang, H. S. P. Wong, and D. J. Milliron, "Synthesis and size-dependent crystallization of colloidal germanium telluride nanoparticles," *Journal of Materials Chemistry*, vol. 20, pp. 1285-1291, 2010.
- [66] D. Ielmini, A. L. Lacaita, and D. Mantegazza, "Recovery and drift dynamics of resistance and threshold voltages in phase-change memories," *IEEE Transactions on Electron Devices*, vol. 54, pp. 308–315, Feb 2007.
- [67] D. Ielmini, D. Sharma, S. Lavizzari, and A. L. Lacaita, "Reliability Impact of Chalcogenide -Structure Relaxation in Phase-Change Memory (PCM) Cells – Part I: Experimental Study," *IEEE Transactions on Electron Devices*, vol. 56, no. 5, pp. 1070-1077, May 2009.
- [68] D. Ielmini, A. L. Lacaita, and D. Mantegazza, "Recovery and Drift Dynamics of Resistance and Threshold Voltages in Phase-Change Memories," *vol. 54, no. 2*, pp. 308-315, Feb 2007.
- [69] Y. Wu and A. Jiang, "Position modulation code for rewriting write -once memories," *IEEE Trans. Information Theory*, vol. 57, no. 6, pp. 3692–3697, Jun 2011.
- [70] R. Bez, E. Camerlenghi, A. Modelli, A. Visconti, "Introduction to flash memory," *Proceedings of the IEEE*, vol. 91, no. 4, pp. 489–502, Apr 2003.
- [71] R. Gabrys, E. Yaakobi, L. Dolecek, P. H. Siegel, A. Vardy, and J. K. Wolf, "Non-binary wom-codes for multilevel flash memories," *Information Theory Workshop (ITW)*, pp. 40–44, Oct. 2011.
- [72] A. Jiang, M. Schwartz, and J. Bruck, "Error-correcting codes for rank

- modulation,” IEEE International Symposium on Information Theory, pp. 1736–1740, July 2008.
- [73] F. Zhang, H. Pfister, and A. Jiang, “Ldpc codes for rank modulation in flash memories,” IEEE International Symposium on Information Theory, pp. 859–863, June 2010.
- [74] Z. Wang and J. Bruck, “Partial rank modulation for flash memories,” IEEE International Symposium on Information Theory, pp. 864–868, June 2010.
- [75] H. Nobukata, S. Takagi, K. Hiraga, T. Ohgishi, M. Miyashita, K. Kamimura, S. Hiramatsu, K. Sakai, T. Ishida, H. Arakawa, M. Itoh, I. Naiki, and M. Noda, “A 144 -mb, eight-level nand flash memory with optimized pulsewidth programming,” IEEE Journal of Solid-State Circuits , vol. 35, no. 5, pp. 682– 690, May 2000.
- [76] D. Slepian, “Permutation modulation,” in Proc. of the IEEE , vol. 53, no. 3, 1965, pp. 228–236
- [77] A. Jiang, R. Mateescu, M. Schwartz, and J. Bruck, “Rank modulation for flash memories,” IEEE Trans. on Inform. Theory, vol. 55, no. 6, pp. 2659–2673, Jun. 2009.
- [78] G. W. Burr and et al. Phase Change Memory Technology, 2010.
<http://arxiv.org/abs/1001.1164v1>.
- [79] C. Bracken and T. Helleseht, “Triple-error-correcting BCH-like codes,” in Proc. IEEE Int.Symp. Inform. Theory, pp. 1723–1725, Seoul, Korea, June 2009.

-
- [80] P. Godlewski, "WOM-codes construits `a partir des codes de Hamming," *Discrete Math.*, no. 65 pp. 237–243, 1987.
 - [81] A. Jiang, V. Bohossian, and J. Bruck, "Floating codes for joint information storage in write asymmetric memories," in *Proc. IEEE Int. Symp. Inform. Theory*, pp. 1166–1170, Nice, France, June 2007.
 - [82] A. Jiang and J. Bruck, "Joint coding for flash memory storage," in *Proc. IEEE Int. Symp. Inform. Theory*, pp. 1741–1745, Toronto, Canada, July 2008.
 - [83] T. Kasami, "The weight enumerators for several classes of subcodes of the second order binary Reed-Muller codes," *Inform. and Control*, vol. 18, pp. 369–394, September 1971.
 - [84] H. Mahdavifar, P.H. Siegel, A.Vardy, J.K.Wolf, and E.Yaakobi, "A nearly optimal construction of flash codes," in *Proc. IEEE Int. Symp. Inform. Theory*, pp. 1239–1243, Seoul, Korea, July 2009.
 - [85] F. Merks, "Womcodes constructed with projective geometries," *Traitement du signal*, vol. 1, no. 2-2, pp. 227–231, 1984.
 - [86] J.K.Wolf, A.D.Wyner, J. Ziv, and J.Korner, "Coding for a write-once memory," *AT&T Bell Labs. Tech. J.*, vol. 63, no. 6, pp. 1089–1112, 1984.
 - [87] Y.Wu, "Low complexity codes for writing write-once memory twice," in *Proc. IEEE Int. Symp. Inform. Theory*, pp. 1928–1932, Austin, Texas, June 2010.
 - [88] R. Brent, S. Gao, and A. Lauder, "Random Krylov spaces over finite fields," *SIAM J. Discrete Math*, vol. 16, no. 2, pp. 276–287, February 2003.

-
- [89] J.H. Conway and N.J. Sloane, "Orbit and coset analysis of the Golay and related codes," *IEEE Trans. Inform. Theory*, vol. 36, no. 5, pp. 1038–1050, September 1990.
 - [90] R.M. Roth, *Introduction to Coding Theory*, Cambridge University Press, 2005.
 - [91] J.P.M. Schalkwijk, "An algorithm for source coding," *IEEE Trans. Inform. Theory*, vol. 18, no. 3, pp. 395–399, May 1972.
 - [92] E.Yaakobi, J. Ma, L. Grupp, P.H. Siegel, S. Swanson, and J.K.Wolf, "Error characterization and coding schemes for flash memories," *Proc. Workshop on the Application of Communication Theory to Emerging Memory Technologies*, Miami, Florida, December 2010.
 - [93] E. Yaakobi and A. Shpilka, "High sum-rate three-write and non-binary WOM codes," in *Proc. of the IEEE International Symposium on Information Theory*, Cambridge, MA, Jul. 2012, pp. 1386–1390.
 - [94] A. Pirovano and A. Redaelli, "Reliability study of phase-change non-volatile memories," *IEEE Transactions on Device and Materials Reliability*, vol. 4, no. 3, pp. 422–427, Sept. 2004.
 - [95] A. Jiang, H. Zhou, Z. Wang, and J. Bruck, "Patterned cells for phase change memories," in *Proc. of the IEEE International Symposium on Information Theory*, St. Petersburg, Russia, Jul.-Aug. 2011, pp. 2333–2337.
 - [96] Q. Huang, S. Lin, Abdel-Ghaffar, and K.A.S., "Error-correcting codes for flash coding," *IEEE Trans. Inform. Theory*, vol. 57, no. 9, pp. 6097–6108, Sept. 2011.

-
- [97] F. Farnoud, V. Skachek, and O. Milenkovic, "Rank modulation for translocation error correction," in *Proc. of the IEEE International Symposium on Information Theory*, Cambridge, MA, Jul. 2012, pp. 2988–2992.
 - [98] L. Grupp, A. Caulfield, J. Coburn, S. Swanson, E. Yaakobi, P. Siegel, and J. Wolf, "Characterizing Flash memory: anomalies, observations, and applications," in *Proc. of the IEEE/ACM International Symposium on Microarchitecture*, New York, NY, Dec. 2009, pp. 24–33.
 - [99] N. Elarief and B. Bose, "Optimal, systematic, q -ary codes correcting all symmetric and symmetric errors of limited magnitude," *IEEE Transactions on Information Theory*, vol. 56, no. 3, pp. 979–984, Mar. 2010.
 - [100] Y. Cassuto, M. Schwartz, V. Bohossian, and J. Bruck, "Codes for asymmetric limited-magnitude errors with application to multi-level flash memories," *IEEE Transactions on Information Theory*, vol. 56, no. 4, pp. 1582–1596, Apr. 2010.
 - [101] A. Barg and A. Mazumdar, "Codes in permutations and error correction for rank modulation," *IEEE Transactions on Information Theory*, vol. 56, no. 7, pp. 3158–3165, Jul. 2010.
 - [102] S. Hudgens and B. Johnson, "Overview of Phase -Change Chalcogenide Nonvolatile Memory Technology," retrieved from http://www.engr.sjsu.edu/sgleixner/mate270/LectureNotes/Hudgens_MRS.pdf.

-
- [103] A. Berman and Y. Birk, "Constrained flash memory programming," in *Proc. of the IEEE International Symposium on Information Theory*, St. Petersburg, Russia, Jul.-Aug. 2011, pp. 2128–2132.
 - [104] S. Augustinovich and S. Kitaev, "On uniquely k -determined permutations," *Discrete Mathematics*, vol. 308, no. 9, pp. 1500–1507, Sept. 2008.
 - [105] H. Zhou, A. Jiang, and J. Bruck, "Error-correcting schemes with dynamic thresholds in non-volatile memories," in *Proc. of the IEEE International Symposium on Information Theory*, St. Petersburg, Russia, Jul.-Aug. 2011, pp. 2143–2147.
 - [106] A. Yelon, B. Movaghar, and H. M. Branz, "Origin and Consequences of the compensation (Meyer -Neldel) law," *Physics Rev. B*, vol. 46, no. 19, Nov 1999.
 - [107] A. Pirovano, A. L. Lacaita, F. Pellizzer, S. A. ostylev, A. Benvenuti, and R. Bez, "Low -field Amorphous State Resistance and Threshold Voltage Drift in Chalcogenide Materials," *IEEE Transactions on Electron Devices*, vol. 51, no. 5, pp. 714-719, May 2004.
 - [108] D. Ielmini, S. Lavizzari, D. Sharma, A. L. Lacaita, "Temperature Acceleration of Structural Relaxation in Amorphous Ge₂ Sb₂ Te₅," *Applied Physics Letters*, vol. 9, no. 19, May 2008.
 - [109] E. Yaakobi and A. Shpilka, "High Sum-Rate Three-Write and Nonbinary WOM Codes," *IEEE Transactions on Information Theory*, vol. 60, no. 11, pp. 7006-7015, Nov 2014.

-
- [110] U. Russo, D. Ielmin, A. Redaelli, and AL. Lacaita, "Modeling of programming and read performance in phase-change memories-Part IT: program disturb and mixed-scaling approach," IEEE Transactions on Electron Devices, vol. 55, no. 2, pp. 515-522, Feb 2008.
- [111] Kui Cai, "Vertical constrained coding for phase-change memory with thermal crosstalk," International Conference on Computing, Networking and Communications (ICNC) 2014, vol., no., pp.312-316, 3-6 Feb 2014.
- [112] W. Xu and T. Zhang, "Using time-aware memory sensing to address resistance drift issue in multi-level phase change memory," International Symposium on Quality Electronic Design, pp. 356– 361, Mar 2010.
- [113] Ventrice, P. Fantini, A. Redaelli, A. Pirovano, A. Benvenuti, F. Pellizzer, "A Phase Change Memory Compact Model for Multilevel Applications," IEEE Electron Device Letters, vol. 28, no. 11, pp. 973-9750, Nov 2007.
- [114] Sonoda et al., "A Compact Model of Phase -Change Memory Based on Rate Equations of Crystallization and Amorphization," IEEE Transactions on Electron Devices, vol. 55, no. 7, pp. 1660 -1672, Jul 2008.
- [115] L. A. Lastras-Montano, M. Franceschini, T. Mittelholzer, J. aridis, and M. Wegman, "On the lifetime of multilevel memories," IEEE International Symposium on Information Theory, 009. ISIT 2009., pp. 1224– 1228, Aug 2009.

-
- [116] S. Friedberg, A. Insel, and L. Spence, "Linear algebra," Prentice Hall, Nov 2002.
- [117] K. Koughia, Z. Shakoor, S. O. asap, and J. M. Marshall, "Density of localized electronic states in a -Se from electron time-of-flight photocurrent measurements," Journal Applied Physics, vol. 97, no. 3, Jan 2005.
- [118] S. Raou and M. Wuttig, "Phase Change Materials: Science and Applications," Springer Verlag, 2009.
- [119] R. A. Cobley and C. D. Wright, "Parameterized Spice Model of Phase - Change RAM Device," IEEE Transactions on Electron Devices, vol. 53, no. 1, pp. 112-117, Jan 2006.