Modeling and Signal Processing Techniques For Flash Memory



Name :	Hachem Yassine
College :	Lincoln
Supervisor :	Justin P. Coon
Department :	Engineering Science

A thesis submitted to the University of Oxford in accordance with the requirements of the degree of Doctor of Philosophy in the Department of Engineering Science.

148 pages

Modeling and Signal Processing Techniques For Flash Memory

Hachem Yassine

October, 2018

Modeling and Signal Processing Techniques For Flash Memory

Name: Hachem Yassine

College: Lincoln

Degree: Doctor of Philosophy

Term and Year of Submission: Hilary 2018

This thesis examines the effects of noise and interference on the performance of NAND flash memory.

Chapter 3 studies the probabilistic input/output relation between the data stored and the read threshold voltage of a cell and generalizes it to a group of cells. It is then concluded that adjacent cells are correlated due to common aggressors. This motivates the study of adequate signal processing techniques to optimize the reliability performance.

Chapter 4 proposes two techniques that can reduce the error rate in light of the result of the previous chapter. The first is based on dividing a group of cells into sub-groups and detecting each sub-group independently. The second approximates the flash system model by a hidden Markov model, then uses the sum-product algorithm to detect the inputs. The soft outputs of the proposed detectors are passed on to the ECC soft decoder. It is shown that the second approach provides significant improvements. Then, it is shown that quantization negatively affects the performance of the sum-product algorithm more in comparison with the first approach. To partially mitigate this effect, an iterative detection/decoding strategy is proposed and shown to improve the performance.

Chapter 5 proposes a novel data representation scheme that provides a trade-off between reliability and the amount of data stored per cell, and partially mitigates the effects of device degradation. The scheme divides the stored data into two streams stored in the indices and levels of the non-erased cells, respectively, allowing the first to be detected without any knowledge about the channel. The simulation results show an improvement in the error rate while partially mitigating the need to track the channel parameters and the read references as the device degrade.

($<$ 300 words)

Acknowledgements

I would like to thank first and foremost my supervisor, Professor Justin Coon, for allowing me to join his group, helping me formulate and solve my research problems, and providing a constant moral support. Throughout the years, I have learned a lot from him and enriched my skill set.

I would also like to thank Dr Mihai-Alin Badiu for inspiring and guiding my work on factor graphs and for being there whenever I needed him. I would also like to thank Dr David Simmons for helping me present my index programming in a much better way.

I would also like to thank the directors of Toshiba Telecommunications Research Laboratory in Bristol, for putting their faith in me and generously funding my DPhil. I would also like to thank Dr. Mohamed Ismail and Dr. Amr Ismail from the laboratory for closely following my work and always providing their advice and support.

I would also like to thank Steve Elston and Stephen Morris for being on my first and third years' assessment committee and for the constructive feedback they provided.

Beyond academic matters, I have been very fortunate to meet a lot of great people in Oxford whom I am grateful to know. First, my office and group colleagues, both current and past. Our academic and non-academic discussions had an immense

effect on challenging and shaping my ideas. Second, the beautiful people I met in Oxford who became close friends and made Oxford feel like home. Finally, the random people in shops and on the streets whom never fading smiles enlightened every day.

My mother, father and sisters have been there at every step, unconditionally guiding and protecting me. Without them, none of this would have been possible. I am grateful to have them in my life.

Lastly, to my wife, Samah, goes my most sincere gratitude for believing in me and supporting me throughout the years. I love you so much.

Acronyms

V_{th}	Threshold Voltage
NVS	Non-Volatile Storage
SLC	Single-Level Cell
MLC	Multi-Level Cell
TLC	Triple-Level Cell
ICI	Inter-Cell Interference
ECC	Error Correcting Code
WL	Word-Line
BL	Bit-Line
FPS	Full Program Sequencing
SPS	Shadow Program Sequencing
BER	Bit Error Rate
HDD	Hard Disk Drive
SSD	Solid State Drive
ISPP	Incremental Step Pulse Programming
MSB	Most Significant bit
LSB	Least Significant Bit
RTN	Random Telegraph Noise
AEBNR	Average Energy per Bit to Noise Ratio

ABL	All Bit-Line
EOBL	Even Odd Bit-Line
PDF	Probability Density Function
CDF	Cumulative Distribution Function
PMF	Probability Mass function
i.i.d	Independent and indentically distributed
i.u.d	Independent and uniformly distributed

Notation

Scalar random variables are represented using upper-case non-boldface letters, and their realization using lower case non-boldface letters. Random vectors are represented using upper-case boldface letters and their realization using lower-case boldface letters. Sets are generally represented using calligraphic notation. $\mathbb{R}$ is the set of reals and $\mathbb{Z}_q = \{0, ..., q-1\}$.

Below, is a list of algebraic notation used in the thesis[1]

X	Input of a cell
Y	Output of a cell
S	ICI-free shift
W	ICI
v_x	Nominal voltage when the input is x
σ_x^2	Noise variance when the input is x
$\Pr(\cdot)$	Probability of an event
$\mathcal{U}(a,b)$	Uniform PDF between a and b

[1]Symbols may contain (extra) subscripts/superscripts denoting, e.g., cell index.

$p_X(x)$	PDF (PMF) of X
$P_X(x)$	CDF of X
$\mathbb{E}[\cdot]$	expectation operator
$H_2(\cdot)$	Binary entropy function
$I(\cdot;\cdot)$	Mutual information
γ_v	Vertical coupling
γ_d	Diagonal coupling
Γ	Coupling matrix
n_{RTN}	Random Telegraph noise term
n_{Ret}	Retention noise term
n_E	Erase noise term
n_P	Program noise term
$\mathbf{C}$	Constant weight codeword
$\mathbf{A}$	Vector of amplitudes
$\epsilon_{i,j}$	Transition probability from level j to i
Δ	ISPP step size
$Q(\cdot)$	Quantizer function
$\mathcal{N}(x\|\mu,\sigma)$	Normal PDF
β	Noise power control
α	Diagonal to vertical coupling ratio
ζ	Fraction of programmed cells

Manuscripts Since Beginning of DPhil

Below is a complete list in reverse chronological order of manuscripts that have been published, are currently under review, or are currently being prepared for submission.

Being prepared	**Hachem Yassine**, Mihai-Alin Badiu and Justin P. Coon. "MAP Detector for Flash Memory Without Accessing the Interfering Cells". Transactions on Communications.
Published	**Yassine, Hachem**, Justin P. Coon, and David E. Simmons. "Index programming for flash memory." IEEE Transactions on Communications 65.5 (2017): 1886-1898.
Published	**Yassine, Hachem**, et al. "Towards an analytical model of NAND flash memory and the impact on channel decoding." Communications (ICC), 2016 IEEE International Conference on. IEEE, 2016.

List of Figures

2.1 Schematic of a floating gate transistor. 10

2.2 Gray mapping of the levels to data bits for MLC. 11

2.3 Schematic of a NAND flash block. 12

2.4 Diagram showing the ISPP process. 14

2.5 The PDFs of the major sources of noises, $\Delta = 0.3$, $\lambda = 0.1$ and $\xi_k = 0.05$ 17

2.6 Two step programming for MLC. 21

2.7 Full program sequencing for MLC using ABL architecture. 21

2.8 Shadow program sequencing for MLC using ABL architecture. 22

2.9 The raw bit error rate of SLC and MLC as a function of the number of P/E cycles for different vendors [1]. 24

2.10 The bit error rate of SLC as a function of the number of P/E cycles when ECCs are considered [1]. 25

3.1 Plot of the excess kurtosis of the V_{th} for an erased and programmed cell. 33

3.2 The common aggressors (in light blue) to two adjacent victim cells (in red). 38

3.3 Normalized mutual information of MLC ($q = 4$) for $\beta = 1$ as a function of vertical coupling γ_v and four different α. (a) $n = 1$, (b) $n = 2$ and (c) $n = 3$. 43

3.4 Normalized mutual information for $\beta = 0.5$ as a function of vertical coupling γ_v and four different α. (a) $n = 1$, (b) $n = 2$ and (c) $n = 3$. . 45

4.1 Coded and uncoded BER using of the naive approach for $n' = 1$ (blue) and $n' = 2$ (red) as a function of γ_v and for different β and α. . 53

4.2 Factor graph representing the ABL structure with FPS. 57

4.3 Coded and uncoded BER using of the naive approach for $n' = 1$ (blue) and $n' = 2$ (red) and of the proposed approach as a function of γ_v and for different β and α. 64

4.4 The quantization boundaries are shown in black when $\beta = 1$, $\gamma_v = 0.08$ and $\alpha = 0.075$. 66

4.5 Coded and uncoded BER using of the naive approach for $n' = 1$ (blue) and $n' = 2$ (red) and of the proposed approach when the output is quantized as a function of γ_v and for different β and α. 67

4.6 Coded BER using the sum-product with and without the iterative detection/decoding strategy. $MaxIn$ is the maximum number of iterations that the decoder performs and $MaxOut$ is the maximum number of times the detection/decoding process is done. 69

5.1 Diagram showing the index programming process 78

5.2 Plot of the asymptotic storage efficiency of IP (eq.(5.4)) as a function of ζ for different number of levels. 81

5.3 The distributions of ICI in the proposed IP schemes for different values of ζ and $q = 4$, $\zeta = 0.75$ corresponds also to ICI in UP. 84

5.4 Diagram showing the ICI prone pattern where the middle cell is susceptible to high ICI shift. 85
5.5 The stream/page error rates as functions of AEBNR, where solid lines represent the theoretical plots and markers their simulated counterparts, for $\zeta = 0.75$ and $\eta = 1.99957$ bits/cell. In this case, the UP:LSB and UP:MSB plots are coincident. The Amp. Stream (dynamic) and Amp. Stream (fixed) plots are also coincident. 97
5.6 The stream/page error rates as functions of AEBNR, where solid lines represent the theoretical plots and markers their simulated counterpart, for $v_0 = 1.5$, $\zeta = 0.75$ and $\eta = 1.99957$ bits/cell. 100
5.7 The stream/page error rates as functions of AEBNR, where solid lines represent the theoretical plots and markers their simulated counterparts, for $\zeta = 0.65$ and $\eta = 1.964$ bits/cell,. In this case, the UP:LSB and UP:MSB plots are coincident. The Amp. Stream (dynamic) and Amp. Stream (fixed) plots are also coincident. 101
5.8 The stream/page error rates as functions of AEBNR for $\zeta = 0.5$ and $\eta = 1.79$ bits/cell, where a dashed or dot-dashed line represent the theoretical plot, while marker of the same color represent its simulated counterpart (the simulated plots are omitted from the legend due to lack of space). 102
5.9 Diagram showing the states and read references for $q = 5$. 103
5.10 Diagram showing the alternative index programming process. 105
5.11 The page error rates as functions of AEBNR the for alternative IP with $k = 0.5n$ and $\eta = 1.9995$ bits/cell (dashed for theoretical and markers for simulated). 106

5.12 The bit error rates as functions of AEBNR for alternative IP with $k = 0.5n$ and $\eta = 1.9995$ bits/cell. 107

B.1 Examples of factor graphs. 119

List of Tables

3.1 Moments of the input-dependent equivalent noise. 32

3.2 Input and noise parameters of the flash device. 42

3.3 Mutual Information at $\gamma_v = 0.2$ and for $\beta = 1$. 44

3.4 Mutual Information at $\gamma_v = 0.2$ and for $\beta = 0.5$. 46

5.1 Performance gain of IP at 10^{-2} 102

Contents

Abstract i

Acknowledgements ii

Acronyms iv

Notation vi

Manuscripts Since Beginning of DPhil viii

1 Introduction **1**

1.1 Reliability Issues 2

1.2 Thesis Outline and Contributions 5

2 Background and Literature Review **9**

2.1 Flash Cell 10

2.2 Planar NAND Flash Memory 11

2.3 NAND Flash Memory Operations 12

2.3.1 Read 12

2.3.2 Erase 13

2.3.3 Program 14

2.4 Reliability Issues of Flash Memory 15

2.4.1 Noise 15

2.4.2 Inter-cell Interference 18

2.4.3 Channel Model 19
2.5 Program Architecture and Sequencing 19
2.6 Technology and Research Trends 23
2.7 Signal Processing for Flash Memory 25

3 Analytical Model of Flash Memory in the Presence of Interference 28
3.1 Introduction 29
3.2 Equivalent Noise Distribution 31
3.3 The V_{th} Distribution of a Single Cell 34
3.4 Joint Conditional Distribution 37
3.5 Mutual Information Between the Input and Output of Flash Channel 41
3.6 Summary 46

4 Inference in the Presence of ICI 48
4.1 Introduction 49
4.2 Problem Statement 51
4.3 Naive Approach 51
4.4 Graphical Model for Flash Memory 55
4.5 Computing the Posterior PMFs 58
4.6 Approximation of the Graphical Model 61
4.7 Performance of the Sum-Product 63
4.8 Effect of Quantization on Data Inference 64
4.8.1 Quantization 65
4.8.2 Performance Under Quantization 67
4.8.3 Iterative Detection/Decoding Strategy 68
4.9 Summary 71

5 Index Programming for Flash Memory **73**

5.1 Introduction 74

5.2 Index Programming 77

5.2.1 Definition 77

5.2.2 Illustrative Examples 79

5.2.3 Storage Efficiency 80

5.3 ICI Reduction Under IP 82

5.3.1 General Interference Model 82

5.3.2 Bit-line Interference Model 85

5.4 Error Analysis of IP 86

5.4.1 Read References 87

5.4.2 Detection of UP 87

5.4.3 IP Detection Using the Fixed References Scheme 91

5.4.4 IP Detection Using the Hybrid References Scheme 92

5.5 Simulations 95

5.5.1 Average Energy per Bit to Noise Ratio 95

5.5.2 Results 96

5.6 Binary Amplitude Pages in IP 102

5.7 Alternative Index Programming 105

5.8 Conclusion 108

6 Conclusion and Future Work **110**

A Random Vectors and Mutual Information **116**

B Factor Graphs and the Sum-Product Algorithm **118**

Bibliography **121**

Chapter 1

Introduction

The modern paradigms of machine learning and artificial intelligence were made possible only through the abundance of computational power and available data. As we realize the untapped value and potential in the data that we generate, it is becoming increasingly important to figure out how are we going to store and manage it. To put things into perspective, it is estimated that by 2020, 12000 terabytes of data will be generated per second [2], while currently, the largest commercial storage device has a capacity of 100 terabytes [3].

The ideal non-volatile storage (NVS) medium, i.e., where data persists when powered off, is fast, reliable, power efficient, cheap and dense in capacity. However, these requirements are yet to be fulfilled simultaneously, and over the years, many technologies have been invented, each having its advantages and setbacks that stimulated further research. In particular, magnetic hard disk drives (HDDs) are cheap and offer a high bit density, and have been popular for mass storage applications, e.g., personal computers and data centers. However, their major limitation is the existence of mechanical parts that can easily break, and limit their access speed [4].

Over the years, computational power grew consistently as predicted by Moore's

law, and data storage became the bottleneck of computation. Consequently, several NVS technologies, as well as computer architectures, were proposed to bridge the gap between the speed of computation and the speed of data access, e.g., caching. In 1987, Toshiba Corporation developed a new semiconductor-based technology called NAND flash memory [5]. Being purely electric, NAND flash provided, compared to HDDs, more reliability, lower power consumption, scalability due to Moore's law, and allowed faster access. In the next 30 years, the technology evolved tremendously to become a standard technology found in a wide variety of applications ranging from consumer electronics to primary memory, e.g., music players, digital cameras, smartphones, personal computers. An attestation to the success of flash memory is that in 2003, the cost of 128 megabytes of flash memory was USD 33, while in 2017, the cost of 128 gigabytes was USD 28, i.e., the megabyte got 1200 times cheaper [6].

The aggressive cost-per-bit reduction was made possible thanks to several innovations in the industry. A crucial step was mastering the techniques to store in each unit, called a cell, a larger number of bits compared to a single bit in older devices. In addition to this, the steady decrease in the size and cost of semiconductor chips helped make flash economically viable. Today, the prices of flash-based storage continue to decrease, and solid state drives (SSDs), which are flash-based, are the strongest competitor to replace HDDs for mass storage applications. However, from the outset, it was always anticipated that as the journey advances, the industry would flounder due to reliability issues.

1.1 Reliability Issues

The first step of flash storage is to place an electric charge into a well-insulated cell such that it is retained for a long time. However, due to imperfections, random variations in the charge are expected. As cells become smaller, the number of

electrons it can hold decreases, and the effect of the random variation in the charge becomes a significant reliability threat. Furthermore, by storing more than a single bit per cell, the distinction between empty and full cells is replaced by a distinction between different charge levels, which also increases the chances of errors. Finally, as the device gets used, it is regularly subject to high electric stress that results in severe degradation of its charge retention characteristics, resulting in data errors.

The second step is to efficiently organize as many cells as possible in a small space and to have a high throughput when data is stored and read. The efficient organization becomes difficult as the device scale down due to the amplification of adverse effects such as circuit noise, interference between cells, writing and reading speed, etc [7].

The aggregate effect of the reliability issues usually manifests itself by the device endurance, i.e., the number of times the device can be erased and programmed before reliable storage is not guaranteed anymore. The endurance of flash memory fell from 10^5 erase/write cycles in the early generations that stored a single bit/cell to a few hundred in the state-of-the-art memories storing three bits/cell [7]. This comes at a time where SSDs aim to replace HDDs for data-intensive applications such as data centers where data is erased and written much more frequently than in the case of personal devices.

To turn the struggle of flash into an opportunity, the entrepreneurial spirit is always set to invent a better technology, e.g., DNA storage, phase change memories, resistive memories, etc. However, the most significant innovation came from within the flash industry itself, where in 2009, Toshiba presented their 3D flash memory [8]. Instead of decreasing the bit-cost by shrinking the cell size, relatively large cells were sucessfully stacked in a 3D grid after being confined to a 2D grid in planar NAND. The new technology is already in commercial products, but it is still expensive and

not fully mastered.

On the other hand, system engineers acknowledge that imperfections are inherent in any storage medium and try to mitigate them by adding a layer of system level techniques. At the abstract level, data storage is data communication taking place in time rather than space. Therefore, the rich history of communication theory offers many solutions that may be used to mitigate the adverse effects of randomness. In fact, error correcting codes (ECCs) which are one of the inventions that made all modern digital communications possible have always been an essential part of any storage system including flash memory. However promising the advances in communication theory may be, flash memory presents a unique set of characteristics that make the straightforward adoption of communication techniques infeasible. These characteristics include low tolerance of access delays, low-precision sensing, the non-conventional nature of noise and interference due to process variability and degradation, etc [4]. While in the past, a simple conventional ECC scheme was suitable for flash memory, the increasing number of errors requires more advanced schemes. In particular, it is becoming necessary to shift from hard decoding schemes such as Reed-Solomon (RS) and Bose–Chaudhuri–Hocquenghem (BCH) codes, towards the stronger soft decoding schemes such as Low-Density-Parity-Check (LDPC) codes. Soft decoding is complex, power-hungry, and requires accurate statistical knowledge about the channel and high-precision channel outputs, all of which are not to be taken for granted [9].

In general, to design any reliability enhancing signal processing technique, it is essential to have a model that accurately represents the different sources of errors. In flash memory, as we shall see, a lot of work has been done to provide an accurate statistical model of the conditional distribution of the output of a cell given its input. However, inter-cell interference has not received an in-depth analysis beyond

being treated as independent noise, or equalized using expensive methods. As a result, a fundamental assumption that is always made is that cells are physically uncorrelated. As interference becomes a major source of errors and a limitation to further size reduction, this assumption becomes very simplistic; and uncovering the deep structure of interference becomes essential to efficiently mitigate its effect. This constitutes a central motivation for this thesis in addition to the reliability issues briefly discussed above, and to be discussed in detail throughout this manuscript. The primary goals of the thesis therefore are

- To advance the mathematical understanding of the error characteristics of flash memory, with emphasis on interference.
- To exploit this understanding in designing advanced signal processing techniques that reduce errors.
- To understand the other inherent limitations of flash memory, such as the variation of optimal read references, and design appropriate techniques that help in alleviating them.

1.2 Thesis Outline and Contributions

The thesis is composed of three research chapters, preceded by a literature review of the flash memory technology and its challenges and succeeded by a conclusion and potential future works. Broadly, the first two research chapters focus on interference. In the first, we incorporate interference in a complex but accurate and comprehensive model, and we show why this can be beneficial. In the second, we dig deeper into the structure of interference and propose two approaches for data inference that significantly improve the performance of flash. The last research chapter aims

at partially alleviating some of the inherent limitations of current flash memory operation. The detailed outline of the thesis is as follows

- Chapter 2 presents the necessary background on flash memory and the major reliability issues along with their models.

- Chapter 3 investigates the effects of noise and inter-cell interference on the outputs of flash cells. First, the distribution of the equivalent noise term, which is independent across cells, is derived and shown to be approximately normal. The chapter then derives the output distribution of a cell given its input in the presence of interference. The chapter proceeds to consider a group of adjacent cells that are subject to interference by common cells and derives the joint conditional distribution of their outputs given their inputs. The proposed model, formulated as a multivariate Gaussian mixture, is the first to account for the correlation between victim cells due to common aggressors. Finally, it is shown that the normalized joint mutual information between the inputs of adjacent cells and their output exhibits an increase as the number of cells in the group increases. The increased mutual information indicates that the adjacent cells hold information about one another. This motivates the next chapter that looks into the joint processing of adjacent cells to extract this mutual information and improve the soft decoding of ECCs.

- Chapter 4 investigates the inference of stored data in the presence of interference. First, the problem is stated as the computation of the posterior probability mass function of the input of a cell given the outputs of all cells. Then, the performance of soft decoding is shown when cells are grouped into pairs. The improved bit error rate shows the potential benefit of joint detection. However, the computational complexity increases exponentially as the number of

cells forming an independent group increases. By exploiting the local nature of interference, the flash memory system is approximated by a hidden Markov model. Subsequently, the sum-product algorithm is used to extract soft information about the inputs of the cells from the outputs of a large group of adjacent cells. The sum-product is shown to provide a significant improvement over the conventional approach where the soft information of a cell is extracted only from its output. Finally, the effect of quantization on the performance of the different approaches is studied, and it is shown that the sum-product approach is the most susceptible to performance degradation due to quantization. To reduce the effect of quantization, an iterative detection/decoding strategy is proposed and shown to improve the bit error rate.

- Chapter 5 proposes a new data representation scheme that provides a trade-off between reliability and the amount of data stored per cell, and partially mitigates the effects of device degradation. First, the problems that detection using conventional hard decision boundaries suffers from are discussed along with one solution which is critically reviewed. Then, the chapter introduces index programming (IP) where the number of non-erased cells is fixed and known a priori. This effectively divides the stored data into two streams that are conveyed by the indices and levels of the non-erased cells, respectively. Consequently, the first stream is retrieved by varying the decision boundary dynamically to obtain the correct number of non-erased cells. This dynamic detection is independent of the variations in the channel due to degradation. The chapter then analyses the storage efficiency of the new scheme, its effect on ICI, and derives the analytical expressions of its error rate. The simulation results show an improvement in the error rate while partially mitigating the

need to track the channel parameters, and the read references as the device degrades. Using IP, the chapter then proposes a method for using a non-power-of-two number of levels, which is usually avoided in conventional architectures. Finally, the chapter introduces an alternative IP scheme that preserves the traditional paging structure and that allows the LSB page to become more resilient to error. This alternative scheme is a potential mitigation technique against programming errors.

- Chapter 6 discusses the conclusions and potential future work.

Chapter 2

Background and Literature Review

In this chapter, we start by presenting the preliminaries on flash memory architecture and operation. Then, we review the major sources of errors and their probabilistic models. The chapter ends with a discussion of the technology trends and previous signal processing work in the field.

2.1 Flash Cell

The basic unit of storage in flash memory, called a cell, is a floating-gate transistor (FG-MOSFET) as shown in Fig. 2.1. The FG is electrically isolated using an interpoly dielectric and a tunnel oxide through which charge can be injected and removed into and from the FG [10]. Like a traditional MOSFET, the FG-MOSFET acts as a switch that can be controlled by applying a voltage to its control gate (CG). The transistor will conduct only if the applied voltage (called reference) is higher than its threshold voltage (V_{th}). However, unlike MOSFET where the V_{th} is fixed, the ability to inject and remove charge into and from the FG allows the V_{th} to be tunable. Consequently, by shifting the V_{th} above or below a fixed reference, a

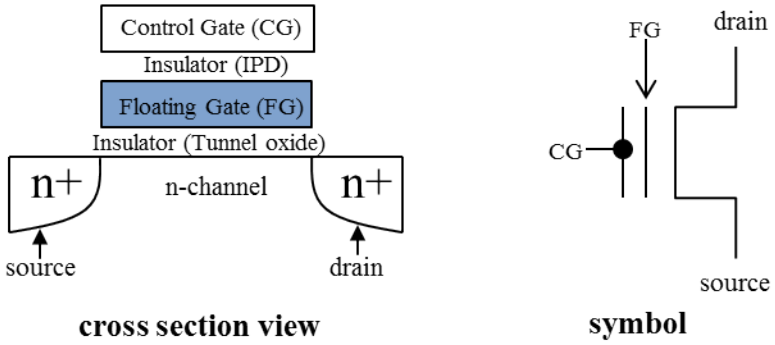


Figure 2.1: Schematic of a floating gate transistor.

storage element that stores one bit per cell is created, e.g., if the transistor conducts when the reference is applied, then a 0 is stored, otherwise, a 1 is stored [4]. This can be straightforwardly extended to more than two levels by dividing the V_{th} range into q distinct levels (windows), indexed by $\mathbb{Z}_q = \{0, ..., q-1\}$. Typically, q is chosen to be a power of 2 so that an integer number of bits is stored in each cell. Data is programmed into a cell by shifting its V_{th} into the level corresponding to that data, and reading it is done by applying several read references to its CG to identify its level. It is common to call the technologies with $q = 2, 4$ and 8 by single, multiple

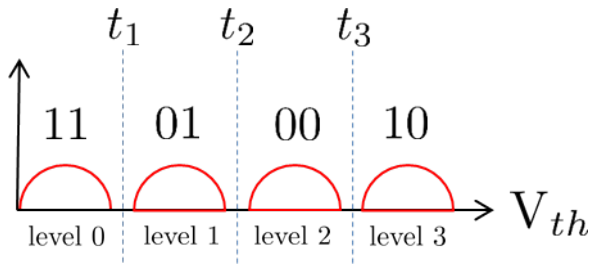


Figure 2.2: Gray mapping of the levels to data bits for MLC.

and triple-level cell (SLC, MLC, and TLC), respectively. Fig. 2.2 shows an MLC example where the V_{th} is divided into 4 levels, and each level is mapped into a unique two-bit sequence using a Gray mapping.

2.2 Planar NAND Flash Memory

A planar NAND flash memory block is a two-dimensional grid of cells as shown in Fig. 2.3. In the vertical direction, cells are connected in series (source to drain) to form a string typically containing 32 to 128 cells. The drain of the first cell in the string is isolated using a drain select transistor from a contact point called bit-line (BL) while the source of the last cell is isolated from the source line using a source select transistor. The cells of a string are addressed using these transistors which are driven by a drain select line (DSL) and a source select line (SSL), respectively, as shown in Fig. 2.3. In the horizontal direction, the CGs of thousands of cells in a row are all connected to a word-line contact (WL) that drives the CG of all cells simultaneously [7]. Each of the $\log_2 q$ bits in a cell resides on a different logical unit called a binary page (bits that are programmed and read simultaneously), e.g., for $q = 4$, the least significant bits in the cells of a WL form the LSB page while the most significant bits form the MSB page.

Note that the same cells are used in the less popular NOR flash architecture but with different connections. In particular, the drains of all cells in a string are

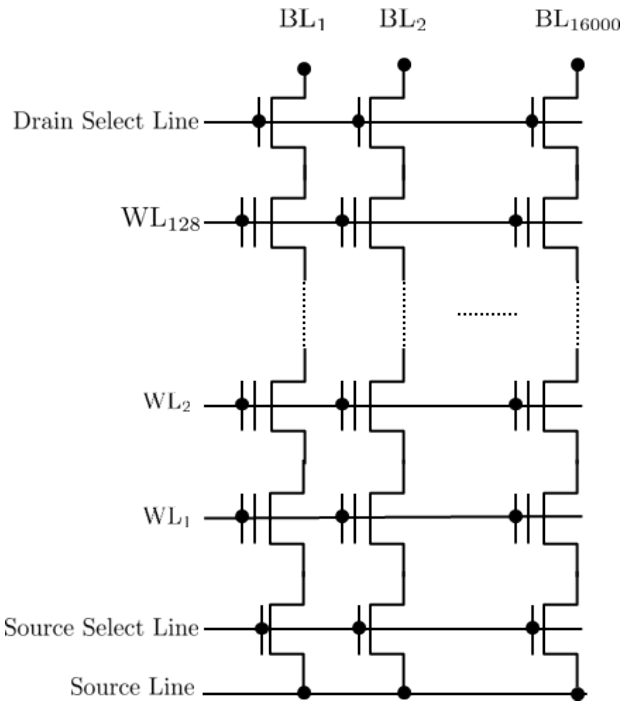


Figure 2.3: Schematic of a NAND flash block.

connected in parallel to a common BL while the sources of all cells in a row are connected in parallel to a common source line. The extra connections needed for the NOR architecture makes it faster especially for random access. However, it significantly reduces the density of cells in the silicon area compared to NAND; a crucial characteristic of NAND that allowed it to become mainstream [10].

2.3 NAND Flash Memory Operations

The three main operations in NAND flash memory are read, program (write) and erase which we explain in the following.

2.3.1 Read

Reading a cell is done by applying a read reference to its CG, and a voltage (V_{pass}) higher than the highest possible V_{th} (typically 6 volts) to the CGs of the unselected cells on the string to make sure they act as pass transistors. The select

transistors are biased to conduct as well. A current will be detected at the BL terminal only if the cell's V_{th} is lower than the applied reference.

In the simplest setting, $q-1$ read references are carefully chosen and applied sequentially to differentiate between the q different levels, i.e., the actual value of the V_{th} of the cell is not of interest but only its estimated level. These references are called *hard references*. In general, each sensing operation can provide more precision on what the actual V_{th} of the cell is (analog to digital conversion). To obtain a high-precision reading of the V_{th} of the cell, a large number of sensing operations is required. Therefore, there is a trade-off between high-precision V_{th} (useful for signal processing), and the read latency.

The duration of a sensing operation, which is equivalent to a discharge in an R-C circuit, is limited mainly by the parasitic capacitive coupling between cells [7]. The NAND organization of cells which allows cost-reduction, increases these couplings. As a result, the time for a sensing operation remained around a few tens of microseconds for the last 15 years. In comparison, NOR flash memories, which use the same cells but a different organization, allow read times well below 100 ns [11]. Nevertheless, the ability to read a very large number of cells simultaneously increases the throughput of NAND well above 100 MB/s (sequential access), which is highly competitive with all other NVS technologies.

2.3.2 Erase

To erase a cell, a high-voltage pulse (20 volts) is applied to the channel (p-well), the CG is grounded, and the drain/source are kept floating [12]. This allows electrons to tunnel from the FG into the channel reducing the V_{th} of the cell. Since all cells in a block share the same p-well, the smallest erasable unit is the whole block. After the erase operation, all cells in the block are in the lowest level (erased

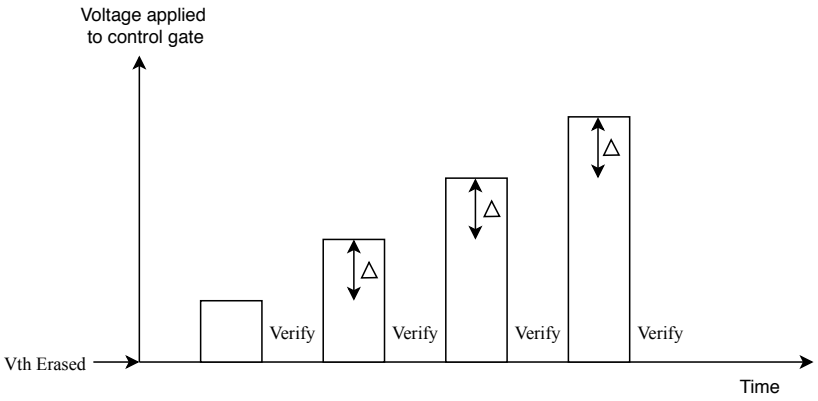


Figure 2.4: Diagram showing the ISPP process.

level). The erase operation lasts few milliseconds which is a long time of exposure to high intensity electrical field that constitutes the main cause of degradation of flash memory. The high cost of erasing in terms of access time and degradation is an important characteristic of NAND flash memory that imposes constraints on the program operation.

2.3.3 Program

To program a cell, a high-voltage pulse (20 volts) is applied to its CG, while the drain, substrate and source are set to 0 volts. This allows electrons, whose amount depends on the magnitude of the applied pulse, to tunnel into the FG. In SLC technology, programming consists of shifting the V_{th} of the cell above a certain threshold, which is relatively easy. In technologies with a higher number of levels, the V_{th} has to be carefully placed between two thresholds, where any overshooting will require a costly erase operation. For this reason, the incremental step pulse programming (ISPP) shown in Fig. 2.4 was introduced [13]. In ISPP, several equal-duration pulses are applied with the magnitude of each pulse incremented by a fixed step size Δ that is carefully chosen. Between every two pulses, a verify (read) operation is performed to check whether the intended level is reached; otherwise a

new pulse is applied. Since all cells on the same WL are connected by their CGs, they form the smallest programmable unit. During the ISPP, whenever a cell reaches its intended voltage, a program inhibit voltage is applied to its BL to inhibit any further charge injection. The typical range for Δ is between 0.2 and 0.5 volts in modern devices [4]. The program time is commonly a few hundreds of microseconds for SLC, 1 ms for MLC, and a few milliseconds for TLC [7]. Despite the longer program time, thanks again to the ability to program a large number of cells simultaneously, the program throughput has been increasing with technology scaling.

2.4 Reliability Issues of Flash Memory

In perfect conditions, the charge and the V_{th} will not change, and the stored data will always be retrieved correctly. However, the V_{th} of a cell is subject to unintentional fluctuations due to several reliability issues. These can be divided into two categories: noise and inter-cell interference.

2.4.1 Noise

The sources of noise in flash memory can be summarized as follows

- Erasure noise: on average, the erase operation brings down the V_{th} of the cell to a nominal value v_0 in the lowest window. However, due to the probabilistic nature of the charge ejection process, and the limited duration of the erase operation, there remain some fluctuations n_E in the V_{th} around this mean. Experiments have shown that the distribution of erasure noise is typically Gaussian. The PDF of n_E is given by

$$p_{n_E}(t) = \mathcal{N}(t|0, \sigma_e^2), \tag{2.1}$$

  and can typically have a large variance σ_e^2 due to the limited erase time and

the variation from cell to cell [12, 14].

- Programming noise: the ISPP is also probabilistic, hence when programming a cell to level $k \neq 0$, there will always be some fluctuations around the nominal voltage v_k that we call programming noise n_P. The distribution of this noise is bounded due to the verify operations, and experiments [15, 16] have shown that it is uniform between $-\Delta/2$ and $\Delta/2$, where Δ is the step size. The PDF of n_P is given by

$$p_{n_P}(t) = \begin{cases} \frac{1}{\Delta} & -\frac{\Delta}{2} \leq t \leq \frac{\Delta}{2} \\ 0 & \text{otherwise} \end{cases}. \tag{2.2}$$

  Naturally, the width of this distribution can be tightened by using a smaller step size. However, this increases the programming time. Hence there is a trade-off between program time and accuracy. Typically, the V_{th} of an erased cell has a large variance compared to that of a programmed cell because erased cells are not subject to ISPP like the programmed ones.

- Random telegraph noise (RTN): it is a fast two-state process of capture and emission of a single electron by/from an oxide trap. This creates a fluctuation in the sensed V_{th}. Due to the Poisson statistics of the capture and emission events [16, 17] the fluctuation n_{RTN} follows a standard $(0, \lambda)$ Laplace distribution. The PDF of n_{RTN} is given by

$$p_{n_{RTN}}(t) = \frac{1}{2\lambda} e^{-\frac{|t|}{\lambda}}, \tag{2.3}$$

  where the parameter λ depends on the number of times the block has been programmed and erased (P/E cycles). As the device goes through more P/E cycles, it degrades creating more traps and amplifying RTN which is reflected

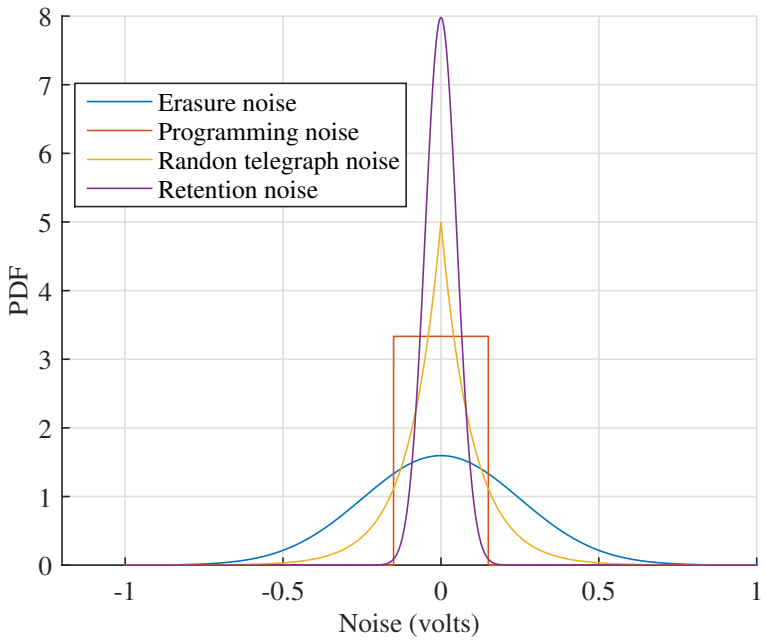


Figure 2.5: The PDFs of the major sources of noises, $\Delta = 0.3$, $\lambda = 0.1$ and $\xi_k = 0.05$.

by an increase in λ.

- Retention noise: after being programmed, phenomena such as de-trapping and interface trap recovery causes electrons to escape from the FG inducing a voltage shift. Experiments [18–20] showed that it follows a Gaussian distribution with variance ξ_k^2 that depends on the number of P/E cycles, retention time and the level k to which the cell is programmed [21]. The PDF of retention noise is given by

$$p_{n_R}(t) = \mathcal{N}(t|0, \xi_k^2). \tag{2.4}$$

  Note that if a cell is kept erased, it does not suffer from retention noise as it is already emptied by the erase operation, i.e., $\xi_0^2 = 0$.

The PDFs of the different sources of noise are shown in Fig. 2.5 for $\sigma_e = 0.25$, $\Delta = 0.3$, $\lambda = 0.1$ and $\xi_k = 0.05$.

2.4.2 Inter-cell Interference

Since cells are containers of charge, the parasitic capacitance coupling between them gives rise to inter-cell interference (ICI), where the voltage shift in one cell induces a proportional shift in its neighbors. Because ISPP is used to inject charge into cells, it automatically mitigates any interference the cell has suffered from beforehand. Hence, a cell is subject to interference only by the cells programmed subsequently to it. As devices become scaled beyond 20 nm, ICI is becoming the main reliability problem of NAND flash as it increases the variance of the V_{th} and consequently the bit error rate [4, 22]. ICI is a local effect since the coupling between two cells is inversely proportional to the distance between them. Thus, the significant coupling in practice is between a cell and the eight cells surrounding it, which we call *direct aggressors* while the coupling with other cells can be considered negligible. The number of direct aggressors actively interfering with a victim cell can be reduced by adopting different programming architecture and sequence that we will explain in section 2.5. Assuming that there are m distinct aggressors, the ICI that the victim cell is subject to can be expressed as follows

$$W = \sum_{j=1}^{m} \gamma_j \Delta Y_j^a \tag{2.5}$$

where ΔY_j^a is the voltage shift that occurs in the jth aggressor in the period between the last programming of the victim cell and reading it, which we call the *active period*. The cell is subject to ICI by the aggressors only in this period due to the anti-causality of ICI. $\gamma_j \ll 1$ is the coupling ratio between the victim cell and the jth interfering cell. Due to the variability of the manufacturing process, a pair of cells may have a different coupling than that of another pair. However, the effect of this variability is negligible compared to the voltage shift ΔY_j^a with which is

it multiplied. This shift is at least one order of magnitude larger. Therefore, to maintain mathematical tractability, in the remainder of this thesis, this variability is ignored, and cells separated by the same distance are assumed to have the same coupling ratio. In particular, the cells that are direct neighbors (same WL/BL or adjacent WL/BL) in the vertical, diagonal and horizontal directions, have coupling ratios γ_v, γ_d and γ_h respectively, independently of their position in the block.

2.4.3 Channel Model

Based on the above review of the different sources of distortions in the V_{th}, it is possible to formulate the problem of reliable storage as a problem of reliable communication over an unreliable channel. This channel is expressed by the mathematical relation between the input and output (read V_{th}) of a cell. If we denote by $X \in \mathbb{Z}_q$ the input level of the cell and by $Y \in \mathbb{R}$ its output voltage, then we can write

$$Y = \begin{cases} v_X + n_E + n_{RTN} + W & \text{if } X = 0 \\ v_X + n_P + n_{RTN} + n_R + W & \text{if } X \in \{1, ..., q-1\} \end{cases} \tag{2.6}$$

where v_x is the nominal voltage of level x. The channel model is then defined by the conditional output distribution given the input denoted $p_{Y|X}(y|x)$ and is the topic of chapter 3.

2.5 Program Architecture and Sequencing

In conventional devices, each two adjacent BLs used to share the same sense amplifier, allowing the program and read of either the even or the odd cells of a WL at the same time. In this even/odd bit-line (EOBL) architecture, the odd cells are programmed after the even ones, subjecting the former to ICI. Note that in this case, the number of pages in a WL becomes $2\log_2 q$, e.g., in MLC, there are two

even and two odd pages (LSB and MSB). However, as chips became smaller, a new architecture called the all-bit-line (ABL) was adopted. In the ABL architecture, each BL is connected to its dedicated sense amplifier, allowing the program and read of all the cells on a WL simultaneously. The consequences are an increased throughput and a reduced intra-WL ICI at the cost of using more space for the sense amplifiers [10]. Since ABL is the widely used architecture in modern devices, we adopt it throughout the thesis, unless otherwise stated. However, it should be clear that the results of the thesis can be easily extended to the EOBL architecture.

The different binary pages on a WL can be programmed separately in an order dictated by the mapping of levels to bits. Fig. 2.6 shows the two-step programming procedure for MLC where initially the cell is erased (in level 0) and does not store any data (XX). This cell is then either kept erased to store 1 in the LSB or shifted to a temporary level (TL) to store a 0, i.e., $X1$ or $X0$, respectively. Then to program the MSB, the LSB bit is first read (using t_0), and depending on the detected value of the LSB and the intended value of the MSB, the cell is shifted to the appropriate level. A similar procedure can be done when 3 or 4 bits are stored per cell, with more temporary levels added. In the remainder of the thesis, the examples will always be based on MLC, and it should be clear that the concepts are readily extendable to higher bit densities.

Typically, the WLs are always programmed in an increasing order starting from the source side to the drain (BL) side and it is prohibited to program them in a random order [4]. In *full program sequencing* (FPS), all the pages on a WL are programmed back-to-back before moving to another WL. Fig. 2.7 shows this program sequencing for an MLC device using the ABL architecture where the numbers between parentheses indicate the order in which the page is programmed. All the pages of a WL are fully programmed before the next WL is programmed, therefore,

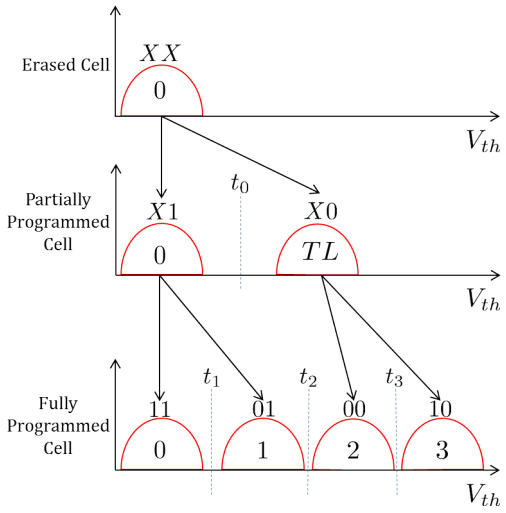


Figure 2.6: Two step programming for MLC.

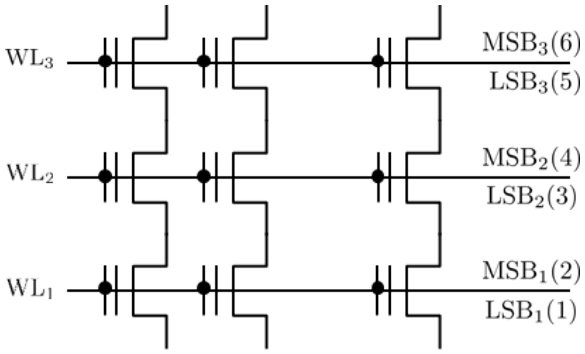


Figure 2.7: Full program sequencing for MLC using ABL architecture.

each WL is subject to interference only by the next WL due to the anti-causality of ICI. While this reduces the number of aggressors on a cell to three, the maximum shift in an aggressor, caused by programming it to the highest level, is significant and can induce errors in the victim cell.

To reduce the maximum shift in the aggressors, the *shadow program sequencing* (SPS), shown in Fig. 2.8, is employed. In this sequencing, the LSB page of the next WL is programmed in between the LSB and MSB pages of the considered WL instead of them being programmed back-to-back. Programming the MSB page of a victim WL will mitigate any previous ICI due to the programming of adjacent pages,

and the fully programmed WL will be subject to ICI only by the programming of the MSB of the subsequent WL, e.g., after programming MSB_2, only the programming of MSB_3 will induce ICI in WL_2. As a result, the maximum shift in the aggressor is

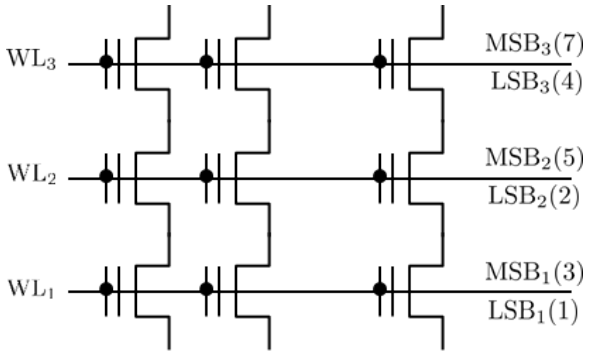


Figure 2.8: Shadow program sequencing for MLC using ABL architecture.

roughly halved since it is caused by shifting the cell from level 0 to level 1 or from level TL to either level 2 or 3. Under SPS, a fully programmed WL is subject to a reduced ICI. However, a partially programmed WL will also be subject to ICI due to the programming of adjacent pages. For example, after programming LSB_2, WL_2 will be subject to ICI by the programming of LSB_3 and MSB_1. To illustrate the consequences of this ICI, we consider a cell in WL_2 where 11 is to be stored. In the first step the cell is kept erased (below t_0), then, due to the programming of LSB_3 and MSB_1, it is possible that its V_{th} is shifted above t_0 changing its LSB value to 0. Consequently, when the MSB of this cell is programmed to 1, its LSB will be read as 0, and hence will be programmed to level 3 instead of 0. In contrast, a partially programmed cell suffers from no ICI in FPS, since the two bits are programmed back-to-back, the final level of the cell will always be correct. This new error mechanism that emerges from the use of SPS has been studied in [23] and is called programming errors. In addition to the increase in errors in the LSB pages, the additional electric stress that is required to program cells into higher levels would deteriorate the cells faster and causes the endurance of the device to diminish. In [23], 2.8 times more

bit errors in the LSB pages were reported when SPS was employed, compared to FPS.

2.6 Technology and Research Trends

The two major advancements that contributed to the success of flash memory are the use of a larger number of levels and the continuous cell-size shrinkage. Since the voltage range tolerated by MOSFET technology is limited (approximately 5 volts), the levels need to be closer to each other. This leads to an increase in the program time due to the smaller ISPP step size required, and a reduction in the error margins, i.e., fluctuations cause errors with higher probability. To make things worse, the shrinkage amplifies these fluctuations, and drastically deteriorates reliability. Fig. 2.9, which is taken from a 2009 report [1], shows the raw bit error rate as a function of the number of P/E cycles for SLC and MLC devices from two vendors. It is clear that the bit error rate increases significantly faster in MLC. It is typical for storage systems to guarantee a bit error rate of 10^{-16} [24]. Therefore, there is a limit to how many times a device can be programmed and erased before it becomes unreliable. While it could be possible to maintain a satisfactory bit error rate in the early life of the device, errors will increase rapidly. To overcome this issue, error correcting codes (ECCs) can significantly extend the lifetime by providing a trade-off between capacity and reliability. Fig. 2.10 illustrates the effect of ECCs with different error correction capability on the extension of lifetime [1].

Since flash memory is an engineered device, its characteristics, including the reliability issues, vary from one generation to another as well as from one vendor to another as is shown in Fig. 2.9. Scaling down amplifies some issues more than others, and each generation tends to focus on solving the most pressing issues of the previous one. For example, ICI steadily evolved to become a major reliability issue

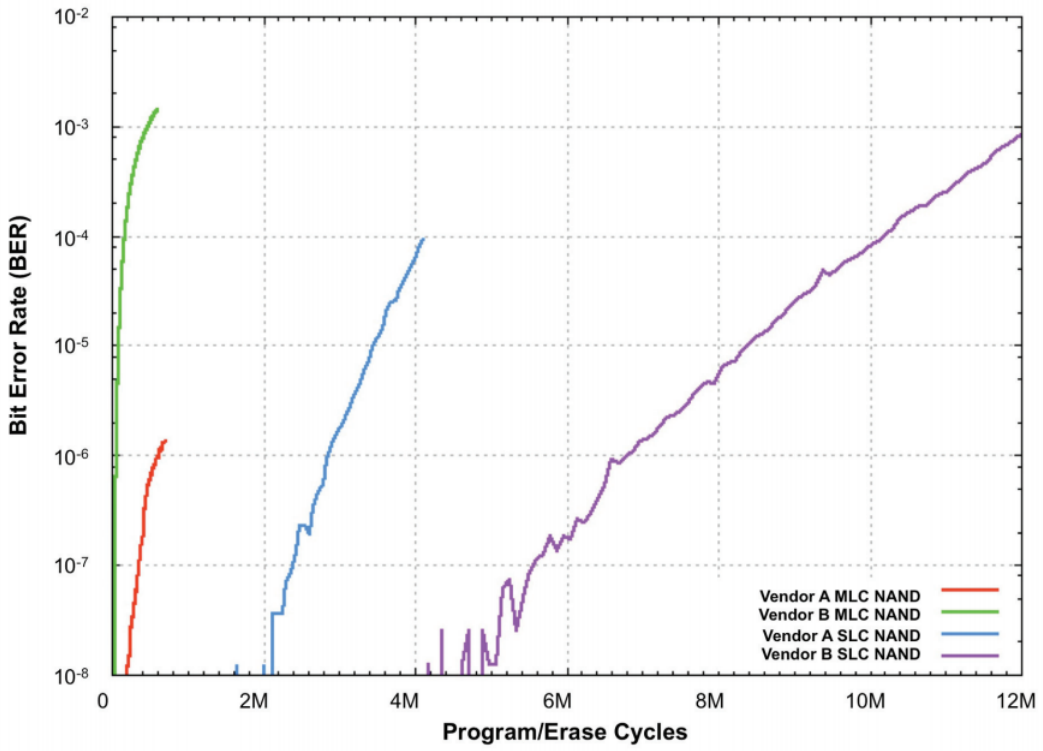


Figure 2.9: The raw bit error rate of SLC and MLC as a function of the number of P/E cycles for different vendors [1].

as devices shrank. Another example is RTN which has been identified since the early days of flash memory as a source of errors. Nevertheless, its characterization is still an active area of research that is studied separately for each generation [16, 17, 25]. Similarly, the 3D NAND technology presents new reliability challenges that were not present in the planar technology [26].

Over the years, a lot of the research was done industrially to solve these short-term, technology-specific problems. Naturally, this research was kept confidential. From an academic point of view, the variability of devices motivated theoretical research to adopt generic and abstract models that can find applications in a range of storage technologies including flash memory. The work in this thesis adopts more or less generic models inspired by communication theory that may not accurately reflect

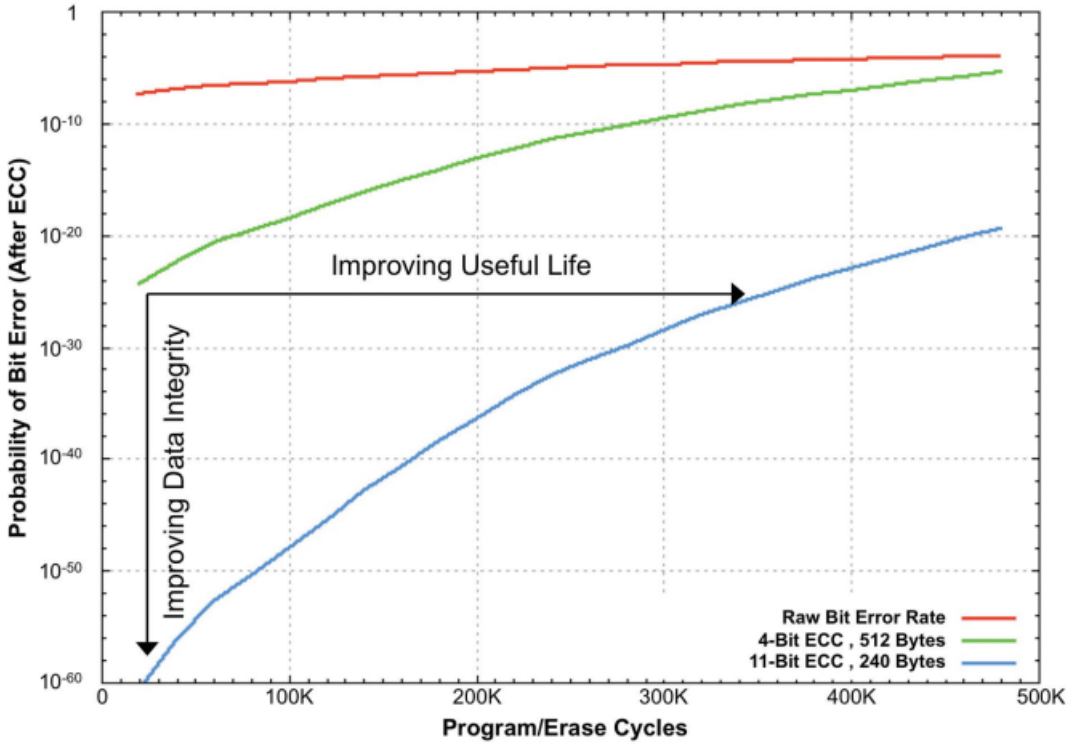


Figure 2.10: The bit error rate of SLC as a function of the number of P/E cycles when ECCs are considered [1].

every reliability aspect of flash memory. By abstracting some of the details and formulating the question as one of reliable communication over unreliable channel, we leverage the rich history of communication theory to provide novel insights and methods that improve the performance of flash memory.

2.7 Signal Processing for Flash Memory

Traditionally, flash manufacturers used a trial and error approach to optimize the device parameters such as the nominal voltages of the levels and the read references. This approach was possible since the reliability issues were less pressing and easier to deal with. However, as devices get scaled to their limit, problems arise, and more careful optimization is necessary. Over the last decade, many solutions tackled the problem at different levels.

At the manufacturing level, 3D flash was introduced, which consists of a new layering approach where the number of cells in a chip can be increased without shrinking the size of a cell [8].

At the system level, many solutions were proposed. In [27], a retention-aware scheme that regularly refreshes the stored data to avoid retention errors was proposed. In [28, 29], ICI post-compensation schemes were proposed to alleviate the problem of ICI. A comprehensive review of such solutions can be found in [30] and the references therein.

Also at the system level, a popular solution is the use of advanced error correcting codes [31] (e.g., LDPC codes with soft decoding) that can correct a larger number of errors. However, equipping flash memory with such capability is challenging for several reasons. One reason is that obtaining accurate soft information requires more sensing operations which may not be feasible due to the limited read time. A second reason is the difficulty of characterizing the flash channel due to process variability, degradation and ICI. From a communication theory perspective, this constitutes a channel with unique characteristics and a lot of work has been done to solve these two issues, e.g., [31–40]. It is worth noting that chapters 3 and 4 fall within this stream of work, focusing on the effect of ICI and on channel characterization for the purpose of soft decoding.

Besides adapting the advanced error correcting codes to flash memory, its unique characteristics begged for different kinds of coding schemes. The asymmetry between the erase and program operations makes it desirable to reduce the number erase operations. As a consequence, codes for write-once-memory were proposed in [41] to maximize the amount of data that can be written into the memory before an erase operation becomes necessary. In the same vein, rank modulation [42] conveys information by the permutation that the ranking of cells' voltages induces, instead

of their absolute voltages. Codes that avoid ICI prone patterns from appearing were proposed in [43–45]. Finally, constant composition codes were introduced in [46, 47] to alleviate the problem of device degradation by allowing a new type of detection using dynamic read references that need not be known in advance. Chapter 5 falls under this category, where we exploit the idea of dynamic read references to partially mitigate some of the inherent problems

Chapter 3

Analytical Model of Flash Memory in the Presence of Interference

In this chapter, we investigate the effects of inter-cell interference on the outputs of the cells. We propose an analytical model of the threshold voltage that accounts for interference. Our model, which is formulated as a multi-variate Gaussian mixture, accounts for an often ignored effect, that is the correlation between victim cells due to common aggressors. We account for this effect and shed light on its importance for scaled down devices where interference is a major source of errors.

3.1 Introduction

The scaling down of device size and the reduction of error margins due to higher bit densities have led inter-cell interference (ICI) to become a major reliability issue in modern flash memories [48]. In order to maintain a virtually error-free storage, it is important to have system level approaches that reduce the probability of errors. One of the most prominent approaches is the use of error-correcting codes, which have been an essential part of any digital storage system. Before designing techniques to mitigate the errors, it is crucial to have an accurate model of the flash channel. This consists of a probabilistic characterization of the output threshold voltage of the cells given their inputs. Having an accurate model is imperative to increase the error-correction capacity of the error correcting codes.

In the flash channel modeling literature, the focus is always on the characterization of a single cell. The underlying assumption is that cells are identical and that their outputs are identically and independently distributed according to the conditional probability distribution $p_{Y|X}(y|x)$ where Y is the output of a cell and X is its input. Several works were focused on fitting empirical measurements from devices to an assumed parametric model of $p_{Y|X}(y|x)$ by minimizing some metric. For example, [32, 49] considered a Gaussian distribution, [50] considered a Normal-Laplace mixture, [35] considered a Student's t-distribution, and [51] considered a discrete Beta-binomial distribution. What these works have in common is that they treat interference as an independent noise, that together with the actual noise, form an overall independent noise term for which the assumed distribution accounts. Alternatively, when an ICI-aware model is pursued, it is often the case that ICI is first processed out of the outputs via equalization and then the ICI-removed output is characterized [28, 29, 34, 52, 53]. However, this approach of dealing with ICI is not

always feasible as it requires access to the interfering cells in order to equalize their effect on the victim cells. This is particularly relevant in SSDs where a single file is not simply written into the WLs of a block, but rather spread over multiple NAND chips of the same device to allow higher degree of parallelism in writing and reading, therefore increasing the throughput [54]. The only case where cancellation is a reasonable assumption is when the whole block is to be retrieved, which is unrealistic in typical systems with high throughput.

Consequently, when equalization is not feasible, ICI must be treated as noise. However, there are two main aspects of ICI that the work mentioned above ignore. The first is that ICI depends on the inputs of the interfering cells and can therefore vary more significantly than what can be accounted for by a simple density, e.g., Gaussian. In fact the empirical investigation of ICI in [55] has shown that a mixture density better describes its distribution. The second aspect is the fact that by definition, ICI creates a correlation between different cells. The obvious cause of this correlation is that one cell interferes with another, but more importantly, even if two cells do not directly interfere with each other, they can be the subject of interference by a third cell, making them correlated by transition.

This chapter is concerned with these two aspects. In particular, we aim to build an analytical model that differentiates between noise, and ICI. Furthermore, the model we propose accounts for the correlation between victim cells due to ICI. As a result, we pave the way for the next chapter which proposes methods for data inference in the presence of ICI.

3.2 Equivalent Noise Distribution

In section 2.4.3, the final V_{th} of a cell was given as follows

$$Y = \begin{cases} v_X + n_E + n_{RTN} + W & \text{if } X = 0. \\ v_X + n_P + n_{RTN} + n_R + W & \text{if } X \in \{1, ..., q-1\}. \end{cases} \tag{3.1}$$

where W is the ICI that the cell is subject to, and its general expression was given by (2.5). The noise terms in (3.1), n_E, n_P, n_{RTN}, and n_R, are all independent, hence by convolution of their PDFs, it is possible to obtain the closed form expression of the conditional distribution of the total noise term Z defined as

$$Z = \begin{cases} n_E + n_{RTN} & \text{if } X = 0 \\ n_P + n_{RTN} + n_R & \text{if } X \in \{1, ..., q-1\} \end{cases}. \tag{3.2}$$

The noise term Z is input-dependent, in particular we get

$$p_{Z|X}(z|0) = \frac{1}{2\lambda} \exp\left(\frac{\xi_0^2}{2\lambda^2}\right) \times \left(\exp\left(-\frac{\xi_0 a_0(z)}{\lambda}\right) Q\left(\frac{\xi_0}{\lambda} - a_0(z)\right) + \exp\left(\frac{\xi_0 a_0(z)}{\lambda}\right) Q\left(\frac{\xi_0}{\lambda} + a_0(z)\right) \right) \tag{3.3}$$

for an erased cell, and

$$\begin{aligned} p_{Z|X}(z|k) =& \frac{1}{2\Delta} \exp\left(\frac{\xi_k a_k(z)}{\lambda} + \frac{\xi_k^2}{2\lambda^2}\right) \left(Q\left(\frac{\xi_k}{\lambda} + a_k(z)\right) - \frac{Q\left(\frac{\xi_k}{\lambda} - a_k(z)\right)}{\exp\left(2\frac{\xi_k a_k(z)}{\lambda}\right)} \right) \\ & - \frac{1}{2\Delta} \exp\left(\frac{\xi_k b_k(z)}{\lambda} + \frac{\xi_k^2}{2\lambda^2}\right) \left(Q\left(\frac{\xi_k}{\lambda} + b_k(z)\right) - \frac{Q\left(\frac{\xi_k}{\lambda} - b_k(z)\right)}{\exp\left(2\frac{\xi_k b_k(z)}{\lambda}\right)} \right) \\ & + \frac{1}{\Delta}\Big(Q\left(b_k(z)\right) - Q\left(a_k(z)\right) \Big) \end{aligned} \tag{3.4}$$

for a cell programmed to level $k \neq 0$ where $a_k(z) := \frac{z-v_k}{\xi_k}$, $b_k(z) := \frac{z-\Delta-v_k}{\xi_k}$, for $k \in \mathbb{Z}_q$, and $\xi_0 := \sigma_e$. $Q(z) = \int_z^{+\infty} \exp(-t^2/2)dt$ is the Q-function. Consequently, the final V_{th} of a cell can be re-written as

$$Y = v_X + Z + W, \tag{3.5}$$

where Z is the input-dependent equivalent noise described by the PDFs in (3.3) and (3.4). It is clear that these expressions are hard to analytically manipulate and use in practice, e.g., in soft decoding. To understand the characteristics of the equivalent noise and approximate it with an easy-to-handle distribution, we resort to its statistical moments.

Table 3.1: Moments of the input-dependent equivalent noise.

	$v_k \neq v_0$	v_0
Mean (v_k)	0	0
Variance (σ_k^2)	$2\lambda^2 + \xi_k^2 + \frac{\Delta^2}{12}$	$\sigma_e^2 + 2\lambda^2$
Skewness	0	0
Excess kurtosis	$\frac{12-\frac{1}{120}\left(\frac{\Delta}{\lambda}\right)^4}{\left(2+\left(\frac{\xi_k}{\lambda}\right)^2+\frac{1}{12}\left(\frac{\Delta}{\lambda}\right)^2\right)^2}$	$\frac{12}{\left(2+\left(\frac{\sigma_e}{\lambda}\right)^2\right)^2}$

The mean, variance, skewness and excess kurtosis of the equivalent noise term can be easily calculated and are given in Table 3.1 for programmed and erased cells. The expressions for the skewness and excess kurtosis of a sum of independent random variables from appendix A were used along with the fact that the excess kurtosis is 0, $-6/5$ and 3 for Gaussian, Uniform and Laplace distributions, respectively. Since the noise components n_E, n_P, n_{RTN} and n_R are all uni-modal and zero-mean, it is evident that the overall conditional distribution $p_{Z|X}(z|x)$ is also uni-modal. The skewness

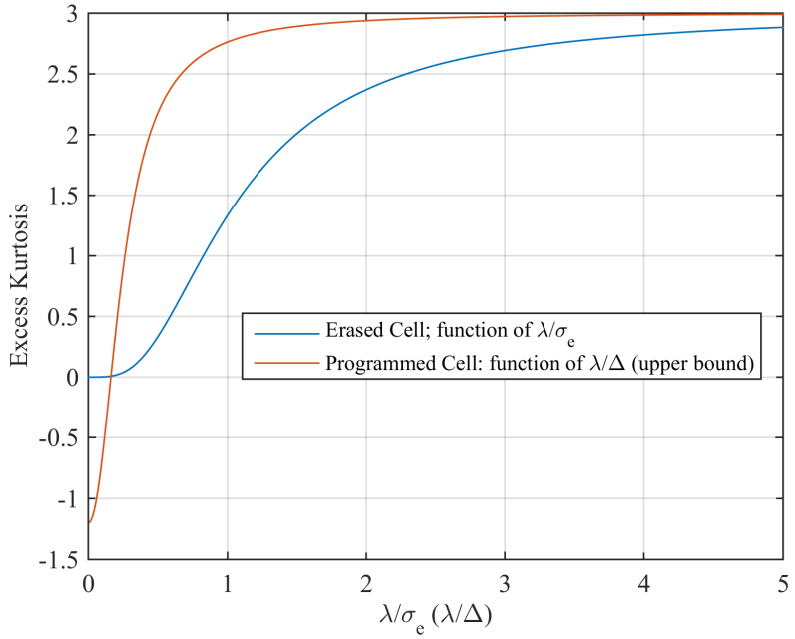


Figure 3.1: Plot of the excess kurtosis of the V_{th} for an erased and programmed cell.

is always 0, hence the equivalent noise is symmetric, which is expected since all its components are symmetric as well. The excess kurtosis of the noise when the cell is kept erased is always positive, and it increases with the RTN parameter λ; whereas when the cell is programmed, excess kurtosis is non-negative for $\frac{\Delta}{\lambda} \leq \sqrt[4]{1440} \approx 6$. In both cases (erased and programmed), the excess kurtosis is bounded from above by 3. It is worth noting that the random telegraph noise and the programming noise act in opposite directions, where the uniform distribution of the latter partially thins the exponential tails of former. The presence of a Gaussian noise component makes the final distribution closer to a Gaussian; hence the case where there is no Gaussian component can be seen as a limiting scenario and the excess kurtosis will then depend on λ/Δ. Fig. 3.1 shows the excess kurtosis in the case of an erased cell (in blue) as a function of λ/σ_e as well as the upper bound on the excess kurtosis of a programmed cell (in red). As the device ages, λ increases and so does the excess

kurtosis, signifying that the exponential RTN tails are becoming more and more significant. It is common in the statistics literature that when the excess kurtosis is in the range of $(-2; 2)$, the distribution can be assumed Gaussian for practical purposes [56]. As stated in the previous chapter, in real devices, typical values for Δ are in the range of 0.2 to 0.5 volts and λ increases with the number of P/E cycles. Hence, it is possible to identify the lifespan in which the noise can be assumed Gaussian and the moments of the Gaussian can be matched with those of the real distribution to minimize the KL divergence [57].

As a conclusion, in the remainder of this thesis, we abstract the different sources of noise and consider a single noise term that follows a zero-mean, input-dependent Gaussian distribution, i.e.,

$$p_{Z|X}(z|k) = \mathcal{N}(z|0, \sigma_k^2) \tag{3.6}$$

where σ_k^2 is the noise variance when the input is the kth level. This approximation will help in the derivation of the output distribution in the presence of ICI, but other distributions can be considered without any change in the analysis.

3.3 The V_{th} Distribution of a Single Cell

In this section, we derive the expression of ICI in a victim cell as well as its distribution. We also derive the conditional distribution of the output of a single cell given its input, i.e., $p_{Y|X}(y|x)$.

The ICI that a victim cell having m direct aggressors is subject to can be expressed as

$$W = \sum_{j=1}^{m} \gamma_j \left(Y_j^a - Y_j^{a,0} \right), \tag{3.7}$$

where the superscript 'a' stands for 'aggressor', γ_j is the parasitic capacitance cou-

pling ratio between the jth direct aggressor and the victim, $Y_j^{a,0}$ and Y_j^a are the V_{th} of the jth direct aggressor at the beginning and end of the active period, respectively. These shifts can be due not only to programming and noise in the aggressor, but also to ICI that the aggressor itself may be subject to in this period. Consequently, (3.7) can be further expanded by replacing Y_j^a and $Y_j^{a,0}$ by their respective expressions as follows

$$\begin{aligned} W &= \sum_{j=1}^{m} \gamma_j \left(v_{X_j^a} + Z_j^a - v_{X_j^{a,0}} - Z_j^{a,0} + W_j^a \right) \\ &= \sum_{j=1}^{m} \gamma_j W_j^a + \sum_{j=1}^{m} \gamma_j \underbrace{\left(v_{X_j^a} + Z_j^a - v_{X_j^{a,0}} - Z_j^{a,0} \right)}_{S_j} \end{aligned} \tag{3.8}$$

where (X_j^a, Z_j^a) and $(X_j^{a,0}, Z_j^{a,0})$ are the level and noise of the jth aggressor at the start and end of the active period, respectively, and W_j^a is the ICI that it was subject to in this period. S_j is then the ICI-free voltage shift in this cell, and it is independent across all cells and depends only on the input of the cell. Its conditional PDF is given by $p_{S|X}(s|x)$. Note that when FPS is employed, the aggressor cells are always initially erased, i.e., $x_j^{a,0} = 0$ for all $j \in \{1, ..., m\}$, since all the bits of a cell are programmed back-to-back. In contrast, when SPS is employed, the aggressor cells can be erased, i.e., $x_j^{a,0} = 0$, when $x_j^a \in \{0, 1\}$ or partially programmed, i.e., $x_j^{a,0} = TL$ when $x_j^a \in \{TL, 2, 3\}$. Hence $x_j^{a,0}$ is a deterministic function of x_j^a. Equation (3.8) shows that ICI can be divided into two terms, the direct interference (rightmost sum) caused by the ICI-free shifts in the direct aggressors, and the indirect ICI (leftmost sum) caused by the propagation of ICI that the direct aggressors themselves are subject to. W_j^a can be further expanded until the last programmed cells in the active period are reached, and W can be re-written simply as a linear combination of the ICI-free shifts in all the cells programmed during the active period. Since typically

coupling ratios are $\ll 1$, the contributions of these shifts diminish exponentially in the number of hops to reach the victim cell, since each hop multiplies the shift by an additional ratio, e.g., $\gamma_j W_j^a$. Therefore, as a first-order approximation, we ignore in our modelling the indirect interference, i.e., we assume that $\sum_{j=1}^{m} \gamma_j W_j^a = 0$, and subsequently assume that $W = \sum_{j=1}^{m} \gamma_j S_j$.

We denote by $\mathbf{X}^{a,0} = (X_1^{a,0}, ..., X_m^{a,0}) \in \{\mathbb{Z}_q \bigcup \{TL\}\}^m$ and by $\mathbf{X}^a = (X_1^a, ..., X_m^a) \in \{\mathbb{Z}_q \bigcup \{TL\}\}^m$ the inputs of the aggressors at the start and end of the active period, respectively. We also denote by $\mathbf{S} = (S_1, ..., S_m)$ the ICI-free shifts in the aggressors, all of which are random. From (3.8), we have that $S_j = \left(v_{X_j^a} + Z_j^a - v_{X_j^{a,0}} - Z_j^{a,0}\right)$ where Z_j^a and $Z_j^{a,0}$ are independent Gaussian variables unless $X_j^a = X_j^{a,0}$ in which case they are equal. Consequently, given X_j^a, S_j follows a Gaussian (the difference of two independent Gaussians is Gaussian), i.e.,

$$p_{S|X}(s|x_j^a) = \mathcal{N}(s|v_{x_j^a} - v_{x_j^{a,0}}, \left(1 - \delta_{x_j^a, x_j^{a,0}}\right)(\sigma_{x_j^a}^2 + \sigma_{x_j^{a,0}}^2)\big), \tag{3.9}$$

where the use of the Kronecker-delta indicator reflects the fact that when a cell does not change its level in the active period, no ICI-free shift occurs in it. Furthermore, the ICI-free shift in a cell depends by definition only on its input, hence $p_{\mathbf{S}}(\mathbf{s}) = \prod_{j=1}^{m} p_S(s_j)$. Therefore, given the inputs of the aggressor cells, ICI is a linear combination of independent Gaussian random variables ($\sum_{j=1}^{m} \gamma_j S_j$), which is also a Gaussian distribution and follows

$$p_{W|\mathbf{X}^a}(w|\mathbf{x}^a) = \mathcal{N}\Big(w|\psi(\mathbf{x}^a), \theta^2(\mathbf{x}^a)\Big), \tag{3.10}$$

where

$$\psi(\mathbf{x}^a) = \sum_{j=1}^{m} \gamma_j (x_j^a - x_j^{a,0}), \text{ and} \tag{3.11}$$

$$\theta^2(\mathbf{x}^a) = \sum_{j=1}^{m} \gamma_j^2 \big(1 - \delta_{x_j^a, x_j^{a,0}}\big)(\sigma^2_{x_j^a} + \sigma^2_{x_j^{a,0}}). \tag{3.12}$$

Now if we assume that the input of the victim cell is also known, then Z also follows a Gaussian distribution. Thus, given the inputs of the victim and aggressors, $W + Z$ is a sum of two Gaussian variables, and

$$p_{Y|X,\mathbf{X}^a}(y|x, \mathbf{x}^a) = \mathcal{N}\Big(y|v_x + \psi(\mathbf{x}^a), \sigma_x^2 + \theta^2(\mathbf{x}^a)\Big). \tag{3.13}$$

The conditional distribution of the output given its input is then obtained by marginalization as

$$\begin{aligned} p_{Y|X}(y|x) &= \sum_{\mathbf{x}^a} p_{\mathbf{X}^a|X}(\mathbf{x}^a|x) p_{Y|X,\mathbf{X}^a}(y|x, \mathbf{x}^a) \\ &= \sum_{\mathbf{x}^a} p_{\mathbf{X}^a}(\mathbf{x}^a) \mathcal{N}\Big(y|v_x + \psi(\mathbf{x}^a), \sigma_x^2 + \theta^2(\mathbf{x}^a)\Big), \end{aligned} \tag{3.14}$$

where $p_{\mathbf{X}^a|X}(\mathbf{x}^a|x) = p_{\mathbf{X}^a}(\mathbf{x}^a)$ since the inputs of victim and aggressor cells are assumed independent. Equation (3.14) shows that the read V_{th} of a cell given its input follows a Gaussian mixture distribution with one component per realization of the inputs of the aggressors. This result coincides with the experimental results of [55] where a similar behaviour was observed.

3.4 Joint Conditional Distribution

In the previous two sections, we derived the distributions of noise and ICI in a single victim cell and subsequently its conditional output distribution. In this

section, we are interested in the conditional joint distribution of two or more cells.

We consider a group of n victim cells and a group of $m' \gg m$ aggressors, where m, as defined before, is the number of aggressors that a single cell is subject to. In general, a cell can be in both groups as it may be at the same time a victim of one or more aggressors and an aggressor to one or more victims. Typically, the victim cells are programmed and read simultaneously, and these cells are by design on the same WL. The positions of the aggressors depend on the programming architecture but are generally in the direct neighborhood of the victim cells (see section 2.4). We denote by $\mathbf{X} = (X_1, ..., X_n)^T \in \mathbb{Z}_q^n$ and $\mathbf{Y} = (Y_1, ..., Y_n)^T \in \mathbb{R}^n$ the inputs and outputs of the victim cells, and we are interested in deriving the conditional joint distribution of the outputs of the victim cells given their inputs, namely $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$.

Since noise is independent from one cell to another, it does not introduce any correlation between them. However, the same can not be said about ICI. The obvious reason is that ICI induces a correlation between the output of the victim and that of the aggressor. But even when the two cells are not interfering with each other, the fact that a third cell can subject both to ICI will induce a correlation between them. The assumption that the outputs of two victim cells are independent given their inputs, i.e., $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} p_{Y|X}(y_i|x_i)$, is valid only in the special case when they have no aggressors in common. However, in realistic scenarios, the victim cells do

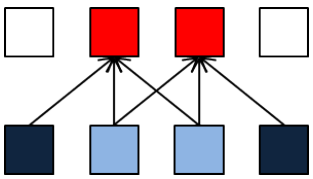


Figure 3.2: The common aggressors (in light blue) to two adjacent victim cells (in red).

have common aggressors, e.g., Fig. 3.2 shows (in light blue) the common aggressors, due to diagonal coupling, of two adjacent cells when ABL is employed with FPS.

The same can be said about SPS, in which case the two victims share four out of six aggressors. Therefore, it is important to derive the exact distribution $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$ that reflects the dependency between adjacent cells, and study its significance as devices become smaller and more prone to high ICI. For this purpose, we define the coupling matrix Γ as the $n \times m'$ matrix containing the coupling ratios between each victim cell and aggressor cell where each row corresponds to a victim and each column to an aggressor. We denote by vectors $\mathbf{S} = (S_1, ..., S_{m'})^T$ and $\mathbf{W}^a = (W_1^a, ..., W_{m'}^a)^T$ the ICI-free voltage shifts and the ICI in the aggressor cells during the active period, respectively, and by $\mathbf{Z} = (Z_1, ..., Z_n)^T$ the noise in the victim cells. Therefore, the single cell channel model of (3.5) can be written in a vector form for the victim group as follows

$$\mathbf{Y} = \mathbf{v}_{\mathbf{X}} + \Gamma\mathbf{S} + \Gamma\mathbf{W}^a + \mathbf{Z}, \tag{3.15}$$

where $\mathbf{v}_{\mathbf{X}} = (v_{X_1}, ..., v_{X_n})$, and $\mathbf{W}^a$ can be further expanded until the last programmed group of cells is reached. Similarly to the single cell analysis, in our theoretical modelling, we assume that indirect ICI is negligible, i.e., $\Gamma\mathbf{W}^a \approx 0$. As explained previously in the single cell analysis, the ICI-free shifts $\mathbf{S}$ are independent and follow each a Gaussian distribution given their inputs $\mathbf{X}^a$. In addition, the components of $\mathbf{Z}$ are independent and follow each a Gaussian distribution given the inputs $\mathbf{X}$ of the victim group. Hence, given the inputs of the victims and aggressors, $\mathbf{Y}$ is a linear transformation of the multi-variate Gaussian vector $\mathbf{S}$ which is known from multivariate probability theory to be a multivariate Gaussian following

$$p_{\mathbf{Y}|\mathbf{X},\mathbf{X}^a}(\mathbf{y}|\mathbf{x}, \mathbf{x}^a) = \mathcal{N}\Big(\mathbf{y}|\mathbf{v}_{\mathbf{x}} + \Gamma\boldsymbol{\Psi}(\mathbf{x}^a), \mathrm{Diag}(\boldsymbol{\sigma}_{\mathbf{x}}^2) + \Gamma\boldsymbol{\theta}^2(\mathbf{x}^a)\Gamma^T\Big), \tag{3.16}$$

where

$$\begin{aligned}\boldsymbol{\Psi}(\mathbf{x}^a) =& \mathbf{v}_{\mathbf{x}^a} - \mathbf{v}_{\mathbf{x}^{a,0}},\\ \boldsymbol{\theta}^2(\mathbf{x}^a) =& \text{Diag}\Big((1 - \boldsymbol{\delta}_{\mathbf{x}^a,\mathbf{x}^{a,0}})(\boldsymbol{\sigma}^2_{\mathbf{x}^a} + \boldsymbol{\sigma}^2_{\mathbf{x}^{a,0}})\Big),\end{aligned}$$

and $\boldsymbol{\sigma}^2_{\mathbf{x}} = (\sigma^2_{x_1}, ..., \sigma^2_{x_n})^T$, $\text{Diag}(\mathbf{x})$ is the diagonal matrix formed by placing the elements of the vector $\mathbf{x}$ on the diagonal and $\boldsymbol{\delta}_{\mathbf{x}^a,\mathbf{x}^{a,0}}$ is the element-wise Kronecker delta operator. Consequently, the conditional joint distribution $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$ can be obtained by marginalization similarly to the case of a single cell. We find

$$p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \sum_{\mathbf{x}^a} p_{\mathbf{X}^a}(\mathbf{x}^a)\mathcal{N}\Big(\mathbf{y}|\mathbf{v}_{\mathbf{x}} + \Gamma\boldsymbol{\Psi}(\mathbf{x}^a), \text{Diag}(\boldsymbol{\sigma}^2_{\mathbf{x}}) + \Gamma\boldsymbol{\theta}^2(\mathbf{x}^a)\Gamma^T\Big), \tag{3.17}$$

which describes a multi-variate Gaussian mixture where the number of components in the mixture depends on q and the total number of aggressor cells m' of the considered victim group.

When ABL is employed with FPS, the cells on the same WL constitute a victim group, and the direct aggressors are all the cells on the next programmed WL, hence $m' = n$. When SPS is employed, $m' = 2n$ since the cells on the previous and next WLs interfere with the victim WL. At the time of reading the victim group, if the aggressor group is still erased, i.e., $\mathbf{x}^a = \mathbf{0}^T$, victim cells are subject only to noise and no ICI, hence $\boldsymbol{\Psi}(\mathbf{0}^T) = \mathbf{0}^T$ and $\boldsymbol{\theta}^2(\mathbf{0}^T) = \text{Diag}(\mathbf{0}^T)$, indicating that the outputs of the victim cells are independent.

Therefore, we have shown that the exact joint conditional output distribution can be expressed as a multivariate Gaussian mixture, which can be seen as a generalization of the single cell distribution. The covariance matrix of the distribution in (3.17) is a function of the covariance matrices $\boldsymbol{\theta}^2(\mathbf{x}^a)$ of the components of the

mixture. In general, these are not diagonal due to the coupling matrix Γ, making the overall covariance matrix non-diagonal. Therefore, the components of vector $\mathbf{Y}$ will be correlated, due to common aggressors. The number of components in the mixture grows exponentially with the number of victim cells considered jointly. In any practical scenario, the use of this distribution is prohibitively complex even if a small number of cells are considered jointly.

It is worth noting that it has been shown in the statistics literature that the Gaussian mixture model is general and that under regular conditions it may approximate any continuous function having a finite number of discontinuities [57, 58]. Hence, regardless of our analytical result, considering a Gaussian mixture is a good starting point in any empirical study that tries to evaluate an accurate parametric flash model in the presence of ICI.

Finally, in contrast with previous models discussed in the chapter introduction, this model accounts for the dependency of ICI on the inputs of the aggressors, as well as for the dependency between victim cells due to common aggressors.

3.5 Mutual Information Between the Input and Output of Flash Channel

In the previous two sections, we derived the conditional output distributions of a single victim cell and that of a group of victim cells (typically a WL). Both distributions are Gaussian mixtures where the number of components is a function of the number of aggressors. From a modeling perspective, it is essential to understand the extent of correlation between the outputs of cells read simultaneously in order to be able to exploit it. Intuitively, the correlation between cells indicates that they hold information about each other. Mathematically, mutual information is a useful metric measuring the correlation between two random variables (vectors) and

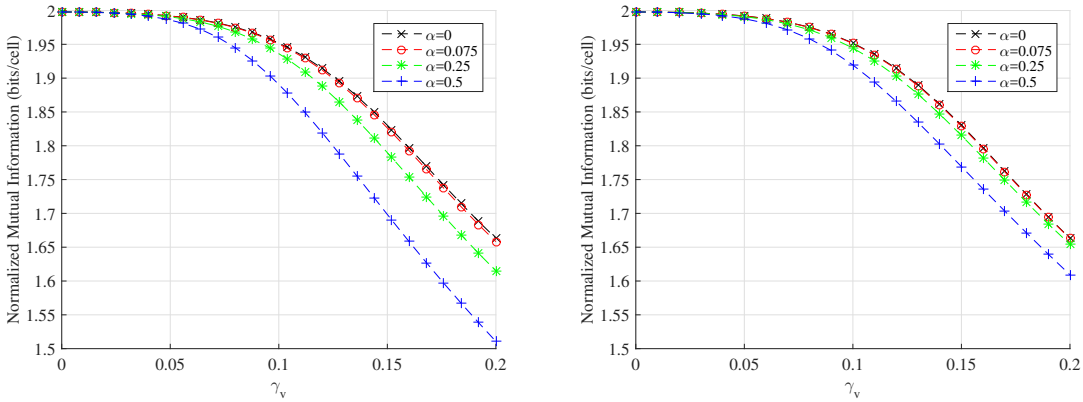
x	0	1	2	3
v_x	1.1	2.7	3.3	3.9
σ_x	0.35β	0.09β	0.09β	0.09β

Table 3.2: Input and noise parameters of the flash device.

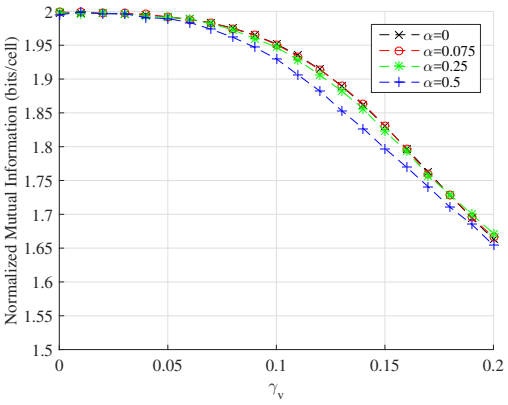
is defined within the context of information theory (see Appendix A).

To assess the correlation between victim cells read simultaneously, we evaluate the normalized mutual information (in bits/cell) between the inputs (assumed i.u.d) and the outputs of the read cells, i.e., $\frac{1}{n}I(\mathbf{Y};\mathbf{X})$, where $I(\mathbf{Y};\mathbf{X})$ is the mutual information, and n is the number of cells processed simultaneously. The mutual information can be calculated directly by plugging the joint conditional distribution $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$, previously obtained into its expression (A.1) given in the appendix and by assuming that the input is i.u.d. We assume that ABL with FPS is employed, and that the coupling between any two adjacent cells is constant and equal to γ_v and $\gamma_d = \alpha\gamma_v$ in the vertical and diagonal directions, respectively, where α controls the strength of diagonal coupling relative to vertical coupling. The coupling in the horizontal direction is negligible due to the ABL structure, which means that the read cells do not interfere with each other but are subject to common aggressors. The input and noise parameters are summarized in Table 3.2 where we consider an MLC device ($q = 4$). The factor β is used to control the significance of noise in the system.

The n cells processed simultaneously are assumed adjacent, hence the total number of aggressors is $n+2$, and Γ is an $n \times (n+2)$ matrix. The analytical evaluation of the mutual information in a Gaussian mixture channel is hard since it requires the evaluation of the entropy of a Gaussian mixture, to which there is no analytical solution. Direct numerical evaluation suffers from the curse of dimensionality where even for small n, e.g., 3, grid-based integration is computationally prohibitive and



(a) $n = 1$. **(b)** $n = 2$.



(c) $n = 3$.

Figure 3.3: Normalized mutual information of MLC ($q = 4$) for $\beta = 1$ as a function of vertical coupling γ_v and four different α. (a) $n = 1$, (b) $n = 2$ and (c) $n = 3$.

advanced simulation-based methods must be employed. Therefore, while in practice thousands of cells are read simultaneously, we restrict our results to $n = 1, 2$ and 3. We use grid-based integration for $n = 1$ and 2 and a Monte Carlo sampling approach for $n = 3$. The results are shown in Fig. 3.3 for $\beta = 1$ where the normalized mutual information is plotted as a function of γ_v for $\alpha = 0, 0.075, 0.25$ and 0.5 in black, red, green and blue, respectively.

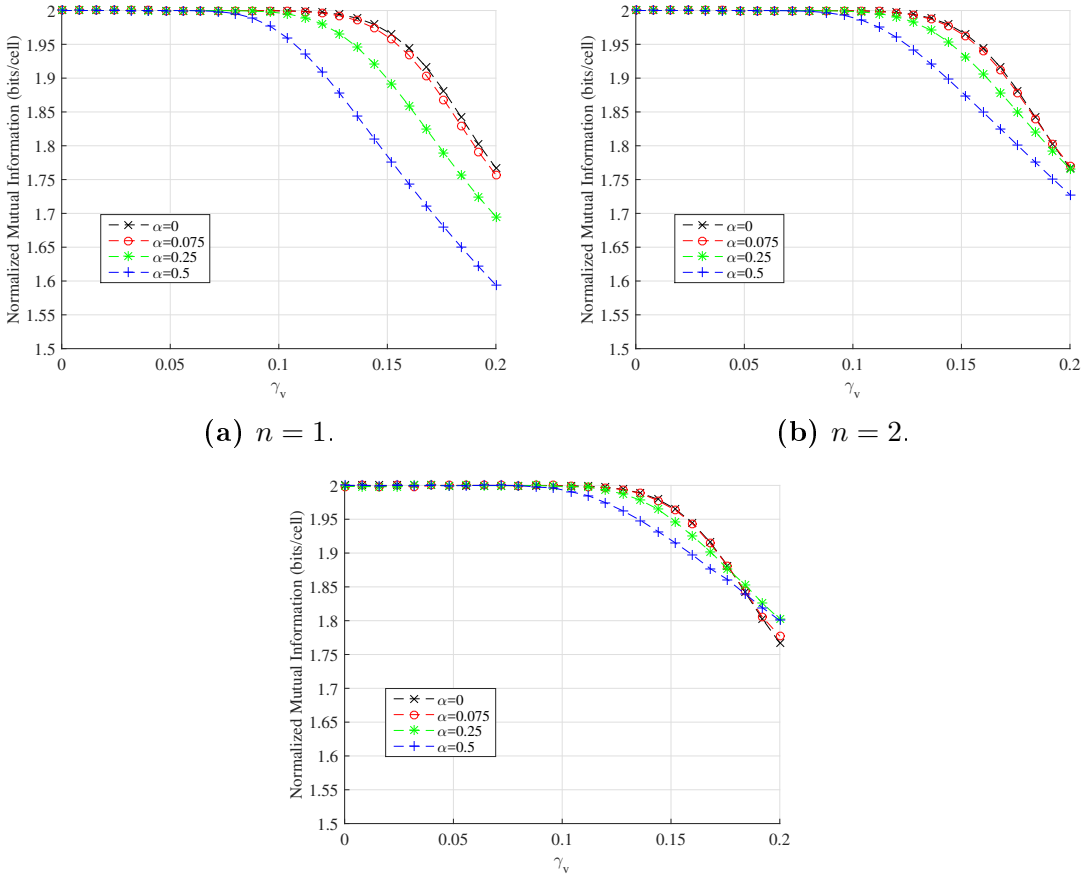
When one cell is processed at a time, i.e., $n = 1$, it is clear from Fig. 3.3a that in general, as γ_v increases, the mutual information decreases. Furthermore,

Table 3.3: Mutual Information at $\gamma_v = 0.2$ and for $\beta = 1$.

n \ α	0	0.075	0.25	0.5
1	1.663	1.658	1.615	1.511
2	1.663	1.663	1.654	1.609
3	1.663	1.667	1.672	1.654

the decrease is faster for higher α, which is expected since a higher α indicates stronger ICI in the system and consequently more uncertainty. As the number of cells that are processed simultaneously increases, i.e., $n = 2$ and 3, the plots for $\alpha = 0$ remain the same as they were for $n = 1$. This is expected since the cells' outputs are completely independent as they do not have any common aggressors and therefore share no information about one another. In contrast, the increase in n has the effect of reducing the speed at which the mutual information decays when α increases. This can be seen from the plots for different α becoming closer to one another as n increases. Table 3.3 shows the mutual information at $\gamma_v = 0.2$ for different values of n and α. It can be seen that as α increases from 0 to 0.5, the mutual information drops by 0.152 bits/cell down to 1.511 when $n = 1$. The drop is markedly less significant when $n = 2$, where a drop by 0.054 bits/cell is observed. When $n = 3$, the drop is a negligible 0.009 bits/cell, thus meaning that the effect of diagonal coupling is virtually annulled. This leads to the conclusion that due to common aggressors, the structure of interference allows for more efficient signal processing techniques by the joint processing of several cells. It is important to note that for $n = 1$ it is evident that the mutual information strictly decreases with α; however, from Table 3.3 , we see that for $n = 3$, the mutual information is higher for $\alpha = 0.25$ than it is for $\alpha = 0$, which means that processing three cells together could not only mitigate the diagonal ICI, but also the vertical ICI.

To gain more insights, we reduce the noise power in the system by setting $\beta = 0.5$.



(a) $n = 1$. **(b)** $n = 2$.

(c) $n = 3$.

Figure 3.4: Normalized mutual information for $\beta = 0.5$ as a function of vertical coupling γ_v and four different α. (a) $n = 1$, (b) $n = 2$ and (c) $n = 3$.

The new plots are shown in Fig. 3.4 and the values at $\gamma_v = 0.2$ are shown in Table. 3.4 . The first interesting observation is that when $\gamma_v = 0$, i.e., noise-limited scenario, the mutual information is almost 2 bits/cell (the maximum number of bits that can be stored with $q = 4$) for $\beta = 0.5$ as it was for $\beta = 1$. However, as γ_v increases, the mutual information decreases much slower for $\beta = 0.5$ than it did previously for $\beta = 1$. As an example, even for $\alpha = 0$, i.e., no diagonal coupling, when $\gamma_v = 0.2$, the mutual information is 1.767 and 1.663 bits/cell for $\beta = 0.5$ and 1, respectively. This significant drop in the mutual information is due to the fact that ICI itself is noisy

Table 3.4: Mutual Information at $\gamma_v = 0.2$ and for $\beta = 0.5$.

n \ α	0	0.075	0.25	0.5
1	1.767	1.757	1.695	1.594
2	1.767	1.777	1.766	1.728
3	1.767	1.777	1.802	1.8

in our model. Going back to Eq. (3.8), it can be seen that the noise that aggressors are subject to, contributes into their final ICI-free shifts. Consequently, there is an interplay between noise and ICI by which noise in the system is amplified by ICI through its propagation from the aggressor to the victim cells. What this means is that as devices scale, ICI colludes with noise to drastically reduce the immunity of flash memories to ICI. We will come back to this observation in the next chapter when we discuss optimal detection.

The second observation is that vertical coupling mitigation thanks to joint processing is further emphasized for $\beta = 0.5$. For $n = 3$, the mutual information increases from 1.767 bits/cell to 1.8 bits/cell as α increases from 0 to 0.5.

In conclusion, the correlation between victim cells is significant enough that it can help alleviate the problem of ICI in highly scaled down devices. This can be done only by reading and processing the victim cells without any assumption of independence between adjacent cells. This, of course, is done without accessing the aggressor cells, which would naturally result in virtually eliminating the effect of ICI even if cells are individually processed. Joint processing of victim cells is the subject of the next chapter.

3.6 Summary

In this chapter, we derived the distribution of the equivalent independent noise that a cell is subject to. We then approximated this distribution by a moments-

matched Gaussian to simplify mathematical manipulations. Then we derived the conditional output distribution of a cell given its input when the cell is subject to ICI. It turns out that it can be written as a Gaussian mixture distribution with one component per realization of the aggressors. Since cells may be subject to ICI by common aggressors, the assumption that the outputs of adjacent cells are independent is only a first-order approximation. To characterize the correlation between adjacent cells, we derived the joint conditional output distribution of a group of victim cells given their inputs. This distribution turns out to be a multivariate Gaussian mixture distribution with a non-diagonal overall covariance matrix, signifying that, indeed, adjacent cells are correlated even if they do not directly interfere with one another. To characterize the extent of correlation between victim cells, we evaluated the normalized mutual information between the inputs and outputs of cells (in bits/cell) when each cell is processed individually as well as when two or more cells are processed simultaneously. We observed that as diagonal interference becomes significant, the information that a cell holds about its neighbors becomes significant and considering cells jointly increases the mutual information. This increase in information can be translated into more effective mitigation of ICI (without access to the aggressors) using signal processing techniques. The exploitation of this dependency is the subject of the next chapter.

Chapter 4

Inference in the Presence of ICI

In the previous chapter, we derived the flash channel model that accounts for ICI and takes into consideration the intra-WL dependence due to common aggressors. In this chapter, we turn to the problem of inferring the input symbols of a WL from the outputs, and without accessing the aggressors which might be infeasible. This amounts to the computation of the symbol posterior probability mass function (PMF), namely, $p_{X_i|\mathbf{Y}}(x|\mathbf{y})$. We first state the problem and discuss the infeasibility of direct estimation of this posterior, then we propose and compare two approaches to solve this inference problem. The first approach which we call the naive approach assumes that a WL can be divided into disjoint groups of cells that can be inferred independently. The second approach is based on the famous sum-product algorithm and exploits the dependency between all of the WL's cells. We then evaluate the effect of quantization on the proposed methods and propose an iterative strategy to alleviate this effect.

4.1 Introduction

When the outputs $\mathbf{Y}$ of the victim cells are read, a reconstruction function is required to estimate the original inputs $\mathbf{X}$ from $\mathbf{Y}$ given that the channel model is known. Naturally, to minimize the symbol error probability of each cell, it is optimal to choose the one that maximizes its posterior probability mass function (PMF) $p_{X_i|\mathbf{Y}}(x|\mathbf{y})$, i.e.,

$$x_i^* = \arg\max_{x \in \mathbb{Z}_q} p_{X_i|\mathbf{Y}}(x|\mathbf{y}) \tag{4.1}$$

where x_i^* is the optimal estimate of the original input of the ith victim cell. In general, $p_{X_i|\mathbf{Y}}(x|\mathbf{y})$ is the marginal of $p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y})$ where the marginalization is done over all unknown variables except x_i, i.e.,

$$p_{X_i|\mathbf{Y}}(x|\mathbf{y}) = \frac{1}{p_{\mathbf{Y}}(\mathbf{y})} \sum_{\sim\{x_i\}} p_{\mathbf{X}}(\mathbf{x}) p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) \tag{4.2}$$

$$\propto \sum_{\sim\{x_i\}} p_{\mathbf{X}}(\mathbf{x}) p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) \tag{4.3}$$

where $\sum_{\sim\{x_i\}}$ is the summation over all unknown variables except x_i, i.e., $x_1, ..., x_{i-1}, x_{i+1}, ..., x_n$. The joint PMF $p_{\mathbf{X}}(\mathbf{x})$ reflects the prior knowledge about $\mathbf{X}$. For example, the components of $\mathbf{X}$ can be i.u.d, i.e., $p_{\mathbf{X}}(\mathbf{x}) = \prod_{i=1}^{n} p_X(x_i)$ and $p_X(x) = \frac{1}{q}$ for all $x \in \mathbb{Z}_q$ and $i = 1, ..., n$. This maximizes the prior uncertainty about the input, which is the worst case scenario, and the performance in this case is known as the *uncoded* performance. On the other hand, if the input is coded using an ECC with parity check matrix H, then $p_{\mathbf{X}}(\mathbf{x}) = 0$ if $H\mathbf{x} \neq \mathbf{0}^T$. The performance of the maximization in this case is known as the *coded* performance. However, it may be highly complex to compute and maximize the posteriors directly, and in practice, it

is commonly divided into two steps. In the first step, also known as detection, the PMFs $\left\{p_{X_i|\mathbf{Y}}(x|\mathbf{y})\right\}_{i=1}^{n}$ are computed with the assumption that the input is uncoded (i.u.d). The computed PMFs, which are the probability distributions of the inputs, given the outputs and the knowledge about the channel, are called the *soft outputs* of the detector. The PMFs can be maximized with respect to the inputs, and the input sequence that maximizes them is called the *hard outputs*. In the second step, known as decoding, the outputs of the detector are passed to the decoder to obtain a decision on the original inputs given the knowledge about the channel and the code with which the input was encoded. The decoder is called soft decoder if it takes as input the soft outputs of the detector (in practice it is more efficient to pass the log-likelihood ratios (LLRs) which can be obtained easily from the soft output), and hard decoder if it takes the hard outputs. It is well known that soft decoding performs significantly better than hard decoding [59], which is why soft decoding is often proposed as a solution to cope with the increasing number of errors in flash. The performance of the soft decoder is very sensitive to the accuracy of the posteriors $p_{X_i|\mathbf{Y}}(x|\mathbf{y})$. Therefore, in this chapter, we will be interested in the evaluation and/or approximation of $p_{X_i|\mathbf{Y}}(x|\mathbf{y})$ when the input is i.u.d, to increase its accuracy and therefore enhance the performance of the soft decoder.

To detect X_i, current approaches use only the output Y_i of the ith cell and therefore rely on $p_{X|Y}(x_i|y_i)$, ignoring the dependence between cells on the same WL. The problem is thus reduced to obtaining the posterior function $p_{X|Y}(x|y)$ that is assumed identical for all cells. The quality of $p_{X|Y}(x|y)$ in this approach depends on the accuracy of the channel model $p_{Y|X}(y|x)$. Alternatively, there are several ways to equalize ICI by reading the aggressors and canceling their effect from the victim cells. However, this assumes that access to the aggressors is available, which is not the case in our investigation. To the author's best knowledge, no previous

work has modeled and exploited the dependency between victim cells due to ICI, to improve the accuracy of the inputs to the soft decoder.

4.2 Problem Statement

In our model, victim cells are not independent and the marginalization in (4.2) is inevitable. The direct evaluation of the posterior of a single cell $\left(p_{X_i|\mathbf{Y}}(x|\mathbf{y})\right)$, requires the evaluation of $p_{\mathbf{X}}(\mathbf{x})p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$ for each possible input $\mathbf{x}$, of which there are q^n possibilities. Furthermore, we have seen in the previous chapter that $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$ is a multi-variate Gaussian mixture with $q^{m'}$ components, where m' is typically equal to n or $2n$ depending on the program sequencing. Hence the direct evaluation of the posterior PMF of a single cell requires the evaluation of $q^{m'+n}$ multivariate Gaussian PDFs (with n variables). Given that n is typically on the order of thousands, direct evaluation is prohibitive, and therefore more efficient (exact and approximate) methods are needed to evaluate the posterior PMFs. The subject of this chapter is to propose a new model-based technique to efficiently approximate $p_{X_i|\mathbf{Y}}(x|\mathbf{y})$ as n increases. The different parameters of the channel are assumed to be perfectly known, i.e., the nominal voltages and noise variance of each level (v_x, σ_x^2), as well as the coupling ratios in the different directions.

4.3 Naive Approach

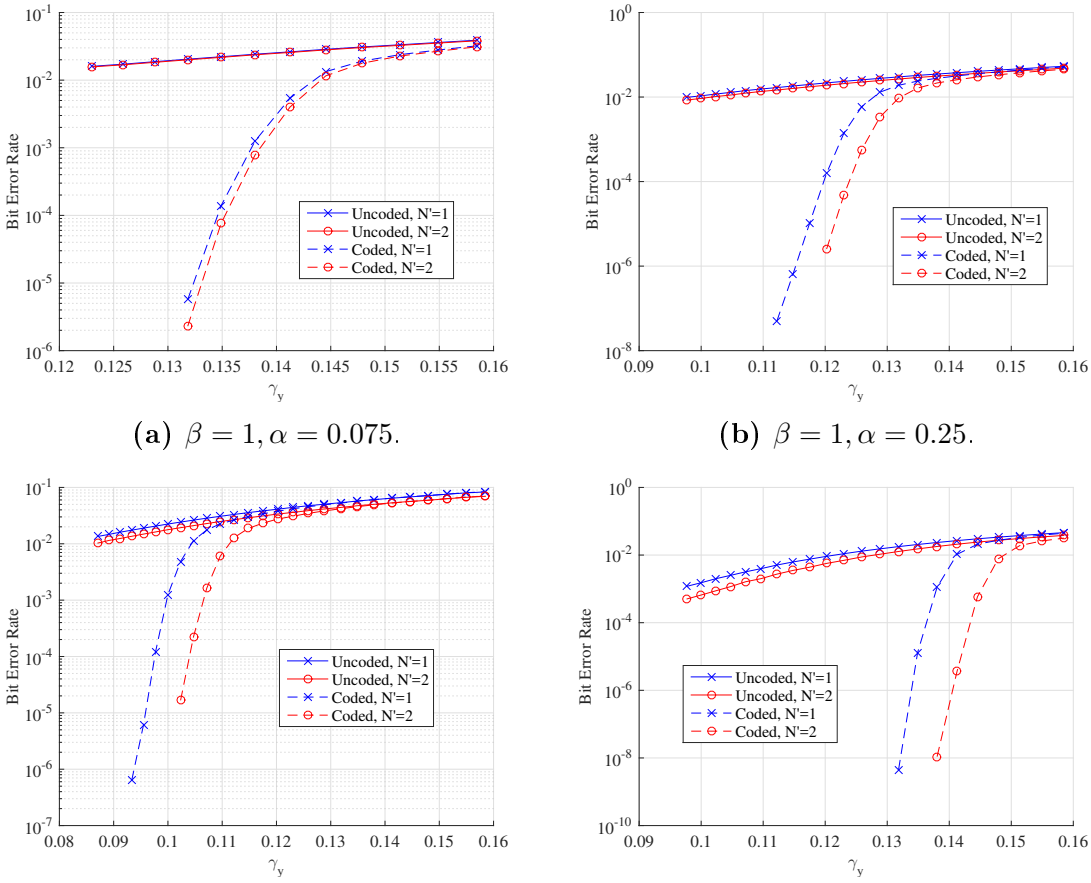
As mentioned earlier, the simplest approach, which is dominant in the literature, is to assume that the flash channel is memoryless, i.e., $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) \approx \prod_{i=1}^{n} p_{Y|X}(y_i|x_i)$. Then, $p_{X_i|\mathbf{Y}}(x_i|\mathbf{y}) \approx p_{X|Y}(x_i|y_i) \propto p_X(x_i)p_{Y|X}(y_i|x_i)$ where $p_{Y|X}(y|x)$ is the single cell output distribution derived in the previous chapter (Eq. (3.14)). The direct evaluation of $p_{Y|X}(y|x)$ is feasible since it consists of the evaluation of q^3 univariate Gaussian PDFs (assuming 3 aggressors). This approach is simple but naive in the sense that it ignores the correlation between the outputs. A step further

would be to partition the cells into $n/2$ contingent pairs, and to assume that the joint outputs of each pair are independent from those of the other pairs, that is $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) \approx \prod_{i=1}^{n/2} p_{Y_o,Y_e|X_o,X_e}(y_{2i-1}, y_{2i}|x_{2i-1}, x_{2i})$, where (X_o, Y_o) and (X_e, Y_e) refer to the input/output pair of a cell in an odd and even position, respectively, and where $p_{Y_o,Y_e|X_o,X_e}(y_o, y_e|x_o, x_e)$ is a bivariate Gaussian mixture according to Eq. (3.17) when the victim group consists of two cells and Γ is a 2×4 coupling matrix. Accordingly, $p_{X_o|\mathbf{Y}}(x|\mathbf{y})$ and $p_{X_e|\mathbf{Y}}(x|\mathbf{y})$ are reduced to $p_{X_o|Y_o,Y_e}(x|y_o, y_e)$ and $p_{X_e|Y_o,Y_e}(x|y_o, y_e)$, respectively. Therefore, instead of each cell being detected independently, each pair of cells is detected independently, but the constituents of a pair are detected jointly, in particular we get that

$$p_{X_o|Y_o,Y_e}(x|y_o, y_e) \propto \sum_{x_e} p_{Y_o,Y_e|X_o,X_e}(y_o, y_e|x, x_e) p_X(x) p_X(x_e) \tag{4.4}$$

$$p_{X_e|Y_o,Y_e}(x|y_o, y_e) \propto \sum_{x_o} p_{Y_o,Y_e|X_o,X_e}(y_o, y_e|x, x_e) p_X(x) p_X(x_o) \tag{4.5}$$

The direct evaluation of $p_{X_o|Y_o,Y_e}(x|y_o, y_e)$ or $p_{X_e|Y_o,Y_e}(x|y_o, y_e)$ consists of calculating q^5 multivariate Gaussian PDFs (assuming 4 aggressors, 2 of them common). This naive approach can be generalized as follows: the victim cells are partitioned into n/n' contingent sub-groups, each containing $n' \leq n$ cells. Then, the outputs of a sub-group are assumed to be uncorrelated with those of other sub-groups but correlated with each other. Therefore, the posterior PMF of a cell will depend only on the outputs of the cells in its sub-group. The cases of $n' = 1$ and 2 are equivalent to the above first an second approaches, respectively. Fig. 4.1 shows the simulated coded and uncoded performances of the naive approach with $n' = 1$ and 2 using the simulation parameters in Table 3.2. For the ECC, a rate-0.89, length-9216 LDPC code was used and decoded using the built-in Matlab decoder with the maximum



(a) $\beta = 1, \alpha = 0.075$.

(b) $\beta = 1, \alpha = 0.25$.

(c) $\beta = 1, \alpha = 0.5$.

(d) $\beta = 0.5, \alpha = 0.25$.

Figure 4.1: Coded and uncoded BER using of the naive approach for $n' = 1$ (blue) and $n' = 2$ (red) as a function of γ_v and for different β and α.

number of iterations set to 50. In this case and throughout the thesis, simulations were run until 100 WLs are detected in error for each data point. We start with the case of $\beta = 1$, i.e., Figs. 4.1a, 4.1b and 4.1c. When $\alpha = 0.075$, the uncoded bit error rate (solid lines) is virtually the same for $n' = 1$ and $n' = 2$. As α increases, the performance for $n' = 2$ improves compared to that of $n' = 1$, but the improvement is not significant as can be seen from the gap between the solid blue and red plots.

On the other hand, a significant improvement can be seen in the coded performance (dashed lines). For $\alpha = 0.075$, the improvement in coded performance is

relatively small, but it becomes significant at $\alpha = 0.25$ where more than a 1.5 orders of magnitude improvement can be observed, e.g., at $\gamma_v = 0.123$, the bit error rate is 1.1×10^{-3} and 5×10^{-5} for $n' = 1$ and $n' = 2$, respectively. For $\alpha = 0.5$, the coded performance is significantly better for $n' = 2$ compared to that of $n' = 1$, e.g., at $\gamma_v = 0.1023$, the bit error rate is 3.3×10^{-3} and 1.2×10^{-5} for $n' = 1$ and $n' = 2$, respectively.

The choice of $\beta = 1$ is relatively high and reflects a significant level of noise in the system, to see what happens when the system is less affected by noise, we show in Fig. 4.1d the same results for $\beta = 0.5$ and $\alpha = 0.25$. In addition to the overall performance becoming better due to the reduction of noise, the gaps between the plots for $n' = 1$ and $n' = 2$ for both coded and uncoded performance is further increased. This shows that in cases where the system is limited by interference, exploiting the dependency between adjacent cells can bring significant improvements to the performance.

The conclusion is that as devices become more and more limited by interference, the dependence between cells due to common aggressors affects the accuracy of the posterior PMFs $p_{X_i|\mathbf{Y}}(x|\mathbf{y})$. By having a more accurate posterior, even if it is obtained using the naive approach, the bit error rate performance of both the uncoded as well as the coded scenarios can be significantly improved. By adopting the naive approach, adjacent cells that are in different groups are assumed independent while in reality, they hold the most significant information about each other. Besides, even the rightmost cell is not independent of the leftmost cell in the wordline. By increasing n', a more accurate approximation of the posterior can be obtained, and hence a better coded and uncoded performance will result. However, the complexity increases exponentially as well, and we are again faced with the problem encountered in the study of mutual information for $n > 3$ in section 3.5. Therefore, we need

a more efficient approach to evaluate/approximate the posterior PMFs.

4.4 Graphical Model for Flash Memory

We have seen in the previous section that the difficulty in evaluating $p_{X_i|\mathbf{Y}}(x|\mathbf{y})$ lies in the direct evaluation of $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})$, and we sought to approximate this likelihood using the naive approach. In this section, we take a step back and try to understand the likelihood function in more depth and whether its complexity can be reduced by exploiting the inherent structure of ICI due to its local nature.

We start by considering the joint distribution of $\mathbf{X}, \mathbf{Y}$, and $\mathbf{S}$, where $\mathbf{S}$ from section 3.4 represents the ICI-free shifts in the aggressor cells. By the chain rule, we can write

$$\begin{aligned} p_{\mathbf{X},\mathbf{S},\mathbf{Y}}(\mathbf{x},\mathbf{s},\mathbf{y}) =& p_{\mathbf{X}}(\mathbf{x}) p_{\mathbf{S}}(\mathbf{s}) p_{\mathbf{Y}|\mathbf{X},\mathbf{S}}(\mathbf{y}|\mathbf{x},\mathbf{s}) \\ =& \Big(\prod_{i=1}^{n} p_X(x_i)\Big)\Big(\prod_{j=1}^{m'} p_S(s_j)\Big) p_{\mathbf{Y}|\mathbf{X},\mathbf{S}}(\mathbf{y}|\mathbf{x},\mathbf{s}), \end{aligned} \tag{4.6}$$

where the second line is true since the inputs of all cells (victim or aggressor) are i.i.d and the ICI-free shifts in the aggressors are also i.i.d. For the ith victim cell, we define the index set $\mathcal{I}_i \subset \{1, ..., m'\}$ as the set containing the indices of its aggressors and where $|\mathcal{I}_i| = m$ for $i \in \{1, ..., n\}$. Subsequently, an equivalent, and more useful representation of $\mathbf{S}$ is obtained by deterministically dividing it into n m-dimensional sub-vectors $\{\mathbf{S}'_i\}_{i=1}^{n}$ where $\mathbf{S}'_i = \mathbf{S}_{\mathcal{I}_i}$ which is the sub-vector of $\mathbf{S}$ indexed by $\mathcal{I}_i$, e.g., if $\mathcal{I}_i = \{i-1, i, i+1\}$, then $\mathbf{S}'_i = (S'_{i,1}, S'_{i,2}, S'_{i,3}) = (S_{i-1}, S_i, S_{i+1})$. In other words, $\mathbf{S}'_i$ represents the ICI-free shifts in the aggressors of the ith victim. By definition, the output of a victim cell depends only on its input and the ICI-free shifts in its m aggressors, which are typically in its neighbourhood. The new representation of $\mathbf{S}$

allows us to equivalently factorize (4.6) as follows

$$p_{\mathbf{X},\mathbf{Y},\mathbf{S}}(\mathbf{x},\mathbf{y},\mathbf{s}) = p_{\mathbf{S}'_1,...,\mathbf{S}'_n}(\mathbf{s}'_1,...,\mathbf{s}'_n)\prod_{i=1}^{n} p_X(x_i)p_{Y|X,\mathbf{S}'}(y_i|x_i,\mathbf{s}'_i), \tag{4.7}$$

where $p_{Y|X,\mathbf{S}'}(y|x,\mathbf{s}')$ is the PDF of the output of a cell given its input X and the ICI-free shifts in its aggressors, which is identical for all cells. A main observation is that while the components of $\mathbf{S}'_i$ are independent by the definition of ICI-free shifts, two vectors $\mathbf{S}'_i$ and $\mathbf{S}'_j$ are not necessarily independent and hence $p_{\mathbf{S}'_1,...,\mathbf{S}'_n}(\mathbf{s}'_1,...,\mathbf{s}'_n) \neq \prod_{i=1}^{n} p_{\mathbf{S}'}(\mathbf{s}'_i)$. This is due to the fact that when $\mathcal{I}_i \cap \mathcal{I}_j \neq \emptyset$, the victim cells i and j have common aggressors. In the case of FPS, the aggressors of a cell are the three adjacent cells on the subsequently programmed adjacent WL. Hence, $\mathcal{I}_i = \{i-1, i, i+1\}$, $\mathcal{I}_i \cap \mathcal{I}_j = \emptyset$ for $|j-i| > 2$, $\mathcal{I}_i \cap \mathcal{I}_{i+1} = \{i, i+1\}$, and $\mathcal{I}_i \cap \mathcal{I}_{i+2} = \{i+1\} \subset \mathcal{I}_i \cap \mathcal{I}_{i+1}$ which means that given $\mathbf{S}'_{i-1}$, $\mathbf{S}'_i$ is independent from $\mathbf{S}'_{i-2}$. Therefore it is easy to see that

$$p_{\mathbf{S}'_1,...,\mathbf{S}'_n}(\mathbf{s}'_1,...,\mathbf{s}'_n) = p_{\mathbf{S}'}(\mathbf{s}'_1)\prod_{i=2}^{n} p_{\mathbf{S}'_i|\mathbf{S}'_{i-1}}(\mathbf{s}'_i|\mathbf{s}'_{i-1}). \tag{4.8}$$

Hence the factorization becomes

$$p_{\mathbf{X},\mathbf{Y},\mathbf{S}}(\mathbf{x},\mathbf{y},\mathbf{s}) = \prod_{i=1}^{n} \underbrace{p_{\mathbf{S}'_i|\mathbf{S}'_{i-1}}(\mathbf{s}'_i|\mathbf{s}'_{i-1})}_{f_i(\mathbf{s}'_i,\mathbf{s}'_{i-1})} \underbrace{p_X(x_i)}_{g_i(x_i)} \underbrace{p_{Y|X,\mathbf{S}'}(y_i|x_i,\mathbf{s}'_i)}_{h_i(y_i,x_i,\mathbf{s}'_i)}, \tag{4.9}$$

where $p_{\mathbf{S}'_1|\mathbf{S}'_0}(\mathbf{s}'_1|\mathbf{s}'_0) = p_{\mathbf{S}'}(\mathbf{s}'_1)$. The sequences of functions f_i, g_i and h_i are introduced for clarity of representation, and the variables of each function will be omitted but should be clear from its index. The same factorization is obtained for SPS using an appropriate indexing of the aggressors, e.g., the upper WL gets even indices while the lower WL gets odd indices. We omit any analysis for the SPS case since it is

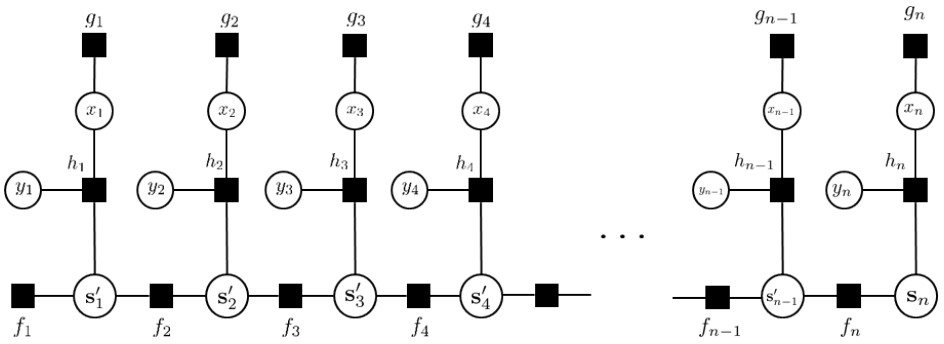


Figure 4.2: Factor graph representing the ABL structure with FPS.

redundant and it should be clear that the presented techniques can be straightforwardly used for SPS.

This factorized expression of the joint distribution encodes the local character of ICI. In particular, even though m' might be very large, only a small number (m) of cells interfere with a single victim cell (generally less than eight). The expression of the joint distribution can be represented by a factor graph (see appendix B) that encodes all the factors of (4.9). Fig. 4.2 shows the factor graph corresponding to ABL with FPS. This factor graph is that of a hidden Markov model where $\mathbf{S}'_i$ represents the state of the system at position i (or simply the state of cell i); however, in contrast with hidden Markov model where the state is discrete, the state here is continuous. The state space is $\mathbb{R}^3$ where three is the number of aggressors that a victim cell is subject to. This representation of the system assumes that the output of a cell depends only on its input and the ICI-free shifts in its aggressors, which is what really happens in flash memory. It is important to note that this representation is not an approximation of our flash channel model, but an exact one. $p_{\mathbf{S}'_i|\mathbf{S}'_{i-1}}(\mathbf{a}|\mathbf{b})$ represents the transition probability between the current and future state of the

system. It is easy to see that for FPS

$$p_{\mathbf{S}'_i|\mathbf{S}'_{i-1}}\Big((a_1,a_2,a_3)|(b_1,b_2,b_3)\Big) \quad = \quad \begin{cases} 0 & \text{if } (a_1,a_2) \neq (b_2,b_3) \\ p_S(a_3) & \text{otherwise} \end{cases}. \quad (4.10)$$

In other words, the two leftmost components of $\mathbf{S}'_i$ are always identical to the two rightmost components of $\mathbf{S}'_{i-1}$, while the rightmost component of $\mathbf{S}'_i$ follows the distribution of the ICI-free shift in any cell.

4.5 Computing the Posterior PMFs

Upon reading $\mathbf{Y}$, it is easy to see that if the states $\{\mathbf{S}'_i\}_{i=1}^n$ are available, the problem of optimal data inference is reduced to a cell-by-cell maximization of the posterior, which is Gaussian. This is in effect the case of ICI-equalization where access to the interfering cells is available that we discussed before. However, when access to the state is unavailable, the quantity of importance $p_{X_i,\mathbf{Y}}(x_i,\mathbf{y}) \propto p_{X_i|\mathbf{Y}}(x|\mathbf{y})$, is the marginal of $p_{\mathbf{X},\mathbf{Y},\mathbf{S}}(\mathbf{x},\mathbf{y},\mathbf{s})$ over all the unknown variables except x_i, i.e.,

$$p_{X_i,\mathbf{Y}}(x_i,\mathbf{y}) = \sum_{\sim\{x_i\}} \int_{\mathbb{R}^n} p_{\mathbf{X},\mathbf{Y},\mathbf{S}}(\mathbf{x},\mathbf{y},\mathbf{s}) d\mathbf{s} \quad (4.11)$$

This marginalization problem can be solved by applying the sum-product algorithm (see Appendix B). Since the corresponding factor graph is a tree, the sum-product algorithm provides the exact marginals for all variables x_i.

When all the messages between nodes are Gaussian functions, the system is called a linear Gaussian system [57] and the message propagation is simple and effecient. However, in our case the messages propagated over the links between different nodes are of discrete, Gaussian, and Gaussian mixture types (functions). This will result in a difficulty in the direct application of the sum-product due to messages along

the forward and backward recursions being Gaussian mixtures instead of simply Gaussians. To illustrate this difficulty, we first evaluate the off-chain messages as follows

$$\mu_{g_i \to x_i}(x_i) = p_X(x_i) = \frac{1}{q} \tag{4.12}$$

$$\mu_{x_i \to h_i}(x_i) = \mu_{g_i \to x_i}(x_i) \tag{4.13}$$

which are trivial, then we have that

$$\begin{aligned} \mu_{h_i \to \mathbf{s}'_i}(\mathbf{s}'_i) &= \sum_{x_i} \mu_{x_i \to h_i}(x_i)\, p_{Y|X,\mathbf{S}'}(y_i|x_i, \mathbf{s}'_i) \\ &= p_{Y|\mathbf{S}'}(y_i|\mathbf{s}'_i) = \sum_{x_i} p_X(x_i) p_{Y|X,\mathbf{S}'}(y_i|x_i, \mathbf{s}'_i) \\ &= \frac{1}{q} \sum_{x_i=0}^{q-1} \mathcal{N}\Big(y_i | v_{x_i} + \sum_{k=i-1}^{i+1} \gamma_k s'_{i,k}, \sigma^2_{x_i}\Big) \end{aligned} \tag{4.14}$$

where γ_k is the coupling ratio between the nth victim and the kth aggressor. This message is a Gaussian mixture with q components. We assume that the message from factor node f_i to node $\mathbf{s}'_i$ is known to be a Gaussian mixture with ℓ components, i.e.,

$$\mu_{f_i \to \mathbf{s}'_i}(\mathbf{s}'_i) = \sum_{k=1}^{\ell} c_k \mathcal{N}(\mathbf{s}'_i | \mathbf{m}_k, \Sigma_k) \tag{4.15}$$

where c_k, m_k and Σ_k are the mixing coefficient, mean vector and covariance matrix of the kth component of the mixture. Consequently, the message $\mu_{\mathbf{s}'_i \to f_{i+1}}(\mathbf{s}'_i)$ is then

computed as follows

$$\begin{aligned}\mu_{\mathbf{s}'_i \to f_{i+1}}(\mathbf{s}'_i) &= \mu_{f_i \to \mathbf{s}'_i}(\mathbf{s}'_i)\, \mu_{h_i \to \mathbf{s}'_i}(\mathbf{s}'_i) \\ &= \frac{1}{q} \sum_{x_i=0}^{q-1} \sum_{k=1}^{\ell} c_k \mathcal{N}\Big(y_i | v_{x_i} + \sum_{k=i-1}^{i+1} \gamma_k s'_{i,k}, \sigma^2_{x_i}\Big) \mathcal{N}\Big(\mathbf{s}'_i | \mathbf{m}_k, \Sigma_k\Big)\end{aligned} \tag{4.16}$$

where the term $\mathcal{N}(y_i | v_{x_i} + \sum_{k=i-1}^{i+1} \gamma_k s'_{i,k}, \sigma^2_{x_i}) \mathcal{N}(\mathbf{s}'_i | \mathbf{m}_k, \Sigma_k)$ can be re-written as a product of two factors: one is a (possibly unnormalized) multivariate Gaussian in $\mathbf{s}'_i$ while the other is completely independent of $\mathbf{s}'_i$ and can be combined with c_k/q to form a new mixing coefficient. Therefore, the forward message from $\mathbf{s}'_i$ is again a Gaussian mixture, however with $q\ell$ components in contrast with the message that $\mathbf{s}'_i$ received, which had only ℓ components. Similarly, if we consider the message from variable node $\mathbf{s}'_i$ to factor node f_{i+1} to be a Gaussian mixture (same expression as (4.15)), then the message from f_{i+1} to $\mathbf{s}'_{i+1}$ can be written as follows

$$\mu_{f_{i+1} \to \mathbf{s}'_{i+1}}\left(\mathbf{s}'_{i+1}\right) = \sum_{k=1}^{\ell} c_k \int_{\mathbb{R}^3} p\left(\mathbf{s}'_{i+1} | \mathbf{s}'_i\right) \mathcal{N}\left(\mathbf{s}'_i | \mathbf{m}_k, \Sigma_k\right) d\mathbf{s}'_i \tag{4.17}$$

and since $p\left(\mathbf{s}'_{i+1} | \mathbf{s}'_i\right)$ is non-zero only when the leftmost two components of $\mathbf{s}'_{i+1}$ are equal to the rightmost two components of $\mathbf{s}'_i$, it is easy to see that

$$\mu_{f_{i+1} \to \mathbf{s}'_{i+1}}\left(\mathbf{s}'_{i+1}\right) = \sum_{k=1}^{\ell} \sum_{x^a_{i+2}=0}^{q-1} \frac{1}{q} c_k \mathcal{N}\left(\mathbf{s}'_{i+1} | \mathbf{m}'_k, \Sigma'_k\right) \tag{4.18}$$

which is again a Gaussian mixture but with $q\ell$ components instead of ℓ. Hence, at each node (variable or factor) in the recursion, the number of components will be multiplied by q. A similar behaviour can be observed in the backward recursion. At the end of the propagation, the computation of the posterior will require the evaluation of Gaussian mixtures with very large numbers of components, which is

infeasible in practice, and some approximations need to be performed in order to reduce the complexity. We discuss our approximation in the next section.

4.6 Approximation of the Graphical Model

The difficulty in applying the sum-product algorithm to the exact model resided in the fact that the messages are Gaussian mixtures with an increasing number of components, which makes their computation inefficient. There are two potential approaches to overcome the complexity of the Gaussian mixture messages. The first consists of approximating the Gaussian mixture messages to limit the growth in the number of Gaussian mixture components from node to node, e.g., at each node, the Gaussian mixture is reduced to a fixed number of components, or even to a single Gaussian such as was done in [60]. The second approach, which we adopt, is to approximate the original channel model in order to reduce the complexity significantly. In particular, we aim to approximate the continuous state variables $\mathbf{S}_i'$ by a closely related but discrete variables, in which case the messages are much easier to compute.

In section 3.5, the interplay between noise and ICI was observed. In particular, we observed that the noise that aggressors are subject to, contributes into their final ICI-free shifts. To see this, the expression of ICI in a victim cell (having m direct aggressors) is as follows (see eq. (3.8))

$$W = \sum_{j=1}^{m} \gamma_j \underbrace{\left(v_{X_j^a} + Z_j^a - v_{X_j^{a,0}} - Z_j^{a,0} \right)}_{S_j} \tag{4.19}$$

$$= \sum_{j=1}^{m} \gamma_j \left(v_{X_j^a} - v_{X_j^{a,0}} \right) + \sum_{j=1}^{m} \gamma_j \left(Z_j^a - Z_j^{a,0} \right) \tag{4.20}$$

which is divided into a discrete interference term, depending only on the nominal

levels of the aggressors, and a continuous propagated noise term that the aggressors are subject to. Hence, in addition to the independent noise that the victim cell is subject to, it is subject to an additional noise that propagates via ICI. However, this propagated noise is not independent, but correlated due to common aggressors. The approximation we make is to assume that this propagated noise is independent. This way, the output of the victim wordline can be approximated as follows

$$\mathbf{Y} \approx \mathbf{v}_{\mathbf{X}} + \Gamma(\mathbf{v}_{\mathbf{X}^a} - \mathbf{v}_{\mathbf{X}^{a,0}}) + \mathbf{Z}', \tag{4.21}$$

where $\mathbf{Z}'$ is an i.i.d noise vector that combines the direct noise that the cells suffer from and the propagated noise. It is easy to see that the propagated noise is also Gaussian since it is a linear combination of Gaussian variables.

Recall that the true channel model dictated that the output of a cell depends only on its input and the *ICI-free shifts* in its aggressors. According to our approximation, the output of a cell now depends only on its input and the *inputs* of its aggressors. The same factor graph is still valid to represent the approximated system. In this graph, the state of a cell is now the *inputs* of its aggressors and the state space becomes $\mathbb{Z}_q^3$. Accordingly, by abuse of notation, we re-purpose the variables $\mathbf{s}'_i$ to represent the new discrete state of the ith victim. Consequently, the function $f_i(\mathbf{s}'_{i+1}, \mathbf{s}'_i) = p(\mathbf{s}'_{i+1}|\mathbf{s}'_i)$ can be written as

$$p_{\mathbf{S}'_i|\mathbf{S}'_i}\Big((a_1, a_2, a_3)|(b_1, b_2, b_3)\Big) = \begin{cases} 0 & (a_1, a_2) \neq (b_2, b_3) \\ \frac{1}{q} & \text{otherwise} \end{cases} \tag{4.22}$$

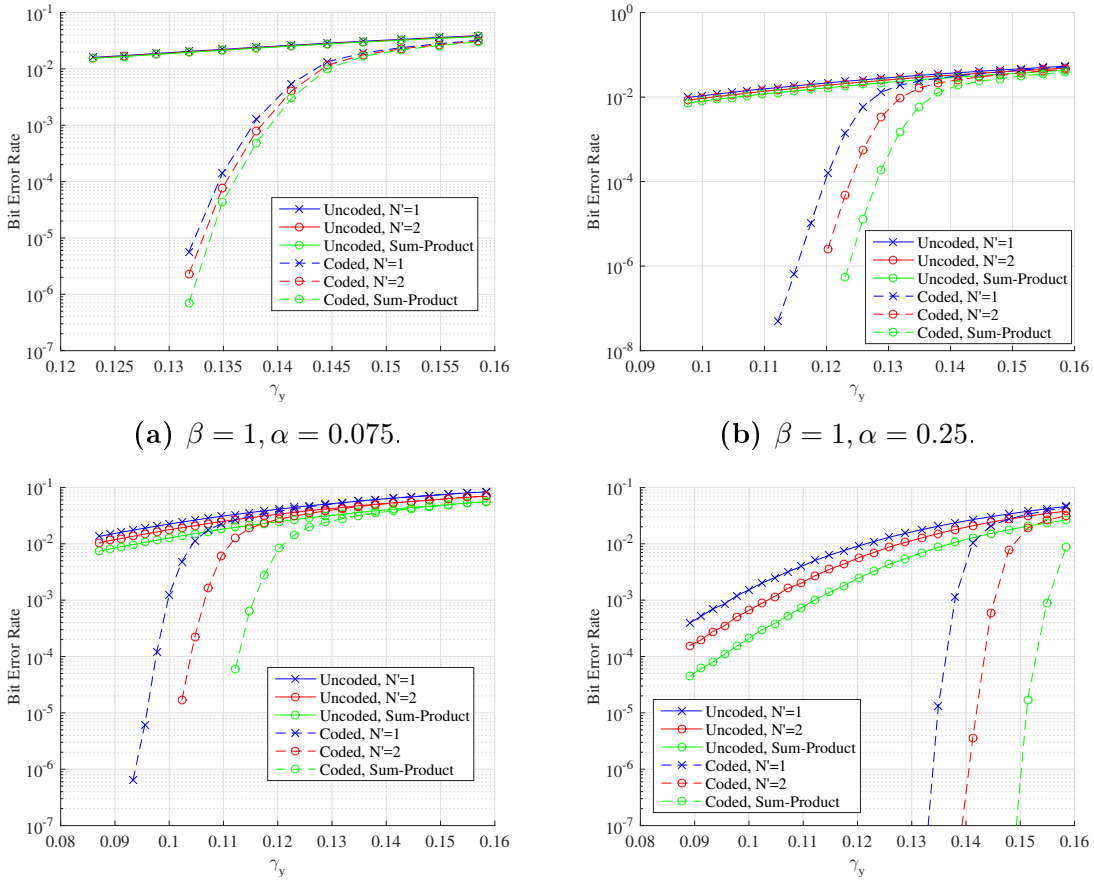
Now that the state space is discrete, it becomes feasible to calculate and propagate the messages between nodes in the forward and backward directions. The transmit-

ted messages will be lookup tables since the variables are discrete along the Markov chain. When the state was continuous, it was not feasible to propagate lookup tables, because there are an infinite number of possible states. The complexity of running the sum-product algorithm now becomes linear in the number of cells n since the messages between the state variables are discrete and easy to compute, as opposed to the exponential growth with n that we saw for the case of Gaussian mixture messages between state variables.

4.7 Performance of the Sum-Product

By running the sum-product algorithm on the proposed factor graph, a new posterior $p_{X_i|\mathbf{Y}}(x|\mathbf{y})$ can be calculated and used for purposes of detection or soft decoding. To demonstrate the benefits of the proposed approach, we compare both the coded and uncoded performances of the new approach against those of the naive approach discussed in section 4.3. We use the same simulation parameters used previously, and we add to Fig. 4.1 the results of the new approach. The new figures are shown in Fig. 4.3.

As expected, when the diagonal coupling is negligible, i.e., $\alpha = 0.075$, the uncoded performance of the three schemes is the same, while the new approach provides a slight improvement in the coded performance over the naive approaches for $n' = 1$ and $n' = 2$. As the diagonal coupling increases, the benefit of the proposed scheme becomes more apparent, especially for the coded case, where the performance gain can reach several orders of magnitude. In general, the results show that the sum-product based posterior is more accurate and provides a more reliable reconstruction of the stored data. Naturally, just like the naive approach with $n' > 1$, processing cells jointly requires diagonal interference to be significant in order to bring a performance advantage. For the ICI-limited regime where $\beta = 0.5$, i.e., Fig. 4.3d, the



(a) $\beta = 1, \alpha = 0.075$.

(b) $\beta = 1, \alpha = 0.25$.

(c) $\beta = 1, \alpha = 0.5$.

(d) $\beta = 0.5, \alpha = 0.25$.

Figure 4.3: Coded and uncoded BER using of the naive approach for $n' = 1$ (blue) and $n' = 2$ (red) and of the proposed approach as a function of γ_v and for different β and α.

benefit of the proposed approach is significant even in the uncoded performance, where almost one order of magnitude improvement in the uncoded bit error rate can be seen at $\gamma_v = 0.88$.

4.8 Effect of Quantization on Data Inference

So far throughout this thesis, it was assumed that the output of a cell is available in floating point precision, i.e., unquantized. While high-precision is possible in practice, there is a trade-off between reading latency and output precision as explained in section 2.3. Therefore, it is desirable to have a small number of required

sensings to achieve an acceptable reliability level. In this section, we assess the effect of having a low-precision output on the inference performance. We show that when the output is quantized, the performance of the sum-product approach deteriorates at a rate faster than the naive approach. Therefore, we propose an iterative strategy to partially recover this drop in performance.

4.8.1 Quantization

Several quantization schemes were proposed to improve soft decoding of LDPC codes. In [31], it was shown that non-uniform quantization improves the performance of soft decoded LDPC codes relative to uniform quantization. In [37], it was shown that a quantizer that maximizes the mutual information between the input and the quantized output improves soft decoding. At the intuitive level, the quantized output should preserve as much information about the input as possible.

Since in practice, the LSB and MSB pages are read independently, we consider three and six read references to read the LSB and MSB page, respectively. Fig. 4.4 shown an example of the quantization boundaries used for detecting the LSB (in black) and the MSB (in red) when $\beta = 1$, $\gamma_v = 0.08$ and $\alpha = 0.075$. The references for each page were obtained such as to maximize the mutual information between the binary input of the page and the quantized output, i.e.,

$$Q_L^* = \arg\max_{Q \in \mathcal{Z}_4} I\big(X_L; Q(Y)\big) \tag{4.23}$$

$$Q_M^* = \arg\max_{Q \in \mathcal{Z}_7} I\big(X_M; Q(Y)\big) \tag{4.24}$$

where $X_L \in \{0, 1\}$ and $X_M \in \{0, 1\}$ are the LSB and MSB bits of a cell, respectively, Q_L^* and Q_M^* are the LSB and MSB quantizers, respectively. $\mathcal{Z}_k$ is the set of all scalar, deterministic quantizers with k outputs, and $Q : \mathbb{R} \to \mathbb{Z}_k$ is a many-to-one function

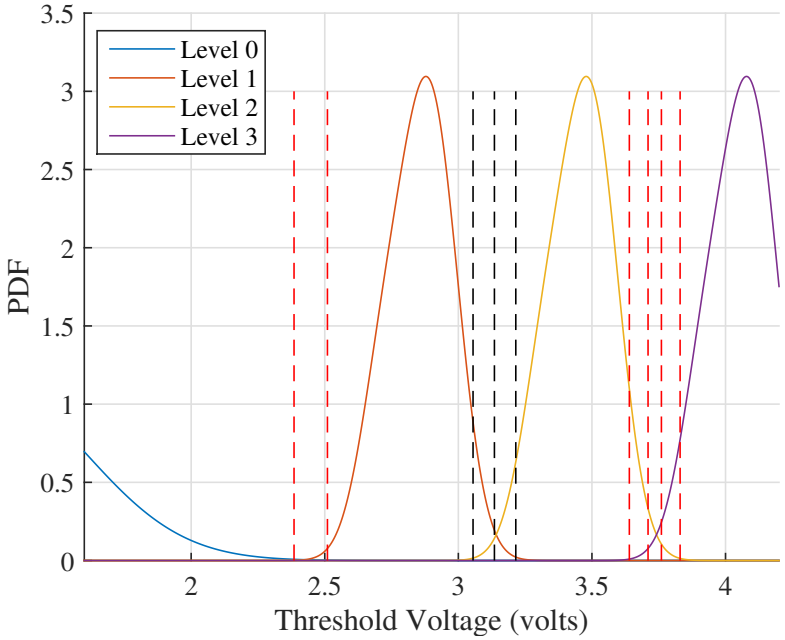
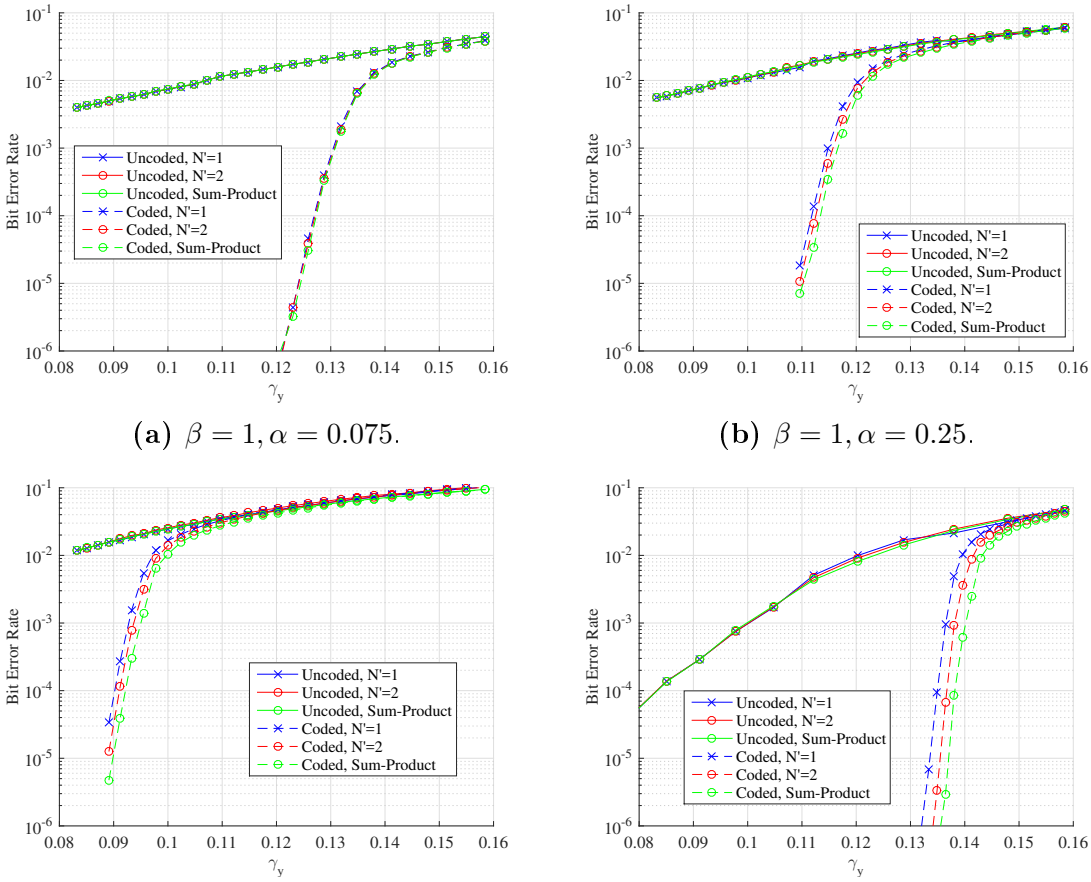


Figure 4.4: The quantization boundaries are shown in black when $\beta = 1$, $\gamma_v = 0.08$ and $\alpha = 0.075$.

in $\mathcal{Z}_k$ that maps the real output to a quantized output in $\mathbb{Z}_k$.

In [61], a low complexity algorithm to obtain such quantizers for binary-input discrete memoryless channels was proposed, and which we adopt. However, it is important to note that this algorithm assumes that the channel is memoryless, therefore, the obtained quantizers are optimal only when the input of each cell is inferred using its output, which is true only in the case of the naive approach with $n' = 1$. However, in the case of the sum-product approach, the input of a cell is inferred from the outputs of a group of cells, and hence, it is not guaranteed that such quantizer maximizes the mutual information between the input and the quantized outputs, i.e., $I\big(X_L; Q(Y_1), ..., Q(Y_N)\big)$ and $I\big(X_M; Q(Y_1), ..., Q(Y_n)\big)$. Nevertheless, we do not aim to design an optimal quantizer, and the performance of the sum-product approach can be considered as a bound on the optimal performance.



(a) $\beta = 1, \alpha = 0.075$. **(b)** $\beta = 1, \alpha = 0.25$.

(c) $\beta = 1, \alpha = 0.5$. **(d)** $\beta = 0.5, \alpha = 0.25$.

Figure 4.5: Coded and uncoded BER using of the naive approach for $n' = 1$ (blue) and $n' = 2$ (red) and of the proposed approach when the output is quantized as a function of γ_v and for different β and α.

4.8.2 Performance Under Quantization

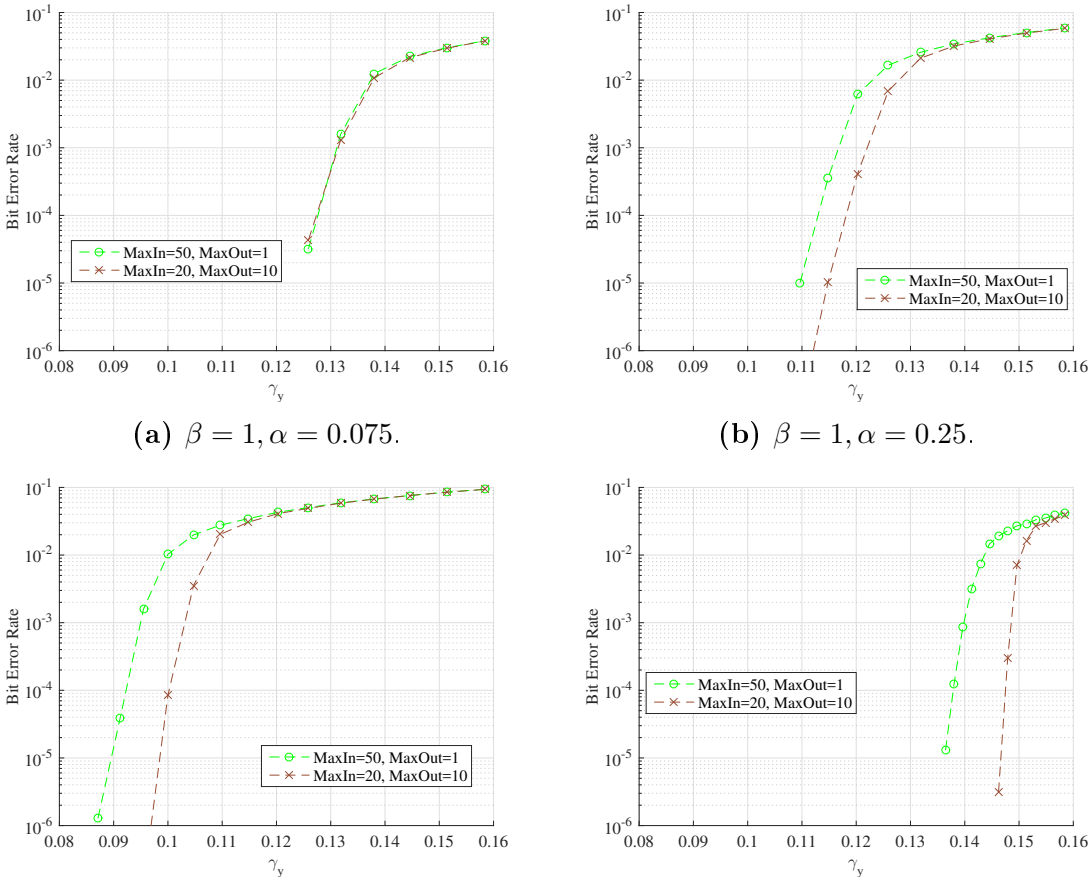
In this section, we present the same results shown in Fig. 4.3, but where the output is quantized as discussed above. The performances of the different approaches are shown in Fig. 4.5. The observations made before still hold regarding the superiority of the sum-product when the diagonal coupling is significant, and the capability of the sum-product to reduce the adverse effects of ICI by exploiting the correlaion between adjacent cells. Despite being superior, the performance of the sum-product is the most negatively affected by quantization compared to the naive

approach with $n' = 1$ and $n' = 2$. This significant deterioration of performance can be explained by the fact that quantization, especially when it has low-precision, discards some information from the output signal. The sensitivity to quantization is then a matter of how much the inference scheme has been initially benefiting from the discarded information. In our case, the sum-product benefited significantly more than the naive approach from the outputs when they were not quantized to learn information about ICI and provide high-quality LLRs to the soft decoder. The quantized output contains information about the input, noise, and ICI. When the noise is significant, a big part of the information kept due to quantization will be related to noise, which is independent, and the performance of the sum-product seems to become very similar to that of the naive approach. However, when the system becomes limited by ICI, e.g., $\beta = 0.5$ as shown in Fig. 4.5d, it is clear that the quantized output carries much more information about ICI, which can now be exploited more efficiently by the sum-product, and we can see the performance gap between the sum-product and the naive approach increasing again.

As a conclusion, the proposed sum-product algorithm is sensitive to quantization and can cope better with it in an ICI-limited scenario. As explained before, the quantization scheme is not necessarily optimal for the sum-product scheme, and it may be possible to design a better quantizer that extracts more information to inform the sum-product algorithm. However, in the next section, we use a different approach to improve the performance of the sum-product algorithm using the same quantization scheme.

4.8.3 Iterative Detection/Decoding Strategy

In our setting, the sum-product algorithm is used to soft detect the input symbols and subsequently pass the soft information to the soft decoder. In other words,



(a) $\beta = 1, \alpha = 0.075$. **(b)** $\beta = 1, \alpha = 0.25$.

(c) $\beta = 1, \alpha = 0.5$. **(d)** $\beta = 0.5, \alpha = 0.25$.

Figure 4.6: Coded BER using the sum-product with and without the iterative detection/decoding strategy. $MaxIn$ is the maximum number of iterations that the decoder performs and $MaxOut$ is the maximum number of times the detection/decoding process is done.

the detector informs the decoder but not vice-versa. The structure of the ECC obtained from the constraint $H\mathbf{x} = \mathbf{0}^T$ can be represented by a factor graph (Tanner graph) [59]. Therefore, the channel and code factor graphs can be combined into a single factor graph representing the system on which inference is to be performed using the sum-product algorithm. In this case, no differentiation is made between detection and decoding.

As a middle ground, we propose a strategy that still differentiates between de-

tection and decoding, but allows the detector to inform the decoder and vice-versa, by iterating multiple times between detection and decoding. In other words, the output of the decoder, which is another posterior distribution on the input, is passed back to the detector as a new prior, after removing from it the intrinsic information. Note that the decoder itself is an iterative instance of the sum-product since it has cycles, while the detector is a cycle-free instance of it. Hence, we differentiate between the number of inner iterations of the decoder, and the number of outer iterations that switches between decoding and detecting. We denote the maximum number of these iterations as $MaxIn$ and $MaxOut$, respectively. The scheme evaluated in the previous sections corresponds to $MaxIn = 50, MaxOut = 1$ which means there is no iteration between detection and decoding. Fig. 4.6 shows the results when the output is quantized and when the iterative strategy between detection and decoding is employed with $MaxIn = 20$ and $MaxOut = 10$ and compares it to the case of the previous subsection. The choice of the of $MaxIn = 20$ $MaxOut = 10$ was made arbitrarily based on a few trials. In the trials, a high number of internal iterations of the decoder lead to an early convergence (local optimum) of the decoder before the detector get a chance to exchange information. At the other extreme by having a single inner iteration, the performance was significantly worse. The parameters may be more carefully analyzed using EXIT chart [62] analysis but this is outside the scope of this thesis and was left for future work. From the figure, it is clear that the proposed strategy can mitigate the effects of quantization and the performance is improved significantly especially for larger diagonal coupling. Note that even in the unquantized case of Fig. 4.3, the significant improvement was for larger diagonal coupling as well. When the system is ICI-limited as shown in 4.6d, the iterative strategy significantly improves the performance by several orders of magnitude.

Finally, it is worth noting that this iterative strategy can be used in the unquan-

tized case, however, at this stage we are interested in scenarios of practical relevance, which is why such analysis was omitted.

4.9 Summary

In this chapter, the problem of inferring the stored data in cells that are subject to noise and ICI was tackled without accessing the aggressor cells. The main objective was to approximate the channel model to reduce the complexity of computing the posterior PMFs. The naive approximation assumes that each group of $n' \ll N$ contiguous cells is independent of other groups and the data stored in it is inferred independently. It was shown by simulations, that both the coded and uncoded performances could be improved by increasing n'. However, this will rapidly become infeasible as the complexity increases exponentially with n'. In order to propose a more accurate approximation, the structure of ICI had to be better understood and exploited. It turns out that the exact representation of the flash system is a hidden Markov model with a continuous state space and the state of a cell is the ICI-free shifts in its aggressor. Accordingly, the states of adjacent cells are correlated due to common aggressors. The sum-product algorithm can be used to infer the stored data. However, it suffers from hard-to-compute messages due to the infinite number of states that a cell can have. To overcome the prohibitive complexity, the state of a cell was approximated by the inputs of its aggressors rather than their ICI-free shifts. As a consequence of this approximation, the state becomes discrete, and the sum-product can be run efficiently where the messages are look-up tables. Simulations of this approach showed significant improvement over the naive approach (even for $n' = 2$), especially when diagonal interference is significant. Finally, we presented and discussed the effect of quantization on the performance of the proposed approach. We concluded that the sum-product approach, being

the best in the unquantized case, is the most sensitive to quantization, due to the fact that quantization discards information about ICI. To alleviate the effect of quantization on the sum-product algorithm, we proposed an iterative strategy that alternate between detection and decoding several times and we showed preliminary results that this strategy can improve performance.

Chapter 5

Index Programming for Flash Memory

In this chapter, we propose a novel data representation scheme for flash memory to help alleviate some of its inherent adverse effects. Our scheme fixes the number of cells in a program level and conveys a first part of the information by their indices and a second part by their amplitudes. When the data is to be inferred, this information can be efficiently exploited using dynamic references to reduce the probability of error. This technique will partially mitigate the need to track the channel parameters, and the read references as the device degrades.

5.1 Introduction

As explained in section 2.4, the reliability of flash memory degrades as it gets older due to the electric stress of the program and erase operations. This degradation is reflected by a change in $p_{Y|X}(y|x)$ as the device ages. Consequently, the hard read references that discriminate between adjacent levels become sub-optimal for detecting the stored data and must be updated to ensure the lowest raw bit error rate. Several methods have been proposed to update $p_{Y|X}(y|x)$ and subsequently the read references [31–33, 37, 38, 50]. These methods assume a theoretical model of $p_{Y|X}(y|x)$, e.g., Normal, Normal-Laplace mixture, Student's-t, or as in this thesis, Gaussian mixture. Then by performing many sensing operations, the conditional histograms of Y given X are built and fitted to the assumed model by minimizing some cost function. Upon obtaining the most up to date distributions, the hard references that minimize the raw symbol error probability are computed and fixed until the next update. We refer to this approach as the *optimal fixed references* approach. Several challenges face the update process; first, it requires high-precision sensing of a large number of cells in order to build accurate histograms. Second, it may require known training data in order to build the conditional histograms, which may not be possible if user data is already stored in the block. Thirdly, the training phase cannot account for long-term effects such as retention. Finally, the chosen model of $p_{Y|X}(y|x)$ must be very simple in order to learn its parameters efficiently, which poses a fundamental limitation on the quality of the read references and the raw bit error rate.

To overcome these limitations, a different approach was suggested in [46] for SLC, and generalized in [47] for MLC. The approach requires that the *composition* of the stored sequence, i.e., the number of cells in each level, to be known to the

detector. Consequently, when reading data, a trial and repeat process is done by applying several references until the detected output's composition is the same as the true composition (see [46]). This detection using *dynamic references* is model agnostic and does not require any references to be estimated in advance. In short, the difference between fixed and dynamic references is that the former are costly to estimate but easy to use while the latter require no estimation, but using them incurs extra sensing at the time of reading. The read latency introduced by dynamic references may be tolerable for a single reference, i.e., SLC, but it might be prohibitive for a larger number of references, e.g., $q-1$ for $q$ary cells.

The knowledge of the composition may be obtained by having a record of the composition of each stored WL in a reliable (secondary) memory. Alternatively, a more systematic way is to use constant composition coding [63] that maps a sequence of unrestricted cell levels into a larger sequence with a pre-defined constant composition. The first method incurs no complexity but requires a reliable secondary memory, while the second method incurs complexity to encode and decode.

The authors of [46,47] have shown that for any given written pattern $\mathbf{X}$, dynamic detection produces a number of level (symbol) errors that is at most twice that of an hypothetical detector where the references are chosen to minimize the error rate (hypothetical because it is impossible to know without knowing the inputs $\mathbf{X}$). The authors then considered the asymptotic case ($n \to \infty$) where the dynamic references converge to the q references that divide the support of $p_Y(y)$ into q intervals with probabilities given by the assumed composition. Naturally, since these references are now fixed, their performance is sub-optimal compared to the optimal fixed references, but at least better than keeping the reference fixed throughout the lifetime of the device. The authors did not analyze the case of finite n.

For the finite case, the dynamic references are those which divide the read cells

into q groups such that the number of cells belonging to each group is prescribed by the composition. In contrast with the asymptotic case, the references now vary from one instance to another and the variation becomes smaller as n increases. Consequently, considering only the asymptotic case undermines the benefit of dynamic references and it is more interesting to analyze the case of finite n. To clarify this point, consider that $q = 2$ and we have a balanced composition, i.e., equal number of ones and zeros. The dynamic reference hence is the one that balances the composition, which is nothing but the sample median of the outputs $\mathbf{Y}$. The sample median is a random variable the variance of which decreases as n grows larger until it converges to the population median of the distribution $p_Y(y)$ which is fixed and no longer dynamic.

In this chapter, we avoid the prohibitive latency of dynamic detection for $q$ary cells by proposing a hybrid scheme where only the first reference is dynamic while the remaining references are fixed. In other words, we fix the number of cells in a program level (or equivalently in the erased level) which allows the use of a dynamic reference to identify the erased and programmed cells. This creates two variables (streams) where information can be stored, the first is in the indices of the programmed cells while the second in their actual amplitudes. We study the error rates in each of the two pieces of information in the finite n regime where the dynamic reference does not converge to a fixed one. We show that, in contrast with the results mentioned above, the new scheme with dynamic reference provides a performance improvement over optimal fixed refernces. We also uncover a trade-off between storage efficiency and resiliency to errors. We refer to this hybrid scheme as *index programming* (IP), inspired by the recent works on index modulation techniques for OFDM systems [64–66].

There are several motives to propose and study this scheme. First, several em-

pirical error analyses of MLC devices [51, 67] have shown that a vast majority of errors correspond to cells in programmed levels being detected in different programmed levels. Intuitively, by storing part of the data in the indices of the erased cells, this information will have higher resiliency to errors. In addition, since erased cells do not subject their neighboring cells to ICI, by increasing the proportion of erased cells, the effect of ICI can be reduced. Also, having to track $q-2$ optimal fixed references is easier then tracking $q-1$ references. Finally, the new scheme enables an interesting trade-off between storage efficiency and error resiliency and allows the use of a non-power-of-two number of levels, which is always a limitation in conventional flash memory.

5.2 Index Programming

5.2.1 Definition

Consider q-ary cells and a WL with n cells that are initially in the erased level (level 0). Under *unrestricted programming* (UP), any write pattern $\mathbf{X} \in \mathbb{Z}_q^n$ can be programmed into these cells allowing the storage of $\log_2 q$ bits per cell. On the other hand, Index programming restricts the number of allowed write patterns that can be written. In particular, it enforces that exactly k cells out of the n cells are in programmed level, i.e., in $\{1, ..., q-1\}$ while the remaining $n-k$ cells are left erased (level 0). We refer to the programmed cells as active and the erased ones as inactive. Let $\mathcal{C}(n,k) = \{\mathbf{c} \in \{0,1\}^n : \sum_{i=1}^n c_i = k\}$ be the set of all constant weight sequences of length n and weight k and $|\mathcal{C}(n,k)| = \binom{n}{k}$. Each element in $\mathcal{C}(n,k)$ is called an *activation pattern*, i.e., a configuration of which cells are active and which are inactive. In particular, a $'1'$ in the ith component indicates that the ith cell is active and it is inactive otherwise. Each active cell can be in one of the $q-1$ non-erased levels and vector $\mathbf{A} = (A_1, ..., A_k) \in \{1, ..., q-1\}^k$ represents the levels

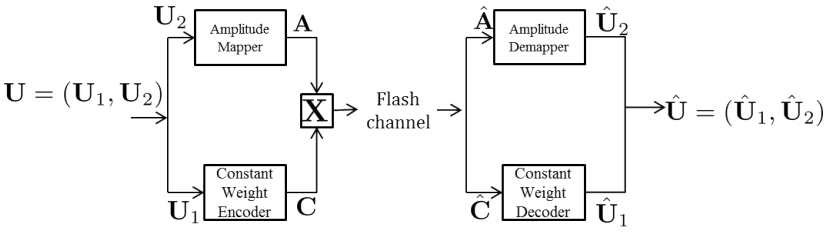


Figure 5.1: Diagram showing the index programming process

to which the active cells are set. The combination of $\mathbf{C}$ and $\mathbf{A}$ defines a valid write pattern $\mathbf{X}$ in which exactly k cells are programmed. The total number of valid write patterns is then $(q-1)^k\binom{n}{k}$. Fig. 5.1 explains the IP process, where an incoming bit-stream $\mathbf{U}$ is first divided into two sub-streams $\mathbf{U}_1$ and $\mathbf{U}_2$, of sizes $b_1 = \left\lfloor \log_2 \binom{n}{k} \right\rfloor$ and $b_2 = \lfloor k \log_2(q-1) \rfloor$ bits, respectively, such that $b = b_1 + b_2$ is the total number of bits. The stream $\mathbf{U}_1$, which we refer to as the *index stream* is then encoded by a constant weight encoder into an activation pattern $\mathbf{C} \in \mathcal{C}(n,k)$. The stream $\mathbf{U}_2$, which we refer to as the *amplitude stream* is mapped into $\mathbf{A} \in \{1, ..., q-1\}^k$ using a binary to $(q-1)$ary mapping. Finally, the write pattern $\mathbf{X}$, which is a deterministic function of $\mathbf{C}$ and $\mathbf{A}$, is constructed and programmed into the n cells. To read the two streams, estimates $\hat{\mathbf{C}}$ and $\hat{\mathbf{A}}$ of $\mathbf{C}$ and $\mathbf{A}$, respectively, are obtained by applying the adequate read references. Finally, these are mapped back to estimates $\hat{\mathbf{U}}_1$ and $\hat{\mathbf{U}}_2$ using the constant weight decoder and the amplitude demapper, respectively. The total number of bits that can be stored in n cells under IP can be bounded by removing the floors in b_1 and b_2 as follows

$$k \log_2 (q-1) \; + \; k \log_2 \binom{n}{k} \; - \; 2 \quad \leq \quad b \quad \leq \quad k \log_2 (q-1) \; + \; k \log_2 \binom{n}{k}. \quad (5.1)$$

Using Stirling's bounds on the factorial, ($\sqrt{2\pi}n^{n+\frac{1}{2}}e^{-n} \leq n! \leq en^{n+\frac{1}{2}}e^{-n}$), which is valid for all positive integers, (5.1) can be re-written as follows

$$b \leq \zeta n \log_2 (q-1) + nH_2(\zeta) - \log_2 \frac{2\pi}{e}\sqrt{\zeta n (1-\zeta)}, \tag{5.2}$$

$$b \geq \zeta n \log_2 (q-1) + nH_2(\zeta) - \log_2 \frac{e^2}{\sqrt{2\pi}}\sqrt{\zeta n (1-\zeta)} - 2, \tag{5.3}$$

where $\zeta = k/n \in (0,1)$ represents the ratio of the number of programmed cells to all available cells and $H_2(p) = -p\log_2 p - (1-p)\log_2(1-p)$ is the binary entropy function.

In this work, we will not discuss the constant weight encoder and decoder operations, but we will use them as black boxes. Constant weight coding is a mature field, and many references exist on how such an operation can be done. In particular, we mention the works in [68–74] and the references therein.

5.2.2 Illustrative Examples

5.2.2.1 $q = 2$

Assume a memory with $n = 4, q = 2$ and $k = 2$. Each cell can either be in the erased level (0) or the programmed level (1). There are $\binom{4}{2} = 6$ ways to program only two out of the four cells (activation patterns). As a result, the possible write patterns that can be written into the four cells are $(1,1,0,0)$, $(1,0,1,0)$, $(1,0,0,1)$, $(0,1,0,1)$, $(0,0,1,1)$, and $(0,1,1,0)$. Hence a maximum of $\log_2(6) \simeq 2.58$ bits can be stored in these 4 cells.

5.2.2.2 $q = 3$

Now consider the same parameters but with $q = 3$, i.e., a cell can be either in the erased (0) or in one of two programmed levels (1 and 2). There are still six possible activation patterns, but each programmed cell now has several possible

states to which it can be programmed. This gives an additional degree of freedom. For example, if the first and third cells are active, there are four possible write patterns when $q = 3$: $(1, 0, 1, 0)$, $(1, 0, 2, 0)$, $(2, 0, 1, 0)$, $(2, 0, 2, 0)$. For each of the other five activation patterns, there are four write patterns, and we can store up to $\log_2(6 \times 4) \simeq 4.58$ bits in four cells.

5.2.3 Storage Efficiency

Storage efficiency is defined as the average number of bits that can be stored in a cell according to a specific scheme when the number of cells goes to infinity. Without IP, $\log_2 q$ bits can be stored in each cell independently from n. It is easy to see that by dividing the bounds in (5.2) by n and letting $n \to \infty$ the bounds meet and the storage efficiency of IP can be written as follows

$$\eta = \lim_{n\to\infty} \frac{b}{n} = H_2(\zeta) + \zeta \log_2(q-1), \tag{5.4}$$

The final expression has two parts. Intuitively, it can be seen that the first term relates to the information conveyed by the activation pattern since a cell is active with probability ζ. The second term relates to the information stored in the actual levels of the programmed cells, that is $\mathbf{a}$. In order to maximize the information conveyed by the activation pattern, it is clear that one should program exactly half the cells (i.e. $k = n/2$) to maximize the number of possible activation patterns (and to maximize $H_2(\zeta)$). However, this does not maximize η. Maximizing η can be done by differentiating η with respect to ζ and solving for ζ the equation $\frac{d\eta}{d\zeta} = 0$, the result is then

$$\zeta_0 = \frac{q-1}{q}, \tag{5.5}$$

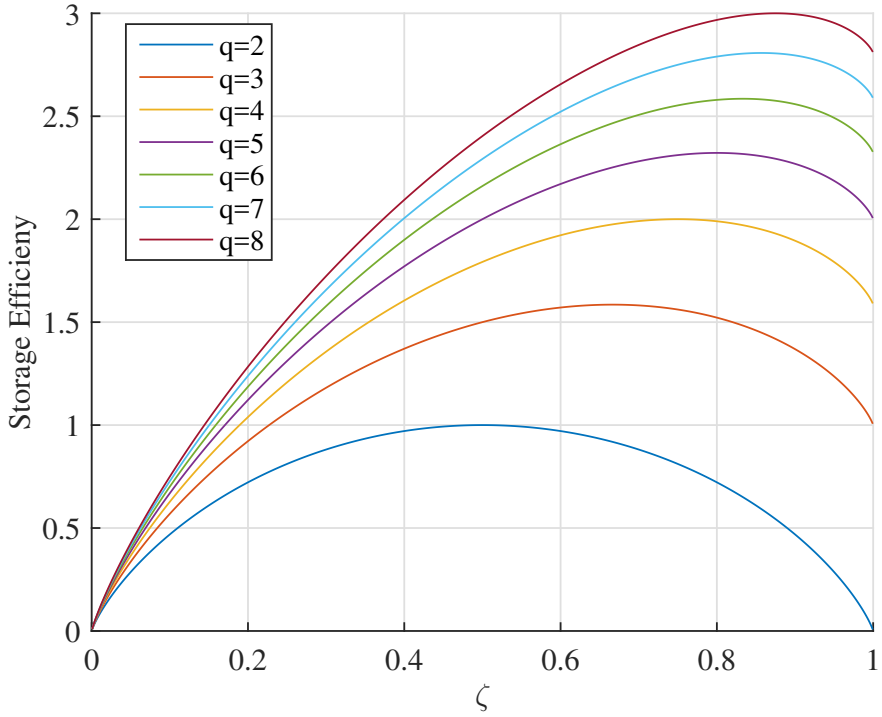


Figure 5.2: Plot of the asymptotic storage efficiency of IP (eq.(5.4)) as a function of ζ for different number of levels.

giving a maximum of

$$\eta_0 = \log_2 q \text{ bits/cell.} \tag{5.6}$$

This maximum η_0 is equal to the storage efficiency of traditional unrestricted programming. Fig. 5.2 shows the plot of η (eq. (5.4)) as a function of ζ for different numbers of levels. It is important to note that the storage efficiencies plotted in Fig. 5.2 assume $n \to \infty$ as opposed to unrestricted programming where the efficiency is $\log_2 q$ no matter how many cells are considered. For the example given in the previous section ($q = 3$ and $\zeta = 0.5$), the efficiency was only 1.145 bits/cell while from the Fig. 5.2 we see that the efficiency can reach 1.5 bits/cell for $n \to \infty$.

The observed reduction in storage efficiency relative to unrestricted programming in the IP scheme comes with the advantage of reduced ICI and increased resiliency

as we will show in sections 5.3 and 5.4.

5.3 ICI Reduction Under IP

In this section, we consider two ICI models. The first one is the general model already presented in 2.4 where each voltage shift in a neighboring cell creates interference in the victim cell. The second is more simplified and is frequently used in the context of constrained coding for ICI mitigation. For the first, we analyze the reduction in the average ICI, while for the second, we analyze the reduction in the probability of occurrence of a specific unwanted pattern.

5.3.1 General Interference Model

According to (2.5), the average ICI that a cell suffers from is given by

$$\begin{aligned}
\mathbb{E}[C] &= \sum_{j=1}^{m} \gamma_j \mathbb{E}[\Delta Y_j^a] \\
&= \mathbb{E}[\Delta Y^a]\Big(\sum_{j=1}^{m} \gamma_j\Big) \\
&= \Big(\sum_{x^a=0}^{q-1} p_X(x^a)\mathbb{E}[\Delta Y^a | X^a = x^a]\Big)\Big(\sum_{j=1}^{m} \gamma_j\Big) \\
&= \Big(\sum_{x^a=1}^{q-1} p_X(x^a)\mathbb{E}[\Delta Y^a | X^a = x^a]\Big)\Big(\sum_{j=1}^{m} \gamma_j\Big), \qquad (5.7)
\end{aligned}$$

where the second equality is due to the fact that the voltage shift in a cell is independent of its position, the third equality is due to the law of total expectation and the fourth equality is due to the fact that erased cells do not cause any ICI. What differentiates IP from unrestricted programming is the distribution of the input $p_X(x^a)$. In the case of unrestricted programming, the input is assumed i.i.d and equally likely, i.e., $p_X(i) = 1/q$ for all $i \in \{0, ..., q-1\}$, while in the case of IP, we

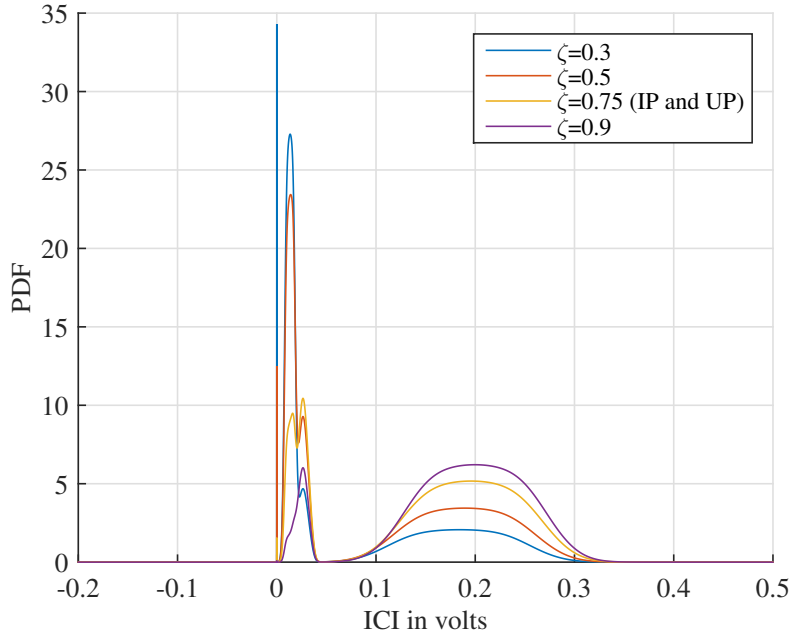
have that $p_X(0) = \frac{n-k}{n}$ and $p_X(i) = \frac{k}{n(q-1)}$ for $i \in \{1, ..., q-1\}$. Consequently, we can write the ratio ξ between the expected ICI of IP and the unrestricted case as follows

$$\xi = \frac{kq}{n(q-1)} \frac{\sum_{x^a=1}^{q-1} \mathbb{E}[\Delta Y^a | X^a = x^a](\sum_{j=1}^{m} \gamma_j)}{\sum_{x^a=1}^{q-1} \mathbb{E}[\Delta Y^a | X^a = x^a](\sum_{j=1}^{m} \gamma_j)} = \zeta \frac{q}{(q-1)}. \tag{5.8}$$

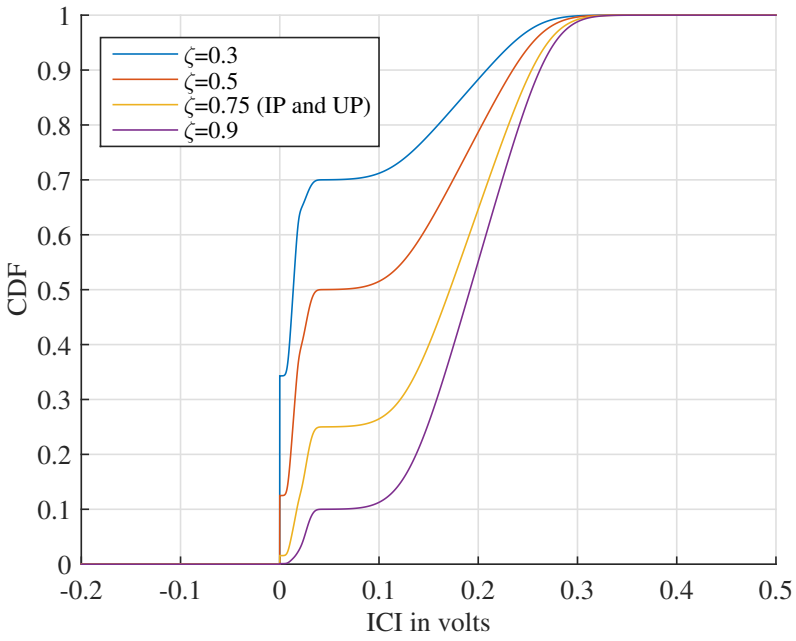
In order to observe a reduction in the average ICI under the IP scheme, ξ should be less than 1. This condition can be rewritten as

$$\zeta \leq \frac{q-1}{q}. \tag{5.9}$$

When (5.9) is met with equality, the average ICI is not reduced. However, as we have seen in (5.6), this value of ζ maximizes the storage efficiency. Hence, there is a trade-off between maximizing storage efficiency and reducing the average ICI. Chip designers should be able to exploit this trade-off to achieve a specified storage efficiency/robustness goal in practice. To demonstrate further the effect of IP on ICI, Fig. 5.3 shows the PDFs and CDFs of ICI for IP and UP for different values of ζ. From (3.3), the distribution of ICI is a Gaussian mixture. We use the same parameters in Table 3.2, and we set $\gamma_v = 0.08$ and $\alpha = 0.075$. It is obvious that the interval in which the majority of the distribution is confined becomes larger as ζ increases. Furthermore, the scaled Dirac delta distribution at zero indicates the chance of no interference, which is the case when the three interfering cells are kept erased. This indicates the increase in the severity of ICI (uncertainty about it). For $\zeta = (q-1)/q$ (yellow) the distributions of ICI are the same for IP and UP. For $\zeta > (q-1)/q$, ICI becomes more severe in IP than in UP since now more cells on average are programmed and hence contribute to the interference. In general,



(a) PDF, $q = 4$



(b) CDF, $q = 4$

Figure 5.3: The distributions of ICI in the proposed IP schemes for different values of ζ and $q = 4$, $\zeta = 0.75$ corresponds also to ICI in UP.

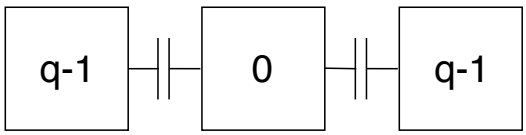


Figure 5.4: Diagram showing the ICI prone pattern where the middle cell is suceptible to high ICI shift.

when IP is employed, with (5.9) satisfied, there is a larger probability that the system exhibits a lower ICI. As mentioned in the introduction, narrowing the ICI distribution will help improve the performance from an error rate perspective, which is exactly what increasing the fraction of erased cells achieves.

5.3.2 Bit-line Interference Model

Consider the following simplistic model: the only source of errors is bit-line interference, i.e., only adjacent cells on the same bit-line interfere with each other. Furthermore, we assume that an error occurs only when an erased cell (level 0) falls between two cells that are on the highest level $(q-1)$. This model is rather simplistic as ICI is not the only source of errors, and the mentioned pattern is not the only pattern to create ICI errors, but this model is often used in the literature of constrained coding for memories [45]. In the case of unrestricted programming with equally likely input, the probability that the mentioned pattern occurs is $1/q^3$, i.e., three consecutive word-lines having levels $q-1, 0$, and $q-1$, respectively on the same bit-line. In the case of IP (assuming the same number of levels q), the probability of occurrence of the pattern is $\zeta^2(1-\zeta)/(q-1)^2$ which is maximized for $\zeta = 2/3$. It is impossible to reduce this probability with unrestricted programming without increasing the number of levels q. However, in the case of IP, for a fixed number of levels, this probability can go to 0 by making ζ go to either 0 or 1[1]. This is possible in practice because of the huge number of cells in a word-line. From Fig. 5.2, it is clear

[1] if $\zeta = 1$, the system transforms into UP with $q-1$ states, which is useless. Hence, ζ should always be less than 1.

that letting ζ go to 0 results in a zero storage efficiency for all q, while ζ going to 1 preserves a non-zero storage efficiency for $q > 2$. For example, consider a word-line with $n = 1000$ cells, $\zeta = 0.999$ and $q = 4$. The probability of occurrence in the IP case is 0.00011 while that of the unrestricted scheme is 0.0156, i.e., approximately two orders of magnitude reduction in the probability of occurrence. On the other hand, the storage efficiency of the IP scheme, in this case, is 1.592 bits/cell, a 20.4% reduction from the traditional 2 bits/cell for MLC.

Finally, the ratio of the two probabilities of occurrence is $\zeta^2 q^3 (1-\zeta)/(q-1)^2$. When q is greater or equal to 4, $\zeta^2 q^3 (1-\zeta)/(q-1)^2 \leq 1$ requires either $\zeta \geq (q-1)/q$, or $\zeta \leq \left(\sqrt{4q-3}+1\right)/2q$. It is always possible to have an ζ that reduces the probability of occurrence of the unwanted pattern. Remember that the maximum storage efficiency of the IP scheme is achieved at $\zeta = (q-1)/q$, hence the emergence of the trade-off: by sacrificing some of the storage efficiency, the probability of occurrence of the ICI prone pattern can be reduced. Under this model, the condition of $\zeta \geq (q-1)/q$ is the opposite of the condition under which the average ICI is reduced in (5.9) for the model considered in the previous subsection.

5.4 Error Analysis of IP

In this section, we are interested in the probability of errors in the the detected index and amplitude streams, i.e., $\Pr(\hat{\mathbf{C}} \neq \mathbf{C})$ and $\Pr(\hat{\mathbf{A}} \neq \mathbf{A})$, respectively. In order to make the problem mathematically tractable, we assume that the outputs of the cells are independent given their inputs, i.e., $p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} p_{Y|X}(y_i|x_i)$. This is in contrast with our previous result that showed that they actually are dependent due to common aggressors. However, as we shall see from the simulation results, for the ICI parameters we consider, this approximation is accurate.

5.4.1 Read References

In an unrestricted programming scheme, the $q-1$ optimal read references denoted $(t_1^*, ..., t_{q-1}^*)$, are chosen to minimize the probability of symbol error. These references can also be used to detect the index and amplitude when IP is employed. This method of detecting IP, which we refer to as *fixed references* detection, consists of applying t_1^* to distinguish between the erased and programmed cells, and therefore estimate $\mathbf{C}$. The remaining $q-2$ references are applied to distinguish between the programmed levels and hence estimate $\mathbf{A}$.

However, this method ignores the prior knowledge that there are precisely k programmed cells. Since t_1^* is fixed, there is no guarantee that there will be exactly k programmed cells, i.e., with levels above t_1^*. In this case, it will not be possible to retrieve both the index and amplitude streams and an error in retrieving the data will be declared. To guarantee that precisely k programmed cells are detected, the first reference is made dynamic so that exactly k cells are above it and subsequently $\mathbf{C}$ is estimated. Hence, the first read reference need not be estimated in advance, and does not depend on any model $p_{Y|X}(y|x)$, but is obtained by trial and repeat. On the other hand, the amplitude stream still has to be detected using the fixed $q-2$ references $t_2^*, ..., t_{q-1}^*$, and hence the IP scheme solves the problem of updating the references only partially for a part of the stored data. We call this detection scheme the *hybrid references* detection, and we will show that by choosing dynamic detection of the index stream, the error rate in the index stream is significantly reduced.

5.4.2 Detection of UP

Under unrestricted programming, the joint distribution of the input of a WL can be written as $p_{\mathbf{X}}(\mathbf{x}) = \prod_{i=1}^{n} p_X(x_i)$, i.e., the inputs are i.i.d. This is different

from the case of IP where the elements of $\mathbf{X}$ are correlated by enforcing the number of programmed cells to be k. The optimal fixed read references are determined by minimizing the symbol error probability, i.e., $\Pr(\hat{X} \neq X) = 1 - \Pr(\hat{X} = X)$ where $\hat{X}$ is the detected level. The probability of correct detection using arbitrary references $t_1, ..., t_{q-1}$ ($t_0 = -\infty$ and $t_q = +\infty$ by convention) can be written as follows

$$\begin{aligned}
\Pr\left(\hat{X} = X\right) &= \sum_{j=0}^{q-1} p_X(j)\, p_{\hat{X}|X}(j|j) \\
&= \sum_{j=0}^{q-1} p_X(j) \left(P_{Y|X}(t_{j+1}|j) - P_{Y|X}(t_j|j)\right) \\
&= p_X(q-1) \\
&\quad + \sum_{j=1}^{q-1} \left(p_X(j-1)\, P_{Y|X}(t_j|j-1) - p_X(j)\, P_{Y|X}(t_j|j)\right),
\end{aligned} \tag{5.10}$$

where $P_{Y|X}(y|x)$ is the conditional CDF of the output of a cell given its input and where $P_{Y|X}(t_q|q-1) = 1$ and $P_{Y|X}(t_0|0) = 0$. To maximize this probability with respect to the read references, each summand of the summation can be maximized individually, because it depends only on a single reference, therefore the optimal read reference t_j^* is obtained by setting the derivative of $\Big(p_X(j-1)\, P_{Y|X}(t|j-1) - p_X(j)\, P_{Y|X}(t|j)\Big)$ to zero and solving for t, which leads to the following equation

$$p_X(j-1)\, p_{Y|X}\left(t_j^*|j-1\right) = p_X(j)\, p_{Y|X}\left(t_j^*|j\right). \tag{5.11}$$

Hence, the optimal references are at the abscissas of the intersection points between the scaled PDFs of adjacent levels. Depending on the functions, these can be obtained either analytically or numerically. We denote by $\epsilon_{ij} = p_{\hat{X}|X}(i|j)$ the probability that a cell originally in level j is detected in level i using the optimal references, and

it can be written as

$$\epsilon_{i,j} = \int_{t_i^*}^{t_{i+1}^*} p_{Y|X}(y|j)dy = P_{Y|X}\left(t_{i+1}^*|j\right) - P_{Y|X}\left(t_i^*|j\right). \tag{5.12}$$

We write that $\Pr\left(\hat{X} = X\right) = \sum_{j=0}^{q-1} p_X\left(j\right)\epsilon_{j,j}$. Since the outputs of the cells are assumed independent, these events occur independently and identically to all cells (conditioned on their input). Hence the write pattern error rate can be written as

$$\Pr\left(\hat{\mathbf{X}} \neq \mathbf{X}\right) = 1 - \Big(\sum_{j=0}^{q-1} p_X(j)\epsilon_{j,j}\Big)^n \tag{5.13}$$

Under UP, the stored data is divided into $\log_2 q$ pages (streams). For a specific binary page, e.g., LSB or MSB in MLC, half the levels ($q/2$) correspond to a binary 0 and the other half to a binary 1. If we denote by $\mathcal{L}_0 \subset \mathbb{Z}_q$ where $|\mathcal{L}_0| = q/2$ the levels that corresponds to a binary 0 and by $\mathcal{L}_1 = \mathbb{Z}_q \backslash \mathcal{L}_0$ those corresponding to a binary 1, the probability that a bit, denoted X_b, of that page is detected correctly can be written as follows

$$\begin{aligned}\Pr\left(\hat{X}_b = X_b\right) = \Pr\left(X \in \mathcal{L}_0\right)\Pr\left(\hat{X} \in \mathcal{L}_0 | X \in \mathcal{L}_0\right) \\ + \Pr\left(X \in \mathcal{L}_1\right)\Pr\left(\hat{X} \in \mathcal{L}_1 | X \in \mathcal{L}_1\right)\end{aligned} \tag{5.14}$$

where for $z = 0, 1$

$$\begin{aligned}
\Pr\left(\hat{X} \in \mathcal{L}_z | X \in \mathcal{L}_z\right) &= \sum_{i \in \mathcal{L}_z} \Pr\left(\hat{X} = i | X \in \mathcal{L}_z\right) \\
&= \sum_{i \in \mathcal{L}_z} \frac{\Pr\left(\hat{X} = i, X \in \mathcal{L}_z\right)}{\Pr\left(X \in \mathcal{L}_z\right)} \\
&= \sum_{i \in \mathcal{L}_z} \frac{\sum_{j \in \mathcal{L}_z} \Pr\left(\hat{X} = i, X = j\right)}{\Pr\left(X \in \mathcal{L}_z\right)} \\
&= \frac{1}{\Pr\left(X \in \mathcal{L}_z\right)} \sum_{i,j \in \mathcal{L}_z} p_X\left(j\right) p_{\hat{X}|X}\left(i|j\right) \qquad (5.15)
\end{aligned}$$

Therefore, we can write

$$\Pr\left(\hat{X}_b = X_b\right) = \sum_{i,j \in \mathcal{L}_0} p_X\left(i\right) \epsilon_{i,j} + \sum_{m,l \in \mathcal{L}_1} p_X\left(m\right) \epsilon_{m,l}, \qquad (5.16)$$

where in unrestricted programming, it is common that the input levels are equally likely, hence $p_X(i) = 1/q$ for $i \in \mathbb{Z}_q$. Consequently, the error rate of the considered page can be determined as follows

$$\begin{aligned}
\Pr\left(\hat{\mathbf{X}}_b \neq \mathbf{X}_b\right) &= 1 - \Pr\left(\hat{\mathbf{X}}_b = \mathbf{X}_b\right) \\
&= 1 - \left(\Pr\left(\hat{X}_b = X_b\right)\right)^n \qquad (5.17)
\end{aligned}$$

For the case of MLC ($q = 4$), with Gray mapping, we get two pages with different page error rates that can be calculated according to the above analysis. The index and amplitude sub-streams of IP can be seen as analogous to the MSB and LSB streams (pages) in MLC when unrestricted programming is employed. Therefore, in the next section, we compare the performance of the IP streams (pages) with those of the MSB and LSB streams in unrestricted programming with equally likely levels.

5.4.3 IP Detection Using the Fixed References Scheme

Detecting IP using fixed references is equivalent to detecting UP. Once the input distribution of either schemes is known, the optimal fixed references can be obtained along with the transition probabilities $\epsilon_{i,j}$. The activation pattern is detected correctly only when the k programmed cells are detected as programmed and the $n-k$ erased cells are detected as erased. Hence, the probability of an error in the index stream can be written as

$$\begin{aligned}\Pr(\hat{\mathbf{C}} \neq \mathbf{C}) &= 1 - \Big(p_{\hat{X}|X}(0|0)\Big)^{n-k}\Big(1 - \frac{1}{q-1}\sum_{j=1}^{q-1} p_{\hat{X}|X}(0|j)\Big)^{k} \\ &= 1 - \epsilon_{0,0}^{n-k}\Big(1 - \frac{1}{q-1}\sum_{j=1}^{q-1}\epsilon_{0,j}\Big)^{k} \end{aligned} \tag{5.18}$$

It is worth noting that the sum $\sum_{j=1}^{q-1}\epsilon_{0,j}$ is dominated by $\epsilon_{0,1}$ due to the fact that the further away are the input and output levels, the less likely of such event to occur. The average bit error rate in the activation pattern $\mathbf{C}$ can be written as follows

$$\begin{aligned}\Pr(\hat{C} \neq C) &= 1 - \Pr(\hat{C} = C) \\ &= 1 - \sum_{j=0}^{1} p_C(j)\Pr(\hat{C} = j | C = j) \end{aligned} \tag{5.19}$$

where $p_C(0) = (1-\zeta)$, $p_C(1) = \zeta$, $\Pr(\hat{C} = 0|C = 0) = \epsilon_{0,0}$, and

$$\begin{aligned}\Pr(\hat{C} = 1|C = 1) &= 1 - \Pr(\hat{C} = 0|X \in \mathbb{Z}_q \backslash 0) \\ &= 1 - \frac{\sum_{j=1}^{q-1} \Pr(\hat{X} = 0, X = j)}{\Pr(X \in \mathbb{Z}_q \backslash 0)} \\ &= 1 - \zeta \sum_{j=1}^{q-1} \epsilon_{0,j} \end{aligned} \tag{5.20}$$

Using the total law of probability, the probability of error in the amplitude stream, $\Pr(\hat{\mathbf{A}} \neq \mathbf{A})$ can be upper bounded as follows

$$\begin{aligned}\Pr(\hat{\mathbf{A}} \neq \mathbf{A}) &= 1 - \Pr(\hat{\mathbf{A}} = \mathbf{A}, \hat{\mathbf{C}} = \mathbf{C}) - \Pr(\hat{\mathbf{A}} = \mathbf{A}, \hat{\mathbf{C}} \neq \mathbf{C}) \\ &\leq 1 - \Pr(\hat{\mathbf{A}} = \mathbf{A}, \hat{\mathbf{C}} = \mathbf{C}) = 1 - \Pr(\hat{\mathbf{X}} = \mathbf{X}). \end{aligned} \tag{5.21}$$

Assuming that it is highly unlikely that the amplitude vector is detected correctly when the activation pattern is detected in error, the term $\Pr(\hat{\mathbf{A}} = \mathbf{A}, \hat{\mathbf{C}} \neq \mathbf{C})$ can be assumed negligible and the bound is tight. Our simulations indeed showed that $\Pr(\hat{\mathbf{A}} \neq \mathbf{A}) \approx \Pr(\hat{\mathbf{X}} \neq \mathbf{X})$ and hence in our results we use the latter as the theoretical and the former as the simulated.

5.4.4 IP Detection Using the Hybrid References Scheme

For $\ell \in \{1, ..., n-k\}$, we denote by $Y^0_{(\ell)}$ the ℓth order statistic of the $n-k$ erased cells. Similarly, for $l \in \{1, ..., k\}$ we denote by $Y^1_{(l)}$ the lth order statistic of the k programmed cells. When the dynamic reference scheme is used, it is guaranteed that k cells will be detected as programmed. Therefore, the activation pattern is detected correctly only when the maximum output among the erased cells is less than or equals the minimum output among the programmed cells, hence we can

write

$$\Pr(\hat{\mathbf{C}} = \mathbf{C}) = \Pr\left(Y^0_{(n-k)} \leq Y^1_{(1)}\right). \tag{5.22}$$

Due to the assumption that the outputs are independent, the outputs of the erased cells are i.i.d according to $p_{Y|X}(y|0)$. Similarly, the outputs of the programmed cells are i.i.d according to $\frac{1}{q-1}\sum_{j=1}^{q-1} p_{Y|X}(y|j)$. Accordingly, from i.i.d order statistics, it is possible to obtain the CDF of $Y^0_{(n-k)}$ and the PDF of $Y^1_{(1)}$, denoted $P_{Y^0_{(n-k)}}(y)$ and $p_{Y^1_{(1)}}(y)$, respectively, as follows [75]

$$P_{Y^0_{(n-k)}}(y) = \Big(P_{Y|X}(y|0)\Big)^{n-k} \tag{5.23}$$

$$p_{Y^1_{(1)}}(y) = k\Big(\frac{1}{q-1}\sum_{j=1}^{q-1} p_{Y|X}(y|j)\Big)\Big(1 - \frac{1}{q-1}\sum_{j=1}^{q-1} P_{Y|X}(y|j)\Big)^{k-1} \tag{5.24}$$

Consequently, using the total law of probability we can write

$$\Pr(\hat{\mathbf{C}} \neq \mathbf{C}) = 1 - \int_{-\infty}^{+\infty} p_{Y^1_{(1)}}(y) P_{Y^0_{(n-k)}}(y) dy. \tag{5.25}$$

The derivation of the average bit error rate in the activation pattern is more involved, since the dynamic detection creates a dependence between the detected bits. In particular, since the detected $\hat{\mathbf{C}}$ will always have a constant weight k, the bit flips come in pairs, i.e., for every programmed cell detected as erased, there is an erased cell that is detected as programmed. In general, in order to have exactly $2f$ bits in error (f zero flips and f ones flip) the following event should be satisfied

$$\mathrm{E}_{2f} : Y^0_{(n-k-f+1)} > Y^1_{(f)} \cap Y^0_{(n-k-f)} < Y^1_{(f+1)}, \tag{5.26}$$

in other words, to guarantee that exactly $2f$ bits are flipped, the f erased cells with the highest V_{th} should all have voltages higher than the f lowest V_{th} of the programmed cells, while the erased cell with the $(n-f)$th lowest V_{th} should be less than the V_{th} of the programmed cell with the $(f+1)$th lowest voltage. By decomposing $\Pr\left(Y^0_{(n-k-f+1)} > Y^1_{(f)}\right)$ using the total law of probability, we can write

$$\begin{aligned}\Pr\left(\mathrm{E}_{2f}\right) &= \Pr\left(Y^0_{(n-k-f+1)} > Y^1_{(f)}\right) - \Pr\left(Y^0_{(n-k-f+1)} > Y^1_{(f)} \cap Y^0_{(n-k-f)} \geq Y^1_{(f+1)}\right) \\ &= \Pr\left(Y^0_{(n-k-f+1)} > Y^1_{(f)}\right) - \Pr\left(Y^0_{(n-k-f)} \geq Y^1_{(f+1)}\right) \end{aligned} \tag{5.27}$$

The average bit error rate in the activation pattern when dynamic reference is used, denoted e_d can be written as follows:

$$e_d = \frac{1}{n}\sum_{f=1}^{f_m}(2f)\Pr\left(\mathrm{E}_{2f}\right), \tag{5.28}$$

where $f_m = \min\left(k, n-k\right)$. Using the relation in (5.27), it can be shown that

$$\sum_{f=1}^{f_m} f\Pr\left(\mathrm{E}_{2f}\right) = \sum_{f=1}^{f_m}\Pr\left(Y^0_{(n-k-f+1)} > Y^1_{(f)}\right) - \sum_{j=1}^{f_m} j\Pr\left(Y^0_{(n-k-j)} = Y^1_{(j+1)}\right), \tag{5.29}$$

where $\Pr\left(Y^0_{(n-k-f+1)} > Y^1_{(f)}\right)$ is the probability that at least f flips occurs. Note that the rightmost is equal to zero because the variables are continuous (zero probability measure). Furthermore, it is easy to see that the terms of the first summation of the RHS decrease with f since $Y^0_{(n-k-f)} > Y^1_{(f+1)}$ necessarily implies that $Y^0_{(n-k-f+1)} > Y^1_{(f)}$ is satisfied. Therefore, it is possible to lower bound the average bit error rate by approximating the sum with its first few terms. When noise is low, the first term will be sufficient to approximate the average bit error. The terms $\Pr\left(Y^0_{(n-k-f+1)} > Y^1_{(f)}\right)$ can be obtained using order statistics the same way we have done for (5.25).

The probability of error-free detection of the amplitude stream under dynamic reference scheme can be obtained by approximating it by the error-free write pattern detection rate as we have done in the case of fixed reference, i.e., $\Pr(\hat{\mathbf{A}} = \mathbf{A}) \approx \Pr(\hat{\mathbf{X}} = \mathbf{X})$ which can be written as follows

$$\begin{aligned} \Pr(\hat{\mathbf{X}} = \mathbf{X}) &= \Pr(\hat{\mathbf{A}} = \mathbf{A}, \hat{\mathbf{C}} = \mathbf{C}) \\ &= \Pr(\hat{\mathbf{C}} = \mathbf{C}) \Pr(\hat{\mathbf{A}} = \mathbf{A} | \hat{\mathbf{C}} = \mathbf{C}) \end{aligned} \tag{5.30}$$

where $\Pr(\hat{\mathbf{A}} = \mathbf{A}|\hat{\mathbf{C}} = \mathbf{C})$ is equivalent to considering the write pattern success rate of a UP scheme with $q-1$ equally likely levels and k cells. In the next section, we analyze the error rate for the UP scheme.

5.5 Simulations

In this section, we suggest a new metric to ensure fairness when comparing the different schemes, that takes into consideration power, noise and storage efficiency. Then we present and discuss the performance results of the index and amplitude streams of IP for several sets of parameters as well as the performance of UP. The expressions derived in the previous section are computed numerically and presented along with their simulated counterparts.

5.5.1 Average Energy per Bit to Noise Ratio

In both the unrestricted programming and the IP schemes, a memory block is initially in the erased state. We assume that in order to program a cell into any state $j \in \{1, ..., q-1\}$, an amount of energy $\mathrm{E}_j = |v_j - v_0|^2$ is required. This amount reflects the fact that more energy is needed to program cells into higher voltages. Since the unrestricted scheme uses equally likely levels, the average energy

consumption per cell is given by

$$\bar{\mathrm{E}}_U = \frac{1}{q}\sum_{j=1}^{q-1}\mathrm{E}_j. \tag{5.31}$$

On the other hand, the IP scheme changes the distribution of the input and the average energy consumption per cell is given by:

$$\bar{\mathrm{E}}_I = \frac{\zeta}{q-1}\sum_{j=1}^{q-1}\mathrm{E}_j. \tag{5.32}$$

Finally, we define a parameter that encompasses the energy consumption and the storage efficiency as well as the noise power. We call this parameter the average energy-per-bit-to-noise ratio (AEBNR) and define it as follows:

$$\bar{\gamma}_U = \frac{1}{\lfloor \log_2 q \rfloor}\frac{\bar{\mathrm{E}}_U}{\bar{\sigma}^2}, \tag{5.33}$$

$$\bar{\gamma}_I = \frac{n}{b_1+b_2}\frac{\bar{\mathrm{E}}_I}{\bar{\sigma}^2}. \tag{5.34}$$

where $\bar{\sigma}^2 = \frac{1}{q}\sum_{x=0}^{q-1}\sigma_x^2$ is the noise power in the system. This parameter, expressed in dB, will allow a fair comparison between the performance of the IP and UP schemes as it takes into account their storage efficiency and the energy requirements.

5.5.2 Results

To simulate the proposed IP scheme we set $n = 16383$ cells on each word-line, $q = 4$ and use again the same parameters used in the previous chapters and summarized in Table 3.2, we fix $\gamma_v = 0.08$ and $\alpha = 0.075$ and vary the AEBNR (in dB) by varying the noise power control β.

We start with setting $\zeta = 0.75$, which is the case where the levels are equally likely. However, the number of erased cells is deterministically (not statistically)

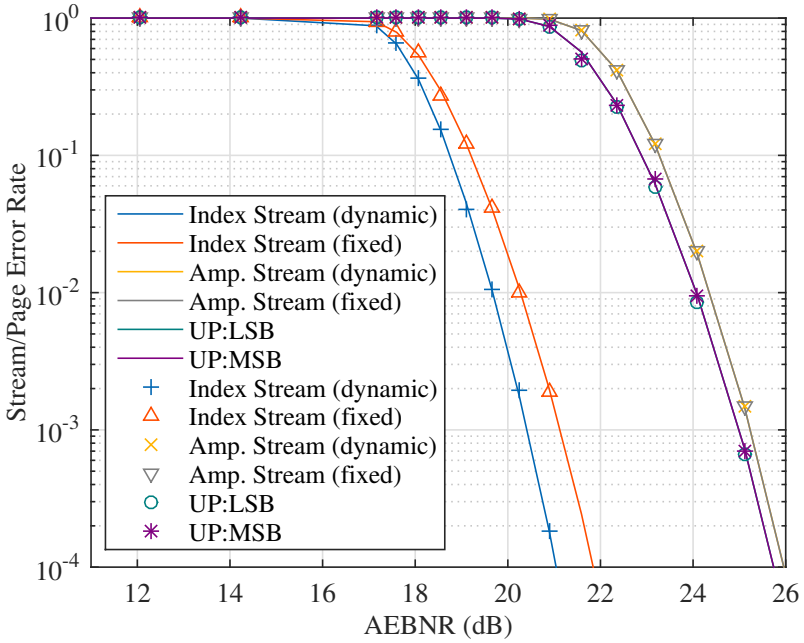


Figure 5.5: The stream/page error rates as functions of AEBNR, where solid lines represent the theoretical plots and markers their simulated counterparts, for $\zeta = 0.75$ and $\eta = 1.99957$ bits/cell. In this case, the UP:LSB and UP:MSB plots are coincident. The Amp. Stream (dynamic) and Amp. Stream (fixed) plots are also coincident.

equal to $n-k$. The page/stream error rate for the different pages/stream are shown in Fig. 5.5 as functions of the AEBNR where the dotted lines are the theoretical plots obtained analytically in the previous subsections, and the markers are the simulated results. Note that while the theoretical results assume that cells are independent, the simulations account for diagonal interference and accordingly cells are correlated. However, as we shall see, the diagonal coupling in our simulation is small enough to make the effect of correlation negligible on the uncoded performance. The error rate of the MSB and LSB pages when UP is employed with equally likely inputs are shown for reference.

The first observation is that for IP, the index stream (which stores 0.657 bits/cell) is much more resilient to errors than its amplitude counterpart with more than 4 dB difference at an error rate of 10^{-2}. The second observation is the superior

performance of the dynamic reference when reading the index stream compared to the optimal fixed reference with a 0.6 dB difference at an error rate of 10^{-2}. This is in contrast with the results cited in the introduction of this chapter [46, 47] where dynamic reference where shown to be sub-optimal in the asymptotic case ($n \to \infty$). In the case of practical relevance, i.e., finite n, the dynamic reference performs better than the optimal fixed reference. At an intuitive level, if the fixed reference can correctly detect the index stream, then the dynamic reference can also detect it, but the opposite is not always true. It is important to recall that the dynamic reference does not require any knowledge about the channel model or complex fitting methods as opposed to the optimal fixed read reference that requires regular updates as it rapidly becomes sub-optimal. The third observation is that the amplitude stream performs the same regardless of whether the index stream is detected using the dynamic reference or the fixed one. The final observation is that the MSB and LSB pages of the UP scheme perform the same and their performance is slightly better than the amplitude stream of the IP scheme with 0.25 dB difference at 10^{-2} error rate, but significantly worse than the index stream with 3.75 dB difference when fixed reference is used and 4.35 dB when dynamic reference is used.

Typically, manufacturers tend to enlarge the distance between the nominal voltages of the erased and first programmed levels compared to that between any two consecutive program levels. This is to balance the errors since the erase process has by nature a large variance that cannot be mitigated if the cell is to be kept erased when data is written. In contrast, the variance of the programmed levels can be reduced using ISPP. In our simulations we imitated this by setting $v_0 = 1.1, v_1 = 2.7, v_2 = 3.3$ and $v_3 = 3.9$. For sufficiently low noise power and using the optimal fixed references, the majority of the errors will be caused by programmed cells being detected in a different program level. Besides, when an erased level is detected as programmed,

it will most likely be in the first program level and vice-versa. As a result, the amplitude stream performs similarly regardless of the type of reference used to detect the index stream. Furthermore, since the programmed levels are equally spaced, and their noise variances are the same, the probability of error between any two adjacent cells is the same. The results is that the MSB and LSB perform equally in the UP scheme. To illustrate this, we change v_0 from 1.1 to 1.5 (or equivalently shift down the programmed levels by 0.4) while keeping the remaining parameters unchanged. The results are shown in Fig. 5.6. By increasing v_0, the chance of transition between the erased and first programmed levels becomes comparable to that between the programmed levels. As a consequence, the LSB page of the UP scheme now performs better than its MSB counterpart, since a significant amount of errors in the MSB happens by transitions between the erased and the first programmed levels. Furthermore, the amplitude stream performs slightly better when the dynamic reference is used to detect the programmed cells, even though the gap between the index and amplitude stream is now reduced. We conclude that by using IP with $\zeta = 0.75$, it is possible to replace the first optimal fixed reference, which is costly to update, by a dynamic reference that is relatively simpler. This dynamic reference provides superior detection performance for the index page and possibly for the amplitude stream. Traditionally, the large variance of the erased level required a large distance between its level and that of the first programmed level. The results above suggest that the index and amplitude streams of IP have quite different error characteristics and it may be possible to differently optimize the distances between the voltages to balance the errors between the the streams.

By setting $\zeta = 0.75$, we have preserved the storage efficiency. However, the IP scheme provides a degree of freedom where the storage efficiency can be reduced in order to increase the robustness to errors. To see this, Fig. 5.7 shows the results for

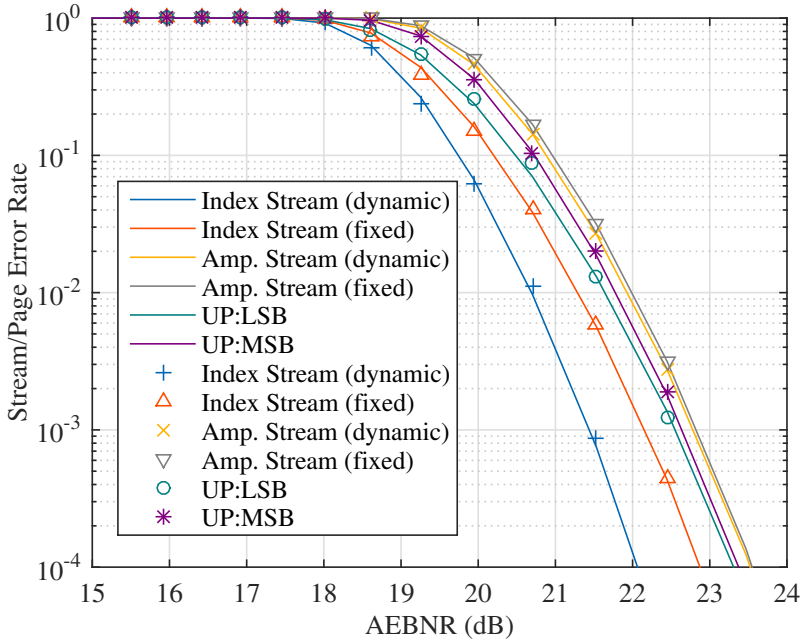


Figure 5.6: The stream/page error rates as functions of AEBNR, where solid lines represent the theoretical plots and markers their simulated counterpart, for $v_0 = 1.5$, $\zeta = 0.75$ and $\eta = 1.99957$ bits/cell.

error performance when $\zeta = 0.65$ and for which $\eta \approx 1.964$ bits/cell. By this small reduction in storage efficiency, both the amplitude and index streams perform significantly better than the MSB and LSB pages when UP is employed. In particular, at an error rate of 10^{-2}, the amplitude page provides a gain of approximately 1.5 dB while the index page provides a gain of 6 dB and 5.4 dB when detected using the dynamic and fixed reference, respectively.

Fig. 5.8 shows the results for $\zeta = 0.5$ in which case $\eta = 1.79$ bits/cell. The trade-off is clear that storage efficiency can be sacrificed so that the resiliency to error is increased. The index stream performs at least 7 dB better than the MSB and LSB pages of UP at 10^{-2} error rate, while the amplitude page performs 3.5 dB better. We conclude that the IP scheme provides a trade-off between storage efficiency and resiliency to errors in addition to an easy way to detect a portion of

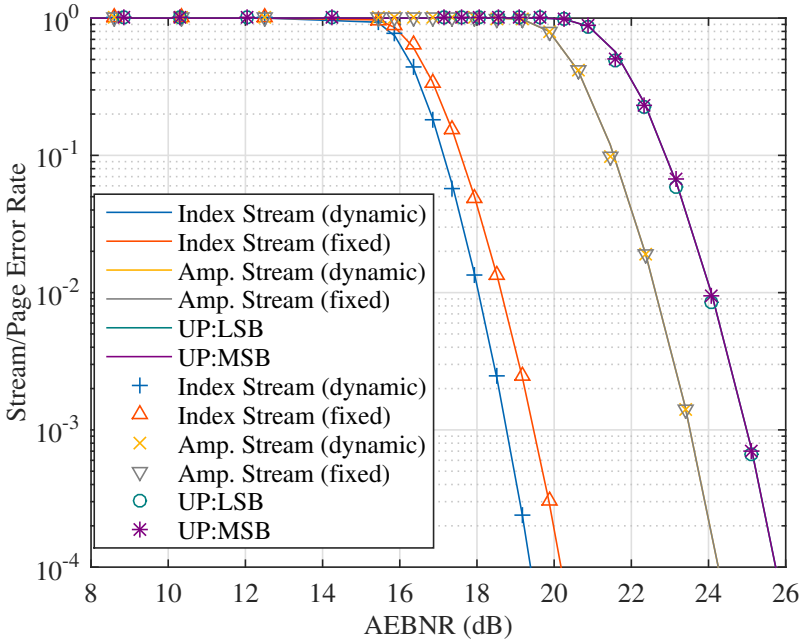


Figure 5.7: The stream/page error rates as functions of AEBNR, where solid lines represent the theoretical plots and markers their simulated counterparts, for $\zeta = 0.65$ and $\eta = 1.964$ bits/cell,. In this case, the UP:LSB and UP:MSB plots are coincident. The Amp. Stream (dynamic) and Amp. Stream (fixed) plots are also coincident.

the data using the dynamic read reference.

Table 5.1 summarizes the difference between IP and UP in the required AEBNR to achieve a 10^{-2} error rate. A positive performance gain indicates better performance of the IP page. To provide a reasonable estimation of the endurance gain provided by the IP scheme, we assume that the noise power is a function of only the number of P/E cycles. According to [21], the noise power scales with the number of P/E cycles in an approximate power-law fashion, i.e., $\sigma^2 = aN^{\xi}$ where a and ξ are fitting constants and N is the number of P/E cycles. In this case, a reduction of g dB in the required average energy-to-noise-ratio when IP is employed translates into an endurance extension of the device by a factor of $10^{\frac{g}{10\xi}}$ in cycles. For example, a reduction of 1 dB corresponds to approximately 1.5 times more P/E cycles before the error probability rises above 10^{-2} when $\xi = 0.6$, which is a typical power law

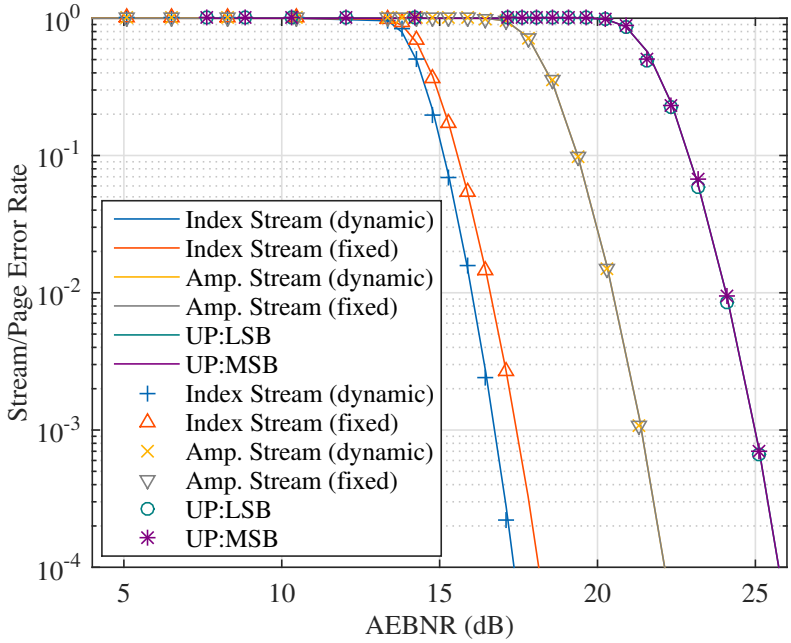


Figure 5.8: The stream/page error rates as functions of AEBNR, where solid lines represent the theoretical plots and markers their simulated counterparts, for $\zeta = 0.5$ and $\eta = 1.79$ bits/cell. In this case, the UP:LSB and UP:MSB plots are coincident. The Amp. Stream (dynamic) and Amp. Stream (fixed) plots are also coincident.

Gain of 'x' over MSB/LSB	$\zeta = 0.5, \eta = 1.79$	$\zeta = 0.675, \eta = 1.963$	$\zeta = 0.75, \eta = 1.9995$
index page (dyn)	8 dB	6 dB	4.25 dB
amplitude page (dyn)	7.5 dB	5.4 dB	3.65 dB
index page (fixed)	3.5 dB	1.4 dB	−0.25 dB
amplitude page (fixed)	3.5 dB	1.4 dB	−0.25 dB

Table 5.1: Performance gain of IP at 10^{-2}

according to [21]. The above results prove the potential of our proposed scheme in making at least one page more robust to errors.

5.6 Binary Amplitude Pages in IP

In conventional flash memory, the number of levels q is typically chosen to be a power of two in order to be able to store an integer number of bits per cell. This is achieved with a binary paging structure where each bit of the cell is on a separate page that is read independently without having to map non-binary symbols from

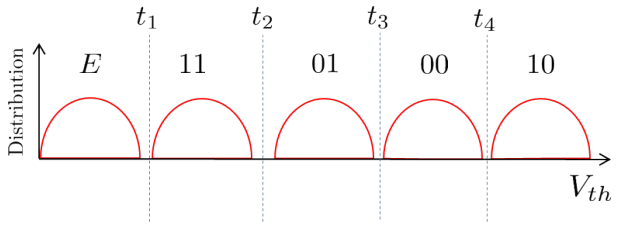


Figure 5.9: Diagram showing the states and read references for $q = 5$.

several cells into a binary sequence. Consequently, increasing the capacity from 3 to 4 bits/cell requires fitting eight additional levels in the voltage range without compromising the reliability, which is challenging with existing technology. It may be desirable to have a structure where a non-power-of-two number of levels exist without increased complexity. The IP scheme creates two logical pages (streams) on each word-line: the index page and the amplitude page. Here, we explain how the amplitude page can be further divided into several logical pages that are written and read in a fashion that is very similar to UP when $q = 2^p + 1$.

In practice, binary pages within a word-line can be read independently, but cannot be programmed independently. As an example, consider an MLC device using Gray labeling (Fig.2.2). Writing '1' in the LSB page is done by keeping a cell in the first voltage window ('11'), whereas writing '0' requires shifting it into the third window ('00'). To write into the MSB page, the LSB page is first read, and the voltages of its cells are adjusted into the corresponding window to reflect a '0' or a '1' in the MSB positions. In the proposed IP scheme, the word-line will always be divided into two pages. In IP, the binary paging structure is maintained by having $q = 2^p + 1$ instead of 2^p levels. The index page is programmed by selecting k cells accordingly and setting them to the first programmed (non-erased) state. This page should always be written before the other pages the same way the LSB page always has to be written before the MSB page in UP. The result is a set of k cells that can be programmed to any of the 2^p programmed states. This means that the k cells

can be dealt with as a word-line with k cells, each storing p bits so that they can be sequentially programmed in the same manner as UP explained above. Hence the total number of pages is $p+1$. Fig. 5.9 illustrates the paging operation of IP when $q = 5$. First, the index page is written by shifting k cells from the erased state (E) to state '11' where the k cells are chosen according to the input bits. To write the first amplitude page (LSB-IP), these k cells are first detected by applying the fixed reference t_1 or the more complex (but more robust) dynamic detector to identify the k programmed cells. Then for each of the k cells, it is either kept in its state or shifted to state '00', depending on the bit to be written on the LSB-IP page in that cell. Finally, to program the second amplitude page (MSB-IP), both the index and the LSB-IP pages need to be read, and then the cells' voltages are adjusted to reflect a '0' or a '1' in the MSB-IP page.

The read process is performed in a very similar way to that of UP. For the example above, the index page is read by detecting the k programmed cells (using fixed or dynamic reference). The information bits can then be retrieved by de-mapping the activation pattern. To read the LSB-IP page, the programmed cells should be detected first, and so that t_3 can be applied to identify their LSB bits. To read the MSB-IP page, the programmed cells are first detected, and then t_2 and t_4 are applied to identify the MSB bits. When a fixed reference is used to detect the programmed cells, the process turns out to be very similar to reading the pages of a TLC device. In a TLC device, a single reference is required to read the first page, two for the second and three for the third page. It is important to note that in this case t_1 is applied when reading any page, as opposed to traditional reading where different pages need a different set of references. This is an advantage when several pages on the same word-line are to be read simultaneously. In general, when $q = 2^p + 1$ it is possible to divide a word-line into $p+1$ logical pages using IP the

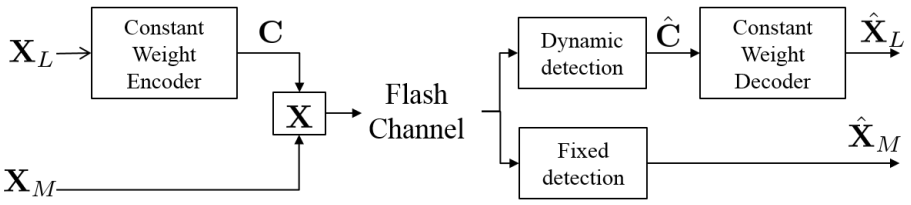


Figure 5.10: Diagram showing the alternative index programming process.

same way as described above.

5.7 Alternative Index Programming

The proposed IP scheme altered the paging structure by logically dividing the WL into an index stream and an amplitude stream, which can be further divided into pages as explained in the previous section. However, it may be desirable to keep the traditional paging structure while exploiting the idea of a dynamic reference. In this section, we propose an alternative IP scheme using $q$ary cells for q a power of two and where the traditional paging is maintained. We focus on MLC ($q = 4$) with Gray labeling, but it is straightforward to generalize to any q that is a power of two with Gray labeling.

Instead of dividing the information into the index and amplitude streams, we keep the UP division of a WL into two independent binary pages, i.e., the LSB and MSB pages. However, in contrast to UP, we fix the weight of the LSB page to k and convey information by their indices. Assuming the Gray labeling shown in Fig. 2.2, the detection of the LSB page will require a single dynamic reference such that k cells are below it. The MSB page is left unrestricted, therefore it will be detected using two optimal fixed references as it is the case with UP. The MSB stores exactly 1 bit/cell while the LSB stores $\frac{1}{n}\left\lfloor \log_2 \binom{n}{k} \right\rfloor \leq 1$ bit/cell which is maximized when the LSB is balanced, i.e. $k = n/2$ and when $n \to \infty$ the maximum storage efficiency of the LSB is $H_2(0.5) = 1$ bit/cell. Hence, a reduction in storage efficiency will

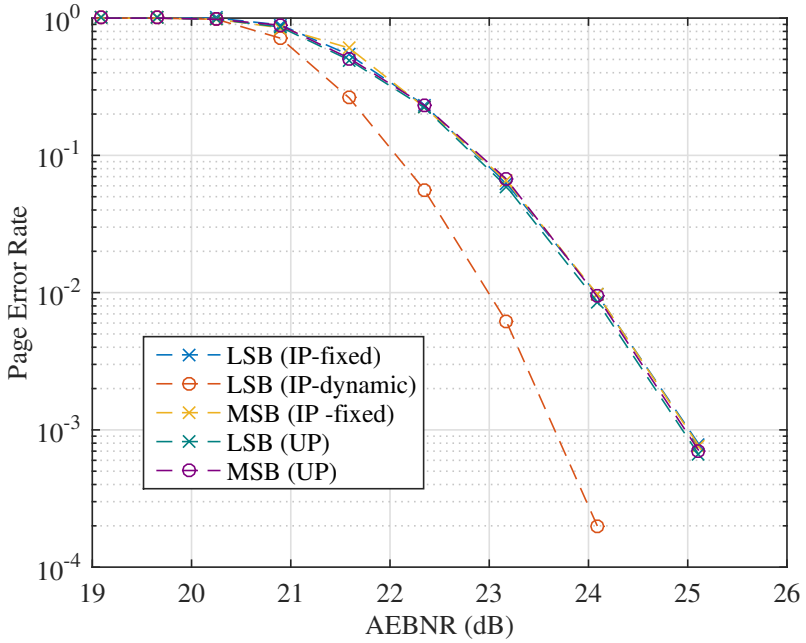


Figure 5.11: The page error rates as functions of AEBNR the for alternative IP with $k = 0.5n$ and $\eta = 1.9995$ bits/cell (dashed for theoretical and markers for simulated).

allow the use of a dynamic reference to detect the LSB and increase its resiliency to errors. Consequently, we have used a single dynamic reference and two fixed references, similar to the original IP, but this time the binary paging structure is preserved. The alternative IP process is shown in Fig. 5.10 where the input bit stream to the LSB and MSB pages are denoted by $\mathbf{X}_L$ and $\mathbf{X}_M$, respectively, where $\mathbf{X}_L$ is of length shorter than n while $\mathbf{X}_M$ is of length n. The constant weight encoder maps $\mathbf{X}_L$ codeword $\mathbf{C}$ of length n which together with the unaltered MSB input form the write pattern $\mathbf{X}$ which is written into the memory. At the reading side, the LSB is read dynamically by detecting the k cells with the highest voltages to estimate $\mathbf{C}$ and then the decoder maps it back to the estimate $\hat{\mathbf{X}}_L$ of $\mathbf{X}_L$, while the fixed references are used to read the MSB as is done in conventional operation.

One potential application of this alternative index programming is the reduction of programming errors. In section 2.5 we have discussed the programming errors

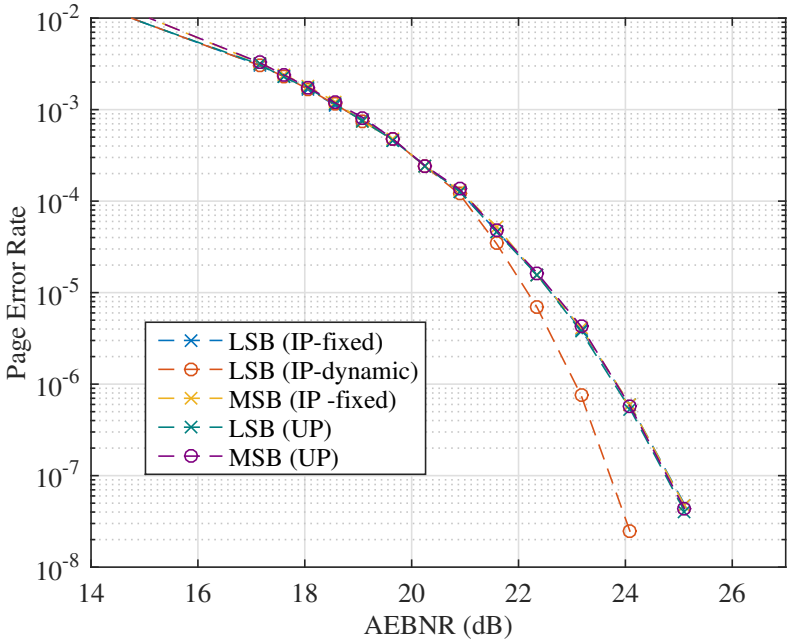


Figure 5.12: The bit error rates as functions of AEBNR for alternative IP with $k = 0.5n$ and $\eta = 1.9995$ bits/cell.

caused by the two-step programming with shadow sequence. For MLC, the problem is illustrated by the bit flips in the LSB page due to ICI in the intermediate time before the MSB page is programmed. Therefore, by reducing the bit error rate in the LSB page, the programming errors are reduced. This can be achieved by using the alternative IP scheme explained above. Fig. 5.11 shows the LSB and MSB page error rates for UP and the alternative IP scheme and for the case where the LSB is balanced , i.e., $k = 0.5n$ in which case the storage efficiency is 1.9995 bits/cell, hence there is little reduction in efficiency. It is clear that if the LSB page is detected using the fixed reference, there is no difference between the alternative IP and the UP schemes. However, detecting the LSB with the dynamic reference results in almost a 1 dB of improvement at an error rate of 10^{-2}.

Fig. 5.12 shows the bit error rate in each of the pages for the same parameters. It is clear that the bit error rate in the LSB is reduced when detected using the

dynamic reference, while the alternative IP is equivalent to UP if fixed references are used.

As a final note, in the previous two chapters, we were interested in the effect of ICI, therefore the noise power in our analysis was fixed and what varied was the strength of ICI. This imitates the trend towards larger coupling ratio due to device scale down. In this chapter, we focused less on ICI and more on degradation, where the noise power increases with device usage due to the adverse effects of electrical stress. These two problems (degradation and ICI) are the major reliability issues facing flash memory.

5.8 Conclusion

In this chapter, we have introduced a new data representation scheme that provides a compromise between using purely fixed references and purely dynamic references for cells with $q > 2$. The IP scheme consists of fixing the number of cells that are programmed in each word-line to k, therefore dividing the information into an index stream and an amplitude stream. The index stream can be detected using a dynamic reference that does not require regular estimation, while the amplitude stream still requires fixed references. We have shown that the stream error rate of the proposed scheme is lower than that of traditional UP in which the WL is divided into binary pages. Furthermore, we have shown that detecting the index stream using the dynamic reference provides a lower error rate than using an optimal fixed reference. This is in contrast with previous results that have shown the opposite but in the asymptotic regime ($n \to \infty$). We have shown that the new scheme provides a trade-off between efficiency and error resiliency. We have analyzed the ICI effect of this scheme and deduced that there is a trade-off between storage efficiency and the degree of ICI reduction. We discussed how $p + 1$ logical pages can be constructed

using IP with $q = 2^p + 1$. Finally, we introduced an alternative IP scheme that preserves the traditional paging and that allows the LSB page to become more resilient to error. This alternative scheme is a potential mitigation technique to the programming errors.

Chapter 6

Conclusion and Future Work

Over the last three decades, flash memory has revolutionized the digital world by going beyond merely providing additional storage capacity for existing application. It ignited a whole new range of applications and inspired storage architectures for personal and enterprise storage. Cost-effectiveness, which was made possible due to cell-size shrinkage and the use of multi-level cells, has been a decisive factor in its popularity.

However, flash memory is reaching its scaling limit, despite the continuous effort to delay it, where the reliability pressure is becoming unbearable. As devices become smaller in size and denser in cells, ICI will keep increasing and hence limiting further device scaling. The decreasing number of electrons that a cell can store, will make them much more susceptible to noise and charge leakage. While 3D flash memory can slow down this trend by reducing the dependency on cell-size shrinkage, the surging demand for capacity and speed will bring it back to the same challenges when vertical stacking of cells reaches its limit. Besides, 3D NAND itself present a whole range of unique reliability issues that are starting to unfold, e.g., severe variability between word-lines, fast initial charge loss and retention interference [26].

It is therefore imperative to gain a deep theoretical understanding of the structure, operation and characteristics of flash memory in order to design reliability methods that extend its lifetime.

This thesis studies novel approaches to understanding the reliability challenges of flash memory and novel methods to address these challenges.

Chapter 3 provides new insights on the ICI phenomenon and proposes a general theoretical model that incorporates this new understanding. First, the noise is approximated by a Gaussian to ensure mathematical tractability, and subsequently, the channel model corresponding to a single cell in the presence of ICI is derived. Then, it is shown that interference has a more complex structure that goes beyond being merely an independent random noise. The intuition behind this is that usually cells are subject to interference by several neighboring cells, which means that adjacent cells have aggressors in common. Accordingly, the joint output distribution of adjacent cells given their inputs is derived. The evaluation of the normalized mutual information shows that as more cells are considered jointly, the normalized mutual information between their outputs and inputs increase. This motivates the investigation of methods through which this additional mutual information can be extracted to improve device performance.

Chapter 4 introduces two novel methods to improve the accuracy of the soft-information that is passed to the soft ECC decoder when ICI cancellation is not desirable. These methods leverage the dependency between cells. In the first method, the cells on a word-line are partitioned into pairs, and the PMFs of each cell, i.e., soft detector output, is computed as a function of the outputs of the pair to which the cell belongs. The soft-information that results from this method, called naive, is then passed to the soft ECC decoder. The second method proposes a probabilistic graphical model that accurately represents the dependency between

adjacent cells. The model turns out to be a hidden Markov model, and consequently, the sum-product algorithm is employed to obtain the soft-input to the decoder. Results show that both methods improve the performance of the decoder compared to when each cell is considered individually, but the second model-based method provides larger improvement.

To bring some practicality into the discussion, the case when the outputs are quantized using a low-precision quantizer is considered. It was shown that while the sum-product algorithm still provides improvements in the quantized case, it suffers the most degradation compared to the unquantized case. This is a natural result of quantization which necessarily discards some information that the sum-product exploits efficiently to improve the performance in the unquantized case. When the system is limited by ICI and not by noise, it is shown that significant improvements re-emerged. To recover some of the losses caused by quantization, an iterative detection/decoding strategy is proposed where multiple iteration of detection and decoding are performed. The results show that such a strategy can improve the performance of the sum-product algorithm.

Chapter 5 addresses the problem of read references becoming sub-optimal for hard decision when the device degrades. To partially overcome this problem, constant weight codes are exploited and a novel technique that leverages dynamic read references which are not fixed in advance is proposed. The results show that dynamic references are superior to the optimal fixed references in the practical case of finite number of cells. This is in contrast to previous work on dynamic references which suggested, by studying the asymptotic case, that dynamic references are sub-optimal. The chapter discusses the different aspects of the new technique and proposes an alternative structure that preserves the conventional page architecture. Simulations show that the alternative structure can help in alleviating programming

errors caused by the two-step programming.

With every theoretical endeavor, a certain level of abstraction is needed, which is the case in the current thesis. While this has the disadvantage of making the results not straightforwardly realizable in products, it makes most of the approaches and techniques proposed in this thesis applicable to different generations of flash memory and even alternative emerging storage technologies.

Future Work

Besides providing insights and solutions, the current thesis motivates several streams of potential future work that is formalized into two categories, the first contains further investigation of the topics of this thesis, while the second consists of other areas that are potentially beneficial to address. Potential ideas belonging to the first category are as follows:

- Due to the lack of experimental data, it was not possible to validate the Gaussian mixture model and the correlation between adjacent cells presented in chapter 3. A natural step is to try and obtain data from real devices, validate the model, and compare it with other models in the literature already mentioned in the thesis.

- In chapters 3 and 4, it was assumed that the model parameters were known, i.e., the parameters of $p_{Y|X,\mathbf{S}'}(y|x,\mathbf{s}')$. However, estimating the parameters as they change due to degradation is a crucial problem that requires a solution. Based on the proposed hidden Markov model, it is possible to use the expectation maximization algorithm to estimate the different parameters of the model.

- In the hidden Markov model presented in Chapter 4, the number of states

depends on the number of interfering cells, which can be very large. This results in a large number of parameters to be estimated which is not practical. A solution may be to generalize the model such that the states are defined differently, to have an abstract meaning and a smaller space. For example, a coarser classification of the interference patterns, e.g., weak, fairly strong, very strong, reduces the state space. It is worth evaluating the performance of the sum-product algorithm when the state space is reduced and see whether the performance deteriorates drastically or not.

- In chapter 4, it was mentioned that the used quantizers may not be optimal for the sum-product algorithm. Therefore, future work may attempt to design adequate quantizer for the sum-product algorithm.

- Also in chapter 4, a natural extension is to have a limited number of reads and investigate the optimal way of using them between the victim and aggressor word-lines. For example, is it better to use all the reads on the victim word-line, increasing its precision, or it is better to spend some reads on the aggressor to estimate and cancel ICI?

- Also in chapter 4, the iterative detection/decoding strategy to recover the loss due to quantization was shown to improve the performance for the chosen values of the maximum numbers of outer and inner iterations. However, this was done by trying different values. Future work may attempt to investigate this scheme in more principled way as mentioned in section 4.8.3, by using EXIT chart to propose how iterative decoding can be done optimally. Furthermore, it is interesting to study whether the iterative strategy is more beneficial in comparison to simply increasing the quantization precision.

- In chapter 5, its was mentioned that implementing dynamic references can introduce read latency. Future work may investigate how they can be implemented efficiently. Recently, some work on this started to appear in the literature [76].

- Finally, it may be interesting to investigate further whether having a single dynamic reference as proposed in IP, can inform us about the other fixed references, i.e., can we predict how the fixed references change with time from knowing the trajectory of the dynamic read reference?

Potential ideas belonging to the second category are as follows:

- As the number of electrons in a cell becomes very small, it would be interesting to study the discrete characteristics of the flash memory channel using probabilistic tools such as the Poisson point process to model the relation between the stored data and the distribution of the electrons in the floating gate.

- As 3D flash memory is beginning to dominate the flash market, there is little information about the specific characteristics of that channel. The most recent characterization showed unique characteristics that did not exist in planar NAND flash [26]. Proposing models and signal processing techniques to cope with these issues should be naturally pursued.

Appendix A

Random Vectors and Mutual Information

An n-dimensional random vector $\mathbf{X}$ with possible outcomes in $\mathcal{X}^n$ has a probability density function (p.d.f) $p(\mathbf{x})$ for $\mathbf{x} \in \mathcal{X}^n$. The mean vector of $\mathbf{X}$ is denoted $\boldsymbol{\mu} = \mathbb{E}[\mathbf{X}]$ while its covariance matrix is denoted by $\Sigma = \mathbb{E}[(\mathbf{X}-\boldsymbol{\mu})(\mathbf{X}-\boldsymbol{\mu})^T]$, where $\mathbb{E}[\cdot]$ is the expectation operator. Consider two random vectors $\mathbf{X}$ and $\mathbf{Y}$ that take values in $\mathcal{X}^n$ and $\mathcal{Y}^m$, respectively. Their joint p.d.f is denoted by $p(\mathbf{x}, \mathbf{y})$ and the conditional p.d.f of $\mathbf{Y}$ given $\mathbf{X}$ is denoted by $p(\mathbf{y}|\mathbf{x})$. If $\mathbf{X}$ and $\mathbf{Y}$ are independent we have that $p(\mathbf{y}|\mathbf{x}) = p(\mathbf{y})$ or equivalently $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x})p(\mathbf{y})$. The mutual information between two random vectors $\mathbf{X}$ and $\mathbf{Y}$ denoted by $I\big(\mathbf{X}; \mathbf{Y}\big)$ measures the dependency between them, or how much information one hold about the other. The mutual information is defined as

$$I\big(\mathbf{X}; \mathbf{Y}\big) := \sum_{\mathbf{y}\in\mathcal{Y}^n} \sum_{\mathbf{x}\in\mathcal{X}^n} p_{\mathbf{X}}(\mathbf{x}) p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) \log_2 \frac{p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})}{\sum_{\mathbf{x}\in\mathcal{X}^n} p_{\mathbf{X}}(\mathbf{x}) p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})}, \tag{A.1}$$

and it is equal to 0 when the two vectors are independent. The sums are replaced

by integrals when the concerned variables are continuous.

The above definitions apply to random variables which are one-dimensional vectors and to discrete vectors in which case the p.d.fs are replaced with probability mass functions (p.m.fs). For a random variable X with mean $\mu_X = \mathbb{E}[X]$, we define the kth standardized central moment as follows

$$\mu_k(X) = \frac{\mathbb{E}\left[(X - \mu_X)^k\right]}{\sigma_X^k}. \tag{A.2}$$

When $k = 3$, this is known as the *skewness* and it measures how symmetric the distribution is around its mean, and it is 0 for symmetric distributions. When $k = 4$, this is known as the *kurtosis* it measures the tails thickness of the distribution. For a Normally distributed random variable, the kurtosis is 3, and it is usually taken as a reference for tails thickness a new measure called *excess kurtosis* is defined as $\mu_4 - 3$. The skewness and excess kurtosis of the distribution of a sum of n independent random variables $X_i, ..., X_n$ are given as follows

$$\mu_3\left(\sum_{i=1}^{n} X_i\right) = \frac{\sum \mu_3\left(X_i\right)}{\left(\sum \sigma_{X_i}^2\right)^{\frac{3}{2}}}, \tag{A.3}$$

$$\mu_4\left(\sum_{i=1}^{n} X_i\right) - 3 = \frac{\sum \sigma_{X_i}^4 \left(\mu_4\left(X_i\right) - 3\right)}{\left(\sum \sigma_{X_i}^2\right)^2} \tag{A.4}$$

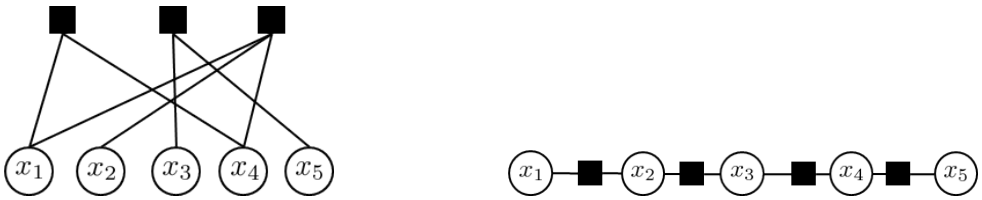
Appendix B

Factor Graphs and the Sum-Product Algorithm

Consider a real function $f(x_1, ..., x_n)$ where the variables $x_1, ..., x_n$ can take values in the sets $\mathcal{X}_1, ..., \mathcal{X}_n$, respectively. We define n marginal functions $f_i(a)$ for $i = 1, ..., n$ by summing all the possible values of $f(x_1, ..., x_n)$ that have $x_i = a$, this operation is also known as summary for x_i of f and is defined as follows

$$f_i(x_i) := \sum_{\sim\{x_i\}} f(x_1, ..., x_n), \tag{B.1}$$

where the summary notation $\sum_{\sim\{x_i\}}$ indicates that the sum is over all possible values of $x_1, ..., x_n$ except x_i. A direct evaluation of the summary can be very hard given the very large number of possible configurations of the arguments. In many practical scenarios, the function can be written as a product of functions each having a smaller number of argument, and we are interested in efficient method of obtaining the summary, by exploiting this structure. Suppose that $f(x_1, ..., x_n)$ can be written as a product of k functions $\{g_j\}_{j=1}^{k}$ having each the element of a subset



(a) A factor graph with cycles. **(b)** A cycle-free factor graph (tree).

Figure B.1: Examples of factor graphs.

$\mathbb{X}_j \subset \{x_1, ..., x_n\}$ as arguments, i.e.,

$$f(x_1, ..., x_n) = \prod_{j=1}^{k} g_j(\mathbb{X}_j). \tag{B.2}$$

The structure of this factorization can be visually represented by a factor graph which is a bipartite graph containing two types of nodes: variable nodes and factor nodes. In this graph, there are n variable nodes and k factor nodes, and there is a link between each factor node j (corresponding to factor g_j) and all its arguments, i.e., the variables in $\mathbb{X}_j$. An example of a factor graph with cycles is shown in Fig. B.1a, where at least two functions have two or more arguments in common, while a cycle-free, also known as tree, example is shown Fig. B.1b.

The well-known *sum-product* algorithm is an efficient way of computing the all the marginals $g_j(x_j)$. It is useful to think of the different nodes as entities that receive messages (functions) from its neighbouring nodes, process them, and send out a newly computed messages to all its neighbouring nodes. These messages are always functions of the variable involved in it (sender or receiver). We denote by $\mu_{x_i \to g_j}(x_i)$ the message sent by variable node x_i to factor node g_j connected to it, and by $\mu_{g_j \to x_i}(x_i)$ the message sent by factor node g_j to variable node x_j connected to it. if we denote by $\mathbb{N}(a)$ the index set of nodes connected to node a (a can be a

factor or variable node), these messages are then computed as follows

$$\mu_{x_i \to g_j}(x_i) = \prod_{k \in \mathbb{N}(x_i) \setminus j} \mu_{g_k \to x_j}(x_j), \tag{B.3}$$

$$\mu_{g_j \to x_i}(x_i) = \sum_{\sim \{x_i\}} g_j(\mathbb{X}_j) \prod_{k \in \mathbb{N}(x_i) \setminus i} \mu_{x_k \to g_j}(x_k). \tag{B.4}$$

One important instance of functions that can be described by factor graphs are joint probability distributions that can naturally be factored using the probability chain rule.

In a cycle-free graph, the algorithm is terminated naturally when each node in the graph has sent and received a message to/from each of its neighbors. Besides, this provides the exact marginals, i.e., the resulting marginals are not an approximation. In contrast, when the graph has cycles, an iterative version of the sum-product is required with a forced stopping criterion, and the marginals may be only approximate. In this thesis, we will only encounter cycle-free graphs, and hence we omit the details of the sum-product in graphs with cycles.

Bibliography

[1] J. Thatcher, T. Coughlin, J. Handy, and N. Ekker, "NAND flash solid state storage for the enterprise: An in-depth look at reliability," *Solid State Storage Initiative (SNIA)*, 2009.

[2] "Data Never Sleeps 6.0." `https://www.domo.com/learn/data-never-sleeps-6`. Accessed: 2018-08-27.

[3] "Nimbus Data Launches the World's Largest Solid State Drive." `https://bit.ly/2OU5quO`. Accessed: 2018-08-27.

[4] S. Aritome, *NAND flash memory technologies*. John Wiley & Sons, 2015.

[5] F. Masuoka, M. Momodomi, Y. Iwata, and R. Shirota, "New ultra high density EPROM and flash EEPROM with NAND structure cell," in *Electron Devices Meeting, 1987 International*, pp. 552–555, IEEE, 1987.

[6] "Flash Memory and SSD Prices (2003-2018)." `https://jcmit.net/flashprice.htm`. Accessed: 2018-08-27.

[7] C. M. Compagnoni, A. Goda, A. S. Spinelli, P. Feeley, A. L. Lacaita, and A. Visconti, "Reviewing the evolution of the NAND flash technology," *Proceedings of the IEEE*, vol. 105, no. 9, pp. 1609–1633, 2017.

[8] R. Katsumata, M. Kito, Y. Fukuzumi, M. Kido, H. Tanaka, Y. Komori, M. Ishiduki, J. Matsunami, T. Fujiwara, Y. Nagata, *et al.*, "Pipe-shaped BiCS flash memory with 16 stacked layers and multi-level-cell operation for ultra high density storage devices," in *VLSI Technology, 2009 Symposium on*, pp. 136–137, IEEE, 2009.

[9] K. Zhao, W. Zhao, H. Sun, T. Zhang, X. Zhang, and N. Zheng, "LDPC-in-SSD: making advanced error correction codes work effectively in solid state drives.," in *FAST*, vol. 13, pp. 244–256, 2013.

[10] J. Brewer and M. Gill, *Nonvolatile memory technologies with emphasis on flash: a comprehensive guide to understanding and using flash memory devices*, vol. 8. John Wiley & Sons, 2011.

[11] J. Javanifard, T. Tanadi, H. Giduturi, K. Loe, R. L. Melcher, S. Khabiri, N. T. Hendrickson, A. D. Proescholdt, D. A. Ward, and M. A. Taylor, "A 45nm self-aligned-contact process 1Gb NOR flash with 5MB/s program speed," in *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, pp. 424–624, IEEE, 2008.

[12] A. Chimenton and P. Olivo, "Reliability of erasing operation in NOR-flash memories," *Microelectronics Reliability*, vol. 45, no. 7-8, pp. 1094–1108, 2005.

[13] K.-D. Suh, B.-H. Suh, Y.-H. Lim, J.-K. Kim, Y.-J. Choi, Y.-N. Koh, S.-S. Lee, S.-C. Kwon, B.-S. Choi, J.-S. Yum, *et al.*, "A 3.3 V 32 Mb NAND flash memory with incremental step pulse programming scheme," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 11, pp. 1149–1156, 1995.

[14] R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti, "Introduction to flash memory," *Proceedings of the IEEE*, vol. 91, no. 4, pp. 489–502, 2003.

[15] C. Calligaro, A. Manstretta, A. Modelli, and G. Torelli, "Technological and design constraints for multilevel flash memories," in *Electronics, Circuits, and Systems, 1996. ICECS'96., Proceedings of the Third IEEE International Conference on*, vol. 2, pp. 1005–1008, IEEE, 1996.

[16] C. M. Compagnoni, M. Ghidotti, A. L. Lacaita, A. S. Spinelli, and A. Visconti, "Random telegraph noise effect on the programmed threshold-voltage distribution of flash memories," *IEEE electron device letters*, vol. 30, no. 9, pp. 984–986, 2009.

[17] C. M. Compagnoni, R. Gusmeroli, A. S. Spinelli, A. L. Lacaita, M. Bonanomi, and A. Visconti, "Statistical model for random telegraph noise in flash memories," *Electron Devices, IEEE Transactions on*, vol. 55, no. 1, pp. 388–395, 2008.

[18] J.-D. Lee, J.-H. Choi, D. Park, and K. Kim, "Degradation of tunnel oxide by FN current stress and its effects on data retention characteristics of 90 nm NAND flash memory cells," in *Reliability Physics Symposium Proceedings, 2003. 41st Annual. 2003 IEEE International*, pp. 497–501, IEEE, 2003.

[19] N. Mielke, H. P. Belgal, A. Fazio, Q. Meng, and N. Righos, "Recovery effects in the distributed cycling of flash memories," in *Reliability Physics Symposium Proceedings, 2006. 44th Annual., IEEE International*, pp. 29–35, IEEE, 2006.

[20] C. Miccoli, C. M. Compagnoni, S. Beltrami, A. S. Spinelli, and A. Visconti, "Threshold-voltage instability due to damage recovery in nanoscale NAND flash memories," *Electron Devices, IEEE Transactions on*, vol. 58, no. 8, pp. 2406–2414, 2011.

[21] G. Dong, Y. Pan, N. Xie, C. Varanasi, and T. Zhang, "Estimating information-theoretical NAND flash memory storage capacity and its implication to memory system design space exploration," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 20, no. 9, pp. 1705–1714, 2012.

[22] Y. Park, J. Lee, S. S. Cho, G. Jin, and E. Jung, "Scaling and reliability of NAND flash devices," in *Reliability Physics Symposium, 2014 IEEE International*, pp. 2E–1, IEEE, 2014.

[23] Y. Cai, S. Ghose, Y. Luo, K. Mai, O. Mutlu, and E. F. Haratsch, "Vulnerabilities in MLC NAND flash memory programming: experimental analysis, exploits, and mitigation techniques," in *High Performance Computer Architecture (HPCA), 2017 IEEE International Symposium on*, pp. 49–60, IEEE, 2017.

[24] J. Gray and C. Van Ingen, "Empirical measurements of disk failure rates and error rates," *arXiv preprint cs/0701166*, 2007.

[25] C. Miccoli, G. M. Paolucci, C. M. Compagnoni, A. S. Spinelli, and A. Goda, "Cycling pattern and read/bake conditions dependence of random telegraph noise in decananometer NAND flash arrays," in *Reliability Physics Symposium (IRPS), 2015 IEEE International*, pp. MY–9, IEEE, 2015.

[26] Y. Luo, S. Ghose, Y. Cai, E. F. Haratsch, and O. Mutlu, "Improving 3D NAND flash memory lifetime by tolerating early retention loss and process variation," *arXiv preprint arXiv:1807.05140*, 2018.

[27] Y. Cai, G. Yalcin, O. Mutlu, E. F. Haratsch, A. Cristal, O. S. Unsal, and K. Mai, "Flash correct-and-refresh: Retention-aware error management for in-

creased flash memory lifetime," in *Computer Design (ICCD), 2012 IEEE 30th International Conference on*, pp. 94–101, IEEE, 2012.

[28] G. Dong, S. Li, and T. Zhang, "Using data postcompensation and predistortion to tolerate cell-to-cell interference in MLC NAND flash memory," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 57, no. 10, pp. 2718–2728, 2010.

[29] D.-h. Lee and W. Sung, "Least squares based coupling cancelation for MLC NAND flash memory with a small number of voltage sensing operations," *Journal of Signal Processing Systems*, vol. 71, no. 3, pp. 189–200, 2013.

[30] Y. Cai, S. Ghose, E. F. Haratsch, Y. Luo, and O. Mutlu, "Reliability issues in flash-memory-based solid-state drives: Experimental analysis, mitigation, recovery," in *Inside Solid State Drives (SSDs)*, pp. 233–341, Springer, 2018.

[31] G. Dong, N. Xie, and T. Zhang, "On the use of soft-decision error-correction codes in NAND flash memory," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 58, no. 2, pp. 429–439, 2011.

[32] D.-h. Lee and W. Sung, "Estimation of NAND flash memory threshold voltage distribution for optimum soft-decision error correction," *IEEE Transactions on Signal Processing*, vol. 61, no. 2, pp. 440–449, 2013.

[33] H. Wang, T.-Y. Chen, and R. D. Wesel, "Histogram-based flash channel estimation," in *2015 IEEE International Conference on Communications (ICC)*, pp. 283–288, IEEE, 2015.

[34] D.-h. Lee and W. Sung, "Soft-decision decoding with cell to cell interference removed signal in NAND flash memory," in *Acoustics, Speech and Signal Pro-*

cessing (ICASSP), 2013 IEEE International Conference on, pp. 2518–2522, IEEE, 2013.

[35] Y. Luo, S. Ghose, Y. Cai, E. F. Haratsch, and O. Mutlu, "Enabling accurate and practical online flash channel modeling for modern MLC NAND flash memory," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 9, pp. 2294–2311, 2016.

[36] G. Dong, N. Xie, and T. Zhang, "Enabling NAND flash memory use soft-decision error correction codes at minimal read latency overhead," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 9, pp. 2412–2421, 2013.

[37] J. Wang, T. Courtade, H. Shankar, and R. D. Wesel, "Soft information for LDPC decoding in flash: Mutual-information optimized quantization," in *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pp. 1–6, IEEE, 2011.

[38] B. Peleato, R. Agarwal, J. M. Cioffi, M. Qin, and P. H. Siegel, "Adaptive read thresholds for NAND flash," *Communications, IEEE Transactions on*, vol. 63, no. 9, pp. 3069–3081, 2015.

[39] X. Yang, "Reference voltage optimization for flash memory," Apr. 17 2012. US Patent 8,159,881.

[40] R. Motwani, "Estimation of flash memory level distributions using interpolation techniques for optimizing the read reference," in *2015 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, IEEE, 2015.

[41] R. Gabrys, E. Yaakobi, L. Dolecek, P. H. Siegel, A. Vardy, and J. K. Wolf, "Non-binary WOM-codes for multilevel flash memories.," in *ITW*, pp. 40–44, 2011.

[42] A. Jiang, R. Mateescu, M. Schwartz, and J. Bruck, "Rank modulation for flash memories," *Information Theory, IEEE Transactions on*, vol. 55, no. 6, pp. 2659–2673, 2009.

[43] A. Berman and Y. Birk, "Constrained flash memory programming," in *Information Theory Proceedings (ISIT), 2011 IEEE International Symposium on*, pp. 2128–2132, IEEE, 2011.

[44] S. Kayser and P. H. Siegel, "Constructions for constant-weight ICI-free codes," in *Information Theory (ISIT), 2014 IEEE International Symposium on*, pp. 1431–1435, IEEE, 2014.

[45] S. Buzaglo, P. H. Siegel, and E. Yaakobi, "Coding schemes for inter-cell interference in flash memory," in *2015 IEEE International Symposium on Information Theory (ISIT)*, pp. 1736–1740, IEEE, 2015.

[46] H. Zhou, A. Jiang, and J. Bruck, "Error-correcting schemes with dynamic thresholds in nonvolatile memories," in *Information Theory Proceedings (ISIT), 2011 IEEE International Symposium on*, pp. 2143–2147, IEEE, 2011.

[47] F. Sala, R. Gabrys, and L. Dolecek, "Dynamic threshold schemes for multi-level non-volatile memories," *Communications, IEEE Transactions on*, vol. 61, no. 7, pp. 2624–2634, 2013.

[48] L. M. Grupp, J. D. Davis, and S. Swanson, "The bleak future of NAND flash memory," in *Proceedings of the 10th USENIX conference on File and Storage Technologies*, pp. 2–2, USENIX Association, 2012.

[49] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis, and modeling," in *Proceedings of the Conference on Design, Automation and Test in Europe*, pp. 1285–1290, EDA Consortium, 2013.

[50] T. Parnell, N. Papandreou, T. Mittelholzer, and H. Pozidis, "Modelling of the threshold voltage distributions of sub-20nm NAND flash memory," in *Global Communications Conference (GLOBECOM), 2014 IEEE*, pp. 2351–2356, IEEE, 2014.

[51] V. Taranalli, H. Uchikawa, and P. H. Siegel, "Channel models for multi-level cell flash memories based on empirical error analysis," *IEEE Transactions on Communications*, vol. 64, no. 8, pp. 3169–3181, 2016.

[52] M. Asadi, X. Huang, A. Kavcic, and N. P. Santhanam, "Optimal detector for multilevel NAND flash memory channels with intercell interference," *Selected Areas in Communications, IEEE Journal on*, vol. 32, no. 5, pp. 825–835, 2014.

[53] H. Yassine, J. Coon, M. Ismail, and H. Fletcher, "Towards an analytical model of NAND flash memory and the impact on channel decoding," in *2016 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2016.

[54] G. Campardo, F. Tiziani, and M. Iaculo, *Memory mass storage*. Springer Science & Business Media, 2011.

[55] Y. Cai, O. Mutlu, E. F. Haratsch, and K. Mai, "Program interference in MLC NAND flash memory: Characterization, modeling, and mitigation," in *Computer Design (ICCD), 2013 IEEE 31st International Conference on*, pp. 123–130, IEEE, 2013.

[56] F. J. Gravetter and L. B. Wallnau, *Statistics for the behavioral sciences*. Cengage Learning, 2016.

[57] C. M. Bishop, "Pattern recognition and machine learning," 2006.

[58] D. M. Titterington, A. F. Smith, and U. E. Makov, *Statistical analysis of finite mixture distributions*. Wiley,, 1985.

[59] W. Ryan and S. Lin, *Channel codes: classical and modern*. Cambridge University Press, 2009.

[60] J. Hu, H.-A. Loeliger, J. Dauwels, and F. Kschischang, "A general computation rule for lossy summaries/messages with examples from equalization," *arXiv preprint cs/0607027*, 2006.

[61] B. M. Kurkoski and H. Yagi, "Quantization of binary-input discrete memoryless channels," *IEEE Transactions on Information Theory*, vol. 60, no. 8, pp. 4544–4552, 2014.

[62] J. Hagenauer, "The exit chart-introduction to extrinsic information transfer in iterative processing," in *Signal Processing Conference, 2004 12th European*, pp. 1541–1548, IEEE, 2004.

[63] W. Chu, C. J. Colbourn, and P. Dukes, "On constant composition codes," *Discrete Applied Mathematics*, vol. 154, no. 6, pp. 912–929, 2006.

[64] D. Tsonev, S. Sinanovic, and H. Haas, "Enhanced subcarrier index modulation (SIM) OFDM," in *2011 IEEE GLOBECOM Workshops (GC Wkshps)*, pp. 728–732, IEEE, 2011.

[65] E. Basar, U. Aygolu, E. Panayirci, and H. V. Poor, "Orthogonal frequency division multiplexing with index modulation," *Signal Processing, IEEE Transactions on*, vol. 61, no. 22, pp. 5536–5549, 2013.

[66] A. I. Siddiq, "Low complexity OFDM-IM detector by encoding all possible subcarrier activation patterns," *IEEE Communications Letters*, vol. 20, no. 3, pp. 446–449, 2016.

[67] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Error patterns in MLC NAND flash memory: Measurement, characterization, and analysis," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012*, pp. 521–526, IEEE, 2012.

[68] T. V. Ramabadran, "A coding scheme for m-out-of-n codes," *Communications, IEEE Transactions on*, vol. 38, no. 8, pp. 1156–1163, 1990.

[69] A. E. Brouwer, L. B. Shearer, N. Sloane, *et al.*, "A new table of constant weight codes," in *IEEE Trans Inform Theory*, Citeseer, 1990.

[70] V. Skachek and K. A. S. Immink, "Constant weight codes: An approach based on Knuth's balancing method," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 909–918, 2014.

[71] D. E. Knuth, "Efficient balanced codes," *Information Theory, IEEE Transactions on*, vol. 32, no. 1, pp. 51–53, 1986.

[72] J. H. Weber and K. A. S. Immink, "Knuth's balanced codes revisited," *IEEE Transactions on Information Theory*, vol. 56, no. 4, pp. 1673–1679, 2010.

[73] K. A. S. Immink and J. H. Weber, "Very efficient balanced codes," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 2, pp. 188–192, 2010.

[74] E. F. Beckenbach, G. Pólya, D. H. Lehmer, and M. Phister, "Applied combinatorial mathematics," 1964.

[75] H. A. David and H. N. Nagaraja, *Order statistics*. Wiley Online Library, 1981.

[76] M. Sforzin, G. Mirichigni, A. Orlando, and P. Amato, "Self-referenced read methodology for EMs," in *2018 IEEE International Memory Workshop (IMW)*, pp. 1–4, IEEE, 2018.