DISSERTATION

SECURITY VULNERABILITIES IN 3D NAND FLASH MEMORY: CHALLENGES IN DATA SANITIZATION AND REVERSE ENGINEERING

Submitted by

Matchima Buddhanoy

Department of Electrical and Computer Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Summer 2026

Doctoral Committee:

Advisor: Biswajit Ray

Sudeep Pasricha
Jie Rockey Luo
Indrajit Ray

ABSTRACT

SECURITY VULNERABILITIES IN 3D NAND FLASH MEMORY: CHALLENGES IN DATA SANITIZATION AND REVERSE ENGINEERING

Non-volatile flash memory, the fundamental component of solid-state storage devices, provides compact, high-capacity, and low-power storage across a wide range of applications, including consumer electronics, automotive, military, industrial, healthcare, and enterprise systems. The global flash memory market currently exceeds $60 billion and is projected to surpass $80 billion by 2030. However, flash memory poses fundamental challenges for secure data management due to its unique architecture. Data can be written at the page level but erased at the block level, preventing instantaneous in-place updates. These architectural limitations make reliable data deletion difficult and can lead to unintended data remanence.

This dissertation investigates security vulnerabilities, data sanitization limitations, and reverse engineering techniques in 3D NAND flash memory. First, we analyze the existing overwrite-based sanitization method and show that it can introduce unintended disturbances to neighboring cells, potentially affecting stored data. To address this issue, we propose the PULSE technique to mitigate overwrite-induced effects while ensuring reliable sanitization. Next, vulnerabilities of the block erase operation are studied. We show that data can persist even after a block erase operation, which is traditionally considered to fully remove stored information, and we develop a method to recover such residual data. Finally, reverse engineering techniques are explored by exploiting the physical organization and behavior of memory cells. We demonstrate that internal scrambling keys and logical encoding schemes can be reconstructed, thereby enabling the recovery of chip-level information that is inaccessible to normal users. These findings reveal fundamental security weaknesses in 3D NAND flash memory and highlight the need for stronger, security-aware designs in future storage systems.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to all those who have supported me throughout my Ph.D. journey. Without their guidance, encouragement, and patience, this work would not have been possible.

First and foremost, I am profoundly grateful to my advisor, Professor Biswajit Ray, for his invaluable insight, expertise, and mentorship that guided me through the challenges of this research. I would also like to extend my sincere thanks to Professor Sudeep Pasricha for his innovative ideas and thoughtful suggestions, which significantly enriched my work. I am equally thankful to my committee members, Professor Indrajit Ray and Professor Jie Rockey Luo, for their time, constructive feedback, and thought-provoking questions during my examinations. Special thanks to Professor Aleksandar Milenković, Chair and Professor of Electrical and Computer Engineering at the University of Alabama in Huntsville, for his guidance and encouragement throughout my academic and professional development.

I would like to acknowledge the dedicated staff of the Electrical and Computer Engineering Department at Colorado State University, especially Katya Stewart-Sweeney, whose assistance throughout the Ph.D. program was indispensable.

I am also thankful to my colleagues at the Reliable and Assured Microelectronics (RAM) Laboratory for their collaboration, encouragement, and support.

This thesis was generously supported by the National Science Foundation under Grants 2403540 and 2317563. I gratefully acknowledge this support, which provided essential research assistantship funding for this work.

DEDICATION

I would like to dedicate this thesis to my family and friends.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

3D NAND flash memory is a non-volatile storage technology that retains data without requiring power. It is widely used in modern electronic devices, such as smartphones, USB drives, SD cards, and solid-state drives (SSDs), where sensitive data is routinely stored [4–6]. Driven by the rapid growth of data centers supporting artificial intelligence workloads, demand for NAND flash memory continues to rise. This growth is fueled by the need for faster data access, higher data processing capabilities, improved power efficiency, and enhanced reliability in large-scale systems. The NAND flash market is projected to grow steadily, reaching an estimated $58.69 billion in 2026, up from $55.73 billion in 2025, and further expanding to $76.03 billion by 2031 [7, 8]. In addition, the rise of content creation, high-resolution gaming, and high-resolution media is driving the demand for higher-capacity storage in consumer devices, further accelerating the adoption of advanced commercial SSDs with larger capacities and faster interfaces [9]. Despite these advantages, the increasing density and complexity of NAND flash memory introduce new challenges in reliability, security, and data management [4–6, 10, 11]. In particular, ensuring secure data deletion and preventing unintended data leakage have become critical concerns in modern storage systems. To address these challenges, this thesis focuses on secure data sanitization methods and investigates security vulnerabilities in NAND flash memory through reverse-engineering techniques.

1.1 The Problem: Poor Data Sanitization Practices

With NAND flash memory widely deployed in modern electronic devices, secure and instant data sanitization has become essential for protecting user privacy. The Data Protection Act (DPA) 2018 [12, 13] mandates that deleted data must be irrecoverable by any means. However, standard sanitization methods are often insufficient [14–18], leaving sensitive information vulnerable to recovery. For example, a report by Blancco Technology Group found that 42% of used SSDs sold on eBay still contained recoverable data despite being labeled as "sanitized" [19, 20]. Similarly,

Robins et al. [21] showed that used flash devices often contain recoverable data from previous owners, and a large-scale forensic analysis of 614 supposedly new USB flash drives found that more than 12% retained recoverable user information [22, 23]. These findings indicate widespread chip reuse and poor sanitization practices, creating risks of personal and corporate data leakage. Therefore, effective and reliable sanitization techniques and careful handling of storage devices are essential to ensure data security throughout their lifecycle.

1.1.1 The Architectural Constraint

A fundamental challenge arises from the mismatch between logical data deletion and the physical constraints of NAND flash memory. Although data is accessed and updated at the page level, NAND flash is physically organized into blocks, each consisting of hundreds of pages. While pages are the unit of read and program operations, erasure is only possible at the block level. Furthermore, due to the erase-before-write constraint, a page cannot be updated unless its entire block is first erased. Because block erasure requires migrating all valid pages within the block elsewhere, in-place updates are impractical. Instead, updated data is written to a new location, while the old page is simply "unlinked" or marked invalid. However, the invalidated data physically remains in the medium, posing a privacy risk.

To bridge this mismatch between logical and physical operations, flash-based storage systems rely on the Flash Translation Layer (FTL), which manages address mapping, wear leveling, garbage collection, and error correction. In general, invalid data is removed during garbage collection through a block erase operation; however, this process is known to introduce significant performance overhead due to valid data migration and long erase latency [24–26]. This creates a key limitation for instant sanitization and accelerates device wear, reducing overall lifetime and usable capacity. Consequently, the FTL delays garbage collection, allowing invalid data to persist in flash memory long after deletion and delaying effective sanitization.

This page and block asymmetry gives rise to two fundamentally different sanitization strategies, each operating at a different granularity and through an opposite physical mechanism. *Block erase*

sanitization acts at the block level and removes stored charge from every cell in the block, restoring the array to a uniform erased state (logical '1'). On the other hand, *page-level overwrite sanitization* operates at the page level and uses an overwrite programming operation to inject charge into the cells of a single page, driving its content to a fixed pattern (typically all-zero) without erasing the surrounding block. The two techniques, therefore, differ not only in granularity but also in direction: block erase removes charge to reset cells, whereas page-level overwrite adds charge to overwrite cells. Both are widely assumed to provide secure deletion in current flash-based storage systems; however, each approach carries distinct limitations and security vulnerabilities that have not been thoroughly examined in modern 3D NAND flash memory.

1.1.2 Block Erase Sanitization Vulnerabilities

As NAND flash continues to scale in density and complexity, achieving efficient, reliable, and secure data sanitization remains an open and pressing challenge. In current flash-based storage systems, block erase operations are assumed to provide the strongest form of logical sanitization by removing stored charge from all cells within a block and restoring the array to a uniform erased state (logical '1'). As a result, they are commonly relied upon for secure data removal prior to device reuse, resale, or disposal, despite their significant latency, energy overhead, and endurance wear. However, the security guarantees of block erase operations have not been thoroughly examined in modern 3D NAND flash technologies. This is particularly concerning given that secure sanitization is essential for protecting user privacy in increasingly data-intensive and cloud-connected environments, where information frequently moves across organizational boundaries and storage platforms. Although encryption and access control can mitigate unauthorized access during use, improperly sanitized flash memory remains vulnerable to post-decommissioning forensic recovery. Prior studies [19–23] have demonstrated that used devices can still contain recoverable user data, indicating widespread sanitization failures and potential chip reuse risks. These findings highlight a fundamental limitation that block erase alone does not guarantee complete data removal in modern 3D NAND flash memory. Motivated by this gap, this thesis provides the first experimen-

tal evidence that block erase is insufficient for secure sanitization in 3D NAND flash and further demonstrates a data recovery technique that exposes residual information from erased flash blocks.

1.1.3 Page-Level Overwrite Sanitization Challenges

To overcome the latency, endurance, and valid-page migration costs of block-level erasure, a page-level overwrite sanitization technique has been proposed to enable fine-grained, low-latency sanitization. Instead of erasing an entire block, this technique performs an overwrite programming operation that writes a fixed pattern (typically all-zero) directly over the target page, destroying the original data without disturbing the block-level erase cycle. Such page-level overwrite sanitization has been demonstrated in single-level cell (SLC) storage devices [27] and multi-level cell (MLC) devices [28]. While this approach avoids the high cost of block erasure, it introduces new reliability concerns. Specifically, because overwrite is implemented as a program operation, it can induce unintended program-disturb effects on neighboring cells, potentially corrupting valid data stored elsewhere in the block. Moreover, as modern storage devices increasingly adopt higher-density triple-level cell (TLC) technology, which is more susceptible to such disturbances, the effectiveness of overwrite-based sanitization in scaled 3D NAND remains an open question. Therefore, a systematic evaluation of page-level overwrite sanitization in 3D NAND flash memory under both SLC and TLC configurations is needed, along with the investigation of potential mitigation techniques to reduce overwrite-induced disturbances.

1.2 The Problem: Reverse Engineering

Although NAND flash memory is widely used in modern storage systems, it contains internal information and device-specific behaviors that are not disclosed to end users through datasheets. This lack of transparency creates difficulties for system-level innovation.

1.2.1 Descrambling an On-Chip Randomization Key Extraction

In NAND flash memory, data scramblers randomize stored bit patterns prior to programming to address several reliability concerns [29,30]. First, randomization prevents uniform data patterns from forcing many cells into the same threshold voltage (V_{th}) distribution, which would otherwise cause uneven electric field distributions and increase susceptibility to program- and read-disturb effects, while also promoting more uniform V_{th} distributions across the array [29–31]. Second, in charge-trap (CT) 3D NAND devices, lateral charge migration is a primary driver of retention degradation [32–34]; randomizing stored data balances local electric fields and reduces pattern-dependent retention loss [32, 33, 35]. Third, randomization reduces repeated programming of identical patterns at the same physical locations, complementing controller-level wear-leveling and improving endurance [29, 30]. Finally, since error-correcting codes (ECCs) perform best with randomly distributed bit errors, scrambling disrupts repetitive patterns (e.g., 0xFF or 0x00) and produces statistically random ECC inputs, enhancing error correction efficiency [36, 37].

The data scrambler is implemented either in the NAND chip or the controller and is typically based on a linear feedback shift register (LFSR) [5, 29]. Before programming, host data is XORed with a pseudo-random sequence (the scrambling key) to generate randomized data patterns. This key is typically dependent on the physical page address. During read operations, the same sequence is regenerated using the corresponding seed and XORed with the stored data to recover the original information, ensuring deterministic yet reversible data transformation.

Although widely used, the scrambling mechanism and its associated keys are proprietary and not disclosed in device documentation. As a result, reverse engineering of on-chip scrambling keys has become an important problem in NAND flash security and system analysis. Accurate extraction of these keys enables deeper understanding of internal data transformations and supports several practical applications. For instance, recent studies have shown that instant sanitization techniques based on page-level overwrite operations rely on knowledge of internal scrambling behavior [18, 28, 38, 39]. Moreover, recovering scrambling mechanisms can significantly enhance forensic analysis by enabling reconstruction of raw data from memory dumps [40, 41]. In addition,

insights into scrambling structures can inform the design of more efficient error correction and data management strategies, ultimately improving overall system robustness.

1.2.2 Logical Encoding of Threshold Voltage (V_{th}) Distributions Extraction

To increase storage density, modern NAND flash memory employs logical scaling, enabling each memory cell to store multiple bits through precise control of its V_{th} distribution. For example, MLC memory stores two bits per cell using four V_{th} levels, while TLC and quad-level cell (QLC) technologies store three and four bits per cell using eight and sixteen levels, respectively [42, 43]. However, this increase in density comes at the cost of reduced reliability due to the shrinking voltage margin between adjacent V_{th} states [42, 44, 45].

By storing multiple bits per cell, each NAND flash cell can be programmed into one of several distinct V_{th} levels. The erased state corresponds to the lowest amount of stored charge, while higher programmed states correspond to increasing charge levels. Each V_{th} level represents a unique combination of bit values. For example, in TLC memory, three bits, referred to as the most significant bit (MSB), center significant bit (CSB), and least significant bit (LSB), are encoded within a single cell [46]. To read the stored data, the controller applies multiple read-reference voltages (V_{ref}) to distinguish between the different V_{th} levels. For example, in TLC memory, seven reference voltages are required to separate the eight levels and accurately determine the stored bit values. Importantly, the mapping between V_{th} levels and their corresponding bit values is not standardized. Different NAND vendors adopt proprietary encoding schemes, meaning that the same physical V_{th} level may correspond to different bit combinations across devices. These encoding schemes are carefully designed to balance multiple factors, including reliability, cell-to-cell interference, read latency, programming efficiency, and device endurance, while remaining compatible with system-level mechanisms such as error correction codes (ECC) and data scrambling.

Understanding the logical organization of V_{th} states is essential for improving system reliability and data management in NAND flash memory. Certain programmed states, particularly those at higher voltage levels, are more susceptible to charge loss, resulting in non-uniform error distribu-

tions across logical pages [4,47,48]. Identifying device-specific V_{th} encoding can therefore enable more effective error mitigation strategies, such as adaptive read-reference schemes and optimized ECC design [35], ultimately improving device endurance and reliability [49]. Moreover, recent studies have shown that knowledge of V_{th} encoding can facilitate instant sanitization of MLC and TLC pages [28, 50], as well as enhance forensic data recovery from NAND flash devices [41]. Therefore, developing a systematic method to extract the logical encoding of V_{th} states in commercial NAND flash devices is critical for both system designers seeking to improve reliability and performance, and forensic analysts aiming to gain deeper insight into device behavior.

1.3 Thesis Organization

This thesis consists of ten chapters: this introductory chapter, a background chapter, an experimental setup chapter, four chapters on data sanitization vulnerabilities, two chapters on reverse engineering, and a final chapter on conclusions and future work. Chapter 2 presents the fundamentals of NAND flash and is organized into three parts. The first part provides an overview of device structures, starting from the system level down to individual NAND flash cells, and introduces and compares floating-gate (FG) and CT NAND technologies. The second part examines NAND flash reliability, including data retention loss mechanisms and the molecular structure of oxide-layer bonds that contribute to data loss. The final part surveys data sanitization techniques in NAND flash and highlights the limitations of standard block erase operations.

Chapter 3 presents the experimental setup, beginning with a prototype board using wire connections through a breadboard and progressing to a printed circuit board (PCB) design. Since the custom-built hardware is compliant with the Open NAND Flash Interface (ONFI) standard, fundamental NAND flash operations, such as block erase, page program, and page read, are described, along with their corresponding interface commands.

Chapters 4 through 7 address issues related to data sanitization vulnerabilities. Chapter 4 investigates page-level overwrite sanitization in 3D NAND flash operating in SLC and TLC configurations. Root-cause analysis reveals that data corruption in adjacent pages is caused by charge

gain in cells intended to remain in the erased state. The chapter concludes by proposing PULSE (Partial Update-based Low-disturbance Sanitization Engine) as a mitigation technique. Chapter 5 examines block erase sanitization vulnerabilities in 3D CT SLC NAND flash, demonstrating that residual information persists in erased blocks due to lateral charge migration and residual hole accumulation. Several mitigation strategies are also discussed and evaluated in this chapter. Chapter 6 extends this analysis to 3D FG SLC NAND flash memory. Due to differences in device structures, previously stored data can become imprinted, enabling an attacker to recover the data across multiple attempts. Further device-level analysis reveals that the leakage is attributed to variations in the oxide-defect-state density. Despite this difference in the underlying mechanism, the same recovery methodology presented in the previous chapter can be applied. Building on this approach, Chapter 7 extends the study to MLC storage devices. Recovery methodologies are proposed and demonstrated through experimental results.

Chapters 8 and 9 present security vulnerabilities uncovered through reverse engineering. Chapter 8 proposes a method for recovering on-chip randomization keys by controlling the page programming sequence, demonstrating successful key extraction across both 2D and 3D NAND architectures. Chapter 9 introduces a technique for decoding logical-to-physical state mappings using controlled partial-program operations. Experimental validation across multiple vendors, cell technologies, and storage densities demonstrates that the technique is broadly applicable and effective. All techniques presented in this thesis rely solely on user-mode commands, without requiring hardware modifications or manufacturer support. Finally, Chapter 10 summarizes the thesis and outlines directions for future research.

Chapter 2

Background

This chapter provides the background necessary for understanding the technical foundations and security challenges discussed in this thesis. It begins with an overview of NAND flash device architecture, moving from the system level down to the cell level. It also discusses reliability and data retention characteristics, including the physical mechanisms that contribute to retention loss. Finally, it introduces data sanitization in NAND flash and explains the limitations of common sanitization techniques, including page-level overwrite and block erase.

2.1 NAND Flash Device Architecture

2.1.1 Fundamentals of NAND Flash-Based Storage Systems

NAND flash-based storage systems, such as solid-state drives (SSDs), consist of a controller and one or more flash memory chips, as shown in Figure 2.1(a). While the NAND flash memory chips serve as the storage medium, the flash controller, called the FTL, acts as the brain of the system, coordinating data movement between the host and the NAND array, as shown in Figure 2.1(b). The primary responsibilities of the FTL include mapping logical addresses to physical locations, performing wear leveling, conducting garbage collection, managing bad blocks, and applying ECC to ensure data integrity, optimize device performance, and extend longevity [51]. In addition, the FTL ensures that data received from the host is properly stored in the flash memory array and can be reliably retrieved when requested by translating host-level commands into basic NAND operations such as read, program, and erase. Together, these functions ensure the performance, reliability, endurance, and lifetime of NAND-based storage systems.

2.1.2 Organization of 3D NAND Flash Chip

A single NAND flash memory chip contains one or more dies, each consisting of peripheral control circuitry and one or more planes [52]. The peripheral circuitry includes page buffers,

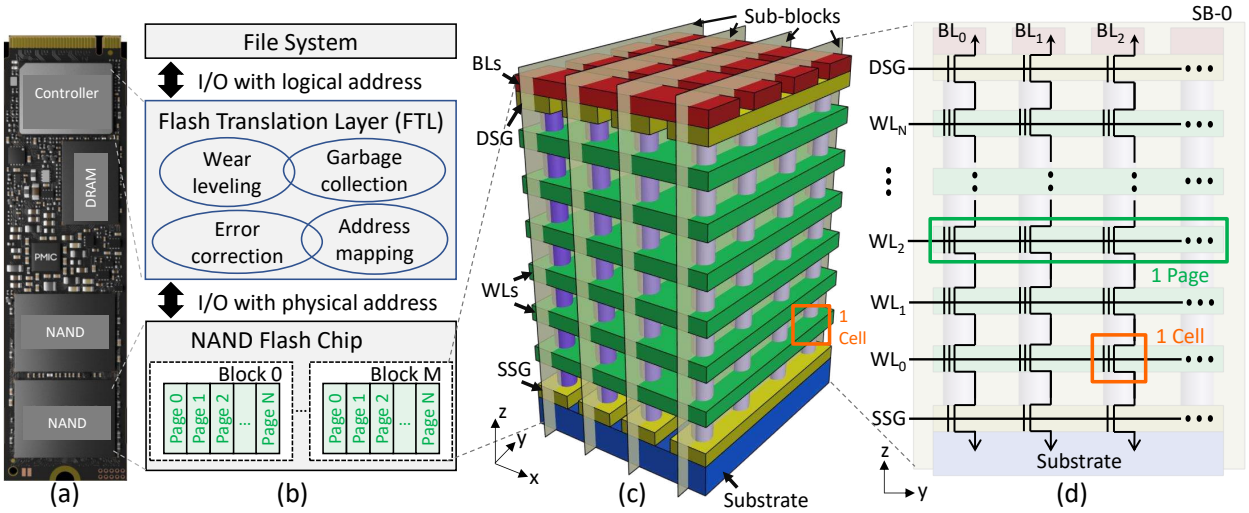


Figure 2.1: (a) Overview of NAND flash-based solid-state drives (SSDs). (b) System-level view of NAND flash–based storage. (c) Physical structure of a single 3D NAND flash block. (d) Circuit diagram of a single 3D NAND sub-block.

row decoders, charge pumps, and sensing circuitry required for memory operations. Each plane contains multiple memory blocks that share the same set of bitlines (BLs), and each block contains multiple flash pages. Pages are the smallest programmable and readable units, while blocks are the smallest erasable units. Each flash page contains thousands of flash memory cells connected in series to form NAND strings, improving storage density and reducing cost per bit. Figure 2.1(c) shows the physical structure of a single 3D NAND flash block, where the green box represents one flash page and the orange square represents a single 3D NAND flash cell.

Sub-block Organization

To achieve high bit density, 3D NAND flash is designed to vertically stack multiple layers of flash cells on top of each other. In Figure 2.1(c), the green planes represent word lines (WLs), or stack layers [53]. In general, a 3D NAND block is physically divided into several sub-blocks to optimize die-area utilization, since all physical sub-blocks share the same WLs. The purple poly-silicon pillars along the same y-axis form one sub-block. All pillars within a sub-block connect to the substrate when the corresponding source select gates (SSGs) are active and to individual BLs when the corresponding drain select gates (DSGs) are active. Sub-blocks are identical in structure

and organization, and all sub-blocks within a block share the same WLs and BLs. Select transistors are used to address a particular sub-block during page read and program operations; however, all sub-blocks within a block are simultaneously erased by a block erase operation.

At the circuit level (Figure 2.1(d)), each sub-block contains multiple flash pages, where the page size is typically 16 kB in state-of-the-art 3D NAND chips. All cells in the same row share the same WL, while all cells in a vertical column connect together through a metal BL. With each new generation of 3D NAND, the physical block size, measured in the number of pages per block, has grown substantially, giving rise to the so-called *big block problem*.

Big Block Problem

The big block problem occurs during garbage collection, when pages containing valid data (live pages) must be copied from the victim block, the block selected for erasing, to another free block before the erase can proceed. Larger blocks contain a larger number of live pages, so the number of pages that must be migrated grows accordingly, increasing garbage collection latency and degrading overall flash-memory performance. In addition to performance loss, the big block problem exacerbates write amplification, the ratio of the amount of data actually written to the flash media (including page writes caused by garbage collection) to the amount of data written by the host, which negatively affects the lifetime and effective capacity of the flash device.

2.1.3 3D NAND Flash Cell

A single 3D NAND flash block contains millions of individual flash cells, and each cell stores information by accumulating electric charge in the storage layer. Typically, commercial 3D NAND uses two storage elements, FG and CT cells, distinguished by their charge-storage materials [54–56]. In FG 3D NAND (Figure 2.2(a)), electrons are stored in an electrically isolated polysilicon FG, shown in yellow. Each cell has a separate storage node surrounded by oxide, which confines the charge. This isolation prevents charge from leaking between cells but makes each cell vulnerable to defects in the oxide layer. In contrast, CT 3D NAND (Figure 2.2(b)) stores charge in a silicon nitride ($\mathrm{Si_3N_4}$) charge-trapping layer. Unlike the discrete FG node, the CT layer is physi-

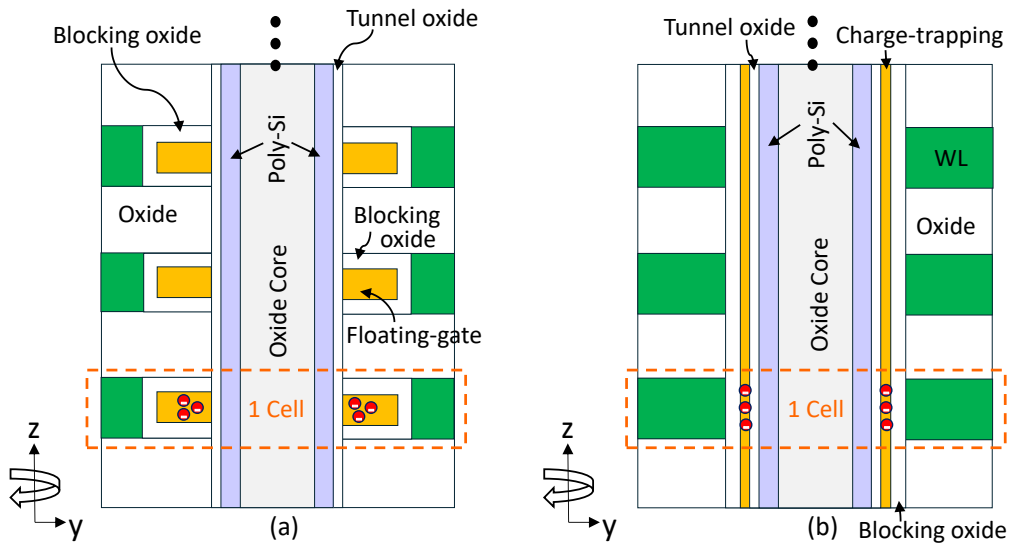


Figure 2.2: Schematics of a (a) floating-gate (FG) and (b) charge-trapping (CT) 3D NAND string.

cally continuous along the vertical stack, illustrated as a continuous yellow pipe in Figure 2.2(b). Charges are captured at localized trap sites within the nitride, enabling a thinner effective oxide thickness and improved scalability. However, the distributed trapping mechanism introduces different retention, lateral charge migration, and detrapping characteristics from those of FG devices. A single NAND flash cell with charge stored in the storage layer, highlighted by the orange box, is shown in Figure 2.2(a) and (b) for FG and CT NAND, respectively. As a result, the memory cell functions as a charge-based analog device with two logical states: an erased state (logic '1') when the storage layer holds no charge, and a programmed state (logic '0') when sufficient charge is stored.

2.1.4 V_{th} Distributions and Logical Encoding

A traditional flash cell stores one bit of data, known as SLC storage. SLC flash uses two distinct V_{th} states, erased and programmed, as illustrated in Figure 2.3(a). These states correspond to the absence or presence of charge in the storage layer of the NAND memory cell. A reference voltage, $V_{ref}^{L_1}$, is placed between the erased and programmed distributions to reliably distinguish between the two states. To increase storage capacity without significantly increasing process com-

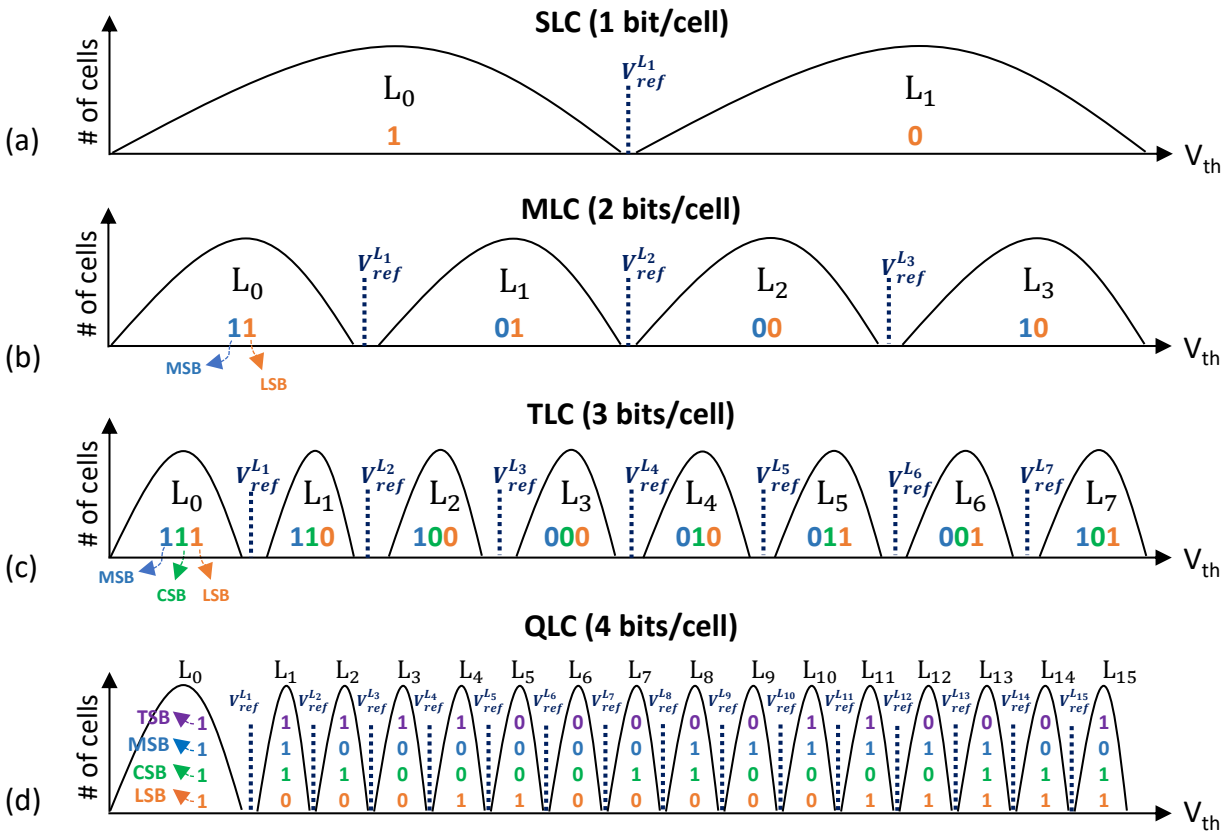


Figure 2.3: Threshold voltage distributions of (a) SLC, (b) MLC, (c) TLC, and (d) QLC NAND flash cell.

plexity, the same fabrication infrastructure used for SLC can be leveraged to enable multi-level storage [57]. Advances in controlling the amount of charge injected into the FG/CT layer and in sensing the charge during a read operation have enabled logical scaling, in which each cell stores multiple bits. Multi-level storage technologies, MLC, TLC, and QLC, store two, three, and four bits per cell, respectively, by introducing additional V_{th} levels, as illustrated in Figure 2.3(b)-(d). It is important to note that SLC, MLC, TLC, and QLC variants share approximately the same overall voltage window; packing more V_{th} states into this fixed window narrows the voltage margin between adjacent states, making them more susceptible to noise and disturb mechanisms [11] (e.g., retention drift, read disturb, and program coupling). While SLC therefore provides a wide V_{th} margin, enabling strong read stability and high endurance, multi-level storage requires precise placement of V_{th} levels to ensure accurate sensing and minimize overlap between adjacent

states [57]. More precise programming circuitry and verification algorithms are correspondingly required to mitigate the resulting performance degradation. Overall, the transition from SLC to multi-level storage technologies increases storage density without requiring substantial additional capital investment [57].

Because each multi-level V_{th} state encodes more than one logical bit, the choice of how bits are mapped to physical states has direct consequences for reliability. To limit error propagation across V_{th} boundaries, vendors commonly adopt *Gray encoding*, in which adjacent V_{th} states differ by exactly one bit. Under such an encoding, a V_{th} shift that crosses a single state boundary causes at most a single-bit error, regardless of which logical page the affected bit belongs to [58]. Formally, an n-bit Gray code defines a Hamiltonian path on the n-dimensional hypercube: vertices are the 2^n possible n-bit values, edges connect values at Hamming distance 1, and the Gray sequence visits every vertex exactly once. As n increases (SLC $\rightarrow$ MLC $\rightarrow$ TLC $\rightarrow$ QLC $\rightarrow$ Penta-Level Cell or PLC), the number of valid Gray assignments grows combinatorially, yielding a large family of possible vendor-specific mappings [59]. For example, TLC (n = 3) admits 144 possible Hamiltonian paths, of which 18 begin at the vertex '111,' corresponding to the erased state [60]. QLC (n = 4) admits 5,712 such assignments, and PLC (n = 5) admits over 5.8 billion. In practice, vendors select among these encodings to balance reliability, program-verify margins, coupling compensation, and controller decoding efficiency.

2.1.5 Basic NAND Operations

Data stored in NAND flash memory can be managed with three basic operations: program, read, and erase. A *program operation* transfers electrons from the substrate into the storage layer of the selected flash cell via Fowler-Nordheim (FN) tunneling, by applying a high voltage to the selected WL. This injection of charge increases the cell's V_{th}, setting it to logic '0' [61, 62]. To place cells accurately at their target states, modern NAND uses an Incremental Step-Pulse Programming (ISPP) scheme [48, 63–65]. ISPP applies a sequence of program pulses to the selected WL, beginning at a low bias and increasing in fixed amplitude steps (ΔV_{ISPP}) between successive

pulses. After each program pulse, a verify pulse checks whether the cell has reached its target V_{th}. If not, the next program pulse is applied at the incremented voltage, and the program-verify cycle repeats until the cell passes verification or the maximum program voltage is reached [64].

To *read* the stored data, a V_{ref} is applied to the selected WL to sense the V_{th} of the addressed flash cell [57, 62, 66, 67]. V_{ref} is placed in the valley between the two adjacent V_{th} distributions to minimize the read error rate. If the cell's V_{th} is below V_{ref}, the cell conducts, and the sense amplifier reads logic '1' (erased state); if its V_{th} is above V_{ref}, the cell does not conduct and the sense amplifier reads logic '0' (programmed state). All unselected WLs in the same NAND string are biased at a higher pass voltage ($V_{pass} > V_{ref}$) so that the corresponding cells remain on regardless of their stored state, allowing the addressed cell's conduction to be sensed at the BL.

The *erase operation* in NAND flash memory occurs at the physical block level. Erasing a block involves setting the substrate to a high erase voltage (> 20 V) and grounding the WLs, which removes charge from the memory cells via FN tunneling and decreases their V_{th} [62]. The erase operation thus resets every cell in the block to logic '1.' Block erase has substantially higher latency than page program and page read operations [62]. Because erase operates only at the block granularity, NAND flash cannot perform in-place updates of individual pages: when a page is updated, its new content must be written to a free, previously erased page, and the page containing the obsolete data is marked as invalid. When most pages in a block have been marked as invalid, the FTL invokes a block erase to reclaim the block for reuse. Before the erase, however, all valid pages in the block must be copied to a different physical block, and the larger the number of valid pages that must be migrated, the higher the latency overhead of garbage collection.

2.2 Reliability and Retention Characteristics

2.2.1 Endurance and Data Retention Loss Mechanisms

NAND flash exhibits limited endurance primarily due to tunnel oxide degradation caused by repeated program/erase (P/E) cycles [68]. During program and erase operations, charge is transferred between the channel and the storage layer via FN tunneling through the thin tunnel oxide.

Repeated electron injection and removal gradually degrade the tunnel oxide by generating defect states, as illustrated in Figure 2.4(a). These defect states can trap electrons in an unstable manner and eventually cause charge loss.

Retention Loss Mechanisms in FG NAND

In FG NAND flash memory, there are four fundamental data retention loss mechanisms: ① interface trap recovery, ② trap-assisted tunneling (TAT), ③ de-trapping, and ④ thermionic emission (TE) [69], as illustrated in Figure 2.4(a). Interface trap recovery occurs when charges trapped at or near the oxide–channel interface transport back into the channel. TAT occurs when a high density of oxide traps forms a percolation path, allowing charges in the storage layer to hop through

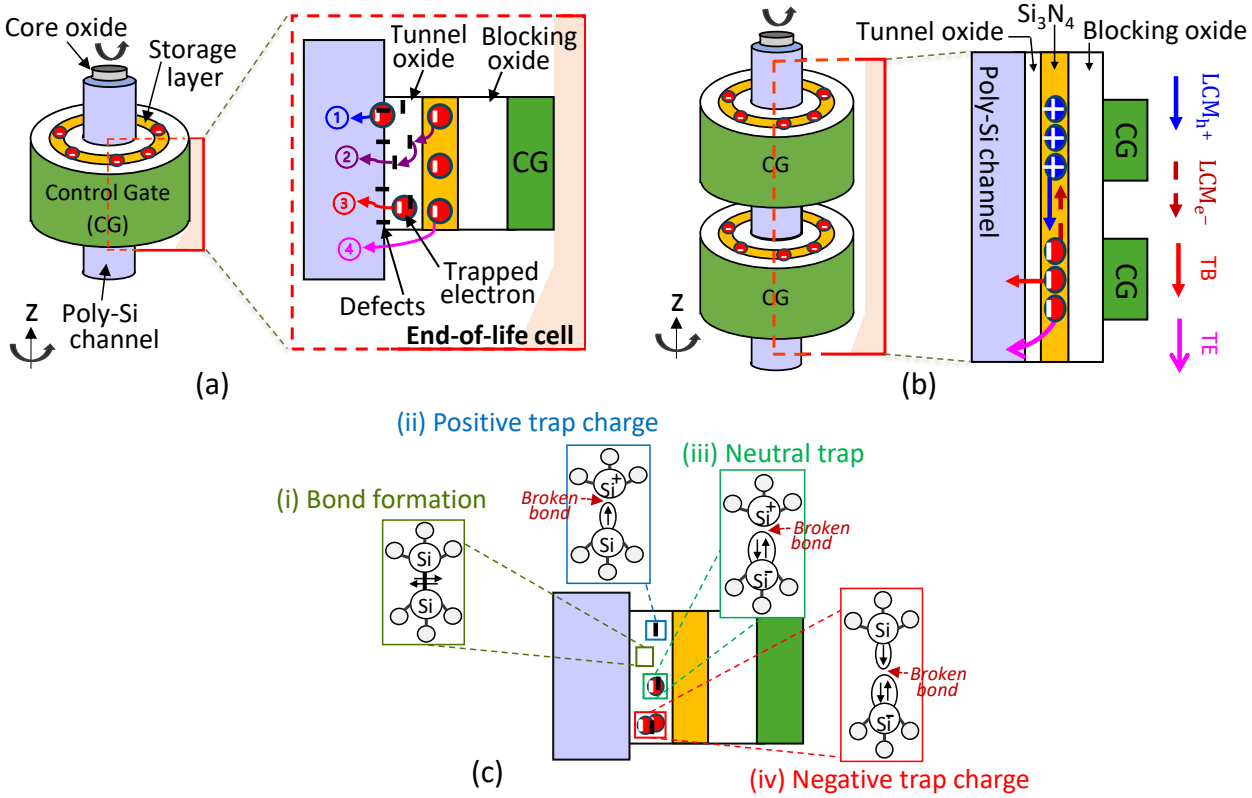


Figure 2.4: (a) Schematic of an end-of-life 3D FG NAND cell showing four data retention loss mechanisms: ① interface trap recovery, ② trap-assisted tunneling (TAT), ③ de-trapping, and ④ thermionic emission (TE). (b) Schematic of 3D CT NAND with trapped charge redistribution and charge loss mechanisms. (c) Types of defects in the tunnel oxide, depicting the molecular structure of the tunnel oxide in four states: (i) bond formation, (ii) positive trap charge state, (iii) neutral trap state, and (iv) negative trap charge state.

these traps to the channel. De-trapping occurs when a trapped charge de-traps from the oxide and returns to the channel. TE occurs when a trapped electron gains enough thermal energy to escape from the storage layer to the channel, but at room temperature, its contribution to charge loss is minimal [70, 71]. These charge-loss mechanisms eventually cause a downward shift in V_{th}. As the device undergoes more P/E cycles, additional defect states are created, making charge loss more pronounced.

Retention Loss Mechanisms in CT NAND

Figure 2.4(b) illustrates the fundamental charge-loss mechanisms in 3D CT NAND flash, including lateral migration of holes (LCM_{h^+}) and electrons (LCM_{e^-}), trap-to-band tunneling (TB), and TE. In 3D CT NAND, LCM_{h^+} and LCM_{e^-} represent major reliability challenges due to the continuous silicon nitride CT layer along the NAND string. Because the CT layer is physically continuous, stored electrons and holes can laterally migrate along the storage layer, leading to V_{th} shifts in the memory cells [70–73]. TB refers to the de-trapping of electrons from the CT layer back into the channel [70, 71], while TE occurs when trapped carriers acquire sufficient thermal energy to escape from the CT layer into the channel; however, this mechanism is generally weak at room temperature and therefore contributes minimally to charge loss compared to other mechanisms [70, 71]. On the other hand, TE becomes more relevant under elevated temperatures used during accelerated retention and recovery experiments.

Molecular Structure in Tunnel Oxide Layer

Figure 2.4(c) illustrates the molecular structures of the tunnel oxide layer under four different bonding states: (i) bond formation, (ii) positive trap charge state, (iii) neutral trap state, and (iv) negative trap charge state [74, 75]. As a flash cell approaches its end-of-life, repeated P/E cycling induces defect generation in the tunnel oxide due to continuous charge injection and removal from the storage layer [74, 76]. These defects originate from broken silicon–oxygen (Si–O) bonds. Immediately after bond breakage, the defect can exist as a positively charged trap, as shown in Figure 2.4(c)-(ii). When one electron is captured by the positively charged defect, it transitions

to a neutral trap state (Figure 2.4(c)-(iii)). If a second electron is captured, the defect becomes negatively charged, as illustrated in Figure 2.4(c)-(iv). The trapped charges in these defect states are weakly confined and can de-trap over time, contributing to charge loss and eventually data retention degradation. Under elevated temperature annealing, a neutral trap may recombine and restore the bond configuration, returning to the bond reformation state, as shown in Figure 2.4(c)-(i).

2.3 Data Sanitization and Security Implications

As NAND flash-based storage devices routinely store sensitive data, secure flash media sanitization prior to reuse, resale, or disposal is critical for protecting user privacy. To address this challenge, NIST SP 800-88 Rev. 1 [3] categorizes sanitization techniques for storage media into three types: *Clear*, *Purge*, and *Destroy*. It also emphasizes selecting the appropriate method based on data sensitivity, device capabilities, and the intended reuse or disposition of the device. Figure 2.5 summarizes the methods associated with each sanitization type. *Clear* employs logical techniques, such as block erasure, factory reset, or overwriting, to protect against simple and non-invasive recovery, providing a moderate level of data protection [77]. *Purge* uses advanced logical or physical techniques designed to resist laboratory-level recovery, such as cryptographic erasure and secure erase commands. It is intended for more confidential data [77]. Lastly, *Destroy* physically renders the media unusable through methods such as pulverization, incineration, and shredding, offering the highest level of security [77]. Although physical destruction guarantees data removal, it is often impractical for electronic storage media such as flash memory [3, 78, 79]. The particle size required for conventional grinding becomes prohibitively small with scaled flash memory technology, and the hardness of flash materials accelerates equipment wear [77]. Destructive methods are also slow, energy-intensive, and incompatible with scenarios where devices must be repurposed or recycled [3, 77–79].

Logical sanitization (Clear/Purge) techniques, implemented through standard read, write, and erase commands, are more practical and scalable. However, NAND flash memories pose unique

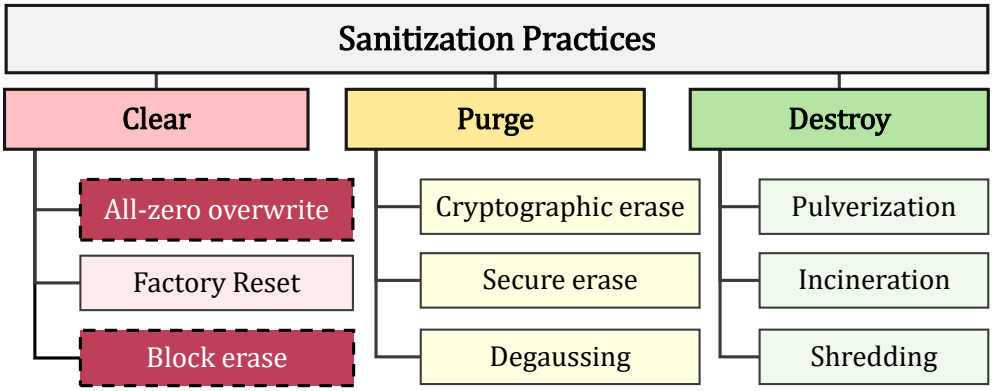


Figure 2.5: Sanitization categories [3].

challenges for secure logical sanitization due to their intrinsic structure and operation. Data is stored in NAND flash blocks, with programming performed at fine granularity (page level), but erasure is only possible at the block level. Furthermore, NAND flash devices follow an erase-before-write paradigm, which requires erasure prior to reprogramming. This constraint makes in-place updates impractical, allowing old data to persist even after being logically invalidated. Moreover, the FTL [80, 81] in NAND flash manages wear-leveling, garbage collection, and performance optimization, but it can delay secure deletion, allowing invalid data to persist in the NAND array long after a delete command is issued.

In this thesis, the term "sanitization" is used interchangeably to describe the process of removing data from a storage device such that previously stored information cannot be recovered [27, 82, 83]. This work focuses on the security implications of all-zero overwrite and block erase sanitization techniques, which fall under the *Clear* category. These techniques are important because they are widely adopted in commercial NAND-based storage devices used in consumer and enterprise applications.

2.3.1 All-Zero Overwrite Sanitization

To achieve page-level, near-instant data sanitization, Wei et al. [27] proposed a "data scrubbing" approach, later refined in subsequent works [18, 39, 84–87]. This method reprograms all cells within a flash page with a solid pattern, such as all-zero data, forcing them into a uniform

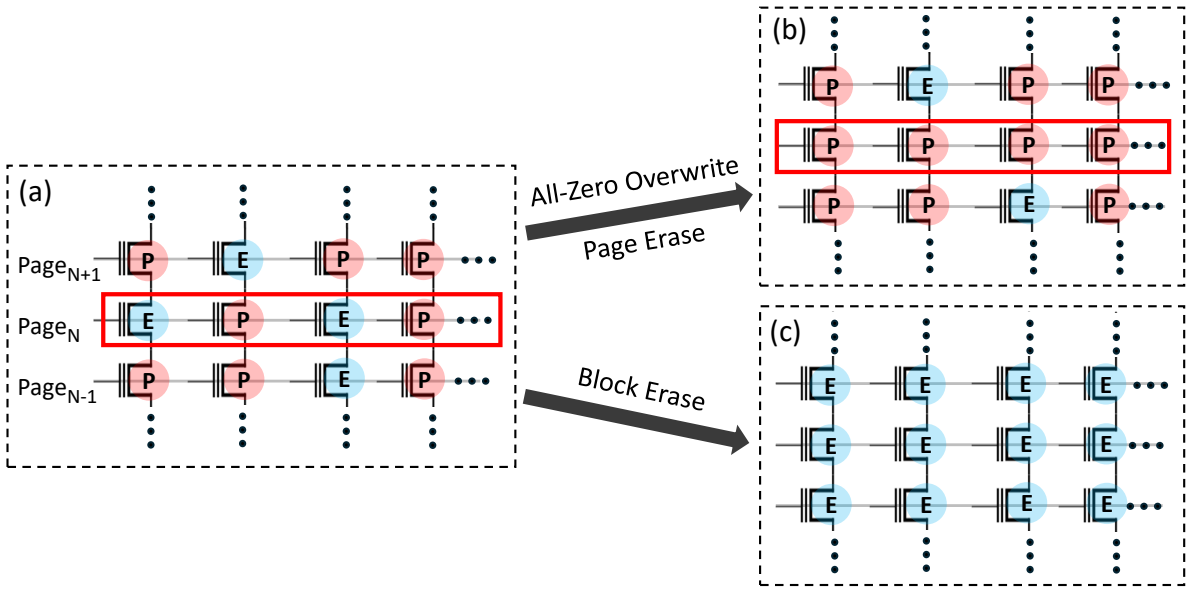


Figure 2.6: (a) A schematic of an SLC NAND block containing random data, where each row is a page. (b) all-zero overwrite page erase and (c) block erase sanitization. Cells in 'E' state represent logic 1 while cells in 'P' state represent logic 0.

programmed state and effectively overwriting the page contents [50]. As shown in Figure 2.6(a)-(b), the erased state corresponds to cells labeled 'E' (logic-1), while the programmed state corresponds to cells labeled 'P' (logic-0). This approach aligns with the program-only capability of NAND flash, where additional charge can be injected into cells without requiring a prior erase cycle. However, once a page is overwritten, it cannot be reprogrammed without an intervening block erase, limiting this technique to scenarios where one-time deletion suffices. Compared to block erase, all-zero overwrite offers significantly lower latency, as page program operations complete in microseconds versus milliseconds for block erase [62], and minimizes wear on surrounding pages.

Despite its efficiency, overwrite-based sanitization may be ineffective against advanced forensic techniques. Writing zeros does not guarantee the complete removal of previously stored data, as residual information may persist within the NAND array, enabling partial or full data recovery [86]. While this method has been shown to be effective for 2D NAND and SLC-based devices, its effectiveness in 3D NAND remains insufficiently explored, particularly given architectural complexities such as sub-block organization.

2.3.2 Block Erase Sanitization

Block erasure is the most fundamental sanitization technique for flash memory and is widely considered effective, as erased data is typically unrecoverable [14, 16, 24–26, 88]. This method removes data from all pages within a block by resetting every cell to a uniform erased state (logic-1), as demonstrated in Figure 2.6(c). In practice, block erase is primarily invoked by the FTL during garbage collection to reclaim space occupied by invalid data; however, its use for immediate sanitization is limited by performance constraints. Block erase operations incur significant overhead due to valid data migration [24–26] and exhibit higher latency compared to page program operations. Additionally, frequent block erase operations accelerate device wear, reducing both the endurance and lifespan of the flash memory. Consequently, modern flash controllers minimize the use of block erase operations, further limiting their suitability for rapid and secure data sanitization.

2.4 Conclusion

This chapter presented the background on NAND flash memory necessary for this thesis. We first introduced the architecture of 3D NAND flash memory. We then described FG and CT memory cells, how data is stored, and the basic program, read, and erase operations. Next, we discussed the physical mechanisms responsible for data retention loss. Both FG and CT NAND experience charge loss due to oxide defects generated during repeated P/E cycling, while CT NAND additionally suffers from lateral charge migration. These charge loss mechanisms accumulate over time and reduce retention reliability. Finally, we reviewed NAND flash sanitization methods, focusing on block erase as the standard Clear operation defined in NIST SP 800-88. Although block erase resets all cells to the erased state, it does not remove the underlying oxide defects caused by device wear. This raises the central question explored in this thesis: whether block erase is sufficient to completely eliminate traces of previously stored data in worn-out NAND flash devices.

Chapter 3

Experimental Setup

Throughout this thesis, a custom-designed test board was developed and used for experimental evaluation. The board consists of an FT2232H Mini Module from Future Technology Devices International (FTDI), connected to a workstation via Universal Serial Bus (USB), and a removable socket that holds a raw NAND flash memory chip. This setup enables basic flash operations, including page read, page program, and block erase, through host-side software that issues low-level Open NAND Flash Interface (ONFI) command sequences. ONFI is widely adopted as a standard interface for NAND flash memory and is expected to remain compatible with future devices [2,89]. Because ONFI is vendor-independent, the same hardware and software framework can be reused across different NAND flash technologies. Note that this design uses Single Data Rate (SDR), which employs an asynchronous interface; as a result, data is not latched on a clock edge, and the control signals are generated directly by the controller [2]. In contrast, Double Data Rate (DDR) interfaces can be used to achieve higher transfer speeds.

The software used in this work emulates basic FTL functionality, enabling direct raw bit error rate (RBER) measurements during page read operations and V_{th} measurements through read-offset operations. The custom-designed test board can bypass built-in ECC and higher-level storage management mechanisms, thereby exposing the intrinsic cell-level behavior required for the experiments. This chapter presents the hardware development process, from the initial prototype to the printed circuit board (PCB) implementation, along with the corresponding schematics and the basic ONFI-supported NAND flash operations required for successful interfacing.

3.1 Hardware Development

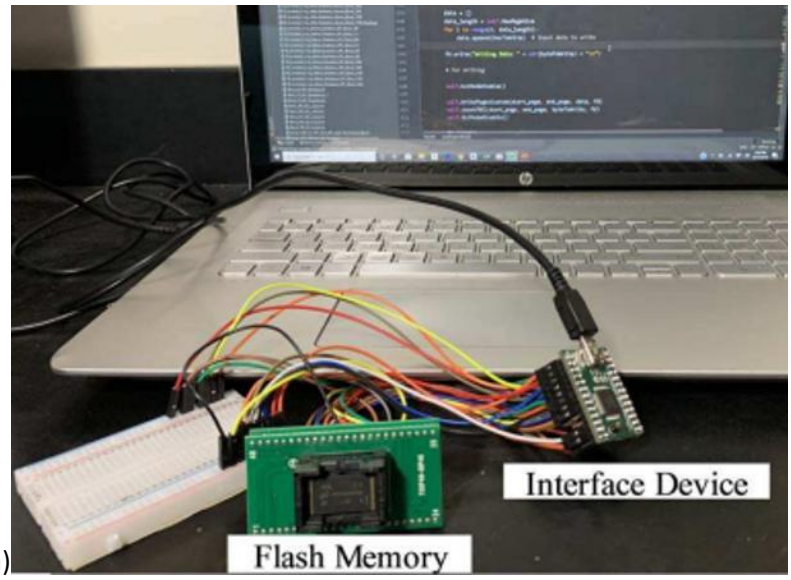
The test board evolved through two generations during this work: an initial breadboard-based prototype used for exploratory experiments, and a PCB implementation used for the measurements reported in subsequent chapters. Both generations share the same architectural concept and

core components. A host workstation communicates with an FT2232H Mini Module via USB, while the FT2232H drives the ONFI bus of a raw NAND flash memory chip, which is inserted into a removable 48-pin Thin Small Outline Package (TSOP-48) socket adapter. The removable socket allows different NAND flash chips to be tested without modifying the hardware, enabling compatibility across multiple devices and technology generations. Figure 3.1(a) shows the initial breadboard-based prototype, in which all signal connections were implemented using jumper wires. Although this setup enabled rapid prototyping and functional verification of the raw flash chip, the large number of wired interconnections introduced reliability issues, including unstable electrical contacts, increased signal noise, and reduced mechanical robustness. To address these limitations, a dedicated PCB was later designed and fabricated, as shown in Figure 3.1(b). The PCB implementation provides shorter and more stable signal paths, improved mechanical reliability, and more consistent experimental operation. As a result, the PCB design enables experimental evaluations beyond the conditions tolerable by the breadboard prototype, such as extreme high- and low-temperature testing, under which the breadboard setup was observed to fail at one point.

3.1.1 FT2232H Mini Module and NAND Socket

The FT2232H Mini Module provides various ways to interact with external devices through a USB connection [1, 90], as it integrates the entire USB protocol on a chip with no firmware required [91]. Bit-banging is a technique in which software directly controls digital I/O signals, rather than relying on dedicated hardware peripherals to perform the transmission; the bit-banging approach for driving NAND flash with the FT2232H was previously discussed by Sprites Mod [1, 92]. Although the FT2232H supports multiple operating modes, the *MCU Host Bus Emulation Mode* is particularly suitable for NAND flash interfacing [1]. In this mode, the FT2232H emulates an 8048/8051 MCU host bus, allowing software to issue commands such as 0x90 (read, 8-bit address) and 0x92 (write, 8-bit address) and to transfer data over the I/O lines [1]. The NAND flash adapter socket used in this work is a TSOP48. However, the setup can be extended to BGA 132/152 packages through a TSOP48-to-BGA adapter, with minor modifications to the connections.

(a)



(b)

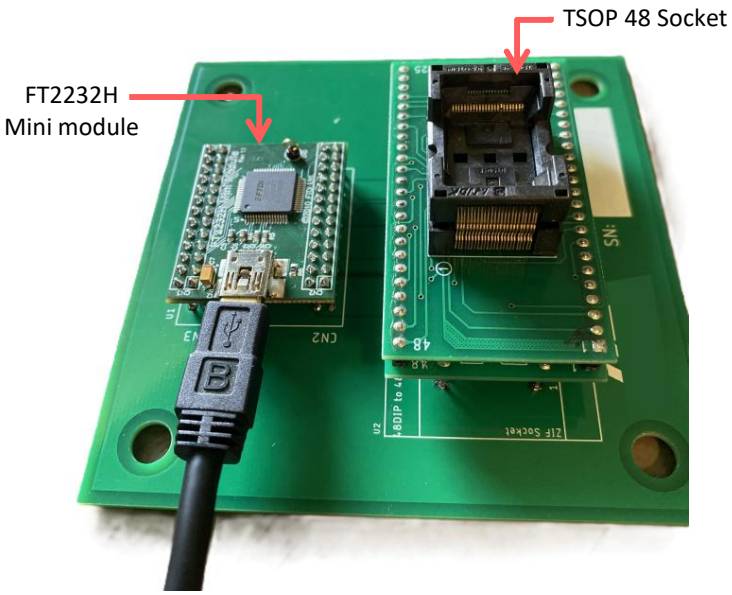


Figure 3.1: Experimental setup: (a) breadboard-based prototype and (b) PCB version.

3.1.2 NAND Flash Memory Pins and Connections to the FT2232H

Figure 3.2 shows a raw NAND chip in a TSOP48 package and its pin configuration, and Figure 3.3 shows the connections between the FT2232H mini module and the NAND chip. To access the memory, the connections include (i) the bidirectional data bus, (ii) the command, address, and write-protect latch lines, (iii) the read and write strobes, and (iv) the chip enable and ready/busy (R/B) status lines.

The bidirectional data bus: The eight FT2232H ADBUS pins (ADBUS0–ADBUS7) are connected directly to the NAND flash 8-bit bidirectional I/O bus (I/O0–I/O7). These pins carry commands, addresses, and data from the FT2232H into the chip [2], and carry page data back out during read operations [1].

The command, address, and write-protect latch lines: The CLE (Command Latch Enable), ALE (Address Latch Enable), and WP (Write Protect) pins control how the chip interprets the bytes presented on the I/O bus. CLE enables command latching, while ALE enables address latching. WP is active-low and must be held high to allow program and erase operations to complete. These three control lines are wired to ACBUS5-ACBUS7. Together, they let the NAND flash distinguish commands, addresses, and data on the shared bus.

The read and write strobes: The RE (Read Enable) and WE (Write Enable) strobes are driven by the FT2232H's BDBUS2 (RE) and BDBUS3 (WE) pins, respectively. Both are active-

Figure 3.2: Pin configuration of the NAND flash memory chip [1].

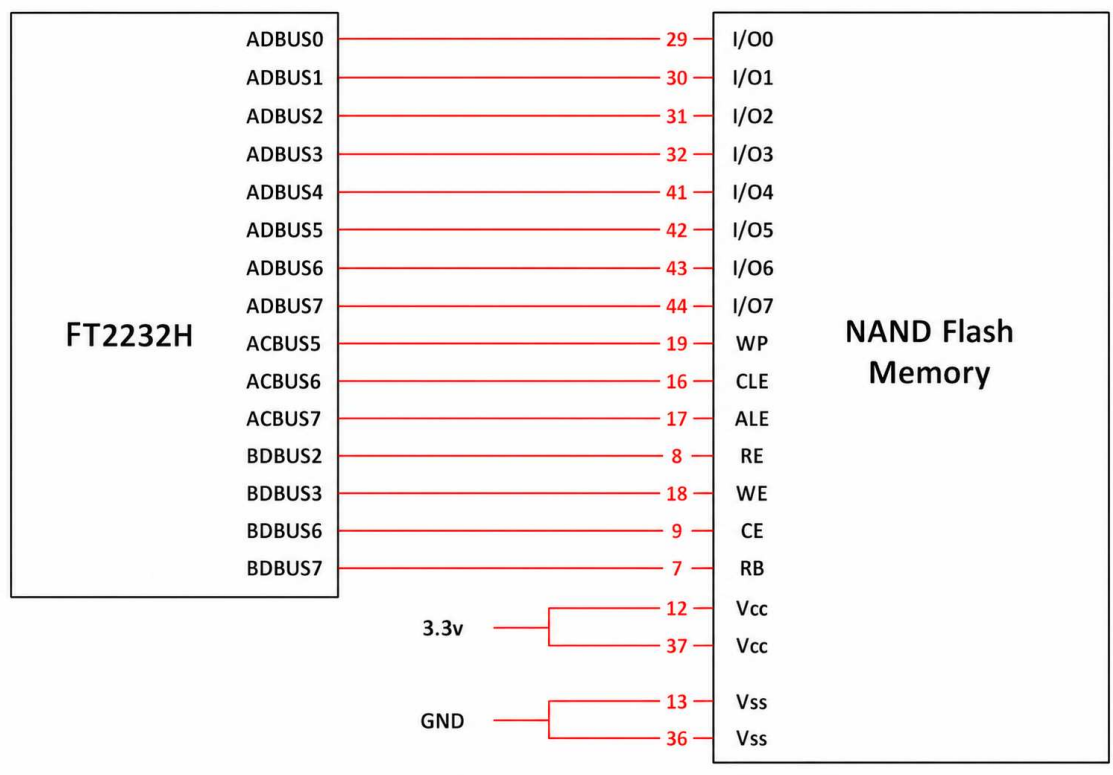


Figure 3.3: Connections between NAND flash chip and the FTDI mini module [1].

low signals. When the FT2232H chip is ready to read data, it sends a falling signal on the RE pin and lets the NAND flash chip send new data. On the other hand, when the WE output is rising, it means a new command, address, or data is available from the FT2232H chip and it lets the NAND flash chip fetch it [1, 2].

The chip enable and R/B status lines: CE (Chip Enable) and R/B are wired to BDBUS6 and BDBUS7, respectively. CE is normally held low to keep the chip selected [1, 2]. R/B is an open-drain status output from the flash chip: it goes low while the device is busy with an internal program, erase, or read operation, and returns high when the operation completes.

Finally, the power and ground pins on each side of the NAND flash chip must be connected. Based on the NAND sample given in Figure 3.2, the 3.3 V V_{CC} supply is connected to pins 12 and 37, while ground (GND or V_{SS}) is tied to pins 13 and 36.

3.2 NAND Flash Basic Operations

This section describes the basic NAND operations performed in the experiments throughout this thesis: Read ID, Read status, Page Read, Page Write, Block Erase, and Reset. Although more advanced commands exist for specific chipsets, these commands are sufficient for fundamental operations and are generally consistent across different models and manufacturers [1, 2]. These operations, including their command values and address cycles, are summarized in Table 3.1.

Table 3.1: Basic NAND Flash Command Set [1, 2].

Command	**1st Cycle**	**2nd Cycle**	**Address Cycles**	**Data Phase**	**Purpose**
Reset	FFh	–	–	–	Reset the chip to its default state
Read ID	90h	–	1	6 data-out	Return ONFI identifier code
Read Status	70h	–	–	1 data-out	Return status register
Page Read	00h	30h	5	N data-out	Read one page into the data register
Page Program	80h	10h	5	N data-in	Program one page from data register into array
Block Erase	60h	D0h	3	–	Erase one block

3.2.1 Read ID Operation

The Read ID operation determines whether a NAND flash chip is ONFI-compliant. After placing the raw NAND chip in the socket, the Read ID operation is issued by sending the command (0x90), followed by the address (0x20), and then reading a 6-byte data output. The returned bytes indicate ONFI compliance: the first four bytes are compared against the ASCII string “ONFI” as follows:

1. ONFI_Array[0] == 0x4F (‘O’)
2. ONFI_Array[1] == 0x4E (‘N’)
3. ONFI_Array[2] == 0x46 (‘F’)
4. ONFI_Array[3] == 0x49 (‘I’)

If all four bytes match, the chip is confirmed to be ONFI-compliant [90]. Note that for devices prior to ONFI 4.0, reading beyond four bytes yields indeterminate values [93]. The fifth byte is used to indicate that the device has powered up on the NV-DDR3 interface, and the sixth byte is reserved (returned as 0x00 on the device under test) [93].

3.2.2 Erase Operation

The erase operation removes all stored data within a block. In NAND flash, erase is performed at the block granularity and cannot be applied to individual pages [90, 93]. The Block Erase operation is issued by sending the command (0x60), followed by the row address of the block to be erased, and then the confirm command (0xD0), as illustrated in Listing 3.1. This operation resets every flash cell in the block to logic-1 by removing the stored charge from the storage layer via FN tunneling.

Listing 3.1: Block erase operation.

```
self.WriteProtect = False
self.sendCmd(0x60)              # Block erase setup command
self.sendAddr(addr, 3)          # 3 row address cycles (block address)
self.sendCmd(0xD0)              # Erase confirm command
self.WaitReady()                # Wait for R/B pin to return ready
err = self.Status()             # Read status register
self.WriteProtect = True
```

3.2.3 Program Operation

The program operation is performed at the page level [90]. The Page Program command transfers either a full page (16 KB) or a partial page of data, identified by the starting column address, into the page register. After that, the data is programmed into the NAND array at the specified row address [93]. The operation is issued by sending the command (0x80), followed by the address cycles (column and row), the input data to be stored, and the confirm command (0x10), as illustrated

in Listing 3.2. During programming, electrons are injected into the storage layer of the selected cells, raising the V_{th} to encode logic-0.

Listing 3.2: Page program operation.

```
self.WriteProtect = False
self.sendCmd(0x80)                # Serial data input command
self.sendAddr(addr, 5)            # 5 address cycles (2 column + 3 row)
self.writeData(data)              # Input data to be programmed
self.sendCmd(0x10)                # Program confirm command
self.WaitReady()                  # Wait for R/B pin to return ready
self.WriteProtect = True
```

3.2.4 Read Operation

The page is the smallest addressable unit for the read operation [90]. The Read function transfers a page of data, identified by the row address, into the page register, from which it can be read out starting at the given column address [93]. Reading beyond the end of a page returns indeterminate values to the host [93]. The Page Read operation is issued by sending the command (0x00), followed by the address cycles (column and row) and the confirm command (0x30), as illustrated in Listing 3.3. After the data transfer from the array to the page register completes (indicated by the R/B signal returning ready), the data is read out sequentially.

Listing 3.3: Page read operation.

```
self.sendCmd(0x00)                # Page read setup command
self.sendAddr(addr, 5)            # 5 address cycles (2 column + 3 row)
self.sendCmd(0x30)                # Read confirm command
self.WaitReady()                  # Wait for tR (array -> page register)
data = self.readFlashData(self.PageSize)   # Read out page register via RE
```

3.2.5 Read Status

The Read Status command (0x70), shown in Listing 3.4, is issued to retrieve the status of the most recently issued operation [93]. For non-multi-plane operations, a returned value of "0" indicates that the operation completed successfully [90].

Listing 3.4: Read status operation.

```
self.sendCmd(0x70)                          # Read status command
status = self.readFlashData(1)[0]           # Read 1-byte status register
```

3.3 Conclusion

This chapter presented the experimental platform used throughout this thesis. We described the development of the test board from a breadboard prototype to a custom PCB, which provides the stability needed for reliable and long-term experiments. We also explained how the FT2232H Mini Module communicates with raw NAND flash chip. We then introduced the basic NAND operations used in this work, including Read ID, Read Status, Page Read, Page Program, Block Erase, and Reset. Because the platform directly accesses the raw NAND array without interference from ECC or the FTL, we can measure the V_{th} of individual cells using a read-offset operation. This experimental platform forms the foundation for the measurements and analysis of post-erase data remanence and end-of-life retention behavior presented in the rest of this thesis.

Chapter 4

Page-Overwrite Sanitization

Page-level overwrite techniques provide fast and fine-grained data sanitization by avoiding the high latency and overhead of block erasure and have been explored in SLC and MLC NAND devices. However, this approach introduces reliability concerns, as overwrite operations can cause unintended disturbances in neighboring cells, potentially leading to data corruption. These challenges are further exacerbated in modern high-density NAND technologies, such as TLC, which are more sensitive to such disturbances. As a result, the effectiveness of overwrite-based sanitization in scaled 3D NAND remains uncertain. Therefore, this chapter systematically evaluates the page-based overwrite sanitization technique and develops mitigation techniques to reduce overwrite-induced disturbances.

4.1 Related works

Several data sanitization methods have been proposed for NAND flash storage systems in recent years to enhance data privacy. We summarize these approaches below.

Block erase: Block erasure is a NAND operation that removes data by discharging all cells in a flash block and is commonly used by FTL garbage collection to reclaim invalid data. However, its high latency, valid-data migration overhead, and contribution to device wear make it inefficient for instant sanitization, leading modern flash controllers to use it sparingly [24–26].

Logical data deletion: To avoid the performance overhead of block erasure, NAND flash controllers typically perform logical deletion by invalidating obsolete page addresses through the FTL mapping table [80, 94]. During page updates, new data is written to a fresh physical location while the old data remains physically stored in the flash memory. Because this invalidated data can still be recovered using advanced memory access techniques, logical deletion does not provide true data sanitization and poses a security risk.

Overwrite-based page sanitization: To enable page-level instant sanitization, prior work proposed "data scrubbing," which overwrites pages with all-zero data. Although effective in 2D NAND and SLC-based devices, overwrite-based sanitization has not been thoroughly evaluated for 3D NAND architectures with sub-block structures [18, 27, 39, 84–87].

Encryption-based data deletion: Encryption-based deletion achieves secure data removal by encrypting data with a cryptographic key and storing the encrypted data on the device, with deletion performed by securely erasing the key rather than the data itself [15–17, 95]. While this approach offers fast and storage-efficient deletion, it has several limitations. First, security depends entirely on the strength and secrecy of the encryption key; vulnerabilities such as weak random number generation or improper key management can compromise the entire scheme. 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 degrade system performance.

Evanesco: Kim et al. [96] introduced Evanesco, a hardware-supported technique for efficient data sanitization in modern flash-based storage systems. Evanesco sanitizes data by blocking access to invalidated data, either at the page or block level. While this approach is highly effective, the data stored in the flash medium is not physically removed, making it susceptible to future direct or indirect data retrieval attacks.

In summary, different sanitization techniques have been proposed in prior work, each offering different trade-offs among security, data reliability, performance overhead, and hardware implementation complexity. Table 4.1 summarizes and compares the key sanitization methods.

Table 4.1: Comparison of different sanitization methods.

Methods	Security guarantee	Data reliability	Performance overhead	Implementation complexity
Block erase	High	High	High	Low
Logical deletion	Low	High	Low	Low
Full overwrite	High	Low	Moderate	Low
Encryption	Moderate	High	Low	High
Evanesco	Moderate	High	Low	High
PULSE	High	High	Moderate	Low

4.2 Key Ideas

This chapter introduces novel contributions that advance the field of instant sanitization of NAND flash memory:

- **First experimental evaluation of overwrite sanitization in 3D NAND**: While prior studies on overwrite sanitization have primarily focused on 2D SLC NAND, this work is the first to experimentally investigate the overwrite sanitization process in modern 3D NAND devices. Through detailed V_{th} measurements and error characterization, we uncover previously unexplored post-sanitization disturbance effects on valid data. These effects are strongly influenced by the unique sub-block organization of 3D NAND and by layer-specific interactions within the vertical stack, leading to non-uniform susceptibility across layers. Our findings highlight critical reliability concerns that are absent or less pronounced in planar NAND, providing new insights for the design of secure and disturbance-resilient sanitization methods in 3D flash technologies.

- **First experimental evaluation of overwrite sanitization in TLC NAND**: Overwrite sanitization has been studied mainly in SLC and MLC NAND flash. This work is the first to demonstrate overwrite sanitization in TLC NAND, exposing the complications that arise when sanitizing all three logical pages (LSB, CSB, MSB) simultaneously.

- **Proposal of the PULSE sanitization technique**: We introduce PULSE, a novel sanitization method designed to reduce post-sanitization disturbance on valid data, while achieving effective sanitization of invalid data. Through experimental validation on multiple 3D NAND chips, we establish practical guidelines for integrating PULSE into future 3D NAND architectures. Our findings lay a strong foundation for implementing overwrite sanitization in next-generation 3D NAND SSDs, ensuring instant data deletion and mitigating data leakage, an essential security concern for SSDs.

4.2.1 Page-Overwrite Sanitization Induced Program Disturbance

While block erase is the conventional method for removing data in NAND flash, page-level overwrite has emerged as an alternative for fast sanitization because it avoids the latency and endurance costs associated with erasing an entire block. However, overwriting a page is not a side-effect-free operation as the high voltage required to program the target page can disturb the V_{th} of neighboring cells that still contain valid data. In 3D NAND, this disturbance is particularly significant due to the sub-block architecture. Figure 4.1 illustrates the biasing scheme during a page-overwrite operation in a 3D NAND block. The block is partitioned into m physical sub-blocks (SBs), each containing pages marked as either valid (green) or invalid (red). To sanitize an invalid page (page 2 of SB-0 in this example), an all-zero pattern is programmed into the page. This operation applies a high programming voltage ($V_{pgm} > 20$ V) to the selected WL, while unselected WLs in the same block are biased at a moderate pass voltage ($V_{pass} \approx 8$ V). This biasing can cause weak program disturbance effects, particularly in erased cells within the block. The highest risk is valid pages that share the same WL as the overwritten page (outlined with a red border in Figure 4.1), which are directly exposed to V_{pgm}. Additionally, pages in adjacent layers (outlined with a yellow border) may also experience moderate interference effects due to capacitive coupling. It is important to note that 2D NAND technology does not employ a sub-block architecture, limiting overwrite-sanitization disturbances to manageable interference effects (Figure 4.1, yellow) while avoiding direct disturbances (Figure 4.1, red).

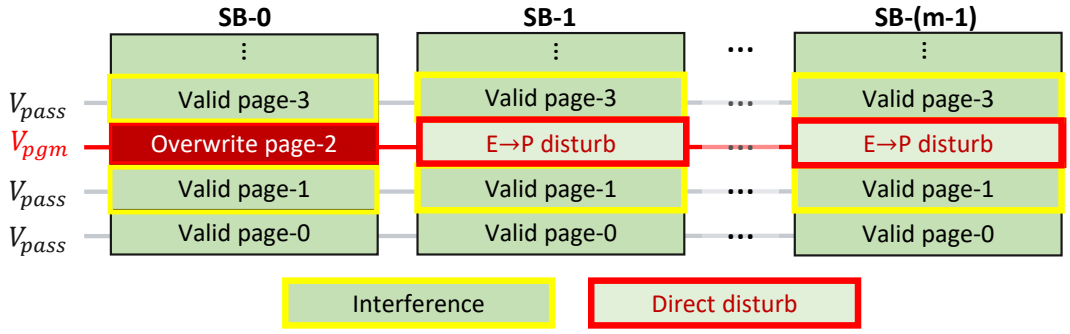


Figure 4.1: Biasing scheme during page-overwrite operation and corresponding disturbances on valid flash pages in 3D NAND sub-blocks.

4.3 Experimental Implementation

The experimental evaluation is conducted on multiple 3D NAND memory chips from three major NAND manufacturers, interfaced via a custom-designed test board, as shown in Figure 3.1(b). In this study, overwrite sanitization is implemented using the user-mode page write command with an all-zero data pattern. The raw NAND chips support out-of-order page write operations and do not include any internal data scramblers, data randomizer engines, or on-chip ECC engines. These functionalities are typically handled at the storage controller level, depending on application needs. For chips equipped with an on-chip ECC engine or data randomizer, standard user-mode commands may not suffice for implementing overwrite sanitization, and assistance from the chip manufacturer may be required.

Figure 4.2 illustrates the experimental flow used to evaluate the effects of overwrite sanitization on both valid and invalid pages within a 3D NAND block. The process begins by writing random data to all flash pages in the block, followed by a verification step using page reads to confirm a zero RBER. Next, overwrite sanitization is performed page-by-page. For systematic evaluation,

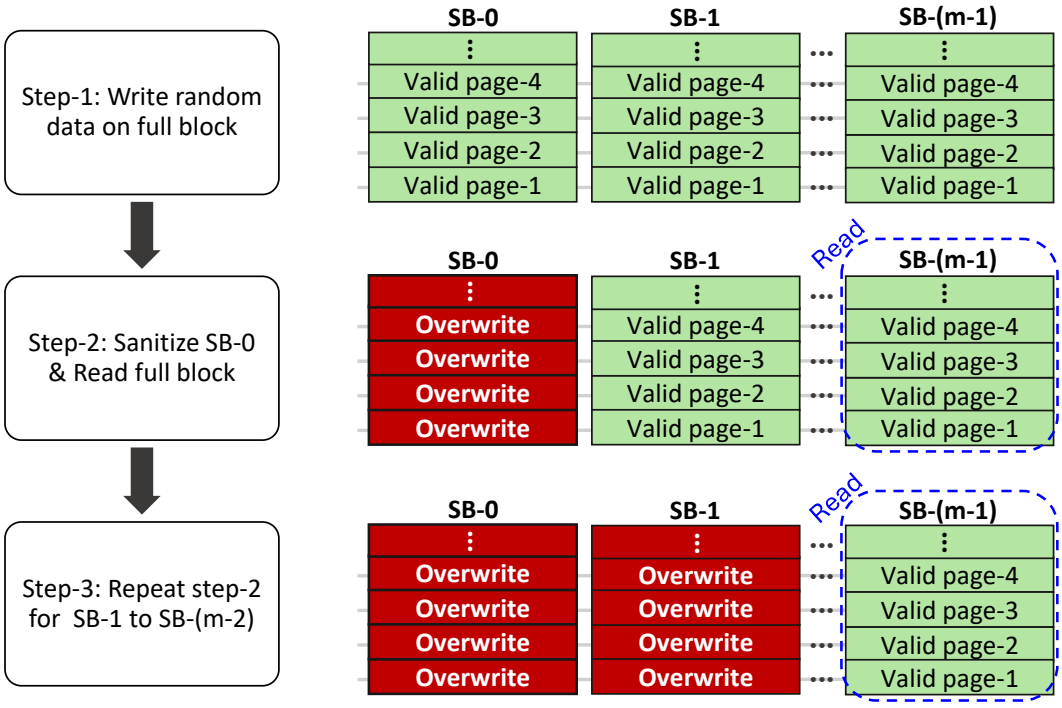


Figure 4.2: Experimental flow for evaluating sanitization efficiency and disturbance effects.

we first sanitize all flash pages within sub-block 0 (SB-0), then read all pages of the block. The sanitization effectiveness is quantified by comparing the sanitized page content to a predefined solid data pattern, typically an all-zero or all-one pattern, depending on the logical page type and the NAND operating mode. Sanitization efficiency of invalid pages ($\eta_{\text{san}}^{\text{inval}}$) is defined as:

$$\eta_{\text{san}}^{\text{inval}} = \frac{\text{Invalid}_{\text{data}} \ \text{XNOR} \ \text{Solid}_{\text{data-pattern}}}{\text{Page}_{\text{size}}} \times 100\%. \tag{4.1}$$

Here, the solid data pattern represents the expected post-sanitization content (e.g., all-zeros or all-ones). For example, in an SLC page, the solid data pattern is all-zeros. In contrast, for TLC storage where the highest V_{th} state is encoded as "101," the solid data pattern corresponds to all-ones for the MSB and LSB pages, while the CSB page retains an all-zero pattern. Note that a 100% sanitization efficiency indicates that the invalid page data has been fully converted into a solid data pattern, and hence, the data is logically irrecoverable.

Simultaneously, disturbances on valid pages are measured by calculating the RBER relative to their original data patterns. This process is repeated for each SB, except for the last one. The last SB typically experiences the worst-case disturbance due to cumulative interference from previous operations. To quantify this, we evaluate the RBER of valid pages in the last sub-block, which offers insight into the most severe reliability challenges posed by overwrite sanitization. In the following subsections, we present detailed experimental observations for three types of 3D NAND flash memory: SLC (Section 4.4-4.5), and TLC CT and FG-based (Section 4.6).

4.4 Evaluation in SLC FG 3D NAND

4.4.1 Evaluation of Sanitization Efficiency on Invalid Target Pages

Figure 4.3(a) visually demonstrates the effectiveness of the overwrite sanitization technique on an invalid page (or a target page) storing a Tesla image. After applying the sanitization operation, the image is transformed into a fully black, all-zero pattern, confirming the successful sanitization

of the invalid flash memory page. To validate this not only at the data level but also at the physical level, we measure the V_{th} distribution using the Read-offset command [97].

The plot in Figure 4.3(a) shows the measured V_{th} distribution from an SLC page, capturing both the erase and program states. The V_{th} values are expressed in arbitrary units (a.u.), as the absolute value of the reference voltage is not disclosed in the chip's datasheet. Due to the limited range of the Read-offset command, only the upper tail of the erase state and the lower tail of the program state are captured. The black curve represents the V_{th} distribution of the original data before sanitization, while the red curve shows the sanitized state. The vertical arrow marks V_{ref}, applied to the WL to sense cell states during read operations. The significant margin between V_{ref} and the erase/program distribution tails ensures a zero RBER. Importantly, the post-sanitization distribution collapses into a single distribution within the program state, indicating that all previously erased cells have been successfully programmed. This confirms the physical effectiveness of the overwrite sanitization process.

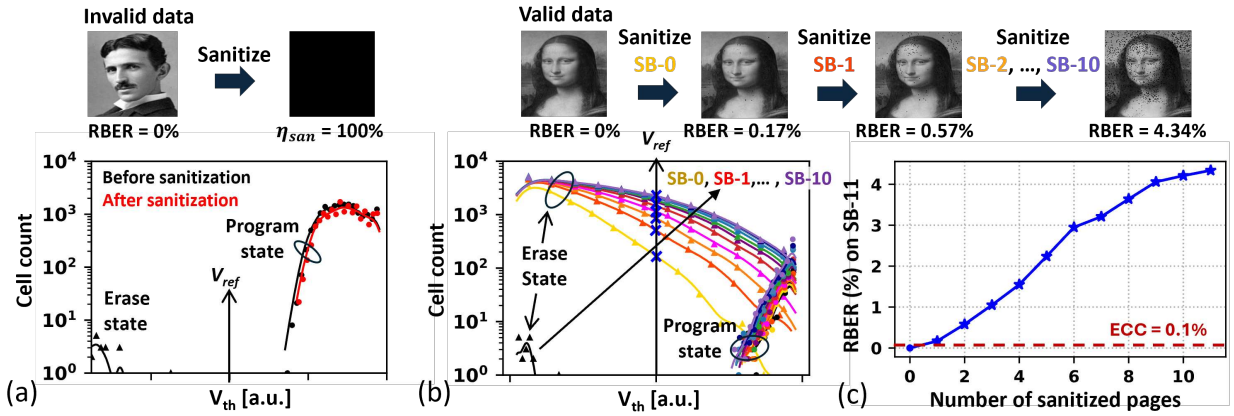


Figure 4.3: (a) Evaluation of overwrite sanitization on an invalid page. Before sanitization, the page contains a Tesla image, and after sanitization, it is converted to a perfect black image (η_{san} = 100%). The corresponding V_{th} distribution before (black) and after (red) sanitization. (b) Evaluation of the sanitization impact on the valid page of the same layer and the corresponding V_{th} distributions. (c) The RBER values on the valid page plotted against the number of sanitized pages within the same layer.

4.4.2 Evaluation of Post-Sanitization Disturbance on a Valid Page V_{th} Distribution

Figure 4.3(b) illustrates the impact of the sanitization process on a valid page that shares the same WL as the invalid pages within a flash block. Specifically, the Mona Lisa image is written on a flash page in the last sub-block (SB-11) of the targeted flash memory layer. The corresponding V_{th} distribution of the programmed flash memory page is shown in black. Initially, a sufficient voltage margin exists between erase and program states, ensuring reliable reads with zero RBER.

As overwrite sanitization operations are performed on pages in other sub-blocks (from SB-0 through SB-10), the V_{th} distribution of the valid page in SB-11 evolves. The plot reveals a notable upward shift in the erase distribution, while the program distribution remains relatively unchanged. This shift reduces the voltage margin between two states, causing increased overlap of distributions and raising the risk of uncorrectable bit errors, thereby compromising the reliability of data stored in the affected valid pages. Note that testing across multiple FG chips of the same part number consistently revealed significant disturbances following sanitization, rendering the implementation of page-overwrite sanitization impractical for this technology.

4.4.3 Evaluation of post-sanitization disturbance on valid page RBER

Figure 4.3(c) illustrates the cumulative increase in RBER of valid pages as the number of sanitized pages within a given vertical layer increase. RBER is closely tied to the cell's V_{th} distribution. During read operations, V_{ref} is used to determine the state of a flash cell. A bit error occurs when a cell's V_{th} shifts across V_{ref}, leading to incorrect readout. As shown in Figure 4.3(b), the erase-state V_{th} distribution shifts upward due to repeated sanitization. This results in a larger number of erase state cells with V_{th} exceeding V_{ref}, causing $1 \rightarrow 0$ bit errors. The total number of such cells directly contributes to the measured RBER.

Figure 4.3(c) quantifies this effect, showing that sanitizing a single page within a layer causes the RBER of adjacent valid pages to exceed 0.1%. This surpasses the correction capability of standard ECC engines, resulting in an uncorrectable bit error rate (UBER) [98]. While NAND vendors

typically mitigate the effects of V_{th} distribution shifts through read-retry mechanisms [99], by adjusting V_{ref} to optimize read accuracy, this technique becomes ineffective when two distributions begin to overlap significantly. Ideally, V_{ref} is positioned at the crossover point between the erase and program distributions, but this crossover itself shifts with increasing sanitization. Beyond a certain point, even the optimal V_{ref} adjustment cannot prevent UBER.

4.4.4 Evaluation of layer-dependent disturbance

Figure 4.4(a) illustrates how disturbance effects caused by the sanitization process vary by layer within the 3D stack. The analysis focuses on the valid flash pages within the last sub-block (SB-11) and tracks cumulative disturbances caused by successive sanitization of other sub-blocks. Initially, before the sanitization process begins, all pages in the SLC configuration exhibit zero RBER. However, even after the sanitization of the first sub-block (SB-0), a non-zero RBER appears in the valid page of SB-11. As more sub-blocks are sanitized, RBER continues to rise. After the sanitization of the second and third sub-blocks, the RBER may exceed the ECC correction threshold, leading to uncorrectable errors. This highlights that, despite SLCs typically having wider voltage margins, 3D NAND is still highly vulnerable to overwrite-induced disturbance.

The layer-dependent RBER trends in Figure 4.4(a) reveal that middle layers in the 3D NAND stack are more susceptible to disturbance than edge layers. This vulnerability is attributed to the

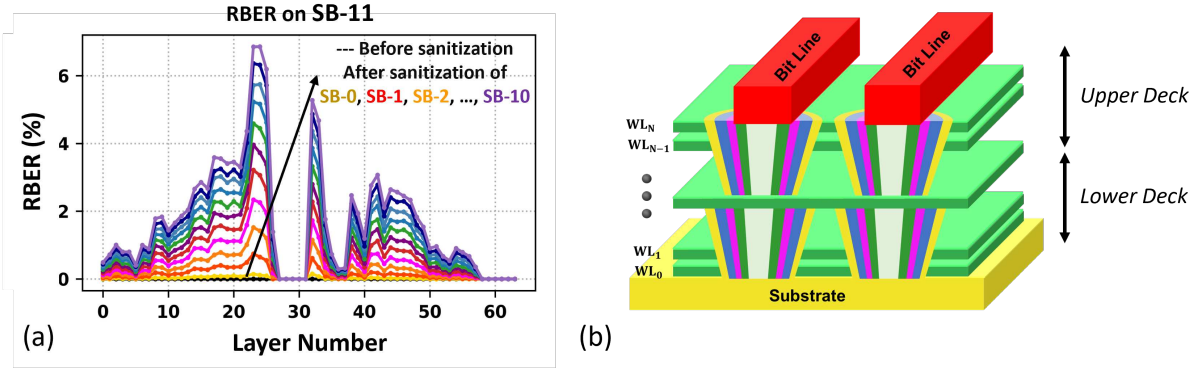


Figure 4.4: (a) Effects of sanitization on the RBER of valid pages across different layers in the last sub-block (SB-11). Different colors represent cumulative disturbance caused by the number of sanitized SBs. (b) A cartoon illustrating dual-deck technology.

pre-charge operation, a process typically performed before programming to minimize program disturbances on erased cells. Pre-charge voltages are typically applied through the BLs (top) and the substrate (bottom). Edge layers, being closer to both, benefit more from pre-charge conditions that reduce their susceptibility to disturbance during overwrite operations. In contrast, middle layers receive attenuated pre-charge. This is especially the case in pre-programmed blocks, where the programmed cells in the edge layers prevent the voltage propagation to the middle layers. This weakened pre-charge operation leaves the middle layers more vulnerable to programming disturbances. Interestingly, this weakening effect is absent when write operations are performed on an erased flash memory block, as the erased state allows pre-charge voltage to propagate freely into the middle layers. To address this issue, 3D NAND manufacturers recommend adhering to a proper page programming sequence, starting with the bottom-layer pages and progressing sequentially to the middle and top-layer pages. This practice ensures more uniform pre-charge conditions across the stack, thereby minimizing program disturbances.

Figure 4.4(a) also reveals a distinct discontinuity in RBER around layer 32, corresponding to the boundary between two fabrication tiers (or decks). This behavior stems from the two-tier or dual-deck (see Figure 4.4(b)) fabrication process used in NAND memory to increase layer count [100–102]. For example, the chip under test contains a total of 64 layers, with the first 32 layers forming the bottom tier (or the lower deck) and the remaining 32 layers constituting the top tier (or the upper deck). This discontinuity in RBER reflects process variation and structural asymmetry between the two decks, underscoring how fabrication techniques influence disturbance behavior across the flash memory stack.

4.5 Evaluation in SLC CT 3D NAND

4.5.1 Evaluation of Post-Sanitization Disturbance on Valid Page RBER

To further assess the feasibility of overwrite sanitization, we evaluate multiple 3D NAND chips with CT architecture from two major manufacturers. Interestingly, our findings reveal that overwrite sanitization can be successfully implemented on CT chips, without affecting the RBER of

Table 4.2: Summary of median RBER impact due to overwrite sanitization.

Samples	SLC			TLC								
	Before San.	Normal San.	PULSE San.	Before Sanitization			Normal Sanitization			PULSE Sanitization		
				LSB	CSB	MSB	LSB	CSB	MSB	LSB	CSB	MSB
64-layer FG	0%	0.93%	0%	0.01%	0.01%	0.02%	1.29%	12.88%	12.65%	0%	0.06%	0.57%
64-layer CT	0%	0%	0%	0.06%	0.04%	0.10%	5.09%	1.61%	0.07%	0.79%	0.02%	0.03%
176-layer CT	0%	0%	0%	–	–	–	–	–	–	–	–	–

valid pages. Table 4.2 summarizes these results, demonstrating that CT 3D NAND chips maintain zero RBER on valid flash memory pages even under worst-case sanitization scenarios (step-3 in Figure 4.2. Simultaneously, the invalid target pages were confirmed to convert to an all-zero state after sanitization, ensuring complete data sanitization. These results suggest that, while overwrite sanitization poses challenges for certain 3D NAND chips (e.g., 64-layer FG SLC chips), it is a viable solution for CT-based 3D NAND chips. This differentiation underscores the critical role of 3D NAND architecture in determining the suitability of data sanitization techniques.

4.5.2 Evaluation of Post-Sanitization Disturbance on V_{th} Distribution

Figures 4.5(a)-(b) illustrates the impact of sanitization operations on the V_{th} distribution for the erase and program states of an invalid and a valid flash memory page located in the last sub-block (SB-5, as the test chip contains 6 sub-blocks total), respectively. The plots show how these distributions evolve as overwrite sanitization is successively applied to sub-blocks SB-0 through SB-4. The results reveal that while the erase state distributions shift upward with each successive sanitization operation, a significant voltage margin persists between the program and erase distributions, even under the worst-case sanitization scenario. This substantial voltage margin ensures that no overlap occurs, thereby ensuring zero RBER in the valid pages. These findings suggest that CT flash memory employs a more robust pre-charge voltage scheme, effectively mitigating the impact of overwrite-induced disturbances.

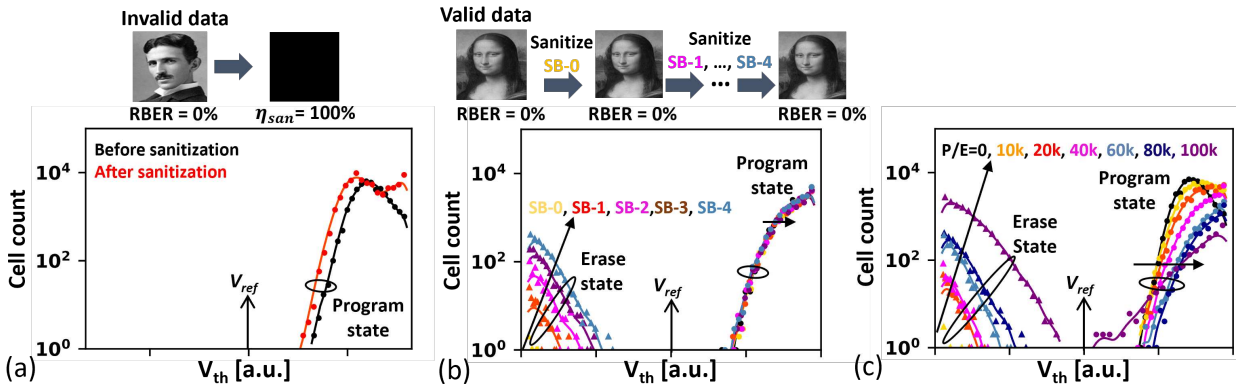


Figure 4.5: Cell V_{th} distribution as a function of successive sanitization of (a) invalid page, (b) valid page, and (c) program/erase (P/E) cycle count.

Although CT chips exhibited zero RBER in our tests, the underlying disturbance mechanisms caused by the overwrite operations are believed to be similar to those observed in FG-based architectures. However, the zero RBER in CT flash memory after sanitization can be attributed to the following factors. First, the CT flash memory chip may implement a more effective pre-charge voltage propagation scheme than the FG flash chip. Enhanced pre-charge efficiency reduces program disturbances by ensuring better charge equalization, making CT flash memory more resilient to disturbances introduced during overwrite sanitization. Second, CT flash memory may possess a wider voltage margin between the erase and program state distributions, providing a greater tolerance to disturbances. Due to the limited measurement range of the Read-offset command, the exact voltage margin of the erase V_{th} distribution immediately after the program operation could not be measured, leaving this hypothesis unverified.

4.5.3 Evaluation of Disturbance on P/E Cycled Block

Figure 4.5(c) illustrates the effect of P/E cycling on post-sanitization disturbances in valid flash memory pages of CT-based 3D NAND memory. The measurements were taken from the same flash memory block under identical sanitization conditions, but with varying P/E cycle counts. The results indicate that disturbance severity increases with higher P/E cycling, indicating that wear-out degrades the flash memory block's ability to withstand overwrite operations. This be-

havior can be explained by P/E-induced degradation that causes gate oxide breakdown, which in turn introduces additional leakage paths within the flash memory array. These leakage paths reduce the efficiency of the pre-charge operation, as a significant portion of the pre-charge voltage dissipates through these leakage pathways. Consequently, diminished pre-charge efficiency leads to increased disturbances in P/E-cycled blocks during sanitization. Although the overwrite-based sanitization technique increases disturbances V_{th} distribution, the trend observed in Figure 4.5(c) suggests that the technique remains applicable even under high P/E cycling conditions in SLC-based 3D CT NAND flash memory.

4.6 Evaluation in TLC CT/FG 3D NAND

Previous studies have primarily evaluated overwrite sanitization in the context of SLC flash memory. However, the concept can be extended to TLC flash memory as illustrated in Figure 4.6. Here we present the results of the first experimental characterization of overwrite sanitization in TLC 3D NAND memory. In TLC devices, each WL stores data across 3 different logical pages, namely LSB, MSB, and CSB pages. Sanitizing a single WL thus removes data from all three pages simultaneously. This is achieved by reprogramming all bits to the highest program level (i.e., L_7 state). Depending on the encoding scheme, the overwritten logical pages will transform into either an all-zero or all-one page following the sanitization process.

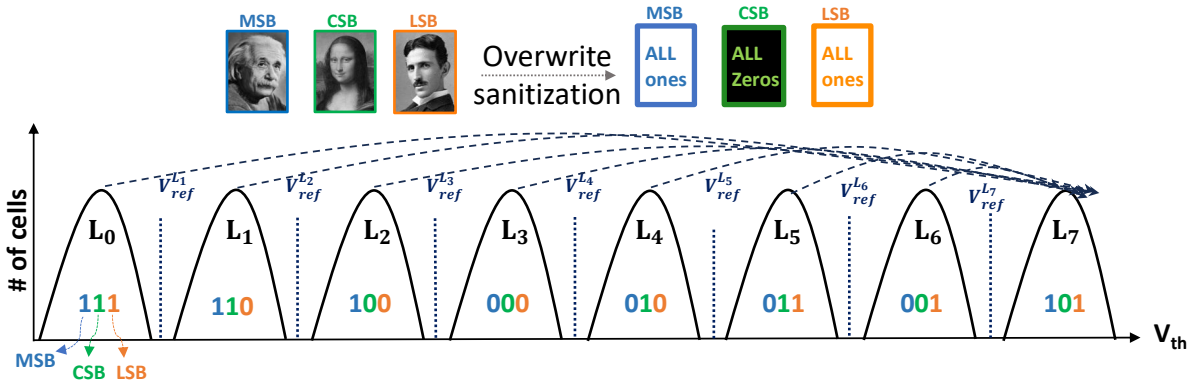


Figure 4.6: TLC storage with corresponding overwrite sanitization scheme.

4.6.1 Evaluation of Disturbance on Different Logical Pages

Figure 4.7(a) illustrates the results of overwrite sanitization applied to a CT 3D NAND chip. Images of Einstein, Mona Lisa, and Tesla were written on LSB, CSB, and MSB pages, respectively. After sanitization, these pages transform into uniform black or white patterns. For a chip using "101" as the Gray code representation of the highest program state, the LSB and MSB pages become solid white, while CSB pages become solid black. Thus, the sanitization operation completely removes data from the invalid pages. However, overwrite sanitization also introduces program disturbance on valid pages located in adjacent sub-blocks sharing the same physical layer. To evaluate this, we perform overwrite sanitization on all pages in SB-0 and measure disturbance on valid pages of the adjacent sub-blocks. Figure 4.7(b) shows the RBER of valid logical pages in the neighboring sub-block across all 64 layers. The RBERs exceed the correction capability of standard ECC in most cases, resulting in unrecoverable errors. Among the three logical pages, LSB pages show the highest susceptibility to disturbances, while MSB pages are the least affected.

The underlying cause of these disturbances is illustrated in Figure 4.7(c), which shows the evolution of the V_{th} distribution across TLC states. Similar to SLC flash memory, the erase state

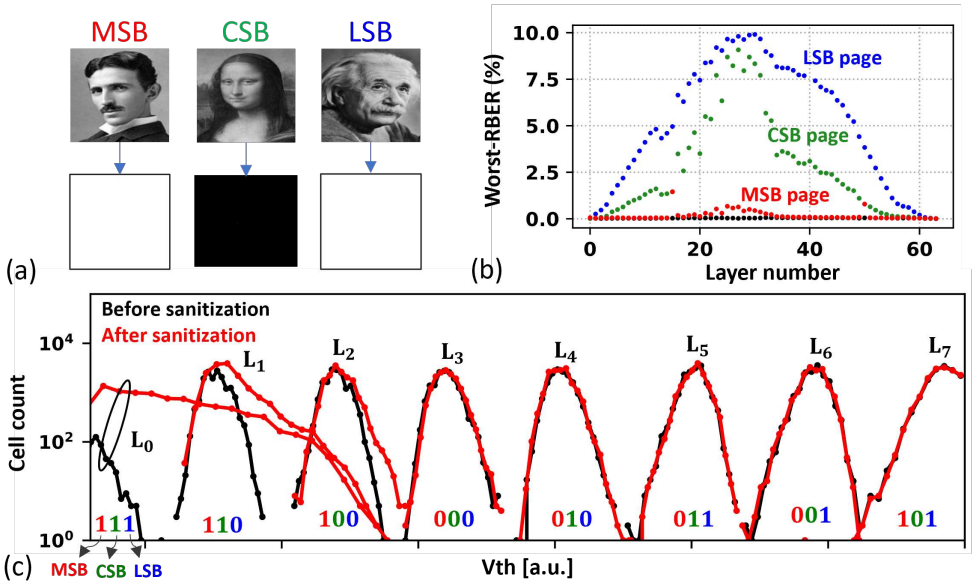


Figure 4.7: (a) Illustration of sanitization operation in TLC flash, where all three logical pages are sanitized together. (b) Post-sanitization disturbances on valid flash pages across different vertical layers of CT 3D NAND flash. (c) Sanitization impact on V_{th} distribution.

(i.e., L_0) distribution shifts significantly upward, crossing the read reference voltage after sanitization. Since the L_0 state is encoded as "111" and the next program state L_1 as "110," this upward shift in the erase state distribution critically impacts the RBER of LSB pages with $1 \rightarrow 0$ bit-flips, leading to higher error rates in LSB pages. Figure 4.7(c) also demonstrates that overwrite sanitization causes distortion to L_1 and L_2 state distributions, leading to increased RBERs of the corresponding CSB and MSB pages. Interestingly, the higher program states (e.g., L_3 and higher) remain relatively unaffected. These results suggest that lower V_{th} states are more vulnerable to program disturbances caused by overwrite sanitization. Such effects could potentially be mitigated by increasing pre-charge voltage during the overwrite process. If NAND manufacturers were to provide a specialized overwrite command with enhanced pre-charge capabilities, the severity of such disturbances could be reduced.

4.6.2 Evaluation of layer-dependent disturbance

Another noteworthy trend, visible in Figure 4.7(b), is the layer-dependent RBER pattern: valid pages located in middle layers are more susceptible to sanitization disturbances. This behavior mirrors the trend observed earlier in SLC FG 3D NAND devices, as described in Figure 4.4(a). We attribute this to the same issue of inefficient pre-charge voltage propagation into the middle layers due to the inherent architecture of 3D NAND technology. Consequently, the layer dependence of sanitization disturbances appears to be a universal characteristic across different 3D NAND architectures, affecting both FG and CT devices.

4.6.3 Multi-chip evaluation

To generalize our findings, we performed overwrite sanitization experiments on multiple TLC 3D NAND chips from three major manufacturers, including both FG and CT variants. In each test, only pages in the first sub-block were sanitized, and the RBERs of valid pages from the last sub-block were measured. This setup represents the minimum disturbance scenario per flash memory layer. A summary of these results is presented in Table 4.2, which shows the median RBERs before and after overwrite sanitization for a range of tested chips. Despite this conservative approach, all

tested chips exhibited significantly elevated sanitization-induced disturbances on valid flash memory pages. These disturbances consistently surpassed the error correction capability of standard ECC, making direct implementation of overwrite sanitization impractical for TLC storage. Note that the evaluation of TLC-mode sanitization on the 176-layer CT NAND chip cannot be properly performed once all pages within a block are programmed (i.e., the block is closed).

4.7 Key Observations/Insights from Evaluation

In summary, our experimental evaluation yields the following key observations and insights:

- **Reliability impact on CT vs. FG**: We find that FG 3D NAND chips are more vulnerable to overwrite-induced disturbances compared to CT chips. In FG devices, post-sanitization corruption of valid data is observed in both SLC and TLC. In contrast, CT flash memory chips demonstrate robustness in SLC mode but are found vulnerable in TLC mode. We believe that the higher sensitivity of the FG chips to sanitization-induced disturbance is mainly due to their unique array structure and pre-charge voltage conditions.

- **Impact on V_{th} distribution for SLC and TLC storage**: Post-sanitization disturbances primarily cause an upward shift in the erase-state V_{th} distribution in SLC flash memory, causing $1 \rightarrow 0$ bit flips. In the case of TLC flash memory, the lower states (L_0, L_1 and L_2) experience a voltage upshift after sanitization, while the higher V_{th} states are relatively unaffected. These effects will cause variable RBER increase on logical pages based on the data encoding scheme.

- **Layer-dependent vulnerability**: Pages located in the middle layers in the 3D NAND stack exhibit significantly higher vulnerability to post-sanitization disturbances than those located at the edges due to inefficient pre-charge voltage. Thus, the number of sanitization operations in the middle layers needs to be restricted to minimize the risk of data corruption.

- **Increased post-sanitization disturbances in P/E cycled blocks**: P/E cycled flash memory blocks are more susceptible to post-sanitization disturbances, which impose further con-

straints on overwrite sanitization on worn-out blocks. The storage controller should implement an adaptive scheme for heavily cycled flash memory blocks to minimize the impacts of sanitization and reduce the risk of data corruption. For example, the number of overwrite sanitization operations can be lowered on higher P/E cycled blocks.

These findings highlight the design-space metrics for implementing overwrite sanitization in 3D NAND flash memory. Successful deployment requires a thorough evaluation of device-specific characteristics and innovative design strategies to mitigate disturbances while ensuring data integrity.

4.8 PULSE: Partial Update for Low-disturbance Sanitization Engine

To mitigate the disturbance caused by overwrite sanitization, we propose PULSE, a Partial Update-based Low-disturbance Sanitization Engine. PULSE is a page-overwrite-based technique that balances the trade-off between sanitization efficiency on invalid target pages and disturbance minimization on valid pages. The sanitization efficiency of invalid target pages is defined in Section 4.3 as Eq. 4.1. It is evaluated by comparing the read-out content of a sanitized (invalid) page with a solid data pattern, which can be either all-zero or all-one, depending on the logical page type. For example, a 100% sanitization efficiency in SLC storage indicates that the invalid page data has been fully converted into an all-zero data pattern, rendering it logically irrecoverable. However, it is important to note that even partial sanitization (i.e., efficiency < 100%) can still achieve logical irrecoverability, while inducing significantly less disturbance on nearby valid pages. The disturbance's impact on the valid pages is measured by $\text{RBER}^{\text{valid}}$. PULSE offers a framework to systematically balance these two competing metrics, $\text{RBER}^{\text{valid}}$ and $\eta_{\text{san}}^{\text{inval}}$, to achieve practical, low-disturbance sanitization for 3D NAND flash memory.

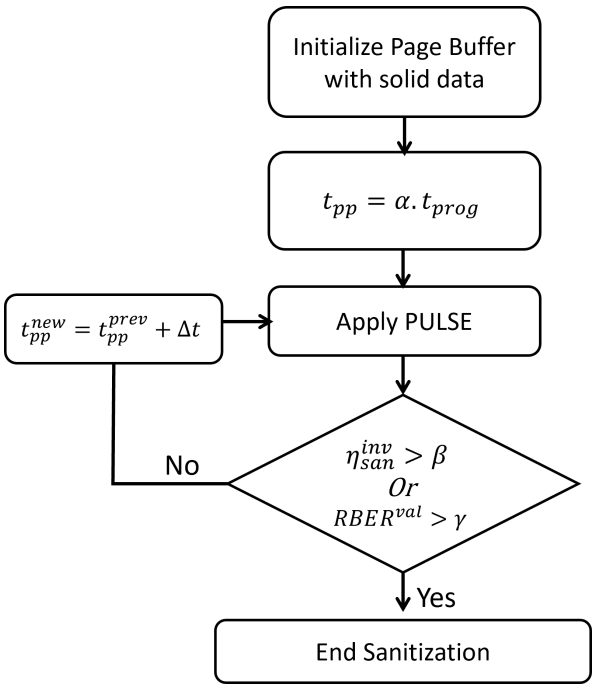


Figure 4.8: Algorithm for PULSE implementation and t_{pp} evaluation.

Algorithm 1 PULSE Sanitization algorithm.

Initialize:
    Select a page where data has become invalid
    Estimate partial program time: $t_{pp} = \alpha \cdot t_{prog}$
    Initialize page buffers with solid data pattern
Perform:
while true **do**
    Issue NAND page program command (0x80-0x10)
    Apply time delay t_{pp}
    Issue NAND RESET command (0xFF)
    Issue page READ command to evaluate η_{san}^{inval} and $RBER^{val}$
    if $\eta_{san}^{inval} < \beta$ **and** $RBER^{val} < \gamma$ **then**
        Increase t_{pp} and repeat
    else
        Stop sanitization
        break
    end if
end while

4.8.1 Implementation Algorithm of PULSE in COTS NAND Chips

The algorithm for implementing PULSE sanitization is presented in Algorithm 1 and illustrated in Figure 4.8. Similar to traditional overwrite sanitization, PULSE involves overwriting the target

page with a solid data pattern (e.g., all-zero for SLC pages). However, the key distinction of PULSE lies in its premature termination of the overwrite process, achieved by issuing a RESET command after the program operation. The delay between the start of the overwrite and the RESET command, referred to as the partial program time (t_{pp}), is a tunable parameter that controls the trade-off between sanitization efficiency and program disturbance. The following algorithm can be used to fine-tune t_{pp} based on the characteristics of the specific NAND flash device. The initial value of the t_{pp} can be set as a fraction of the nominal page program time (t_{prog}) as $t_{pp} = \alpha \cdot t_{prog}$, where α is a design parameter with a typical value ≈ 0.5. If this initial t_{pp} does not meet the required sanitization efficiency, the overwrite process can be repeated with a longer t_{pp}, as illustrated in Figure 4.8. Two additional design parameters, β and γ, define the target thresholds: β represents the minimum acceptable sanitization efficiency for the invalid page (typically 80–90%), and γ denotes the maximum tolerable RBER on valid pages (typically 0.1-0.5%). If PULSE is implemented through FTL, these parameters (α, β, and γ) needs to be pre-defined in the FTL firmware based on characterization results for the given storage technology.

4.8.2 Mechanism Behind PULSE: Sanitization Efficiency and Trade-Offs

Figure 4.9(a) illustrates the inherent trade-off between η_{san}^{inval} and $RBER^{val}$ in the PULSE sanitization approach, as demonstrated using a 3D SLC NAND device. The key tunable design parameter in PULSE is the t_{pp}, which impacts both η_{san}^{inval} and $RBER^{val}$. To illustrate this trade-off, we write a Mona Lisa image on a valid page and a Tesla image on a page designated for sanitization. The latter is then sanitized using an all-zero data pattern. At shorter t_{pp} durations, the overwrite process is prematurely terminated, leaving parts of the Tesla image visible. The resulting sanitization efficiency is poor ($\eta_{san}^{inval} < 80\%$) when compared to a complete black (all-zero) image. However, shorter t_{pp} values minimize disturbance on the valid page, keeping the Mona Lisa image intact with its RBER near zero and no visible corruption.

Conversely, as t_{pp} increases, the invalid data progressively transitions into a completely black image, reflecting improved sanitization efficiency. However, this improvement comes at the cost

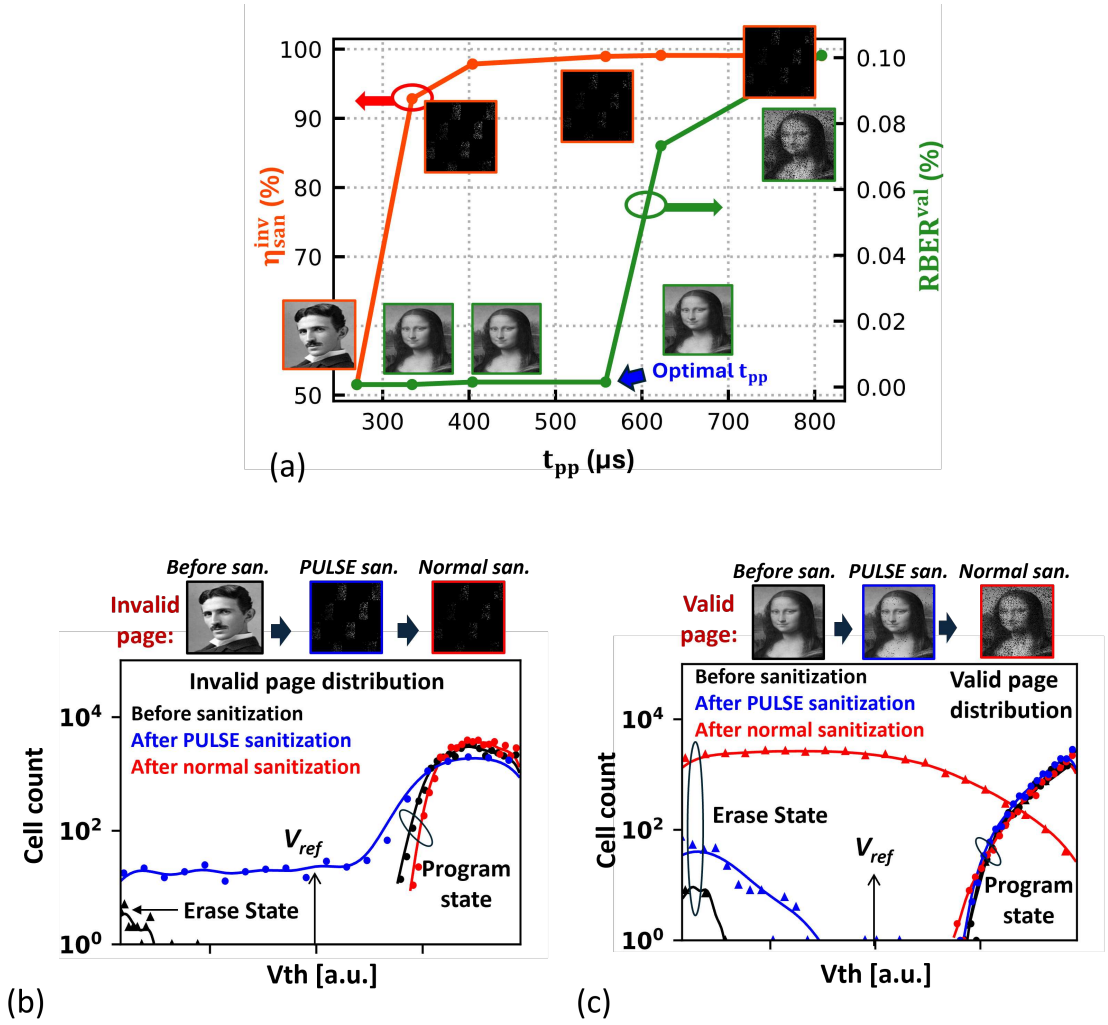


Figure 4.9: (a) Illustration of sanitization efficiency and RBER trade-off in PULSE sanitization as a function of t_{pp}. Cell V_{th} distribution on (b) invalid page and (c) valid page after PULSE vs. normal sanitization.

of higher disturbance on the valid page, resulting in an increased RBER. The plot in Figure 4.9(a) highlights the optimization window around $t_{pp} \approx 550$ µs, where η_{san}^{inval} reaches 98% and RBER^{val} remains as low as 0.002%. This point represents a balanced trade-off, offering an adequate level of data sanitization with minimal disturbance to valid data. Note that normal sanitization results in $t_{pp} \approx 810$ µs, achieving $\eta_{san}^{inval} = 99.9\%$ but increasing RBER^{val} to 0.1%. For a practical sanitization purpose, we can choose $\eta_{san} = 90\%$ corresponding to $t_{pp} \approx 330$ µs, which will cause even lower impact on RBER. Thus, PULSE introduces a flexible and practical approach to mitigating overwrite-induced disturbances in 3D NAND flash.

The physical mechanism behind PULSE's reduced disturbance is further examined in Figure 4.9(b)-(c) through the cell V_{th} distribution. Figure 4.9(b) presents the V_{th} distribution of an invalid page before and after sanitization, while Figure 4.9(c) shows the V_{th} distribution of a valid page that shares the same WL as the invalid page. Before sanitization, two distinct V_{th} distributions are observed, corresponding to the erase and program states. Although the erase-state distribution is only partially captured due to measurement limitations, its tail remains visible. After normal sanitization, the V_{th} of the originally erased cells shifts and merges with that of programmed cells, effectively converting all data to logical '0' and achieving a sanitization efficiency close to 100%. In contrast, PULSE sanitization preserves some cells in the erase state, resulting in a wider lower tail of the program distribution. However, the advantage of PULSE is evident in the V_{th} distribution of valid data, as shown in Figure 4.9(c). With normal sanitization, the erase-state distribution of the valid page shifts upward, encroaching into the program-state and causing overlap between distributions, which increases the RBER. However, PULSE sanitization minimizes this upward shift of the erase-state distribution, maintaining a sufficient V_{th} margin between the states and thereby keeping the RBER of valid pages near zero.

4.8.3 Evaluation of PULSE on SLC NAND

While the mechanism behind PULSE sanitization was explained in the previous section, Figure 4.10 presents a detailed layer-dependent evaluation of PULSE on FG-type SLC 3D NAND. The FG-type is chosen for this evaluation because it exhibits significant RBER on valid pages after standard sanitization operations. In contrast, CT-type SLC 3D NAND exhibits strong immunity to such disturbances, rendering PULSE unnecessary for that class of devices.

Figure 4.10(a) shows the RBER observed on valid pages across different storage layers under the worst-case sanitization conditions. Here, the "worst-case" refers to a scenario where all but one page in a layer is sanitized, and the RBER is measured on the last valid page. For example, the tested FG SLC chip contains 12 sub-blocks per layer. Thus, 11 pages in each layer are sanitized sequentially, and the cumulative RBER is measured on the last remaining page. Different colors

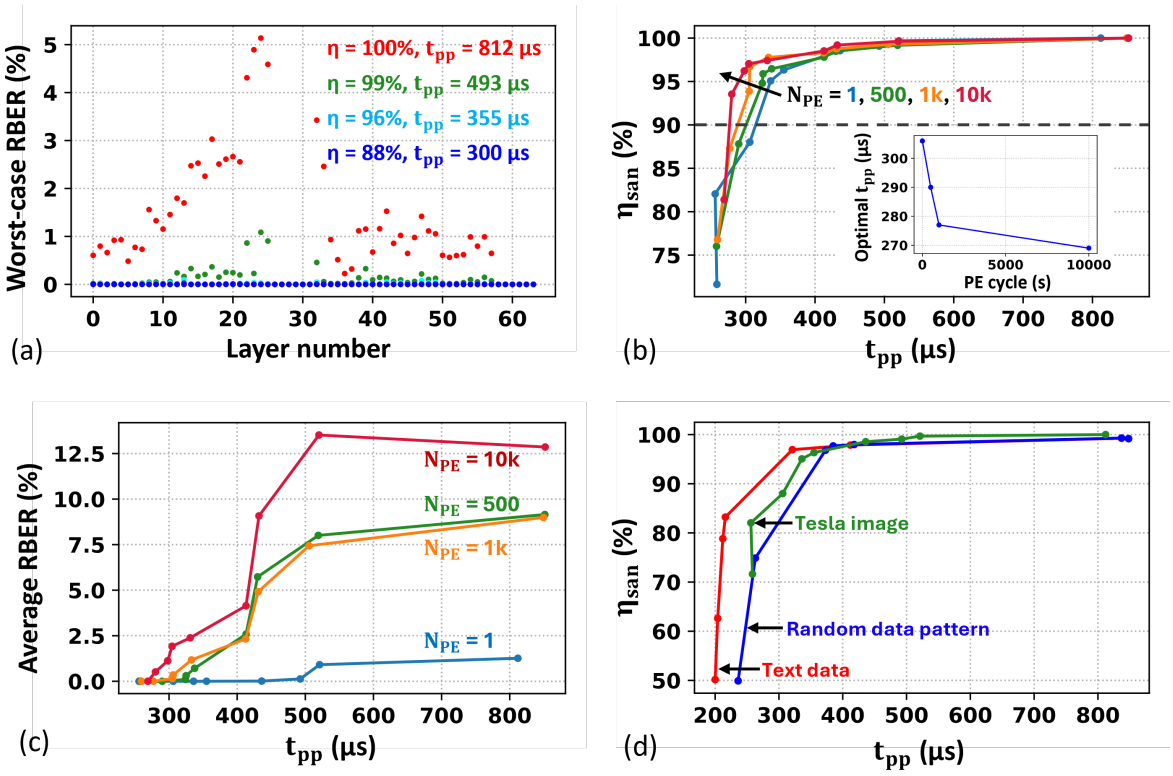


Figure 4.10: (a) RBER impact on valid pages across different layers after sanitization in SLC FG storage. (b) The η_{san} for different P/E cycled blocks as a function of t_{pp}, with optimal t_{pp} vs. P/E cycle (s) provided. (c) RBER impact of PULSE on different P/E cycled blocks. (d) The η_{san} for different types of data files as a function of t_{pp}.

in Figure 4.10(a) correspond to varying target η_{san}, each achieved by adjusting the t_{pp}, as indicated in the inset. A longer t_{pp} leads to higher η_{san}, but also greater RBER on the valid page. The results show that the RBER impact can be significantly reduced by selecting a lower η_{san} for invalid pages. For instance, maintaining η_{san} below 96% keeps RBER under 0.1% across all layers, which is well within the correction capability of standard ECC. Since the middle layers are more susceptible to disturbance, adopting a layer-dependent η_{san} strategy can optimize both sanitization efficiency and RBER impact. In this chip, for example, high η_{san} ($\sim 99\%$) can be applied to the top and bottom 10 layers, while slightly reduced efficiency ($\sim 96\%$) in the middle layers ensures effective sanitization without triggering ECC failures.

NAND flash storage has finite endurance, typically quantified by the number of P/E cycles a block can undergo, denoted as N_{PE}. Figure 4.10(b) illustrates a strategy for adapting PULSE

implementation based on the P/E cycle condition of an SLC block. To achieve the same target η_{san}, a fresh block requires a slightly longer t_{pp} compared to a block that has undergone multiple P/E cycles. The inset in Figure 4.10(b) shows how the required t_{pp} for achieving η_{san} = 90% decreases as N_{PE} increases. This trend arises because the page program time in NAND flash tends to slightly decrease with P/E cycling due to effects such as oxide trapping. Consequently, t_{pp} must be dynamically adjusted based on the P/E cycle condition of the SLC block to ensure the target sanitization efficiency. Figure 4.10(c) shows the RBER impact of PULSE sanitization on different P/E cycled blocks. Fresh blocks experience significantly lower RBER impact compared to the P/E cycled blocks. In general, RBER increases with longer t_{pp}. The results in Figure 4.10(b) and Figure 4.10(c) indicate that the choice of t_{pp} is critical to optimize the trade-offs between η_{san} and RBER.

Although image data are used in this paper for visualization, in the following, we evaluate PULSE on other types of data patterns. Figure 4.10(d) shows η_{san} as a function of t_{pp} for three different data patterns: (1) a Tesla image, represented by 8-bit grayscale pixels; (2) a text data pattern, composed of ASCII characters; and (3) a random data pattern, consisting of an equal number of 1s and 0s. Each data type is first converted to hexadecimal values and then written to flash pages. After that, we apply PULSE with different t_{pp}, and the corresponding η_{san} is recorded. The experimental results in Figure 4.10(d) show that the three curves representing three different data files show a very similar trend. To achieve η_{san} = 90%, the optimal t_{pp} differs by only a few microseconds in all data types. Note that when η_{san} = 90%, all data types become unrecognizable. Based on these results, we conclude that PULSE is effective regardless of the data pattern, confirming the robustness of this technique.

4.9 Evaluation of PULSE on TLC CT/FG NAND

In this section, we evaluate the effectiveness of the PULSE sanitization technique on 3D TLC CT and FG NAND flash memories, with the results for the 3D CT NAND flash shown in Figure 4.11(a)-(f). Figure 4.11(a) presents the impact of sanitization on the invalid target pages, while Figure 4.11(b) illustrates the unintended disturbance experienced by valid pages that share the same

WL during the sanitization process. Images outlined in black represent data before sanitization, while red and blue indicate data after normal and PULSE sanitization, respectively. Given that the highest V_{th} state in CT TLC is encoded as '101,' the MSB and LSB pages are converted to solid white (all-ones), and the CSB page becomes solid black (all-zeros). Normal sanitization achieves nearly 100% efficiency across all three pages, effectively erasing all traces of prior data. In contrast, PULSE sanitization achieves $\eta_{\text{san}} \approx 100\%$ for the MSB and CSB pages, but the LSB page retains partial remnants of the original data, resulting in a lower sanitization efficiency of approximately 90%.

Figure 4.11(b) illustrates the unintended disturbance experienced by valid pages residing on the same WL as the sanitized pages. As shown, normal sanitization introduces significant corruption to the valid data, resulting in visible degradation of image quality. Interestingly, we observe that the LSB and CSB pages exhibit greater distortion compared to the MSB page. In contrast, PULSE sanitization effectively mitigates these disturbances, preserving the integrity of all three valid pages with minimal distortion.

Figure 4.11(c) illustrates the origin of asymmetric disturbances on the logical valid pages by plotting the corresponding cell V_{th} distribution under three conditions: the state immediately after write (black), after normal sanitization (red), and after PULSE sanitization (blue). Based on Figure 4.11(c), it is evident that normal sanitization induces significant disturbance, particularly in the erase state (L_0) and the lowest two programmed states (L_1 and L_2), as indicated by the extended upper tails of the red distribution. Since MSB page data remains the same (logic-1) for all three lower V_{th} states (L_0, L_1 and L_2), it experiences minimal disturbance due to the V_{th} distortion. Only a small fraction of bits, illustrated by $L_2 \rightarrow L_3$ shift in the distribution plot, will cause errors in the MSB page. In contrast, the LSB page experiences the highest disturbance due to a significant $L_0 \rightarrow L_1$ shift. The CSB page experiences a disturbance due to the $L_0 \rightarrow L_2$ and $L_1 \rightarrow L_2$ shifts. By comparison, the distribution following PULSE sanitization (blue) closely aligns with the original valid data distribution (black), demonstrating that the partial program approach significantly reduces the disturbance caused by normal sanitization.

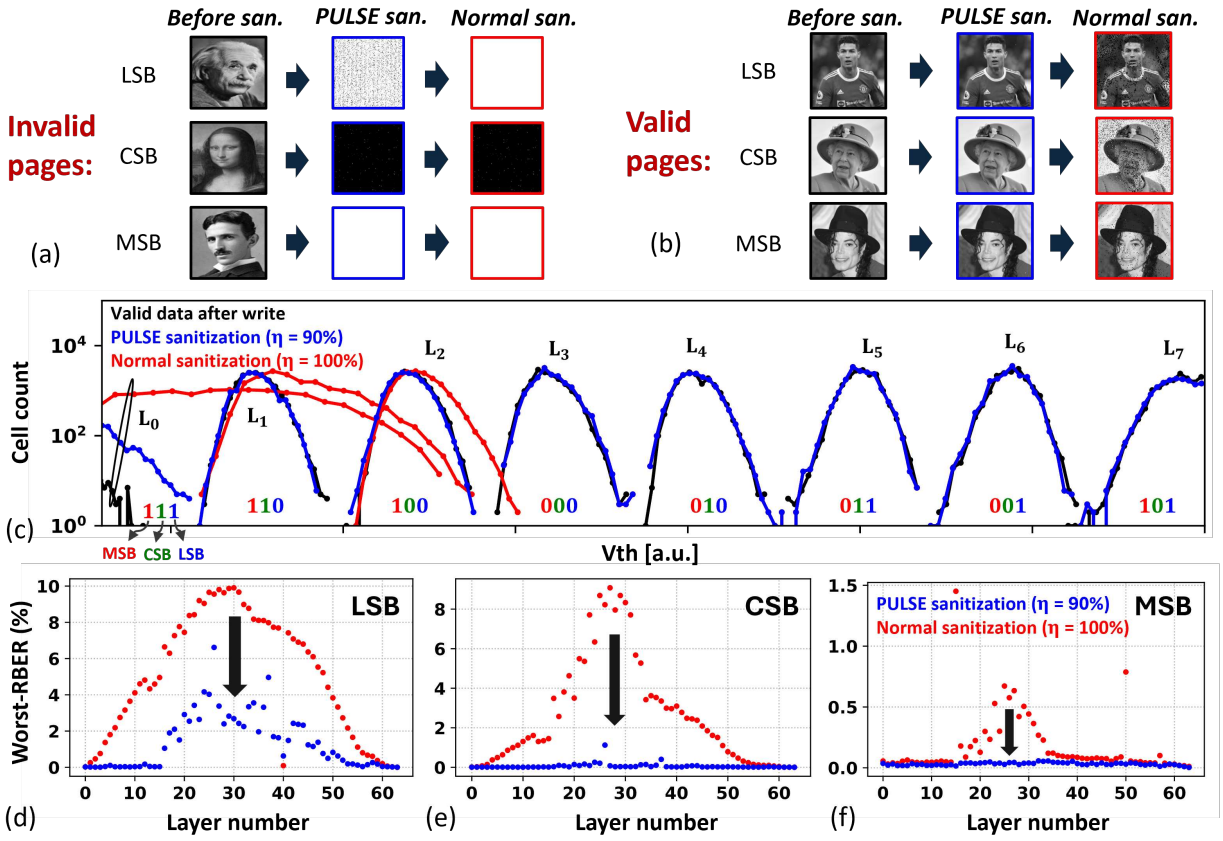


Figure 4.11: Evaluation of normal and PULSE sanitization on 3D CT TLC flash memory: Images stored in (a) invalid pages and (b) valid pages before sanitization (black) and after normal (red) and PULSE (blue) sanitization. (c) Effect of the two sanitization methods on V_{th} distribution of valid data stored in 3D CT TLC flash cells. Comparison of RBER for normal and PULSE sanitization on valid flash pages across different vertical layers of 3D CT NAND flash memory: (d) LSB pages, (e) CSB pages, and (f) MSB pages.

Next, we evaluate the impact of PULSE sanitization on the RBER across different stacking layers. The results are presented in Figure 4.11(d)–(f), where Figure 4.11(d) corresponds to the LSB pages, Figure 4.11(e) to the CSB pages, and Figure 4.11(f) to the MSB pages. Our findings indicate that PULSE sanitization effectively reduces RBER across all three page types. However, the middle layers continue to exhibit relatively high RBER due to inefficient pre-charge voltage, as discussed previously. Based on these results, different sanitization policies can be proposed. For example, sanitization operation on middle-layer pages can be prevented if there are valid pages on the same vertical layer. Alternatively, the number of sanitization operations on the middle layers can be restricted to a small number (e.g., one or two) to minimize RBER impact.

The effectiveness of the PULSE sanitization on FG 3D TLC NAND flash memory is evaluated, and the results are shown in Figure 4.12(a)-(f). Similar to CT 3D TLC NAND, the FG 3D NAND exhibits a similar trend. Figure 4.12(a) shows the sanitization efficiency on target (invalid) pages, while Figure 4.12(b) depicts disturbance to valid pages sharing the same WL. In 3D FG NAND under test, the highest programmed state (L_7) is encoded as '011' for LSB, CSB, and MSB pages, resulting in a black (all-zero) image for the LSB page and white (all-one) images for CSB and MSB pages. Similar to CT NAND, FG TLC also achieves 100% sanitization efficiency after normal sanitization, while PULSE sanitization leaves some residue traces on the invalid CSB page, as shown in Figure 4.12(a). However, severe data corruption is observed on CSB and MSB pages under normal sanitization, whereas PULSE causes minimal corruption in all valid pages, as shown in Figure 4.12(b).

Figure 4.12(c) depicts the V_{th} distribution of valid pages before and after PULSE sanitization. The results indicate that PULSE sanitization effectively mitigates the disturbance introduced by the normal sanitization method, particularly for flash cells in the lower states (L_0-L_2). Figure 4.11(d)-(f) further evaluates the RBER across different stacking layers and shows that PULSE significantly reduces the disturbance caused by normal sanitization. Notably, the middle layers, which are more susceptible to the disturbance under normal sanitization, benefit the most.

Next, we summarize the t_{pp} required to achieve 90% of η_{san} on different NAND flash memory chips from various manufacturers in Table 4.3. Note that the timing values of each NAND flash memory chip are measured by probing the ready/busy (RB) pin during normal (t_{prog}) and PULSE (t_{pp}) sanitization. Typically, SLC page write requires significantly less time than TLC pages, which directly results in shorter t_{pp} for SLC configuration. The findings in this section confirm that PULSE sanitization can be successfully deployed on both CT and FG 3D NAND flash memories with both SLC and TLC configurations, resulting in a reduction in disturb-induced errors compared with normal sanitization.

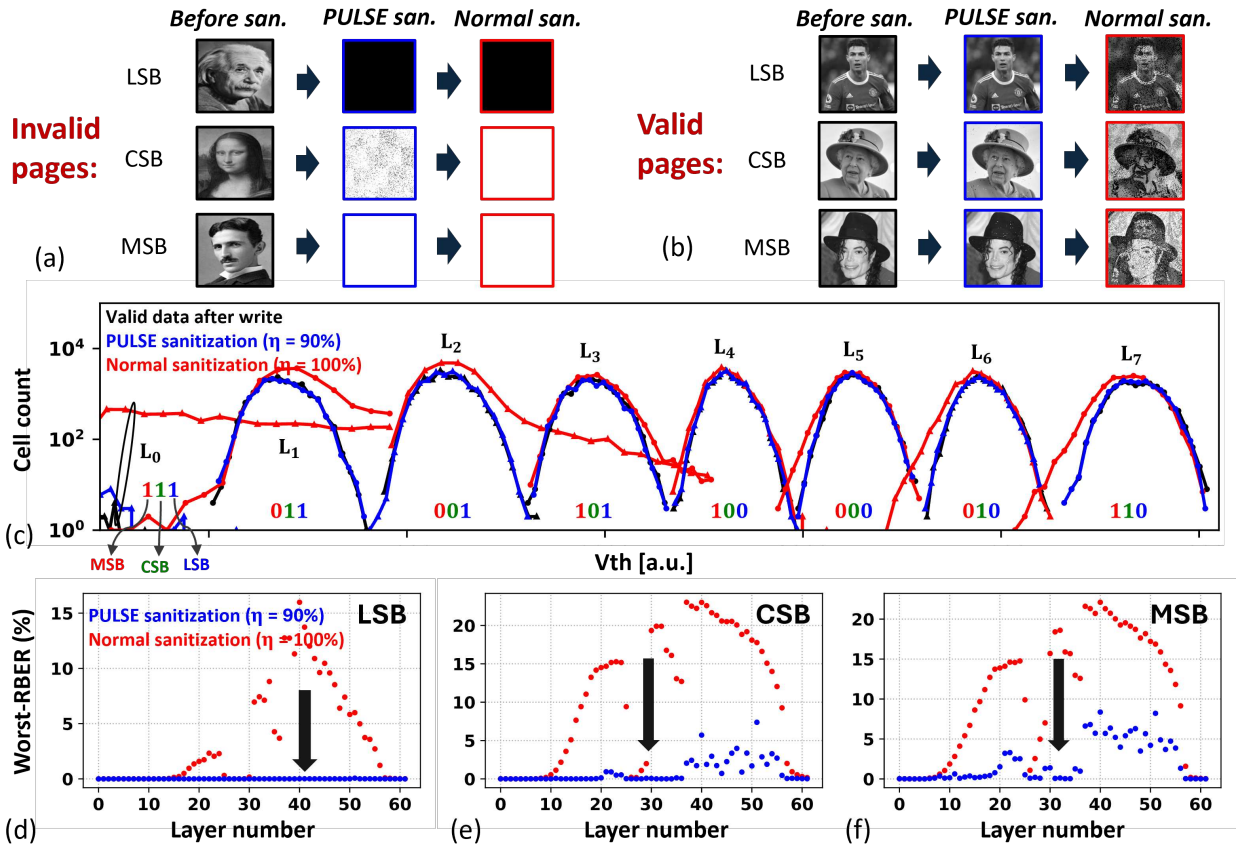


Figure 4.12: Evaluation of normal and PULSE sanitization on 3D FG TLC flash memory: Images stored in (a) invalid pages and (b) valid pages before sanitization (black) and after normal (red) and PULSE (blue) sanitization. (c) Effect of the two sanitization methods on V_{th} distribution of valid data stored in 3D FG TLC flash cells. Comparison of RBER for normal and PULSE sanitization on valid flash pages across different vertical layers of 3D FG NAND flash memory: (d) LSB pages, (e) CSB pages, and (f) MSB pages.

Table 4.3: Partial program time on different NAND flash memory chips.

Samples	t_{pp} (η_{san} = 90%)	t_{prog} (η_{san} = 100%)
64-layer FG SLC	306 µs	849 µs
64-layer FG TLC	1.20 µs	1.79 ms
64-layer CT TLC	1.37 ms	1.95 ms

4.9.1 Implementation Challenges and Overhead Analysis of PULSE

PULSE can be implemented on unmodified NAND chips through firmware-level support in the FTL. However, this approach introduces several challenges and performance overheads, which are discussed below:

Overwrite permissions: Some NAND chips do not support overwrite operations once a flash memory block is fully programmed (i.e., the block is "closed"). As a result, PULSE cannot be implemented on such chips unless the manufacturer provides specific features or commands to enable overwrite functionality. Additionally, some NAND chips are equipped with built-in data randomizers and internal ECC engines, which can interfere with the PULSE overwrite process. However, most of the recent raw 3D NAND chips do not include these features, as NAND vendors typically offer greater flexibility to system integrators, allowing them to implement their own FTL and associated functions. Hence, PULSE can be implemented through FTL firmware on most NAND available today.

Precise timing control: PULSE implementation relies on a partial overwrite operation that is highly sensitive to the t_{pp} and the specific behavior of the RESET command. Precisely controlling t_{pp} can be challenging for the FTL, especially during the run-time of the system. Furthermore, variations in RESET command implementations across different NAND manufacturers can introduce additional uncertainty, further complicating reliable PULSE execution. Thus, PULSE implementation will require FTL design with fine-grained timing control of NAND chips.

Performance overhead: The PULSE implementation may introduce performance overheads for the FTL. Since PULSE requires multiple overwrite operations with varying partial program times, along with the determination of sanitization efficiency and RBER on valid pages, it incurs additional latency and computational overhead. However, the FTL can mitigate the impact on host-visible performance by scheduling the sanitization process in the background during the idle phase of the SSD, thereby hiding the latency from the host system.

RBER impact: The most direct impact of overwrite sanitization is increased RBER on valid pages sharing the same WL as the sanitized page. While PULSE sanitization mitigates this effect, it still induces non-negligible RBER in TLC storage. Because NAND flash follows an out-of-place update strategy, overwrite operations, though supported, are rare and not optimized for latency or disturbance control. In 3D NAND, the subblock architecture further amplifies vulnerability to disturbance during overwrites.

Some of the above-mentioned implementation challenges and the overhead can be mitigated with the support of NAND manufacturers. For example, an ISPP-based programming sequence is not needed for PULSE. Instead, a single high-voltage program pulse is more appropriate for overwrite operations. NAND manufacturers can design a special command sequence to offer such a single pulse-based overwrite operation, which will minimize latency impact. In addition, the pre-charge voltage conditions can be optimized for the single pulse-based PULSE implementation, which will minimize the RBER impacts on the valid pages.

4.10 Implications of PULSE for Future 3D NAND SSD Design

Our findings have the following implications for the implementation of overwrite sanitization in next-generation 3D NAND SSDs:

- **Trade-off between sanitization efficiency and data integrity**: PULSE provides an effective mechanism for balancing sanitization efficiency with minimized disturbance on valid data. This makes overwrite-based sanitization feasible for 3D NAND architectures when tailored to specific applications and system requirements.

- **Enhanced PULSE implementation requires manufacturer support**: Our work demonstrates the feasibility of implementing overwrite-based sanitization, including PULSE, on commercial 3D NAND flash chips without requiring any special support from the chip manufacturers. However, even greater sanitization efficiency and lower disturbance overhead can be achieved with modest support from manufacturers. For example, vendors could introduce specialized command sequences to support shorter programming times, leveraging ISPPs, or allow for adjustable pre-charge voltage conditions during page overwrite operations. These enhancements would further reduce RBER on valid pages and improve timing precision, facilitating more reliable and efficient PULSE integration into the FTL firmware.

- **PULSE for future generations of 3D NAND**: The 3D NAND technology is poised to have more than 1000 layers per flash memory block in the next decade. This will increase the

block size of future 3D NAND chips in terms of the number of pages, causing "Big Block Issues" [52]. Thus, the implementation of PULSE will be increasingly important for the future generation of 3D NAND chips to prevent data leakage without incurring wear-out and significant performance overheads.

- **PULSE on P/E cycled block**: PULSE needs to be optimized based on the P/E cycle condition of a flash memory block. Experimental evaluation shows that the optimal t_{pp} decreases with increasing P/E cycles. In addition, storage controllers may limit the number of sanitized pages per layer to ensure tolerable RBER levels for valid pages within the same layer based on underlying storage configuration, such as SLC vs. TLC.

- **Layer-specific management**: Since middle layers of the 3D stack are inherently more susceptible to post-sanitization disturbances than edge layers, storage controllers could use this insight to intelligently manage the number of sanitized pages based on layer location.

4.11 Conclusion

This paper explores the thermal annealing effects on the endurance of 3D NAND flash memories when reaching the end of their lifetime. The experimental evaluation is carried out on a COTS 3D FG TLC and MLC NAND flash memory chips. Our experimental results on 3D TLC chips reveal that thermal annealing of the chip in the programmed states offers the highest endurance recovery. Moreover, we find that retention RBER of the post-annealed worn-out (N_{PE} = 1k) block remarkably drops to a value equivalent to $\sim$ 500 P/E cycles. However, the post-annealed worn-out block can take only $\sim$ 300 P/E cycles extra, which gives $\sim$ 1,300 P/E cycles total before reaching its second lifetime. This leads to $\sim$ 30% endurance improvement. In addition, we find that the erase time slightly drops after thermal annealing. Finally, we conclude that the annealing method is applicable to both TLC and MLC 3D NAND flash memory chips.

Chapter 5

Block Erase Vulnerabilities in CT 3D SLC NAND

3D NAND flash memory is the dominant storage technology in modern consumer and enterprise systems. As these devices routinely store sensitive personal and organizational data, secure sanitization prior to reuse, resale, or disposal is critical to preserve user privacy and prevent unauthorized data disclosure. Block erase operations are widely assumed to provide complete data removal; however, their effectiveness in modern 3D NAND has not been fully validated. This is particularly concerning given the importance of data privacy and evidence showing that supposedly sanitized devices can still contain recoverable data. These observations suggest that block erase alone may be insufficient for secure sanitization, motivating this work to experimentally demonstrate residual data recovery from erased blocks. Such findings point to widespread chip reuse and inadequate sanitization practices, creating significant risks of personal and corporate data leakage. Therefore, effective and reliable sanitization techniques are essential to ensure data security throughout a device's lifecycle. In this chapter, we provide the first experimental demonstration that block erase does not ensure secure data sanitization in 3D CT NAND flash memory, based on commercially available NAND devices. We further analyze the underlying physical mechanisms that lead to residual data leakage, examine the factors affecting its severity, and propose and evaluate practical mitigation techniques to minimize post-sanitization data remanence.

5.1 Key Ideas

In this chapter, we present the first experimental evidence that block erase is insufficient to securely sanitize modern 3D NAND flash memory. Using raw NAND access on commercial devices from multiple vendors and technology generations, we show that substantial residual information persists even after a full block erase. By exploiting analog state differences that survive logical sanitization, an attacker can recover a significant fraction of previously stored data using only standard NAND operations. Our key contributions are as follows:

- **Post-erase data recoverability.** We demonstrate that $\sim 80\%$ of previously stored data can be recovered from a fully block-erased 176-layer CT 3D NAND device with SLC storage. For a 64-layer CT SLC device, a slightly lower but still substantial recovery rate of $\sim 70\%$ is achieved, indicating that this vulnerability persists across technology generations.

- **Physical root-cause analysis.** We identify lateral charge migration and residual hole accumulation in the silicon-nitride charge-trap layer as the underlying mechanism for post-sanitization data leakage.

- **Leakage characterization.** Through controlled experiments, we quantify how retention time, temperature, device wear, stored data type, and vertical layer location influence the severity of data leakage.

- **Mitigation strategies.** We propose and evaluate practical, non-destructive countermeasures, including thermal-assisted sanitization and aging-aware all-zero refresh, that significantly reduce data remanence.

5.1.1 Threat Model

We consider an attacker with physical access to a NAND flash–based storage device that has been logically sanitized using a standard block erase operation (Figure 5.1(a)) to prior disposal, resale, or reuse (Figure 5.1(b)). The attacker's objective is to recover sensitive data that was previously stored on the device but is no longer accessible through normal system interfaces following sanitization. After acquiring the device, the attacker extracts the raw NAND flash memory chip (Figure 5.1(c)) and connects it to a custom or commodity flash controller that provides low-level access to the flash array (Figure 5.1(d)). This setup enables the attacker to issue standard NAND operations, including page read, page program, block erase commands, and read-offset operations, and to directly observe cell-level behavior influenced by previously stored data. In many storage devices, similar low-level access can also be achieved through controller debugging interfaces or chip-off forensic techniques. The attacker does not rely on invasive semiconductor probing, chip

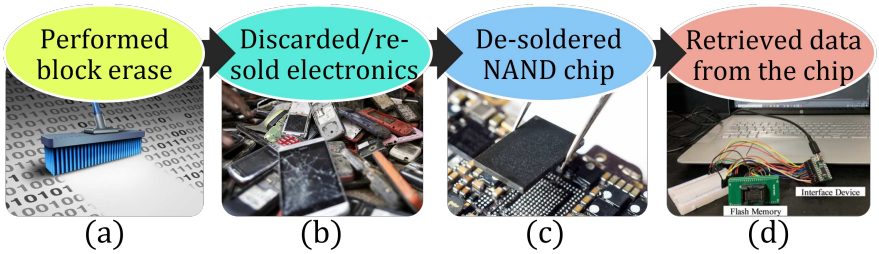


Figure 5.1: Threat model illustrating post-sanitization data recovery from NAND flash after device disposal. (a) Block erase is performed before device disposal. (b) The device is discarded, resold, or reused. (c) The raw NAND chip is desoldered from the device. (d) Data is retrieved using a low-level interface.

modification, or manufacturer-internal test modes; all operations used in our attack are supported by standard NAND flash interfaces.

This attacker model aligns with the laboratory-level adversary explicitly considered by NIST SP 800-88 for Purge sanitization, which block erase is intended to satisfy. Prior forensic studies have repeatedly shown that raw NAND flash chips from discarded, refurbished, or secondary-market devices are frequently accessible and reused, particularly in embedded, IoT, industrial, and consumer electronics. In these settings, block erase is commonly assumed to provide sufficient protection against post-decommissioning data recovery.

We further assume that the stored data are not encrypted. This assumption reflects common practice in resource-constrained systems that lack hardware encryption support and does not weaken the guarantees expected from logical sanitization mechanisms such as block erase. Even in systems that employ encryption, sanitization primitives are expected to remove residual physical information that could otherwise be exploited if encryption keys are compromised or improperly managed.

Under this threat model, we show that block erase eliminates logical data, but fails to remove underlying analog state differences in modern 3D CT NAND flash. An attacker with the raw flash access can exploit these residual physical artifacts to recover a substantial fraction of previously stored data, violating the security guarantees traditionally associated with logical sanitization.

5.1.2 Algorithm for Data Recovery

Figure 5.2 illustrates the flowchart of the proposed data recovery algorithm, while Algorithm 2 shows the corresponding ONFI command sequence. The recovery approach exploits residual analog state differences that persist after block erase by amplifying these differences through controlled thermal stress and probing them using read-offset operations.

The recovery process begins by selecting a block that has been logically sanitized using a block erase operation. A target page within the block is then programmed with an all-zero data pattern. This reprogramming establishes a uniform logical baseline across cells. The device subsequently undergoes a recovery bake at $T_{Rec.}$, which induces differential charge loss between cells that held 0s and cells that held 1s during the retention phase.

Following the recovery bake, shifted read operations are performed by adjusting the read reference voltage according to V_{ref} as $V_{ref} = V_{ref}^{0} + V_{offset}$, where V_{ref}^{0} denotes the default read reference voltage and V_{offset} is the offset voltage. According to the datasheet specifications for Micron 176-layer CT NAND, V_{offset} can be swept from -1280 mV ($V_{offset,\,min}$) to 1270 mV ($V_{offset,\,max}$) with a

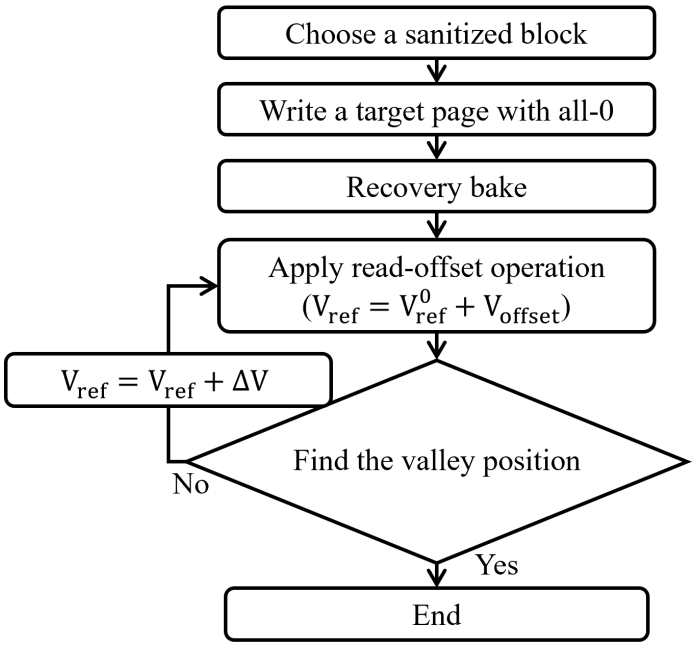


Figure 5.2: Flowchart of the proposed algorithm.

Algorithm 2 : Data recovery through read-offset operation

Initialization:
Program the target page(s) with an all-0 pattern
Perform recovery bake

Data recovery via read-offset operation:
Apply an adjusted V_{ref} ($V_{ref} = V^{0}_{ref} + V_{offset}$)
while true **do**
    Issue the NAND read-offset prefix command sequence: 0x2E $\rightarrow$ address cycles with the adjusted V_{ref} value
    Issue the NAND page read command sequence: 0x00 $\rightarrow$ address $\rightarrow$ 0x30 (READ CONFIRM)
    Record the readout data and find the valley position
    if the valley position is identified **then**
        break
    else
        Increase the adjusted V_{ref} by ΔV
    end if
end while

resolution of 10 mV (ΔV). The read-offset prefix command (0x2E) is then issued, followed by the address cycles with read voltage offset. Finally, the NAND page read command (0x00-0x30) is executed to retrieve the data. After each read operation, the readout data are recorded.

The use of a fixed 50% threshold implicitly assumes that the original data contains an approximately equal proportion of logic '0's and '1's, i.e., a uniform data distribution. While this assumption simplifies analysis for random datasets, it is not optimal in scenarios where data is biased toward zeros or ones. This limitation can be addressed using a more robust approach based on valley-search techniques to determine the optimal decision threshold between two overlapping distributions [103]. Specifically, the threshold is adapted by identifying the minimum (valley) point between the probability density functions of the two states. If the valley point has been identified, the recovery process is terminated; otherwise, the reference voltage is incremented by ΔV, i.e., $V^{n}_{ref} = V^{n-1}_{ref} + \Delta V$, and the process is repeated. This method improves classification accuracy under non-uniform data patterns and provides a more general criterion than a fixed 50% threshold. The read-offset prefix command (0x2E) is then issued, followed by the address cycles with read

voltage offset. Finally, the NAND page read command (0x00-0x30) is executed to retrieve the data. After each read operation, the readout data are recorded.

5.2 Experimental Results

This section describes the experimental flow used to induce and recover residual information in 5.2.1. Sections 5.2.2 to 5.2.6 present a progressive evaluation of recovery behavior, the device-physics behind the recovery process, and the device-level variability.

5.2.1 Experimental Flow

The experimental procedure consists of four stages: data programming, retention stress, block erase, and recovery stress, followed by read-offset reconstruction, as illustrated in Figure 5.3. These stages emulate the lifecycle of flash data, including normal device usage, long-term retention, sanitization prior to device disposal, and post-sanitization data recovery.

The experiment begins by subjecting a subset of flash blocks to repeated P/E cycling, denoted by N_{PE}, while maintaining other blocks in a factory-out (fresh) condition. This setup models real-world usage scenarios in which blocks within the same device experience different wear levels. Next, a grayscale Tesla image is programmed into a 16 kB flash page, while the remaining pages in the block are filled with pseudo-random data patterns. A read operation is then performed to verify programming correctness. The programmed Tesla image is visualized using a matrix plot, as illustrated in Figure 5.3, providing a reference for subsequent recovery analysis. Unless otherwise specified, all operations are conducted at 25°C.

Next, the device enters the retention phase. Rather than waiting for months or years under normal operating conditions, long-term retention effects are accelerated by baking the programmed chip at an elevated retention temperature, denoted as $T_{Ret.}$. Elevated temperature accelerates charge migration and trapping dynamics, enabling controlled emulation of extended retention within practical experimental time scales. Following retention, the experiment proceeds to the sanitization phase, where a block erase operation is performed. At this stage, the NAND flash–based device

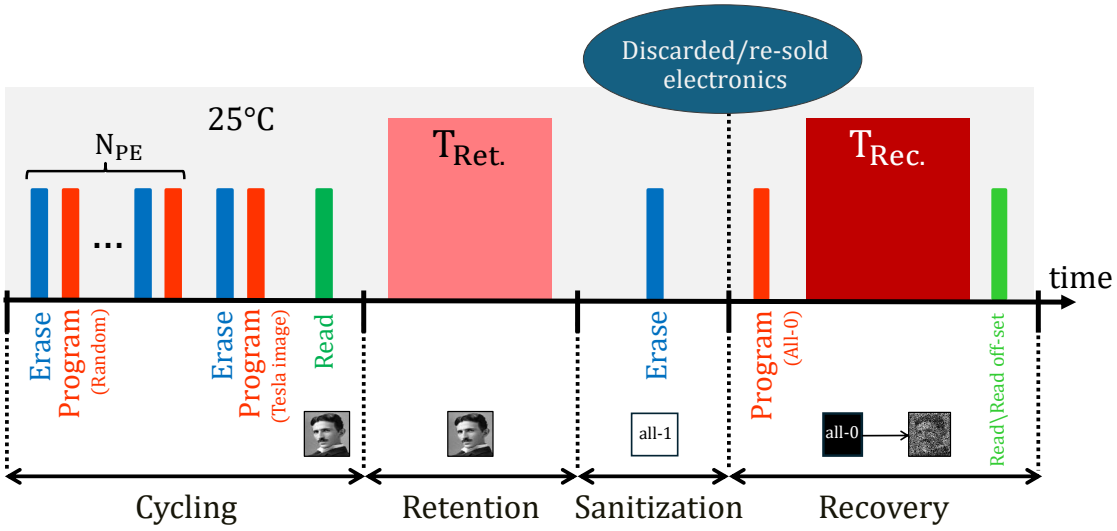


Figure 5.3: Experimental flow for data recovery in 3D CT NAND.

is assumed to have been discarded, resold, or otherwise transferred through secondary channels, making the raw NAND chip accessible to an attacker.

The experiment then enters the data recovery phase. To facilitate recovery, the sanitized pages are first programmed with an all-zero data pattern. The chip is subsequently maintained at an elevated recovery temperature ($T_{Rec.}$) to amplify residual physical state differences associated with previously stored data. Finally, standard read-offset operations are performed to reconstruct the previously stored image, as illustrated in Figure 5.3. The detailed recovery algorithm and its quantitative evaluation are presented in the following subsections.

5.2.2 Demonstration of Data Recovery

Figs. 5.4(a)–(f) demonstrate the life cycle of the NAND flash-based device, including device usage, block erase sanitization, and the proposed data recovery process. As a test pattern, a Tesla image is programmed into a selected NAND memory page, as shown in Figure 5.4(a). To emulate a real-world data retention scenario, in which user data is stored in flash memory for an extended period, the device is baked at a $T_{Ret.}$. This step represents accelerated aging, with the corresponding readout image shown in Figure 5.4(b). Note that the steps in Figs. 5.4(a)-(b) reflect common real-

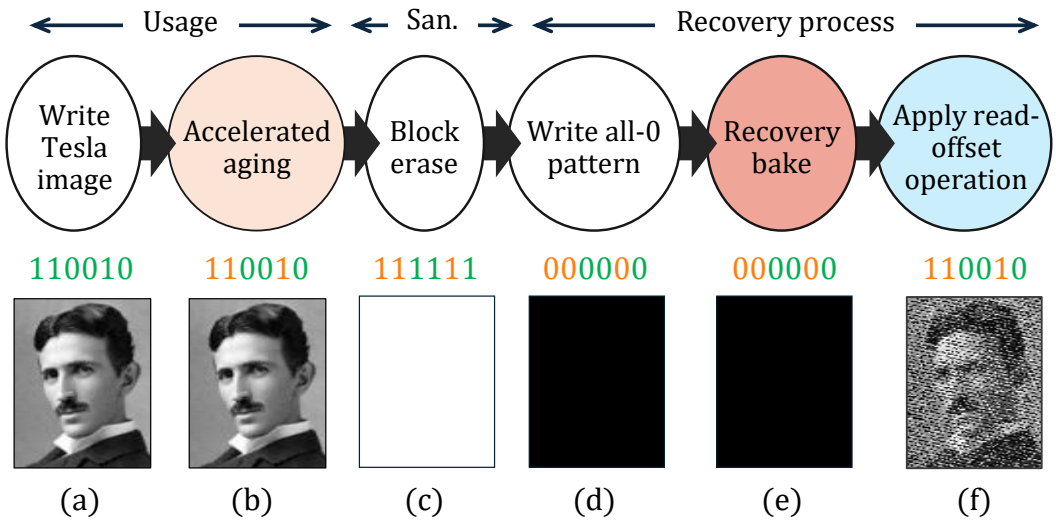


Figure 5.4: Data recovery process: (a) write a Tesla image into NAND flash memory, (b) accelerate aging by baking the chip at $T_{Ret.}$, (c) perform block erase operation, (d) write with all-zero pattern, (e) perform recovery bake, and (f) apply read-offset operation to retrieve the Tesla image.

world use cases in which storage devices are written infrequently and retain data for prolonged periods of time.

Following the accelerated aging phase, the programmed block undergoes logical sanitization via a block erase operation, according to the NIST SP 800-88 Rev. 1 recommendations, as shown in Figure 5.4(c). At this point, the logical contents appear fully erased, producing a uniformly white image. The attacker then obtains the device and initiates the data recovery process by re-programming the sanitized page with an all-zero data pattern. As illustrated in Figure 5.4(d), the white image is converted into a uniformly black image. Next, the attacker applies a recovery bake at $T_{Rec.}$ to amplify the analog state differences (Figure 5.4(e)). Finally, by sweeping the V_{ref} using a read-offset operation, the Tesla image can be recovered, as demonstrated in Figure 5.4(f).

To explain the analog state differences of flash cells throughout their life cycle, we use a binary bit map of the Tesla image for demonstration. In Figure 5.4(a), all cells are electrically stable ("good") with no residual hole accumulation, represented by the green bit values. During accelerated aging (Figure 5.4(b)), some cells experience retention-induced degradation ("bad") due to

residual hole accumulation, while others remain "good." In the figure, the orange values indicate degraded cells with residual holes accumulated near the spacer region.

During block erase sanitization, all cells are erased and return logic '1' upon readout, while their electrical properties remain unchanged, as shown in Figure 5.4(c). In the recovery process, an all-zero data pattern programming forces all cells to logic '0,' as shown in Figure 5.4(d). Although all cells are now logically at '0,' their underlying physical states are not identical due to the residual hole accumulation from the previously stored data. To amplify these physical differences, a recovery bake is applied (Figure 5.4(e)). This step accelerates electron-hole pair recombination and further separates the V_{th} distributions of "good" and "bad" cells. Subsequently, read-offset operations are performed (Figure 5.4(f)) to distinguish "good" and "bad" cells, enabling reconstruction of the original Tesla image. In the recovered image, green zeros correspond to cells with no accumulated residual holes, whereas orange ones indicate cells with residual holes associated with the original data. These differences stem from the originally stored data, which is altered by the long-term retention stress (accelerated aging phase), and are not removed by a standard block erase.

Crucially, "bad" cells holding a logic '0' lose charge faster than "good" cells holding a logic '0.' By deliberately sweeping V_{ref}, orange cells flip to logic '1,' while green cells remain at logic '0.' Reading back to the flash page after applying the read-offset operation reveals a clear reconstruction of the original image, as shown in Figure 5.4(f). This demonstration confirms that:

- Standard block erase sanitization removes logical content, but does not eliminate underlying physical state differences.
- These differences can be exploited through read-offset operations to recover sensitive information.

Together, these results highlight a fundamental limitation of current sanitization protocols for NAND flash and underscore the persistence of data leakage.

5.2.3 Optimizing Data Recovery

The data recovery process can be optimized through the read-offset operation by varying V_{offset}. Figure 5.5 quantifies the impact of V_{offset} on the data leakage percentage (η_{leak}), with V_{offset} swept from $V_{offset,\,min}$ to $V_{offset,\,max}$. The experiments are carried out in a block with $N_{PE} = 1$ after 4 days of data retention at $T_{Ret.}$ = 25°C, with the Tesla image stored. The full recovery procedure described in Algorithm 2 is then applied with $T_{Rec.}$ = 150°C for 60 minutes.

At low V_{offset} values, the page remains fully read as logic '0,' producing a dark image, as shown in the inset. In this case, $\eta_{leak} \approx 54\%$, which reflects the proportion of zero bits in the original Tesla image. As V_{offset} increases, η_{leak} rises and peaks at ≈ 0.7 V, where the recovered image exhibits the highest visual fidelity and a maximum η_{leak} of about 70%. Beyond this point, further increases in V_{offset} lead to a decline in η_{leak}, and the recovered image gradually fades toward an all-one (white) state.

The insets with V_{th} distributions in Figure 5.5 visually illustrate the observed behavior. Note that with the recovery bake, the original one cells lose stored charge faster than the original zero cells, leading to a progressive separation of the two distributions. For small V_{offset}, both the original

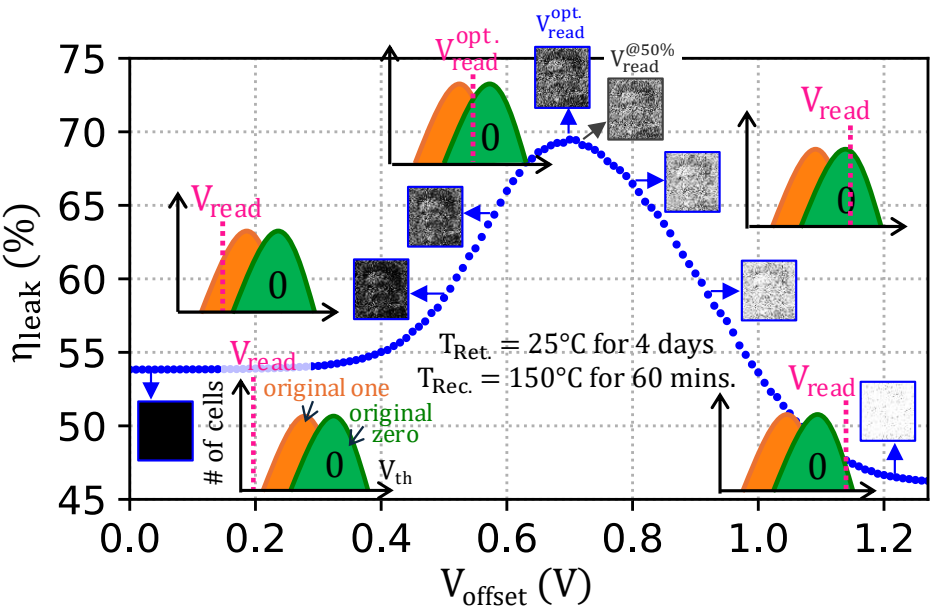


Figure 5.5: Data leakage percentage obtained at varying read-offset voltages.

zero and one cell distributions remain above the V_{ref}, producing an all-zero readout value. As V_{offset} is raised, this separation becomes more distinguishable, and optimal recovery occurs when the V_{ref} aligns near the crossover point of the two distributions, maximizing the distinguishability between the states. Across tested 3D NAND flash devices, the optimal V_{offset} consistently falls within a range of 400-1200 mV across different memory pages, depending on the recovery conditions, the device's usage level, and the recovery stress. While achieving 100% data recovery with the proposed recovery may not be feasible due to distribution overlap, recovery efficiencies of around 70% still pose a significant security risk for many applications.

We emphasize that elevated retention and recovery temperatures do not introduce new data leakage mechanisms. Instead, they accelerate physical processes, such as lateral hole migration and electron–hole recombination, that are also present at room temperatures. As shown in Section 5.2.5, data remnants become observable even at 25°C after sufficient retention time, confirming that the demonstrated leakage reflects real-world behavior rather than a temperature-induced artifact. Elevated temperatures are therefore used solely to compress time scales and enable controlled evaluation within practical experimental durations.

5.2.4 Device-Physics Explanation Behind Data Recovery

Figure 5.6 presents a device-physics explanation of the observed data leakage in CT 3D NAND flash memory, where the top and bottom rows represent the evolution of a "bad" and a "good" cell, respectively. Figure 5.6(a)-(b) schematically depicts the charge distribution in erased and programmed cells, respectively. Figure 5.6(a) shows the charge distribution after an erase operation, occurring via hot hole injection, while hot-carrier injection transfers electrons from the channel into the silicon nitride CT layer during programming (Figure 5.6(b)). During accelerated aging, holes migrate laterally within the nitride layer [73] and accumulate near the spacer region [72], as shown in Figure 5.6(c). In contrast, previously programmed cells experience electron migration along the nitride layer, as illustrated in Figure 5.6(d). As a result, even after a subsequent block erase operation, the net charge remaining in the nitride layer differs between cells that were

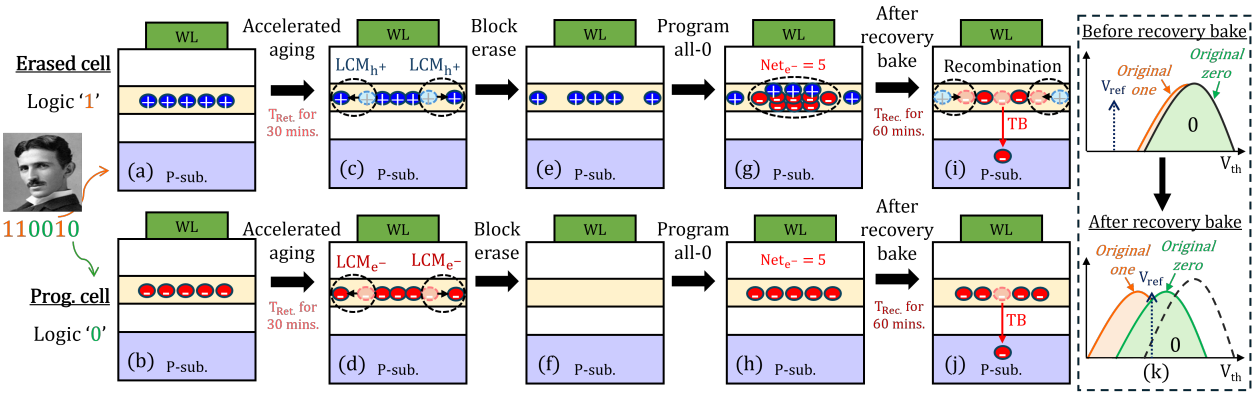


Figure 5.6: Root cause analysis: (a) erased cell and (b) programmed cell before accelerated aging; (c) and (d) the same cells immediately after the aging; (e) and (f) the same cells after block erase operation. Charge distribution in the storage layer after an all-zero program operation is shown in (g) and (h), and the overall charges are shown in (i) and (j). (k) V_{th} distribution shift during the recovery bake under the assumption that the proportions of original ones and zeros are equal (50% ones and 50% zeroes). Note that the V_{th} shift behavior remains valid even when the proportions of original ones and zeros are unequal.

originally erased and those that were originally programmed, as shown in Figure 5.6(e) and (f), respectively. The presence of residual holes in the nitride layer reduces the cell V_{th} [73]. As a result, standard block erase procedures leave residual holes intact, allowing analog traces of the originally stored data to persist in the memory array.

Figure 5.6(g)-(h) illustrates how these residual differences enable data recovery. The recovery process begins by programming all cells to logic '0,' such that cells that originally stored a '0' and those that originally stored a '1' are forced into the same digital state. Although they now appear identical at the logical level, residual holes in the nitride layer remain distinctly different: previously erased cells retain trapped holes (Figure 5.6(g)), whereas previously programmed cells contain none (Figure 5.6(h)). As shown in Figure 5.6(g), the presence of pre-existing trapped holes increases the total amount of injected charge required to reach a given number of stored electrons. For example, to achieve a net charge corresponding to five electrons, eight electrons must be injected to compensate for the three pre-existing holes beneath the WL. In contrast, previously programmed cells require only five injected electrons, as no residual holes are present. During the subsequent recovery bake, residual holes migrate within the nitride layer and recombine with stored electrons, leading to net charge loss and a reduction in V_{th} [104]. Figure 5.6(i) illustrates

this mechanism for a previously erased cell, where recombination removes trapped holes and some stored electrons, leaving a net charge of two electrons after recovery bake. In contrast, a previously programmed cell contains only newly injected electrons and no holes; therefore, its charge state remains largely unchanged during baking (from a net charge of five electrons to four electrons), as shown in Figure 5.6(j).

The final recovery step is a read-offset operation, which gradually shifts the V_{ref} to distinguish the V_{th} distributions of the original zero and original one cells in a controlled step-by-step manner. A key observation is that original '1' cells lose stored charge faster than original '0' cells. Consequently, the orange distribution (original '1') crosses V_{ref} sooner than the green distribution (original '0'). If the adjusted V_{ref} lies at the crossover point of these two distributions, as illustrated in Figure 5.6(k), the stored image can be effectively recovered. Since a cell is read as '1' when $V_{th} < V_{ref}$ and '0' otherwise, this optimal V_{ref} position ensures that most original ones are read as '1' and most original zeros are read as '0,' resulting in a recovered image that closely matches the original image.

Note that perfect data recovery (100% accuracy) may not be achievable with this approach due to the overlap in V_{th} distributions between original zeros and ones. Greater overlap results in lower recovery effectiveness. Furthermore, precisely controlling the adjusted V_{ref} to lie exactly between the two distributions presents an additional challenge. The next section introduces an algorithm for data recovery that addresses these challenges and examines various factors influencing recovery efficiency.

5.2.5 Factors Influencing Data Leakage

In this section, we discuss the factors that influence data leakage in 3D NAND flash memory. The η_{leak} is obtained by performing a bitwise comparison between the originally stored data (the Tesla image) and the data recovered using Algorithm 2. A value of η_{leak} = 100% corresponds to perfect reconstruction, while a fully erased (all '1's) or fully programmed (all '0's) yields η_{leak} around 50%, assuming the original image contains approximately equal proportions of '0's and

'1's. The data leakage percentage is influenced by several factors, including (i) the recovery stress (time and $T_{Rec.}$), (ii) the retention stress (time and $T_{Ret.}$), (iii) the layer location dependencies, (iv) the device wear-out level characterized by the N_{PE}, and (v) data pattern types. In this section, each of these factors is examined systematically.

Recovery Stress

Figure 5.7 shows the optimal data leakage percentage ($\eta_{leak}^{opt.}$) as a function of recovery bake duration and temperature. Note that the evaluation is carried out on a fresh block (N_{PE} = 1), with $T_{Ret.}$ fixed at 120°C for 30 minutes during accelerated aging. Green, blue, and red curves correspond to $T_{Rec.}$ values of 120°C, 150°C, and 180°C, respectively. Each data point represents a recovered image obtained during the recovery process. Under the Arrhenius acceleration model with an activation energy (E_a) of 1 eV [105], baking the chip at 120°C for 30 minutes corresponds to approximately nine months of storage at room temperature (25°C). The results in Figure 5.7 indicate that $\eta_{leak}^{opt.}$ increases with recovery time and then saturates after approximately 30–60 minutes, leading to improved visibility of the Tesla image. At higher recovery temperatures (150°C and 180°C), $\eta_{leak}^{opt.}$ increases from around 68% to about 76%, whereas a lower temperature (120°C) results in a lower saturation level of around 74–75%. For all subsequent evaluations, we choose $T_{Rec.}$ and the recovery bake time to be 150°C and 60 minutes, respectively.

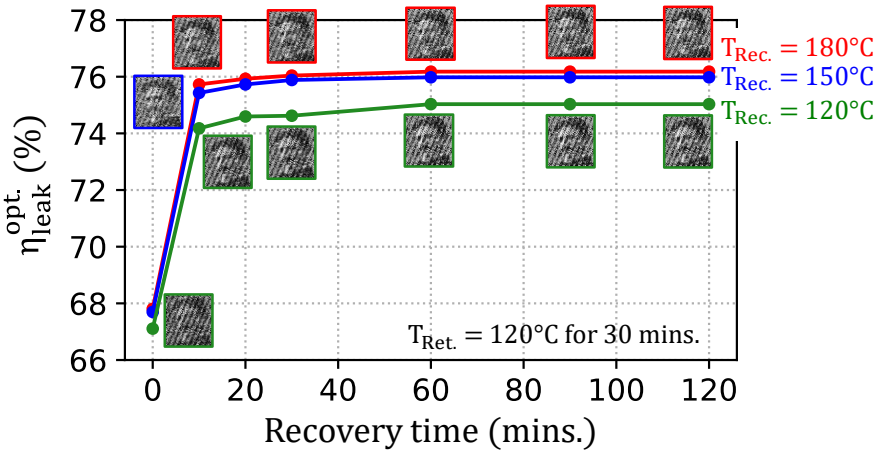


Figure 5.7: Percentage of $\eta_{leak}^{opt.}$ as a function of recovery bake temperature ($T_{Rec.}$) and duration.

Retention Stress

Figure 5.8 shows $\eta_{\text{leak}}^{\text{opt.}}$ for a fresh block baked under various retention durations and temperatures. Note that the retention duration on the x-axis is plotted on a logarithmic scale. In this evaluation, three chips, each programmed with the same Tesla image on fresh blocks, are baked for durations ranging from 30 minutes to 7 days and at temperatures from 25°C to 120°C. All three chips undergo recovery baking at 150°C for 60 minutes. The red, blue, and green curves in Figure 5.8 represent $T_{\text{Ret.}}$ of 120°C, 85°C, and 25°C, respectively. Based on these results, we find that data leakage increases with (i) longer retention durations and (ii) higher $T_{\text{Ret.}}$.

In Figure 5.8, we observe that, for all tested $T_{\text{Ret.}}$ values, data leakage increases with retention time. In particular, at $T_{\text{Ret.}}$ = 120°C, $\eta_{\text{leak}}^{\text{opt.}}$ increases from around 76.5% to about 80% when the chip is left for 30 minutes and 13 hours, respectively. This trend is primarily driven by the increasing density of residual holes near the spacer region. As retention time increases, more holes migrate and accumulate near the spacer region, resulting in a high density of trapped holes that can recombine with electrons during the recovery phase. Because the diffusion kinetics of holes in the nitride layer are inherently slow, significant time is required for substantial charge accumulation.

Retention temperature also plays a critical role in data leakage (Figure 5.8). Higher retention temperatures result in a greater density of holes migrating toward the spacer region, thereby in-

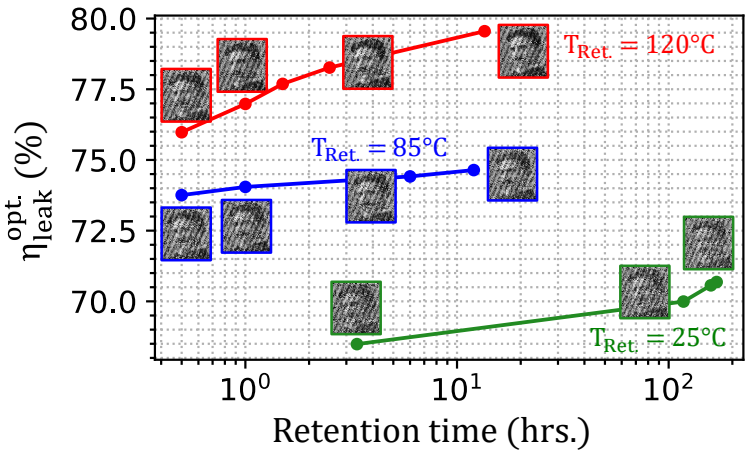


Figure 5.8: Percentage of $\eta_{\text{leak}}^{\text{opt.}}$ as a function of retention bake temperature ($T_{\text{Ret.}}$) and duration in a fresh block ($N_{\text{PE}} = 1$), with $T_{\text{Rec.}}$ = 150°C for 60 minutes.

creasing the $\eta_{\text{leak}}^{\text{opt.}}$. This behavior is expected, as the diffusion coefficient increases exponentially with temperature.

Further evaluation at the real-world operating temperature of 25°C shows that the remnants of the Tesla image become visible after approximately three hours of room-temperature data retention. These results confirm that the same underlying physics governs data leakage: LCM_{h^+} and electron–hole pair recombination in the nitride layer. Elevated temperatures and prolonged retention time merely accelerate these mechanisms, amplifying the observed data leakage.

Layer Location Dependencies

Figure 5.9(a) shows the percentage of data leakage at different locations of the 3D NAND layer after accelerated aging at $T_{\text{Rec.}}$ = 120°C for 30 minutes. The black, blue, orange, green, and red curves represent recovery durations of 0, 10, 20, 30, and 60 minutes at $T_{\text{Rec.}}$ = 150°C, respectively. The results reveal a strong dependence of data leakage on layer position. Notably, after reading with the optimal adjusted V_{ref}, some layers exhibit visible remnants of the Tesla image even without a recovery bake (0-minute recovery). This "zero-minute" recovery is particularly concerning, as it can be executed online without physical access to the device. With increasing recovery time, the

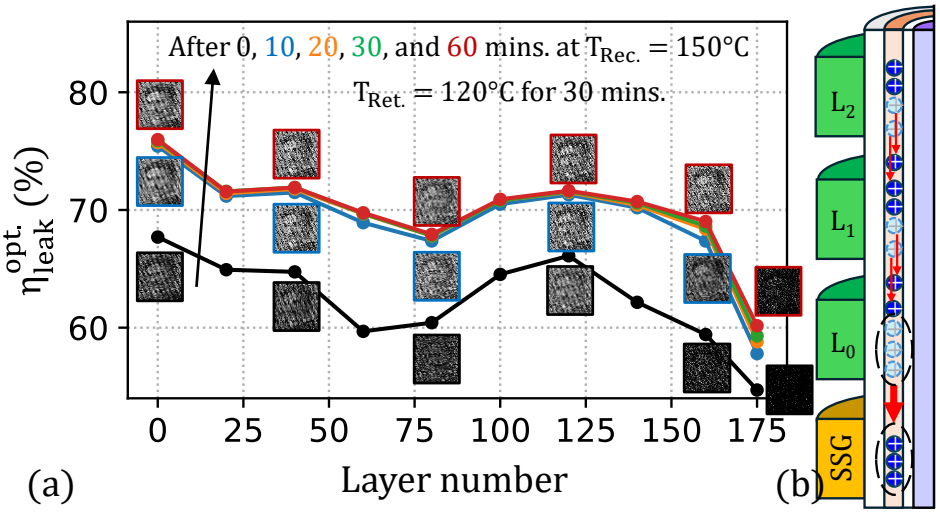


Figure 5.9: (a) Percentage of $\eta_{\text{leak}}^{\text{opt.}}$ when storing the Tesla image in different 3D NAND stack layers. (b) Schematic of a NAND string showing residual holes laterally migrating toward the select-gate side, making the layers closest to the select gate the most vulnerable [72].

recovered images become progressively clearer, including pages that showed no visible remnants at time zero.

Furthermore, we observe that the bottom-most layer (layer-0) exhibits the highest data leakage, whereas the remaining layers show comparatively lower leakage. This behavior can be attributed to the retention phase, during which holes in the nitride layer of all WLs laterally migrate toward the select-gate regions, as shown in Figure 5.9(b). In particular, holes in layer-0 migrate first toward the select gate (SSG), followed by those in layer-1 and layer-2, respectively [72]. This migration leads to increased residual hole accumulation near the select gates, which in turn enhances electron–hole recombination during the subsequent recovery bake phase, making the layers closest to the select gate the most vulnerable.

Device Wear-Out Levels

Figure 5.10 compares data leakage between a fresh block (N_{PE} = 1) and a moderately worn block (N_{PE} = 15k), with the x-axis representing the recovery bake time. In this evaluation, all flash blocks are subjected to the same recovery temperature ($T_{Rec.}$ = 150°C) and the same retention temperature and duration ($T_{Ret.}$ = 120°C for 30 minutes). The layer number is fixed at layer 80. The blue line represents N_{PE} = 1, while the red line represents N_{PE} = 15k. The results show that, after the first 10 minutes of recovery baking, the moderately worn block consistently exhibits higher data leakage than the fresh block, suggesting that data leakage vulnerability increases with device wear. Note that the endurance specified in the datasheet is 60k P/E cycles, meaning that both N_{PE} = 1 and N_{PE} = 15k are still far from the device's end-of-life, highlighting that data leakage can occur even at the very beginning of a device's lifetime.

The pronounced leakage observed in the block with N_{PE} = 15k can be attributed to a higher density of residual holes in the nitride layer, which subsequently amplifies the contrast between the originally stored '0' and '1' states. This increased contrast enhances the effectiveness of the read-offset operation in separating the two groups of bits.

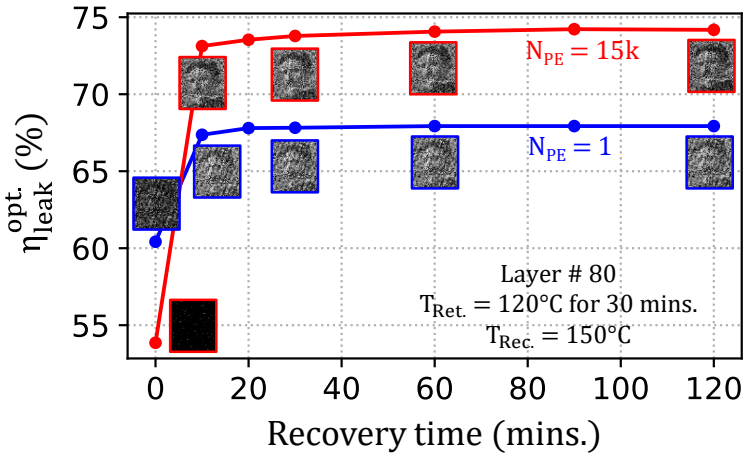


Figure 5.10: Percentage of $\eta^{opt.}_{leak}$ obtained with different worn-out levels.

Data Pattern Types

Figure 5.11 illustrates recovered data corresponding to different originally stored data patterns: (a) a grayscale Tesla image, (b) a grayscale Queen Elizabeth image, (c) pseudo-random binary data, and (d) a text file containing ASCII characters. In this evaluation, $T_{Ret.}$ is fixed at 120°C for 30 minutes, while $T_{Rec.}$ is set at 150°C for 60 minutes. Note that the $\eta^{opt.}_{leak}$ for these four data types is very similar, ranging from 76% to 78%.

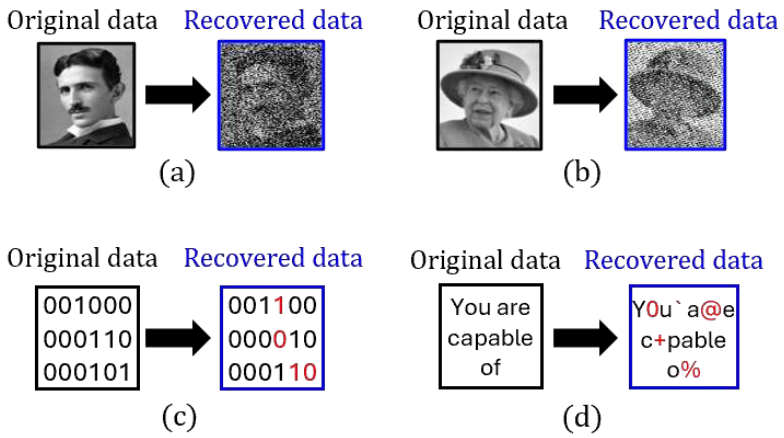


Figure 5.11: Recovered data corresponding to different originally stored data patterns: (a) grayscale Tesla image, (b) grayscale Queen Elizabeth image, (c) pseudo-random binary data pattern, and (d) text file containing ASCII characters.

For grayscale images in Figs. 5.11(a)-(b), the recovered data exhibit noticeable noise; however, key visual features remain clearly identifiable for both the Tesla and Queen Elizabeth images, demonstrating robustness across different image types. Binary patterns from pseudo-random data are also recoverable (Figure 5.11(c)). For text files containing ASCII characters (Figure 5.11(d)), the recovered data show higher corruption, although some characters remain readable. Further improvements are possible through post-processing techniques, such as frequency-based character analysis, to enhance text recovery. Overall, while most of the results are demonstrated using image data, the proposed recovery technique is broadly applicable to multiple data types.

5.2.6 Multi-Chip Evaluations

We implemented the proposed recovery algorithm on several COTS 3D NAND flash devices with the SLC storage configuration. Table 5.1 summarizes the evaluated samples, which span 64-layer to 176-layer CT NAND technologies from multiple vendors and cover a range of wear conditions. Across all tested devices, measurable data leakage is consistently observed under the specified recovery conditions. Moreover, the extent of leakage increases with device wear: for example, in the 64-layer CT device, the leakage percentage rises from 62.4% in a fresh block (N_{PE} = 1) to 68.6% after N_{PE} = 30k. A similar leakage behavior is also observed in the 176-layer CT device, indicating that the vulnerability persists across technology generations. Note that according to the datasheet, the 64-layer CT device reaches its end-of-life at 30k P/E cycles.

Table 5.1: Summary of data leakage across various 3D NAND flash samples under different recovery conditions. Note that $T_{Ret.}$ is fixed to be 120°C for 30 minutes.

Sample	**Part number**	**Vendor**	**Recovery bake condition ($T_{Rec.}$)**	N_{PE}	$\eta_{leak}^{opt.}$
64-layer CT	YMN08TE1B1DC3B	YMTC	120°C for 60 minutes	1	62.4%
64-layer CT	YMN08TE1B1DC3B	YMTC	120°C for 60 minutes	30k	68.6%
176-layer CT	MT29F1T08EELKEJ4	Micron	150°C for 60 minutes	1	76.5%
176-layer CT	MT29F1T08EELKEJ4	Micron	150°C for 60 minutes	30k	78.0%

5.3 Mitigation Strategies

In this section, we propose possible strategies to mitigate the effects of data leakage in 3D CT NAND flash memory.

Aging-aware all-zero refresh: A practical system-level mitigation is aging-aware all-zero refresh. Data leakage arises primarily when information remains in the same physical block for extended periods, allowing residual holes to accumulate over time. To mitigate this risk, the storage system can periodically migrate live data to new physical locations and enforce an all-zero state on vacated pages through controlled overwrite operations. Writing an all-zero pattern promotes the recombination of accumulated holes with programmed electrons, thereby securely eliminating residual traces of the original data. As a result, no single physical location is allowed to accumulate charge imbalances that could later be exploited by an adversary.

This policy must be carefully designed to balance security with flash endurance, as aggressive overwriting or frequent all-zeroing can accelerate wear-out. Endurance-aware scheduling, such as selectively zeroing only stale or long-resident blocks, or piggybacking refresh operations onto normal garbage-collection or wear-leveling events, can significantly reduce unnecessary P/E cycles while still providing strong protection against data remanence and leakage.

Thermal-assisted sanitization practices: Another effective countermeasure is the adoption of enhanced sanitization protocols prior to device disposal or repurposing. Conventional block-erase operations are insufficient to fully eliminate the remanent charge, as residual holes may persist even after erase. To improve sanitization efficacy, we propose a thermal-assisted sanitization approach in which the chip is first programmed with an all-zero data pattern and then subjected to a controlled thermal bake before performing the final block erase.

Baking the device while storing an all-zero pattern leverages thermal energy to accelerate charge redistribution and recombination within the storage medium, thereby neutralizing residual holes that could otherwise be exploited by an adversary to reconstruct previously stored information. Figure 5.12 plots the optimized leakage metric, $\eta_{\text{leak}}^{\text{opt.}}$, as a function of recovery time, comparing standard block erase with the proposed thermal-assisted sanitization.

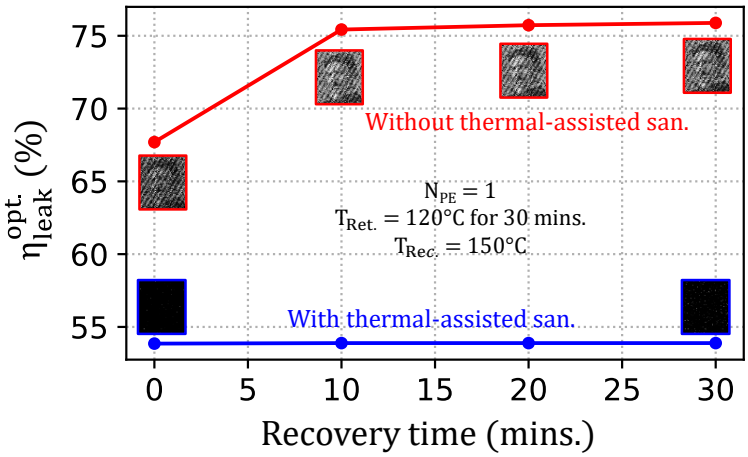


Figure 5.12: Percentage of $\eta^{opt.}_{leak}$ when the thermal-assisted sanitization method is applied, compared with when it is not used.

The experiment was conducted on the same memory chip using two fresh blocks. A Tesla image was written to both blocks and subjected to a retention bake to induce charge degradation. Sanitization was then performed using two different methods: one block was sanitized using only a conventional block erase (red curve), while the other block was sanitized by first baking the chip with an all-zero data pattern, followed by a block erase. The results demonstrate a substantial reduction in data leakage when thermal-assisted sanitization is applied. Depending on the required security level, the proposed method can be extended to multiple passes using randomized data patterns prior to the final erase, further increasing the cost and complexity of forensic data recovery. Although the proposed thermal-assisted sanitization technique may incur time overhead, the device remains reusable through annealing-based restoration processes that help extend its lifetime [76]. However, destructive approaches permanently damage the storage medium, rendering it unusable. This trade-off enables a non-destructive sanitization strategy that balances data security with device longevity

Compared to destructive sanitization methods such as shredding, pulverizing, or degaussing, these approaches have two key advantages: (1) they preserve the functionality of the memory device, which is valuable for scenarios where media re-purposing or resale is desired, and (2) they are more scalable, avoiding the logistical challenges and costs of physically destroying large

numbers of chips. While destructive sanitization may still be necessary for the most sensitive environments, thermal-assisted erasure and random-pattern writing provide a non-destructive yet effective means of mitigating imprinting risks in many practical applications.

5.4 Conclusion

This paper presented the first experimental study of post-sanitization data leakage in commercial NAND flash memory. We demonstrated that CT 3D NAND flash devices are vulnerable to data leakage, where residual holes in the silicon nitride storage layer persist after standard block erase operation. Using the controlled read-offset operation, we showed that around 70-80% of the original content can be recovered from sanitized blocks when the devices operate in SLC mode. The severity of this leakage is strongly influenced by recovery stress, retention stress, the location of layers, and P/E wear. Through device-physics analysis, we identified that the mechanism behind the data leakage is the recombination of residual holes and electrons in the nitride charge trapping layer, which a standard block erase cannot eliminate. These findings highlight a critical gap in current sanitization practices: the block erase operation is insufficient for secure data removal in CT 3D NAND. To address this gap, we outlined system-level and device-level countermeasures, including periodic aging-aware all-zero refresh and thermal-assisted sanitization, which can mitigate the data leakage without requiring destructive methods.

Chapter 6

Block Erase Vulnerabilities in FG 3D SLC NAND

As discussed in the previous chapter, block erase in NAND flash is generally the most effective logical sanitization method, as it removes stored charge from all cells, restoring them to a uniform erased state. However, the security of the block erase operation has not been thoroughly evaluated for 3D FG NAND flash memory, particularly under end-of-life conditions. In end-of-life NAND cells, repeated P/E cycles create defect states in the oxide layer, leading to both defect-repair and non-repair mechanisms during data retention. These effects can create residual physical variations, potentially enabling previously stored data to persist despite conventional assumptions of complete sanitization. In this chapter, we present the first experimental evidence that block erase is insufficient to securely sanitize 3D FG NAND flash memory, using commercially available NAND chips from multiple vendors and across multiple technology generations. We further investigate the underlying physical mechanisms responsible for this leakage, identify factors that influence its severity, and propose and evaluate practical mitigation strategies to reduce post-sanitization data leakage.

6.1 Key Ideas

By exploiting physical state differences that survive logical sanitization, an attacker can recover a significant fraction of previously stored data using only standard NAND operations. Our key contributions are as follows:

- **Post-erase data recoverability:** We show that approximately 70% of previously stored data can be recovered from an end-of-life, post-erased block of 64-layer FG 3D NAND flash memory when the data is stored in the SLC page. Similar recovery is also observed in an end-of-life 96-layer FG SLC page, indicating that this vulnerability persists across technology generations.

- **Physical root-cause analysis:** We identify the repair of defect states in the oxide layer, referred to as a broken-bond reformation, which reduces oxide trap density, as the primary mechanism responsible for post-erase data leakage.

- **Leakage characterization:** We quantify how recovery duration, recovery temperature, retention stress, device wear, stored data type, and vertical layer location influence the extent of residual data.

- **Mitigation strategies:** We propose practical and non-destructive countermeasures, including sanitization with inverted stored data, thermally assisted sanitization, and aging-aware all-zero refresh, that significantly reduce data remanence.

6.1.1 Threat Model

We consider an attacker with physical access to a NAND flash–based storage device at its end of life that has been logically sanitized using a standard block erase operation (Figure 6.1(a)) before disposal, resale, or reuse (Figure 6.1(b)). The attacker's objective is to recover sensitive data that was previously stored on the device but is no longer accessible through normal system interfaces due to logical sanitization. After acquiring the device, the attacker desolders and removes the raw NAND chip from the discarded device (Figure 6.1(c)) and connects it to a controller that provides low-level access (Figure 6.1(d)). Using standard NAND commands such as read, program, erase,

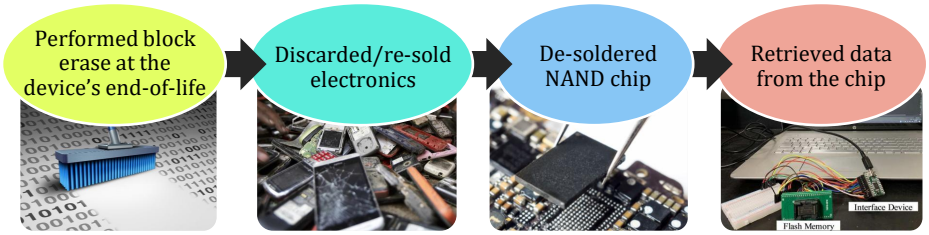


Figure 6.1: Threat model illustrating post-sanitization data recovery from NAND flash using standard NAND operations after device disposal. (a) Block erase is performed at the device's end-of-life prior to disposal. (b) The device is discarded, resold, or reused. (c) The raw NAND chip is desoldered from the device. (d) Data is retrieved using a low-level interface.

and read-offset, the attacker can directly observe residual cell-level effects from previously stored data, with no invasive semiconductor probing, chip modification, or manufacturer-internal test modes required.

We assume the stored data is not encrypted, reflecting common practice in resource-constrained systems without hardware encryption support. This does not weaken the guarantees expected from logical sanitization (i.e., block erase). Even when encryption is used, sanitization primitives are still expected to eliminate residual physical information that could be exploited if keys are compromised or mismanaged. Under this threat model, block erase removes logical data but not residual analog state differences in 3D FG NAND. An attacker with raw flash access can exploit these artifacts to recover a substantial portion of previously stored data.

6.1.2 Algorithm for Data Recovery

Figure 5.2 illustrates the flowchart of the proposed data recovery technique, while the corresponding NAND command sequence used in this chapter is detailed in Algorithm 3. The recovery process begins by selecting a target page within an erased block and programming it with an all-

Algorithm 3 : Data recovery through read-offset operation

Initialization:
Program the target page(s) with an all-zero pattern
Perform recovery bake

Read-offset sweep for data recovery:
Apply an adjusted V_{ref} as $V_{ref} = V_{ref}^{0} + V_{offset}$
while true **do**
    Issue the read-offset configuration command: 0xEF (SET FEATURES) $\rightarrow$ feature address (0xA4) $\rightarrow$ P1 subfeature value
    Issue the NAND page read command sequence: 0x00 $\rightarrow$ address $\rightarrow$ 0x30 (READ CONFIRM)
    Record the readout data and find the valley position
    if the valley position is identified **then**
        break
    else
        Increase the adjusted V_{ref} by ΔV
    end if
end while

zero data pattern. Programming all cells to zero establishes a uniform logical baseline across the page. Next, the device undergoes a recovery bake, which induces differential charge loss between cells that were previously programmed (stored 0s) and those that were previously erased (stored 1s). Following the recovery bake, read-offset operations are performed by shifting the V_{ref}, such that $V_{ref} = V^0_{ref} + V_{offset}$, where V^0_{ref} denotes the default reference voltage and V_{offset} is the applied offset. According to the datasheet specifications for Micron 3D 64-layer FG NAND flash memory, V_{offset} can be swept from -960 mV ($V_{offset,min}$) to 952.5 mV ($V_{offset,max}$) with a resolution of 7.5 mV (ΔV). The SET FEATURES command (0xEF) is then issued, followed by the feature address and the P1 subfeature value. Finally, the NAND page read command (0x00–0x30) is executed to retrieve the data, the readout is recorded, and the valley position is computed. The recovery process is terminated if the valley position is identified. Otherwise, the V_{ref} is increased by ΔV, i.e., $V^n_{ref} = V^{n-1}_{ref} + \Delta V$, and the process is repeated.

6.2 Experimental Results

This section details the experimental flow, data recovery demonstration and optimization, root-cause analysis, and a comprehensive evaluation of the factors influencing data imprinting and data leakage.

6.2.1 Experimental flow

The experimental procedure, illustrated in Figure 6.2, consists of four phases: P/E cycling to emulate user usage, data retention, block erase, and data recovery. These stages emulate the lifecycle of flash data, including normal device usage, long-term retention, sanitization prior to device disposal, and post-sanitization data recovery.

In the first phase (Cycling), a subset of flash blocks undergoes repeated P/E cycling up to the end-of-life specified in the datasheet (N_{PE} = 3k). Additional blocks are kept in intermediate cycle counts (e.g., N_{PE} = 1, 100, 500, 1k, etc.) to represent different wear levels. After cycling, a 16 kB Tesla image is programmed into a few flash memory pages, while the remaining pages in the

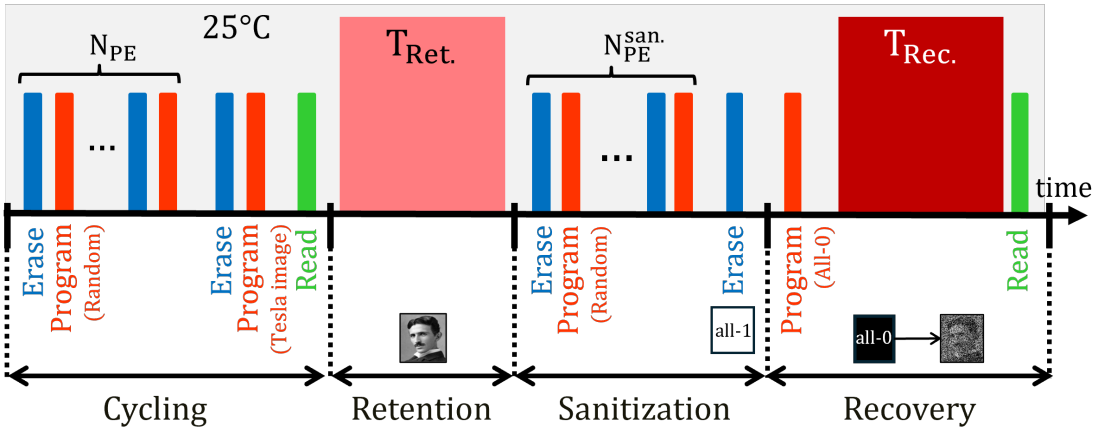


Figure 6.2: Experimental flow for data recovery in 3D FG SLC NAND.

block are filled with pseudo-random data. A read operation is performed to verify the correct programming. The programmed Tesla image is visualized as a matrix plot, as shown in Figure 6.2, serving as a reference for the subsequent recovery analysis. Unless otherwise specified, all experiments are conducted at room temperature (25°C).

The device then enters the retention phase. Rather than waiting for months or years under room temperature, long-term retention effects are accelerated by baking the programmed chip at an elevated T_{Ret}. This elevated temperature accelerates charge-loss mechanisms and the defect-repair process, enabling controlled emulation of extended retention within practical experimental timescales. The device then undergoes additional P/E cycles with a new set of pseudo-random data patterns ($N_{PE}^{san.}$). Next, the experiment proceeds to the sanitization phase, during which a block erase operation is performed. At this stage, the raw NAND chip is assumed to be discarded or resold, making it accessible to a potential attacker.

In the recovery phase, the sanitized page is first programmed with an all-zero data pattern. After that, the chip is held at an elevated $T_{Rec.}$ to amplify the physical state differences associated with previously stored data. Then standard read-offset operations are applied to reconstruct the original image, as illustrated in Figure 6.2. The recovery algorithm and its quantitative evaluation are detailed in the following subsections.

6.2.2 Demonstration of Data Recovery

Figures 6.3(a)–(g) illustrate the lifecycle of a NAND flash–based device, including user usage at end-of-life operation, data sanitization, and the proposed data recovery process. Although the proposed recovery technique is applicable to various types of stored data, a grayscale Tesla image is used for clear visualization. During end-of-life operation, a Tesla image is programmed into a selected NAND flash memory page, and the corresponding readout is shown in Figure 6.3(a). To emulate a real-world data retention scenario, in which user data remains stored in flash memory for an extended period, the device undergoes accelerated aging by baking at $T_{Ret.}$. The resulting readout after this step is illustrated in Figure 6.3(b). These steps reflect common real-world use cases, where storage devices are written infrequently and retain data for prolonged durations.

The device then enters the sanitization phase. The programmed block undergoes five cycles of P/E operations using new pseudo-random data patterns (Figure 6.3(c)), followed by logical sanitization via a block erase operation (Figure 6.3(d)), in accordance with the NIST SP 800-88 Rev. 1 recommendations. At this point, the logical contents of the Tesla image transition from a noisy pattern (Figure 6.3(c)) to a fully erased state, appearing as a uniformly white image (Figure 6.3(d)).

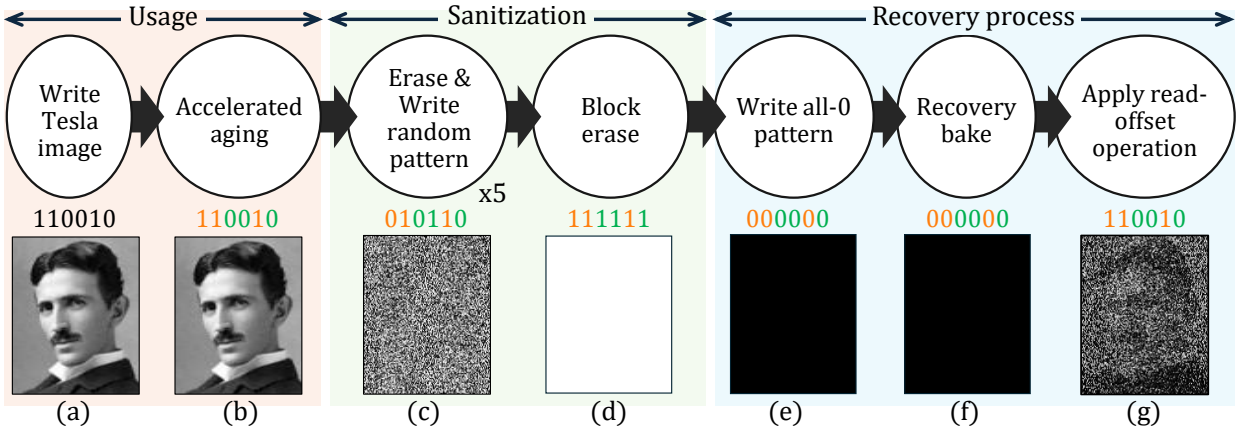


Figure 6.3: Process of data imprinting and recovery in an end-of-life NAND Device: (a) write a Tesla image into NAND flash memory, (b) accelerate aging by baking the chip at 120°C for 30 minutes, (c) repeatedly perform block erase and write random data pattern for five times, (d) perform block erase, (e) write with all-zero pattern, (f) perform recovery bake, and (g) apply read-offset operation to retrieve the Tesla image.

The data recovery process begins by reprogramming the sanitized page with an all-zero data pattern, converting the fully erased state into a fully programmed state and producing a uniformly black image (Figure 6.3(e)). A recovery bake is then applied at $T_{Rec.}$ to amplify the physical differences between previously erased and previously programmed cells. At this stage, a default read operation may reveal little to no visible data remanence due to non-optimized read conditions, resulting in the black image observed in Figure 6.3(f). Subsequently, to enhance data leakage, read-offset operations are performed by sweeping V_{offset} from $V_{offset,min}$ to $V_{offset,max}$. This enables successful reconstruction of the original Tesla image, as shown in Figure 6.3(g).

To illustrate the evolution of flash cell physical states, we use the binary bitmap of the Tesla image, consisting of logic '0' and '1,' as shown in Figure 6.3(a). During accelerated aging (Figure 6.3(b)), retention stress causes some cells to degrade ("bad"), while others improve ("good") due to defect repair. In the figure, the orange bits indicate degraded cells with high defect density, and the green bits indicate cells with lower defect density. During the sanitization phase (Figure 6.3(c)–(d)), repeated P/E cycles followed by block erase remove the logical data. However, the underlying physical differences between "good" and "bad" cells remain.

In the recovery phase, all cells are first programmed to logic '0' (Figure 6.3(e)), creating a uniform logical state while preserving these physical differences. A recovery bake is then applied to accelerate charge loss and increase the separation in V_{th} between cells (Figure 6.3(f)). However, these differences are not visible under the default read operation. Finally, read-offset operations are applied by sweeping V_{ref}. Due to faster charge loss, "bad" cells storing logic '0' transition to logic '1' earlier than "good" cells, causing the orange cells to flip to '1' while the green cells remain at '0.' Reading back to the flash page after applying the read-offset operation reveals a partial reconstruction of the original image, as shown in Figure 6.4(g). These results demonstrate that standard block erase removes logical data but leaves exploitable physical-state differences, highlighting a fundamental limitation of current NAND flash sanitization protocols and the persistence of data leakage.

6.2.3 Optimizing Data Recovery

This section quantifies the impact of V_{offset} on the η_{leak} when V_{offset} is swept from $V_{offset,min}$ to $V_{offset,max}$. The experiment is performed on an end-of-life block (N_{PE} = 3k), following the procedure illustrated in Figure 6.3 with $N_{PE}^{san.}$ set to 0 to illustrate the optimal data leakage. During the accelerated aging phase, the Tesla image is stored at $T_{Ret.} = 120°C$ for 60 minutes. Under the Arrhenius acceleration model with an E_a of 1 eV [105], this condition corresponds to approximately 17 months of storage at room temperature (25°C). After accelerated aging, the device undergoes the sanitization phase, followed by the full recovery procedure described in Algorithm 3, including a recovery bake at $T_{Rec.} = 120°C$ for 60 minutes. The resulting η_{leak} values for each V_{offset} are shown along with their corresponding recovered images in Figure 6.4.

At low V_{offset} values, the page remains fully read as logic '0,' producing a dark image, as shown in the inset. In this case, $\eta_{leak} \approx 50\%$, corresponding to the proportion of zero bits in the original Tesla image. As V_{offset} increases, η_{leak} rises and reaches a peak at approximately 202.5 mV, where the recovered image exhibits the highest visual fidelity with a maximum η_{leak} of about 70.40%. A comparable recovery level is observed at V_{offset} = 247.5 mV. Beyond these points, further increases

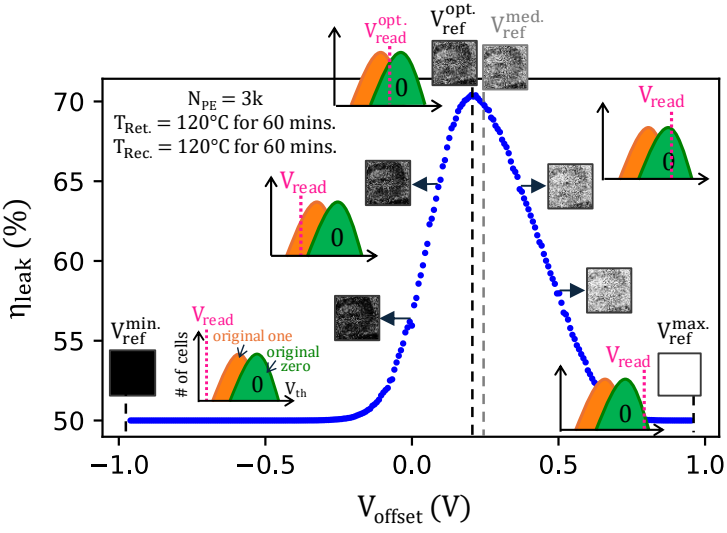


Figure 6.4: Data leakage percentage obtained at varying read-offset voltages.

in V_{offset} lead to a decline in η_{leak}, and the recovered image gradually transitions toward an all-one (white) state.

The schematic cell V_{th} distributions further illustrate the observed behavior. Note that with the recovery bake, the original one cells lose stored charge faster than the original zero cells, leading to a progressive separation of the two distributions. For small V_{offset}, both the original zero and one cell distributions remain above the V_{ref}, producing an all-zero readout value. As V_{offset} is raised, this separation becomes more distinguishable. The optimal recovery occurs when the V_{ref} aligns near the crossover point of the two distributions, maximizing the distinguishability between the states.

Across tested 3D NAND flash devices, the optimal V_{offset} consistently falls within a range of 0-500 mV across different memory pages, depending on the recovery conditions, the device's usage level, and the recovery stress. While achieving 100% data recovery with the proposed recovery may not be feasible due to distribution overlap, recovery efficiencies of around 70% still pose a significant security risk for many applications.

6.2.4 Device-Physics Explanation Behind Data Recovery

Figure 6.5 presents a device-physics explanation of the observed data leakage in the end-of-life 3D FG NAND flash memory cells. Figures 6.5(a)–(b) schematically illustrate the physical states of an erased cell (logic '1') and a programmed cell (logic '0') at end-of-life. Due to extensive P/E cycling, both cells exhibit a high density of oxide defect states. However, their electrical conditions differ significantly. As shown in Figure 6.5(a), the erased cell contains a large number of defect states, which induce an inversion region in the channel. In contrast, the programmed cell (Figure 6.5(b)) includes both stored charge in the FG layer and trapped electrons in the oxide layer, leading to the formation of a depletion region in the poly-silicon channel. These trapped electrons neutralize the positive trap charge states, converting the defect states into neutral trap states.

During the accelerated aging phase, the two cells evolve differently. In the erased cell, channel inversion inhibits defect repair by suppressing the recombination of broken bonds (Figure 6.5(c)).

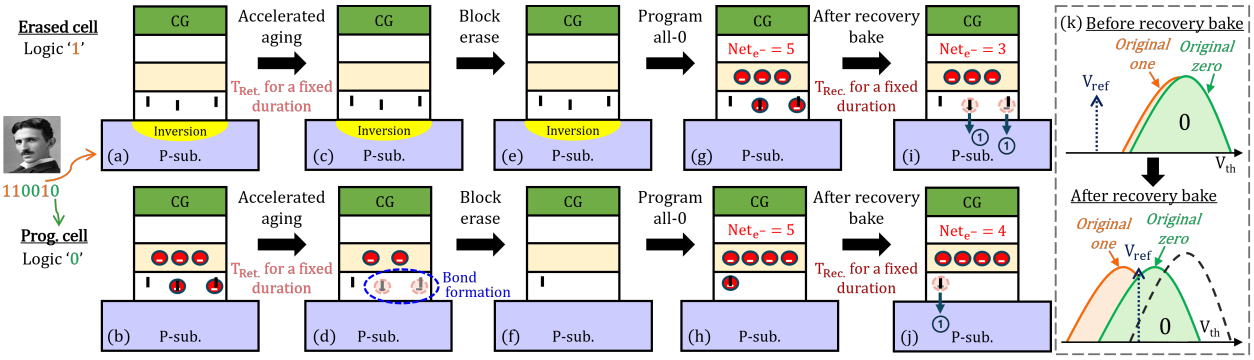


Figure 6.5: Root cause analysis: (a) erased cell and (b) programmed cell before accelerated aging; (c) and (d) the same cells immediately after aging; (e) and (f) the same cells following a block erase operation; and (g) and (h) the same cells after an all-zero program operation. The overall charges after the recovery bake are shown in (i) and (j). (k) illustrates the V_{th} distribution shift during the recovery bake under the assumption of equal proportions of original ones and zeros (50% ones and 50% zeros). Note that the V_{th} shift behavior remains valid even when the proportions of original ones and zeros are unequal.

In contrast, in the programmed cell, trapped charges facilitate the repair of broken bonds in the oxide, thereby reducing the effective defect density, as shown in Figure 6.5(d). After a subsequent block erase operation, both cells return to the erased logic state; however, their defect densities remain different. As shown in Figures 6.5(e)–(f), the originally erased cell retains a higher density of defect states than the originally programmed cell.

During the recovery phase, an all-zero data pattern is programmed into both cells to expose this hidden physical disparity. This operation results in the same nominal net charge (five electrons) in each cell. However, the spatial distribution of electrons differs: in the previously erased cell, a larger fraction of electrons becomes trapped in oxide defect states (Figure 6.5(g)), whereas in the previously programmed cell, most electrons are stored in the FG layer, with only one electron trapped in the oxide layer (Figure 6.5(h)). At this stage, the flash controller cannot digitally distinguish between the two cells, even though their physical properties differ.

Electrons trapped in the oxide are weakly confined and therefore susceptible to de-trapping. As a result, these electrons rapidly migrate back, leading to charge-loss mechanisms. A recovery bake at $T_{Rec.}$ further accelerates this charge-loss process. Consequently, the previously erased cell loses more electrons, reducing its net charge to three (Figure 6.5(i)), whereas the previously programmed

cell loses only one electron and retains a net charge of four (Figure 6.5(j)). Therefore, although a standard block erase restores the logical state, residual differences in defect density persist. These analog remnants enable reconstruction of the originally stored data during the recovery phase.

The last recovery stage applies a read-offset technique, in which the V_{ref} is gradually tuned to progressively separate the V_{th} distributions of cells that originally stored '0' and '1.' An important observation is that cells initially programmed to '1' lose charge more rapidly than those programmed to '0.' As a result, the orange distribution (original '1') shifts past V_{ref} earlier than the green distribution (original '0'). When the adjusted V_{ref} is positioned near the intersection of these two distributions, as shown in Figure 6.5(k), the stored image can be largely reconstructed. Because a cell is interpreted as '1' when $V_{th} < V_{ref}$ and '0' otherwise, placing V_{ref} at this optimal point allows most original '1's to be correctly read as '1' and most original '0's as '0,' producing a recovered image that closely resembles the original.

However, perfect recovery (100% accuracy) may not be achievable with this approach due to the intrinsic overlap between the V_{th} distributions of original zeros and ones. Increased overlap directly degrades recovery performance. In addition, accurately tuning V_{ref} to the exact boundary between the two distributions is technically challenging. The following section presents a data recovery algorithm designed to mitigate these issues and analyzes the key factors that affect recovery efficiency.

6.2.5 Factors Influencing Data Leakage & Imprinting in FG vs. CT NAND

This section analyzes the key factors affecting data leakage in 3D NAND flash memory, as well as compares data imprinting between FG and CT memory types. The optimal data leakage percentage ($\eta_{\text{leak}}^{\text{opt.}}$) is obtained by performing a bitwise comparison between the originally stored data (the Tesla image) and the data recovered using Algorithm 3. A value of $\eta_{\text{leak}} = 100\%$ corresponds to perfect reconstruction, while a fully erased (all '1's) or fully programmed (all '0's) yields η_{leak} $\sim$50%, assuming the original image contains approximately equal proportions of '0's and '1's. The data leakage percentage is influenced by several factors, including (i) the recovery stress (time

and $T_{Rec.}$), (ii) the retention stress (time and $T_{Ret.}$), (iii) the layer location dependencies, (iv) the device wear-out level characterized by the N_{PE}, and (v) data pattern types. In this section, each of these factors is examined systematically.

Recovery Duration and Temperature

Figure 6.6 shows $\eta_{leak}^{opt.}$ as a function of recovery bake duration and temperature under end-of-life cell conditions (N_{PE} = 3k). In this study, three chips of the same part number were subjected to the procedure shown in Figure 6.2(b), with recovery baking performed at various $T_{Rec.}$. The green, blue, and red curves in Figure 6.6 correspond to $T_{Rec.}$ values of 85°C, 120°C, and 150°C, respectively. Each data point represents a recovered image obtained at a different recovery duration. Note that during the accelerated aging phase, the retention condition is fixed at $T_{Ret.}$ = 120°C for 60 minutes for all three chips. The results in Figure 6.6 show that at a low recovery temperature ($T_{Rec.} = 85$°C), it takes a long bake time of about 300 minutes (approximately five hours) to reach $\sim 67\%$ of $\eta_{leak}^{opt.}$. In contrast, at higher recovery temperatures ($T_{Rec.}$ = 120°C and $T_{Rec.}$ = 150°C), a higher leakage level of $\approx 71\%$ is reached within only 60 minutes and 10 minutes, respectively. Based on these results, the following evaluations use either a high-temperature, short-duration recovery condition (150°C for 10 minutes or 120°C for 60 minutes) or a low-temperature,

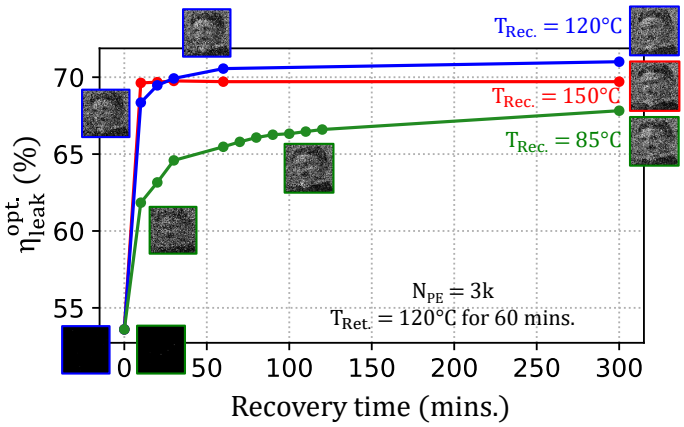


Figure 6.6: Percentage of $\eta_{leak}^{opt.}$ as a function of recovery bake duration and $T_{Rec.}$.

long-duration condition (85°C for more than five hours), since both approaches result in similar levels of data leakage.

This behavior can be explained by the underlying device physics. A higher recovery temperature supplies sufficient thermal energy to promote charge loss mechanisms. This effect is more pronounced in previously erased cells, which contain a larger population of trapped charges in the oxide layer (Figure 6.5(g)) than previously programmed cells (Figure 6.5(h)). Consequently, the separation between the V_{th} distributions of original '1' and '0' cells becomes more distinct, resulting in increased data leakage. Higher temperatures, therefore, enable the V_{th} distributions to separate more rapidly. However, a faster initial separation does not necessarily translate into a larger final separation at extended bake times. At lower recovery temperatures, the same physical mechanisms occur but require substantially longer bake duration to achieve a comparable effect. Therefore, the recovery process is not restricted to a single temperature–time condition; instead, different combinations of higher temperature with shorter duration or lower temperature with longer duration can produce similar levels of leakage amplification.

Retention Duration and Temperature

Both retention duration and $T_{Ret.}$ play critical roles in determining the extent of data leakage. Figure 6.7 presents $\eta_{leak}^{opt.}$ for end-of-life blocks (N_{PE} = 3k) subjected to varying retention durations and temperatures. In this study, three chips of the same part number were processed following the

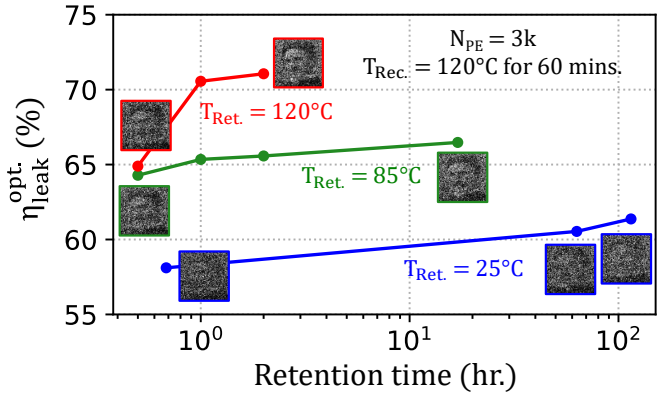


Figure 6.7: Percentage of $\eta_{leak}^{opt.}$ as a function of retention bake duration and $T_{Rec.}$.

procedure shown in Figure 6.2, with the retention durations ranging from 30 minutes to 4 days at temperatures between $T_{Ret.}$ = 25°C and $T_{Ret.}$ = 120°C. During the recovery phase, all samples were subjected to the same recovery condition ($T_{Rec.}$ = 120°C for 60 minutes). In Figure 6.7, the blue, green, and red curves correspond to $T_{Ret.}$ values of 25°C, 85°C, and 120°C, respectively. The y-axis shows $\eta_{leak}^{opt.}$, which reflects how much of the original data can be recovered, while the x-axis represents retention time. The small images (insets) illustrate the quality of recovered data. Overall, the results show that data leakage increases with both (i) longer retention duration and (ii) higher retention temperature.

For all temperatures ($T_{Ret.}$), $\eta_{leak}^{opt.}$ increases as retention time increases. This increase is slow at low temperature (25°C), moderate at 85°C, and much faster at 120°C. For example, portions of the original Tesla image become visible after only 30 minutes at 85°C; after approximately 20 hours at the same temperature, the recovered image becomes more visible. These results reveal a critical weakness in standard sanitization methods. Notably, even at room temperature (25°C), faint traces of the image emerge about one hour, underscoring real-world security implications. The increase in $\eta_{leak}^{opt.}$ with longer retention time can be attributed to the recombination of broken bonds in the oxide. As retention time increases, progressive bond recombination reduces the defect-state density in cells originally storing '0' (previously programmed cells), as shown in Figure 6.5(d). In contrast, cells originally storing '1' (previously erased cells) maintain a relatively higher defect-state density due to no bond reformation, as shown in Figure 6.5(c). This asymmetry enhances the separation between the V_{th} distributions during the subsequent recovery bake, thereby increasing leakage.

At any given retention time, higher temperatures result in greater data leakage. At elevated temperatures, the recovered image becomes more visible, demonstrating that higher operating temperatures during the retention phase substantially increase the risk of data leakage. This occurs because elevated $T_{Ret.}$ accelerates the broken bond recombination kinetics, promoting faster reduction in defect density in previously programmed cells (Figure 6.5(d)). Consequently, this leads to greater V_{th} distribution separation during the data recovery phase. These observations confirm that

broken-bond recombination in the oxide layer governs leakage behavior, with higher temperatures and longer retention times primarily accelerating this underlying physical mechanism.

Device Wear-Out Levels

Figure 6.8 shows the leakage metric as a function of retention time, ranging from 30 to 120 minutes at $T_{Ret.}$ = 120°C, for blocks with different P/E cycling conditions. The green, blue, purple, pink, and red curves correspond to N_{PE} = 1, 500, 1k, 2k, and 3k, respectively. Note that the recovery bake is fixed at $T_{Rec.}$ = 120°C for 60 minutes. The results indicate that data leakage increases as the number of P/E cycles increases. At 30 minutes, blocks with higher N_{PE} already exhibit noticeably higher leakage compared to fresh blocks (N_{PE} = 1). This trend is also reflected in the inset images, where the recovered Tesla image becomes clearer as N_{PE} increases. The image becomes visible around $N_{PE} \approx$ 1k and becomes significantly clearer at higher cycling levels (2k-3k). Similar behavior is observed at 60 and 120 minutes of retention time. These findings demonstrate that more heavily used devices are more vulnerable to data leakage.

Data leakage also increases with retention time for all P/E cycling conditions. This increase is more pronounced in heavily cycled blocks (N_{PE} = 2k-3k), where the change between 30 and 60 minutes is larger, while lightly cycled blocks show only a modest increase. Beyond 60 minutes, the leakage begins to saturate. This behavior suggests that the density of defects (e.g., broken bonds) in the oxide layer increases with cycling, and that these defects gradually recover over

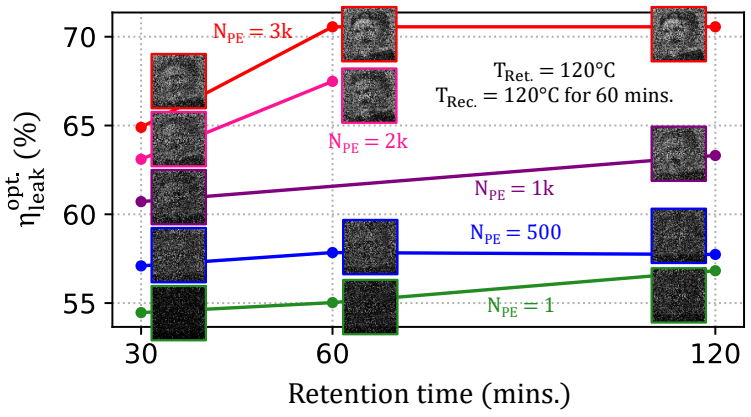


Figure 6.8: Percentage of $\eta^{opt.}_{leak}$ as a function of device wear-out levels (N_{PE}).

time. Overall, we conclude that both device wear (P/E cycling) and retention time contribute to data leakage. However, P/E cycling has a stronger impact, as it determines the overall leakage level, while retention time further amplifies it.

Layer Location Dependencies

Figure 6.9 illustrates $\eta_{\text{leak}}^{\text{opt.}}$ as a function of layer number within a flash block with N_{PE} = 20k, measured after $T_{Ret.}$ = 120°C for 30 minutes and $T_{Rec.}$ = 120°C for 60 minutes. Overall, the leakage exhibits clear layer-to-layer variation across the vertical stack. While most layers cluster within a relatively narrow range (approximately 65–70%), certain layers show pronounced deviations. In particular, the bottom layer (Layer 0) exhibits the lowest leakage value ($\sim$ 62%), followed by a sharp increase in the next few layers, reaching a peak near Layer 5 ($\sim$ 70%). After this early fluctuation, the leakage stabilizes across the middle layers, with moderate oscillations but no systematic upward or downward trend. These results suggest that the leakage behavior is not uniform along the vertical direction and is strongly dependent on the physical layer position. This layer dependency highlights the need to consider vertical position when evaluating reliability and recovery characteristics in 3D NAND structures.

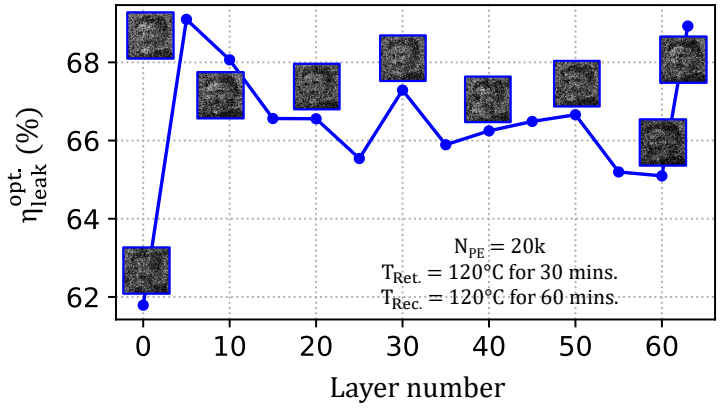


Figure 6.9: Percentage of $\eta_{\text{leak}}^{\text{opt.}}$ as a function of layer number.

Data Type Dependencies

Figure 6.10 illustrates recovered data corresponding to different originally stored data patterns: (a) grayscale Tesla image, (b) grayscale Mona Lisa image, (c) pseudo-random binary data, and (d) a text file containing ASCII characters. In this evaluation, a 3D FG NAND flash device is cycled to N_{PE} = 3k, and data are recovered after 60 minutes with $T_{Ret.}$ = 120°C. It is important to note that $\eta_{leak}^{opt.}$ for these four data types are very similar, ranging from 63.5% to 65%.

For a grayscale image, the recovered data exhibit noticeable noise; however, the remanence of the original Tesla image remains clearly visible (Figure 6.10(a)), indicating that the proposed recovery algorithm can successfully separate a significant portion of the V_{th} distributions corresponding to originally programmed '0' and erased '1' cells. The Mona Lisa image (Figure 6.10(b)) is also included to demonstrate that the proposed technique is not limited to a specific image and generalizes across different visual content. In addition to grayscale images, synthetic binary patterns generated from pseudo-random data are also recovered, as shown in Figure 6.10(c). This further confirms that residual physical differences between previously stored states can be exploited to reconstruct meaningful information. Figure 6.10(d) shows the case where the originally stored data type is a text file containing ASCII characters. The recovered data show higher corruption, although some characters remain readable. Further improvements are possible through post-processing techniques, such as frequency-based character analysis, to enhance text recovery. Overall, while most results are demonstrated on image data, the proposed recovery technique is broadly applicable to multiple data types.

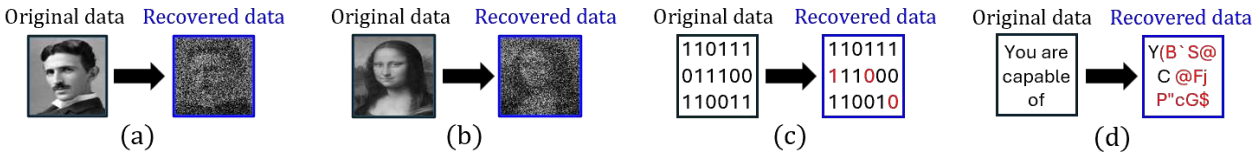


Figure 6.10: Recovered data corresponding to different originally stored data patterns: (a) grayscale Tesla image, (b) grayscale Mona Lisa image, (c) pseudo-random binary data pattern, and (d) text file containing ASCII characters.

Data Imprinting in FG versus CT 3D NAND

To evaluate data imprinting in end-of-life (N_{PE} = 3k) FG and CT NAND chips, we followed the procedure in Figure 6.2, with $T_{Ret.}$ = 120°C for 30 minutes and $T_{Rec.}$ = 120°C for 60 minutes. Figure 6.11 compares results by plotting $\eta_{leak}^{opt.}$ of the recovered Tesla image versus $N_{PE}^{san.}$. The blue curve represents CT NAND, while the red curve represents FG NAND. The results reveal distinct differences between technologies. FG NAND exhibits strong imprinting: when only block erase is applied ($N_{PE}^{san.}$ = 0), $\eta_{leak}^{opt.} \approx 65\%$ with clear Tesla image remnants. After one cycle of erase and pseudo-random data programming ($N_{PE}^{san.}$ = 1), the leakage drops to approximately 61%, and it slightly decreases to around 60% after the second cycle, with faint image traces remaining. After the third cycle, leakage falls to around 56%, and the image is no longer clearly visible. In contrast, CT NAND enables only one-time recovery with no repeated imprinting. While both technologies exhibit data leakage, FG NAND is more vulnerable due to imprinting, which enables data persistence across multiple sanitization cycles. CT NAND, lacking the imprinting effect, requires only minimal additional cycles to ensure successful data sanitization.

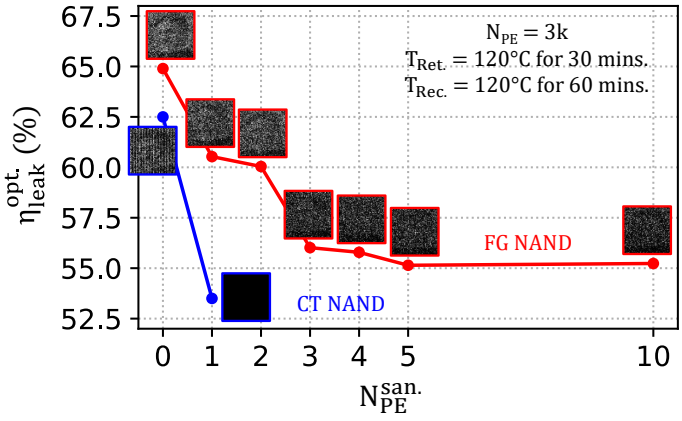


Figure 6.11: Percentage of $\eta_{leak}^{opt.}$ as a function of number of P/E cycles during sanitization ($N_{PE}^{san.}$).

6.2.6 Multi-Chip Evaluations

We implemented the recovery algorithm on commercial 3D NAND flash devices with the Tesla image stored in SLC pages. Table 6.1 summarizes results from 64-layer and 96-layer FG 3D NAND flash memories under similar retention and recovery conditions ($T_{Ret.} = T_{Rec.} = 120°C$ for 60 minutes). All devices showed consistent data leakage; for example, the 64-layer FG device exhibited $\sim 70.6\%$ of the optimal leakage at end-of-life condition ($N_{PE} = 3k$), while the 96-layer FG device achieved $\sim 67.3\%$ at the same wear level. These results show that the vulnerability does not depend on specific devices and persists across technology generations.

Table 6.1: Summary of $\eta_{\text{leak}}^{\text{opt.}}$ across various 3D FG NAND samples. The retention condition is fixed at 120°C for 60 minutes.

Sample	Part number	Condition	N_{PE}	$\eta_{\text{leak}}^{\text{opt.}}$
64-layer	MT29F512G08EEHAF	120°C for 60 mins	3k	70.6%
96-layer	MT29F2T08GELBE	120°C for 60 mins	3k	67.3%

6.3 Mitigation Strategies

In this section, we propose possible strategies to mitigate the effects of data leakage in 3D FG NAND flash memory.

6.3.1 Aging-Aware All-Zero Refresh

A practical system-level approach to reducing data leakage is the aging-aware all-zero refresh. Leakage primarily occurs when data remains in the same worn-out physical block for long periods, allowing the broken bond repair processes to accumulate. To mitigate this issue, the storage system can periodically migrate valid data to new physical blocks, preferably those with lower P/E cycles, and overwrite the vacated blocks with an all-zero pattern. This controlled zeroing process helps neutralize defect states under both high- and low-trap-density conditions, effectively eliminating residual traces of the original data. As a result, no physical block is allowed to retain long-term

charge residues that could later reveal sensitive information. The refresh policy must carefully balance data security with flash durability, since excessive overwriting or frequent zeroing increases wear. Endurance-aware strategies, such as only refreshing long-idle or aging blocks, or integrating zeroing into existing garbage collection or wear-leveling routines, can minimize unnecessary P/E cycles while maintaining a strong defense against data remanence and leakage.

6.3.2 Multiple P/E Cycles Sanitization Practices

Another approach to mitigate data leakage is performing multiple P/E cycles prior to the final block erase. Figure 6.12 presents $\eta_{\text{leak}}^{\text{opt.}}$ when additional sanitization P/E cycles are applied using either pseudo-random data patterns (red curve) or inverted stored data (e.g., the inverted Tesla image) (blue curve). The detailed procedure follows Figure 6.2(b), with retention at $T_{\text{Ret.}}$ = 120°C for 30 minutes and recovery at $T_{\text{Rec.}}$ = 120°C for 60 minutes. As shown in the figure, applying pseudo-random data during sanitization gradually reduces the leakage. After three sanitization cycles with pseudo-random data pattern, $\eta_{\text{leak}}^{\text{opt.}}$ decreases to approximately 56%, and no visible traces of the original Tesla image remain. In contrast, sanitization using the inverted Tesla image is significantly more effective. After only one or two additional sanitization cycles, the remanence of the original Tesla image becomes minimal to unobservable, indicating a much faster suppression of data leakage.

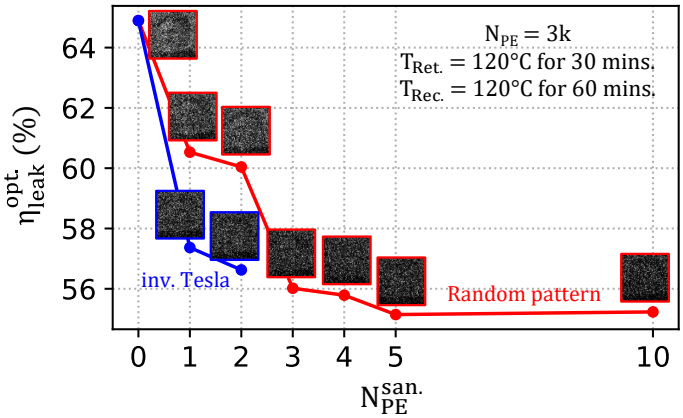


Figure 6.12: Percentage of $\eta_{\text{leak}}^{\text{opt.}}$ as a function of number of P/E cycles during sanitization ($N_{PE}^{\text{san.}}$).

6.3.3 Thermal-Assisted Sanitization Practices

Another viable countermeasure involves implementing advanced sanitization procedures before a device is discarded or repurposed. Standard block erase operations alone are insufficient, as oxide defect density variations can persist even after erasure. To enhance the effectiveness of data sanitization, we introduce a thermal-assisted method that first programs the chip with an all-zero data pattern, then subjects it to a controlled heating process before performing the final block erase. This thermal treatment, applied while the device holds an all-zero pattern, uses heat energy to accelerate the broken bond repair process in all cell conditions. As a result, it neutralizes defect states under both high- and low-trap-density conditions that could otherwise permit adversaries to reconstruct erased data.

Figure 6.13 presents the optimized leakage metric as a function of thermal-assisted sanitization count, with N_{PE} = 3k, T_{ret} = 120°C for 30 minutes, and T_{rec} = 120°C for 60 minutes. Note that a thermal-assisted sanitization count of zero represents the case where the technique is not applied (recovery after standard block erase only). After holding the chip with an all-zero pattern at 120°C for 60 minutes prior to the final block erase, data leakage during recovery decreases from 65% to 60%. As the number of thermal-assisted cycles increases, the leakage decreases, with no visible trace observed. As a result, for higher security requirements, multiple passes of the thermal-

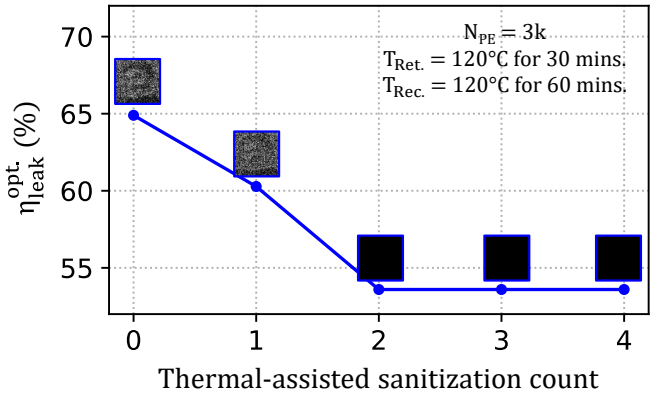


Figure 6.13: Percentage of $\eta_{leak}^{opt.}$ as a function of thermal-assisted sanitization count.

assisted sanitization cycles may be required before the final erase, further increasing the cost and difficulty of forensic data reconstruction.

In contrast to destructive sanitization techniques such as shredding, pulverizing, or degaussing, these proposed methods offer two main benefits: (1) they keep the memory device operational, which is important when the media may be reused or resold, and (2) they scale more easily, since they avoid the logistical burden and expense of physically disposing of large volumes of chips. Although destructive sanitization can still be required in the most security-critical settings, thermal-assisted erasure delivers a non-destructive yet effective way to reduce imprinting risks in many real-world deployments.

6.4 Conclusion

This paper presents the first experimental investigation of post-sanitization data leakage in commercial FG 3D NAND flash memory. Our results demonstrate that end-of-life FG 3D NAND devices remain vulnerable even after a standard block erase operation. The remanence of previously stored images originates from differences in oxide trap-state density that persist even after a block erase operation. Using a controlled read-offset technique, we show that up to 70% of the original data can be recovered from sanitized blocks at end-of-life conditions when data is stored in SLC pages. The severity of this leakage is strongly dependent on recovery stress, retention stress, layer location within the stack, P/E cycling wear, and the types of stored data. Through device-physics analysis, we identify the root cause as asymmetric defect-state evolution: programmed cells experience broken-bond recombination during retention, whereas erased cells inhibit broken-bond repair. These persistent physical disparities are not eliminated by a conventional block erase. By exploiting these intrinsic differences between programmed and erased states, we successfully capture and reconstruct the residual information. Therefore, our findings reveal that block erase alone is insufficient for secure data removal in FG 3D NAND flash memory; to mitigate this vulnerability, we propose system- and device-level strategies, including aging-aware all-zero refresh and thermally assisted sanitization.

Chapter 7

Block Erase Vulnerabilities in MLC NAND

Block erase is the standard logical sanitization mechanism in NAND flash memory. By resetting all cells in a block to the erased state (logic-1, L_0), it is intended to render previously stored data permanently inaccessible. Despite its widespread adoption and endorsement by standards such as NIST SP 800-88 Rev. 1, the physical security guarantees of block erase in 3D FG NAND flash under end-of-life conditions have not been thoroughly characterized, particularly for MLC configurations, which store two bits per cell across four V_{th} levels (L_0–L_3). This chapter presents an experimental investigation into post-erase data remanence in commercial 64-layer FG 3D NAND flash memory operating in MLC mode. Specifically, we focus on the recovery of data stored in the LSB page, which corresponds to the lower bit of each two-bit cell. We demonstrate that residual charge asymmetries introduced by prior programming survive the block erase operation and can be exploited to reconstruct previously erased content. A targeted read-offset recovery algorithm is applied. The goal of this chapter is to characterize the extent of the vulnerability, explain the underlying physical mechanisms that enable recovery in the MLC context, and evaluate per-state recoverability from the four-level V_{th} distribution.

7.1 Key Ideas

The main finding of this chapter is that post-erase data recovery is achievable in MLC NAND flash, extending the vulnerability established for SLC storage in the preceding chapters to higher-density configurations. Approximately 65% of the LSB page data can be recovered from an end-of-life, post-erased block of 64-layer FG 3D NAND. This recovery rate is significant because MLC storage is far more prevalent in commercial SSDs and modern NAND flash products than SLC. Data recovery is enabled by a differential charge-loss mechanism unique to the MLC context. In MLC storage, four distinct V_{th} levels encode two bits of data. Following a block erase, all cells are nominally driven to L_0. However, cells that previously occupied different programmed levels (L_1,

L_2, or L_3) retain subtly different residual charge signatures. When the erased page is subsequently reprogrammed with a known all-zero pattern and subjected to an elevated-temperature bake, these differences are amplified through asymmetric charge loss, making previously stored data partially recoverable via read-offset operations.

7.1.1 Recovery Algorithm for MLC Storage

The recovery algorithm for LSB pages of MLC storage follows the same general structure as the SLC approach introduced in Chapter 6 (Algorithm 3, Figure 5.2), with one key distinction in how the read-offset command is configured. In SLC mode, the SET FEATURES command (0xEF) references a feature address that targets a single V_{th} boundary. In the LSB page of MLC mode, the relevant feature address differs because the LSB page is decoded with reference to a different pair of voltage boundaries within the four-level distribution. Specifically, the LSB page encodes the MSB of the two-bit symbol, meaning its read reference lies between L_1 and L_2 in the standard MLC V_{th} mapping. As a result, according to the datasheets, the feature address for the LSB page in MLC mode is 0xA1, while SLC mode uses 0xA4.

The algorithm proceeds as follows. A target LSB page within an erased block is first identified. That page is then programmed with an all-zero data pattern to place all cells into a defined programmed state (L_2 in the MLC scheme). A recovery bake is applied at $T_{Rec.}$ to induce differential charge loss between cells according to their prior history. The V_{offset} is then swept from $V_{offset,min}$ to $V_{offset,max}$ in increments of ΔV by issuing the appropriate SET FEATURES command with the MLC-specific feature address and P1 subfeature value, followed by a standard NAND page read (0x00–0x30). At each offset step, the readout is recorded, and the valley position in the V_{th} distribution is computed. The process terminates once the optimal valley is identified; if not found, the offset is incremented, and the read is repeated.

7.2 Experimental Evaluation

This section describes the experimental methodology adapted for MLC storage, presents a visual demonstration of LSB page recovery, and quantifies the effect of read-offset voltage on recovery efficiency. All experiments were conducted on commercial 64-layer FG 3D NAND flash devices at 25°C unless otherwise stated.

7.2.1 Experimental Flow

The experimental procedure mirrors the four-stage pipeline used for SLC devices in Chapter 5 (Figure 5.3), comprising data programming, retention stress, block erase sanitization, and recovery, while accounting for MLC-specific write and read operations. Blocks are first subject to repeated P/E cycling to bring them to the end-of-life wear state, while other blocks are left in a factory-fresh condition to capture the effect of nonuniform wear across the device. During the data programming stage, two grayscale test images, a Tesla image and an Einstein image, are written to the shared LSB and MSB pages of a selected flash block. Each image contains approximately 50% one-bits and 50% zero-bits, which ensures that the four MLC V_{th} levels (L_0–L_3) are populated roughly equally at $\sim 25\%$ each, providing a balanced and representative distribution for recovery analysis. The remaining pages in the block are filled with pseudo-random data, and the programmed page is verified by a read operation.

Accelerated aging is then performed by baking the programmed chip at an elevated $T_{\mathrm{Ret.}}$ to emulate prolonged real-world data storage. Following retention stress, a standard block erase is executed to simulate device sanitization before disposal or reassignment. To initiate recovery, the erased page is reprogrammed with all zeros and baked at a $T_{\mathrm{Rec.}}$ to amplify residual charge differences. Finally, read-offset operations are applied to reconstruct the original image content.

7.2.2 Demonstration of LSB Page Recovery in MLC Storage

Figures 7.1(a)–(d) illustrate the four key stages of NAND flash memory operation: end-of-life data storage, block erase sanitization, all-zero programming, and recovery bake. These stages are described below.

End-of-life data storage (Figure 7.1(a)): The Tesla and Einstein images are programmed into the shared LSB and MSB pages of the target block at end-of-life wear conditions. Because each image has a balanced bit distribution (∼50% ones and zeros), the corresponding V_{th} distribution is spread roughly uniformly across all four MLC states, approximately 25% of cells in each of L_0, L_1, L_2, and L_3. The device is then subjected to accelerated retention aging at $T_{Ret.}$. to simulate realistic storage durations during typical device use.

Block erase sanitization (Figure 7.1(b)): A standard block erase is applied in accordance with NIST SP 800-88 Rev. 1 guidelines. This operation drives all memory cells to the L_0 erased state, deleting the stored data. The resulting page readout is a uniform all-ones (white) image, and the V_{th} distribution collapses into a single L_0 state. In theory, at this stage, no data content is recoverable and the flash-based devices can be discarded or resold.

All-zero programming (Figure 7.1(c)): The recovery process begins by programming the erased LSB and MSB shared pages with a uniform all-zero data pattern. In MLC storage, this causes all cells to transition to the L_2 state, which encodes a logical zero on both the LSB and MSB pages. The programmed page, therefore, appears as a uniform all-zero (black) image. Importantly, although all cells reside in the same L_2 state, their retention characteristics still reflect their original states (L_0–L_3 prior to block erase sanitization), exhibiting subtly different residual charge signatures due to distinct charge-loss histories.

Recovery bake and read-offset reconstruction (Figure 7.1(d)): A recovery bake at $T_{Rec.}$ is applied to amplify residual differences between cells with differing prior histories. After baking, a default read at the standard V_{ref} yields little to no visible data. To enhance data extraction, the V_{offset} is swept across its full range from $V_{offset,\,min}$ to $V_{offset,\,max}$. At the optimized offset, the Tesla image is successfully reconstructed, as shown in Figure 7.1(d).

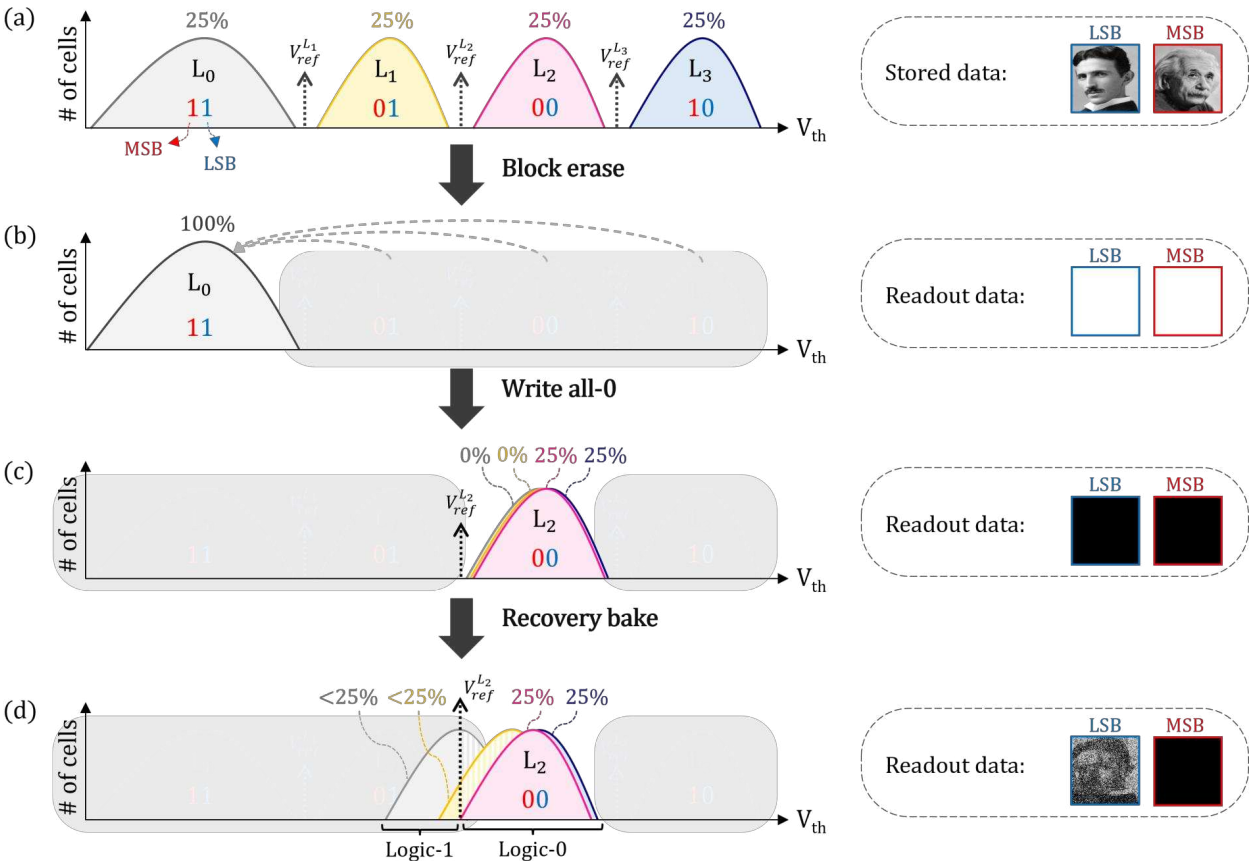


Figure 7.1: Demonstration of LSB page data recovery in MLC storage during (a) end-of-life user data storage, (b) block erase sanitization, (c) all-zero reprogramming, and (d) recovery bake with read-offset reconstruction.

7.2.3 Recovery Optimization and V_{th} Distribution Analysis

This section quantifies how the V_{offset} affects the η_{leak} and presents the observed recovery behavior through the measured V_{th} distribution shift.

Offset voltage sweep and η_{leak} characterization (Figure 7.2(a)): The experiment is conducted on an end-of-life block at N_{PE} = 5k cycles. Prior to sanitization, both test images are stored under accelerated retention at $T_{Ret.}$ = 120°C for 30 minutes. Under the Arrhenius model with an activation energy of 1 eV, this corresponds to approximately 17 months of room-temperature storage. The full recovery procedure (Algorithm 3) using 0xA1 as a feature address is then executed, including a recovery bake at $T_{Rec.}$ = 120°C for 60 minutes.

Three distinct behavioral regimes are observed as V_{offset} is swept from the minimum to the maximum values. At low offset values, the shifted read reference falls entirely below the V_{th} distribution, causing all cells to be interpreted as logic-0. This results in a uniformly dark image with $\eta_{leak} \approx 50\%$, which simply reflects the proportion of zero-bits in the original Tesla image rather than any meaningful reconstruction. As V_{offset} increases toward an intermediate optimum near 300 mV, the read reference crosses into the valley region between the partially separated distributions formed by cells with different prior histories. In this regime, recovery accuracy is maximized, and η_{leak} reaches its peak of approximately 65%, with the Tesla image clearly reconstructed. Beyond this optimum point, further increases in V_{offset} push the V_{ref} above both distributions, causing all cells to read as logic-1, and η_{leak} declines as the recovered image transitions toward an all-white output.

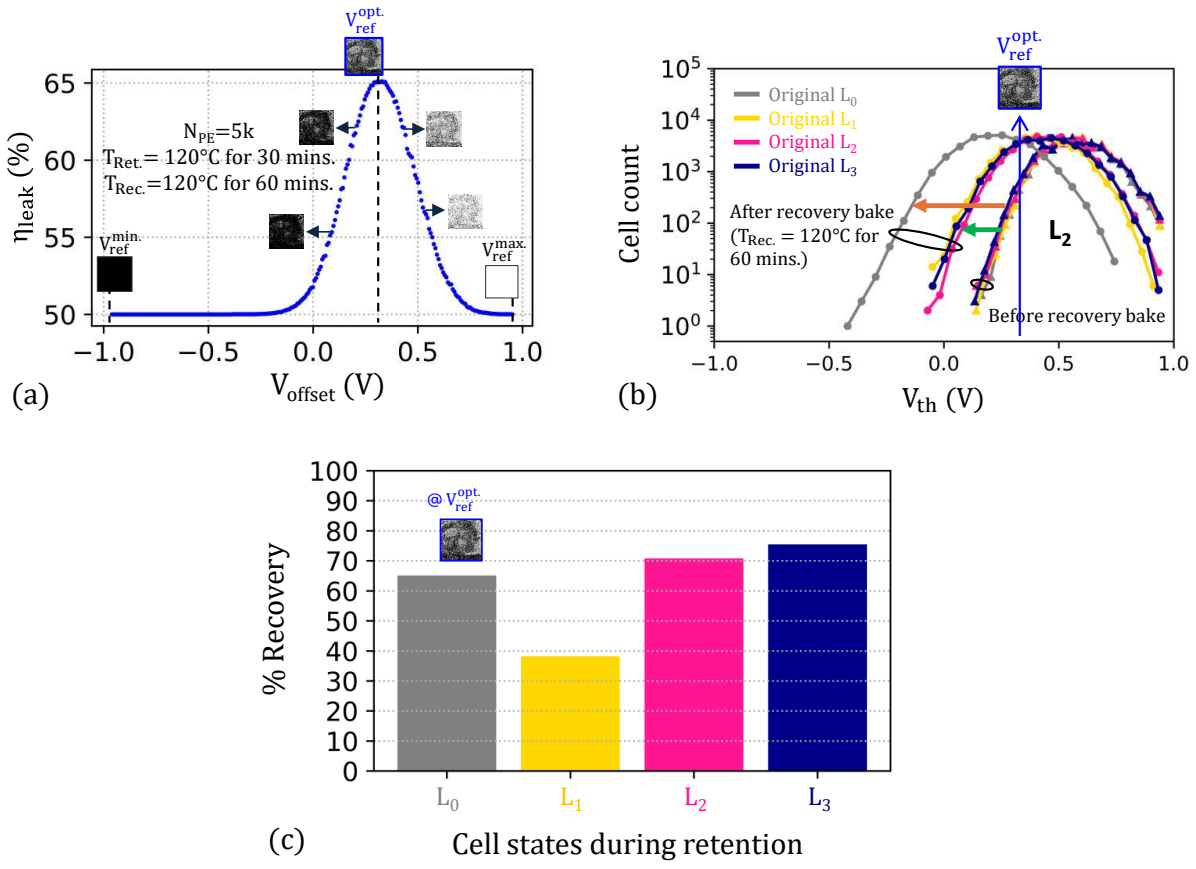


Figure 7.2: (a) Percentage of η_{leak} as a function of V_{offset}. (b) Measured V_{th} distribution before and after the recovery bake. (c) Per-state percentage of correctly recovered cells across L_0–L_3.

The 65% recovery ceiling reflects an inherent physical constraint: after the recovery bake, the V_{th} distributions of cells from different prior states do not fully separate. Their tails overlap at the valley position, meaning some cells from one prior state are misclassified as belonging to the other. Achieving 100% recovery would require perfect separation of distributions, which the high-temperature baking process does not achieve. Nevertheless, 65% recovery is sufficient to reconstruct visual images and identify structured data patterns, representing a meaningful security threat.

Threshold voltage distribution before and after recovery bake (Figure 7.2(b)): Figure 7.2(b) shows the measured V_{th} distribution before and after the 60-minute recovery bake at $T_{Rec.}$ = 120°C. Before baking, all cells are in the L_2 state due to all-zero programming, resulting in a single, relatively tight V_{th} distribution peak. After the bake, this peak broadens and begins to split into two partially overlapping distribution groups. The separation arises because cells that were originally in the L_0 state before the block erase, preventing the broken bond repair process, lose electrons more rapidly during the bake than cells that had been in higher programmed states (L_3, L_2, and L_1). This asymmetric charge loss is the physical basis for data remanence in the MLC context. Cells that previously held logic-1 data on the LSB page (mapped to L_0 or L_1 in the MLC encoding) shift to lower V_{th} values faster than cells that previously held logic-0 (mapped to L_2 or L_3), creating a measurable distribution difference that the read-offset operation can exploit.

Per-state recovery rates (Figure 7.2(c)): Figure 7.2(c) breaks down the percentage of cells correctly identified for each of the four MLC V_{th} levels. The per-state recovery rates are approximately 65% for L_0 cells, 40% for L_1 cells, 70% for L_2 cells, and 75% for L_3 cells. These results are consistent with the charge-loss asymmetry described above. Specifically, L_0 cells, which contain the highest density of broken bonds, exhibit a recovery rate of $\sim$65%. L_1 cells, with a moderate broken-bond density, show the lowest recovery at $\sim$40%. In contrast, L_2 ($\sim$70%) and L_3 ($\sim$75%) cells achieve higher recovery rates because they were previously programmed to higher charge states, resulting in fewer broken bonds and thus smaller shifts in their post-bake V_{th} distributions

compared to previously erased cells. In aggregate, these per-state recovery rates yield the overall $\sim$65% LSB page recovery observed in the V_{offset} sweep.

7.3 Conclusion

This chapter demonstrates that the post-erase data remanence vulnerability in 3D FG NAND flash memory extends to MLC storage configurations. Using commercially available 64-layer devices, we show that applying a recovery bake followed by read-offset operations enables reconstruction of up to 65% of LSB page content from blocks sanitized via standard block erase, even under end-of-life wear conditions. This result extends the threat beyond the SLC case examined in prior chapters to the higher-density, more commercially prevalent MLC architecture. The underlying mechanism, differential oxide defect repair across distinct prior programming states, manifests in the MLC context and causes an asymmetric shift in the post-bake V_{th} distribution. These findings highlight a fundamental limitation of block erase as a standalone sanitization method in 3D FG NAND: it is insufficient to prevent data recovery under adversarial conditions in both SLC and MLC configurations. Hardware manufacturers, system integrators, and data security practitioners relying solely on block erase for sanitization should therefore consider supplementary measures to mitigate residual data exposure at the end of device life.

Chapter 8

Descrambling scrambler keys in commercial NAND

Data scramblers in NAND flash memory randomize stored bit patterns to improve reliability by reducing disturbance effects, minimizing pattern-dependent retention loss, enhancing wear leveling, and improving ECC performance. Typically implemented using LFSR-based pseudo-random sequences, scrambling involves XORing data with a page-dependent key that can be deterministically regenerated during read operations. However, these mechanisms are proprietary, making reverse engineering of scrambling keys an important challenge. Extracting these keys enables applications such as efficient data sanitization, forensic data recovery, and improved error management, highlighting their significance in NAND flash security and system design.

8.1 Related Work

Only limited work exists in the public domain on reverse-engineering the NAND data scrambling process. Zhang et al. [37] proposed a firmware reverse-engineering method to extract the scrambling seed. Their approach resets the SSD into factory mode and reads the initial firmware pages, specifically block 0, page 0, and bank 0, where the firmware code is stored and typically left unscrambled, allowing the seed to be recovered. However, their technique was demonstrated only on SSDs using the Barefoot controller, which employs a specific scrambling algorithm. Moreover, the method is not applicable when scrambling is performed inside the raw NAND chip.

Wise [40] investigated the internal data scrambler used in commercial 2D MLC NAND flash memory from Samsung and demonstrated that the chip employs XOR-based scrambling with a pseudo-random key typically generated by LFSR tied to the physical page address. By reverse-engineering hardware behavior and analyzing raw data dumps, the author reconstructed the scrambling sequence and extracted the key using statistical and probabilistic analysis. However, the approach is highly device-specific and depends on a particular controller from one type of flash

memory chip, limiting its applicability to newer devices and more advanced scrambling schemes. Therefore, scramblers used across most of today's NAND flash chips remain unexplored.

8.2 Key Ideas

This chapter presents the first experimental demonstration of scrambling key extraction from several unmodified COTS NAND flash chips. The key contributions are summarized as follows:

- **Scrambling key extraction:** We show that the scrambling key can be retrieved from unmodified COTS NAND flash devices by manipulating the page write operations to force programmed memory cells into a known physical state.
- **Key characteristics:** We characterize the extracted scrambling keys and show that they exhibit consistent patterns across multiple NAND flash chips, indicating similar internal algorithms. The key length typically ranges from 16 to 64 bytes, depending on the NAND technology, and each key repeats periodically before transitioning to a new sequence.
- **Method generality:** We demonstrate that the proposed extraction method applies to both 2D and 3D NAND flash chips, including SLC and MLC storage types.

8.2.1 Data Scrambler Algorithm

Modern NAND flash incorporates data scrambling mechanisms to enhance reliability and extend device lifetime. Figure 8.1(a) provides an overview of the NAND flash scrambling process. Before data is written to the memory array, the on-die logic applies an internal scrambling algorithm that randomizes the bit patterns [106–108]. The scrambling engine performs a bitwise XOR operation between the user data (Data_{usr}) and a device-specific scrambling key immediately before writing the data into the NAND array ($\text{Data}_{\text{NAND}}$) [107, 108]. During the read operation, illustrated in Figure 8.1(b), the device retrieves the scrambled data or $\text{Data}_{\text{NAND}}$ exactly as it was recorded in the memory array, where the physical cell Vth values reflect the randomized bit distribution. To reconstruct the original Data_{usr}, the controller performs a bitwise XOR operation that applies the

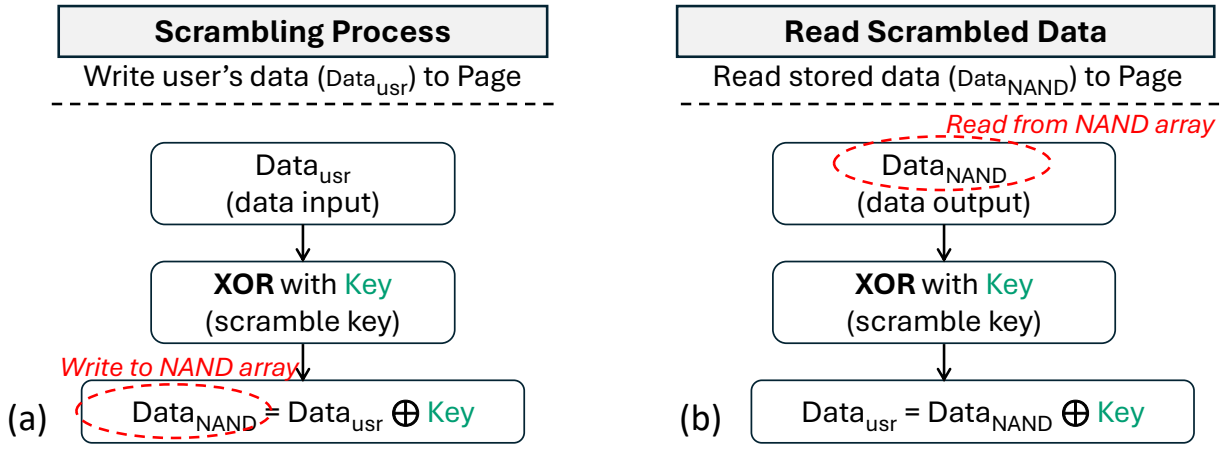


Figure 8.1: Overview of the 3D NAND flash processes for (a) scrambling data during writes, (b) descrambling stored data during reads.

same scrambling key used during the write operation [107, 108]. This pair of operations, scrambling during writing and descrambling during reading, ensures that user data is preserved correctly while maintaining randomized physical storage patterns that improve overall device reliability.

8.2.2 Scrambling Key Extraction

Reverse-engineering the 3D NAND flash data scrambling process is critical for system integrators seeking to enhance SSD data reliability, improve error management strategies, perform forensic analysis, and enable fast sanitization. Our proposed scrambling-key extraction method is illustrated in Figure 8.2. We first describe the method for a single SLC page and then discuss its extension to an MLC device.

When user data is written to the NAND array, it is first XORed with a scrambling key that depends on the physical page address. The scrambled data is then programmed into the NAND array. This step is indicated by ❶ in Figure 8.2. For a fixed physical page address, the scrambling key remains constant; therefore, if the same page is overwritten, the same key is reused.

Our extraction algorithm exploits this property by overwriting the page with the bitwise complement of the user data, as shown in step ❷ of Figure 8.2. Let the original user data be D, and the page-specific scrambling key be K. The first write stores D $\oplus$ K in the array. The second write

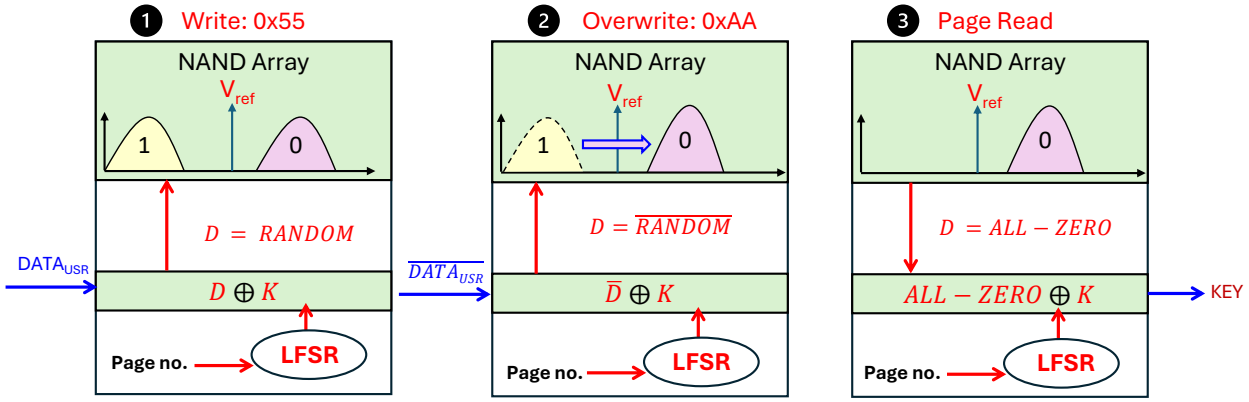


Figure 8.2: Process for scrambling key extraction: (Step ❶) write an all "0x55" data pattern, (Step ❷) overwrite with an all "0xAA" data pattern, the bitwise inverse of "0x55," and (Step ❸) readout the resulting scrambling key.

uses $\overline{D}$ as input, so the scrambled value to be programmed is $\overline{D} \oplus K = \overline{D \oplus K}$, i.e., the bitwise complement of the previously stored scrambled pattern. Because NAND programming is monotonic in V_{th} (cells can only be shifted toward a higher V_{th} state during programming), any cell that is already in the programmed state remains there. In contrast, cells that are still erased are further programmed. As a result, after the overwrite operation, all cells in the page are driven into the highest programmed state.

In terms of logical values, a cell storing '0' (programmed) does not change its state even if a '1' is requested during the overwrite, whereas a cell storing '1' (erased) can be programmed to '0' when a '0' is requested. Consequently, a subsequent page read on this overwritten page returns an all-'0' pattern at the NAND interface. When this all-'0' pattern is XORed with the same scrambling key K, the resulting user-visible data is exactly the key itself. Thus, by driving all NAND cells within a page to a uniform physical state, the data scrambling key can be extracted, as shown in step ❸ in Figure 8.2.

Scope of the proposed technique. The scope and applicability of the proposed method are summarized as follows:

1. Support for multiple storage configurations. Although the proposed method is illustrated for SLC 3D NAND flash devices for clarity, the same principle applies to 3D NAND flash

MLC and TLC devices. The main difference is how data is organized into logical pages and transferred into the internal buffer: an MLC WL is associated with two logical pages (LSB and MSB), while a TLC WL is associated with three (LSB, CSB, and MSB). Note that CSB stands for the central significant bit. Furthermore, the key observed at the interface depends on how the highest V_{th} state encodes the page bits. For example, in our MLC devices, the highest V_{th} state (L_3) corresponds to '10' for the (MSB, LSB) pair. When all cells are forced into L_3, the readout behavior of the two logical pages differs in a predictable way. The LSB page, which stores the '0' bit of the pair, remains in an all-zero state; therefore, its readout directly yields the scrambling key (after XOR). In contrast, the MSB page stores the '1' bit of the pair and thus reads as all ones. As a result, its readout yields the bitwise complement of the key since an all-'1' pattern XORed with the key inverts every bit.

2. Page programming interface constraints. Some 3D NAND flash chips do not allow separate write commands for the LSB and MSB pages and instead require the user to provide all page data before a programming command is issued to write into both pages. Our method still applies to such chips, as the overwrite operation and monotonic V_{th} behavior remain unchanged; only the way the data is loaded into the internal buffer differs.

3. Overwrite and programming constraints. Our method does not modify the scrambling or descrambling processes; instead, it transitions the NAND array to a controlled physical state. While an overwrite operation is one approach to shifting NAND cells to this state, alternative techniques, such as partial programming, can also be employed. This approach is particularly relevant for NAND devices that prohibit overwriting. In such cases, our method remains applicable while complying with device-level programming constraints.

8.2.3 Experimental Procedure

We custom-designed a test board and used it for our evaluation, as illustrated in Figure 3.1(b). To extract the scrambling key for an SLC NAND memory, we follow the experimental procedure

shown in Figure 8.3. The NAND command sequences employed in the scrambling key identification process are shown in Algorithm 4. The process begins by erasing a flash memory block using a block erase operation. Then, an "all-0x55" data pattern is written to the selected page, followed by overwriting the same page with its bitwise complement, the "all-0xAA" pattern. Reading the page after this overwrite returns data that has been XORed with the same scrambling key in both steps, thereby revealing the key. While all-0x55 and all-0xAA are chosen for simplicity to drive all cells to a known physical state, any first-write pattern can be used as long as the overwrite pattern is its bitwise complement.

Algorithm 4 Scrambling key identification

1: **Initialize:**
2: Select an erased block
3: Select a WL within the block
4: **Perform:**
5: Load the page buffers with "all-0x55" data pattern
6: Issue the NAND page program command sequence: 0x80 (PAGE PROGRAM SETUP) → address cycles → data transfer → 0x10 (PROGRAM CONFIRM)
7: Issue the NAND page READ command sequence: 0x00 → address → 0x30 (READ CONFIRM)
8: Repeat Steps 5 to 7 using "all-0xAA" data pattern
9: Record the readout values

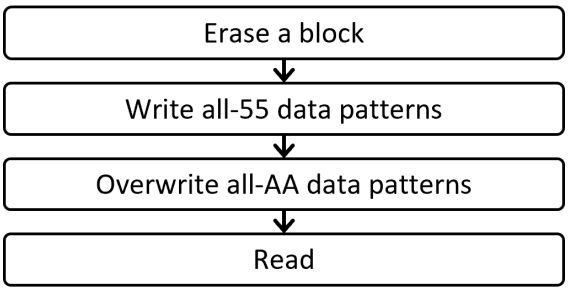


Figure 8.3: Experimental procedure to extract the scrambling key for the SLC NAND memory.

8.3 Experimental Results

8.3.1 Evaluation on 32-layer SLC 3D NAND Flash Chip

To initiate the descrambling procedure, we first select an erased page and program it with the canonical "0x55" data pattern. The page size is 16 KB. A subsequent read confirms the success of the page write operation with zero RBER. It is important to note that although the controller issues the "0x55" pattern for the SLC page-program command, the actual bit sequence written into the NAND array is substantially different due to the internal data scrambler. We refer to this internal randomized bit pattern as RANDOM, as illustrated by ❶ in Figure 8.2. The only guaranteed behavior during this first write is that a subset of cells transitions from the erased state to the programmed state based on RANDOM.

Next, we overwrite the same page using the "0xAA" pattern. Because the scrambling key is fixed on a per-page basis, the newly written pattern corresponds to the bitwise complement of RANDOM. Given that NAND programming is monotonic, cell V_{th} can only shift rightward. This second program step ensures that all cells end in the programmed state, as illustrated by ❷ in Figure 8.2, regardless of the underlying scrambled pattern. Consequently, sensing the NAND array will result in an all '0's pattern at the NAND interface. When this all-'0' pattern is XORed with the scrambling key, the user-visible data equals the key itself, as illustrated by ❸ in Figure 8.2. As expected, the final readout after the overwrite does **not** return the user level "0xAA" pattern. Instead, it yields an apparently random bit sequence, which, according to our proposed algorithm, corresponds directly to the scrambling key.

The readout values following the key extraction method from a single SLC page of 16 kB size are shown in Figure 8.4. For clarity, the readout bytes are arranged into 1,024 rows to highlight the distinctive structure, which indicates that the readout values are not a random bit-stream but produced by the device's internal randomization algorithm. At first glance, the scrambling key for the SLC page appears to be 16 bytes long (corresponding to one row). However, closer inspection reveals that this 16-byte sequence repeats for 32 consecutive rows; after that, a new 16-byte segment appears. This pattern indicates that the device uses a repeated and structured scrambling

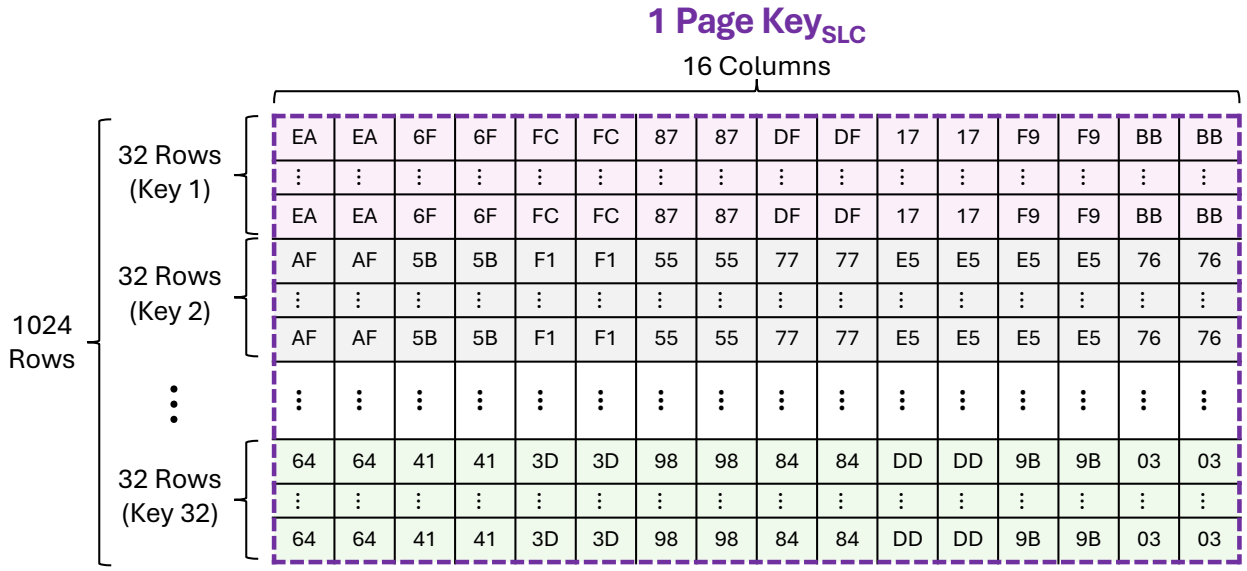


Figure 8.4: The scrambling key retrieved from an SLC page of a 32-layer 3D MLC NAND flash memory chip (MT29F256G08CBCBB).

scheme. Evaluating the remaining SLC pages in the block, we find that each SLC page contains a unique scrambling key.

8.3.2 Evaluation on 32-layer MLC 3D NAND Flash Chip

We implemented the proposed technique and extracted scrambling keys from a 32-layer 3D MLC NAND flash memory chip. The programming steps are slightly different in MLC storage due to its shared page architecture. The chip under test supports separate LSB and MSB programming steps as well as overwrite operations. Similar to the SLC case, an erased WL is selected, and the "0x55" data pattern is written into an LSB page of 16 kB size. Then, we read the same page to verify the success of the write operation. During this process, the cells' V_{th} values transition from the erased state (L_0, Figure 8.5(a)) to L_2 states or remain in the erased state based on the scrambled data pattern (Figure 8.5(b)), as illustrated by ❶. Next, we overwrite the same page with the "0xAA" pattern and perform another read. After this step, all cells' V_{th} shift to the L_2 states, as shown by ❷ in Figure 8.5(c). The readout was not "0xAA" but instead an apparently random data pattern, which, according to our proposed algorithm, corresponds to the scrambling key.

We next extract the scrambling key corresponding to the shared MSB page, starting from the state where the LSB page has been overwritten and all cells reside in the L_2 state. Following a similar procedure, we first write "0x55" to the MSB page, followed by an overwrite using the "0xAA" pattern, and finally perform a page read. The cells' V_{th} distributions for these two operations are shown in Figure 8.5(d)-(f). In this sequence, cells initially transition from L_2 state to either L_2 or L_3 states after writing "0x55" (❸ in Figure 8.5(e)) and then shift to the highest V_{th} state (L_3) after the "0xAA" overwrite (❹ in Figure 8.5(f)). The MSB readout values are therefore all '1's and, when XORed with the scrambler, yield the inverted key. Consequently, extracting the MSB key requires inversion of the readout values.

The resulting scrambling keys for the shared LSB and MSB pages of a 32-layer 3D MLC NAND flash memory chip (MT29F256G08CBCBB) are shown in Figure 8.6(a) and Figure 8.6(b), respectively. Note that Figure 8.6(b) shows the MSB key after inverting the readout data to obtain the true MSB key. The results are arranged similarly to the SLC key, revealing that both the LSB and MSB keys form a 16-byte sequence that repeats every 32 rows before a new 16-byte segment begins. Unlike the SLC key, some bytes exhibit bit flips (as shown with values in red font in the figures), likely due to factors such as the narrower V_{th} margin in MLC storage, which makes cells more susceptible to readout noise [30, 109, 110]. However, because the key repeats every 32

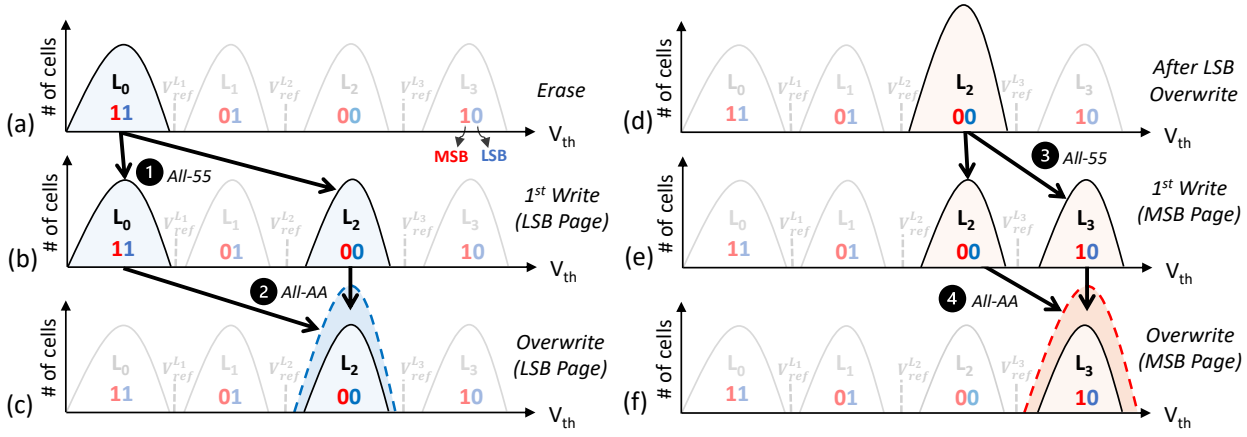


Figure 8.5: V_{th} distributions of the MLC NAND flash during the scrambling key extraction process for (a)-(c) LSB page and (d)-(f) MSB page.

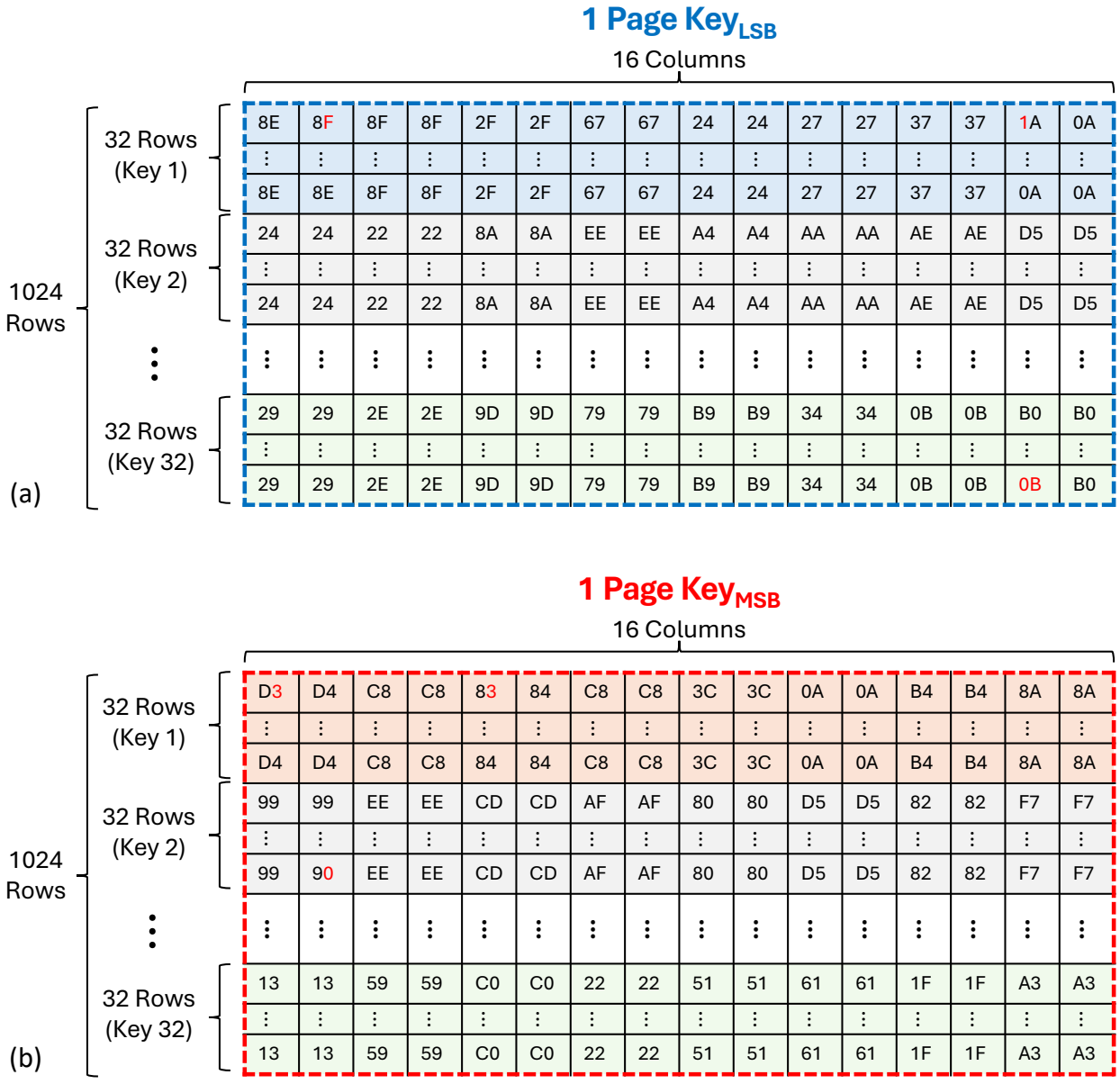


Figure 8.6: The resulting scrambling key derived from the shared (a) LSB and (b) MSB pages of a 32-layer 3D MLC NAND flash memory chip (MT29F256G08CBCBB).

rows, analyzing the majority trend across rows allows reliable identification of the underlying key sequence.

Based on the experimental results, we conclude that the proposed technique can successfully extract scrambling keys from both SLC and MLC 3D NAND flash memory devices, even when the keys are generated by an undocumented vendor-specific algorithm.

8.3.3 Evaluation on 2D NAND Flash Chips

We implemented the proposed descrambling method to extract scrambling keys from additional COTS 2D NAND flash memory chips spanning technology nodes from 34nm to 20nm. Across five different 2D NAND flash generations, distinct key scrambling behaviors were observed that differ markedly from 3D NAND devices. In particular, the extracted keys exhibit two characteristic types of repetition structures.

First, in a 2D 34nm MLC NAND flash (MT29F32G08CBACA), the LSB page key shows a clear repetitive pattern, as illustrated in Figure 8.7(a). Each unique 32-byte sequence repeats identically for 16 consecutive lines before transitioning to a new repeated sequence. In addition, we find that, unlike 3D NAND, the 2D NAND device contains a total of 128 distinct keys within a single LSB page. This behavior appears consistently in both the shared LSB and MSB pages. A nearly identical repetition scheme was observed in the 2D 25nm MLC NAND flash (MT29F16G08CBACA).

Secondly, Figure 8.7(b) shows the retrieved keys for the LSB page of a 2D 28nm MLC NAND flash chip (MT29F64G08CBAAA). In this device, the repeated structure consists of a 64-byte pattern that recurs 16 times before transitioning to the next repeating segment. Within each segment, every four bytes are repeated once. Further analysis of two additional 2D 20nm NAND chips (MT29F32G08CBADA and MT29F64G08CBABA) revealed that both devices produce nearly identical keys for their LSB and MSB pages. It is important to note that failed byte locations may vary from chip to chip; however, the dominant byte values across consecutive lines still allow correct key reconstruction despite such failures.

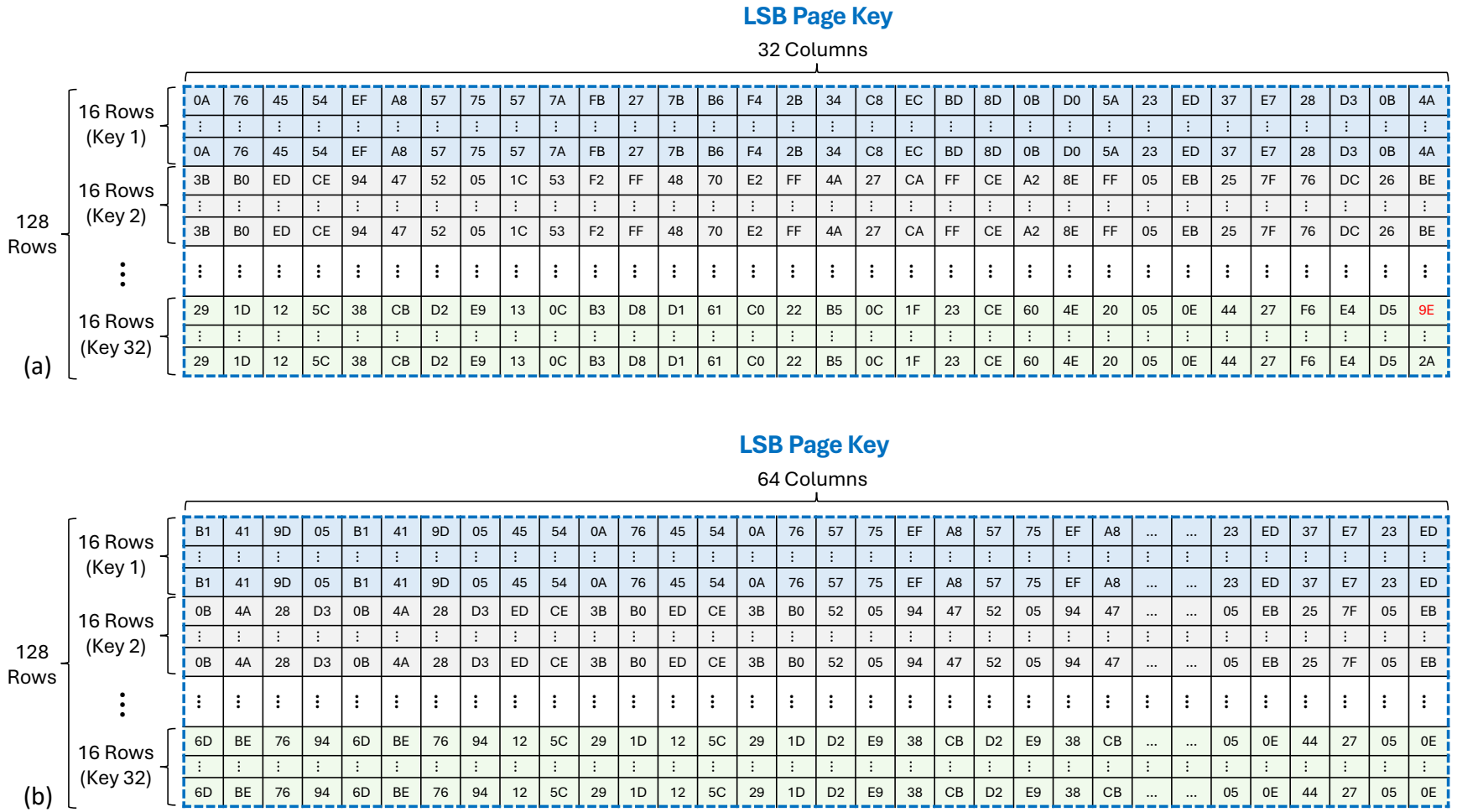


Figure 8.7: The resulting scrambling keys derived from the shared LSB pages of (a) 2D 34nm MLC NAND (MT29F32G08CBACA), with comparable behavior observed in 2D 25nm MLC NAND (MT29F16G08CBACA), and (b) 2D 28nm MLC NAND (MT29F64G08CBAAA), where similar periodicity is observed in two 2D 20nm MLC NAND flash devices (MT29F32G08CBADA and MT29F64G08CBABA).

8.3.4 Multi-Chip Evaluation

Table 8.1 summarizes the extracted scrambling key characteristics from all NAND flash chips, including chip information, part numbers, storage types, key lengths, the number of unique keys per page, and the uniqueness of keys at the block and chip levels. Note that three different NAND part numbers (MT29F256G08CBCBB, MT29F32G08CBADA, and MT29F64G08CBABA) contain a small number of WLs that operate in SLC storage even when the chip is configured for MLC operation. For these implicitly SLC-type WLs, our method successfully retrieves the corresponding scrambling keys.

Table 8.1: Summary of the scrambled key.

Chip info.	Part number	Storage type	Key length	# of keys/page	Uniqueness*	# of parts
3D 32-layer	MT29F256G08CBCBB	SLC	16 bytes	32	Yes	5
2D 20nm	MT29F32G08CBADA	SLC	64 bytes	8	Yes	4
2D 20nm	MT29F64G08CBABA	SLC	64 bytes	8	Yes	4
3D 32-layer	MT29F256G08CBCBB	MLC	16 bytes	32	Yes	5
2D 34nm	MT29F32G08CBACA	MLC	32 bytes	8	Yes	3
2D 28nm	MT29F64G08CBAAA	MLC	64 bytes	8	Yes	3
2D 25nm	MT29F16G08CBACA	MLC	32 bytes	8	Yes	4
2D 20nm	MT29F32G08CBADA	MLC	64 bytes	8	Yes	4
2D 20nm	MT29F64G08CBABA	MLC	64 bytes	8	Yes	4

*Uniqueness indicates that the keys are unique across blocks within a chip and chips of the same part number.

8.4 Properties of the Scrambling Key

We evaluated the properties of the scrambling key across all pages within a 3D NAND flash memory block, as well as across different blocks and five different chips with the same part number. Our experiments reveal several unique properties of the keys, which are summarized as follows:

- **Different keys across different pages of a block:** Every SLC page and every shared LSB/MSB page contain a different set of keys. Thus, we believe that the physical page number (PPN) within a given block plays a crucial role in deciding the key value. The 32-layer 3D

NAND flash chip (part # MT29F256G08CBCBB) contains 1,024 pages distributed across 32 layers and 16 different sub-blocks, and each of these pages inherits a distinct key.

- **Same set of keys across different blocks:** Different 3D NAND blocks within the same chip share an identical set of scrambling keys. Thus, the physical block number of a page does not play any role in the key generation algorithm. In other words, extracting the keys from one block is sufficient to recover the keys for all remaining blocks in the NAND flash device.

- **Same keys across chips of the same part number:** Different 3D NAND chips with the same manufacturing part number produce the same keys. Therefore, once the keys are extracted from one chip, those keys apply to all chips of the same part number.

8.5 Conclusion

This chapter presents the first experimental demonstration of scrambling key extraction from unmodified COTS 2D and 3D NAND chips. By controlling the page programming sequence to drive all cells to a known physical state, we successfully recover the internal scrambling keys used by the device's randomization engine. The recovered keys exhibit consistent structural patterns, with lengths of 16 to 64 bytes and periodic repetition. We also find that the keys are page-dependent, block-independent, and consistent across chips of the same part number, enabling full-key reconstruction from a single block or device. Additional experiments across 2D and 3D MLC NAND chips confirm that the same behavior persists, demonstrating the generality and robustness of the proposed method.

Chapter 9

Decoding Gray-Code Encodings via Partial-Program Timing

Modern NAND flash employs logical scaling to increase storage density, allowing each memory cell to store multiple bits by precisely controlling its V_{th} distribution. However, this density improvement comes at the cost of reduced reliability in NAND flash, primarily due to the smaller voltage margin between adjacent V_{th} states [42–45]. Figure 9.1 illustrates the logical encoding of V_{th} states using TLC memory as an example, where Figures 9.1(a)-(b) show representative V_{th} distributions for CT and FG NAND devices, respectively. TLC NAND flash with eight V_{th} levels contains L_0 to L_7, ranging from the erased state (L_0, with the least stored charge) to the highest programmed state (L_7, with the maximum stored charge). Each cell's V_{th} level is determined by the triplet of MSB/CSB/LSB bits [46]. To clearly illustrate page types, we use the following color code: blue for LSB, green for CSB, and red for MSB. Note that even though both plots represent TLC storage, their V_{th} encodings differ significantly. For instance, the L_7 state corresponds to the '110' triplet in Figure 9.1(a), but to '101' in Figure 9.1(b). The choice of a specific V_{th} encoding is vendor-dependent and proprietary. In general, it is a complex optimization problem that balances reliability, inter-cell interference, read latency, program efficiency, and endurance, while staying compatible with page sequencing, ECC, and data-scrambling logic.

Understanding how logical states are organized in NAND flash is important for system integrators who seek to improve data reliability and optimize error management. For example, the highest programmed state is known to experience faster charge loss than lower states, promoting error rates on specific logical pages [4, 47, 48]. Therefore, identifying the device-specific logical mapping of V_{th} states can notify system-level techniques to mitigate intrinsic V_{th} drift, enabling the design of more effective ECC [35] and adaptive read-reference schemes, eventually improving device endurance [49]. Recently, it was also demonstrated that instant data sanitization of ML-

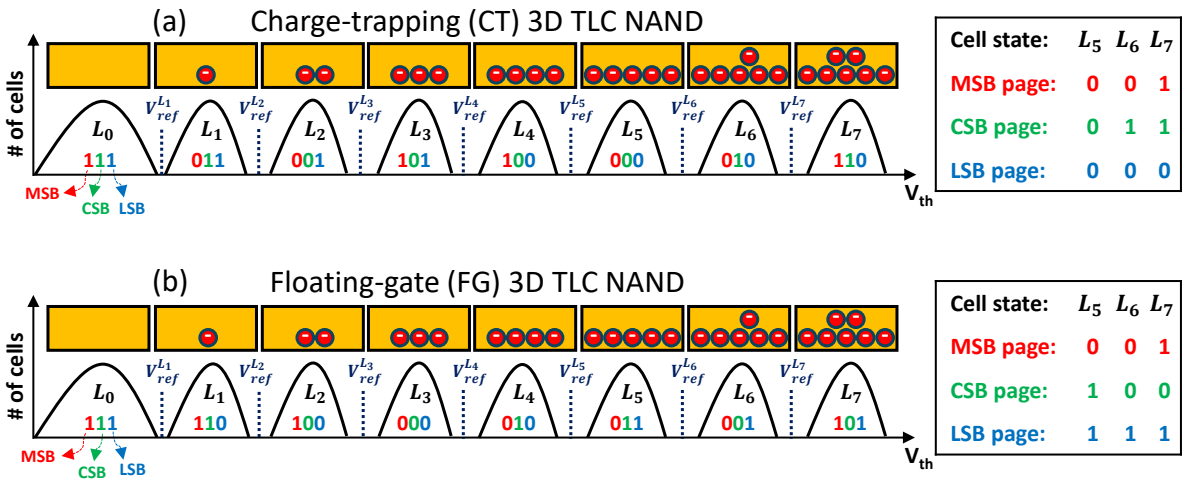


Figure 9.1: V_{th} distributions of 3D TLC NAND flash chips with (a) CT and (b) FG technologies, showing their corresponding Gray logical encoding values.

C/TLC memory pages is possible with the knowledge of cell V_{th} encoding [28, 50]. In addition, a clear understanding of state encoding schemes can significantly benefit forensic analysis of NAND flash devices [41, 111].

The logical mapping of programmed V_{th} states is typically proprietary and not disclosed in datasheets. While vendors generally employ Gray encoding, so adjacent V_{th} states differ by a single bit, the number of valid Gray assignments grows with the number of stored bits per cell, making the mapping more complex. Therefore, a systematic method to identify the logical encoding scheme of commercial NAND flash devices would greatly benefit both system integrators, who seek to optimize reliability and performance, and forensic analysts, who require deeper insight into device behavior.

9.1 Related Work

Recently, there have been several studies on developing technology-specific Gray Code to enhance read/write performance of QLC storage [58, 112, 113]. However, reverse engineering of the underlying Gray code from the commercial device is still mostly unexplored. In 2013, Ingalls et al. [114] proposed a method to determine the organization of logical states in MLC memories via

radiation response. One can infer the sequence of V_{th} levels associated with each logical state by monitoring data corruption (i.e., failed bits) caused by radiation exposure. However, the method requires specialized access to a radiation facility, limiting its practicality. Similarly, certain test-mode commands allow direct measurement of the programmed cell V_{th} distribution, which can reveal the underlying state-encoding scheme [48]. However, these features are not accessible during normal user-mode operation, making it impossible to decode the state mapping without explicit vendor support.

9.2 Key Ideas

This chapter presents the experimental demonstration of logical state mapping extraction from unmodified COTS NAND flash chips. The key contributions are summarized as follows:

- To the best of our knowledge, this is the first work to determine the logical state mapping of NAND flash using vendor-agnostic user-mode commands. The proposed algorithm leverages partial-program operations and subsequent shared page reads to decode the underlying V_{th} states.
- We evaluate the effectiveness of the proposed algorithm on COTS 2D and 3D NAND flash chips, including both FG and CT cell technologies, as well as MLC, TLC, and QLC configurations from major NAND vendors, such as Toshiba, Micron, Intel, and SK Hynix.
- The experimental results show that the proposed technique reliably reveals these device-specific mappings, providing a low-cost and easily implementable solution.

9.2.1 Partial–Program-Based Decoding Technique

Figure 9.2 conceptually illustrates the proposed cell V_{th} decoding methodology using TLC as an example. Initially, when a block is in the erased state, its cell V_{th} values cluster near the erased distribution, and the corresponding logical bit values for all pages are set to '1,' as shown in Figure 9.2(a). To initiate programming, data for all three pages sharing the WL must be provided. In our

decoding approach, we use a fixed three-page solid data pattern intended to drive cells toward a single target level under full program. Our goal is to program all cells to the highest program V_{th} state (L_7 in case of TLC). In this particular example, that pattern is LSB = all-ones, CSB = all-zeros, and MSB = all-ones. If a full program is executed with this pattern, all cells on the selected WL transition to the same V_{th} state (L_7).

To reveal the logical state ordering without prior knowledge of the mapping, we use a partial-program operation [50, 115], which intentionally terminates the page-programming process prematurely by issuing a RESET command after a predefined delay, referred to as the t_{pp}.

Figure 9.2(b) shows the V_{th} distribution after a partial program with a short t_{pp}. In this case, most cells have advanced only to the first programmed V_{th} level (L_1). Subsequent page-read operations reveal the associated bit triplet. For example, in Figure 9.2(b), reading MSB, CSB, and LSB pages shows MSB = all-ones, CSB = all-ones, and LSB = all-zeros. Thus, the L_1 state encodes '110' under this ordering. As t_{pp} gradually increases during partial-program operation, the cell's V_{th} values shift rightward. This progression continues until the highest programmed state (L_7 for TLC) is reached; cells cannot be programmed beyond this level. Leftward transitions between programmed states do not occur under program operations, as indicated by the black arrows. A block-erase operation resets all cells to the L_0 state.

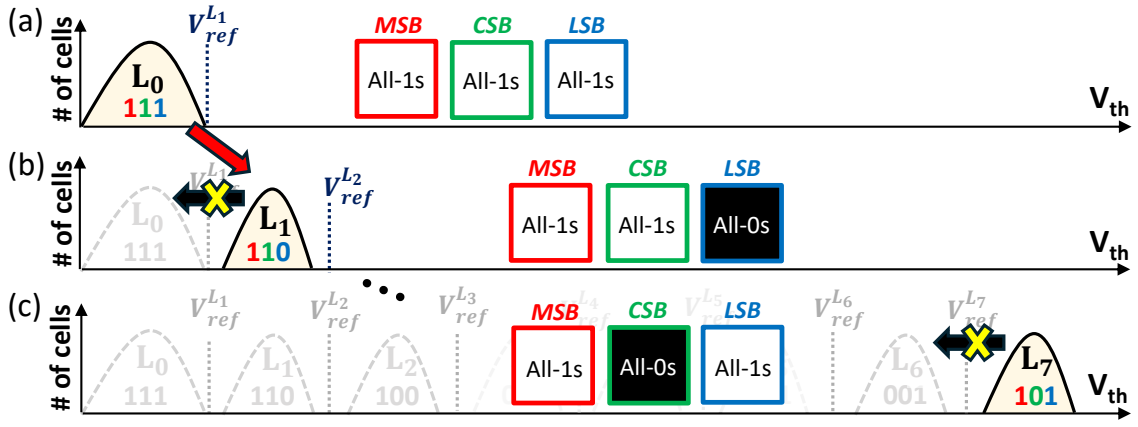


Figure 9.2: Transition of flash cell V_{th}: (a)-(b) rightward shift during partial program, and (c) leftward shift during erase.

Choosing the t_{pp} is a key design parameter in the proposed decoding method. We gradually increase t_{pp} so that the WL population steps through each adjacent Gray state. Precise t_{pp} values are not required: any value within a range that causes a measurable fraction of cells to cross the next boundary is sufficient to label that V_{th} state. For example, if the t_{pp} used in Figure 9.2(b) is slightly shorter than the optimal value, only a fraction of the LSB bits flip to 0, while all MSB and CSB bits remain at 1. Nevertheless, this partial transition still identifies L_1 as '110' because Gray encoding constrains one-bit changes between adjacent states. A detailed t_{pp} tuning procedure is presented in the following section. The key insight is that partial-program timing and Gray encoding together ensure that observing page-level bit patterns at successive t_{pp} values reveals the ordering of V_{th} states.

9.2.2 V_{th} Logical-State Decoding Algorithm

We divide our V_{th} logical-state decoding technique into two stages. Stage 1 identifies the logical doublet/triplet/quadruplet corresponding to the highest programmed V_{th} level in MLC/TLC/QLC storage. Stage 2 decodes all remaining states using a sequence of partial program operations.

Using TLC as an example for Stage 1 (Figure 9.3(a)), we first erase the target flash block, initializing all pages so they read as all '1's. Then, the LSB, CSB, and MSB pages are programmed with each of seven solid data-pattern combinations (excluding '111'), e.g., LSB = all-ones, CSB = all-zeros, MSB = all-zeros. For each combination, we load the three-page buffers and issue a full page-program sequence in the device's page order.

Once a cell reaches the highest programmed state (L_7), its V_{th} saturates, and further programming cannot change its state. Since programming only increases V_{th}, the order of programming the seven combinations does not affect L_7 identification. After all seven data patterns are programmed, page-read operations retrieve the stored data from the LSB, CSB, and MSB pages. For example, if the readout shows all-zeros in MSB and CSB and all-ones in LSB, the highest logical state corresponds to '001.'

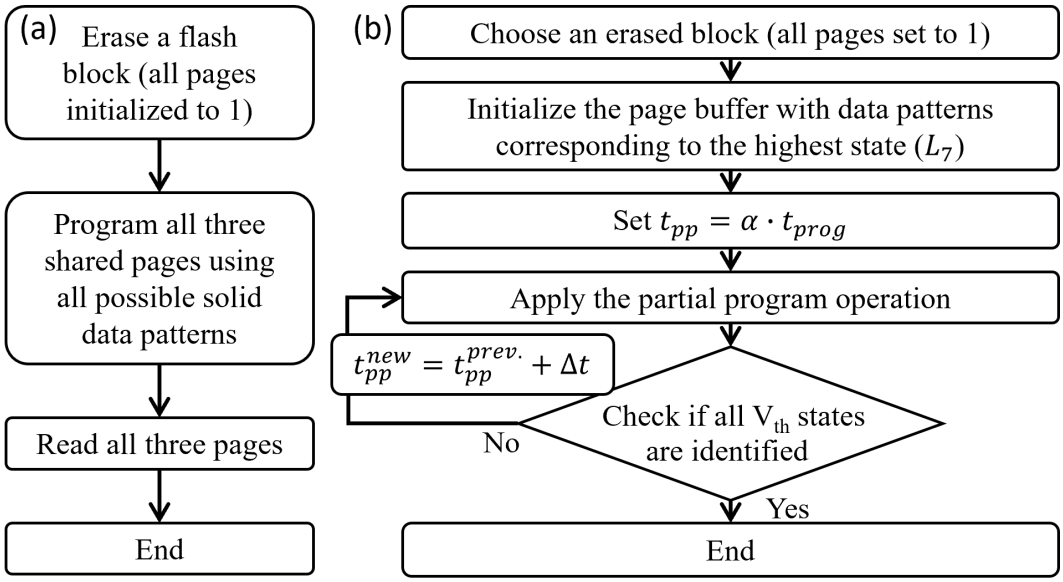


Figure 9.3: (a) Steps to identify the logical state corresponding to the highest programmed V_{th} state in TLC (L_7). (b) Algorithm to decode logical states of the V_{th} distribution.

The same approach applies to QLC NAND. After erasing the block, we fully program the LSB, CSB, MSB, and TSB (the top significant bit) pages with the fifteen solid patterns (excluding '1111'). Once cells reach the highest programmed state (L_{15}), further programming no longer occurs. Page-read operations of the four pages reveal the highest logical state.

After identifying the logical value corresponding to the highest programmed state, the next step (Stage 2) is to decode the remaining levels using a sequence of partial program operations. The overall flow is shown in Figure 9.3(b). The procedure begins by selecting one WL in an erased block. The page buffers are then loaded with the solid data pattern that corresponds to the highest logical state identified in the previous stage. Next, a partial program operation is performed by prematurely interrupting the programming process with a RESET command (0xFF) after a controlled delay (t_{pp}). Here, t_{pp} is a tunable parameter. Its initial value can be set as a fraction of the nominal page program time (t_{prog}), expressed as $t_{pp} = \alpha \cdot t_{prog}$, where α is a design parameter typically initialized at ≈ 0.05. If this initial t_{pp} does not provide sufficient state change, the partial program operation can be repeated with a longer t_{pp}, incremented by Δt. In this context, Δt is a constant value that can be chosen as any value in the range 50 μs to 100 μs. The parameters Δt

and α vary across NAND storage types, following the relationships ΔtMLC > ΔtTLC $\geq$ ΔtQLC and αQLC > αTLC $\geq$ αMLC. The algorithm terminates once all distinct V_{th} states have been accurately identified through read operations.

9.3 Experimental Results

9.3.1 Evaluation on MLC NAND

The state encoding for MLC NAND is straightforward, as only two bits must be mapped to four V_{th} levels. Most commercial chips follow this Gray-coded sequence: '11' (L_0, erased), '01' (L_1), '00' (L_2), and '10' (L_3), which supports two-step programming and improves LSB read performance. Although well understood, we still apply our decoding algorithm to MLC devices to demonstrate its generality across storage technologies.

Figure 9.4(a) shows results from Micron 64-layer 3D FG MLC WL, plotting the percentage of '1' bits observed on the shared LSB and MSB pages as a function of t_{pp}. At small t_{pp} values (region W_0), both pages remain fully erased and exhibit 100% '1's, confirming the encoding of L_0 state as '11.' As t_{pp} enters the time interval defined by window W_1 in the figure, the MSB page

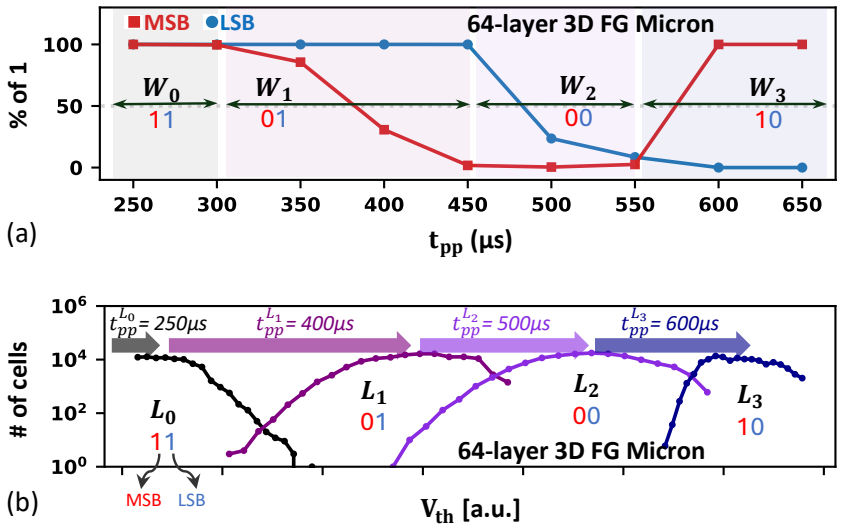


Figure 9.4: (a) Percentage of '1' as a function of t_{pp}. (b) Corresponding V_{th} distribution under varying programming pulse durations.

shows a sharp reduction in '1's while the LSB page remains at 100% '1's, revealing L_1 as '01.' In the time window W_2, the LSB page experiences the next major drop while the MSB page remains at 0% '1's, indicating that L_2 is encoded as '00.' Finally, in window W_3, the MSB page shows an increase in '1's while the LSB page remains at 0%, confirming L_3 encoding as '10.' These results match the expected Gray sequence, validating the effectiveness of the proposed technique for MLC NAND. Figure 9.4(b) shows the corresponding V_{th} distribution (for t_{pp} ranging from 250 µs to 600 µs), with each curve illustrating the rightward V_{th} shift as t_{pp} increases.

9.3.2 Evaluation on TLC NAND

Figure 9.5 presents the state-decoding characterization results for two different TLC chips. Specifically, Figure 9.5(a) and (c) correspond to 64-layer CT memory, whereas Figure 9.5(b) and (d) correspond to 64-layer FG memory. Similar to MLC decoding, we measure the percentage of '1' bits observed in three shared pages after partial write as a function of t_{pp} in Figure 9.5(a). The figure legend captures page types: LSB-blue circles, CSB-green triangles, MSB-red squares. Initially, all cells are in the erased state, corresponding to the triplet '111' across the three pages. After the first partial-program step (t_{pp} = 100 μs), the cells remain in the erased state. At t_{pp} = 200 µs, the LSB page shows a high fraction of '0's while CSB and MSB remain all '1's. Thus, we decode the L_1 state as '110.' As the partial-program pulse duration is further increased, most of the bits in the CSB page change from '1' to '0,' indicating the logical state for L_2 is '100.' This one-bit-at-a-time progression continues with subsequent programming pulses, with each step pushing cells to the next programmed state. Eventually, at t_{pp} = 1.10 ms, the cells reach the highest programmed state (L_7), corresponding to the logic value '101.' At this point, the CSB page reads all '0's, while the other two pages read all '1's. Note that the regions labeled W_0-W_7 are defined by the t_{pp} interval in which the percentage of '1' bits changes only in one page, indicating V_{th} transitions between two adjacent states.

Figure 9.5(b) shows the same procedure on a 64-layer FG TLC device. The recovered sequence and t_{pp} values differ. For example, after applying a t_{pp} of 300 μs, state L_1 corresponds to the triplet

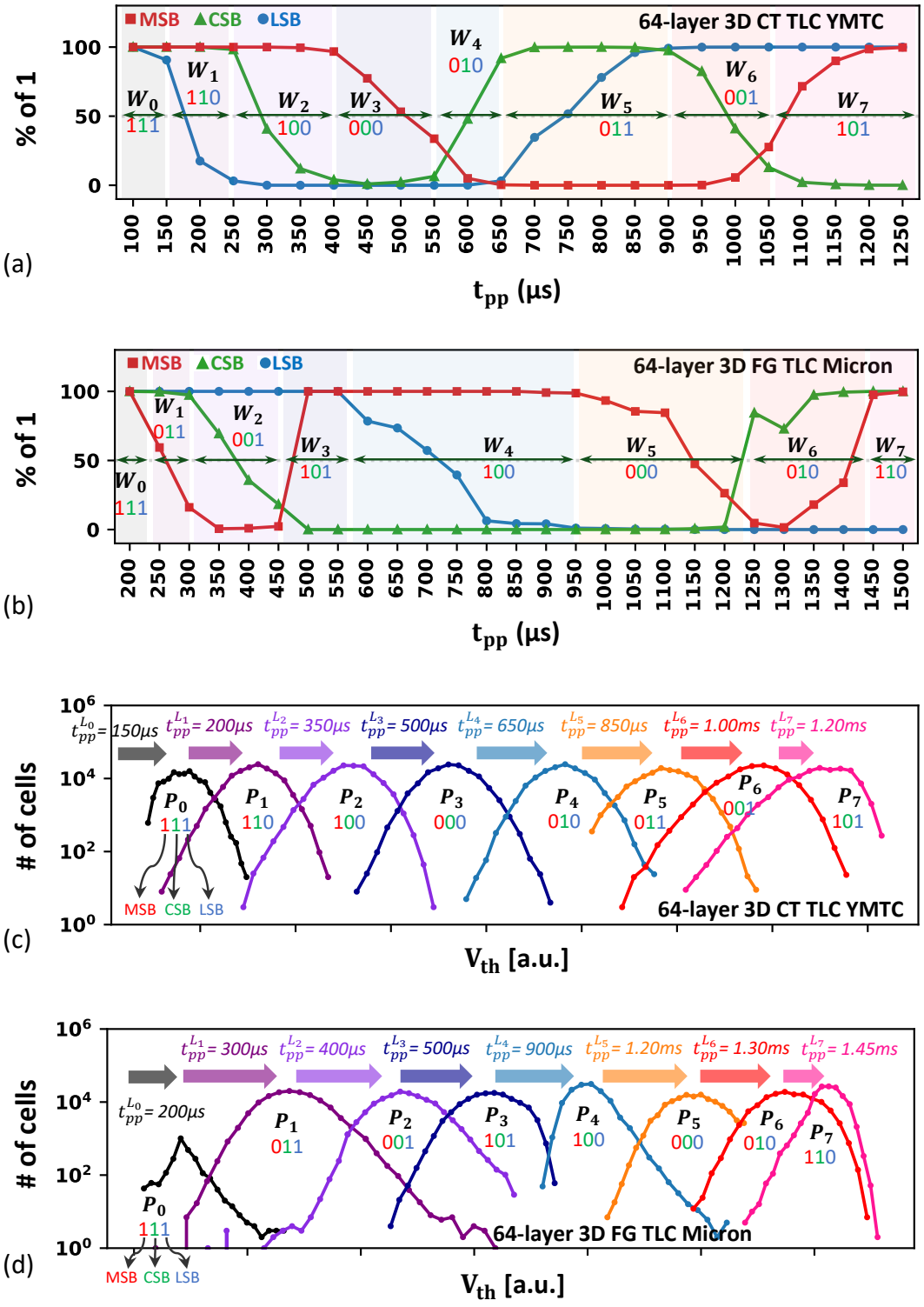


Figure 9.5: Percentage of cells in the logic '1' state as a function of tpp for 64-layer 3D TLC (a) CT and (b) FG NAND flash memories. The measured V_{th} distributions under varying programming pulse durations for (c) the corresponding CT and (d) FG.

‘011,’ while the highest programmed state L_7 yields ‘110.’ This illustrates that while adjacent transitions remain single-bit (Gray), the page-to-level assignment is vendor- and device-dependent. Note that the t_{pp} values shown on the x-axes of Figures 9.5(a)-(b) correspond to the delay parameters used in the NAND-interfacing firmware to trigger partial-program operations. The actual duration of the program pulse applied to the NAND array may differ from these firmware settings because several factors influence the effective programming time, including the RESET-command latency, the underlying ISPP scheme, and timing variability introduced by the host-side firmware execution.

Figures 9.5(c)-(d) present measured V_{th} distributions for the 64-layer CT (t_{pp} from 150 μs to 1.20 ms) and the FG (t_{pp} from 200 μs to 1.45 ms) devices, respectively. There is a one-to-one correspondence between these V_{th} distributions and the measured bit values shown in Figures 9.5(a)-(b), respectively. As the programming-pulse duration increases, the V_{th} distributions shift rightward, indicating that the cells are being programmed to higher voltage stages, which in turn results in changes to the logical state. These distribution plots are used for visualization and confirmation; the decoding itself relies only on page reads and the Gray one-bit adjacency.

Together, the results in Figures 9.5(a)-(d) confirm that our t_{pp}-based sweep approach reconstructs the logical values assigned to each V_{th} level, from erased state (L_0 or ‘111’) to the highest programmed level (L_7) on both 64-layer 3D TLC CT and FG chips, validating the proposed method across distinct technologies and vendor mappings.

9.3.3 Evaluation on QLC NAND

In this section, we evaluate the proposed algorithm on QLC storage. The experiments were conducted on a 176-layer 3D CT QLC NAND chip. In this device, most WLs in a block operate in QLC mode, while several edge WLs operate in MLC mode.

Figure 9.6 presents the characterization results for V_{th} decoding in QLC storage. Note that QLC storage contains four shared pages per WL, corresponding to LSB, CSB, MSB, and TSB pages. The percentage of ‘1’ bits in these pages is shown in the plot using different color codes as

a function of t_{pp}: blue for LSB, green for CSB, red for MSB, and purple for TSB. We follow the same partial-program-based V_{th} decoding technique as in the MLC and TLC evaluations. All four shared pages are initialized with all '1's, corresponding to the erased state. By increasing t_{pp}, only one page changes its content at a time, thereby revealing the corresponding V_{th}-state encoding. Note that QLC programming requires a significantly longer duration than MLC and TLC storage to reach the highest V_{th} state.

The results in Figures 9.4-9.6 confirm that the proposed t_{pp} sweep reconstructs the logical values corresponding to each V_{th}-level. The technique is applicable to various types of NAND chips, including older and newer generations, without requiring any hardware modifications and using only standard page reads combined with controlled partial-program operations.

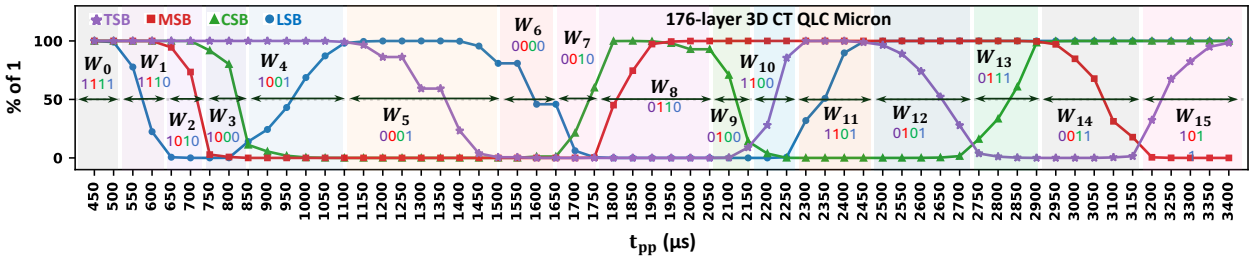


Figure 9.6: Percentage of cells in the logic '1' state as a function of t_{pp} for Micron 176-layer 3D CT NAND flash in QLC mode.

9.3.4 Multi-Chip Evaluation

We implemented the proposed algorithm on multiple COTS MLC, TLC, and QLC NAND chips. All MLC NAND chips evaluated in this paper follow the same logical value sequence shown in Section 9.3.1. The results, summarized in Table 9.1 (TLC chips) and Table 9.2 (QLC chips), show that the device-specific Gray assignments (i.e., the logical values associated with each V_{th} level) vary across vendors. The diversity of mappings in Tables 9.1 and 9.2 demonstrates that our decoding algorithm is compatible with a wide range of NAND flash devices without requiring any hardware changes.

Table 9.1: Logic state mapping for different TLC NAND flash chips using our proposed V_{th} decoding algorithm

Part number	TLC chip	L_0	L_1	L_2	L_3	L_4	L_5	L_6	L_7
H27QFG8PEM5R	2D 16nm FG SK Hynix	111	011	001	000	010	110	100	101
MT29F256G08EBHAF	64-layer FG Micron	111	011	001	101	100	000	010	110
YMN08TE1B1DC3B	64-layer CT YMTC	111	110	100	000	010	011	001	101
TH58TFT1V23BA8H	64-layer CT Toshiba	111	110	100	000	010	011	001	101
SDTNBIAMA032G	64-layer CT SanDisk	111	110	100	000	010	011	001	101
YMN09TC1B1JC6C	128-layer CT YMTC	111	110	100	000	010	011	001	101
H25T3TM88C-X383	128-layer CT SK Hynix	111	011	001	000	010	110	100	101
H25G9TC18C-X488	176-layer CT SK Hynix	111	011	001	000	010	110	100	101
MT29F1T08EELKEJ4	176-layer CT Micron	111	110	100	000	010	011	001	101

Table 9.2: Logic state mappings in QLC NAND flash memory chips for (1) 96-layer 3D Micron (MT29F2T08GELBEJ4), (2) 144-layer 3D Intel (29F08T2A0CQKI), and (3) 176-layer 3D Micron (MT29F4T08GMLCEJ4)

#	L_0	L_1	L_2	L_3	L_4	L_5	L_6	L_7	L_8	L_9	L_{10}	L_{11}	L_{12}	L_{13}	L_{14}	L_{15}
(1)	1111	0111	0011	1011	1001	0001	0101	1101	1100	0100	0000	1000	1010	0010	0110	1110
(2)	1111	0111	0011	1011	1001	1101	0101	0001	0000	1000	1100	0100	0110	0010	1010	1110
(3)	1111	1110	1010	1000	1001	0001	0000	0010	0110	0100	1100	1101	0101	0111	0011	1011

9.4 Limitations and Challenges

Our algorithm can be applied to COTS NAND. However, successful deployment depends on several considerations.

Data randomization: Some NAND chips employ data randomizers or scramblers that XOR user data with a pseudo-random sequence to reduce pattern sensitivity and wear. These randomizers can interfere with the solid data patterns required during partial program operations, potentially causing inaccurate interpretation of cell states. However, most raw NAND chips do not incorporate this feature, as vendors typically provide more flexibility to system integrators by allowing them to implement their own FTL. Therefore, the proposed algorithm remains applicable to most raw NAND available today.

Permissions to overwrite in closed block: Our technique involves repeated partial programming of a single WL, causing the target WL to undergo multiple successive programming cycles. This process, often referred to as an "overwrite" operation, involves reprogramming already-programmed pages. Some NAND chips restrict overwrite operations, which would prevent the application of our method. However, none of the raw NAND chips evaluated in this study exhibit such restrictions, allowing our algorithm to be implemented successfully.

9.5 Conclusions

This paper presents a vendor-agnostic technique for determining the logical-to-physical state mapping in NAND flash using partial-program operations. Without requiring privileged commands or hardware changes, the method gradually moves cells from erased (L_0) to the highest programmed level (L_{2^n-1}) by sweeping the partial-program time. At each step, the standard page reads are used to obtain the n-bit value corresponding to the reached level. Experimental evaluations on multiple COTS 2D/3D NAND devices across vendors, cell technologies (CT and FG), and storage types (MLC, TLC, and QLC) demonstrate that the proposed technique is broadly applicable and highly effective, recovering device-specific Gray assignments using only ONFI-standard program/erase/read/reset sequences.

Chapter 10

Conclusions and future work

10.1 Conclusions

This thesis investigated security vulnerabilities in commercial 3D NAND flash memory across two themes: the inadequacy of standard data sanitization and the feasibility of extracting proprietary device properties using only user-accessible commands. All results were obtained from unmodified, commercially available devices without hardware modification or manufacturer support.

Chapter 4 demonstrated that page overwrite induces disturbance-driven errors on adjacent valid pages and proposed PULSE as a low-disturbance alternative. Chapters 5–7 showed that block erase sanitization, the sanitization method endorsed by NIST SP 800-88 Rev. 1, fails to guarantee secure data removal in modern 3D NAND. Chapters 5 and 6 provided the first experimental evidence that block erase is insufficient in 3D CT and FG SLC NAND, respectively, with 70–80% of original content recoverable via read-offset operations after a recovery bake. Chapter 7 extended this finding to FG MLC storage, recovering approximately 65% of LSB page data. Across all device types, recovery exploits analog V_{th} asymmetries that survive logical erasure, residual hole trapping in CT NAND, and asymmetric oxide defect-state evolution in FG NAND; none of these is eliminated by a standard block erase.

Chapters 8 and 9 addressed reverse engineering vulnerabilities. Chapter 8 demonstrated that proprietary on-chip scrambling keys can be fully extracted from unmodified COTS devices by moving NAND cells to a specific V_{th} state, with extracted keys shown to be block-independent and consistent across chips of the same part number. Chapter 9 introduced a vendor-agnostic technique for decoding proprietary logical-to-physical state mappings (Gray-code encodings) using controlled partial-program timing, validated across CT and FG devices in MLC, TLC, and QLC storage types.

Collectively, these findings demonstrate that standard block erase and overwrite sanitization methods are insufficient to prevent data recovery from modern 3D NAND flash memory, and that proprietary device properties can be extracted using only user-accessible commands. The root cause lies in the analog physics of charge storage, which persists beneath the digital abstraction presented to the host system and remains accessible through standard NAND command sequences. These results highlight the need to revisit current sanitization standards and security assumptions for flash-based storage, and motivate the development of more robust, physics-aware sanitization protocols for modern NAND devices.

10.2 Future Work

This section discusses future research directions for addressing remaining challenges and evaluating our method on a broader range of 3D NAND flash storage devices.

10.2.1 Block Erase Vulnerabilities

Extending the data recovery work on higher logical density storage: While this work demonstrates data leakage in SLC storage, the underlying physical mechanism is not limited to single-level operation. In higher-density TLC and QLC NAND, the LSB pages are typically decoded using a single V_{ref}, and multiple lower programmed states reside below this reference. Prolonged data retention in these states can similarly lead to hole accumulation and differential charge behavior, potentially enabling partial or full data recovery from the LSB pages even after sanitization. Future research will systematically investigate this vulnerability across higher logical densities, characterize its dependence on program state distributions and retention time, and evaluate mitigation strategies that extend beyond SLC-based sanitization assumptions.

Careful evaluation of thermally assisted sanitization: While our results demonstrate the effectiveness of thermally assisted sanitization, the required bake duration remains significant. Promising directions include combining thermal treatment with controlled programming using carefully chosen data patterns during the sanitization process; for example, writing inverse or

stress-inducing patterns to accelerate charge neutralization. Exploring such hybrid sanitization techniques may substantially reduce the sanitization time while preserving strong data-erasure guarantees.

Alternate sanitization technique: Future research should systematically evaluate alternative data sanitization mechanisms and secure disposal protocols for flash-based storage devices. Beyond thermal recovery techniques, our findings suggest that ionizing radiation can effectively neutralize residual charge states responsible for data remanence. In particular, controlled X-ray-based sanitization may offer a faster and more uniform means of eliminating latent data across large flash arrays. Future studies should characterize the required radiation dose, exposure duration, and potential device-level side effects, as well as assess the practicality, safety, and cost of deploying radiation-assisted sanitization as a scalable alternative to conventional erase-based methods.

10.2.2 Reverse Engineering

Evaluating the proposed descrambling technique on other manufacturers and newer technologies: Future work should evaluate the proposed descrambling method across a broader range of NAND flash memory devices, including chips from different vendors and newer technology generations. In addition, its effectiveness on higher-density storage technologies, such as TLC and QLC NAND, remains unexplored.

Bibliography

[1] Jeong Wook (Matt) Oh. Reverse engineering flash memory for fun and benefit. https://blackhat.com/docs/us-14/materials/us-14-Oh-Reverse-Engineering-Flash-Memory-For-Fun-And-Benefit-WP.pdf. [Accessed: Apr. 30, 2026].

[2] Akash Kumar, Jitendrabhai Ardeshana, and Santosh Jagtap. Design & verification of ONFI complient high performance NAND flash controller. In *2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, pages 942–945, 2016.

[3] Richard Kissel, Andrew Regenscheid, Matthew Scholl, and Kevin Stine. Guidelines for Media Sanitization. NIST Special Publication (SP) 800-88 Rev. 1, National Institute of Standards and Technology, December 2014.

[4] Yixin Luo, Saugata Ghose, Yu Cai, Erich F. Haratsch, and Onur Mutlu. HeatWatch: Improving 3D NAND Flash Memory Device Reliability by Exploiting Self-Recovery and Temperature Awareness. In *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 504–517, 2018.

[5] Yu Cai, Saugata Ghose, Erich F. Haratsch, Yixin Luo, and Onur Mutlu. Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives. *Proceedings of the IEEE*, 105(9):1666–1704, 2017.

[6] Yu Cai, Erich F. Haratsch, Onur Mutlu, and Ken Mai. Error patterns in MLC NAND flash memory: Measurement, characterization, and analysis. In *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 521–526, 2012.

[7] Mordor Intelligence. NAND Flash Memory Market Size & Share Analysis - Growth Trends And Forecast (2026-2031). https://www.mordorintelligence.com/industry-reports/nand-flash-memory-market, Jan. 2026. [Accessed: Apr. 30, 2026].

[8] Intelmarket Research. North America NAND Flash Market Growth Analysis, Dynamics, Key Players and Innovations, Outlook and Forecast 2026-2034. https://www.intelmarketresearch.com/north-america-nand-flash-market-market-41250, Apr. 2026. [Accessed: Apr. 30, 2026].

[9] kingspec. How KingSpec Meets the Demand for High-Capacity and Cost-Effective SSDs. https://www.kingspec.com/news/how-kingspec-meets-the-demand-for-high-capacity-and-cost-effective-ssds.html, Apr. 2025. [Accessed: Apr. 30, 2026].

[10] Laura M. Grupp, John D. Davis, and Steven Swanson. The bleak future of NAND flash memory. In *Proceedings of the 10th USENIX Conference on File and Storage Technologies*, FAST'12, page 2, USA, 2012. USENIX Association.

[11] Yu Cai, Saugata Ghose, Erich F. Haratsch, Yixin Luo, and Onur Mutlu. Errors in Flash-Memory-Based Solid-State Drives: Analysis, Mitigation, and Recovery. *ArXiv*, abs/1711.11427, 2017.

[12] Legislation.gov.uk. Data Protection Act 2018. https://www.legislation.gov.uk/ukpga/2018/12/contents/enacted, 2018. [Accessed: Apr. 30, 2026].

[13] virtual College. The data protection act (2018): Its purpose and principles. https://www.virtual-college.co.uk/resources/the-data-protection-act-2018, May. 2025. [Accessed: Apr. 30, 2026].

[14] Peter Gutmann. Secure deletion of data from magnetic and solid-state memory. In *Proceedings of the 6th Conference on USENIX Security Symposium, Focusing on Applications of Cryptography - Volume 6*, SSYM'96, page 8, USA, Jul. 1996. USENIX Association.

[15] Joel Reardon, David Basin, and Srdjan Capkun. SoK: Secure Data Deletion. In *2013 IEEE Symposium on Security and Privacy*, pages 301–315, Jun. 2013.

[16] Joel Reardon, David Basin, and Srdjan Capkun. On Secure Data Deletion. *IEEE Security & Privacy*, 12(03):37–44, May 2014.

[17] Jaeheung Lee, Junyoung Heo, Yookun Cho, Jiman Hong, and Sung Y. Shin. Secure deletion for NAND flash file system. In *Proceedings of the 2008 ACM Symposium on Applied Computing*, SAC '08, page 1710–1714, New York, NY, USA, 2008. Association for Computing Machinery.

[18] Wei-Chen Wang, Chien-Chung Ho, Yuan-Hao Chang, Tei-Wei Kuo, and Ping-Hsien Lin. Scrubbing-Aware Secure Deletion for 3-D NAND Flash. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(11):2790–2801, 2018.

[19] Blancco. Blancco reveals 42 [Accessed: Apr. 30, 2026].

[20] Blancco. Privacy for Sale: Data Security Risks in the Second-Hand IT Asset Marketplace, Apr. 2019. [Accessed: Apr. 30, 2026].

[21] Nikki Robins, Patricia A. H. Williams, and Krishnun Sansurooah. An Investigation into Remnant Data on USB Storage Devices Sold in Australia Creating Alarming Concerns. *International Journal of Computers and Applications*, 39(2):79–90, 2017.

[22] Martin Westman. Where Did That Incriminating Evidence Come From? In *DFRWS EU 2018*. DFRWS, 2018. [Accessed: Apr. 30, 2026].

[23] Janine Schneider, Immanuel Lautner, Denise Moussa, Julian Wolf, Nicole Scheler, Felix Freiling, Jaap Haasnoot, Hans Henseler, Simon Malik, Holger Morgenstern, and Martin Westman. In Search of Lost Data: A Study of Flash Sanitization Practices, 2025.

[24] Sarah Diesburg, Christopher Meyers, Mark Stanovich, Michael Mitchell, Justin Marshall, Julia Gould, An-I Andy Wang, and Geoff Kuenning. TrueErase: per-file secure deletion

for the storage data path. In *Proceedings of the 28th Annual Computer Security Applications Conference*, ACSAC '12, page 439–448, New York, NY, USA, 2012. Association for Computing Machinery.

[25] Joel Reardon, Claudio Marforio, Srdjan Capkun, and David Basin. User-level secure deletion on log-structured file systems. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '12, page 63–64, New York, NY, USA, 2012. Association for Computing Machinery.

[26] Kyoungmoon Sun, Jongmoo Choi, Donghee Lee, and Sam H. Noh. Models and Design of an Adaptive Hybrid Scheme for Secure Deletion of Data in Consumer Electronics. *IEEE Transactions on Consumer Electronics*, 54(1):100–104, 2008.

[27] Michael Wei, Laura M. Grupp, Frederick E. Spada, and Steven Swanson. Reliably erasing data from flash-based solid state drives. In *Proceedings of the 9th USENIX Conference on File and Stroage Technologies*, FAST'11, page 8, USA, 2011. USENIX Association.

[28] Md Raquibuzzaman, Matchima Buddhanoy, Aleksandar Milenkovic, and Biswajit Ray. Instant data sanitization on multi-level-cell NAND flash memory. In *Proceedings of the 15th ACM International Conference on Systems and Storage (SYSTOR'22)*, page 85–95, New York, NY, USA, 2022. Association for Computing Machinery.

[29] Michele Favalli, Cristian Zambelli, Alessia Marelli, Rino Micheloni, and Piero Olivo. A Scalable Bidimensional Randomization Scheme for TLC 3D NAND Flash Memories. *Micromachines*, 12, 2021.

[30] Jaewon Cha and Sungho Kang. Data Randomization Scheme for Endurance Enhancement and Interference Mitigation of Multilevel Flash Memory Devices. *ETRI Journal*, 35(1):166–169, 2013.

[31] Yu Cai, Yixin Luo, Saugata Ghose, and Onur Mutlu. Read Disturb Errors in MLC NAND Flash Memory: Characterization, Mitigation, and Recovery. In *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pages 438–449, 2015.

[32] Jaeyong Lee, Myungsunk Kim, Wonil Choi, Sanggu Lee, and Jihong Kim. Tailcut: improving performance and lifetime of ssds using pattern-aware state encoding. In *Proceedings of the 59th ACM/IEEE Design Automation Conference*, DAC'22, page 409–414, New York, NY, USA, 2022. Association for Computing Machinery.

[33] Kyoji Mizoguchi, Shohei Kotaki, Yoshiaki Deguchi, and Ken Takeuchi. Lateral charge migration suppression of 3D-NAND flash by vth nearing for near data computing. In *2017 IEEE International Electron Devices Meeting (IEDM)*, pages 19.2.1–19.2.4, 2017.

[34] Fei Wang, Rui Cao, Yachen Kong, Xiaolei Ma, Xuepeng Zhan, Yuan Li, and Jiezhi Chen. Lateral charge migration induced abnormal read disturb in 3D charge-trapping NAND flash memory. *Applied Physics Express*, 13(5):054002, Apr. 2020.

[35] Yixin Luo, Saugata Ghose, Yu Cai, Erich F. Haratsch, and Onur Mutlu. Improving 3D NAND Flash Memory Lifetime by Tolerating Early Retention Loss and Process Variation. *Proc. ACM Meas. Anal. Comput. Syst.*, 2(3), Dec 2018.

[36] Qiao Li, Yu Chen, Guanyu Wu, Yajuan Du, Min Ye, Xinbiao Gan, Jie Zhang, Zhirong Shen, Jiwu Shu, and Chun Xue. Characterizing and Optimizing LDPC Performance on 3D NAND Flash Memories. *ACM Trans. Archit. Code Optim.*, 21(3), Sep 2024.

[37] Li Zhang, Shen-gang Hao, Jun Zheng, Yu-an Tan, Quan-xin Zhang, and Yuan-zhang Li. Descrambling data on solid-state disks by reverse-engineering the firmware. *Digit. Investig.*, 12(C):77–87, Mar 2015.

[38] Yun-Shan Hsieh, Bo-Jun Chen, Po-Chun Huang, and Yuan-Hao Chang. PRESS: Persistence Relaxation for Efficient and Secure Data Sanitization on Zoned Namespace Storage :

(Invited Paper). In *2024 29th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 341–348, 2024.

[39] Ping-Hsien Lin, Yu-Ming Chang, Yung-Chun Li, Wei-Chen Wang, Chien-Chung Ho, and Yuan-Hao Chang. Achieving fast sanitization with zero live data copy for MLC flash memory. In *Proceedings of the International Conference on Computer-Aided Design*, ICCAD '18, pages 1–8, New York, NY, USA, 2018. Association for Computing Machinery.

[40] Joshua Wise. Reverse Engineering a NAND Flash Device Management Algorithm. https://joshuawise.com/projects/ndfrecovery, Aug. 2014. [Accessed: Nov. 17, 2025].

[41] Aya Fukami, Saugata Ghose, Yixin Luo, Yu Cai, and Onur Mutlu. Improving the reliability of chip-off forensic analysis of NAND flash memory devices. *Digital Investigation*, 20:S1–S11, 2017.

[42] Lingjun Kong, Jiaying Wen, Guojun Han, Shengmei Zhao, Ming Jiang, and Chunming Zhao. Quantization and reliability-aware iterative majority-logic decoding algorithm for LDPC code in TLC NAND flash memory. In *2016 8th International Conference on Wireless Communications & Signal Processing (WCSP)*, pages 1–5, 2016.

[43] Shaoqi Yang, Meng Zhang, Xuepeng Zhan, Peng Guo, Xiaohuan Zhao, Guangkuo Yang, Xinyi Guo, Jixuan Wu, Fei Wu, and Jiezhi Chen. Retention Accelerated Testing for 3-D QLC nand Flash Memory: Characterization, Analysis, and Modeling. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 44(7):2779–2788, 2025.

[44] Jiaying Wen, Guojun Han, Xusheng Lin, and Yi Fang. Performance of the modified SRBI-MLGD algorithm for LDPC codes in MLC NAND flash memory. In *2015 10th International Conference on Information, Communications and Signal Processing (ICICS)*, pages 1–5, 2015.

[45] Debao Wei, Xiaoyu Chen, Hua Feng, Liyan Qiao, and Xiyuan Peng. A high-efficiency threshold voltage distribution test method based on the reliability of 3D NAND flash memory. *Microelectronics Reliability*, 114, 2020.

[46] Linxin Yin, Yingzhao Li, Xiaoyi Zhang, Xiongfei Zhai, and Guojun Han. An Efficient Dynamic Threshold Voltage Detection Scheme for Improving 3-D NAND Flash Reliability. *IEEE Transactions on Device and Materials Reliability*, 24(4):529–543, 2024.

[47] Yunjie Fan, Zhiqiang Wang, Shengwei Yang, Kun Han, and Yi He. Investigation of Retention Characteristics in a Triple-level Charge Trap 3D NAND Flash Memory. In *2022 IEEE International Reliability Physics Symposium (IRPS)*, pages P31–1–P31–4, 2022.

[48] Yu Cai, Erich F. Haratsch, Onur Mutlu, and Ken Mai. Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis, and modeling. In *2013 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1285–1290, 2013.

[49] Md Raquibuzzaman, Aleksandar Milenkovic, and Biswajit Ray. Intrablock Wear Leveling to Counter Layer-to-Layer Endurance Variation of 3-D NAND Flash Memory. *IEEE Transactions on Electron Devices*, 70(1):70–75, 2023.

[50] Matchima Buddhanoy, Aleksandar Milenkovic, Sudeep Pasricha, and Biswajit Ray. Page-Overwrite Data Sanitization in 3D NAND Flash: Challenges, Feasibility, and the PULSE Solution. *ACM Trans. Embed. Comput. Syst.*, 24(5s), Sep. 2025.

[51] Biswajit Ray, Shyam Raghunathan, and Sudeep Pasricha. Reliability, security, and sustainability challenges in 3-d nand flash ssds. *IEEE Design & Test*, 43(1):7–22, 2026.

[52] Matchima Buddhanoy, Kamil Khan, Aleksandar Milenkovic, Sudeep Pasricha, and Biswajit Ray. Improving Block Management in 3D NAND Flash SSDs with Sub-Block First Write Sequencing. In *Proceedings of the Great Lakes Symposium on VLSI 2024 (GLSVLSI'24)*, page 296–300, New York, NY, USA, 2024. Association for Computing Machinery.

[53] Mondol Anik Kumar and Biswajit Ray. Exploring cross-temperature reliability in 3d nand through layer-dependent bit error analysis. In *2024 IEEE 8th International Test Conference India (ITC India)*, pages 1–5, 2024.

[54] Christian Monzio Compagnoni and Alessandro S. Spinelli. Reliability of nand flash arrays: A review of what the 2-d–to–3-d transition meant. *IEEE Transactions on Electron Devices*, 66(11):4504–4516, 2019.

[55] Biswajit Ray and Mondol Anik Kumar. *Reliability Challenges of NAND Flash Memory in Harsh Environments*, chapter 3, pages 53–81. John Wiley & Sons, Ltd, 2026.

[56] Mondol Anik Kumar and Biswajit Ray. Cross-temperature reliability of 3d nand: Cell-to-cell variability analysis and countermeasure. In *2024 IEEE International Reliability Physics Symposium (IRPS)*, pages P25.MR–1–P25.MR–7, 2024.

[57] Rino Micheloni, Luca Crippa, and Alessia Marelli. *Inside NAND Flash Memories*. Springer Science & Business Media, Dordrecht, 2010.

[58] Shijun Liu and Xuecheng Zou. QLC NAND study and enhanced Gray coding methods for sixteen-level-based program algorithms. *Microelectron. J.*, 66(C):58–66, Aug. 2017.

[59] Carla Savage. A survey of combinatorial gray codes. *SIAM Rev. Soc. Ind. Appl. Math.*, 39(4):605–629, Jan. 1997.

[60] Chung-Meng Lee, Jimmy J. M. Tan, and Lih-Hsing Hsu. Embedding hamiltonian paths in hypercubes with a required vertex in a fixed position. *Inf. Process. Lett.*, 107(5):171–176, Aug. 2008.

[61] N. Zamani J. Maserjian. Behavior of the Si/SiO_2 interface observed by Fowler-Nordheim tunneling. *J. Appl. Phys.*, 53(559–567), 1982.

[62] Beomjun Kim and Myungsuk Kim. REO: Revisiting Erase Operation for improving lifetime and performance of modern NAND flash-based SSDs. *Electronics (Basel)*, 14(4):738, Feb. 2025.

[63] Chi-Weon Yoon. The Fundamentals of NAND Flash Memory: Technology for tomorrow's fourth industrial revolution. *IEEE Solid-State Circuits Magazine*, 14(2):56–65, 2022.

[64] Bojie Lou, Qihao Liu, Zhaogui Zeng, Yongwang Zhou, and Ji Zhong. A Review of Program disturb of 3D NAND Flash Memory. In *2023 24th International Conference on Electronic Packaging Technology (ICEPT)*, pages 1–5. IEEE, Aug. 2023.

[65] Kang-Deog Suh, Byung-Hoon Suh, Young-Ho Lim, Jin-Ki Kim, Young-Joon Choi, Yong-Nam Koh, Sung-Soo Lee, Suk-Chon Kwon, Byung-Soon Choi, Jin-Sun Yum, Jung-Hyuk Choi, Jang-Rae Kim, and Hyung-Kyu Lim. A 3.3 V 32 Mb NAND flash memory with incremental step pulse programming scheme. *IEEE Journal of Solid-State Circuits*, 30(11):1149–1156, 1995.

[66] Matchima Buddhanoy, Habib Ur Rahman, and Biswajit Ray. Rowhammer vulnerabilities in 3d nand flash: Attack characterization and mitigation strategies. In *2026 IEEE International Reliability Physics Symposium (IRPS)*, pages 1–6, 2026.

[67] Md Raquibuzzaman, Md Mehedi Hasan, Aleksandar Milenkovic, and Biswajit Ray. Layer-to-layer endurance variation of 3d nand flash memory. In *2022 IEEE International Reliability Physics Symposium (IRPS)*, pages 1–5, 2022.

[68] Jae-Duk Lee, Jeong-Hyuk Choi, Donggun Park, and Kinam Kim. Degradation of tunnel oxide by FN current stress and its effects on data retention characteristics of 90 nm NAND flash memory cells. In *2003 IEEE International Reliability Physics Symposium Proceedings, 2003. 41st Annual.*, pages 497–501, 2003.

[69] Kyunghwan Lee, Duckseoung Kang, Hyungcheol Shin, Sangjin Kwon, Shinhyung Kim, and Yuchul Hwang. Analysis of failure mechanisms in erased state of sub 20-nm nand flash

memory. In *2014 44th European Solid State Device Research Conference (ESSDERC)*, pages 58–61, 2014.

[70] Jounghun Park et al. Decomposition of Vertical and Lateral Charge Loss in Long-term Retention of 3-D NAND Flash Memory. In *2023 IEEE IRPS*, pages 1–4, 2023.

[71] Kyunghwan Lee et al. Analysis of Failure Mechanisms and Extraction of Activation Energies (E_a) in 21-nm nand Flash Cells. *IEEE Electron Device Letters*, 34(1):48–50, 2013.

[72] Jaeyeol Park et al. Modeling of Lateral Migration Mechanism of Holes in 3D NAND Flash Memory Charge Trap Layer during Retention Operation. In *2019 Silicon Nanoelectronics Workshop (SNW)*, pages 1–2, 2019.

[73] N.K. Zous et al. Lateral migration of trapped holes in a nitride storage flash memory cell and its qualification methodology. *IEEE Electron Device Letters*, 25(9):649–651, 2004.

[74] Matchima Buddhanoy and Biswajit Ray. Enhancing 3d nand flash endurance via thermal annealing: A step towards everlasting ssds. In *2025 Device Research Conference (DRC)*, pages 1–2, 2025.

[75] Mondol Anik Kumar, Matchima Buddhanoy, Justin Bell, Alexander Brandl, Indranil Chatterjee, and Biswajit Ray. Radiation induced leakage current in 3-d nand technology. *IEEE Transactions on Nuclear Science*, pages 1–1, 2026.

[76] Matchima Buddhanoy et al. Life-after-Death: Exploring Thermal Annealing Conditions to Enhance 3D NAND SSD Endurance. In *Proceedings of the 16th ACM Workshop on Hot Topics in Storage and File Systems (HotStorage'24)*, page 79–85, New York, NY, USA, 2024. Association for Computing Machinery.

[77] Blancco. What is NIST 800-88, and What Does "Media Sanitization" Really Mean?, May. 2023. [Accessed: Apr. 30, 2026].

[78] SEM Shred. Options for Physical Destruction of Storage Media. https://www.semshred.com/explore/insights/learning-library/options-for-physical-destruction-of-storage-media/, 2025. Accessed: Aug. 25, 2025.

[79] EE Times. Flash Media Data Destruction Guaranteed. https://www.eetimes.com/flash-media-data-destruction-guaranteed/, 2025. Accessed: Aug. 25, 2025.

[80] Lorenzo Zuolo, Cristian Zambelli, Rino Micheloni, and Piero Olivo. Solid-State Drives: Memory Driven Design Methodologies for Optimal Performance. *Proceedings of the IEEE*, 105(9):1589–1608, 2017.

[81] Feng Chen, Tong Zhang, and Xiaodong Zhang. Software Support Inside and Outside Solid-State Devices for High Performance and High Efficiency. *Proceedings of the IEEE*, 105(9):1650–1665, 2017.

[82] Ignacio Marco-Pérez, Beatriz Pérez, Angel Luis Rubio Garcia, and María A. Zapata. The Many Faces of Data Deletion: On the Significance and Implications of Deleting Data. *ACM Comput. Surv.*, 58(7), Jan. 2026.

[83] International Data Sanitization Consortium. Data Sanitization Terminology and Definitions. https://www.datasanitization.org/data-sanitization-terminology/, 2023. [Accessed: Apr. 30, 2026].

[84] Bo Chen and Radu Sion. HiFlash: A History Independent Flash Device. *CoRR*, abs/1511.05180, 2015.

[85] Jinhua Cui, Weiguang Liu, Jianhang Huang, and Laurence T. Yang. ADS: Leveraging Approximate Data for Efficient Data Sanitization in SSDs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(6):1771–1784, 2022.

[86] Md Mehedi Hasan and Biswajit Ray. Data recovery from "scrubbed" NAND flash storage: need for analog sanitization. In *Proceedings of the 29th USENIX Conference on Security Symposium*, SEC'20, USA, 2020. USENIX Association.

[87] Shijie Jia, Luning Xia, Bo Chen, and Peng Liu. NFPS: Adding Undetectable Secure Deletion to Flash Translation Layer. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '16, page 305–315, New York, NY, USA, 2016. Association for Computing Machinery.

[88] Bo Chen, Shijie Jia, Luning Xia, and Peng Liu. Sanitizing data is not enough! towards sanitizing structural artifacts in flash media. In *Proceedings of the 32nd Annual Conference on Computer Security Applications*, ACSAC '16, page 496–507, New York, NY, USA, 2016. Association for Computing Machinery.

[89] Open NAND Flash Interface ONFI. Discover the advantages of an ONFI world. https://www.onfi.org/, 2024. [Accessed: Apr. 30, 2026].

[90] Preeti Kumari. *Total ionizing dose effects on the state-of-the-art nand flash memories with an emphasis on dosimeter design*. PhD thesis, University of Alabama in Huntsville, 2021. [Accessed: Apr. 30, 2026].

[91] Future Technology Devices International Ltd. FT2232H Dual High Speed USB to Multipurpose UART/FIFO IC. https://ftdichip.com/wp-content/uploads/2024/09/DS_FT2232H.pdf. [Accessed: Apr. 30, 2026].

[92] Sprites mods. FT2232H NAND flash reader - Hardware. https://spritesmods.com/?art=ftdinand&page=2. [Accessed: Apr. 30, 2026].

[93] ONFI. Open NAND Flash Interface Specification. https://onfi.org/files/ONFI_5_2_Rev1.0.pdf, Feb. 2024. [Accessed: Apr. 30, 2026].

[94] Sparsh Mittal and Jeffrey S. Vetter. A Survey of Software Techniques for Using Non-Volatile Memories for Storage and Main Memory Systems. *IEEE Transactions on Parallel and Distributed Systems*, 27(5):1537–1550, 2016.

[95] Joel Reardon, Srdjan Capkun, and David Basin. Data Node Encrypted File System: Efficient Secure Deletion for Flash Memory. In *21st USENIX Security Symposium (USENIX Security 12)*, pages 333–348, Bellevue, WA, 2012. USENIX Association.

[96] Myungsuk Kim, Jisung Park, Genhee Cho, Yoona Kim, Lois Orosa, Onur Mutlu, and Jihong Kim. Evanesco: Architectural Support for Efficient Data Sanitization in Modern Flash-Based Storage Systems. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '20, page 1311–1326, New York, NY, USA, 2020. Association for Computing Machinery.

[97] Mondol Anik Kumar, Md Raquibuzzaman, Matchima Buddhanoy, Timothy Boykin, and Biswajit Ray. Origin of Post-Irradiation V_{th}-Shift Variability in 3-D NAND Memory Array. *IEEE Transactions on Nuclear Science*, 71(4):405–411, 2024.

[98] Rino Micheloni, A Marelli, and R Ravasio. *Error Correction Codes for Non-Volatile Memories*. Springer, New York, NY, 2008 edition, May 2008.

[99] Preeti Kumari, Umeshwarnath Surendranathan, Maryla Wasiolek, Khalid Hattar, Narayana P. Bhat, and Biswajit Ray. Radiation-Induced Error Mitigation by Read-Retry Technique for MLC 3-D NAND Flash Memory. *IEEE Transactions on Nuclear Science*, 68(5):1032–1039, 2021.

[100] Satoshi Inaba. 3D Flash Memory for Data-Intensive Applications. In *2018 IEEE International Memory Workshop (IMW)*, pages 1–4, 2018.

[101] Jianquan Jia, Lei Jin, Kaikai You, and Anyi Zhu. Investigation of the Connection Schemes between Decks in 3D NAND Flash. *Micromachines*, 14, 2023.

[102] Xinlei Jia, Lei Jin, Jianquan Jia, Kaikai You, Kaiwei Li, Shan Li, Yali Song, Yuanyuan Min, Ying Cui, Wenzhe Wei, Xiangnan Zhao, Weiming Chen, Hongtao Liu, An Zhang, and Zongliang Huo. A Novel Program Scheme to Optimize Program Disturbance in Dual-Deck 3D NAND Flash Memory. *IEEE Electron Device Letters*, 43(7):1033–1036, 2022.

[103] Qianhui Li et al. Optimal Read Voltages Decision Scheme Eliminating Read Retry Operations for 3D NAND Flash Memories. *Microelectronics Reliability*, 131:114509, 2022.

[104] Shinkeun Kim et al. Separation of Lateral Migration Components by Hole During the Short-Term Retention Operation in 3D NAND Flash Memories. *IEEE TED*, 67(6):2645–2647, 2020.

[105] Kyunghwan Lee, Myounggon Kang, Seongjun Seo, Duckseoung Kang, Shinhyung Kim, Dong Hua Li, and Hyungcheol Shin. Activation energies (e_a) of failure mechanisms in advanced nand flash cells for different generations and cycling. *IEEE Transactions on Electron Devices*, 60(3):1099–1107, 2013.

[106] Quan Xu. *Data reliability and error correction for NAND Flash Memory System*. PhD thesis, City University London, Aug. 2016.

[107] JP2016213838A. XOR-based scrambling and descrambling method and system for SSD communication protocol, Dec. 2016. Accessed: Nov. 17, 2025.

[108] Matchima Buddhanoy, Habib Ur Rahman, Aleksandar Milenkovic, Sudeep Pasricha, and Biswajit Ray. Descrambling the scrambler: Experimental extraction of data scrambling keys in cots nand flash. In *2026 27th International Symposium on Quality Electronic Design (ISQED)*, pages 1–8, 2026.

[109] Yu Cai, Yixin Luo, Erich F. Haratsch, Ken Mai, and Onur Mutlu. Data retention in MLC NAND flash memory: Characterization, optimization, and recovery. In *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, pages 551–563, 2015.

[110] Nikolaos Papandreou, Thomas Parnell, Haralampos Pozidis, Thomas Mittelholzer, Evangelos Eleftheriou, Charles Camp, Thomas Griffin, Gary Tressler, and Andrew Walls. Enhancing the Reliability of MLC NAND Flash Memory Systems by Read Channel Optimization. *ACM Trans. Des. Autom. Electron. Syst.*, 20(4), Sep. 2015.

[111] Habib Ur Rahman, Suresh Tharini, Sudeep Pasricha, and Biswajit Ray. Tcflash: In-flash bulk bitwise processing via dynamic sensing and tlc encoding in 3d nand. In *2025 IEEE 43rd International Conference on Computer Design (ICCD)*, pages 491–494, 2025.

[112] Shaoqi Li, Tianyu Wang, Yongbiao Zhu, Chenlin Ma, Yi Wang, Zhaoyan Shen, and Zili Shao. One Gray Code Fits All: Optimizing Access Time with Bi-Directional Programming for QLC SSDs. In *2025 Design, Automation & Test in Europe Conference (DATE)*, pages 1–2, 2025.

[113] Yina Lv, Liang Shi, Qiao Li, Congming Gao, Yunpeng Song, Longfei Luo, and Youtao Zhang. MGC: Multiple-Gray-Code for 3D NAND Flash based High-Density SSDs. In *2023 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pages 122–136, 2023.

[114] J. David Ingalls, Matthew J. Gadlage, Adam R. Duncan, Matthew J. Kay, Patrick L. Cole, and Ken K. Hunt. Implications of the Logical Decode on the Radiation Response of a Multi-Level Cell NAND Flash Memory. *IEEE Transactions on Nuclear Science*, 60(6):4451–4456, 2013.

[115] Prawar Poudel, Biswajit Ray, and Aleksandar Milenkovic. Microcontroller TRNGs Using Perturbed States of NOR Flash Memory Cells. *IEEE Transactions on Computers*, 68(2):307–313, 2019.