Reliable and energy-efficient 3D NAND flash storage system design using run-time device and system interaction

Md Raquibuzzaman

Follow this and additional works at: https://louis.uah.edu/uah-dissertations

Recommended Citation
Raquibuzzaman, Md, "Reliable and energy-efficient 3D NAND flash storage system design using run-time device and system interaction" (2023). Dissertations. 349.
https://louis.uah.edu/uah-dissertations/349

This Dissertation is brought to you for free and open access by the UAH Electronic Theses and Dissertations at LOUIS. It has been accepted for inclusion in Dissertations by an authorized administrator of LOUIS.
RELIABLE AND ENERGY-EFFICIENT 3D NAND FLASH STORAGE SYSTEM DESIGN USING RUN-TIME DEVICE AND SYSTEM INTERACTION

Md Raquibuzzaman

A DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in The Department of Electrical and Computer Engineering to The Graduate School of The University of Alabama in Huntsville

August 2023

Approved by:
Dr. Biswajit Ray, Research Advisor/Committee Chair
Dr. Aleksandar Milenkovic, Committee Member
Dr. Timothy Boykin, Committee Member
Dr. Ravi Gorur, Committee Member
Dr. Aubrey N. Beal, Committee Member
Dr. Aleksandar Milenkovic, Department Chair
Dr. Shankar Mahalingam, College Dean
Dr. Jon Hakkila, Graduate Dean
Abstract

RELIABLE AND ENERGY-EFFICIENT 3D NAND FLASH STORAGE SYSTEM DESIGN USING RUN-TIME DEVICE AND SYSTEM INTERACTION

Md Raquibuzzaman

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Electrical and Computer Engineering
The University of Alabama in Huntsville
August 2023

NAND Flash memory is a non-volatile solid-state data storage technology widely used in electronic devices such as smartphones, tablets, laptops, digital cameras, USB drives, solid-state drives (SSDs), autonomous vehicles, space applications, and data centers. NAND Flash memory-based SSDs are increasingly replacing the hard-disk drives (HDD) due to their high-density memory storage, low latency, low power consumption, and light weight. The NAND Flash memory has evolved from its planer 2D architecture to its contemporary 3D vertical layer architecture to meet the exponential storage demand. However, the 3D NAND architecture poses new reliability challenges, such as big block management, poly-Si-related read instability, low cell current, and layer-to-layer variability. In addition, the ever-increasing Flash memory density increases energy consumption and degrades the performance of Flash SSDs. Hence, innovation at the system level is necessary to improve the reliability, energy efficiency, performance, and security of future 3D NAND-based storage systems. The dissertation explores factors affecting 3D NAND Flash memory performance and reliability due to vertical stacking and scaling. First, endurance variability is found
among pages due to increasing vertical layers. This causes the under-utilization of the memory blocks leading to lower endurance. An intra-block wear-leveling algorithm based on dynamic, layer-aware logical-down-scaling of Flash memory blocks is proposed to improve overall memory utilization. Second, we identified a disproportionate energy-accuracy trade-off during memory write operation. To address this, the dissertation proposes a technique called EXPRESS, which increases the energy efficiency of Flash memory by up to 50% relative to traditional Flash writes while maintaining data integrity. Third, deleting data instantly from NAND Flash memories incurs hefty overheads and increases wear levels with the risk of data leakage. This work introduces an instant data sanitization method for multi-level-cell Flash memory. The dissertation also proposes a new page-writing technique for hiding secret information using the threshold voltage variation of programmed memory cells. Overall, this dissertation significantly contributes to NAND Flash memory reliability and energy efficiency and proposes several novel techniques and algorithms for improving the performance and endurance of 3D NAND Flash memory-based storage systems.
Acknowledgments

Completing this Ph.D. has been a journey that I could not have embarked upon, let alone completed, without the support, guidance, and encouragement of many people.

First and foremost, I would like to express my profound gratitude to my supervisor, Dr. Biswajit Ray, for his unwavering guidance and support throughout this journey. His knowledge, experience, and patience have been invaluable to me. He has been both a supervisor, mentor, and source of inspiration.

I am also deeply grateful to my committee members, Dr. Aleksandar Milenkovic, Dr. Ravi Gorur, Dr. Timothy Boykin, and Dr. Aubrey N.Beal. for their insightful feedback, constructive criticisms, and helpful suggestions. Their expertise and guidance have significantly enriched this work.

I owe tremendous gratitude to my colleagues at my research lab named: “Hardware Security and Reliability Lab”, who have enriched my graduate studies in numerous ways. Their company and collaboration have made this journey enjoyable and fulfilling.

I am fortunate to have a supportive family who has been there for me throughout this journey. To my parents, Md Shahiduzzaman and Rozina Akter, and my sister Rakaiya Zaman, thank you for your unconditional love and unwavering belief in me. To my spouse, Zareen Tasnim, your encouragement and steadfast faith in me have been my source of strength. To my beloved two-year-old daughter, Zyva Zaman, who has been my constant source of joy and motivation throughout this journey. Your innocent laughter and boundless curiosity have been the brightest reminders of why perseverance and dedication matter.
Finally, I would like to acknowledge the financial support from National Science Foundation (NSF, grant CNS-2007403), the “Mary Makima and Lester Ross” scholarship, and Alabama EPSCoR (Graduate Research Scholars Program (GRSP)). Their support has been instrumental in the completion of this research.

In conclusion, while this dissertation is an academic requirement, it would not have been possible without the academic, personal, and financial support I have received. Please accept my heartfelt thanks to everyone who contributed to this journey.
Dedication

I dedicate my Ph.D. thesis to my mother, Rozina Akter.
Table of Contents

Abstract ........................................................................ ii

Acknowledgments .......................................................... v

Dedication ........................................................................ vii

Table of Contents ........................................................... xii

List of Figures ................................................................. xiii

List of Tables ................................................................. xvii

List of Symbols ............................................................... xviii

Epigraph ........................................................................... xx

Chapter 1. Introduction ...................................................... 1

Chapter 2. Background ...................................................... 7

2.1 History of NAND Flash Memory Development ............... 7

2.2 Fundamentals of NAND Flash Memory ....................... 9

2.3 NAND Flash Memory Operations in Cell .................... 12

viii
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.4  NAND Flash Memory Array Structure</td>
<td>15</td>
</tr>
<tr>
<td>2.5  NAND Flash Memory Operations in Array</td>
<td>17</td>
</tr>
<tr>
<td>2.6  3D NAND Flash Memory</td>
<td>26</td>
</tr>
<tr>
<td>2.7  Fabrication of NAND Flash Memory</td>
<td>30</td>
</tr>
<tr>
<td>2.8  Flash-based Storage System</td>
<td>31</td>
</tr>
<tr>
<td>2.9  Experimental Setups</td>
<td>33</td>
</tr>
<tr>
<td>2.10 Reliability Issues in 3D NAND Flash Memory</td>
<td>34</td>
</tr>
<tr>
<td>2.10.1 Program Disturb</td>
<td>34</td>
</tr>
<tr>
<td>2.10.2 Read Disturb</td>
<td>37</td>
</tr>
<tr>
<td>2.10.3 Endurance</td>
<td>38</td>
</tr>
<tr>
<td>2.10.4 Data Retention</td>
<td>39</td>
</tr>
<tr>
<td>2.10.5 Program Errorsways</td>
<td>41</td>
</tr>
<tr>
<td>2.10.6 Temperature Dependency and Cross-temperature Effect</td>
<td>43</td>
</tr>
<tr>
<td>2.10.7 Open Block Issue</td>
<td>43</td>
</tr>
<tr>
<td>2.10.8 Temporal Read Errors</td>
<td>44</td>
</tr>
<tr>
<td>Chapter 3. Layer-to-Layer Endurance Variation of 3D NAND Flash Memory</td>
<td>45</td>
</tr>
<tr>
<td>3.1  Introduction</td>
<td>45</td>
</tr>
<tr>
<td>3.2  Experimental Methodology</td>
<td>47</td>
</tr>
</tbody>
</table>
Chapter 5. Instant Data Sanitization on Multi-Level-Cell NAND Flash Memory

5.1 Introduction .................................................. 91
5.2 Background of Data Sanitization in NAND Flash ................. 94
5.3 Proposed Sanitization Method .................................. 97
  5.3.1 Sanitization of LSB Page Only ............................. 98
  5.3.2 Sanitization of MSB Page Only ............................ 100
  5.3.3 Concurrent LSB and MSB Page Sanitization ............... 101
5.4 Experimental Evaluation ....................................... 102
  5.4.1 Evaluation of LSB Page Only Sanitization ................. 105
  5.4.2 Evaluation of MSB Page Only Sanitization ................. 107
  5.4.3 Evaluation of Concurrent LSB and MSB Page Sanitization 109
  5.4.4 Effects of Page Sanitization on Retention of Valid Data 109
  5.4.5 Effects of Page Sanitization on Valid Data in Neighbor Layers 111
  5.4.6 Evaluation on Multiple Chips .............................. 113
  5.4.7 Limitations .................................................. 114
  5.4.8 Future work .................................................. 114
5.5 Conclusions .................................................... 116
## List of Figures

1.1 Trends of bit density of planar and vertically stacked 3D NAND Flash memory technology. ........................................ 3

2.1 (a) A 2D NAND Flash memory cell. (b) Current sensor used to determine the charge stored in FG. ............................... 10

2.2 (a) NAND cell erase operation. (b) Energy band diagram after erase operation. ....................................................... 12

2.3 (a) NAND cell program operation. (b) Energy band diagram after program operation. ............................................... 14

2.4 NAND Flash cell during a read operation (a) ON cell. (b) OFF cell. ................................................................. 15

2.5 NAND Flash memory array structure. ........................................... 16

2.6 NAND Flash memory block during erase operation. ............. 17

2.7 NAND Flash memory block during program operation. ........ 19

2.8 ISPP scheme of NAND Flash memory. ................................. 20

2.9 NAND Flash memory block during a page read operation. .... 23

2.10 NAND Flash memory state coding for SLC, MLC and TLC memory. ................................................................. 25

2.11 (a) 3D NAND Flash memory cell. (b) 3D NAND Flash memory vertical array structure. ............................................. 27

2.12 Circuit diagram of a 3D NAND Flash memory block. .......... 29

2.13 (a) Film stack deposition. (b) High aspect ratio etching (RIE). (c) Vertical cross-section of the 3D NAND gate stack. ........ 31

2.14 (a) Simplified view of a storage system containing a memory chip and controller. (b) Main functions of FTL. ................. 32

2.15 Experimental set-up for interfacing raw COTS NAND memory chips. .............................................................. 33

2.16 (a) Various disturb mechanism in NAND array. (b) Program disturb in 3D NAND (collected from [1]). .......................... 35
4.8 (a) Cumulative distribution of program time for LSB (blue) and MSB (red) pages in a block in fresh condition. (b) Express algorithm for MSB pages. .................................................. 82

4.9 Cumulative distribution of page program time for different PE cycle conditions. ........................................................... 85

4.10 BER as a function of retention loss for traditional programming (blue) and proposed partial programming (red) for (a) SLC page. (b) LSB page. (c) MSB pages. .......................................................... 87

4.11 Percentage of programmed bits as a function of partial page program time for the image data on (a) SLC and (b) LSB (blue) and MSB (red) pages. .......................................................... 89

5.1 Proposed instant data sanitization techniques. ......................... 98

5.2 Extracting the threshold voltage distribution of various states of NAND Flash memory. .................................................... 103

5.3 (a) MSB and LSB page before sanitization. (b) MSB and LSB page after LSB sanitization. .................................................... 106

5.4 (a) MSB and LSB page before sanitization. (b) MSB page and LSB page after MSB sanitization. .................................................... 108

5.5 (a) MSB and LSB page before sanitization. (b) MSB and LSB page after concurrent MSB and LSB sanitization. ....................... 110

5.6 Effects of page sanitization on the retention errors of valid data of the shared page. .......................................................... 111

5.7 Effects of proposed page sanitization on the valid data on neighboring memory layers. .......................................................... 112

6.1 System overview of the proposed solution. ......................... 120

6.2 Illustration of data hiding method. ................................. 121

6.3 The evolution of the public image and corresponding $V_t$ distribution of program bits after steps 1, 2 and 3 are shown in blue, black, and red color, respectively. .................................................... 125

6.4 Experimental evaluation of hidden data recovery method. ....... 126
6.5 Trade-off between the accuracy of secret data and public data explored by varying partial program time in step 3 of the hiding method. .......................................................... 129

6.6 Comparison of $V_t$ distribution of zero bits with secret encoding (proposed write) and without secret encoding (traditional write in gray color). .................................................. 130
List of Tables

4.1 Comparison of EXPRESS method with existing works. . . . . . . 65
4.2 EXPRESS characterization for NAND Flash block in fresh condition. 84
4.3 EXPRESS characterization for NAND Flash block after PE cycles. 86

5.1 Multiple chip sanitization results. . . . . . . . . . . . . . . . 114
## List of Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_t$</td>
<td>Transistor’s threshold voltage</td>
</tr>
<tr>
<td>$V_{ref}$</td>
<td>Read reference voltage</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Standard deviation</td>
</tr>
<tr>
<td>2D NAND</td>
<td>Planar two dimensional NAND Flash memory</td>
</tr>
<tr>
<td>3D NAND</td>
<td>Vertically stacked three dimensional NAND Flash memory</td>
</tr>
<tr>
<td>BL</td>
<td>Bit line</td>
</tr>
<tr>
<td>CT</td>
<td>Charge trap</td>
</tr>
<tr>
<td>EPROM</td>
<td>Electrically programmable read-Only memory</td>
</tr>
<tr>
<td>FG</td>
<td>Floating gate</td>
</tr>
<tr>
<td>FN</td>
<td>Fowler-Nordheim tunneling</td>
</tr>
<tr>
<td>FTL</td>
<td>Flash translation layer</td>
</tr>
<tr>
<td>GSL</td>
<td>Ground selection line</td>
</tr>
<tr>
<td>ISPP</td>
<td>Incremental step pulse program scheme</td>
</tr>
<tr>
<td>Symbol</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>MLC</td>
<td>Multiple-level cell (2 bits per cell)</td>
</tr>
<tr>
<td>MOSFET</td>
<td>Metal-oxide-semiconductor field-effect transistor</td>
</tr>
<tr>
<td>MP3</td>
<td>MPEG audio layer 3</td>
</tr>
<tr>
<td>PDA</td>
<td>Personal digital assistant</td>
</tr>
<tr>
<td>PLC</td>
<td>Penta-level cell (5 bits per cell)</td>
</tr>
<tr>
<td>QLC</td>
<td>Quad-level cell (4 bits per cell)</td>
</tr>
<tr>
<td>SL</td>
<td>Source line</td>
</tr>
<tr>
<td>SLC</td>
<td>Single-level cell (1 bit per cell)</td>
</tr>
<tr>
<td>SSL</td>
<td>String selection line</td>
</tr>
<tr>
<td>TLC</td>
<td>Triple-level cell (3 bits per cell)</td>
</tr>
<tr>
<td>WL</td>
<td>Word line</td>
</tr>
</tbody>
</table>
Your vast knowledge is a mere raindrop from a different perspective.

- Raquib
Chapter 1. Introduction

We are witnessing exponential demand for robust, energy-efficient, and high-performance non-volatile memory technologies. This burgeoning demand is driven by various applications, from the ever-evolving field of consumer electronics that seeks seamless data accessibility to the vast expanse of cloud computing requiring immense data storage and fast processing. In the domain of non-volatile storage technologies, including Read-Only Memory (ROM), Electrically Erasable Programmable Read-Only Memory (EEPROM), NOR Flash memory, along with burgeoning technologies such as Resistive Random Access Memory (ReRAM), and Magnetoresistive Random Access Memory (MRAM), NAND Flash memory distinguishes itself considerably. Its ability to offer superior data density and fast read, write, and erase operation, coupled with the power-off data retention capability, makes it an efficient and cost-effective choice. These attributes have facilitated its deployment in various applications, from personal electronic devices to complex enterprise-grade storage systems. In the chronicle of NAND Flash memory technology, the advent of 3D NAND Flash memory marks a significant milestone. This innovation is rapidly gaining attention, offering a higher storage density, improved reliability, and lower power consumption than its predecessor, the traditional planar 2D NAND Flash memory. As we march into an increasingly
digital future, the 3D NAND Flash memory is poised to play a pivotal role in defining the next era of data storage. Rapid 3D NAND Flash memory advancement has made notable progress in device architectures, manufacturing processes, and system-level integration techniques. A shift to 3D NAND technology opened up a new “scaling” dimension - the number of vertical layers – enabling further increases in the bit density for a given die area [5, 6]. Continual advances in this technology resulted in several generations of 3D Flash memory chips, each having a larger number of stacked layers, from early 32- to contemporary 178-layer designs, as shown in Figure 1.1. These advances promise to extend an astonishing growth of bit-density over the next decade [7, 8, 9]. However, these trends increase the number of pages in a memory block leading to large-sized memory blocks. Managing such blocks in the Flash translation layer (FTL) presents a new set of challenges for data mapping and wear-leveling of the storage media [10, 11, 12]. To fully realize the potential benefits of this technology, it is crucial to address several critical challenges.

The reliability of 3D NAND Flash memory is a critical aspect directly affecting its practical usability. Due to the stacking of memory cells in multiple layers, 3D NAND Flash introduces new failure modes and reliability concerns. Understanding the underlying mechanisms of these failure modes and developing mitigation strategies is paramount. This thesis delves into the reliability issues specific to 3D NAND Flash memory. Energy efficiency is another crucial aspect that plays a vital role in designing and operating 3D NAND Flash memory. As data centers and mobile devices strive for lower power consumption, developing
energy-efficient memory architectures and system-level optimizations becomes imperative. Additionally, this research strongly emphasizes the security and privacy aspects of 3D NAND Flash memory. As the adoption of 3D NAND technology increases, ensuring the confidentiality and integrity of data stored in Flash memory becomes critical.

- **Chapter 2** serves as a comprehensive exploration of the literature, background, and other illustrative aspects pertaining to the emerging reliability issues in 3D NAND Flash memory. Building upon the introductory discussion on the importance of reliability in practical usability, this chapter delves into an extensive review of relevant research studies, industry developments, and technological advancements. Chapter 2 provides a solid
foundation for understanding the complex reliability challenges specific to 3D NAND Flash memory by synthesizing existing knowledge and highlighting the key findings and insights. Moreover, this chapter presents various illustrations, including graphical representations, to enhance the reader’s comprehension and provide a visual context for the discussed reliability issues. The comprehensive analysis and synthesis of this chapter’s literature and illustrative content pave the way for subsequent chapters, where innovative solutions and methodologies will be proposed to address these emerging reliability concerns effectively.

- **Chapter 3** examines endurance variability and the “big block issue” of 3D NAND Flash memory. The shift to 3D NAND technology has resulted in Flash memory blocks with many pages, giving rise to challenges in managing these “big blocks” within storage systems. Through experimentation, it is discovered that pages in the top and bottom layers exhibit lower endurance compared to those in the middle layers. This endurance disparity can lead to the underutilization of Flash memory. In response, an intra-block wear-leveling algorithm is proposed, leveraging dynamic, incremental, and layer-aware downsizing of Flash memory blocks to improve overall utilization [13].

- **Chapter 4** explores energy-efficiency associated with storing data in the 3D NAND array. The continual increase in storage density and block size gives rise to a disproportionate increase in latency and energy consumption. This disproportionate energy consumption challenges data integrity during write operations in Flash memory. To address this challenge, a novel
technique called EXPRESS is presented, which increases energy efficiency by exploiting premature termination of Flash write operations. Experimental evaluations demonstrate that EXPRESS reduces energy expenditures by 20%-50% compared to traditional Flash writes, with minimal data integrity loss. The effects of page-to-page variability, program-erase cycling, and data retention on the implementation of EXPRESS are also evaluated, and enhancements are proposed to mitigate their impact [14].

- **Chapter 5** addresses the issue of instant data deletion in NAND Flash memories. Data deletion in NAND Flash memory often incurs significant overhead and contributes to wear levels, posing substantial challenges for maintaining the longevity and efficiency of the memory device. Existing solutions involve unlinking physical page addresses but carry the risk of data leakage. An instant page data sanitization method is proposed to tackle this challenge for NAND Flash memories. This method prevents the leakage of deleted information without affecting valid data in shared pages. The proposed method is implemented and evaluated on commercial 2D NAND and 3D NAND Flash memory chips, demonstrating its effectiveness in preserving data security while maintaining system performance [15].

- **Chapter 6** introduces a novel technique to hide secret information, providing an additional layer of security. By evaluating the accuracy of recovered private data and the detectability of confidential information, this technique ensures that sensitive data remains hidden even in the presence of potential adversaries with low-level access. These contributions in security and pri-
vacy reinforce the overall functionality and reliability of 3D NAND Flash memory, enabling its adoption in applications where data confidentiality is critical.

- **Chapter 7** summarizes the key conclusions of the work and suggests topics and ideas for future study.

The outcomes of this research endeavor are expected to contribute significantly to the advancement of storage systems utilizing 3D NAND Flash memory technology. The proposed techniques will enable the development of highly reliable and energy-efficient storage systems, facilitating the widespread adoption of 3D NAND Flash memory in various applications. Furthermore, the insights gained from this study will pave the way for future innovations in non-volatile memory technologies, addressing the challenges posed by the ever-increasing demand for data storage and processing.
Chapter 2. Background

2.1 History of NAND Flash Memory Development

The genesis of NAND Flash memory dates back to 1967 when Kahng and Sze proposed the first floating gate MOSFET [16]. Equipped with a 5nm thermal oxide layer and a 100nm blocking oxide layer, the transistor retained data for up to one hour, demonstrating its potential for non-volatile memory applications. Intel’s Dov Frohman developed the Intel 1702 in 1971, marking the arrival of the first commercially available Electrically Programmable Read-Only Memory (EPROM) chip. Despite its seemingly modest storage of 2Kb (2048 bits), ‘1702’ was a significant achievement for its time, with a data retention specification of 10 years. However, erase operation for this chip requires strong UV exposure which makes it less convenient.

Efforts to achieve fast erase and longer data retention in EPROM started a competitive race in its development. This led to Fujio Masuoka proposing a new floating gate architecture enabling a faster erasure mechanism for a memory section. He published the first paper on NAND Flash memory at IEDM in 1984 [17, 18]. The term ‘Flash’ was suggested in 1987 due to the fast erase process. Toshiba unveiled the first NAND-type Flash memory in 1989 with accelerated
erase and write times and a reduced chip area per cell, effectively lowering the cost.

The standard form of non-volatile memory throughout the 1990s was Flash memory, serving various digital applications ranging from digital cameras to PDAs and MP3 players. The 2000s marked a period of remarkable growth for NAND Flash memory in mobile devices and solid-state drives (SSDs). Although SanDisk introduced the first commercial NAND-based SSDs in 1991, their widespread adoption did not occur until the mid to late 2000s.

The inception of Multi-Level Cell (MLC) NAND Flash, a breakthrough in memory technology, can be traced back to 1996 [19]. This innovation, initially demonstrated by Intel, enabled the storage of two bits of information per transistor cell, presenting a significant improvement over its predecessor, the Single-Level Cell (SLC) NAND Flash [20]. This advancement amplified the storage capacity and reduced the cost per bit, enabling a new era of data storage efficiency. Almost a decade later, Toshiba introduced the Triple-Level Cell (TLC) NAND Flash in 2009, further pushing the boundaries of Flash memory technology [21]. The advent of TLC, storing three bits of information per cell, increased storage capacities even further. Recognizing its potential, Samsung started mass production of TLC memory a year later, in 2010. In the same year, Toshiba pioneered another innovation in Flash memory—“3D-NAND” [22]. This technology involved stacking memory cells vertically in a grid structure, enabling even greater density and significantly augmenting storage capacity.
Later, the industry saw the introduction of Quad-Level Cell (QLC) and Penta-level cell (PLC) NAND, capable of storing four bits and five of data per cell, respectively. While these developments enabled greater storage densities, they also led to reduced endurance. In the present day, NAND Flash memory stands as a testament to technological evolution. It boasts numerous advantages over other nonvolatile memories, such as high density, smaller cell size per bit, and low cost. These benefits have precipitated an explosive acceleration in the advancement of NAND Flash memory [23]. The remarkable growth rate in storage capacity and reduction in cell size has effectively decreased the cost of data storage.

### 2.2 Fundamentals of NAND Flash Memory

At the heart of NAND Flash memory is a critical technology known as the floating gate metal-oxide-semiconductor field-effect transistor (FG MOSFET). The foundation of NAND Flash memory is the Floating Gate (FG) cell, a variant of the metal-oxide-semiconductor field-effect transistor (MOSFET). As depicted in Figure 2.1(a), this cell resembles a standard MOSFET, featuring source and drain regions and a control gate. However, a distinctive feature is the integration of a floating gate, nestled between the control gate and the silicon substrate and insulated by oxide layers. The floating gate serves a pivotal role, namely the storage of charge. Methods such as Fowler-Nordheim (FN) tunneling, rooted in quantum mechanical effects, facilitate the charging and discharging of the floating gate. This capacity to alter the charge state of the floating gate consequently modulates the cell’s stored data representation. The induced charge on the floating
Figure 2.1: (a) A 2D NAND Flash memory cell. (b) Current sensor used to determine the charge stored in FG.

Gate manipulates the transistor’s threshold voltage ($V_t$), influencing its conductive state during read operations. This control over the threshold voltage is the cornerstone of FG NAND Flash memory data storage. The charge-trap (CT) NAND Flash cells utilize a storage layer instead of a floating gate (FG). The CT layer is typically made of silicon nitride film which is substantially thinner than the FG. The CT cells necessitate a lower programming voltage compared to their FG counterparts, thereby reducing wear on the tunnel oxide layer. CT cells are faster to program and have enhanced endurance, reliability, and decreased latencies associated with NAND Flash memory compared to FG cells.

Figure 2.1(b) presents the sensing mechanism during a read operation. A voltage level denoted as $V_{ref}$ is applied on the gate of a memory cell to perform read operation. If the cell’s $V_t$ falls below the applied $V_{ref}$, the transistor activates,
initiating a current flow via the bit line. This sensed current is translated as a ‘1’ state. Conversely, if the cell’s $V_t$ surpasses $V_{ref}$, the transistor remains off with no current traversing the bit line. The absence of current is perceived as a ‘0’ state. Hence, the NAND Flash memory cell data representation is a charge-based system. The fundamental types of planar NAND Flash memory technologies can be categorized as follows:

- **Single-Level Cell (SLC):** Storing one bit of data per cell, SLC employs two distinct states of charge, leading to two threshold voltage levels. This attribute imparts high reliability, durability, and rapid response time, cementing SLC as the premier grade of Flash memory. However, a higher cost offsets these advantages, making SLC suitable for high-performance and enterprise applications.

- **Multi-Level Cell (MLC):** MLC technology enhances storage capabilities, holding two bits of data per cell. Through precise control of charge stored in the floating gate, each cell can embody one of four states. The increased density and reduced cost per gigabyte of MLC offer a more economical alternative to SLC. However, this is achieved at the expense of poor reliability and slower write speeds.

In recent years, there has been a growing interest in higher-density technologies such as TLC, QLC, and PLC, primarily due to their enhanced storage capabilities and cost efficiency. The subsequent sections provide a detailed overview
of the cell mechanisms involved in the primary Flash operations: programming, erasure, and reading.

2.3 NAND Flash Memory Operations in Cell

**Erase** operation of a NAND Flash FG cell, as illustrated in Figure 2.2(a), involves the removal of charge from the floating gate. The erase operation employs the Fowler-Nordheim tunneling effect, a quantum mechanical phenomenon. A high positive voltage is applied to the substrate, while the control gate is grounded. This voltage arrangement establishes a strong electric field across the tunnel oxide layer, enabling electrons on the floating gate to overcome the energy barrier and tunnel through the oxide layer to the substrate, thus discharging the floating gate. Figure 2.2(b) presents the energy band diagram of the FG cell.

![Figure 2.2](image)

**Figure 2.2:** (a) NAND cell erase operation. (b) Energy band diagram after erase operation.
during the erase operation. When the high positive voltage is applied to the substrate, the conduction band energy of the silicon substrate drops below the Fermi level of the floating gate. This enables electrons in the floating gate to tunnel through the thin oxide layer, reducing the floating gate’s charge and effectively erasing the data. The result is an increase in the cell’s threshold voltage, reverting it to the ‘1’ state. Thus, the interplay of quantum mechanical principles and precise voltage application in FG NAND Flash cells enables data erasure.

Program operation in a NAND Flash FG cell, represented in Figure 2.3(a), entails the injection of charge into the floating gate, effectively changing the stored data in the cell. In this operation, a high voltage is applied to the control gate (\sim 10V), while the drain is moderately biased, and the source and substrate are grounded. The high voltage causes electrons to be injected from the substrate channel into the floating gate through a process known as Fowler-Nordheim tunneling. The electrons are essentially trapped in the floating gate due to an insulating oxide layer. The trapping of these electrons changes the cell’s threshold voltage. With electrons in the floating gate, it now takes a higher voltage to make the cell conductive, effectively changing the state of the cell to ‘0’. Figure 2.3(b) demonstrates the energy band diagram of the FG cell during the programming operation. Applying a high voltage to the control gate raises the energy barrier of the poly-silicon substrate below the Fermi level of the FG, allowing electrons from the substrate to inject into the floating gate through the thin tunnel oxide layer. The charge accumulation in the FG increases the cell’s threshold voltage ($V_t$), transitioning it to the ‘0’ state. In summary, FG NAND
Flash cells utilize the FN tunneling process and meticulous voltage application to enable data programming.

**Read** operation in a NAND Flash FG cell involves applying a reference voltage ($V_{ref}$) to the control gate and interpreting the cell state based on the current flow through the cell as shown in Figure 2.4(a) and 2.4(b). Figure 2.4(a) illustrates the “ON” cell state. When the threshold voltage ($V_t$) of the cell is lower than the applied $V_{ref}$, the cell’s transistor turns on, and a current flows through the bit line. This current is sensed and interpreted as a ‘1’ state. On the other hand, Figure 2.4(b) represents the “OFF” cell state. If the cell’s $V_t$ is higher than the $V_{ref}$, the transistor remains off, and no current flows through the bit line. This absence of current is sensed and interpreted as a ‘0’ state. In summary, NAND Flash FG cells use this sensing mechanism during the read operation to interpret stored data. The ability to control and detect the charge on the floating gate effectively allows NAND Flash memory to store and retrieve data reliably.
Figure 2.4: NAND Flash cell during a read operation (a) ON cell. (b) OFF cell.

2.4 NAND Flash Memory Array Structure

Figure 2.5(a) shows a commercial off-the-shelf Flash memory chip. The chip is internally partitioned into dies, planes, blocks, and finally, individual cells to form an intricate, hierarchical structure. This layout efficiently manages data read, write, and erase operations. Figure 2.5(b) further delves into the hierarchical structure within a chip, focusing on the layout of planes and blocks. A plane is a subdivision within a die and contains a certain number of blocks. Each block is a group of cells that serves as the smallest unit that can be erased in a NAND Flash memory array. Blocks are arranged in a grid-like configuration within a plane, facilitating efficient access and management of data storage.
Figure 2.5: (a) Raw NAND Flash memory chip in BGA package. (b) Memory blocks within the chip are arranged in multiple planes. (c) Circuit diagram of a Flash memory block. (d) 2D NAND Flash memory array.

Figure 2.5(c) elucidates the circuit diagram. This depiction demonstrates how memory cells within a block are electrically interconnected. Each memory block consists of a fixed number of memory pages. The cells in each memory page are electrically connected through a metal WL that acts as their control gate. Each column of cells in a block is connected to a BL. Memory read and program operations are performed at the page-level granularity, while erase operations are performed at the block-level granularity. Any Flash cell set to logic ‘0’ by a page program operation can only be reset to logic ‘1’ by erasing the entire block.

Figure 2.5(d) illustrates the architecture of a 2D NAND Flash memory array. This layout arranges memory cells in a planar or 2D manner, forming a grid-like structure. Each cell, essentially a floating-gate MOSFET, is placed at
the intersection of a word line (WL) and a bit line (BL). This configuration aligns the cells in a matrix with word lines running horizontally and bit lines running vertically.

2.5 NAND Flash Memory Operations in Array

**Block Erase** operation in a NAND Flash memory is a crucial part of the write cycle, and it sets all bits in a block back to a ‘1’ state before new data can be written. The process essentially involves the removal of electrons from the floating gate of each memory cell in a block. Figure 2.6 describes the erase operation in the NAND Flash memory array. The erase operation is performed on a block-by-block basis, meaning all the cells in a block are erased simultaneously.

![NAND Flash Memory Array Diagram](image)

**Figure 2.6:** (a) NAND Flash memory block during erase operation. (b) All the cells will move to the ‘1’ state after block erase operation.
chosen block of cells is isolated from the rest of the array. This is typically done by applying a high voltage (around 20V) to the source line (SL) and the bit lines (BLs) while grounding the word lines (WLs). This creates a strong electric field across the tunnel oxide layer. The strong electric field results in FN tunneling, a quantum mechanical phenomenon. This effect causes electrons to tunnel from the floating gate through the tunnel oxide and to the substrate, thereby reducing the cell’s threshold voltage. After the erase operation, the cells’ threshold voltages are read to verify the successful process. Additional erase pulses might be applied if all the cells are not erased. After the erase operation, all the cells in the block are in a ‘1’ state and are ready to be programmed again.

It’s important to note that the erase operation and the program operation induce stress on the tunnel oxide layer of the Flash memory cells. Repeated programming and erase cycles will eventually lead to wear-out and failure of the cells. This is why NAND Flash memories have a finite lifespan of \(~3,000–100,000\) program-erase cycles, depending on the type of Flash memory.

**Page Program** operation, also referred to as the write operation, in NAND Flash memory is used for writing new data into the memory. The essence of this process is the injection of electrons into the floating gate of each memory cell within a selected page, thereby modifying the cells’ threshold voltage and representing the stored information. This program operation can be done at page-level granularity, unlike the erase operation, which happens on a block level. The selected page is electrically activated within the array by applying a high voltage, typically around 20V, to the respective word line (WL). Concurrently, the other
Figure 2.7: (a) NAND Flash memory block during program operation. (b) All the program cells in a page will move to the ‘0’ state after the page program operation.

Word lines in the same block, especially the neighboring ones, are supplied with a pass voltage \( V_{\text{pass}} \), generally around 7-10V. This \( V_{\text{pass}} \) prevents disturbance of the unselected cells during the program operation. Figure 2.7(a) shows the page program operation in NAND Flash array.

After the programming pulse, a verify operation is performed, where a read operation is executed to confirm if the cell’s threshold voltage has reached the intended level, as shown in Figure 2.7(b). Additional program pulses are applied if the cell’s threshold voltage level is lower than intended. A page program operation in the NAND array utilizes an incremental step pulse program (ISPP) scheme with multiple program cycles, as illustrated in Figure 2.8(a).
Figure 2.8: ISPP scheme (a) An illustration of the ISPP scheme with four program cycles. (b) Evolution of the \( V_t \) state for the program state over consecutive program cycles; the distribution in yellow at the bottom represents the final program state.

Each program cycle consists of a program pluse followed by a verification phase. During the program pulse phase, a high voltage (\(
\sim 15 - 18 \, \text{V} \)) is applied to the corresponding WL to cause the injection of electrons onto the FG/CTs of memory cells that need to be programmed. The verification phase identifies the cells that reached the required threshold voltage, \( V_t \), by performing a page read operation with a program verify voltage, \( V_{\text{ref}}^{\text{PVY}} \), applied on the corre-
sponding WL. Thus, $V_{\text{refPVY}}$ represents the minimum voltage of the program state distribution. The cells whose $V_t$ exceeds $V_{\text{refPVY}}$ are identified as programmed and are subsequently locked out from further programming using a program-inhibit scheme. The following program cycle starts with an incrementally higher voltage on the WL, increasing the chances that cells that did not switch their state in the previous cycle get programmed. This sequence of the program and verify steps continues until most of the cells that are supposed to be programmed are indeed programmed.

Figure 2.8(a) illustrates different steps associated with one-page program operation as a function of time. The steps include a high voltage program pulse phase of duration $t_p$, a relatively lower voltage verify phase of duration $t_{\text{vfy}}$, and two setup time intervals — one for the program pulse of duration $t_{\text{su}}$ and the other for the verify phase $t_{\text{svfy}}$. The primary purpose of the ISPP scheme is to tighten the program state $V_t$-distribution relative to the initial erase state $V_t$-distribution, which is typically wider due to intrinsic cell-to-cell process variation. The evolution of the cell $V_t$ distribution with the ISPP scheme is further illustrated in Figure 2.8(b). For simplicity, we consider an SLC memory, although the same principle holds for MLC and TLC Flash memories. We choose a program operation with four program cycles to illustrate the ISPP scheme. In practice, the number of program cycles could be higher. The distribution depicted with the dashed line represents the right-shifted erase state $V_t$ distribution after each program cycle if all the cells get programmed.
In practice, a certain number of cells that attain $V_t$ exceeding the program verify voltage are locked out (or inhibited) from further programming cycles. Thus, the ISPP scheme tightens the cell $V_t$ distribution by selectively providing fewer program pulses to the fast program cells and more program pulses to the slow program cells. As a result, the final program state distribution becomes much tighter than the erase state, as illustrated with the yellow distribution in Figure 2.8(b). Since tighter $V_t$ distribution is essential for ensuring data integrity, the ISPP scheme is invariably used in all NAND Flash memories. The $V_t$ distributions in Figure 2.8(b) may not follow the perfect Gaussian distribution. We used Gaussian-like distribution for illustration purposes only. Thus, it is not a faithful illustration of actual cell $V_t$ distributions. The wider an erase state $V_t$ distribution is, the more significant number of program cycles is required to complete the write operation. Note that the slow program cells may require several additional ISPP cycles. The percentage of such cells in practice falls well below 1% of all Flash cells on a page. Thus, the ISPP scheme entails a disproportionate energy-accuracy trade-off, where a significant fraction of program time and, therefore, energy is spent for programming a tiny fraction of memory cells. The energy-accuracy trade-off is even more skewed for 3D NAND technology, which exhibits significant cell-to-cell variations due to poly-Si channel material and non-uniformity in the cell dimensions caused by the reactive ion etching process [24, 25]. Thus, long-tail erase state distribution is a fundamental nature of 3D NAND. Hence, the energy-accuracy trade-offs in the ISPP programming scheme of the 3D NAND need to be evaluated carefully for energy-efficient storage applications.
Page Read in NAND Flash memory is depicted in Figure 2.9. Read operations are also performed at a page level, like the page program operation. A page read operation in the NAND array involves applying a read reference voltage ($V_{\text{ref}}$) on the selected page’s WL and sensing the threshold voltage of the cells connected to that WL. The WLs of all other pages in the selected block are set to a high voltage ($V_{\text{Rpass}}$), turning on all Flash cells from the non-selected pages. This way, the state of the selected page can be sensed through the bit lines. An erased cell conducts the current, which is sensed as a logic ‘1’, whereas a programmed cell does not conduct the current, which is sensed as a logic ‘0’. The read reference voltage is set between the erased state and the programmed state distributions to identify the cell states correctly.
In the case of MLC or TLC memory, where each cell can be in one of four or eight states, respectively, a series of read operations are performed with different $V_{\text{ref}}$ values to distinguish between the states. The exact $V_{\text{ref}}$ values used for these operations depend on the specific $V_t$ distributions for the different states in the memory device. The $V_t$ distributions for different memory cell types are discussed next.

**NAND $V_t$ distributions:** SLC (Single-Level Cell) Flash memory cells are engineered to store 1 bit of data, leading to two potential states: $L_0$ and $L_1$, as represented in Figure 2.10(SLC). In this case, the single bit can be either ‘0’ or ‘1’, which corresponds to the memory cell’s state. Upon erasing a block, all cells revert to the erased state, $L_0$. Programming involves injecting charges onto selected cells’ floating gates to transition their state from $L_0$ to $L_1$, creating two distinct $V_t$ states within the memory array. Consequently, only one read reference voltage is needed to interpret the data stored in SLC Flash memory.

MLC Flash memory cells store two bits of data, and they are programmed to have their $V_t$ in one of four different states, $L_0, L_1, L_2$ and $L_3$ as shown in Figure 2.10(MLC). The data bits corresponding to each $V_t$ state are shown in two colors to indicate the most significant bit (MSB) in red and the least significant bit (LSB) in blue. Gray’s code is commonly used to encode the states ($L_0 = 11, L_1 = 01, L_2 = 00$ and $L_3 = 10$). The MSB bits of all memory cells’ states in a row form the MSB page. Similarly, the LSB bits of all the memory cells’ states form the LSB page. Thus, every LSB page has a corresponding MSB page, and they are called shared pages because they share the same set of physical memory.
Figure 2.10: NAND Flash memory state coding for SLC, MLC and TLC memory.

cells. When a block is erased, all charges are removed from the cells’ floating gate (FG), placing the cells into the erased state ($L_0$). Programming an LSB page involves injecting charges on selected cells’ FG to move their state from $L_0$ to $L_2$. The corresponding MSB page is programmed after the LSB page programming is finished. During MSB page programming, certain cells transition from $L_0$ to $L_1$ and certain cells transition from $L_2$ to $L_3$. Thus, after programming both pages, four $V_t$ states are created in the memory array. Two read reference voltages are used to read the MSB page data ($V_{ref}^{MSB}$ and $V_{ref}^{MSB}_2$), whereas only one read reference voltage is needed to read the LSB page data ($V_{ref}^{LSB}$).
TLC (Triple-Level Cell) Flash memory cells are designed to store three bits of data, thus enabling eight potential states: $L_0$, $L_1$, $L_2$, $L_3$, $L_4$, $L_5$, $L_6$, and $L_7$, as illustrated in Figure 2.10 (TLC). The three bits, which include the extra page bit (XP), upper page bit (UP), and the least significant bit (LSB), correspond to the state of the memory cell. Gray’s code is typically employed to encode these states ($L_0 = 111$, $L_1 = 011$, $L_2 = 001$, $L_3 = 101$, $L_4 = 100$, $L_5 = 000$, $L_6 = 010$, $L_7 = 110$). Each of these shared pages is programmed sequentially, with each adding an additional level of charge to selected memory cells. Hence, after programming all pages, eight distinct $V_t$ states are established in the memory array. Due to this complexity, more read reference voltages are needed to decipher the data stored in TLC Flash memory, as depicted in Figure 2.10(TLC).

2.6 3D NAND Flash Memory

3D Floating Gate NAND Flash cell is structured similarly to its 2D counterpart, with key differences to accommodate the vertical or 3D structure. The essential principle remains the same: information is stored as charges in a “Floating Gate” that is insulated from the rest of the transistor. As shown in Figure 2.11(a), the floating gate is entirely surrounded by an insulating oxide layer, and the control gate surrounds this structure, akin to the Charge Trap (CT) design. However, unlike CT Flash, where the charge is stored in silicon nitride, the FG Flash stores the charge in the polysilicon floating gate. When a voltage is applied to the control gate, it induces a charge in the floating gate. The presence
Figure 2.11: (a) 3D NAND Flash memory cell. (b) 3D NAND Flash memory vertical array structure.

or absence of this charge can be detected when a read voltage is applied, representing a stored bit of data. The building block of the 3D NAND Flash array is a vertical stack or “string” of memory cells. Figure 2.11(b) demonstrates the innovative construction of a 3D NAND Flash memory array. This string is a stack of floating gate transistors arranged vertically along a channel hole. Referring to Figure 2.11(b), these channel holes, shown as vertical pillars in yellow color, contain the channels of the Flash memory cells. Each memory cell within the string is connected to a separate word line (represented as green layers, $WL_0 - WL_n$ in the figure), arranged horizontally in layers throughout the 3D structure. Between the layers of word lines are layers of oxide-nitride-oxide (ONO) which serve as the charge storage layer (FG) for each memory cell. The control gate surrounds
them, also known as the WL, on one side and by the channel on the other side. At the ends of each memory cell string are select gate transistors, often called string selection line (SSL) and ground selection line (GSL), which allow the string to be connected to or isolated from the bit line and the ground line, respectively. The bit line runs perpendicularly to the word lines (the red pillars in Figure 2.11(b)), connecting each string of cells in a block to the peripheral circuitry that controls the operations of reading from and writing to the cells.

Figure 2.12 shows the circuit diagram of a 3D NAND Flash memory block. A 3D NAND Flash block consists of multiple “strings,” where each string is a vertical stack of memory cells arranged along a channel hole etched into a substrate. These memory cells are connected via word lines (WLs) that run horizontally through the 3D stack. Each string is linked to a bit line (BL) at the top and grounded at the bottom, forming the basic unit of the NAND structure. The bit lines, running perpendicular to the word lines, provide a path for the current to flow when a memory cell is read or programmed. At both ends of each string, there are select gates: SSL and GSL. The SSL, located at the top, connects the string to the bit line. The GSL, at the bottom, connects the string to the ground. They serve to isolate the memory cells in a string during read, program, and erase operations, minimizing interference between the strings. Overall, this 3D NAND block circuit diagram helps visualize how the individual memory cells, organized in strings, interact with each other and with external signals during various operations. It is important to note that the descriptions and terminologies might vary based on the architecture and specific design of the 3D NAND Flash memory.
For a more detailed and precise definition, referring to a particular manufacturer’s data sheet or related technical documentation is recommended. 3D NAND technology allows for stacking many layers of cells, significantly increasing the density of memory cells on a chip. This results in increased storage capacity without having to reduce the size of the cells, which is a significant challenge in scaling planar NAND technology. However, the fabrication process of 3D NAND Flash memory is complex and described next.

Figure 2.12: Circuit diagram of a 3D NAND Flash memory block.
2.7 Fabrication of NAND Flash Memory

Figure 2.13(a) shows the 3D NAND Flash memory cell film stack. Unlike the 2D NAND array with a planar structure, the 3D NAND has a layered structure consisting of alternate metal and insulating layers. The layered structure is created by depositing alternating oxide and metal layers. The metal layers are shown with green color, whereas intermediate insulating layers are kept transparent in Figure 2.13(a). Each metal layer forms a word line of the memory array connecting multiple memory cells. The number of layers depends on the design specification and can range from tens to hundreds in more recent technology. The channel of the memory cell is formed by creating a hole through the 3D stack with the reactive ion etching (RIE) process. Vertical yellow pillars in Figure 2.13(b) are memory holes that contain Flash memory cells. The gate stack of the cells is then formed by sequentially depositing blocking oxide (Oxide-nitride-oxide), a charge trap layer or floating gate, and tunnel oxide along the sidewall of the cylindrical trench.

Polysilicon is used to form the FG, which stores the charge that represents the stored data. It is deposited on the gate oxide layer using a process known as Chemical Vapor Deposition (CVD) [26]. Interpoly Dielectric, which separates the floating gate from the control gate (shown in pink color in Figure 2.13(c)), is formed from an oxide-nitride-oxide (ONO) sandwich. This is also deposited using CVD [27]. Figure 2.13(c) shows the vertical cross-section of the 3D array, illustrating different layers of the gate stack. Each memory block consists of a
Each layer height = 80 nm; Total height: \( H = 176 \times 80 \text{ nm} \)

\[ D \approx 50 - 100 \text{ nm} \]

**Figure 2.13:** (a) Film stack deposition. (b) High aspect ratio etching (RIE). (c) Vertical cross-section of the 3D NAND gate stack.

The tapered shape of the cylindrical memory hole is due to the high aspect ratio that makes the RIE inefficient at the bottom layers. Thus, memory cells in the bottom layers typically have a smaller diameter than the ones in the top layers. The unique tapered shape of the array structure leads to significant layer-to-layer variability, as reported by several recent papers [28, 29, 30, 31].

2.8 Flash-based Storage System

Figure 2.14(a) shows a simplified view of a Flash memory-based storage system consisting of a Flash memory controller and one or more NAND Flash
memory chips. The Flash controller manages the Flash media by executing several tasks defined in the FTL. FTL provides an interface between the host file system on one side and the physical NAND Flash memory chips on the other side by mapping the logical addresses to physical addresses in NAND Flash, as shown in Figure 2.14(b). In addition, FTL contains firmware modules that perform error correction, garbage collection, and wear-leveling [32]. The garbage collection module periodically reclaims all the invalid pages in the media to perform a block-erase operation, freeing up memory space for new data. The wear-leveling module manages the limited endurance of the Flash media by ensuring uniform program-erase operations across different NAND Flash memory chips and blocks within a chip.
2.9 Experimental Setups

The experimental setup used in the study consists of a TSOP-48 socket/BGA-132 socket that holds the Flash memory chip, an FT2232H mini module from Future Technology Devices International (FTDI), and a workstation. The FT2232H module acts as a bridge between the workstation and the device, implementing an asynchronous 8-bit parallel interface to the device. A software package running on the workstation executes the ONFI commands and works as the FTL for the Flash device. The FLT executes the ONFI commands for sending data to the Flash memory chip, erasing a block, writing a page, reading a page, or retrieving the data from the device. This hardware setup allows us to access raw memory bits without any error correction. We also use a logic analyzer and a Digilent Analog Discovery II multi-function instrument to measure time and capture voltage samples from a shunt resistor connected to the power line of the memory chip holding sockets.

Figure 2.15: Experimental set-up for interfacing raw COTS NAND memory chips.
2.10 Reliability Issues in 3D NAND Flash Memory

The technological advancement comes with the cost of reliability and performance issues for NAND Flash memory [33, 34, 35]. Reliability for NAND Flash is defined as the integrity of data stored inside the storage memory. The study discusses fundamental reliability constraints for NAND Flash memory and emerging reliability issues as technology advances from planner to vertical architectures.

The decrease in cell size results in severe cell-to-cell interference in the planner 2D architecture of NAND Flash memory [36]. This limitation provokes a vertical 3D layer structured architecture for NAND Flash memory [37, 38, 39]. While the new architectural evolution solved the miniaturization problem of increasing storage density, new reliability issues also emerged. Hence, understanding the reliability concerns for NAND Flash memory is crucial for advancing the storage industry. This section discusses some well-known reliability issues of NAND Flash memory and the new reliability issues that are reported in the literature for 3D NAND Flash memory.

2.10.1 Program Disturb

Program disturb occurs to the cells in the same word line as the selected cell but in another bit line (BL) set to program inhibit (red box in Figure 2.16). Program-inhibit boosts the channel potential with $V_{ch}$, but a large number of program pulses may issue soft programs to those inhibited cells. The effecting program voltage for those cells is $V_{prg} - V_{ch}$. $V_{ppass}$ voltages are provided to the
other WLs to boost the channel potential $V_{ch}$. So, by raising the $V_{pass}$ voltages, the effect of the program disturb can be reduced. With the increase of $V_{pass}$ to 

![Image](image_url)

**Figure 2.16:** (a) Various disturb mechanism in NAND array. (b) Program disturb in 3D NAND (collected from [1]).

reduce program disturbance, the effective program voltage on the unselected WLs will increase. This high pass voltage will increase pass disturb on the cells located in the same BL as the selected cell to program (green boxes in Figure 2.16)(a). The channel potential is set to ground, and the gate voltage is set to $V_{pass}$. So, the effecting programming voltage is $V_{pass} - 0V$. So these green cells will be weakly programmed, and this phenomenon is called the pass disturb. Usually, there is no pass distursb on the cells of the unselected WLs in program-inhibited NAND bit lines(yellow region). The self-boosted channel potential shows only minor differences, which is insufficient for tunneling. However, in NAND strings-based cell concepts with a high boosting efficiency like SOI devices [40] or gate-all-around
concepts, a negative pass disturbs (soft erase) during the program operation can be expected and recently reported in [41].

As discussed earlier, the $V_{ppass}$ voltages are applied to the adjacent WLs of the selected WL (for page program) to reduce the program disturbance. For $WL_0/WL_n$, the $V_{ppass}$ voltage is different in one of the adjacent gates. Especially, the GSL (ground selector) gate is set to GND. Hence, a large GIDL (gate-induced drain leakage current) current may be generated at the drain edge of the GSL transistor as the potential at the drain node of the GSL transistor is raised by channel boosting. Electron-hole pair generation will follow, and the generated electrons will be accelerated at the GSL-$WL_0$ space region. These accelerated electrons can be hot enough to be injected into the floating gate of the cells of $WL_0$ transistor in the inhibited BLs. Thus, the threshold voltage of $WL_0$ will be shifted to a positive direction by the hot-electron injection[42]. In summary, the cells in the edge WL (blue cell in Figure 2.16(a)) will be affected by the potential drop between the select transistor and the program-inhibited NAND string with its boosted channel potential. The accelerated electrons will move to the edge cell and change their threshold voltages. The dummy WLs are adopted to shield the edge disturbs, as mentioned in [41, 42, 43].

In a conventional 2D planar NAND Flash cell, there is only one cell string per block and corresponding program disturbance mode. Channel self-boosting is obtained by applying a bit line voltage. On the other hand, three kinds of program disturb are reported for 3D NAND Flash memory[44]. The bit lines in vertical NAND Flash memory consist of several strings, as shown in Figure
2.16(b). Due to its architectural variations, three modes of program disturbance exist in 3D NAND Flash. The Y-mode disturbs dominate in 3D, whereas X-mode disturbs 2D Flash memory. By providing an asymmetric pass voltage, Kwon et al. proposed to reduce the 3D program disturbance phenomenon [45]. It is reported that both the initial precharged channel potential and leakage current from the channel to the bit-line impact the program disturbance [46]. Zhang et al. proposed an erase-assisted precharge scheme [47] to reduce this Y-mode disturbance in 3D NAND.

2.10.2 Read Disturb

Read disturbs are the most frequent source of disturbance in NAND architectures. This kind of disturbance may occur when reading many times the same cell without any erase operation. Read operations happen at a page-level granularity in NAND Flash memory. During a read operation, read voltage \(V_r\) is applied to the selected WL (Figure 2.17). Where a read pass voltage, \(V_{r\text{pass}}\) is applied to other unselected WLs. Though these other cells are not being read, this high pass-through voltage induces electric tunneling that can shift the threshold voltages of these unread cells to higher values, thereby disturbing the cell contents on a read operation to a neighboring page [48]. Zhang et al. proposed a new read scheme to suppress the read disturb mechanism by reversing the read sequence and skipping the precharge of the select gates [49].
Figure 2.17: Read disturb in NAND Flash memory (collected from [2]).

2.10.3 Endurance

Endurance is defined as the maximum number of program erase (PE) cycles a Flash memory block can endure with data integrity. In NAND Flash cells, program and erase operations rely on charge transport through thin oxides; this is accomplished via Fowler-Nordheim (FN) tunneling into/from a storage layer, which can be either a polysilicon FG or an interfacial trapping layer in CT technology. The block-erase operation in NAND Flash memory involves a high electric field and damages the insulating oxide layers. The damaged oxide layer can trap charges, and this quantity increases with a higher number of PE cycles [42]. The stored charges in the FG or CT memory will be trapped in the oxides, deteriorating the efficiency of page program operation. If the amount of trapped is high enough in the oxides, the program/erase verification will fail for that block and the block will be marked as a bad block. Hence, it limits the number of program and erase cycles of a NAND Flash memory block. As the
number of PE cycles increases, the program time is expected to reduce, whereas the erase time is expected to grow. The endurance of 2D Flash memory is higher than 3D vertical NAND Flash. Also, SLC (1bit/cell) has higher endurance than memory with multi-bits/cell (MLC, TLC, QLC).

2.10.4 Data Retention

Data retention loss can be defined as the change in the threshold voltage of the Flash memory cells while the chip is idle. The tunneling damages the tunnel oxide and traps charges [50]. The number of trapped charges increases with the PE cycle. These trap charges in FG form an electric tunnel, and they leak away much faster through the intrinsic electric field [51]. The phenomenon can be modeled with the percolation concept [52]. The percolation paths hamper the insulating properties of the gate dielectric and result in data retention failure. This is called trap-assisted tunneling or TAT. The threshold voltage will decrease due to the TAT mechanism.

The trapped charges can be detrapped over time [53, 54]. The polarity of the trap charges can be either positive or negative. Hence, the charge detrapping mechanism can increase or decrease the cell threshold voltage, depending on the polarity of the detrapped charges [51]. It is clear that these mechanisms are strongly related to oxide degradation, and therefore, data retention decreases with the number of applied writing pulses. In the case of cells with multiple-bit information, programmed at higher threshold voltage are more prone to data retention issues. The threshold voltage distribution of the higher state cells has
higher voltages and shifts faster than lower states [51] because of the accelerated TAT mechanism.

Charge trap (CT) memory cell data retention has two main discharging paths: tunneling from the storage layers to the conduction band of the substrate gate and thermal emission from traps to the conduction band of the storage layer[55]. Tunneling is more dominant than thermal emission for the traps close to the substrate, while for the traps far away from the substrate, thermal emission is the primary mechanism for data retention[56].

---

**Figure 2.18:** Fast electron detrapping in CT NAND Flash right after program operation (collected from [3]).

The stress from the program/erase cycles makes the conventional 2D NAND (FG-Flash) cell gradually lose charge over its retention period. On the other hand, 3D NAND demonstrates a quick drop in threshold voltage relatively soon after
a write/program operation, as shown in Figure 2.18. Chen et al. [57] reported a fast charge loss phenomenon in charge trap memory. The fast charge loss may come from shallowly trapped electrons that immediately de-trap and tunnel out of the silicone nitride after the programming pulse is terminated.

Another unique data retention reliability concern for charge trap Flash memory is its lateral charge migration (LCM). When the electrons enter the storage layer, they will move to the space region between adjacent WLs [18, 45], especially when storing the cold data, as shown in Figure 2.19. Besides, the degree of LCM is related to the number of electrons stored in the storage layer of the adjacent WLs [58]. When the difference in the number of electrons stored in adjacent storage layers is larger, the possibility of LCM will increase when the threshold voltage difference between adjacent WLs is larger [4, 58, 59, 60]. Hence, charge trap NAND Flash memory suffers from both vertical and lateral charge loss.

### 2.10.5 Program Errors

Program errors are due to the inaccurate increase of cell threshold voltage during programming [61]. The cells with different parameters require different program pulse numbers to reach a quasi-static equilibrium state. This variation increases the cell threshold voltage distribution after programming. This variation motivates the introduction of program-and-verify with the inhibited cell mechanism [62]. The ISPP (incremental step pulse programming) is applied for a page program operation [63]. Program-and-verify algorithms made of multi-
Figure 2.19: Lateral charge migration in 3D charge trap NAND Flash memory (collected from [4]).

Multiple programming pulses in ISPP stop the increase of cell threshold voltage after reaching the desired voltage level [64]. Inaccuracies in threshold voltage placement typically consist of cell over-programming. This inaccuracy may arise from fundamental fluctuations in the number of electrons tunneling from the channel to the storage layer of the memory cells (typically called program noise) [64, 65], from anomalous or erratic tunneling in the presence of tunnel-oxide defects [66, 67] or from abrupt changes in the tunneling rate due to floating-gate depletion effects [68, 69]. This issue is more dominant in higher bits/cell memory as their page program involves a complicated two-step foggy-fine algorithm [70, 71].
2.10.6 Temperature Dependency and Cross-temperature Effect

The 3D NAND Flash has a less absolute current at low temperatures because the undoped polysilicon channel has lower thermal emission at a lower temperature [72]. The monocrystalline p-doped silicon used in 2D NAND Flash behavior is the opposite of 3D [9, 73]. The magnitude of the total current is lower in 3D. Also, the temperature dependency on cell current raises reliability concerns, namely the cross-temperature effect [74]. The cross temperature can be defined as performing program and read operations in different temperatures. Zhao et al. found that the high-temperature program and low-temperature read will shift the cell threshold voltage higher, and the upper tail of the distribution will be stretched more. In the opposite case, the lower tail of the distribution will stretch more than the upper tails [47]. The cell threshold voltage increases with reading temperature due to the grain boundary trap effect in the polysilicon channel [75].

2.10.7 Open Block Issue

While big blocks aid NAND Flash memory with higher density; they can impact performance significantly due to high RBER. Recently, Li et al. observed the blocks that take a long time to close and reported open block issues [76]. The word lines of an open block experience different threshold voltages than those of the closed blocks. The threshold voltages of the closed block word lines are much higher than the open block WLs. The charge leakage of the WLs that reside near
the boundary WLs in the first open period is much more significant than others in the same block [4, 76, 77, 78, 79].

2.10.8 Temporal Read Errors

Zambelli et al. [80] reported a temporary read issue in 3D NAND Flash memory when a read operation is performed after an idle state. They mention it as a temporary read error (TRE). The TRE issue is due to a transient instability of cell VT occurring during the idle phase and the first read operations performed on a block. Xia et al. proposed that the grain boundary traps (GBTs) discharging during the idle time lead to the low GBTs occupancy, which may be the reason for the high temporary RBER in the first read [80, 81].
Chapter 3. Layer-to-Layer Endurance Variation of 3D NAND Flash Memory

3.1 Introduction

With the continued scaling down of 2D NAND Flash technology to nodes below 15 nm, fabrication and reliability challenges have become more pronounced [64, 82]. These challenges encompass higher lithography costs and a host of fundamental cell reliability issues such as random telegraph noise, program noise, and cell-to-cell interference. These issues have prompted the Flash memory industry to transition towards 3D NAND Flash fabrication processes, which inherently present a monolithic design.

As we progress with this technological transition, the increase in the number of pages, and layers per Flash memory block has introduced complex big block management issues in the Flash Translation Layer (FTL). Figure 3.1(a) shows the trends for the increase in the number of layers and pages for the 3D NAND Flash memory block. It is apparent that the number of layers will increase further in the future, leading to a more complex FTL. The wear-leveling algorithms of the FTL are designed to distribute the program/erase (PE) cycles evenly across all blocks. The endurance of a block is predominantly determined by the weakest page in that block. Consequently, a block is labeled as “bad” when the FTL es-
Figure 3.1: (a) Trends for increasing number of layers and pages in 3D NAND Flash. (b) Block endurance leaves the storage capacity underutilized.

Establishes that the integrity of the data stored in a page cannot be ensured. This usually transpires when the number of bit errors surpasses the error-correction codes’ capability during the reading process of a page or when the controller fails in its attempt to erase a block. Such a mechanism potentially leads to the underutilization of storage resources, especially in circumstances where there is a significant disparity in endurance among the pages within a memory block, as shown in Figure 3.1(b). Thus, it becomes paramount to comprehend the variability in endurance among different pages in a 3D NAND memory block to efficiently utilize the available storage resources. As the intricacies of managing larger 3D NAND blocks grow, there is an exigent need for refined methodologies to leverage the full potential of these memory devices.
This chapter provides a systematic analysis of the endurance of individual pages in relation to their physical location within the 3D array. It has been observed that pages situated at the edge layers of the 3D stack deteriorate faster than those located in the middle layers. The root cause of this endurance variation is scrutinized, revealing a fundamental connection to the layer-to-layer erase-state threshold voltage \( V_t \) variation [83]. In order to mitigate the impacts of uneven page endurance and enhance the overall usage of 3D NAND Flash memories, an intra-block wear-leveling algorithm is proposed. This algorithm is predicated on dynamic, incremental, and layer-aware downsizing of Flash memory blocks. Qualitative analysis indicates that this strategy can considerably enhance the endurance of storage systems. The rest of the chapter discusses the experimental methodology used in this study, the results of the endurance characterization, elucidates root causes for the endurance variation, and describes how an intra-block wear-leveling algorithm can leverage it to improve the overall block endurance.

3.2 Experimental Methodology

For the reliability characterization, the entire memory block is programmed with a random data pattern and read back the data immediately after programming. The tests were performed for a commercial 32-layer 3D NAND Flash memory. The word lines and layers are used interchangeably in the thesis. Details about the 3D NAND structure are described in chapter 2. A raw bit error rate (RBER) is determined for each page in each layer of the Flash block. Then an average bit error rate of all the pages in each layer is calculated to determine the
layer-specific RBER. Then we analyze the variation of RBERs among different layers. To understand the endurance variation among different layers as a function of the array’s wear level, we perform repeated program-erase (PE) operations on different blocks and repeat the experiment to determine the layer-specific RBERs. The RBERs determined immediately after programming may not represent the actual endurance status as Flash memory is expected to retain data for a long time. For more realistic endurance characterization, we perform accelerated data retention (DR) tests by baking the memory chip at a high temperature (120°C) for 1, 2, and 3 hours. We then perform read operations to determine RBERs on both fresh (PE = 0 k) and PE cycled Flash memory blocks (PE = 20 k).

3.3 Results and Discussion

3.3.1 Experimental Results

We first present the layer-dependent RBERs for the fresh and 20K PE cycled blocks in Figure 3.2(a) and 3.2(b), respectively. RBER is analyzed for four DR conditions (0-3 hours bake time). Each point in Figure 3.2 represents the average RBER of 16 pages located in that layer. The fresh blocks have a very low RBER, with the middle layers having slightly higher RBER than the edge layers, as shown in Figure 3.2(a). Changing the DR conditions does not significantly impact the fresh block’s RBERs. However, the RBERs of the PE cycled blocks show a very different behavior, as shown in Figure 3.2. First, the RBERs remain very low without accelerated aging (0 hours bake time). In contrast, the RBERs
Figure 3.2: RBER of pages in a block as a function of the layer number. (a) Fresh condition (b) 20K PE cycles. (c) Layer-dependent RBER for different PE cycle counts. (d) RBER from two different chips with similar PE cycle counts (PE = 20K) and 3 hours of bake time. Different lines represent different memory blocks of the chip.

After 1, 2, and 3 hours of baking show a characteristic profile. The RBERs of the bottom layers ($L_2 - L_{14}$) are significantly higher than RBERs of all other layers. Next, the RBERs of pages close to the top ($L_{28} - L_{31}$) are also slightly higher than the RBERs of the pages located in the middle layers. Comparing the effects of DR=3 hours vs. DR=1 hour, the increases in the RBERs on edge layers are significantly larger than the increases in RBERs of the middle layers.
Figure 3.2(c) shows the layer-dependent RBERs for different PE cycle conditions (0, 5K, 10K, 15K, and 20K), while DR is fixed and set to 3 hours of baking time. It is interesting to note that RBERs remain relatively low and flat for fresh memory blocks. The characteristic RBER profile becomes more prominent as we gradually increase the number of PE cycles. This result confirms that edge layers, especially bottom layers, degrade faster than the middle layers of the 3D stack. Figure 3.2(d) profiles layer-dependent RBERs gathered from two different memory chips with the same part number. Data are collected from multiple memory blocks (~10 blocks per chip) that are exposed to the same PE cycle count and 3 hours of baking. Even though the absolute RBER values vary from chip to chip, the layer-dependent RBER profile is found to be quite robust across different memory blocks and memory chips. In all cases, we find that pages in the bottom layers have the highest RBER. The RBERs gradually decrease as we move from the bottom to the upper layers; however, they slightly increase for a few top layers. In the next section, we provide a root cause analysis for this characteristic RBER profile.

3.3.2 Root-cause Analysis

We postulate that layer-dependent endurance variation originates from the unique architecture of the 3D NAND memory array, which fundamentally dictates uneven erase speed between different layers of the 3D stack. Specifically, the root causes of endurance variation are as follows: (a) layer-dependent structural
variation – uneven cell diameters and (b) layer-dependent electrical variation - uneven erase voltage distribution.

(a) Uneven cell diameters

The monolithic 3D fabrication process of 3D NAND dictates a tapered shape of the cylindrical memory hole, as shown in Figure 3.3(a). The tapering is caused by the memory hole’s high aspect ratio, which makes the RIE inefficient at the bottom layers. Thus, memory cells at the bottom layers typically have a smaller diameter ($D$) than cells in the upper layers. The diameter difference ($\Delta D$) between the top and bottom layer cells depends on the tapering angle ($\theta$) and the height of the 3D stack ($H$) as follows: $\Delta D \sim 2H \cdot \tan(\theta)$. The exact values for the tapering angle and the height of the 3D stack are proprietary information. However, we can estimate $\Delta D$ by assuming typical values of array dimensions from the published literature [84]. Assuming tapering angle, $\theta \sim 0.5\degree$, and height of the 3D stack, $H \approx 32 \times 80$ nm = 2.56 $\mu$m, we can estimate the diameter difference $\Delta D \sim 45$ nm for a 32-layer stack. Since the typical diameter of the memory hole is $\sim 90$ nm, the estimated diameter difference is quite significant. It may result in roughly two times smaller diameter in the bottom layer cells compared to the top layer cells. Since the electric field in the oxide layers is inversely proportional to the cell diameter, cells in the bottom layer experience a significantly higher electric field during erase operation than cells in the middle and upper layers. Since the probability of damaging the oxide layer increases
exponentially with the magnitude of the electrical field across the oxide [85], memory cells at the bottom layer experience faster cell degradation.

Figure 3.3: (a) Tapered shape of the cylindrical memory hole. (b) Erase pulse delivery in 3D array. (c) $V_t$ distribution of edge (blue) and middle (red) layers. Higher erase depth in edge layers creates a higher shift in threshold voltage ($\Delta V_{t_{\text{bot}}} > \Delta V_{t_{\text{mid}}}$).

(b) Uneven erase voltage distribution

Erase operation takes place at a block level, meaning all memory layers of a given block are erased together. The erase voltage is typically applied through the Si substrate and the bit lines while all the word lines are grounded (Figure 3.3(b)). Since the erase voltage gradually propagates into the vertical memory channel through the edge layers (top and bottom), memory cells at the edge layers see high voltages during block erase, as illustrated in Figure 3.3(b). As a result, their gate oxides are stressed a bit more relative to the cells in the middle. The degradation rate in the bottom layer cells is the highest due to the combined effects of a relatively smaller diameter and higher erase voltage. The cells at the top layers, even though they have a larger diameter, experience higher erase voltage which causes a slightly higher electric field in the gate oxide. Such
uneven erase voltage in different layers is inevitable and will be amplified as more layers are stacked in future generations of 3D NAND Flash. The uneven erase voltage between different memory layers amplifies the layer-to-layer endurance difference as the memory block is PE cycled. Figure 3.3(c) illustrates the layer-dependent erase $V_t$ variation using threshold voltage distributions. We choose SLC memory for simplicity, but the reasoning holds for multi-bit Flash memories. Note that ISPP (details in chapter 2) scheme is used during page programming – multiple voltage pulses, each followed by a verify phase and program prohibit scheme. This mechanism ensures that cells in the programmed state have similar $V_t$ distribution, irrespective of their layer number, as shown in Figure 3.3(c). Since an erase operation takes place on all the layers simultaneously, the faster-to-erase layers (e.g., bottom layers) will experience a higher $V_t$ shift compared to slow-to-erase layers (e.g., middle layers). In other words, the bottom layer will be deeply erased whereas the middle layers are shallowly erased. Since shallow erase improves endurance [85], the middle layers exhibit better endurance compared to the bottom layers.

3.3.3 Validation of Root-cause

To validate the layer-dependent erase $V_t$ hypothesis from above, ideally, we would measure the $V_t$ distribution for cells in different layers after a block-erase operation. However, the chip under test does not allow for such profiling. Instead, we validate this hypothesis by measuring the percentage of the erased bits as a function of the layer number by varying partial erase time. The experiment flow is
as follows. All cells in a block are programmed, and an erase operation is started and terminated prematurely. The solid black line in Figure 3.4(a) represents the nominal block erase time $t_e$. We terminate the erase operation before it is completed using a RESET command at a partial erase time $t_{pe}$ as shown by the dashed red line in Figure 3.4(a). The state of the cells is determined by reading all pages in the block. Figure 3.4(b) illustrates the state of a block after a particular partial erase time is used. Note that only faster-to-erase cells will change their state from 0 to 1. Slow-to-erase cells will remain at logic 0. We count the total number of erased bits in a page.

Figure 3.4: (a) Block erase operation (solid black line) and partial erase operation (dashed red line) (b) Transitions of cells state through a partial erase operation. (c) The percentage of erased bits as a function of layer number for different partial erase times - ‘U’ curves show that cells in edge layers are faster to erase than cells in middle layers.

Figure 3.4(c) shows the percentage of the erased cells as a function of the layer number. Different colors in Figure 3.4(c) represent different partial erase times. For a given partial erase time, we observe a clear trend in the number of erased cells as a function of the layer number. Please note that the percentage of the erased bits correlates strongly with the erase speed in that layer, where a
higher percentage of erased bits implies a faster erase speed. Thus, the partial erase data on the 3D memory block illustrates that the middle layers are slow to erase compared to the top and upper-layer cells.

3.4 Intra-block Wear-leveling

Layer-to-layer endurance variation of 3D NAND memory may lead to severe underutilization of a memory block. For example, an entire Flash block will be marked as bad even though just a few worn-out pages on the bottom layer may reach RBER that exceeds levels that can be corrected using error correction codes (ECC). As the number of layers and thus pages in a block keeps increasing with each new generation of 3D NAND Flash chips, this uneven layer-to-layer endurance variation may be a limiting factor resulting in either underutilization of storage media and/or reduced data integrity. Reduced data integrity will require a more sophisticated ECC, limiting the throughput and increasing the latency of Flash operations. To improve the utilization of storage media and its endurance, we propose an intra-block wear-leveling algorithm that exploits the observed endurance variation within 3D Flash arrays.

Figure 3.5 illustrates the proposed algorithm. We explain our algorithm using an MLC memory block, but the proposed algorithm applies to all other multi-bit Flash memories. Figure 3.5 shows an \( n \)-layer 3D stack corresponding to a sub-block, where a word line is shared by corresponding LSB and MSB pages. The top and bottom two layers operate exclusively in the SLC mode. The remaining layers contain shared memory pages. MSB pages are usually more
Figur 3.5: (a) Proposed intra-block wear-leveling algorithm. (b) Comparison of worst-case page RBER using traditional PE cycling and algorithm-based PE cycling. Symbols are from experimental evaluations, and the solid lines are fitted curves.

equero than LSB pages as they involve high $V_t$ cells, which are more affected by retention loss. Thus, MSB pages of the bottom layer ($L_2$) will have the highest RBER for a given PE cycle condition. Thus, the endurance of a memory block will be limited by the MSB pages of $L_2$, which will fail ECC after a certain number of PE cycles. Thus, we propose to keep using the memory block by skipping the programming of the MSB pages of $L_2$ after $PE = n_1$. This parameter is determined by measuring the RBER; when the RBER reaches a certain fraction of the maximum correctable RBER. Similarly, after $n_2$ PE cycles, we need to skip the programming of MSB pages of the next layer from the bottom ($L_3$). In implementing the algorithm, one may skip a group of layers instead of just one to keep a safety margin for data integrity. Similarly, if the top layer also shows very high RBER, which may be the case for specific chips, we need to skip MSB pages of top layers as well, as shown in Figure 3.5(a). A more sophisticated algorithm can involve a series of PE cycles (e.g., $n_2, n_3, \ldots$) that trigger the logical reshaping of the Flash memory block.
The critical design parameters for the implementation of the proposed algorithm are the threshold PE cycle counts \( n_1, n_2, n_3 \), as well as the number and location of pages to be skipped/excluded at each step. These parameters need to be pre-determined for each family of chips that use a given technology process through a detailed characterization. This characterization is necessary because layer-dependent RBER may vary among chips from different families and/or manufacturers. It will be conducted once by either the chip manufacturer or a system integrator. The Flash memory controller is in charge of implementing logical reshaping of Flash memory blocks, taking into account these parameters and the current state of each Flash block.

In order to illustrate the improvement in endurance achieved by the proposed algorithm, we perform the following experiment. A Flash memory block is exposed to traditional block management. All pages of the block are stressed by repeated PE cycling. Once the RBER in any page of the block reaches the threshold level set to 0.2% in this experiment, the block is marked as bad, and the corresponding PE cycle count is reported.

Another Flash memory block is managed by a simplified version of the proposed algorithm. All pages of the block are repeatedly stressed until we reach \( n_1 = 5k \) PE cycles. Please note that the worst-case RBER after 5k PE cycles is only 0.1%, which is 50% below the threshold RBER level of 0.2%. We made this choice of \( n_1 = 5k \) in order to keep an extra RBER margin to account for the block-to-block or chip-to-chip variation. At this point, the Flash memory block is reshaped, and the MSB pages in a group of bottom layers (\( L_2 \) to \( L_6 \)) are excluded.
for further use. The next logical reshaping of the Flash memory block takes place when the number of PE cycles reaches $PE = n_2 = 10k$. At this moment, the MSB pages residing in a group of top layers ($L_{29}$ to $L_{31}$) are excluded from further use. The remaining pages are further stressed until the total number of PE cycles reaches $PE = 20k$. At this moment, the experiment is concluded.

Figure 3.5(b) shows the results of the experimental evaluation. We plot the worst-case page RBER from a memory block as a function of PE cycle count for the traditional case (blue line) and for the simplified block management described above (red line). Using traditional block management, the threshold RBER of 0.2% is reached when $PE = 11k$. The proposed simplified 2-step reshaping block management reaches the threshold RBER when $PE = 16k$. With the proposed implementation, we find that the maximum PE cycle of a block can be enhanced by $\sim 45\%$ (from 11k to 16k). The trade-off for this endurance enhancement is a reduced block size (12.5% reduction in the current implementation) at its end of life. The improvement of endurance will be significantly higher if the algorithm is implemented with finer granularity or with more than two reshaping steps.

3.5 Conclusions

We observe that pages within a 3D NAND Flash block show a large variation in their endurance. These variations are correlated to the geometrical location of the pages within a vertical 3D structure. We show that the endurance of memory pages at the top and bottom layers is significantly lower than that of the pages in the middle layers. Our findings suggest that more sophisticated block
management schemes can be developed that will take into account layer-to-layer wear-out differences within a Flash memory block. Such management schemes will improve the endurance and utilization of storage media. The next chapter discusses how these variations can be exploited for an energy-efficient storage system.
Chapter 4. EXPRESS: Exploiting Energy-Accuracy Trade-offs in 3D NAND Flash Memory for Energy-Efficient Storage

4.1 Introduction

Non-volatile NAND Flash memories are the basic building blocks of data storage components found in a range of systems, from IoT and edge-computing platforms, wearable electronics, smartphones, self-driving cars, and drones to solid-state drives (SSDs) used in personal computers and cloud computing infrastructures [86]. Energy efficiency is a key requirement for data storage components used in emerging edge computing devices, as most of them are constrained by limited power sources [87, 88, 89]. Designers of modern Flash storage systems, such as SSDs, focus exclusively on long-term data integrity rather than on energy efficiency. In the light of many emerging approximate AI-based edge computing applications, e.g., machine learning, data analytics, vision, object classification, and others as described in [90, 91, 92, 93], where approximate and short-lived data are very common, new opportunities arise for developing energy-efficient approximate storage systems.
A typical Flash memory-based storage system consists of two discrete components: Flash storage media with one or more Flash memory devices and a Flash memory controller. Often the controller and the Flash memory devices are made by different companies, and system integrators integrate these components to design storage solutions tailored for specific applications. Flash memory manufacturers comply with a chip-interfacing specification defined by the Open NAND Flash Interface (ONFI) working group [94]. This specification offers a few application-agnostic storage functions that are not tailored toward energy-efficient approximate storage applications. Thus, there remain several opportunities for the system integrators to design energy-efficient storage systems by utilizing trade-offs between data accuracy and energy efficiency that are inherent to the NAND Flash memory technology.

Even though NAND Flash memory-based storage solutions require less power than other non-volatile storage solutions, e.g., hard disk drives, they still account for a significant portion of the total energy expenditures of computing systems [95]. Several recent research proposals have emphasized the prospect of approximate storage for achieving high energy efficiency in emerging edge-computing applications [95, 96, 97, 98, 99, 100, 101, 102]. To curb energy consumption of Flash memories in ultra-low-power microcontrollers, Salajegheh et al. [102] propose an energy-saving technique utilizing lower operating voltages that jeopardize correct memory operations. To remedy the possible loss of information, they employ (a) repeated in-place write operations, (b) multiple places write operations, or (c) RS-Berger coding of data. They report energy savings for
in-place write operations of up to 34% on lower-end microcontrollers. Similarly, a study by Tseng et al. [96] shows that up to 45% of energy consumed could be saved using dynamic voltage scaling based on Flash operations being performed. Sampson et al. [98] propose an approximate storage technique in solid-state memories by relaxing the threshold voltage margins between different memory states during write operations by using varying program pulse widths. Through detailed simulation, they showed that their proposal would make memory write 1.7 times faster. However, implementation of their method on the common-off-the-shelf (COTS) NAND chips requires privileged commands that are not available in the ONFI command set. Li et al. [100] propose to leverage approximate data in the NAND Flash memory to improve read performance and enhance the reliability of regular data. Papirla et al. [103] find that the energy required by Flash write operations heavily depends on data patterns. Thus, they propose an encoding scheme that minimizes the frequency of power-hungry bit patterns in codewords (‘10’ and ‘01’), reducing the total energy of Flash write operations by up to 34%. Nath et al. [104] propose a lazy amnesic compression-based technique for storing data in Flash memories. The required energy for Flash write operations is reduced by using lossy compression; the compression ratio is adjusted based on the age of data. Mathur et al. introduce Capsule [95], a log-structured object storage system for Flash memories that supports fine-grain allocation of space for storage objects such as streams, files, arrays, queues. Poudel and Milenkovic [97] introduce a technique that reduces time and energy consumed by critical Flash
operations in embedded NOR memories by introducing partial or aborted Flash operations.

Though these techniques demonstrate significant potential in reducing the total energy consumed, they often introduce extra overhead in time, compute resources, and/or memory space [95, 102]. Next, they usually consider lower-density Flash memories, e.g., NOR Flash memories used in low-end embedded systems [97, 102]. Although using partial write operations is suggested in Sampson et al. [98] as a way to increase energy efficiency of SSDs, its effectiveness is evaluated using a simulation-based environment only, without taking into account the physical properties of COTS Flash memory chips. Consequently, the effectiveness of this approach on COTS Flash memory chips remains unknown. In addition, we are not aware of any study considering the now dominant three-dimensional (3D) NAND Flash memory technology and the unique challenges it presents. For example, timing and data integrity parameters are often layer-dependent. Thus, we believe there is a need to explore the energy efficiency of the now dominant 3D NAND Flash memories and experimentally evaluate the effectiveness of techniques for improving their energy efficiency.

The complex organization of NAND Flash memories and their physical properties demand disproportionate latency and energy expenditures to ensure high data integrity when writing data into Flash memories. This chapter experimentally explores this disproportionality on the state-of-the-art commercial 3D NAND Flash memories and introduces EXPRESS – a technique for increasing energy efficiency of Flash memory writes. EXPRESS utilizes partial program
operations, thus exploiting this disproportionality between latency and energy expenditures on one side and data accuracy on the other side. The proposed method can be implemented in the storage controller’s firmware without requiring any privileged Flash operations or changes in the system design. An experimental evaluation shows that EXPRESS reduces energy expenditures by 20%-50% relative to the traditional Flash writes, at the cost of a minimal loss in data integrity (<1%). In addition, the chapter experimentally explores the impact of page-to-page variability and program-erase cycling on the implementation of EXPRESS and offers strategies to cope with those undesired effects. Compared to the existing techniques, EXPRESS offers the following advantages: (a) it can be applied to both 2D and 3D Flash memories, (b) it does not require any privileged operations, (c) it can be combined with and is orthogonal to other techniques (e.g., voltage scaling), and (d) it does not require any data pre-processing or special data encoding. Table 4.1 gives a comparative analysis of the major characteristics of the previously proposed related techniques and EXPRESS. The following are the key contributions of the chapter.

- We explore and quantify disproportionate trade-offs between data accuracy and energy efficiency of Flash memory program operations using COTS 3D NAND Flash memory chips. We find that more than 20% of the energy and time is spent on improving less than 1% of bit accuracy during memory write operations. We shed more light on this phenomenon and identify slow memory cells belonging to tails of state distributions, a main reason for disproportionate energy-accuracy trade-offs.
Table 4.1: Comparison of EXPRESS method with existing works.

<table>
<thead>
<tr>
<th>Proposed Methods</th>
<th>Energy saved [%]</th>
<th>Bit Error Rate Range [%]</th>
<th>Methodology</th>
<th>Layer variation consideration</th>
<th>Applicable to COTS chips</th>
</tr>
</thead>
<tbody>
<tr>
<td>Half-wits</td>
<td>&lt;34</td>
<td>5 to 11</td>
<td>Experiment</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Dynamic scaling</td>
<td>&lt;45</td>
<td>2 to 19</td>
<td>Experiment</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Data encoding</td>
<td>&lt;33</td>
<td>Not reported</td>
<td>Simulation</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Approximate Storage</td>
<td>&lt;40</td>
<td>2 to 4</td>
<td>Simulation</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>EXPRESS</td>
<td>&lt;52</td>
<td>&lt;1%</td>
<td>Experiment</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

- We propose a novel technique called EXPRESS that utilizes partial write operations to increase energy efficiency at a minimal loss of accuracy. We characterize NAND Flash operations and experimentally explore energy-accuracy trade-offs as a function of the partial program time. Based on the results of the experimental evaluation, we propose an algorithm to choose the partial program time that strikes an optimal balance between energy efficiency and data accuracy.

- We perform a detailed characterization of page-to-page variability, program-erase cycling effects, and data retention effects on the effectiveness of EXPRESS. We propose several countermeasures that can be adopted to properly address these variability and reliability issues.

The rest of the chapter is organized as follows. Section 4.2 gives background on the Flash memory interfacing from the host controller. Section 4.3 introduces the proposed technique. Section 4.4 explores the effectiveness of the proposed technique when applied to 3D Flash memories operating in SLC (Single-
Level Cell) and MLC (Multi-Level Cell) modes. Section 4.4 also discusses challenges due to page-to-page variability, program-erase cycling, and data retention issues and offers enhancements to EXPRESS to address these challenges. Section 4.6 gives a summary of the chapter.

4.2 Interfacing NAND Chip from the Host Controller

COTS Flash memory chips use a standardized low-level interface developed by the Open NAND Flash Interface (ONFI) working group, a consortium of Flash memory manufacturers [94]. The ONFI specifications define standard physical interfaces, chip identification mechanisms, a standard command set for reading, writing, and erasing NAND Flash; timing requirements; and data integrity features. Depending on the chip package and type of interface, the number of bytes sent to or received from a device at a time can vary. In our case, both commands and data are carried through 8 data lines ($DQ_0 – DQ_7$). Control lines CE# (Chip Enable, active low), CLE (Command Latch Enable), ALE (Address Latch Enable), RE# (Read Enable, active low), and WE# (Write Enable, active low) allow for functional and timing control of the interface.

As shown in Figure 4.1, a command placed on the data lines by the host is written into the device’s command register on the rising edge of WE#, when CE# is low, ALE is low, CLE is high, and RE# is high. An address placed on the data lines by the host is written to the device’s address register on the rising edge of WE#, when CE# is low, ALE is high, CLE is low, and RE# is high. Data placed on the data lines by the host is written into the device’s data
Figure 4.1: Timing diagram for a sequence of steps carried out by the host during page program operation in asynchronous interface.

register on the rising edge of WE# when CE# is low, ALE is low, CLE is low, and RE# is high. Data is output from the device if it is in a ready state. The data from the device’s data register is output to the data lines on the falling edge of RE# when CE# is low, ALE is low, CLE# is low, and WE# is high. Figure 4.1 illustrates a sequence of commands that carry out a page program operation. The operation is initiated by the host that sends the command 0x80 to the device through the data lines. Next, the host writes five address cycles (A0-A4) while keeping the ALE signal high. Next, the host controller sends data to be written to the device’s data register, byte by byte. Finally, the host sends the PAGE PROGRAM command (0x10) that initiates the write operation into the specified page of the Flash memory array. During the page program operation, the device’s status control pin RB (Ready/Busy#) is low, indicating that the device is currently busy. Upon completion of the program operation, the RB
signal is set high. Thus, the host can determine the page program time \( t_{\text{prog}} \) by monitoring the state of this pin after issuing the command sequence.

### 4.3 Proposed Technique - EXPRESS

The EXPRESS technique reduces the energy consumed during Flash program operations at the cost of a negligible loss of accuracy. It relies on a partial page program operation to counter the disproportionate energy-accuracy trade-off inherent in the ISPP scheme. Figure 4.2 (a) illustrates the proposed EXPRESS technique. The solid black line represents the status of the RB pin during a regular page program operation. This pin goes low, indicating that the NAND array is busy for the duration of the program operation, \( t_{\text{prog}} \). The \( t_{\text{prog}} \) value lies in the range of 300-600 \( \mu \text{s} \) for an SLC memory page of the chip used in this study. The program operation, however, can be terminated prematurely using a RESET command like program suspend operation [105]. The state of the RB pin, in this

---

**Figure 4.2:** Nominal (solid black line) and partial page program (dashed red line) operations. (a) Timing diagrams. (b) The threshold voltage distributions for the erased and programmed states and corresponding reference voltages.
case, is illustrated with the red dashed line. The premature termination of the program operation results in a partial program operation. Although this operation may slightly increase the bit error rate (BER), it can significantly reduce the time and energy of page program operations. The critical parameter that enables exploring trade-offs between energy and accuracy is the partial program time, $t_{pp}$. Assuming that all these times remain constant across all the program cycles, the total page program time can be expressed as follows: (details about program cycle and ISPP are described in chapter 2)

$$t_{prog} = t_{init} + n \times (t_{su}^i + t_p + t_{su}^v + t_{vfy}) = t_{init} + n \times t_{pcy}. \quad (4.1)$$

Here $t_{pcy} = (t_{su}^i + t_p + t_{su}^v + t_{vfy})$ represents the time required for one full program cycle, $n$ stands for the total number of program cycles required for the page program operation, and $t_{init}$ is the initial time required by the NAND array to verify the page status before applying a series of program cycles. Please note that Equation 4.1 captures the common features of NAND page program operations. However, it may need to be adjusted depending on a specific implementation of on-chip control logic in a particular Flash memory chip.

The critical parameter that enables exploring trade-offs between energy and accuracy is the partial program time, $t_{pp}$. The following equation can be used to estimate $t_{pp}$:

$$t_{pp} = t_{init} + (n - n_{skip} - 1) \times t_{pcy} + (t_{su}^m + t_p) = t_{prog} - (n_{skip} + 1) \times t_{pcy} + (t_{su}^m + t_p). \quad (4.2)$$
Here $n_{\text{skip}}$ is the number of program cycles that can be skipped to achieve higher energy efficiency. Note that we have not included the verify phase of the last program cycle in Eq. 4.2 as no additional bits get programmed during a verify phase. In general, Eq. 4.2 can be used as a guideline for finding an optimal $t_{\text{pp}}$, and it needs to be pre-characterized based on the properties of the particular family of Flash memory chips.

Figure 4.2(b) sheds more light on the rationale behind EXPRESS by illustrating three different reference voltages that correspond to three different memory operations. The erase operation ensures that threshold voltages of all erased cells in the block are below the reference voltage $V^E_{\text{ref}}$. Similarly, $V^P_{\text{ref}}$ is the reference voltage used during a program verify phase of the page program operation. The ISPP scheme ensures that the threshold voltages of all programmed cells are above the reference voltage $V^P_{\text{ref}}$. Finally, a read reference voltage $V^R_{\text{ref}}$ is used to distinguish between the cell’s erase and program states during a page read operation. All NAND manufacturers keep a sufficient voltage margin between the read and program-verify voltages to minimize read errors. However, this margin can be exploited to increase energy efficiency in all applications where the BER is sufficiently low and can be corrected using error-correction techniques. In addition, EXPRESS can be used even when a somewhat higher BER can be tolerated, e.g., in applications where approximate, short-lived data are common. For example, if we terminate the program operation prematurely, the resulting threshold voltage distribution will be mostly above the read reference voltage as shown with dashed lines in Figure 4.2(b). The resulting distribution may have
some area below the read reference voltage, and that will create errors which we are trading off for the saved energy.

Since 3D NAND Flash memory cells in the erased state exhibit long tails of the threshold voltage distribution, programming those cells may require several extra program pulses. Since left tail cells usually represent less than 1% of the total page size, a premature termination of the program operation may cause that just 1% of cells have their threshold voltages below $V_{\text{ref}}^{PVY}$. Interestingly, not all of these tail bits will show up as error bits during a read operation as there is a sufficient voltage margin between the read and the program verify voltages. Thus, one can improve the energy efficiency of Flash memory program operations with very little or no sacrifice in the bit accuracy if the partial program time is chosen appropriately. However, such partial programming may lead to increased retention loss due to reduced reliability margin. The following section presents the experimental evaluation of the energy-accuracy trade-offs in the state-of-the-art 3D NAND Flash memory. It formulates guidelines for choosing the appropriate partial program time. In our experimental evaluation, we use a 3D NAND Flash memory chip that supports both SLC and MLC modes of operation.

Whereas EXPRESS promises energy savings at a negligible loss of accuracy, it is important to address practical issues that can impact the efficacy of the proposed technique, including page-to-page variability, wear-out of gate oxides, and data retention. Hence, the experimental evaluations also discuss the effects of page-to-page variability, PE cycling, and data retention on EXPRESS.
4.4 Experimental Evaluation

4.4.1 Evaluation of the Proposed Technique on SLC Memory

We first validate EXPRESS by configuring a NAND chip to operate in the SLC mode. An all-zero data pattern is written using partial page program operations while varying the partial program time, $t_{pp}$. Later in this section, we perform a similar experiment with a random data pattern with equal distribution among all available Flash cell states. Figure 4.3(a) shows the percentage of the programmed bits as a function of the partial program time. Each point in the plot represents the percentage of programmed bits collected from 10 experiments on the same page. Each partial program experiment is proceeded by a full block erase operation. Figure 4.3(b) shows the current drawn by the NAND chip during a regular page program operation. The corresponding status of the RB pin during a regular page program operation is illustrated by a red dashed line. The current drawn increases notably during the program operation relative to the current drawn in the device’s idle state. The current waveform reveals two distinct profiles that are repeated alternatively. We hypothesize that these characteristic current profiles correspond to the program (blue shaded regions) and verify (red shaded regions) phases of the page program operation and its ISPP scheme. The plot in Figure 4.3(a) shows that the percentage of programmed bits resembles a step function. The Flash memory cells are programmed only during program pulses. The transition points of the percentage of programmed bits align with the program pulse phases in Figure 4.3(b). Furthermore, the percentage of
programmed bits remains constant during the verification phases. This confirms our hypothesis that the ISPP scheme is used in a page program operation and that characteristic waveforms correspond to the program pulses and verify phases of the page program operation.

**Figure 4.3:** Effects of partial page program operation. (a) Percentage of programmed bits in an SLC memory page as a function of partial page program time, \( t_{pp} \). (b) The current is drawn by the memory chip during a regular page program operation.

Further, the results from Figure 6 support the following two observations. Figure 4.3(a) shows that just three program cycles out of five used in a regular program operation are sufficient to achieve the bit accuracy above 99.9%. The last two program pulses are mainly used to program a tiny fraction of bits located in the lower tail of the erase \( V_t \) distribution, as illustrated in the inset of Figure 4.3(a).
Figure 4.3(b) illustrates that there is periodicity in terms of the program and verify cycles and that all program pulses and verify phases have the similar duration and current profiles. Thus, Eq. 4.2 can be used for determining a suitable partial program time. As there is no tangible advantage in terminating the program operation in the middle of a verify or a program cycle, the optimal $t_{pp}$ should correspond to the end of a program pulse. The number of program pulses required to achieve the desired bit accuracy may be specific for a family of chips, the location of the page in the 3D structure, and its usage conditions. Still, all these can be pre-characterized and then used to inform a proper implementation of the partial program operations.

4.4.2 Evaluation of the Proposed Technique on MLC Memory

MLC Flash memory cells store 2 bits of information; hence, two different types of logical pages share a single word line. These two bits correspond to 4 states of Flash memory cells, i.e., the information is encoded in the form of 4 threshold voltage distributions (Er-11, A-01, B-00, C-10) as illustrated in Figure 4.4. The most significant bit (MSB) of the logic states of all the memory cells connected to a given word line forms the MSB page. Similarly, the least significant bit (LSB) of the logic states of the memory cells from the same word line forms the logical LSB page. The LSB page programming involves raising the erase state $V_t$ of certain cells to B-state as shown in Figure 4.4.

The MSB page programming is performed after the LSB page programming is finished. During MSB page programming, certain memory cells from the
Er state go to the A state, and certain cells from the B state go to the C state, as shown in Figure 4.4. Two read reference voltages are used to read MSB page data, whereas only one read reference voltage is needed for reading LSB page data.

Figure 4.4: $V_t$ distribution for four states in MLC mode. The top distributions show the $V_t$ states after LSB page program and the bottom plot shows the complete distribution after MSB page program.

Figure 4.5(a) shows the percentage of programmed bits as a function of $t_{pp}$ for MSB and LSB pages in red and blue solid lines, respectively. The experiments
Figure 4.5: Percentage of programmed bits for a page in MLC mode as a function of partial page program time, $t_{pp}$. Measured current drawn during regular page program operations in MLC mode for (b) LSB page. (c) MSB page.

are conducted as follows. Logical LSB and MSB pages are used from a freshly erased block. First, we partially program an LSB page and then the corresponding MSB page with an all-zero data pattern. Please note that the chip used in this study, when configured in MLC mode, by default implements data scrambling that
ensures that all four states are uniformly utilized in a physical page, regardless of the input data pattern. Thus, writing all zeros in LSB and MSB pages does not imply that all cells are in the B state. Therefore, the data pattern does not impact the results of our experiments. After the partial program operation, we perform page read operation for both LSB and MSB pages and determine the percentage of programmed bits for each experiment. The programmed bit percentage for the LSB pages looks quite similar to the one observed for the SLC mode of operation. Since writing on an LSB page involves only one programmed $V_t$ state (B state), its ISPP scheme is quite similar to the one used in the SLC mode.

The programmed bit percentage for MSB pages, however, has distinctively different characteristics. There are two plateaus because two different $V_t$ states, A and C, are formed during MSB programming. The first plateau corresponds to the construction of the A-state as it has a lower $V_t$ and is thus formed first. The second plateau corresponds to the formation of the C state. The time to complete an MSB page program operation is significantly longer than the time needed to program the corresponding LSB page. As programming of an MSB page involves transitioning Flash cells from Er to A and from B to C states, it thus requires more ISPP cycles and consequently more time to complete a program operation relative to its LSB counterpart. Another distinctive feature of the MSB page programming is its verification phases, which are more complex than the LSB counterparts. An LSB page verification requires only one read to verify that the cell $V_t$ exceed the lower bound of B state ($V_{LSB}^{REF}$) whereas an MSB page verification requires two reads to check the lower bounds of both the A and C states. These
hypotheses are confirmed by inspecting current profiles as discussed in the text below.

Figure 4.5(b) and 4.5(c) show the current drawn by the chip during a page program operation for an LSB page and MSB page, respectively. Similar to the SLC current profiles, we observe the periodic program pulses and verify phases in the current waveform. For example, the LSB page analyzed in Figure 4.5(b) requires 9 ISSP cycles with the total program time $t_{\text{prog}}^{\text{LSB}} \approx 1000\mu s$. However, the bit accuracy reaches above 99% with only 7 program pulses ($t_{\text{pp}}^{\text{LSB}} \approx 750\mu s$), indicating 25% energy saving with less than 1% bit accuracy loss. Programming MSB pages generally require more time than programming LSB pages. For example, the MSB page analyzed in Figure 4.5(c) requires $t_{\text{prog}}^{\text{MSB}} \approx 1500\mu s$ or 11 ISSP cycles. In addition, the verification phases in the case of MSB program operations take more time than those that take place during LSB program operations. Still, we find that partial program operations can be utilized on MSB pages, offering more than 20% in energy savings with a negligible (less than 1%) bit-accuracy loss. The optimal partial program time for MSB pages is $t_{\text{pp}}^{\text{MSB}} \approx 1150\mu s$.

We observe a considerable page-to-page variability in the bit accuracy (error bar in Figure 4.5(a)) even though the $t_{\text{pp}}$ was fixed. Such a page-to-page variability may arise in the NAND memory due to inherent process variations, physical organization, and the presence of program and read noise. In the next section, we elaborate further on the page-to-page timing variability and possible countermeasures.
4.4.3 Effects of Page-to-page Variability

3D NAND Flash memories exhibit page-to-page variations due to the unique nature of the array geometry and the intrinsic process variations within the array. Figure 4.6(a) shows the organization of a 3D NAND memory block configured in the SLC mode. The pages in a block are organized in rows that correspond to physical vertical layers \((L_0, L_1, \ldots, L_{N-1})\) and columns that correspond to sub-blocks \((0, 1, \ldots, M - 1)\). A page number within a block can be expressed as \(P_{L_j}^i\), where \(L_j\) represents the layer number and \(i = 0, 1, \ldots, M - 1\). The chip under test has 32 layers \((N = 32)\), where each layer contains 16 logical pages \((M = 16)\) of a given memory block. Thus, there are a total of \(16 \times 32 = 512\) pages within a block. We performed characterization of page program times by sequentially programming all the pages of a memory block using a random data pattern. Our characterization results are shown as a cumulative distribution plot in Figure 4.6(b). The results indicate that the standard page program time varies significantly among different pages within the same block. Since the implementation of EXPRESS requires an estimation of \(t_{pp}\) based on the nominal page program time \((t_{prog})\), it needs to be adapted to account for the page-to-page variability.

To understand the precise nature of page-to-page variability, we measure each page’s nominal page program time in a block in SLC mode. Figure 4.7 shows the results of these measurements. We can make the following two observations from these results:
Figure 4.6: (a) Organization of a 3D NAND memory block. (b) Cumulative distribution of the measured nominal page program time for SLC memory pages of a given block.

1. The first page to be programmed in a given layer takes more time to complete a program operation. We classify these pages as slow pages, shown in blue in Figure 4.7.

2. The $t_{\text{prog}}$ variability is minimal among memory pages located in the same vertical layer ($P_{L_j}^{L_j}$ to $P_{15}^{L_j}$) of the array. Consequently, we argue that the memory controller can learn the $t_{\text{prog}}$ value from the page $P_{L_j}^{L_j}$ (referred to as a learning page) and then apply EXPRESS when programming the remaining pages.

   To further illustrate the variability, we compute the median $t_{\text{prog}}$ as the last column in Figure 4.7. We find that the median $t_{\text{prog}}$ varies between different layers, but within the same layer, $t_{\text{prog}}$ remains relatively unchanged (except for the first page of a layer). We exploit this observation and propose an adaptive learning algorithm to maximize energy saving for the EXPRESS method.
To address the observed variabilities, we propose the following modification to EXPRESS. The nominal program time variation among slow pages (marked as blue boxes in Figure 4.7) is minimal. Consequently, the Flash controller may apply EXPRESS on the slow pages by learning the corresponding \( t_{\text{prog}} \) from the first page of the block \( (P_{0}^{L_{0}}) \). The remaining pages of the block are classified as learning pages (yellow) and EXPRESS pages (green). Nominal page program operations are performed on the learning pages to acquire the exact \( t_{\text{prog}} \) value, and EXPRESS is applied on the remaining pages of the layer \( (P_{2}^{L_{j}} \text{ to } P_{15}^{L_{j}}) \) by estimating the corresponding \( t_{\text{pp}} \) using Eq.4.2.

Next, we discuss the page-to-page variability for a Flash block configured in the MLC mode. Figure 4.8(a) shows the cumulative distribution of \( t_{\text{prog}} \) for LSB and MSB pages. We find significant page-to-page variability for both LSB (blue line) and MSB (red line) pages. LSB pages behave similarly to the SLC
Figure 4.8: (a) Cumulative distribution of program time for LSB (blue) and MSB (red) pages in a block in fresh condition. (b) Express algorithm for MSB pages. The numbers represent the nominal $t_{prog}$ values in $\mu s$ corresponding to the page location.

pages, where the first LSB page of a given layer requires significantly longer $t_{prog}$ compared to the other LSB pages in the same layer. These slow LSB pages constitute the upper tail (10%) of the cumulative distribution in Figure 4.8(a). The average $t_{prog}$ for the MSB pages is distinctively higher than the average $t_{prog}$ for LSB pages. Unlike LSB pages, $t_{prog}$ variability for MSB pages is relatively small. The first MSB pages in a layer do not require higher $t_{prog}$ than other MSB pages in the given layer.

Since LSB pages behave similarly to SLC pages, the algorithm for implementing EXPRESS on LSB pages can mirror the algorithm proposed for the SLC pages as described above. For the MSB pages, a slight modification of the algorithm is introduced by treating the first MSB page ($P_{0}^{L_j}$) in a given layer as the learning page. Figure 4.8(b) shows the EXPRESS algorithm for MSB pages where $t_{prog}$ is learned from the first MSB page of a given layer $P_{0}^{L_j}$. Eq. 4.2 is
used to estimate the $t_{pp}$ from the corresponding $t_{prog}$; that value is applied to the remaining $P_{n}^{L_{j}} (n = 1, 3, ..., 15)$ pages.

The adaptive learning algorithm for EXPRESS widens an opportunity window for performance and energy enhancement. Table 4.2 summarizes the measured $t_{prog}$ (or nominal program time of a page) and the corresponding optimal $t_{pp}$ for pages in both the SLC and MLC configurations. The table also quantifies the effectiveness of EXPRESS by reporting the bit error rate and the average percentage of energy saved. The results are broken down based on page types, as discussed above. We calculate the number of program loops that can be skipped for EXPRESS to acquire acceptable accuracy loss ($< 1\%$) for each page type. We find an optimal value of the parameter $n_{\text{skip}}$ in Eq. 4.2; $n_{\text{skip}} = 1$ or 2 for SLC pages, depending on their type, and $n_{\text{skip}} = 2$ for MLC pages. For higher values of $n_{\text{skip}}$, the BER in the written data is found to be more than 1%. However, $n_{\text{skip}}$ needs to be pre-characterized for each class of chips for optimal EXPRESS implementation. Note that the table is prepared based on data collected from 1024 pages of an MLC Flash block and 512 pages of an SLC Flash block. We find that EXPRESS can save an average of 20% to 50% of write energy depending on the page type. Whereas the exact figure for energy savings may differ for Flash memory chips with a different organization or are manufactured in different technology nodes, the proposed technique applies to all of them because it exploits accuracy-energy disproportionality common for all modern Flash memory chips.
<table>
<thead>
<tr>
<th>Storage mode</th>
<th>Page type</th>
<th>$t_{prog}$ ($\mu$s)</th>
<th>n</th>
<th>$t_{pp}$</th>
<th>$n_{skip}$</th>
<th>Average BER [%]</th>
<th>Average energy saved [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLC</td>
<td>Slow</td>
<td>560</td>
<td>5</td>
<td>300</td>
<td>2</td>
<td>$5 \times 10^{-1}$</td>
<td>46.48</td>
</tr>
<tr>
<td></td>
<td>Nominal</td>
<td>320</td>
<td>3</td>
<td>220</td>
<td>1</td>
<td>0</td>
<td>31.43</td>
</tr>
<tr>
<td>LSB</td>
<td>Slow</td>
<td>1022</td>
<td>9</td>
<td>760</td>
<td>2</td>
<td>$5 \times 10^{-2}$</td>
<td>25.67</td>
</tr>
<tr>
<td></td>
<td>Nominal</td>
<td>460</td>
<td>4</td>
<td>220</td>
<td>2</td>
<td>$8 \times 10^{-2}$</td>
<td>52.15</td>
</tr>
<tr>
<td>MSB</td>
<td>Nominal</td>
<td>1470</td>
<td>11</td>
<td>1150</td>
<td>2</td>
<td>$5 \times 10^{-1}$</td>
<td>21.76</td>
</tr>
</tbody>
</table>

**4.4.4 Effects of Program Erase Cycling on EXPRESS**

NAND Flash memory exhibits limited endurance, typically specified by the maximum number of program-erase operations (or PE cycles) allowed on a memory block. The number of PE cycles may impact nominal page program time, $t_{prog}$; stressed pages with a high number of PE cycles may take more time to program. Hence, an implementation of EXPRESS needs to consider the number of PE cycles. Figure 4.9(a) shows the cumulative distribution of the nominal page program time for SLC pages in a fresh Flash memory block and a memory block that has been exposed to 10,000 PE cycles. Similarly, Figure 4.9(b) shows the cumulative distribution of the nominal page program times for the LSB and MSB pages in the MLC mode for a fresh block and a block exposed to 5,000
PE cycles. We find that the average $t_{\text{prog}}$ increases with PE cycling in the MLC mode, whereas a minimal change is observed in the SLC mode.

**Figure 4.9:** (a) Cumulative distribution of page program time for SLC pages in fresh (solid blue line) and 10K PE cycles (dashed blue line) condition. (b) Cumulative distribution of page program time for LSB pages in fresh (solid blue line) and 5K PE cycles (blue dashed line) condition and MSB pages in fresh (solid red line) and 5K PE cycles (dashed red line) condition.

Even though the average $t_{\text{prog}}$ increases with an increase in the number of PE cycles in the MLC mode, the intra-layer and inter-layer $t_{\text{prog}}$ variations remain unchanged relative to the fresh memory blocks. Specifically, our observations (a) and (b) of section 4.4.3 remain true even on stressed memory blocks. Therefore, the algorithm proposed in section 4.4.3 can be used unchanged because EXPRESS learns the $t_{\text{prog}}$ from the learning page, regardless of PE cycles.

Table 4.3 summarizes the updated $t_{\text{prog}}$ and the corresponding $t_{\text{pp}}$ on PE-cycled memory blocks. We find that for 10K PE cycles in an SLC memory
block, the optimal value for $n_{\text{skip}}$ is 1. Higher $n_{\text{skip}}$ values cause a very high bit error rate (BER) in the written data. With $n_{\text{skip}} = 1$ in the SLC mode, we find that EXPRESS saves approximately 30\% of write energy for nominal SLC pages. Similarly, in the MLC mode operation, we find that the optimal $n_{\text{skip}}$ is 2, which ensures a BER of less than 1\%. Thus, the energy savings are found to be approximately 16\% for MSB pages and approximately 46\% for LSB pages. Since the $t_{\text{prog}}$ values for MSB pages are longer compared to the LSB pages, the percentage of energy savings is lower for MSB pages for the same $n_{\text{skip}}$ value.

Table 4.3: EXPRESS characterization for NAND Flash block after PE cycles.

<table>
<thead>
<tr>
<th>Storage mode</th>
<th>Page type</th>
<th>$t_{\text{prog}}$ ($\mu$s)</th>
<th>$n$</th>
<th>$t_{pp}$</th>
<th>$n_{\text{skip}}$</th>
<th>Average BER [%]</th>
<th>Average energy saved [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLC-10K</td>
<td>Slow</td>
<td>467</td>
<td>4</td>
<td>300</td>
<td>1</td>
<td>$9 \times 10^{-4}$</td>
<td>35.78</td>
</tr>
<tr>
<td></td>
<td>Nominal</td>
<td>320</td>
<td>3</td>
<td>220</td>
<td>1</td>
<td>$5 \times 10^{-5}$</td>
<td>31.29</td>
</tr>
<tr>
<td>LSB-5K</td>
<td>Slow</td>
<td>1001</td>
<td>9</td>
<td>760</td>
<td>2</td>
<td>$2 \times 10^{-1}$</td>
<td>24.12</td>
</tr>
<tr>
<td></td>
<td>Nominal</td>
<td>561</td>
<td>5</td>
<td>300</td>
<td>2</td>
<td>$1 \times 10^{-1}$</td>
<td>46.56</td>
</tr>
<tr>
<td>MSB</td>
<td>Nominal</td>
<td>1600</td>
<td>12</td>
<td>1330</td>
<td>2</td>
<td>$5 \times 10^{-1}$</td>
<td>16.87</td>
</tr>
</tbody>
</table>
4.4.5 Data Retention Effects

Data retention is an essential consideration for non-volatile Flash memories. The charge stored on the FG/CT of Flash cells tends to leak out through the tunnel oxides at room temperature, lowering the cell threshold voltage over a period of time [106, 107, 108]. Hence, Flash memory manufacturers keep wider voltage margins between the program $V_t$ and the read reference voltage to guarantee long-term data retention (approximately 10 years for many products). Since EXPRESS trades off the voltage margin for improving energy efficiency, it is important to characterize data retention time.

Figure 4.10: BER as a function of retention loss for traditional programming (blue) and proposed partial programming (red) for (a) SLC page. (b) LSB page. (c) MSB pages.
Figure 4.10 summarizes the results of an experiment that explores the effects of EXPRESS on data retention for both the SLC and MLC modes of operation on PE-cycled blocks. It shows the bit error rate (BER) of data written by EXPRESS (red bars) and data written by the nominal program operation (blue bars). To accelerate the retention loss, we bake the chip at a higher temperature (120°C) for 1, 2, or 3 hours. Using the acceleration factor-based calculation, we find that the 3 hours of baking time corresponds to 5 years at room temperature, assuming activation energy for charge loss in 3D NAND as $E_A = 1\text{ eV}$ \[109\].

The results in Figure 4.10 show that the BERs for EXPRESS write increase relative to the traditional programming after the accelerated retention test. The temporary read error is a new reliability issue in 3D NAND Flash \[80, 81\]. It is not considered in this case as the BERs are less than 1% for all types of pages and hence can be corrected using standard error-correction techniques \[110, 111, 112\].

4.5 Validity of the Proposed Technique for an Arbitrary Image Data

In this section, we verify that EXPRESS is applicable to any data patterns with similar results. The chip under test uses an internal data randomizer that randomizes the user data before writing them to the NAND array. The purpose of data randomization is to ensure memory reliability by utilizing all four analog $V_t$ states. Without the data randomizer, all-zero data on both LSB and MSB pages would result in all cells being programmed into the B state. However, with data randomization, the specific cell $V_t$ state is determined by the randomization...
key, leading to an even distribution of $V_t$ states among the memory cells. This even distribution improves cell endurance and reliability, making randomization an integral feature in state-of-the-art NAND Flash chips [113].

In order to demonstrate EXPRESS for any arbitrary data, we write an Einstein image. Figure 4.11 summarizes the evaluation results for both the SLC and MLC modes of operation. We observe the same trend as in section 4.4. The bit error rate (BER) starts from 40% because the chosen image initially has 40% of the cells in the erase state. Similar to previous results, the percentage of programmed bits exceeds 99% with $n_{\text{skip}} = 2$. Nevertheless, it will be interesting to study the performance gain with the EXPRESS method when it is used for error-tolerant image classification applications using neuromorphic computing systems, as demonstrated in previous works [114].

Figure 4.11: Percentage of programmed bits as a function of partial page program time for the image data on (a) SLC and (b) LSB (blue) and MSB (red) pages.
4.6 Conclusions

In this chapter, we experimentally demonstrate energy-accuracy disproportionality in 3D NAND Flash memory chips. We propose EXPRESS, a new method for improving the energy efficiency of NAND write operations using a partial programming technique. We demonstrate EXPRESS on a 32-layer 3D NAND memory operating it in both SLC and MLC modes. We propose an adaptive algorithm for EXPRESS, considering the effects of page-to-page variability, PE cycling, and data retention. We find that energy savings in the range of 20% to 50% are achievable depending on the page type at the cost of less than a 1% loss in accuracy with EXPRESS. We also find that retention loss with EXPRESS is slightly higher than the traditional write operation. Accelerated retention test shows that BER with EXPRESS write remained below 1% for 5 years of retention time. We demonstrate the robustness of EXPRESS using an arbitrary image as a testing data pattern. The next chapter discusses the data deletion vulnerabilities in NAND Flash memory.
Chapter 5. Instant Data Sanitization on Multi-Level-Cell NAND Flash Memory

5.1 Introduction

Due to the increasing popularity of flash memories for storing private data in smartphones, SD cards, USB flash drives, and solid-state drives (SSDs), instant sanitization of user data from the flash media has become extremely important to preserving user privacy. According to the Data Protection Act (DPA) 2018 [115], the deletion of information must be real, i.e., the content should not be recoverable in any way. Unfortunately, the standard data deletion methods of today’s solid-state storage devices do not offer any instant data sanitization capabilities to the end-user [116, 117, 118, 119, 120]. Although the state-of-the-art data deletion methods make the data inaccessible through standard memory interfaces, recent research efforts demonstrate that data is partially or fully recoverable by employing advanced memory characterization techniques. According to a recent report from Blancco Technology Group, 42% of used SSDs sold on eBay hold sensitive data [121, 122]. Furthermore, the report states that the data is recoverable even from SSDs that were subjected to standard data sanitization methods. This highlights a major concern that while sellers clearly recognize the importance of
data deletion, they are, in fact, using methods that are inadequate and do not ensure true sanitization of data.

NAND flash memories present several challenges to instant data deletion due to their organization and operation. First, flash memory chips are organized in flash blocks, where each block consists of multiple pages. Basic flash operations are page write (program), page read, and block erase. Thus, flash-write operations that store data in the flash memory take place at a page-level granularity, whereas erase operations take place at a block-level granularity. Second, the NAND flash memories employ an erase-before-write paradigm that requires that blocks be erased before pages within a block are written into. This requirement makes in-place page update operations impractical because all valid pages in a block need to be copied into another block before the block is erased and the page is updated. Instead, the updated data is written into a different page, whereas the existing page is “unlinked” or marked as invalid. However, the original content remains on the “unlinked” flash page. Third, NAND flash memories have finite endurance, meaning that only a fixed number of program and erase operations are allowed on a NAND block during its lifetime; once this number is exceeded, the block becomes unreliable for storing data. To manage these basic flash operations (erase block, program page, read page) and map high-level storage abstractions into physical pages and blocks, an intermediate firmware layer called Flash Translation Layer (FTL) is employed [123]. This firmware typically runs on a dedicated flash memory controller that employs sophisticated algorithms for wear leveling,
garbage collection, error correction, and others. These algorithms try to minimize read latency, ensure data integrity, and maximize the lifetime of physical media.

To alleviate the significant overhead of erase-based techniques and achieve page-level sanitization of deleted data in flash-based storage, a page-overwrite with all-zeros technique was introduced by Wei et al. [124] and later improved by others [125, 126, 127]. The method creates an all-zero page by an overwrite operation, thus removing the information from the page. Even though all-zero sanitization works for single-level cell (SLC) memories, it cannot be directly applied to MLC memories where two logical pages share the same set of memory cells. An all-zero overwrite on one logical page may corrupt data on the other shared page. Thus, instant data sanitization remains an open problem for multi-level cell (MLC), triple-level cell (TLC), or quad-level cell (QLC) flash memories.

This chapter introduces an instant page sanitization method for MLC flash memories that prevent leakage of deleted information without any negative effects on valid data in shared pages. Our experimental evaluations on commercial NAND flash memory chips show that the proposed method ensures true data sanitization that leaves no traces of the original data, even in the analog threshold voltage distribution of the sanitized pages. The proposed method incurs minimal disturbance on valid data residing in neighboring pages. It does not require hardware modification and can be implemented using standard user-mode flash commands in the FTL firmware.

The rest of the chapter is organized as follows. Section 5.2 describes the background through a summary of existing techniques and academic research in
the area of NAND flash data sanitization. Section 5.3 describes our proposed sanitization method. Section 5.4 describes the results of the experimental evaluation, and section 5.5 concludes the chapter.

5.2 Background of Data Sanitization in NAND Flash

Standard overwrite-based erasure techniques used in hard drives do not work in NAND flash-based storage systems since in-place updates are not possible in NAND flash memories. Instead, the flash controller typically employs the following methods for data sanitization.

Block Erase

The block erasure is a basic NAND command to remove data from all pages in a flash block. This method essentially removes charges from all flash cells in the block and hence physically erases the data from the media. Typically, the garbage collection function of the FTL uses this method to remove old invalid data once the drive is almost full. A major drawback of using the block erasure for instant sanitization is its poor performance caused by significant overhead due to valid data migration [128, 129, 130] and a block erase operation that takes more time than a page program operation. In addition, this technique increases wear level and thus limits the effective capacity/operating time of storage systems. Thus, this command is sparingly used by the flash controller.
Logical Data Deletion

Since the block erasure suffers from poor performance, solid-state storage devices usually perform logical deletion of data by invalidating the page addresses of obsolete data [131, 132, 133]. The page address mapping is handled by the FTL, which performs one-to-one mapping between a logical page address and a physical page address on the flash media. Thus, for any page update operation, the FTL will write the new content to another fresh page (or block) and update the address map table to point to the new page. As a result, the old version of the data remains in the physical storage medium, from where it can be retrieved by an adversary with advanced memory interfaces. Thus, this method does not ensure true sanitization of data.

Encryption-based Data Deletion

Several authors have recently proposed data deletion methods based on encryption [117, 118, 119, 134]. The basic idea in this method is to encrypt the user file with an encryption key and store the encrypted data and the key in two separate NAND blocks. Secure data deletion is achieved by removing the keys, which can be done efficiently as keys require less space in memory. Even though encryption-based techniques are promising, they suffer from the following drawbacks. First, the encryption-based deletion method carries the risk of data recovery as its implementation may have certain issues, e.g., random number generation (for encryption key) that can be compromised by a motivated adversary. Second, encryption-based deletion requires the proper removal of encryption keys.
and any other derived values that might be useful in cryptanalysis. Third, many existing storage systems and embedded platforms do not include hardware modules for accelerating encryption/decryption tasks and rely on software solutions that can severely limit system performance.

**All Zero-overwrite-based Page Sanitization**

In order to achieve the page-level deletion in flash medium, the idea of “data scrubbing” was proposed by Wei et al. [124] and later improved by others [125, 126, 127]. The “scrubbing” based sanitization relies on programming an all-zero data, which is equivalent to the deletion of data from that page. Thus, “scrubbing” provides an alternative route to digital sanitization by reprogramming all the cells in the page. Unfortunately, the “scrubbing” method only applies to SLC flash memories, leaving the problem of instant data sanitization unsolved for MLC flash memories.

**Sanitization Using One-shot Programming**

Lin et al. [135] propose a technique for partial sanitization for MLC flash memories. Specifically, they utilize one-shot programming to sanitize data in MSB and LSB pages. Whereas this technique can achieve fast sanitization, it relies on proprietary flash commands for one-shot programming that are not typically exposed through standard NAND flash interfaces. In addition, as it does not involve any verification steps, it requires prior characterization to determine one-shot programming voltage for individual cell states, limiting its practicality. Next,
one-shot programming does not completely prevent data leakage as it eliminates only one of the possible four cell states. Note that to prevent any data leakage from a sanitized shared page, the flash cells in the shared page should be in at most two cell states.

**Hardware Supported Data Sanitization**

Kim *et al.* [136] introduce Evanesco – a hardware-supported technique for efficient data sanitization in modern flash-based storage systems. It provides data sanitization by blocking access to invalidated data either on a page or a block level. Whereas this approach is very effective, it requires additional hardware and does not apply to the existing flash memory chips. In addition, the data stored in the flash medium is not physically removed, making it susceptible to future direct or indirect data retrieval attacks.

### 5.3 Proposed Sanitization Method

To alleviate problems of the existing approaches, we introduce a method for instant sanitization that does not require any special hardware support and can be fully implemented in the FTL using non-privileged commands on commercial chips. The proposed method allows for the sanitization of (a) an LSB page while preserving the data in the corresponding MSB page, (b) an MSB page while preserving the data in the corresponding LSB page, and (c) both LSB and MSB shared pages. Even though shared LSB and MSB pages are logically independent of each other, physically, they are interdependent as they share the same set
of memory cells. Thus, overwriting an LSB or MSB page may corrupt the data stored in the corresponding shared page. Hence, data sanitization requires careful implementation in order to ensure the data integrity of the corresponding shared page.

**Figure 5.1:** (a) Four $V_t$ states of MLC memory cells with read reference voltages. State transitions for the proposed (b) LSB-only sanitization, (c) MSB-only sanitization, and (d) concurrent LSB and MSB sanitization. The sequence of steps to carry out (e) LSB sanitization, (f) MSB sanitization, and (g) concurrent LSB and MSB sanitization.

### 5.3.1 Sanitization of LSB Page Only

We start off with valid data in both LSB and MSB shared pages. Figure 6.1(a) illustrates $V_t$ distribution among all four states. If random data are written in both pages, memory cells sharing a single word line will be evenly split among four states (25% each). Figure 6.1(b) shows state transitions of cells that are required to sanitize an LSB page while preserving data in the corresponding MSB page. Specifically, all cells in the $L_1$ state need to transition into the $L_2$ state, thus converting the LSB bits from 1 to 0 while retaining values of the corresponding
MSB bits. Similarly, all cells from the $L_0$ state need to transition into the $L_3$ state. After these transitions are carried out, all cells will be in either $L_2$ or $L_3$ state. An LSB read will return all 0s because all the cells will have $V_t > V_{ref}^{LSB}$. An MSB read will return the original content from the MSB page.

To carry out the state transitions described above using common flash operations, the flash controller needs to carry out a sequence of steps as illustrated in Figure 6.1(e). The first step is to read the MSB page and store it in a temporary buffer. Next, the controller sends all-zero data for the LSB page and the buffer data for the MSB page and initiates a page program operation.

The overhead of the proposed sanitization method includes the time needed for (a) one MSB page read, (b) sending data for both LSB and MSB pages, and (c) programming data into the selected memory cells. However, this overhead is orders of magnitude smaller than the overhead required to copy all valid pages from the target flash block to another fresh block and then perform a block erase operation. It should be noted that MLC flash memory chips use two different approaches when programming MLC pages. The chip used in this research requires both pages to be loaded into the internal buffers of the flash memory chip before a program command triggers program operation on both pages. Other flash memory chips allow LSB and MSB pages to be independently programmed. In that case, the overhead of LSB sanitization includes the time needed for one page read, sending data for both pages, programming an LSB, and programming an MSB page.
5.3.2 Sanitization of MSB Page Only

Figure 6.1(c) shows the required state transitions of memory cells to sanitize an MSB page while preserving data in the corresponding LSB page. Memory cells in the $L_0$ state need to transition into the $L_1$ state, converting the MSB bits from 1 to 0. To create an all-zero MSB page, we would need to transition cells from the $L_3$ into the $L_2$ state. However, cell $V_t$ cannot be decreased with a page program operation. Thus, we cannot create an all-zero MSB page without affecting LSB data. To circumvent this limitation, we propose to move cells from the $L_2$ into the $L_3$ state. This way, the new content of the MSB page is a mirrored image of the data from the LSB page. Still, the sanitization goal is achieved as there are no traces of the original data from the MSB page, while the LSB page data is fully preserved. We can expect that the reliability of the LSB data will even improve after the sanitization of the MSB page. The voltage margin between $L_1$ and $L_3$ states is significantly higher than before the sanitization, so $0 \rightarrow 1$ bit flips in the LSB page are less likely.

Figure 6.1(f) illustrates the sequence of steps needed to sanitize only an MSB page. First, the flash controller reads data from the corresponding LSB page and stores it in a buffer. Then, the inverted buffer content is transferred to the chip for the MSB page programming. Finally, the program command is initiated that programs the selected physical page. Please note that we typically do not have to resend the content for the LSB page as it can be copied internally in the flash memory chip. The overhead of the proposed MSB page only sanitization
includes the time needed for (a) one LSB page read, (b) sending data for the MSB page, and (c) programming data into the selected memory cells. Please note that we do not count the controller’s processing time as inverting the content of the buffer is a very simple operation that is not going to place any computational burden to the flash controller.

5.3.3 Concurrent LSB and MSB Page Sanitization

In order to sanitize concurrently both the LSB and MSB shared pages, all the cells of the memory layer need to be moved into the $L_3$ state, as illustrated in Figure 6.1(d). In that case, the sanitized LSB page will contain all 0s, and the sanitized MSB page will hold all 1s. Figure 6.1(g) shows the sequence of steps needed to carry out the concurrent sanitization of shared pages. The controller sends all 0s for the LSB page and all 1s for the MSB page into the respective flash memory buffers and initiates a shared page program operation. This method incurs even smaller overhead than the LSB-only or MSB-only sanitization, as it is agnostic of the original data stored in the shared page and does not require any page read operation. Please note that the proposed method allows for the serialization of individual page sanitizations in the case of NAND flash chips with independent programming of LSB and MSB pages. For example, to sanitize the corresponding MSB page when its LSB pair is already sanitized, the flash controller will send an ‘all-1’ data pattern and issue an MSB page program operation. To sanitize the corresponding LSB page when its MSB pair is already
sanitized, the flash controller will send an ‘all-0’ data pattern and issue an LSB page program operation.

5.4 Experimental Evaluation

The experimental evaluation is performed on multiple 2D and 3D NAND memory chips from two different major NAND manufacturers. The chips support so-called read offset operations that allow the flash controller to adjust read reference voltages $V_{\text{ref}}^{\text{LSB}}, V_{\text{ref}}^{\text{MSB}}_1, V_{\text{ref}}^{\text{MSB}}_2$ (Figure 6.1(a)). Nominally, this feature is used to help in recovering data when standard ECC correction fails. In this research, we progressively increase an offset and thus shift the reference voltages to the right from the default level to $V_{\text{ref}} + \Delta V \times i$, $i = 0, \ldots, 127$, $\Delta V = 7.5$ mV and then read data from the LSB and MSB pages. This way, we can extract the $V_t$ distribution of memory cell states. For example, by shifting $V_{\text{ref}}^{\text{MSB}}_1$ from its default value to the right by $\sim 952$ mV in steps of $7.5$ mV, we can extract the distribution of the $L_1$ state.

5.4.0.1 Extracting the $V_t$ Distribution

The process being described is a method for extracting the threshold voltage ($V_t$) distributions of the various states of memory cells in a NAND Flash memory chip. It operates by adjusting the read reference voltage ($V_{\text{ref}}$) to observe the behavior of the memory cells.

In NAND Flash memory, the data stored in each cell is represented by the cell’s threshold voltage ($V_t$). Depending on the specific technology used (SLC,
MLC, TLC, QLC, etc.), each cell can represent one or more data bits, with each possible data value corresponding to a range of threshold voltages.

![Graph showing threshold voltage distributions](image)

**Figure 5.2:** Extracting the threshold voltage distribution of various states of NAND Flash memory.

The read reference voltage ($V_{\text{ref}}$) is the voltage level that the Flash controller uses to determine the stored data in a cell. By shifting $V_{\text{ref}}$ and observing the resulting read values, we can infer the threshold voltage distributions of the memory cell states as shown in Figure 5.2.

The detailed steps of the process are as follows:
1. The read offset operation begins with the default read reference voltage, $V_{\text{default}}$.

2. An offset voltage, $\Delta V$ (7.5 mV in this case), is added or subtracted from $V_{\text{default}}$ incrementally to shift the reference voltage. This is represented by the formula $V_{\text{ref}} = V_{\text{default}} + \Delta V \times i$, where $i$ ranges from -127 to 127.

3. For each adjusted $V_{\text{ref}}$, the Flash controller reads data from the corresponding memory cell pages.

4. By shifting $V_{\text{ref}}$ to the right of its default value by about 952 mV in increments of 7.5 mV, we are essentially probing the right tail of the threshold voltage distribution. This allows us to extract the distribution of the programmed state, which corresponds to the higher threshold voltages.

5. The same process is repeated for other cell states, adjusting $V_{\text{ref}}$ in increments of $\Delta V$ to the left and/or right of the default reference voltage. In doing so, we can extract the threshold voltage distributions of all memory cell states.

By following this procedure, the performance of the memory cells in the NAND Flash chip under test can be characterized and potentially diagnose issues or optimize the performance of the Flash controller. The numbers used in the process may differ depending on the chip manufacturer or the chip technology.
5.4.1 Evaluation of LSB Page Only Sanitization

To evaluate the effectiveness of the proposed LSB-only sanitization method, we conduct the following experiment. First, we write two different images of 16 KiB into shared pages; an Albert Einstein image is written to an LSB, and a Mona Lisa image is written to the corresponding MSB page. We read the written images and plot them in Figure 6.2(a). The next step involves the characterization of memory cells’ states before sanitization is performed. We utilize read offset operations on the read reference voltages $V_{ref1}^{MSB}$, $V_{ref}^{LSB}$, $V_{ref2}^{MSB}$ to extract $V_t$ distributions for the $L_1$, $L_2$, and $L_3$ states, respectively. Black dots in Figure 6.2(c)-(e) show the percentage of cells from the shared page with $V_t$ in each 30 mV range ($4 \times 7.5 = 30$ mV), whereas the black curves approximate the extracted $V_t$ distribution for the $L_1$, $L_2$ and $L_3$ states, respectively. Please note that $V_t$ of cells in the $L_0$ state is not accessible with the given range of $V_{ref}$ sweep.

The next step in the evaluation is to carry out the LSB page only sanitization as described in Figure 6.1(e). The pages are then read and plotted in Figure 6.2(b). The LSB page with the blue frame is fully sanitized (contains all 0s), whereas the MSB page is intact. Thus, the LSB page is fully sanitized with no information leakage. We calculate the raw bit error rate (BER) for the Mona Lisa image after sanitization. We find that it is the same or even lower than before sanitization. In addition, we extract the $V_t$ distribution of cells’ states after sanitization. Green dots in Figure 6.2(c)-(e) show the percentage of cells with $V_t$ in the 30 mV range after LSB sanitization, whereas the green curves approximate the
Figure 5.3: (a) MSB and LSB page before sanitization. (b) MSB and LSB page after LSB sanitization. (c)-(e) Measured $V_t$ distribution of cells before and after sanitization for the $L_1$, $L_2$, and $L_3$ states, respectively. Cell $V_t$ is measured with respect to the default read reference voltage of the page shown on the x-axis.

extracted $V_t$ distribution for the $L_1$, $L_2$ and $L_3$ states, respectively. There are several takeaways from this evaluation. First, $V_t$ distribution after sanitization does not distinguish the newly transferred cells from the original ones indicating true sanitization of the LSB page. Even if an adversary performs cell $V_t$ analysis on a sanitized page, he/she will not be able to recover the original data. Second, the $V_t$ distribution forms slightly longer tails after the sanitization operation. Since there is a sufficient voltage margin between the reference voltage and cell $V_t$, the BER of the valid data in the MSB page is not affected by these long tails.
5.4.2 Evaluation of MSB Page Only Sanitization

Figure 5.4 summarizes our evaluation results for the MSB page sanitization. The selected MLC layer is programmed to contain images of Einstein and Mona Lisa images on shared LSB and MSB pages, respectively, as shown in Figure 5.4(a). The cell states are characterized like in the previous experiment, and then the steps for the MSB page sanitization are performed as described in Figure 6.1(f). The MSB and LSB pages are read and plotted in Figure 5.4(b). We find that the LSB data remains intact, and an inverted Einstein appears instead of the original Mona Lisa image in the MSB page. The effectiveness of the proposed method is further analyzed through $V_t$ distribution measurements on the shared pages. Figure 5.4(c)-(e) show the $V_t$ distribution of flash cell states before sanitization in black and after sanitization in green. As expected, we find that after MSB page sanitization, there are only two $V_t$ states, $L_1$ and $L_3$. We also verify that cells from the $L_0$ state have moved to $L_1$ and cells from the $L_2$ state have moved to $L_3$.

We would like to emphasize several important points that we can infer from the $V_t$ distributions. First, the evaluation confirms that the MSB page is truly sanitized as the original Mona Lisa image is replaced by an inverted Einstein image. $V_t$ distribution of transferred cells and the original cells in a given state remains indistinguishable. Second, the BER of the LSB data improves after sanitization. Indeed, the separation between $L_1$ and $L_3$ is quite high to cause any errors due to noise or retention loss. Thus, we expect the reliability of the LSB
Figure 5.4: (a) MSB and LSB page before sanitization. (b) MSB page and LSB page after MSB sanitization. (c)-(e) $V_t$ distribution of cells before and after sanitization for the $L_1$, $L_2$ and $L_3$ states, respectively. Cell $V_t$ is measured with respect to the default read reference voltage of the page as shown on the x-axis.

data to increase after the sanitization of the corresponding MSB page. Third, an inverted LSB page data is copied on the MSB page. This can be utilized as a redundant copy of the LSB data to correct errors in the original LSB page if required.

It should be noted that having an inverted copy of data from the LSB page in the sanitized MSB page does not imply any information leakage. The sanitized MSB page does not contain any trace of the original data, and a copy of the data that is already in the flash memory cannot be considered as information leakage. However, the FTL will have to keep track of all sanitized pages. Thus, if the LSB page (Einstein image in our example) is deleted after the corresponding MSB
page is deleted (which now holds inverted Einstein image), we need to sanitize both pages as described in Figure 6.1(d). All the cells of the shared pages are moved to $L_3$ state after both LSB and MSB pages are sanitized, creating an all zero LSB page and all one MSB page.

5.4.3 Evaluation of Concurrent LSB and MSB Page Sanitization

Finally, we evaluate the effectiveness of the concurrent LSB and MSB page sanitization. We start by writing Einstein and Mona Lisa images on shared LSB and MSB pages, respectively (Figure 6.3(a)). Then, we sanitize both pages following the steps described in Figure 6.1(g). After both the pages are sanitized, we find all-zero data on the LSB page and all-one data on the MSB page (Figure 6.3(b)). The corresponding $V_t$ distributions before and after sanitization are shown in Figure 6.3(c)-(e). We find that all cells are transferred to the $L_3$ state after concurrent sanitization of LSB and MSB pages.

5.4.4 Effects of Page Sanitization on Retention of Valid Data

We evaluate the page sanitization effects on the retention errors of the valid data in the shared pages by baking the memory chip at a higher temperature (120°C) for 1, 2, or 3 hours. Using the acceleration factor based calculation, we find that the 3 hours of baking time corresponds to 5 years at room temperature assuming activation energy for charge loss in 3D NAND as $E_A = 1 \text{ eV}$ [137].

Figure 6.4 summarizes our evaluation results. We compare retention errors of unsanitized standard LSB/MSB pages with the retention errors on the valid
Figure 5.5: (a) MSB and LSB page before sanitization. (b) MSB and LSB page after concurrent MSB and LSB sanitization. (c)-(e) $V_t$ distribution of cells before and after sanitization for the $L_1$, $L_2$ and $L_3$ states, respectively. Cell $V_t$ is measured with respect to the default read reference voltage of the page as shown on the x-axis.

Data of the LSB/MSB pages after the corresponding shared pages are sanitized. We find that the retention reliability of the LSB page improves after the corresponding MSB sanitization. However, the retention reliability of the MSB page after LSB sanitization degrades compared to the unsanitized MSB pages. Since $V_t$ margin between program states increases after MSB sanitization as shown in Figure 6.1(c), retention reliability improves for the LSB page. However, the $V_t$ margin between $L_2$ and $L_3$ states reduces after LSB sanitization as shown in Figure 6.1(b), and hence MSB page shows more retention errors after LSB sanitization. Nevertheless, the BER after 3 hours of baking remains significantly lower than what can be corrected by the standard error correction code (ECC). Thus,
we find that the proposed mechanism does not severely impact the retention of shared valid pages.

**Figure 5.6:** Effects of page sanitization on the retention errors of valid data of the shared page.

### 5.4.5 Effects of Page Sanitization on Valid Data in Neighbor Layers

Since sanitization of a page involves an extra write operation on a memory page, it may disturb the valid data on neighboring memory layers. We evaluate the effects of MLC page sanitization on the BER of the neighboring memory layers. We first write random data in the entire memory block. Then, we sanitize
shared memory pages in the MLC layer one by one. There are 12 shared pages (12-LSB and 12-MSB pages) in an MLC layer of the 3D NAND memory block. We perform concurrent LSB and MSB sanitization as it causes the worst-case disturbance on the neighboring memory layers.

Figure 5.7 summarizes the evaluation results where we plot the BER of the valid pages in the neighboring memory layers as a function of the number of sanitized memory pages in a given memory layer. We observe an increase in

Figure 5.7: Effects of proposed page sanitization on the valid data on neighboring memory layers.
BER for the data stored in the nearest neighboring layer (Layer $n - 1$) as we increase the number of sanitized pages in a target layer (Layer $n$). The maximum BER increase is $\sim 5$ times after all the pages of a given layer are sanitized. Though this is seemingly a significant increase in BER, the actual magnitude is still significantly below what can be corrected using standard ECC. Interestingly, other memory layers ($n - 2$ and beyond) remain unaffected by the sanitization process.

5.4.6 Evaluation on Multiple Chips

Table-5.1 summarizes our evaluation results that demonstrate that the proposed technique is applicable to 2D and 3D MLC chips manufactured by two different vendors. Table-5.1 quantifies the impact that the proposed sanitization has on the valid data in the corresponding shared page by comparing its average BER before and after sanitization is performed. The first row in the table corresponds to a 64-layer 3D NAND chip that has been used in the evaluation thus far. The second row in the table shows the results from a 19 nm 2D NAND chip from a different vendor. We find that sanitization of an MSB page improves the BER of valid data on the corresponding LSB page, whereas sanitization of an LSB page increases the BER of valid data on the corresponding MSB page. These findings are in line with those described in section 5.4.
Table 5.1: Multiple chip sanitization results.

<table>
<thead>
<tr>
<th>Chip Type</th>
<th>LSB Page BER</th>
<th>MSB Page BER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Before MSB</td>
<td>After MSB</td>
</tr>
<tr>
<td>Sanitization</td>
<td>Sanitization</td>
<td>Sanitization</td>
</tr>
<tr>
<td>64-Layer 3D NAND</td>
<td>3.9x10^{-4}%</td>
<td>0%</td>
</tr>
<tr>
<td>19-nm 2D NAND</td>
<td>0.007%</td>
<td>0.001%</td>
</tr>
</tbody>
</table>

5.4.7 Limitations

It should be noted that some flash memory chips may not support overwriting of memory pages once a block is fully programmed, thus preventing any overwrite-based data sanitization. Next, internal data randomization of NAND memory chips can also pose challenges to overwrite-based sanitization. However, we believe these are not fundamental limitations and can be addressed by chip vendors.

5.4.8 Future work

The proposed mechanism has been tested for 2D and 3D MLC flash memory chips. This work can be extended in several directions as follows. The first is to evaluate the proposed mechanism in TLC/QLC flash memories. Although the proposed mechanism should work for TLC/QLC chips with some minor modifica-
tions, a detailed evaluation is needed to determine its effectiveness and robustness. The higher number of bits per cell in TLC/QLC chips results in lower threshold voltage margins between the cell states. Thus, sanitization steps may result in increased BERs in shared and neighboring pages.

The second direction is to further investigate original data recoverability after sanitization is performed. Whereas our evaluation establishes a full sanitization by observing the states of individual cells, it is conceivable that more sophisticated attacks may try to uncover the differences in threshold voltages between cells that were originally in a given state and cells that are moved into the given state through the sanitization. This evaluation requires a detailed flash cell characterization.

The third direction requires investigation of reliability issues in worn-out pages. Our experiments mainly focused on fresh memory blocks and more rigorous evaluation is needed on worn-out blocks.

The fourth direction involves analyzing subpage sanitization. With subpage sanitization a portion of a page is sanitized while the rest of it retains the original data. We have performed a set of experiments with subpage sanitization and they indicate that this approach is feasible in all combinations discussed in this chapter. However, more detailed experiments are needed to investigate its robustness.
5.5 Conclusions

This chapter describes an instant page sanitization method for MLC flash memories with minimal impact on the corresponding shared memory page. The proposed method does not affect memory endurance (no block erase operation is used), does not reduce the available memory space, and can be implemented within the standard FTL with a minimal firmware change. The next chapter discusses a new method to hide secret information, providing an additional layer of security inside 3D NAND flash memory.
Chapter 6. Hide-and-Seek: Hiding Secrets in Threshold Voltage Distributions of NAND Flash Memory Cells

6.1 Introduction

Steganography is a well-established technique for embedding secret information into digital objects such as images by intentionally adding small distortions [138, 139]. Steganography has a distinct advantage compared to encryption-based data protection as it hides the very presence of secret data from the adversary. Encryption techniques can defend against a passive adversary trying to steal data storage devices’ secrets. However, it cannot defend against an active adversary, who can find ways to coerce the device owner into disclosing the decryption key.

Plausibly deniable data storage solution in the solid state drive has recently gained significant traction due to the pervasive usage of solid-state storage media in our daily life (e.g., in mobile devices). Several solutions for achieving plausible deniability in the solid state drives have been proposed recently, such as INFUSE [18], PEARL [140], DEFTL [141], MDEFTL [142]. Similarly, several steganographic files system-based solutions have been proposed [143, 144, 145, 146]. However, most existing solutions incorporate deniability in the file system layer or the flash translation layer. In contrast, the proposed work incorporates the
plausible deniability in the physical properties of the NAND storage media, which is immune to software-based deniability compromises.

Wang et al. [147] investigated data hiding in flash memory, proposing a covert channel using inherent variations of program times in memory cells to hide data. By manipulating the physical properties of selected cells, program times can be varied subtly to hide data. The process is slow and reduces device lifetime due to repeated program-erase cycling.

Zuck et al. [148] propose a new technique called “Stash in a Flash” to address the limitations of Wang et al.’s method. This method hides data in flash memory by manipulating the threshold voltage of randomly selected cells in the erase state. The threshold voltage variations of hidden and public data are indistinguishable, resulting in improved hiding and recovery throughput. However, the technique relies on special flash memory operations that are not commonly available to users, and hiding data in erased state cells makes it vulnerable to read, program disturb, and cell interference effects [149, 150, 151, 152, 153, 154, 155].

This chapter proposes a new method of steganographic storage in NAND flash media using the threshold voltage variation of programmed memory cells. Unlike previous methods, the proposed method offers more design space variables to control the bit accuracy of secret/public data with increased tolerance against memory disturbances. The approach requires no special memory operations and works with MLC (2-bits/cell), TLC (3-bits/cell), and QLC (4-bits/cell) flash memory configurations. Experimental evaluations show negligible distortions in threshold voltage distributions that are undetectable by an adversary.
The proposed method achieves over 97% accuracy of recovered secret data with minimal impact on co-existing public data. Design trade-offs are also explored.

The rest of the chapter is organized as follows. Section 6.2 presents the threat model and provides a system overview of the proposed technique. Section 6.3 details the suggested approach for writing/programming and reading secret data. Section 6.4 delves into the experimental evaluation results. Finally, Section 6.5 concludes the chapter.

6.2 Threat Model and System View of the Solution

We assume that the adversary has physical access to the storage device and he/she is capable of performing low-level memory operations such as page read/write, block erase, and cell $V_t$ distribution measurement of the storage media. We also assume that the adversary knows the storage encryption key and can retrieve all the content in the storage device. However, we assume the adversary does not know the secret key to select memory cells containing the hidden data. Hence the adversary will not be able to recover the secrets and will not be sure if any hidden secrets are there.

The system view of the proposed data hiding technique is described in Figure 6.1. The Einstein image, representing the public data, is assumed to conceal the Lincoln image as the secret data. To encode the secrets, the memory bits of the public data are selected using a secret key. A hiding flash encoder (HFE) encodes the secret data. We assume the HFE can be implemented within the FTL of the storage device. If an adversary gains physical access to the storage device,
they can only retrieve the Einstein image (public data) using the common flash decoder (CFD). The adversary would be unable to extract the secret data through separate probing of the NAND flash chip using its controller and conducting $V_t$ analysis unless she or he possesses knowledge of the secret key. Furthermore, the adversary would be unable to ascertain whether any secret data has been encoded in the storage medium. The user with the secret key will be able to decode the secret message with a hiding flash decoder (HFD).

### 6.3 Data Hiding and Reading Technique

This section describes the proposed hidden data writing and reading scheme that exploits the physical properties of flash memory cells. Figure 6.2 illustrates our data-hiding technique. We use an SLC flash memory to simplify the discussion, though other configurations are possible (MLC, TLC). We start from a page
in a memory block that was previously erased, and thus all cells are initially in the erased state (green dots). In our example, we assume we want to hide secret data, Lincoln’s image, within the programmed (0 bits) of public data, an Einstein’s image. So, the size of the secret data needs to be lower than the number of 0 bits in the public image. For example, the size of the secret image in Figure 6.2 is only 5.64% bits of the total number of 0s in the public image.

To hide Lincoln’s image, we create subtle differences in the threshold voltages of the selected flash memory cells within the programmed state. We encode strong 0s (colored in blue) that correspond to 0 bits of Lincoln’s image and weak 0s that correspond to 1 bits of Lincoln’s image (colored in red). The other programmed cells of Einstein’s image have neutral zeros (colored in black), meaning they do not carry any secret data. Flash memory inherently poses cell-to-cell $V_t$ variation after a program operation. Several physical mechanisms, such as program noise, read noise, cell-to-cell process variation, and interference effects, all contribute to the $V_t$ variation of the programmed cells. Thus, the proposed
method hides secret data from the programmed memory cells’ inherent $V_t$ variation.

### 6.3.1 Hiding Secret Data

The process of hiding data involves three distinct steps (Figure 6.2(b)), starting from an erased memory page shown as green cells on the top row. Based on the secret key, public data, and hidden data, we create the three binary contents strong zeros, weak zeroes, and neutral zeros.

**Step 1.** The first step is to write strong zeros. We exploit the neighbor word line (WL) interference property of modern flash arrays to guarantee that the programmed cells will have threshold voltage in the upper portion of the $V_t$ distribution for the programmed state. By writing into physically adjacent neighboring word lines ($WL_{n+1}$), the threshold voltage in the target word line ($WL_n$) can be increased. Thus, step 1 involves regular page programming operation into the hiding page of the target word line ($WL_n$) and a page from the neighboring word line ($WL_{n+1}$), ensuring that programmed cells in the hiding page become strong 0s. The page in the neighboring word line ($WL_{n+1}$) holds valid data eliminating the chance of leaving a clue for reverse steganography.

**Step 2.** This step involves programming cells that contain neutral zeros. We create a bit vector to program neutral zeros in the target page based on the public data and secret key. The bit vector is sent to the flash memory chip, and a regular page program operation is issued. In this step, most 0s from Einstein’s image get programmed.
**Step 3.** The final step involves programming weak zeros. Whereas the previous two steps rely on regular page program operations, this step utilizes a partial program operation [14] that starts as a regular page program operation but gets terminated by prematurely issuing a RESET command. Partial program operations result in weak zero bits whose threshold voltage will reside in the lower tail of programmed $V_t$ distribution, as shown in Figure 6.2(c). The 1 bits in Lincoln’s image are defined as these weak 0s.

### 6.3.2 Recovering Secret Data

The standard (or default) memory read operation cannot distinguish between the strong and weak zero bits; hence, the secret image remains invisible to the adversary. An elaborate memory read operation is needed to recover the secret image from the visible public image. Figure 6.2(c) illustrates the secret image recovery scheme. The proposed reading method critically depends on the choice of the secret read reference voltage ($V_{secret}^{ref}$), which allows to distinguish between the strong and weak zero bits. We utilize the Read Offset [16] features to distinguish between strong and weak zeros to recover the secret image. A shared secret key can determine the bit position of the public image that holds the secret.

The overhead of the proposed hiding method involves two extra page program operations for implementing steps 2 and 3, as shown in Figure 6.2(b). Overhead in the recovery process involves shifting the read reference voltage to $V_{secret}^{ref}$ by read offset command [156]. Except for the last word line in a block, all other word lines can be used for data hiding.
6.4 Experimental Evaluation

6.4.1 Experimental Setup

We perform the experimental evaluation on several commercial-off-the-shelf (COTS) 64-layer 3D NAND TLC chips. The chips under test support so-called read offset operations that allow the flash controller to adjust read reference voltage, $V_{\text{ref}}$. In this process, we add an offset voltage of 7.5mV incrementally to the reference voltage and thus shift the reference voltage. This way, we can extract the $V_t$ distributions of all memory cell states [157]. The detailed step-by-step extraction process is described in the previous chapter.

6.4.2 Experimental Evaluation of Writing Method

Figure 6.3 provides an experimental demonstration of the three programming steps. After each programming step, the state of the Einstein image is shown by reading the memory content using the default read operation. Note that, after Step 1, a small portion of the Einstein image is written corresponding to zero bits, coinciding with strong 0 bits of Lincoln’s image. For simplicity, we chose the bit positions located on the left side of the Einstein image as the secret zeros. Hence, all the secret zeros are depicted in the left portion of the image after Step 1 (colored in blue in Figure 6.3). In practice, the location of the secret zeros can be randomly chosen from all possible locations of the public image. After Step 2, most of the Einstein image is written. The zeros in this step do not carry any secrets. Finally, after Step 3, the complete Einstein image is visible.
Figure 6.3: The evolution of the public image and corresponding $V_t$ distribution of program bits after steps 1, 2, and 3 are shown in blue, black, and red color, respectively.

This step’s zeros are weak and represent secret ‘1’ data. Partial programming is employed in Step 3 to control the $V_t$ values of the weak zero bits. The cell $V_t$ distribution, measured after programming steps 1, 2, and 3, is shown in Figure 6.3 using blue, black, and red colors, respectively. We use a cumulative distribution plot to show the measured cell $V_t$ for the three sets of zero bits in the public data: (a) the strong zero bits (blue) which are secret zero bits (b) the neutral zero bits (black) which constitute the majority of the public zero bits and (c) the weak zero bits (red) which are secret ones. Note that there is a distinguishable difference between the $V_t$ distribution of the strong and weak zero bits. We exploit this $V_t$ difference to recover the secret data described in the following section.
6.4.3 Experimental Evaluation of Reading Method

Figure 6.4 shows the experimental evaluation results for the reading method of secret data. The figure shows the recovered Lincoln image for six different $V_{secret}^{ref}$. We quantify the reading efficiency using the bit accuracy percentage of the recovered image. The default read reference voltage ($V_{secret}^{ref} = V_{default}^{ref}$) reads the hidden image as an all-zero image (black), and hence bit accuracy is poor. Similarly, if $V_{secret}^{ref} > V_{default}^{ref} + 1V$, the secret image is read as an all-one image.

![Figure 6.4: Experimental evaluation of hidden data recovery method.](image)

Figure 6.4: Experimental evaluation of hidden data recovery method.
Thus, there exists an optimal $V_{\text{secret ref}}^\text{opt}$ for which the bit accuracy of the recovered image is the highest. In this particular example, the optimal read reference voltage is found to be $V_{\text{ref}}^\text{secret (opt)} = V_{\text{ref}}^\text{default} + 0.45\text{V}$. The optimum $V_{\text{ref}}^\text{secret}$ is a design parameter that needs to be pre-characterized for a given chip. Note that the accuracy of the recovered image is not 100% even with optimum $V_{\text{ref}}^\text{secret}$. This is due to the overlap between the $V_t$ distribution of strong and weak zero bits. It is very hard to precisely control the cell $V_t$ even with a three-step programming method, and hence achieving very high bit accuracy of the hidden data will be challenging. Multiple redundant copies of the secret image should be stored, and a majority voting scheme might be employed to achieve close to 100% bit accuracy of the hidden data.

### 6.4.4 Trade-off Between the Accuracy of Public and Secret Data

Here is an inherent trade-off between the accuracy of the public data and the secret data. Figure 6.5 illustrates this trade-off. If step 3 of the writing scheme is skipped, the accuracy of the secret data will be approximately 100% with the default read. Since weak zeros are not programmed due to skipping step 3, distinguishing weak and strong zeros becomes very efficient. However, the corresponding public data will have very poor accuracy due to $0 \rightarrow 1$ bit flip errors, as illustrated in Figure 6.5.

Next, if we perform step 3 with a partial program operation with increasing partial program time, the accuracy of the public data improves, as shown with red data points in Figure 6.5. With longer partial program duration, more zero bits
are programmed, and hence bit-accuracy of the public data improves. However, the accuracy of the secret data degrades with longer partial program time, as demonstrated with blue data points in Figure 6.5. With increasing $V_t$ values of the weak zero bits, it becomes increasingly difficult to distinguish between the strong and weak zeros, and hence bit accuracy of the secret data decreases. Therefore, there exists an optimum partial program time that offers the best trade-off by ensuring high bit accuracy (>$99\%$) of the public data as well as secret data. In our characterization, the optimum partial program time is found to be around $\approx 600 \, \mu s$. The optimum partial program time is a design parameter in our technique that needs to be pre-characterized for a given chip.

We find that the accuracy of the secret data can be improved further if one employs a sequence of partial write operations with a fixed partial program time in step 3. The multiple rounds of issuing a partial page program operation followed by a read operation from the target page will provide the bit accuracy estimate of the public data after each partial write operation. The process is terminated once all weak zeros are programmed. We have implemented step 3 of our algorithm using five consecutive partial write operations with partial time $= 400 \, \mu s$ and found that it offers a very good trade-off between the bit accuracy of public and secret data as described in the next section.

6.4.5 Trade-off Between Accuracy and Detectability

The proposed method selectively modifies the threshold voltage ($V_t$) values of zero bits in the public data. Consequently, the $V_t$ distribution of these zero bits
Figure 6.5: Trade-off between the accuracy of secret data and public data explored by varying partial program time in step 3 of the hiding method. could be slightly distorted compared to the distribution that doesn’t contain any secrets. This slight alteration could potentially provide an observant adversary with a hint about the existence of concealed secrets.

Figure 6.6 (a) presents a comparative analysis of the $V_t$ distribution of zero bits, both with and without the presence of hidden secrets. The gray lines in
Figure 6.6: Comparison of $V_t$ distribution of zero bits with secret encoding (proposed write) and without secret encoding (traditional write in gray color). $V_t$ distribution with the proposed write is shown with three colors corresponding to three different programming conditions. The standard deviation of the $V_t$ distribution in the (b) lower tail and (c) the upper tail obtained from several memory pages within a chip.

![Diagram](image)

the plot signify the $V_t$ distribution in the absence of any secrets. The presence of several $V_t$ distribution curves derived from the same chip emphasizes the inherent variability in the $V_t$ distribution shape within a single memory chip.

Our findings also suggest that if step 3 of the proposed writing scheme is not optimized properly, it may inadvertently generate a significant signature of hidden secrets that can be revealed through meticulous $V_t$ analysis. For instance, the $V_t$ distribution that results from a lower partial write duration (at step 3) exhibits a long lower tail, suggesting the potential presence of hidden secret data. Increasing the partial write time reduces this lower tail but at the expense of
lowering the bit accuracy of the secret data. Hence, a longer partial write time is beneficial for minimizing any anomalous signatures in the $V_t$ distribution. Our study shows that the optimal balance between detectability and the bit accuracy of recovered data is achieved with multiple partial write operations.

The detectability can be quantified by measuring the standard deviations ($\sigma$) of the $V_t$ distribution. Given the asymmetry in the upper and lower tails of the cell threshold voltage distributions, we calculate the $\sigma$ of both tails separately. Figure 6.6 (b) and Figure 6.6 (c) illustrate the $\sigma$ values of the lower tail and upper tail, respectively. The grey dots represent the deviations in the $V_t$ distributions that do not contain secret information. We conducted tests on several pages from different blocks and layers, and the scattered grey dots symbolize the inherent process variations across pages in a flash memory chip.

Comparatively, the colored symbols in Figure 6.6 (b) and (c) represent the $\sigma$ values of the lower and upper tails of distributions that contain hidden information. The alignment of the $\sigma$ values for the hidden distributions confirms that the secret data is indistinguishable and merges seamlessly with the inherent process variations observed in the flash memory cell threshold voltage distributions.

6.5 Conclusions

In this chapter, we have demonstrated a data hiding technique in the $V_t$ variation of programmed flash memory bits using commercially available high-density 3D NAND flash memory. Our experimental evaluation results show that the proposed technique can hide secret data without significantly distorting the
programmed state $V_t$ distribution. Still, it offers more than 97% bit accuracy in hidden data and more than 99% bit accuracy in public data. In general, our proposed method is universally applicable to all NAND flash chips from any manufacturer, and it can be implemented within the FTL of the storage system.
Chapter 7: Conclusions and Future Work

In conclusion, this thesis provides an in-depth examination of the 3D NAND flash memory technology, addressing its potential and challenges. This research delved into the difficulties of device architecture, manufacturing processes, and system-level integration of 3D NAND flash memory, highlighting its dominance over its planar counterpart in terms of storage density, reliability, and energy efficiency.

Chapter 3 delves into the endurance variability and the “big block issue” in 3D NAND Flash memory. With the advent of 3D NAND technology, managing large flash memory blocks has become a significant challenge. Experiments reveal that pages in the top and bottom layers have lower endurance than the middle layers, leading to flash memory storage being underutilized. A novel intra-block wear-leveling algorithm is proposed to enhance overall utilization, adjusting dynamically to the different endurance levels in the memory blocks.

Chapter 4 investigates the energy efficiency in storing data in 3D NAND memory. As storage density and block size increase, latency and energy consumption follow, challenging data integrity during program operations. The chapter introduces a technique, EXPRESS, to improve energy efficiency by prematurely terminating flash write operations. Tests show EXPRESS can reduce energy use
by 20%-50% compared to standard flash writes, with minimal data integrity loss.
Further, potential issues from page-to-page variability, program-erase cycling, and
data retention are examined, with proposed solutions to lessen their impact.

Chapter 5 overcomes the challenge of immediate data erasure in NAND
flash memories. Traditional data deletion processes increase overheads and can
contribute to storage memory wear, impacting memory device endurance and ef-
ficiency. Current solutions, like unlinking physical page addresses, run the risk of
data exposure. This chapter proposes an instant page data sanitization approach
specifically for NAND flash memories that prevent data leaks from deleted in-
formation without impacting valid data on shared pages. This method, tested
on commercial 2D and 3D flash memory chips, effectively balances data security
with system performance.

Chapter 6 proposes a new method for hiding sensitive data, enhancing
security in 3D NAND flash memory. The technique, tested for accuracy and
detectability, ensures data remains undisclosed even when low-level access threats
exist. This improves overall functionality and reliability, making 3D NAND flash
memory suitable for applications requiring critical data confidentiality.

The discoveries gleaned from this dissertation significantly impact the ad-
vancement of non-volatile memory technologies. The insights procured serve as
a beacon for future investigations, fostering the creation of increasingly reliable,
efficient, and secure data storage solutions. Here are some promising directions
for future research that could build upon this study:
1. The application of undoped poly-silicon in 3D NAND flash memory tends to reduce the cell current magnitude. This could impact storage memory performance and overall efficiency. Future research should investigate mechanisms to mitigate this reduction by a system-level approach, thereby optimizing the performance of 3D NAND flash memories.

2. The cross-temperature effect, which refers to the variation in memory performance due to changes in operating temperature, is an under-explored area in 3D NAND flash memory research. A more detailed study of this effect could lead to fascinating insights, helping to design memory devices that perform reliably across a range of operating conditions. A real-time on-chip temperature sensor can be used to counter this effect.

3. In-memory computing, where data processing is performed in the same place where data is stored, presents a promising approach to enhancing processing speeds and efficiency in storage systems. It would be worthwhile to examine how in-memory computing can be effectively implemented with 3D NAND flash memory, potentially leading to transformative advancements in-memory technology.

4. Energy efficiency is a vital concern in memory technologies. As such, developing efficient strategies for erase and read operations in 3D NAND flash memory could be a promising avenue for research. This could involve exploring ways to minimize energy consumption during these operations or
devising techniques to maximize performance while maintaining a low energy profile.

5. TLC (Triple-Level Cell) sanitization and sub-page sanitization represent essential strategies for enhancing data security and integrity in NAND flash memories. A thorough examination of these techniques and how they can be applied effectively to 3D NAND flash memory could reveal innovative methods for ensuring data privacy and reliability.

The implications of this research transcend 3D NAND flash memory, sparking new ways to tackle data storage dilemmas in a progressively digitized world. Ultimately, this dissertation emphasizes the potential of 3D NAND flash memory as a cornerstone technology balanced to satisfy the escalating needs for high-performing, dependable, and energy-efficient data storage.
References


[27] S. Choi et al., “A novel NAND-type MONOS memory using a field-enhanced tunneling scheme for a better programming efficiency and a higher speed


[53] R. Yamada, Y. Mori, Y. Okuyama, J. Yugami, T. Nishimoto, and H. Kume, “Analysis of Detrap Current Due to Oxide Traps to Improve Flash Mem-


