THESIS

EXPOSING DATA REMANENCE IN SANITIZED NOR FLASH: CHARACTERIZATION AND COUNTERMEASURES

Submitted by

Anjali Murali

Department of Electrical and Computer Engineering

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2025

Master's Committee:

Advisor: Biswajit Ray

Sudeep Pasricha
Jeremy Daily

ABSTRACT

EXPOSING DATA REMANENCE IN SANITIZED NOR FLASH:
CHARACTERIZATION AND COUNTERMEASURES

The rapid advancement of digital technology has led to the frequent decommissioning of electronic devices, many of which retain sensitive data in their non-volatile memory. NOR flash memory, widely used as a non-volatile storage medium in Internet of Things (IoT) devices, often stores proprietary firmware as well as sensitive user data collected by these systems. Although standard sanitization procedures are typically applied before disposing of such devices, residual data may persist in the storage medium, posing a significant risk of data leakage. In this work, we present a novel investigation into data remanence and recovery techniques in sanitized NOR flash memory chips. Our findings show that data can be partially or fully recovered from NOR flash arrays that have undergone an all-zero sanitization process. Given the widespread adoption of NOR flash in IoT devices, these results raise serious concerns regarding sensitive data leakage and the security of firmware updates in such systems.

ACKNOWLEDGEMENTS

I wish to extend my sincere gratitude to my advisor, Dr. Biswajit Ray, for his unwavering support and motivation, which were pivotal in the successful completion of this thesis. His expertise and comprehensive knowledge of flash memory devices significantly enhanced my understanding of the underlying principles and shaped the focus of this research. I am particularly appreciative of the opportunity to work under the guidance of Dr. Ray and for the collaborative environment fostered throughout the process. The patience demonstrated and the invaluable support provided during the experimental phases greatly contributed to the success of this work.

I would like to extend my sincere gratitude to the honorable committee members, Dr. Sudeep Pasricha and Dr. Jeremy Daily, for their time, thoughtful consideration, and constructive feedback throughout this process.

I am grateful to the Department of Electrical and Computer Engineering at Colorado State University for providing the academic environment, resources, and support essential for the execution of this research.

Additionally, I wish to acknowledge my colleagues at the Reliable and Assured Microelectronics (RAM) Lab for their support and collaboration throughout this academic endeavor.

I am deeply grateful to my family for their constant support and encouragement throughout my academic journey. I am especially thankful to my mother - Seethalakshmy CS, whose persistent efforts and dedication have played a vital role in shaping my future. I also thank my father - C. Muraleedharan, my grandmother - Chandravathy Amma, and my aunt - Santhi CS, for their prayers and heartfelt blessings. My sincere appreciation goes to my spouse - Harikrishnan CP, for his constant encouragement and motivation. Finally, I extend my deepest gratitude to all my close family members and friends who have supported me throughout this endeavor.

I'm thankful to God Almighty for providing me with the strength, opportunities, and perseverance necessary to achieve success.

This research was generously supported by the National Science Foundation under Grants 2403540 and 2317563. I gratefully acknowledge this support, which enabled the successful completion of data collection and analysis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

This chapter highlights two main motivations driving this research: the widespread use of NOR flash technology and the potential risks of data breaches linked to obsolete electronic devices that utilize NOR flash as a storage medium. Additionally, this chapter presents a comprehensive overview of embedded flash memory and introduces the two predominant types of flash technologies available in the market: NAND flash and NOR flash. We examine the fundamental architectural and functional differences between NAND and NOR flash, establishing the rationale for focusing this thesis on NOR flash memory. Finally, the chapter concludes with a structured outline of the thesis organization and scope, emphasizing its primary contributions.

1.1 Motivation

Flash memory is widely used in contemporary embedded systems for program and data storage, with NOR flash being a preferred choice for code storage due to its fast access times, support for random access, and superior reliability compared to NAND flash memory.

NAND flash dominates the market with a valuation of USD 65.1 billion in 2024 [1], due to the increasing demand of high-performance storage solutions in consumer electronics, and advancements in 3D NAND technology. In 2024, smartphones and tablets led the NAND flash market with a 31.4% share, driven by the demand for high-capacity storage in devices supporting high-resolution media, AI apps, and 5G-enabled real-time processing.

However, the recent growth of the Internet of Things (IoT) and automotive systems has increased the demand for NOR due to its fast random-access read speeds in demanding execute-in-place applications, high reliability, and functionality at extreme temperatures. It's been a staple in applications where code storage is crucial, such as in automotive, cloud compute, and industrial systems, as well as in applications that might undergo multiple updates, such as over-the-air (OTA) transmission, due to its high write endurance.

The NOR flash memory segment has a market size of $2.9 billion in 2024 and is expected to reach $3.82 billion by 2030 with a compound annual growth rate (CAGR) of 6.01% [2, 3]. This growth is expected to accelerate with the introduction of Macronix's new 3D NOR technology [4–6], which offers a single-chip 4Gb solution at lower costs, addressing the scaling limitations of traditional 2D NOR flash that stalled around the 40 nm node. By leveraging this architecture, end-system solution providers are able to reduce their need for multiple storage devices, such as eMMC and/or NAND. Moreover, 3D NOR flash offers shorter latency, thereby improving boot performance, which is an essential feature for applications that need near-instantaneous access to stored data.

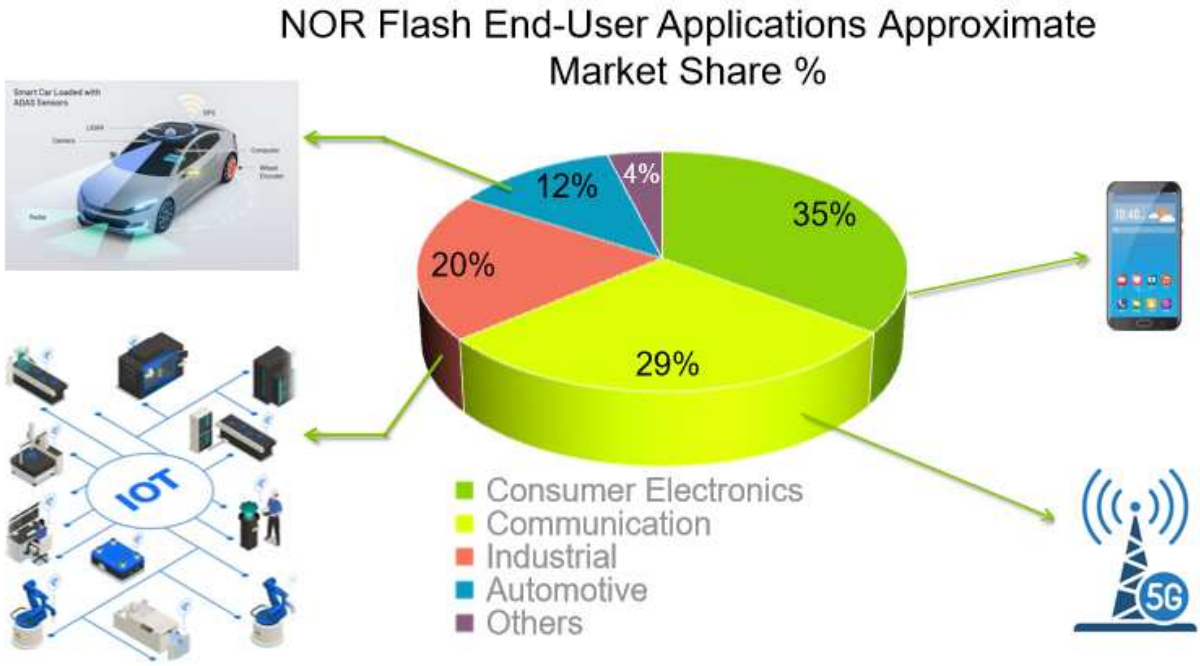


Figure 1.1: NOR flash market share based on end-user application

Figure 1.1 shows the market share of NOR flash in the year 2024 based on different end user applications such as consumer electronics, automotive, communication, industrial and others.

The increasing popularity and widespread adoption of NOR flash in modern embedded systems, particularly in control critical real time applications such as those found in the Internet

of Things (IoT) and automotive devices, highlights the importance of studying the security vulnerabilities associated with NOR flash memory. As these devices reach end-of-life, the risk of unauthorized data access from residual data in NOR flash memory becomes a significant concern.

The emerging popularity of NOR flash is the primary motivation for choosing NOR flash for the research presented in this thesis.

1.2 Non-Volatile Memories: NOR vs NAND

Memory devices are essential components of all electronic systems and constitute a significant share of the semiconductor market. Almost every electronic system requires some data to be retained even after power loss. This includes program code for microcontrollers, parameters for digital signal processors, and various system and user data such as security codes and configuration settings. Due to their wide applicability, non-volatile memories are widely used across industries including industrial, consumer, automotive, telecommunications, and computing peripherals. Thus, non-volatile memory forms an integral part of data storage in modern digital architectures. Even personal computers, which rely on magnetic hard drives for mass storage and RAM for working memory, use solid-state non-volatile memories for critical functions such as system boot. These memories come in a wide range of capacities and can be embedded directly into processor chips, enabling the development of complex system-on-chip (SoC) designs like smart cards. Before the introduction of flash memory, electrically programmable non-volatile memories consisted mainly of EPROMs and EEPROMs; EPROMs offered high density but required ultraviolet light for erasure and were primarily used during development or in low-volume production. Flash memories, offering electrical erase capabilities traditionally limited to costly EEPROMs, provide this functionality at costs and densities comparable to EPROMs. Consequently, flash memories have captured a significant portion of the market previously held by earlier memory types and expanded the scope of non-volatile memory applications. This expansion is closely tied to the rise of portable personal devices such as PDAs and mobile phones, which cannot use magnetic disks due to size and power constraints. Therefore, beyond the standard requirements for storing firmware and sys-

tem parameters, there is a growing demand for mass storage of operating systems, applications, and user data to be met by semiconductor memories [7].

Flash memories are non-volatile memories which are widely used for storing firmware, configuration data, and user-defined settings in embedded systems. These memories operate using two fundamental mechanisms: programming and erase. Programming involves writing a logic '0' by injecting charge into the storage element of a memory cell, while erasure restores the logic '1' by removing this charge. A key architectural characteristic of flash memory is the asymmetry between these operations—programming can target individual cells selectively, whereas erasure must be performed on entire sectors or blocks. This constraint influences memory array design, allowing compact and cost-efficient layouts. Depending on how memory cells are organized and connected, flash technologies are categorized as NAND or NOR flash, each with distinct characteristics that influence their suitability for specific embedded applications [8]. These differences are examined in the following section.

1.2.1 NOR Flash Technology

NOR flash is the mainstream technology for applications that require code and parameter storage, particularly in embedded systems where fast and random memory access is essential. It was first introduced in the late 1980s as a replacement of EPROM (Erasable Programmable Read Only Memory) technology [9], which initially required external programming tools and bulk erase processes. By the mid-1990s, the second generation of NOR Flash was developed, incorporating important advancements such as on-chip charge pumps for single-supply operations, sector-based erasure, and embedded erase algorithms that improved in-system reprogramming efficiency. The rapid expansion of mobile phone technology contributed to the evolution of a third generation, which was designed to optimize low power consumption, flexible voltage management, high-speed read operations, and read-while-write functionality [10]. These innovations have enabled the seamless integration of both code and data storage within a single chip. Presently, NOR Flash supports a wide variety of specifications concerning density, voltage, interface width, and memory

architecture, establishing it as a versatile and reliable solution for numerous embedded applications. Therefore, NOR flash is the mainstream technology for applications that require code and parameter storage, particularly in embedded systems where fast and random memory access is essential.

The "NOR" flash name originates from the array configuration of its memory cells, which resembles the structure of a NOR logic gate. In this architecture, cells are organized in a grid of rows and columns. Cells that share a common control gate are connected along a horizontal line known as the wordline (WL), while those that share a common drain electrode, where one contact is shared between two adjacent cells, form a vertical connection called the bitline (BL). The source electrode, in contrast, is typically shared across all cells in the array, establishing a uniform grounding scheme [11].

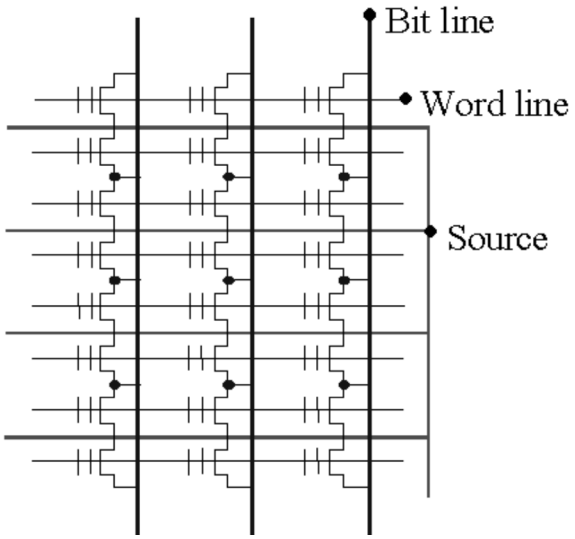


Figure 1.2: NOR Flash array equivalent circuit [11]

This structural organization is illustrated in Figure 1.2. In other words, each cell in NOR flash memory is connected in parallel to the bitline and shares a common source, which allows individual cells to be directly accessed by selecting specific wordlines and bitlines. This configuration enables true random access in NOR flash memory. As a result, individual cells can be addressed

independently, making NOR memory well suited for applications that require fast and frequent random reads, such as code execution. Additionally, the direct access to the memory cell makes this architecture ideal for high speed and noise immunity. Recent developments in NOR flash technology have made it highly suitable for multilevel storage, enabling higher memory densities at lower costs. High-density NOR flash devices capable of storing two bits per cell are now commercially available, offering 30–40% cost savings compared to traditional single-bit-per-cell alternatives at the same technology node. Moreover, the compatibility of NOR flash with advanced logic processes has made it a preferred choice for embedded memory in System-on-Chip (SoC) designs [8].

1.2.2 NAND Flash Technology

NAND flash technology is a leading choice for nonvolatile storage of high-volume data. Since its introduction in 1987 [12], the technology has undergone continuous scaling, enabling significant improvements in storage density and cost efficiency. Today, NAND flash offers terabyte-level capacities, making it the dominant form of flash memory for a wide range of applications [13–15]. Though NAND flash offers only serial access, its higher storage density and lower cost compared to NOR flash make it a dominant technology for data storage applications and memory cards.

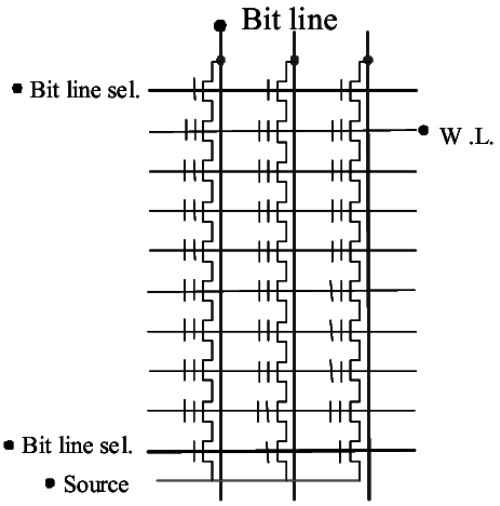


Figure 1.3: NAND Flash array equivalent circuit [7]

In NAND flash memory, multiple cells (typically 16 or 32) are connected in series between the bitline and ground, as shown in Figure 1.3. This configuration improves storage density compared to NOR flash, which requires more routing resources, but comes at the cost of speed. Data must be read through several intermediate cells, resulting in longer access times in the microsecond range, which is significantly slower than the nanosecond-scale access times of NOR flash. This serial access structure limits NAND flash to sequential operations and makes it unsuitable for random-access applications. Additionally, the read-through mechanism increases sensitivity to noise and data patterns, making multi-level cell implementation more challenging. As a result, most NAND flash products still operate in single-bit-per-cell mode.

1.2.3 NOR vs. NAND Architectures

A comparison of NOR and NAND flash based on its key characteristics is given in Table 1.1. Output size indicates how many bits can be transferred to the output at once, read and write size refer to how many bits can be read or programmed at a time and read access time is the time required to complete a read operation before data is available at the output [8].

Table 1.1: Comparison of NOR and NAND flash memory characteristics

	NOR	**NAND**
Memory size	$\leq$512 Mbit	1–8 Gbit
Sector size	$\sim$1 Mbit	$\sim$1 Mbit
Output size	Byte/Word/32-bits	Byte/Word
Read size	8–16 Words	2 KBytes
Write size	8–16 Words	2 KBytes
Read access time	$<$80 ns	20 μs
Program time	9 μs/Word	400 μs/Page
Erase time	1 s/Sector	1 ms/Sector

These differences significantly influence their applications, with NOR flash being a mainstream technology for suitable for fast code execution and random access, while NAND flash offers higher storage density and lower cost per bit [7, 8, 16].

1.3 Embedded Flash Memory

In this section, we define embedded flash memory and explain its role in embedded system design for modern applications. An embedded system is defined as "a computer system with a dedicated function within a larger mechanical or electrical system, often with real-time computing constraints, embedded as part of a complete device often including hardware and mechanical parts" [17]. They achieve superior cost and performance efficiency by utilizing minimal hardware resources, typically through one-chip microcontroller units (MCUs). These systems are now ubiquitous, appearing in consumer electronics, automobiles, medical devices, and industrial control systems. As technology advances, embedded systems are becoming more important for creating smart and connected devices such as those in the Internet of Things or IoT. The factors of core technology that support these advancements are security, safety, communication, signal sensing and processing, and low power to be embedded in real-time edge computing schemes under large-scale system designs [17]. Among the foundational elements of embedded systems, embedded memory is one of the most critical technologies, as it directly influences the overall cost, performance, and power efficiency of a system. Embedded memory refers to memory that is integrated within the chip and does not have an externally exposed interface. This design enables compact and power-efficient architectures, tailored for specific applications requiring real-time performance and minimal hardware overhead.

Embedded flash memory (eFlash) offers a combination of programmability, non-volatility, and on-chip integration, making it particularly well-suited for embedded system design. These properties contribute to supply-chain flexibility, improved power management, and compact system architectures. eFlash has been instrumental in advancing embedded applications such as automotive engine control and smart-card technologies, where its ability to support reprogrammable and secure code and data storage has been critical [8, 17–19].

Embedded flash-memory technology originally inherited much from stand-alone flash-memory devices such as NOR flash memory. It has diverged from mainstream stand-alone technologies such as NAND flash due to embedded-specific requirements, leading to the development of a

distinct technological path [18]. Standardization of specifications and dedicated innovations have enabled eFlash to support a wide range of applications with optimized performance, reliability, and cost. Since the 1990s, eFlash has driven rapid market growth by offering a high cost-to-value advantage. Its adoption in microcontroller units (MCUs) has enabled widespread integration into embedded applications, including automotive and smart-card systems, due to its compactness, low power consumption, and high reliability.

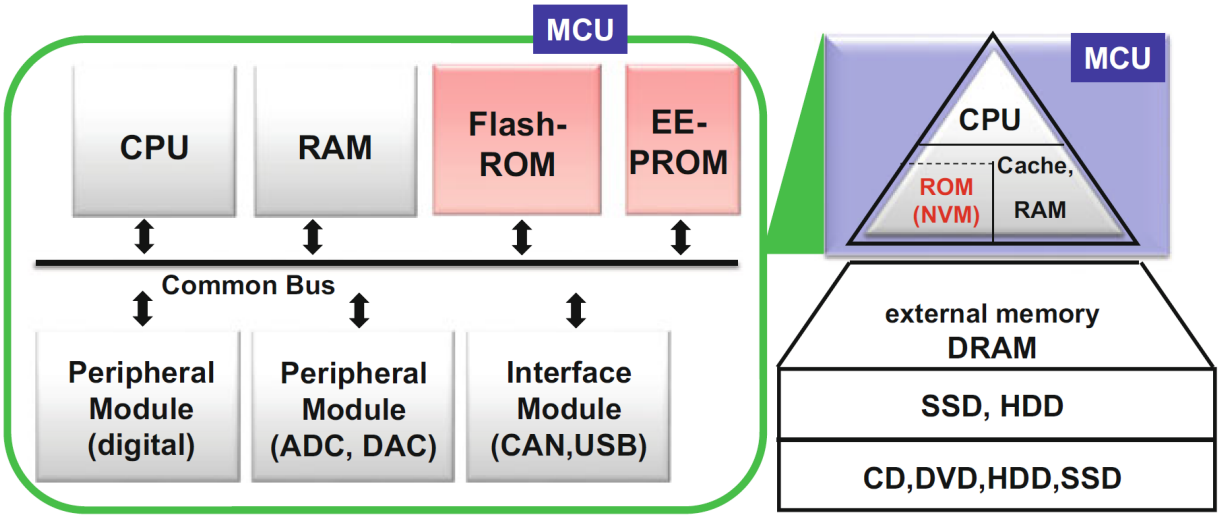


Figure 1.4: Non-volatile memory in MCU architecture [17]

Flash-based MCUs have become a cornerstone of modern embedded systems, owing to their ability to integrate program code and data storage within a single chip. This one-chip configuration, which utilizes embedded flash (eFlash) memory, offers a compact and efficient alternative to traditional designs that rely on external EEPROM for data retention. As illustrated in Figure 1.4, typical flash-MCU architectures integrate code and data ROM within on-chip flash memory, resulting in a standard and efficient configuration for small embedded systems. This architectural unification has facilitated the development of applications requiring multi-time programmability, particularly in real-time control scenarios, while simultaneously offering cost benefits. Consequently, flash-based MCUs have become the preferred solution across a wide range of high-volume markets.

Among these, automotive control systems and smart card technologies have emerged as key segments, significantly contributing to the overall growth and importance of flash-MCU solutions in the embedded systems landscape [18].

Integrating embedded flash memory into MCU products has significantly transformed embedded system design and supply chains. Unlike mask ROM-based MCUs that require lengthy development cycles due to fixed programming, flash-MCUs enable shorter turn-around times by allowing software to be programmed after system assembly, thus supporting parallel hardware and software development. This flexibility also simplifies production and inventory management, as a single hardware platform can be customized through firmware updates at later stages. Moreover, flash-MCUs facilitate hardware-software separation and co-design, enabling advanced features such as software-controlled power management for energy-efficient operation. However, this shift has also introduced challenges, as increasingly complex system requirements have extended software development timelines, marking a paradigm shift towards a more sophisticated, software-centric embedded design approaches.

In summary, eFlash has become an essential component in modern embedded systems, offering a compact, energy-efficient, and reprogrammable storage solution. Its widespread adoption in applications such as automotive control and secure identification systems highlights its importance in environments demanding reliability and performance. As embedded devices continue to evolve, the challenges associated with managing the data stored in their non-volatile memory also increase. This thesis builds on that context by focusing on NOR flash memory and examining its data retention behavior in obsolete embedded systems, with the aim of uncovering potential security vulnerabilities due to remanent data.

1.4 Thesis Outline

The thesis work presented here focuses on data recovery from a sanitized charge trap based NOR flash. The previous sections of this chapter described the history, applications, and significance of embedded flash memory in the modern world, highlighting its evolution into a critical

technology that underpins the flexibility, performance, and efficiency of today's embedded systems. This section presents an overview of the research conducted in this thesis, outlining its objectives, scope, and key contributions.

The Chapter 2 the fundamentals of NOR flash is discussed, which includes the different types of NOR flash memory based on the memory cell structure and number of transistors used. We use NOR flash memory of two different cell structures to validate our results. However, the primary evaluations are performed on a MirrorBit™ flash and its principles are explained. The flash memory operations such as read, program and erase are covered in details with reference to the underlying device physics. Moreover, distinctions in these mechanisms with respect to each type of memory cells are illustrated with the help of figures.

The experimental setup, including hardware and firmware development, is elaborated in Chapter 3. This chapter outlines the part numbers and features of the various devices used in the experimental procedures, along with detailed descriptions of the interfacing techniques employed to control and verify flash memory operations. Additionally, timing information extracted from these devices is characterized and compared against datasheet specifications to validate performance and gain deeper insights into their endurance and architectural behavior. These analyses form the foundation for the key contributions presented in this thesis.

The key contributions of this thesis are presented in Chapter 4 and Chapter 5. Chapter 4 focuses on reverse engineering the sector layout of MirrorBit™ NOR flash to uncover structural and architectural details that are not explicitly documented in datasheets or publicly available sources. Additionally, Chapter 5 presents an in-depth investigation into data remanence in sanitized NOR flash memory. This work demonstrates that sensitive information, such as proprietary firmware and user data, can be partially or fully recovered even after standard sanitization procedures are applied. These findings highlight critical vulnerabilities in existing data removal practices and propose countermeasures to enhance the reliability and security of NOR flash-based storage, particularly in IoT devices where data confidentiality is paramount.

Chapter 6 concludes the thesis by summarizing the key findings and contributions, and offers recommendations for potential improvements and expansion of this work in the field of flash memory security.

Chapter 2

Background on NOR Flash

In this chapter, the structure and operational principles of NOR flash memory are reviewed. The NOR flash memory cells are reviewed in terms of their cell structures, operational principles, and key characteristics, with a focus on performance and reliability aspects.

2.1 Flash memory cell

A flash memory cell is basically a floating gate MOS transistor, in which the gate is electrically isolated and fully enclosed by insulating materials, as seen from Figure 2.2a. The floating gate stores charge and is capacitively controlled by a separate control gate. This structure allows the threshold voltage of the transistor to be modulated based on the charge present in the floating gate, enabling data storage. The insulating layer between the floating gate and the channel, commonly referred to as the tunnel oxide, typically measures around 9 to 10 nanometers and facilitates Fowler Nordheim electron tunneling during programming and erasure. Above the floating gate, the inter-poly dielectric separating it from the control gate is composed of a triple layer of oxide, nitride, and oxide, with an effective thickness of 15 to 20 nanometers. This oxide nitride oxide stack improves the quality of the tunnel oxide by avoiding the need for high temperature oxidation on poly-silicon, which can degrade the underlying thin oxide. If both dielectric layers behave ideally, the energy band diagram of the cell reveals that the floating gate serves as a potential well, with the tunnel and oxide nitride oxide layers acting as barriers. The cell represents a logical one when the floating gate is neutral or positively charged, and a logical zero when electrons are stored in the floating gate.

Figure 2.1 shows the broad classification of NOR flash memory. The flash memory cells are categorized into two types in terms of structure of the storage: floating-gate structures (FG) and charge-trapping structures (CT) [11, 17, 20]. The flash cells are further classified based on the

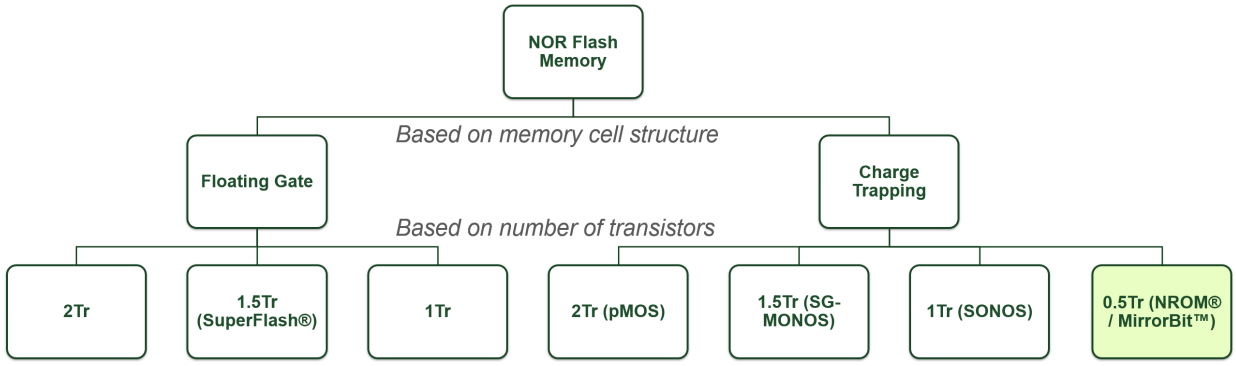


Figure 2.1: Classification of NOR Flash

number of transistors in a cell. This research is based on the highlighted category of NOR flash - NROM®/MirrorBit™.

2.1.1 Floating-Gate (FG) Flash Technology

An FG type flash is further classified into three categories based on the number of transistors as shown in Figure 2.2. The 1 transistor floating gate (1 Tr FG) cell [19] is a simple NOR-type cell designed for high density applications, where a stacked-gate transistor functions both as a selection gate and for data storage. In contrast, the 1.5 transistor floating gate (1.5 Tr FG) cell [21], also known as the split-gate FG cell [22], and the 2 transistor floating gate (2 Tr FG) cell [23] consist of two transistors connected in series, with one transistor acting as the selector and the other serving as the data storage element [24]. Although these cells occupy a larger area compared to the 1 Tr FG cell, they offer superior performance and lower power consumption [19, 25].

Figure 2.3 illustrates the read operation mechanism in NOR flash memory, using a conventional single transistor (1 Tr) floating gate (FG) cell as an example. The figure depicts the current-voltage (IV) characteristics and channel formation behavior under programmed and erased states, which correspond to distinct charge storage conditions. During a read operation, data is accessed by applying an appropriate voltage to the control gate (CG). In the programmed state, excess electrons are trapped in the FG, raising the threshold voltage (V_{th}) and inhibiting the formation of a conductive channel between the source and drain. Alternatively, the FG contains few or no electrons in the erased state, resulting in a lower V_{th} that permits channel formation and allows

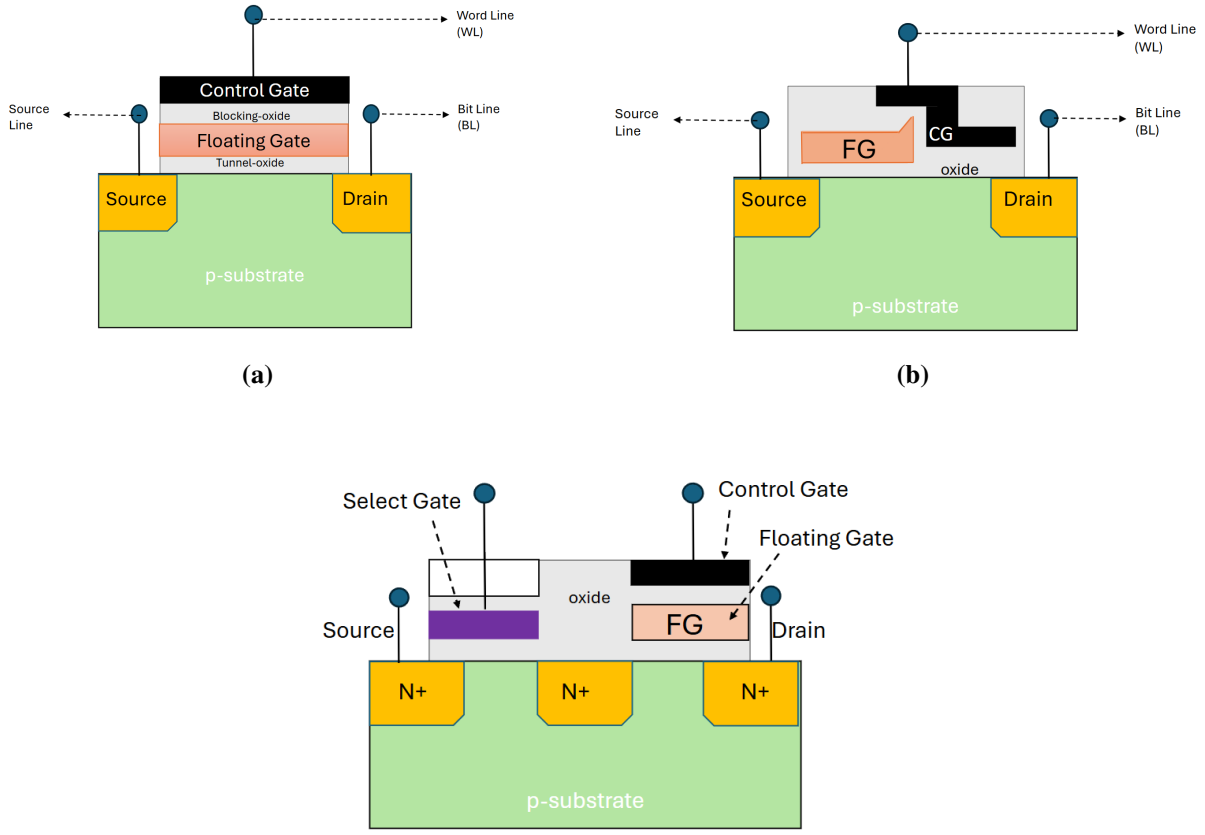


Figure 2.2: Floating Gate Cell Structure (a) 1 Tr Cell (b) Split-gate cell (c) 2 Tr Cell

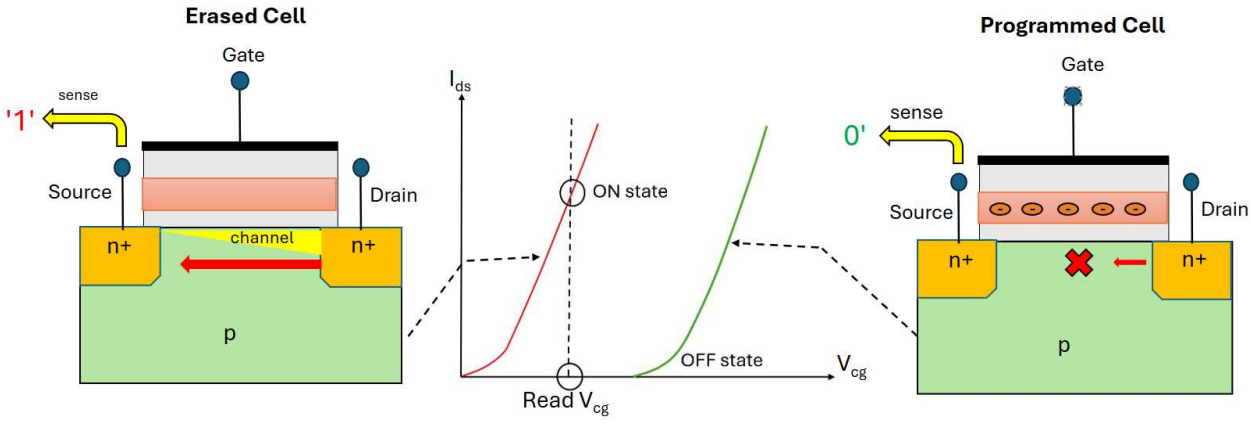


Figure 2.3: Read-operation mechanism for FG-type cells

current to flow, corresponding to the ON state. The sense amplifier then registers this condition as a logical '1'. This contrast in electrical behavior underlies the binary data representation scheme employed in NOR flash memory.

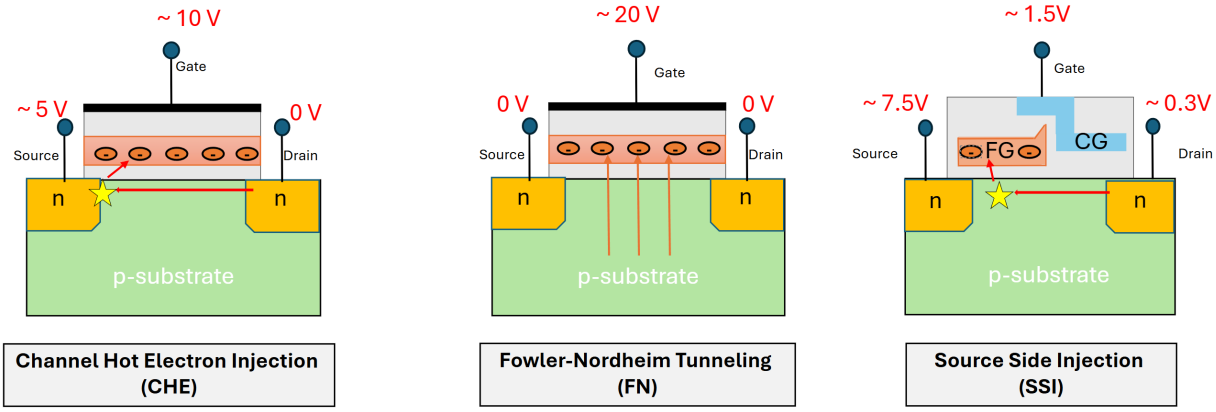


Figure 2.4: Programming mechanisms for FG-type cells

Programming and erasure in flash memory cells involve the transfer of electrons into or out of the floating gate through insulating dielectric layers. The physical mechanisms governing these operations vary depending on the cell architecture and the electrical conditions applied. The major electron injection mechanisms used in flash memory are summarized in Figure 2.4:

- Channel Hot Electron (CHE) injection [26, 27] occurs when high voltages are applied to the control gate and source. This creates a strong lateral electric field near the source region, which accelerates electrons toward the drain. A simultaneous vertical electric field facilitates the injection of these energetic electrons into the floating gate. Although CHE injection supports fast programming, it typically requires relatively high program current. This programming mechanism is used in NROM® or MirrorBit™, which we termed as 0.5 Tr cell, 1Tr FG type cells and 2Tr CT type cells (pMOS) [17, 28].

- Fowler Nordheim (FN) tunneling [29] is based on quantum mechanical tunneling of electrons through a thin oxide layer. A strong vertical electric field, induced by a high control

gate voltage, enables electrons to tunnel from the substrate into the floating gate. Compared to CHE injection, this method consumes significantly less current but demands a higher voltage and longer programming time. A 2Tr FG type cell and a 1Tr CT type cell (SONOS) utilizes this mechanism for programming [17].

- Source Side Injection (SSI) [30,31] is employed primarily in split gate or 1.5 transistor cells. In this technique, a high source voltage combined with a moderate control gate voltage generates strong electric fields in the region between the floating gate and the control gate. This results in efficient generation of hot electrons that are then injected into the floating gate. SSI achieves lower program current while maintaining programming efficiency. This mechanism is used by the 1.5Tr cells in both FG and CT type flash, namely SuperFlash™ [32, 33] and SG-MONOS [34] respectively.

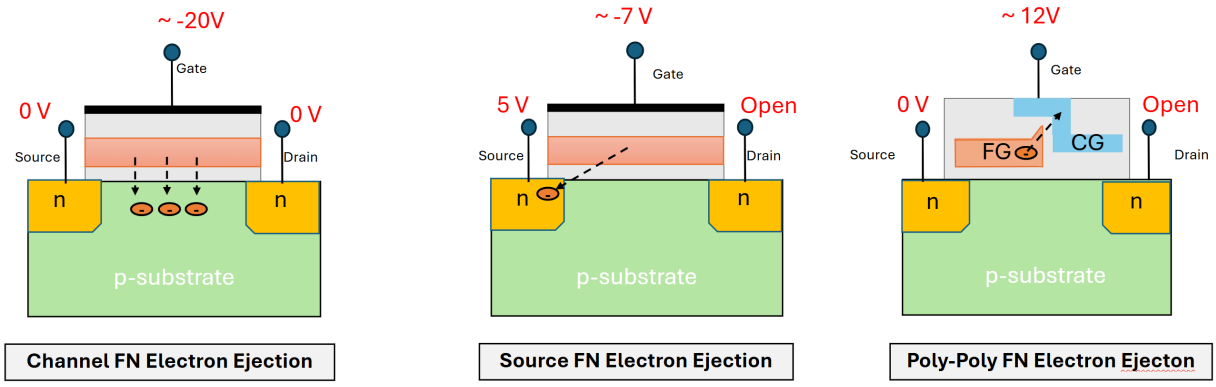


Figure 2.5: Erase mechanisms for FG-type cells

In flash memory technologies, the erase operation is executed either by ejection of electrons from the floating gate or the injection of holes into the floating gate to neutralize stored electrons. However, hole injection is generally avoided in floating gate based cells due to the risk of creating defects in the insulating layer. These defects can severely degrade data retention performance. In

contrast, charge trap based cells, which exhibit greater resilience to dielectric damage, often utilize hole injection for erase operations, as will be discussed later in this section.

Figure 2.5 summarizes the primary Fowler Nordheim based electron ejection mechanisms used in floating gate memory cells:

- **Channel Fowler Nordheim Electron Ejection**: In this method, a negative high voltage is applied to the control gate and/or a positive high voltage is applied to the p-type well. These voltages create a strong electric field across the tunnel oxide, allowing electrons to tunnel from the floating gate to the channel [35]. Although this mechanism requires only a small erase current, it demands high voltages. Additionally, the memory array's well regions must be segmented and biased independently to support the positive high voltage, which increases the area overhead. This erase mechanism is adopted by 1Tr and 2Tr FG type cells [19, 23].

- **Source Fowler Nordheim Electron Ejection**: By applying a positive high voltage to the source line, electrons can be ejected from the floating gate to the source through Fowler Nordheim tunneling. However, this requires a deep source junction to suppress leakage current, which in turn limits the scalability of the memory cell [17].

- **Poly to poly Fowler Nordheim Electron Ejection**: In this mechanism, electrons are ejected from the floating gate polysilicon layer to the control gate polysilicon layer. This is achieved by applying a positive high voltage to the control gate. The electric field enhancement effect, often achieved by a needle-shaped structure at the corner of the floating gate, helps reduce the required erase voltage. An advantage of this method is that it eliminates the need for a negative voltage during the erase operation. The 1.5Tr FG type cell utilizes this mechanism for erase operation [21, 32].

2.1.2 Charge-Trapping (CT) Flash Technology

Charge trap memory cells represent an alternative embedded flash memory structure alongside conventional floating gate cells. These cells use thin silicon nitride films or silicon nanodots as

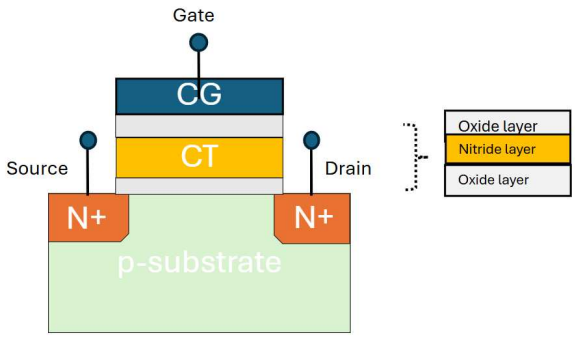


Figure 2.6: Structure of a 1Tr CT cell

the storage medium and offer significant advantages in terms of reliability and compatibility with standard logic Complementary Metal-Oxide-Semiconductor (CMOS) processes. Figure 2.6 shows the structure of a CT cell.

In floating gate memory cells, charges are stored uniformly within the conductive floating gate. Though this structural uniformity is efficient for charge modulation, presents a critical vulnerability since the presence of a single conductive defect in the surrounding oxide can result in the complete loss of stored charge. However, charge trap memory cells store electrons in localized trap sites within the silicon nitride film or at the interface between the nitride and oxide layers. As these trap sites are spatially separated, a defect typically leads to only partial charge loss, preserving the integrity of the remaining data. This distributed charge storage mechanism grants charge trap cells inherently greater reliability than their floating gate counterparts.

Furthermore, charge trap cells benefit from a significantly reduced vertical profile due to their thin film structure. This enables them to be fabricated at the same gate height as standard logic transistors, even in advanced technology nodes. As a result, charge trap cells exhibit superior scalability and better integration potential with logic CMOS technologies, making them highly suitable for use in modern embedded systems.

In CT type cells memory cells, a nitride film or nano-crystals are used for data storage instead of a poly-silicon floating gate. They are also categorized into four types according to the number of transistors, as described in Table 2.1. The program operation mechanism of CT-type cells is

Table 2.1: Variations of charge-trapping type NOR flash memory

Cell Structure	Program	Erase	Advantage
0.5 Tr cell (NROM) [28]	CHE	BTBT	Two bits per cell, high density
1 Tr cell (SONOS) [36]	FN	FN	High density
1.5 Tr cell (SG-MONOS) [34]	SSI	BTBT	Fast program, Low power read
2 Tr cell (pMOS) [37]	CHE	FN	Low power P/E

almost the same as that for FG-type cells. However, there is a significant difference in the erase operation between both FG and CT type cells. Erased FG-type cells store no or fewer electrons in the floating gate, whereas erased CT-type cells store holes in the charge pockets in nitride layer, as will be explained in detail later in this section.

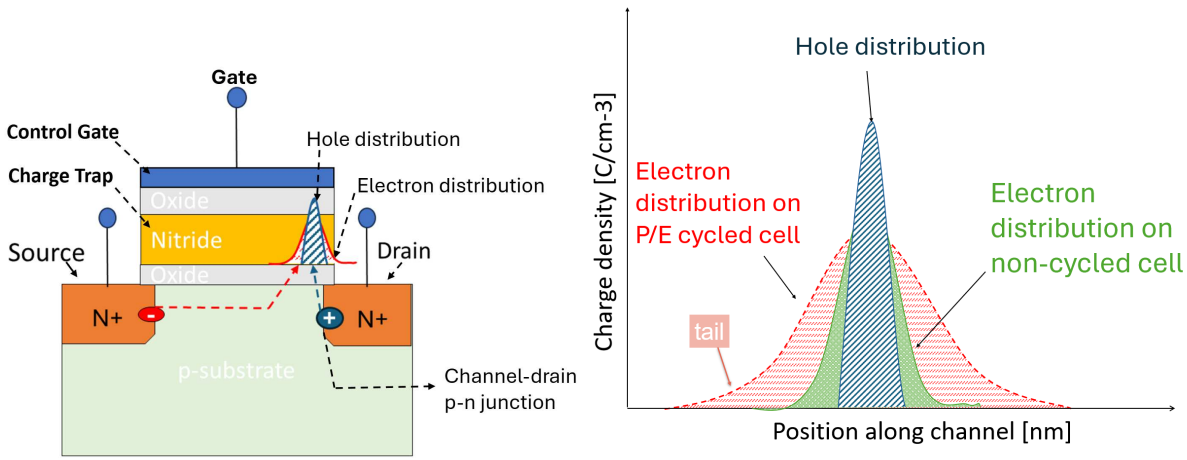


Figure 2.7: Mismatch of CT-type cell charge at the channel-drain p-n junction (a) Charge trapping type flash cell (b) distribution of electrons formed at the channel during program operation for a fresh and cycled conditions, and holes generated at the drain side p-n junction during erase operation [38, 39]

The program-operation mechanism in charge trap (CT) type memory cells generally resembles that of floating gate (FG) type cells, employing either Fowler–Nordheim (FN) tunneling or Channel Hot-Electron (CHE) injection. However, an important difference arises from the nature of the charge storage medium. Because, the charge-trap layer in CT-type cells is an insulator (typically

silicon nitride), while the floating gate in FG-type cells is a conductor. Therefore, the injected electrons are localized near the injection site in CT-type cells, rather than being uniformly distributed as in FG-type cells [39]. When programming is performed using channel FN injection, electrons are trapped across the full channel area. Conversely, with CHE injection, trapping occurs predominantly near the source where the electric field is strongest, and not near the drain where the field is weak. This spatial dependence of charge trapping also applies to hole injection during erase operations. Therefore, the mismatch in the spatial distribution of electrons and holes, as shown in Figure 2.7, must be carefully considered in the design and operation of CT-type memory cells to ensure reliable data storage and retrieval [17, 28, 40].

Unlike the programming mechanism, erase mechanism in CT type memory cells differs significantly from that in FG type cells. This is primarily due to insulator property of the charge trap storage medium. CT type memory cells utilize hole injection for erase operations, owing to their robustness against insulator defects. Two main mechanisms are employed for erase operation:

- **Channel Electron Ejection and Hole Injection**: A negative bias at the control gate or a positive bias at the substrate induces electron ejection and subsequent hole injection across the channel. Although the erase current is low, relatively high voltages are required. Initially, trapped electrons are removed from the trap layer, followed by hole injection from the p-well. This results in a threshold voltage (Vt) lower than the neutral Vt due to trapped holes. Electron ejection occurs through Fowler–Nordheim (FN) tunneling either from the trap layer to the conduction band or from the silicon nitride (SiN) trap sites to the substrate. Hole injection occurs via direct tunneling, facilitated by the ultra-thin (2nm) bottom oxide in CT-type cells, in contrast to the thicker (9nm) oxide in FG-type cells.

- **Band-to-Band Tunneling (BTBT) Hot-Hole Injection**: A high positive voltage at the source and a negative voltage at the metal gate generate hole-electron pairs through band to band tunneling [41]. This mechanism allows faster erase operations at lower voltages, but with increased current per cell. Hole injection is localized near the source junction, and to address this limited distribution, CHE injection or SSI programming is often used

in conjunction with BTBT. The 0.5 Tr CT cell NROM®/MirrorBit™ and 1.5 Tr CT cell SuperFlash™ adopt this mechanism for erase [17, 28].

2.2 MirrorBit™ Technology

The Spansion MirrorBit™ device (originally developed as Saifun NROM) [42, 43] is a unique charge-trap-based non-volatile memory technology that utilizes two separate, physical, and narrowly localized charge storage regions situated near each of the two junction edges of the cell. These localized charge pockets leverage the insulating properties of the charge-trap layer to enable independent bit storage within a single cell. Therefore, a MirrorBit™ cell offers higher density, as it can store two bits per cell. Due to its virtual-ground architecture, MirrorBit™ technology is relatively immune to gate-to-drain shorts and single-bit charge loss or gain, which are commonly found in floating-gate (FG) NOR flash technologies [44]. Therefore, this technology is ideal for the the code flash segment and meets the highest reliability requirements in the flash memory market, making it well suited for applications such as the automotive industry. In addition, the non-conductive nature of its storage material enables reliable operation in high-radiation environments.

2.2.1 Cell Structure

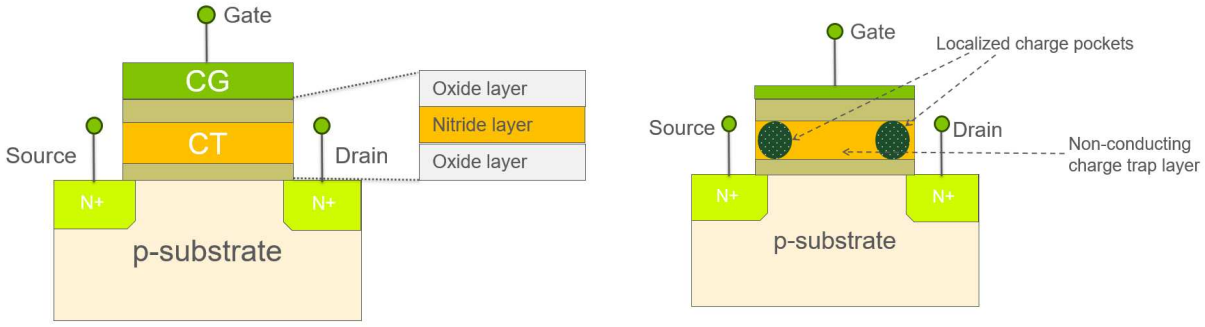
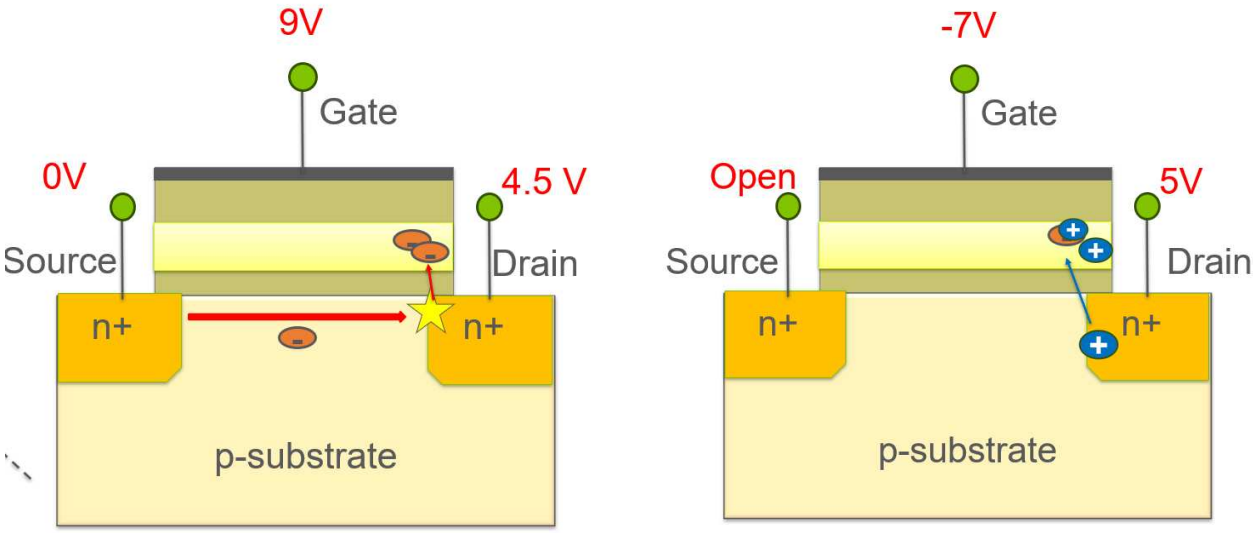


Figure 2.8: (a) Charge trapping type flash cell with ONO stack storage medium (b) Localized charge traps in MirrorBit™ cell

The cell structure of the MirrorBit™ device is provided in Figure 2.8. The storage medium consists of a silicon-nitride layer sandwiched between two silicon dioxide layers called the oxide-nitride-oxide stack (ONO) [45], as can be seen in Figure 2.8a. in other words, the memory cell structure can be described as an N-channel MOS transistor with dielectric ONO stack as gate insulator. The typical equivalent oxide thickness ranges from 15 to 20 nm, with the box oxide thickness between 3.5 and 7 nm, the nitride layer measuring 3 to 6 nm, and the tunnel oxide thickness varying from 8 to 12 nm. The polygate serves as the word line, while the buried n^+ diffusion acts as the bit line [46].

2.2.2 Operations

The device utilizes Channel Hot Electron Injection (CHE) for programming and Band to Band Tunneling Hot Hole Injection [41] for erase.Both injections produce narrow trapped charge regions in the nitride above junction edge. The read operation is performed by sensing the bit located at source side, hiding the opposite bit at drain side and vice versa.



(a) **(b)**

Figure 2.9: (a) Programming mechanism - CHE (b) Erase mechanism - BTBT Hot Hole Injection

To program the localized charge pocket above the drain side junction edge, a high positive voltage is applied at the gate terminal and a medium positive voltage is applied at the drain side as shown in Figure 2.9a. The positive voltage at drain forms a channel due to the horizontal electric field and high positive voltage at gate forms a vertical electrical field. The electrons from the channel accelerates towards drain due to horizontal field and gets injected to the charge pocket due to the vertical field, producing narrow trapped charge regions in the nitride above junction edge. Thus the drain side is programmed and the source side can be programmed alternatively. To erase the device, band-to-band tunneling (BTBT) hot hole injection is employed, as illustrated in Figure Figure 2.9b. A positive voltage is applied to the terminal to be erased (e.g., the drain), while a high negative voltage is applied to the gate. This creates conditions for holes to be injected from the n^+-doped drain into the localized charge pocket, where they recombine with stored electrons and neutralize the charge, thereby achieving the erase operation.

Two conditions are necessary to read a bit without interference from the adjacent bit. First, the charge distribution must be sufficiently narrow, and second, the drain-to-source voltage, V_{DS}, must be sufficiently high during the read operation. These conditions enable high sensitivity to the charge trapped near the source side, where the source voltage V_S is 0 V, effectively concealing the trapped charge near the drain, where the voltage V_D ranges from 1.3 to 1.7 V. Although the trapped charge near the drain creates a high local threshold voltage, V_T, in that region, the elevated drain voltage reduces this barrier, allowing for accurate reading. This operation can be reversed to read the charge near the drain side.

2.3 Conclusion

In this chapter, we presented an overview of two categories of flash memory based on cell structure, floating-gate (FG) memory cells and charge-trap (CT) memory cells. Each type is further classified according to the number of transistors per cell. This thesis employs FG 1Tr cells, the 1.5Tr FG cell known as SuperFlash™, and the 0.5Tr CT cell referred to as NROM®/MirrorBit™

for experimental evaluation. The programming and erase mechanisms of each flash memory type are thoroughly analyzed, and their structural distinctions are also discussed in detail.

Chapter 3

Experimental Setup and Characterization

This chapter elaborates on the experimental framework for interfacing NOR flash memory with a computer, including the implementation and evaluation of flash memory operations such as read, write, and erase, as well as memory characterization. The parallel NOR flash memory chipsets shown in Table 3.1 are interfaced with Arduino Due to perform different relevant operations for the project. Different memory cell technologies and structures are chosen for a comprehensive evaluation of the NOR flash reliability and performance. Each device supports parallel interfacing with the host system, complies with the Joint Electron Device Engineering Council (JEDEC) standards, Common Flash Interface (CFI) specification, and includes at least eight sectors (blocks for SST39VF1601C) of size 64 KB or 32 Kwords.

Table 3.1: Details of the NOR flash memory devices used in the experimental evaluation

Part Number	Manufacturer	Cell Structure	Technology
S29GL064S70TFI040	Infineon	Charge Trap	MirrorBit™
SST39VF1601C	Microchip	Floating Gate	SuperFlash®
MX29GL640EBTI	Macronix	Floating Gate	Conventional
M29F800FT	Micron	Floating Gate	Conventional

The S29GL064S70TFI040 [47] device is the primary platform for most experimental evaluations. It is fabricated using Spansion's proprietary 65-nm MirrorBit™ process technology [48]. This flash memory part number was arbitrarily chosen initially, and the remaining devices were subsequently chosen to match key characteristics such as matching sector size, varied process technologies, consistent TSOP-48 packaging, and adherence to a common command interface standard.

The hardware framework developed to support custom firmware implementation and ensure reliable flash memory operation is discussed in the next section. The subsequent sections present

the firmware architecture that enables flash memory access, data analysis, and system-level debugging. This is followed by a detailed validation of flash memory operations, which forms the foundation for the experimental evaluations conducted in this work. The chapter concludes with a comprehensive analysis of timing characteristics, including measurements of sector erase and word program durations, as well as the impact of data patterns and iteration count on operation latency.

3.1 Hardware Setup

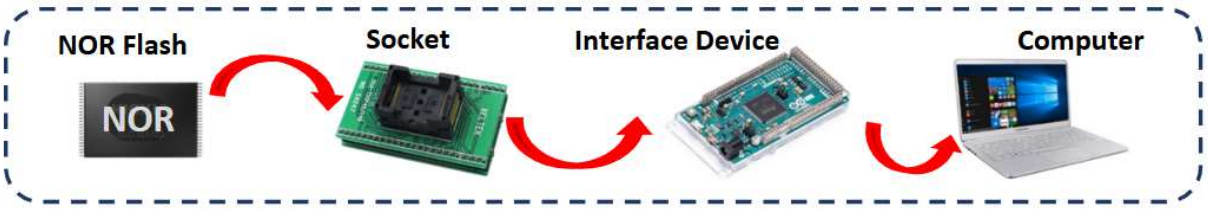


Figure 3.1: Overview of the experimental setup for interfacing the NOR flash memory

The hardware setup consists of the components shown in Figure 3.1. This is the overall setup for interfacing NOR flash memory with a computer. The flash chips are mounted on a TSOP-48 to Dual Inline Package (DIP) socket adapter, which maps the pins in the chip to the corresponding inputs on an Arduino Due board. The Arduino is connected to a host computer via USB, and flash operations such as read, write, and erase are executed using custom firmware developed with the Arduino programming environment. Additionally, a logic analyzer is connected to the setup to capture the digital control signals to and from the flash chip for debugging and timing verification.

A wired setup for establishing connections between hardware components is useful during the initial stages for validating pin mappings and verifying functional operation. However, such a setup is inherently cluttered, especially due to the parallel interfacing which involves wired connections to and from the 48 pin leads, as shown in the actual hardware setup picture in Figure 3.2. Moreover, the wired setup is prone to electrical noise and signal instability. Additionally, it is difficult to transport this setup since any physical disturbances can lead to intermittent disconnections.

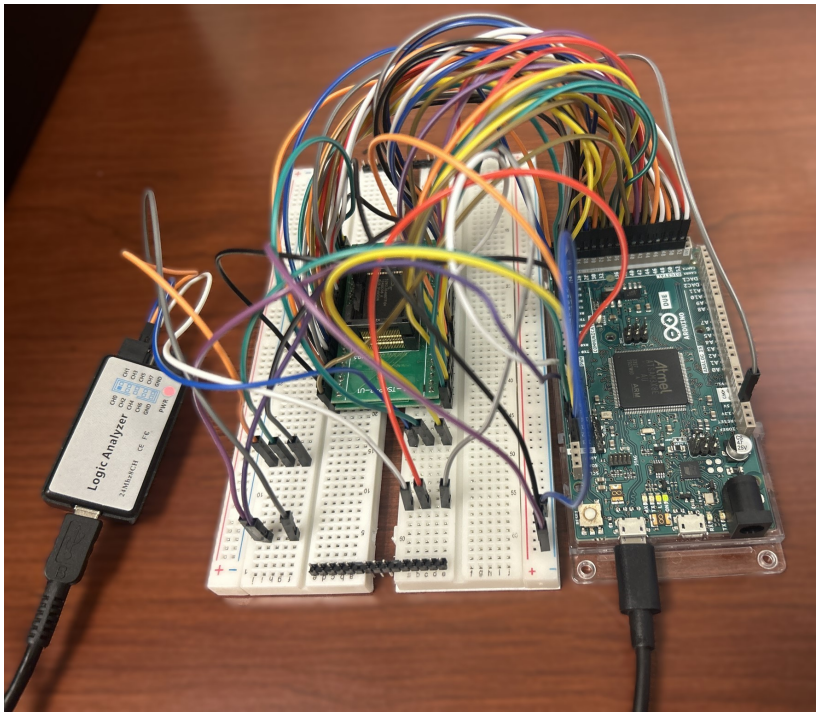


Figure 3.2: Wired setup for NOR flash interfacing

Over time, increased contact resistance may further degrade the signal integrity, resulting in erratic behavior. To ensure accurate and consistent evaluation, a more reliable and robust solution was necessary. Therefore, a custom Printed Circuit Board (PCB) was designed to provide a stable and durable hardware interface.

3.1.1 Custom PCB Design

A Printed Circuit Board (PCB) is a laminated sandwich structure of conductive copper and insulating substrate materials [49]. The copper layers are etched with specific patterns to form traces and planes that enable electrical interconnections between various components. This configuration serves the function of traditional wiring but in a more compact, reliable, and manufacturable form. Components are typically soldered onto designated copper pads on the board, establishing both mechanical support and electrical connectivity.

The custom printed circuit board (PCB) was designed using the PCB Design Software *Autodesk Eagle*. Figure 3.3 shows the steps involved in the PCB design process and are summarized below:

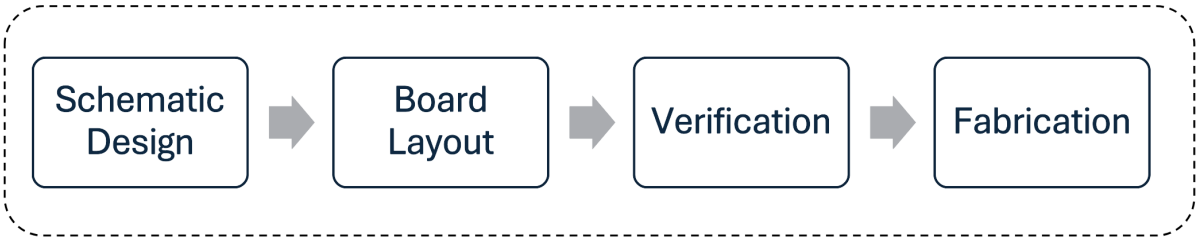


Figure 3.3: Steps involved in the PCB design process

- Schematic Design: It involves adding the electronic components, connecting the components and adding labels and values to them.
- Board Layout: Placing and arranging the components in the physical representation of the board, adding routing traces and ground plane.
- Verification: Electrical Rule Check (ERC) and Design Rule Check (DRC)
- Fabrication: Generate Gerber files to be sent to a manufacturer for PCB fabrication

The first step in designing a PCB is creating a schematic of the required board. A schematic is a diagram that shows the electronic components and their connections in the circuit [50]. The schematic design of the custom board for NOR flash memory is included in Figure 3.4. It includes the electronic components Arduino Due, DIP48 to TSOP48 socket, and a 5-pin male connector for the control pin monitoring. The pin headers from the TSOP-48 have to be mapped to the Arduino board GPIO headers according to the defined functionality, along with the 3.3V power and GND signals. An additional pin headers are provided to probe the control pins in the flash memory such as Chip Enable (CE#), Write Enable (WE#), OutputEnable (OE#), and Ready / Busy (RY/BY#) to enable debugging using a Logic Analyzer.

A board layout is a physical representation of the PCB, showing the placement of components and the routing of traces [50]. The components are arranged the way it is required on the physical board. The traces are routed following the guidelines to minimize interferences and guaranteeing

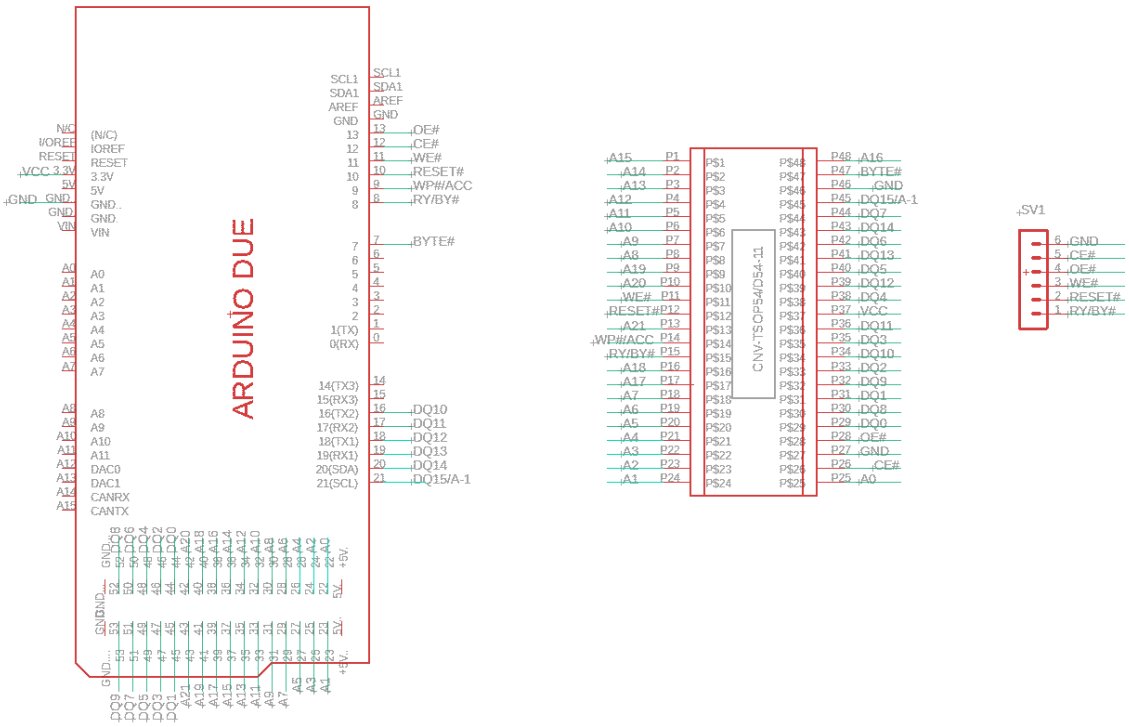


Figure 3.4: Schematic design for the NOR flash memory and Arduino interfacing board

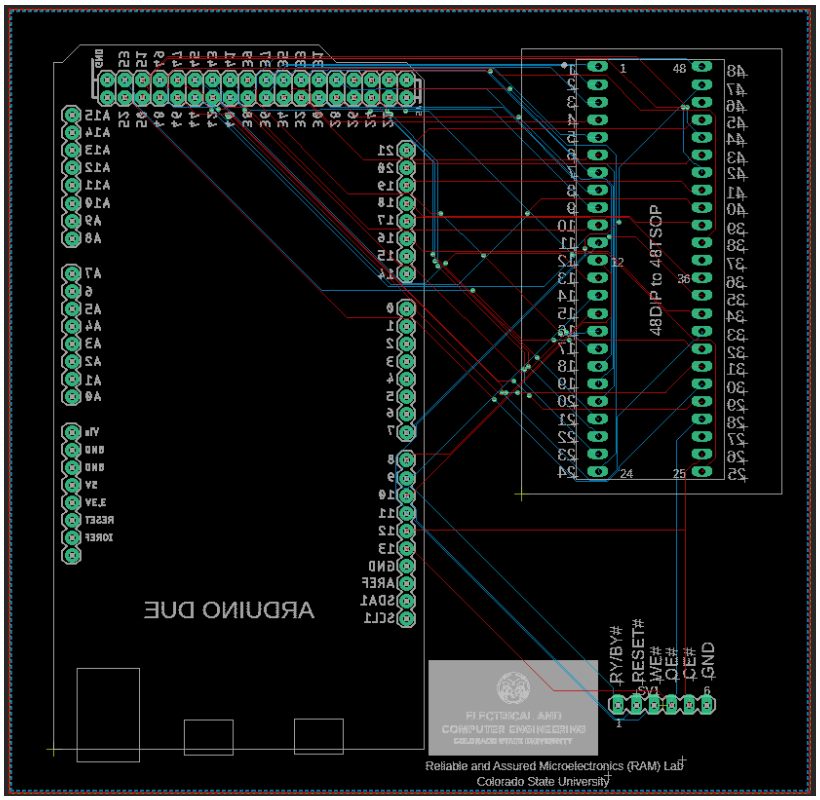


Figure 3.5: Layout design for the NOR flash memory and Arduino interfacing board

signal integrity, as shown in Figure 3.5. Here, the traces are routed in the top and bottom layer which can be seen as red and blue traces respectively.

When designing the PCB, it is important to follow design rules and constraints to ensure that the board will function correctly and is manufacturable [50]. Some common design rules and constraints include:

- Minimum trace width and spacing
- Minimum drill size and spacing
- Minimum clearance between components and traces
- Maximum number of layers

To validate if the board design meets these rules and constraints, the AutoDesk Eagle software offers Electrical Rule Check (ERC) and Design Rule Check (DRC) tools. The manufacturer website provides a set of design rules depending on the number of layers n the board, which can be imported to the software. The PCB designed for this project is a two layer board, and is manufactured through PCBWay. Therefore, the DRC test for a 2 layer board specified by this manufacturer was imported.

Once both the ERC and DRC tests are completed without any errors, the Gerber files can be generated for fabrication of the board. Gerber files are the industry standard for PCB manufacturing and contain all the necessary information to fabricate the board [50]. These files are provided to the manufacturer for PCB fabrication.

Finally, the Arduino Due development board, the DIP48 to TSOP48 socket adapter, and the required pin headers were carefully assembled and soldered onto the fabricated PCB. This step marks the completion of the hardware integration phase, enabling direct interfacing between the microcontroller and the NOR flash memory device through the custom-designed PCB. The soldered components establish reliable electrical connections and ensure mechanical stability necessary for consistent operation during experiments.

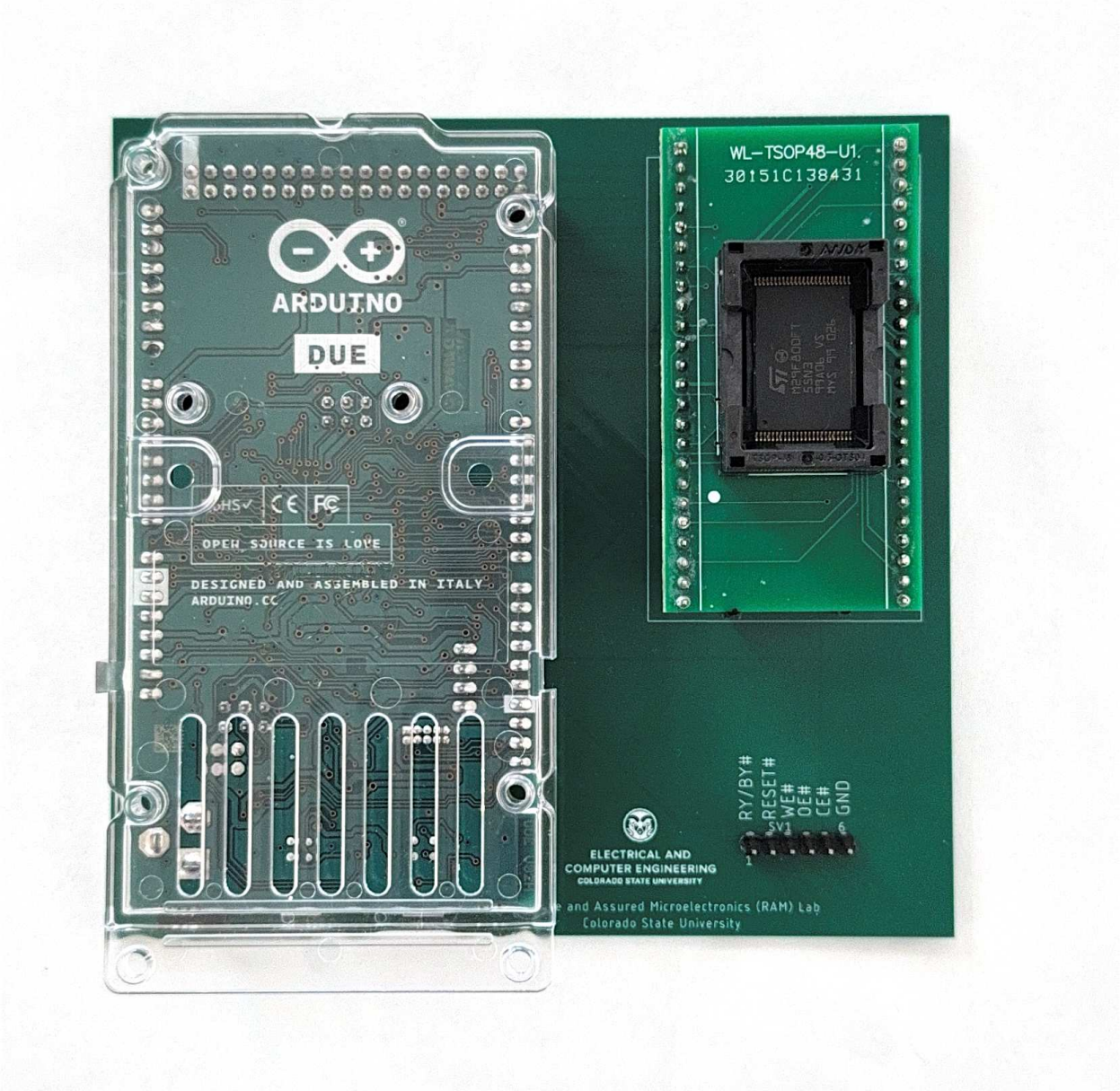


Figure 3.6: The actual Printed Circuit board (PCB) designed and assembled for NOR flash interfacing

Figure 3.6 presents the actual assembled PCB setup, which was used for conducting the experimental evaluations and validating the methodologies and findings presented in this thesis. The layout was designed to minimize signal integrity issues and to support accurate, repeatable testing of flash memory behavior under various conditions.

3.2 Firmware Design

Firmware is a class of software that provides low-level control for a device's specific hardware. It is typically embedded directly into hardware components and operates as a bridge between hardware and higher-level software, managing functions like device initialization, hardware abstraction, and communication protocols [51]. The firmware design is a vital part of the project, enabling communication between the development platform and the flash memory. It facilitates access and control of flash operations such as erase, program, and read. Firmware development is implemented using the Arduino development environment for hardware control, while Python is used separately for data analysis, and visualization.

The control signal status, timing requirements, and address-data sequences to the command register required for executing various operations are obtained from the datasheet of each flash memory devices [47, 52–54].

The correct logic levels on the control signals determine the device bus operations. The different control pins in the flash memory device are as follows:

- Chip Enable (CE#): This pin has to be driven low for all the operations. A logic '0' on this pin selects the chip for its operations.

- Write Enable (WE#): To write to the device bus, this pin has to be driven low. A low on this pin enables writing into the flash memory which includes program and erase operation. During this time the flash memory doesn't output any data. Therefore, this pin has to kept at logic '1' during read operation.

- Output Enable (OE#): A logic '0' on this pin enables reading from the device bus. Hence, this pin has to be driven low for reading data from the device. This pin must have a logic high during program and erase operations.

- Ready / Busy# (RY/BY#): This is a status monitoring pin because the logic level on this pin indicates whether the device is actively erase / programming or not. Unlike other control pins, this is an output signal from the device. A logic low on this pin indicates the device is busy and logic high means the device ready for next operation.

- Hardware Reset (RESET#): A logic low on this pin resets the flash device and it terminates all ongoing operations. Therefore, it has to be kept at high impedance state for normal operations. This pin is used for implementing hard reset of the device, which is significant for the partial erase and partial program implementation, discussed later in this chapter.

Table 3.2: Logic levels required on the control signals to perform device bus operations - read, write and erase

Operation	**CE#**	**WE#**	**OE#**	**RESET#**
Read	LOW	HIGH	LOW	HIGH
Program	LOW	LOW	HIGH	HIGH
Erase	LOW	LOW	HIGH	HIGH

A summary of the control pin logic levels for different flash memory operations is provided in Table 3.2. All control pins are active low.

The CE# pin must be driven low first, followed by either WE# or OE# depending on the operation. The RY/BY# pin goes down during an ongoing programming or erase operation and the device doesn't respond to any commands other than reset or program/erase suspend at this time. Therefore, a valid command can be sent only after waiting till the RY/BY# pin goes high.

The address bits (Amax - A0) are populated with a valid address prior to enabling the chip. The address gets latched at the falling edge of the CE# pin. This is followed by writing the valid data on the data pins (DQ15 - DQ0) if it is a write operation. In case of read operation, the data

appears in the data line at this stage. The CE#, WE# and OE# are driven back to logic high after the operation is completed.

Apart from the control pin configurations, the type of operation to be performed is dependent on a sequence of values on the address and data lines. These sequences are discussed next in this chapter.

3.2.1 Device Identification

The device provides identification codes which are unique to that particular class of devices on sending a certain sequence of commands to the command register. This predefined codes are useful for validating the memory interfacing and also to identify the type of device that is being interfaced.

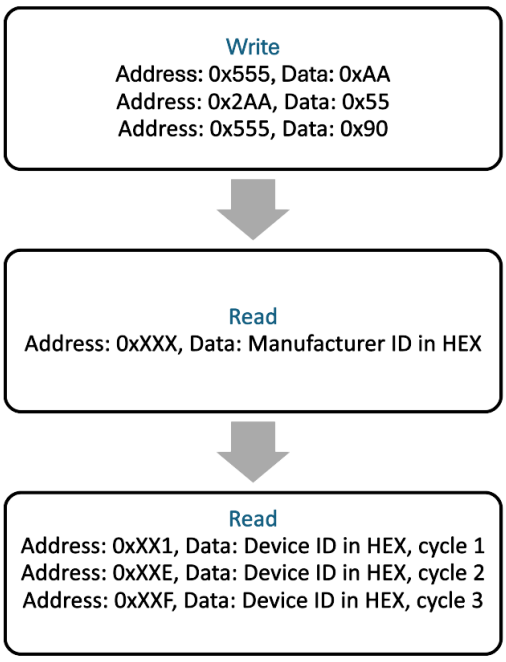


Figure 3.7: Command sequences required to read the manufacturer and device ID from the flash device devices

The work flow provided in Figure 3.7 shows the command sequences required to read the manufacturer and device ID from the flash device devices. The devices S29GL064S70TFI040 and MX29GL640EBTI give device ID in 3 cycles, while the remaining provides in a single cycle. The flowchart represents the device ID output in 3 cycles.

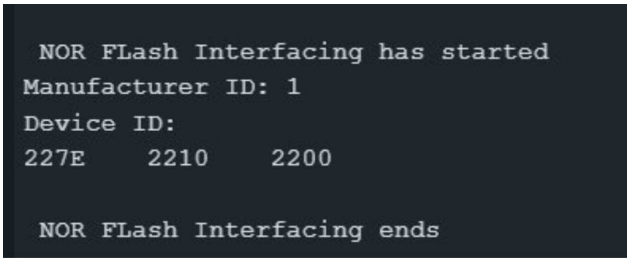
The datasheet of each device specifies the Manufacturer and Device ID for them. If the values read from the device matches with those specified in the respective datasheet, the interfacing is validated. In Figure 3.8, the output obtained for each devices are shown, which validates the flash access.

3.2.2 Program and Erase Operations

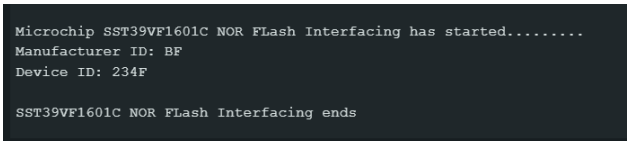
The NOR flash memory supports word-level programming within a sector. Programming a single word typically involves a four-cycle command sequence. The first two cycles consist of unlock write commands, followed by a program setup command in the third cycle. The final cycle specifies the target address and the data to be programmed. Sector programming is achieved by iteratively performing the word program operation across the entire sector, which comprises 32 K words.

The erase operation in NOR flash memory is supported only at a sector and chip level. Word or page level granularity is not available for erase operations. This is because the erase operation is done by applying a high voltage at the gate to remove charge from the floating gate / charge trap, as discussed in Chapter 2.

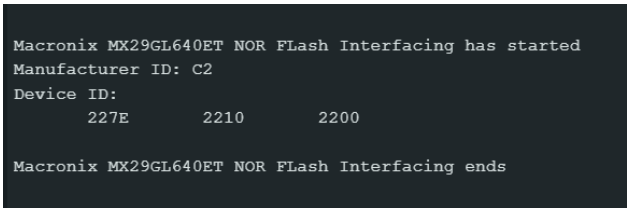
A complete sector erase operation takes an average of 250 ms in *S29GL064S70TFI040* device, whereas it takes only 17 ms in *SST39VF1601C*. The erase duration can be found either from the firmware or by capturing the logic analyser timing information during erase. Similarly, the word program time for *S29GL064S70TFI040* is an average of 200 µs, whereas for *SST39VF1601C* it is as low as 5 µs. This variation is due to the difference in the memory cell structure of these two devices. *S29GL064S70TFI040* is fabricated using Spansion's proprietary MirrorBit™ technology, which stores two bits per cell by using charge-trap architecture, whereas *SST39VF1601C* employs



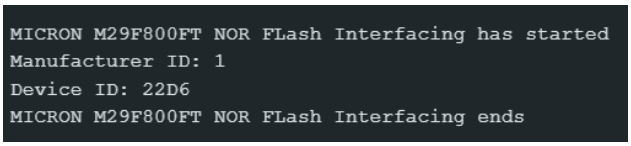
(a)



(b)



(c)



(d)

Figure 3.8: The Manufacturer ID and Device ID read in HEX format from different flash memory devices to validate interfacing (a) Infineon S29GL064S70TFI040 (b) Microchip SST39VF1601C (c) Macronix MX29GL640EBTI (d) Micron M29F800FT

Microchip's SuperFlash® technology based on a split-gate cell design for enhanced endurance and performance.

3.2.3 Partial Program and Partial Erase

The premature termination of an erase and program operation is referred to as a *partial erase* and *partial program*, respectively. The termination is done by triggering a hardware reset, which is done by writing a logic LOW to the RESET# pin. All operations are terminated at this time and the device resets. Partial erase is an important step involved in the data recovery performed in Chapter 5.

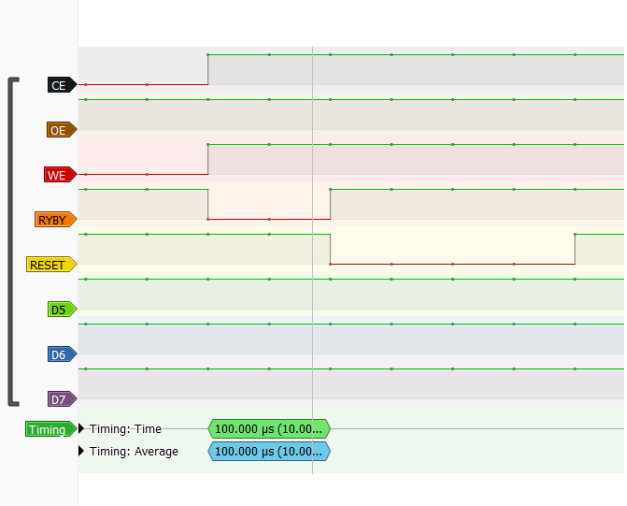


Figure 3.9: The actual captured timing information of partial erase implementation

The timing information shown in Figure 3.9 is a record of the partial erase performed on the NOR flash device. Here, the RY/BY# goes down after the rising edge of CE# and WE#, indicating an active erase operation. After a delay of 100 µ s, the RESET# pin is driven LOW, which terminates the ongoing erase operation and resets the device, leaving all control pins in high-impedance state. The delay after which hardware reset is activated is called the *partial program time* tpp or *partial erase time* tpe.

Implementation and a detailed analysis of both the partial program and the partial erase were carried out on the *Infineon S29GL064S70TFI040* device. A fresh sector in a new device was chosen for analysis. The partial erase and partial program experiments are explored in detail in Chapter 5.

3.2.4 Data Logging and Analysis

The flash device control and operations are performed in Arduino development environment, as mentioned previously. Data generated from these operations are transmitted through the serial communication port (COM) for logging and subsequent analysis. The timing information can also be exported as a text file from a logic analyzer, which is discussed in the next section. Python scripts are developed to analyze various aspects of flash memory behavior in this thesis project, including data integrity, image reconstruction, statistical correlations, and performance metrics.

- **Image Processing:** This involves generating a binary image and converting it into hexadecimal format to program in flash memory. The hexadecimal data read from the flash are subsequently reconstructed into a two-dimensional matrix or image format for visualization and analysis.

- **Mathematical Analysis:** Quantitative metrics such as bit accuracy and correlation coefficient are calculated to assess the integrity and consistency of flash memory operations.

- **Data Visualization:** Various plots relevant to the evaluation of flash memory behavior are generated using Python's *matplotlib* library. These include scatter plots, correlation plots, and histograms, which provide comprehensive insight into data distribution and behavior. This serves as the foundation for the Timing Characterization

Data transmitted from the Arduino is logged into a text file using a serial terminal application, with CoolTerm employed for this purpose. The recorded data is subsequently analyzed using Python scripts developed and executed within the PyCharm Integrated Development Environment (IDE).

3.3 Timing Characterization

Timing characterization involves measuring and associating the latency of flash memory operations with specific events to identify patterns and understand system behavior. This study was performed on the selected NOR flash devices to investigate how latency varies with different influencing factors and to establish relationships among them. For instance, the variation in program time was analyzed for different data values being written in *S29GL064S70TFI040*. Specifically, it was examined whether the program time is shorter for data such as 0xFFFE, where only one bit is programmed while the remaining bits remain in the erased state, and conversely, whether it is longer for data like 0x0000, where all bits are programmed. These analyses provide valuable insights into the low-level nuances of internal mechanisms in a commercial flash memory device, which are typically undocumented in datasheets. Additionally, the impact of program-erase (PE) cycling on the erase time and word program time was evaluated to determine whether latency increases with the number of PE cycles. All of these factors and their effects on NOR flash performance are discussed comprehensively in this section.

3.3.1 Sector Erase Time Characterization

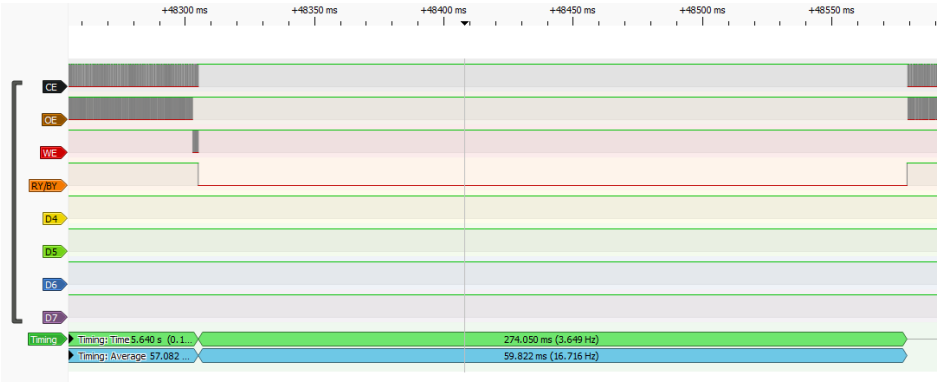


Figure 3.10: Logic analyzer capture of erase latency measurement

Erase time refers to the duration required for a flash memory sector to be fully erased, which corresponds to turning all bits in that sector to a logical '1'. The erase time is determined using

a logic analyzer by monitoring the RY/BY# signal behavior following the initiation of an erase command. Specifically, it is measured as the duration for which the RY/BY# signal remains LOW, indicating that the device is actively performing the erase operation. Figure 3.10 illustrates a logic analyzer capture showing an erase-time measurement where the RY/BY# signal stays LOW for approximately 274 ms.

The erase time varies as the flash memory undergoes more program/erase (P/E) cycles, reflecting changes in the device's internal characteristics. Characterizing the evolution of erase latency over cycling is therefore critical for assessing device reliability and understanding wear mechanisms.

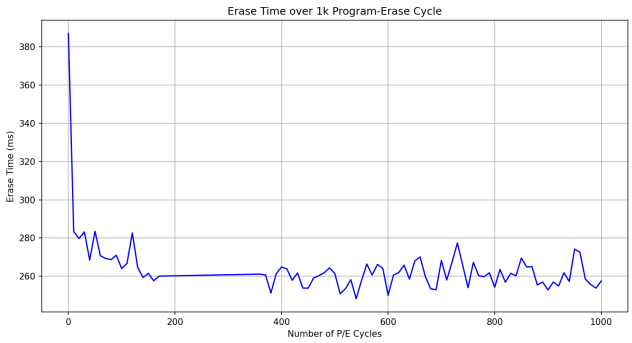
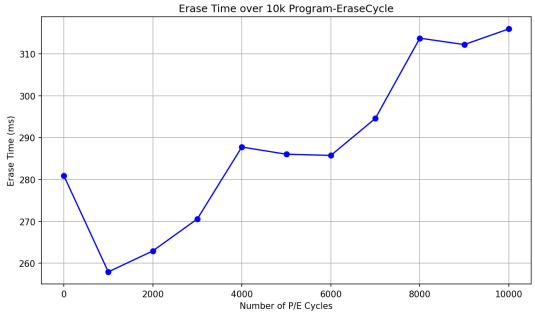


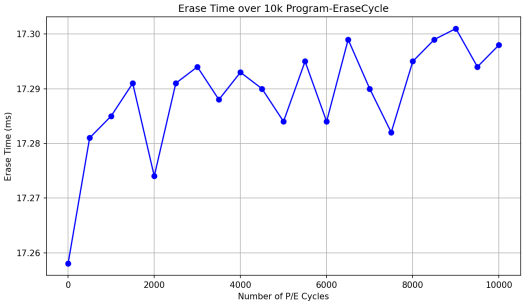
Figure 3.11: Effect of P/E cycle on sector erase time of *S29GL064S70TFI040* for 1k P/E cycles : Erase Time in (ms) vs Number of P/E Cycles

The sector erase latency trend over different P/E cycles was evaluated on all the four NOR flash devices selected for this study. Figure 3.12a shows the erase time variation for a 10,000 P/E cycles. It can be observed that the erase time increases from a 280.95 ms for fresh to 316.94 ms. This data was collected at an interval of 1000 cycles. It can be observed from the figure that the erase time decreases at first for a few P/E cycles and then keeps on increasing.

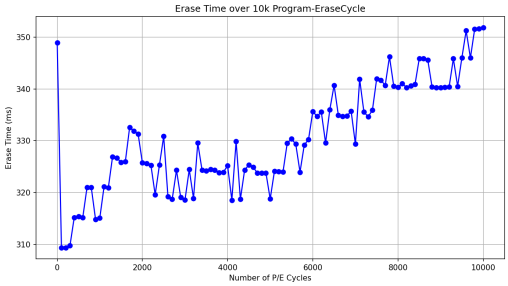
The Figure 3.11 shows the same data collected in a 10 P/E cycle resolution. It can be observed that the erase time is high only for the first P/E cycle. This trend is also observed in the MX29GL640EBTI and M29F800FT devices, which use conventional floating-gate (FG) cells,



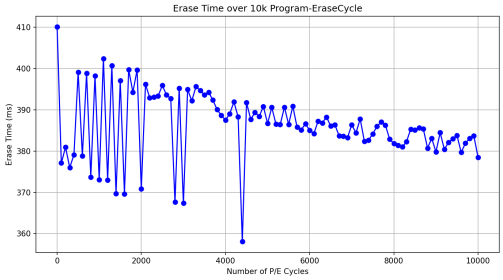
(a)



(b)



(c)



(d)

Figure 3.12: Effect of P/E Cycle on sector erase time of all other selected NOR flash devices for 10k P/E cycles - Erase Time in (ms) vs Number of P/E Cycles (a) Infineon S29GL064S70TFI040 (b) Microchip SST39VF1601C (c) Macronix MX29GL640EBTI (d) Micron M29F800FT

whereas the SST39VF1601C, which employs a split-gate cell architecture, demonstrates a different behavior, as evident from Figure 3.12.

This variation can be attributed to the differences in charge storage mechanisms and tunneling characteristics inherent to each cell structure. In floating-gate devices, the first erase cycle may exhibit longer latency due to the need to fully remove residual charges from manufacturing or prior use, establishing a baseline condition for subsequent erasures [55, 56]. Split-gate architectures, on the other hand, are known for improved control of the tunneling field and reduced variation in erase behavior, which contributes to more consistent timing across P/E cycles [57, 58].

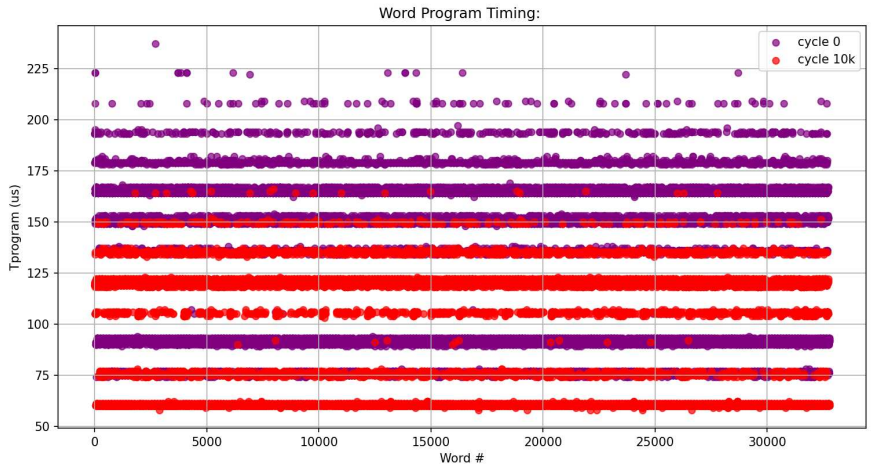
Table 3.3: Average word program and sector erase latency for different NOR flash devices

Flash Device	**Erase Time (ms)**		**Program Time (µs)**	
	Non-Cycled	10k P/E Cycled	Non-Cycled	10k P/E Cycled
S29GL064S70TFI040	280.95	315.94	123.35	92.08
SST39VF1601C	17.26	17.28	7.55	7.55
MX29GL640EBTI	348.92	351.86	9.35	9.33
M29F800FT	410	378.4	5.11	5.11

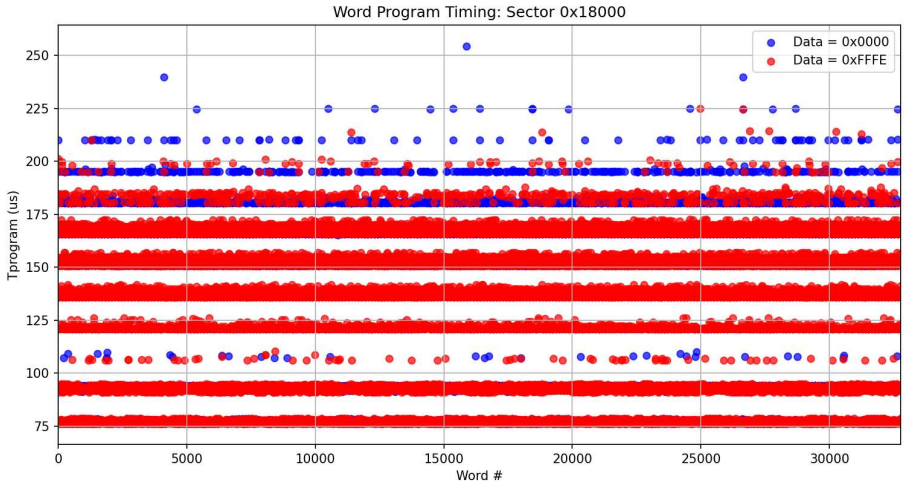
3.3.2 Word Program Time Characterization

The NOR flash supports programming at the word (or byte) level granularity unlike a NAND flash. The program latency of each word can reveal many information about the flash devices, which forms the basis for Chapter 4. An analysis similar to the sector erase latency is performed for word program time in all the four chips. The sectors of interest in the selected flash devices contain 32768 words. The program time of each word can be measured using Arduino or a logic analyzer capture. The measured values are visualized in a meaningful format using Python.

The data is collected across different sectors, for writing different data, and for P/E cycled and non-cycled sectors. The scatter plot shown in Figure 3.13 shows the program time measured for each words in a sector. It is evident that word program time is not uniform across the sector. Figure 3.13a shows the variation of word program time in a fresh sector and a 10k P/E cycled



(a)



(b)

Figure 3.13: (a) Variation in word program time based on P/E Cycles (b) Variation in word program time based on Program Data

sector. Unlike the erase time trend, here the latency is found to reduce with cycling. Whereas the Figure 3.13a shows the program time for writing different data within the same sector. Writing data 0x0000 to the flash, where all bits need to be flipped, consumes more time than writing a 0xFFFE in which only a single bit has to be flipped. This shows that program dependent on the type of data written, and it also hints about an internal program-verify operation.

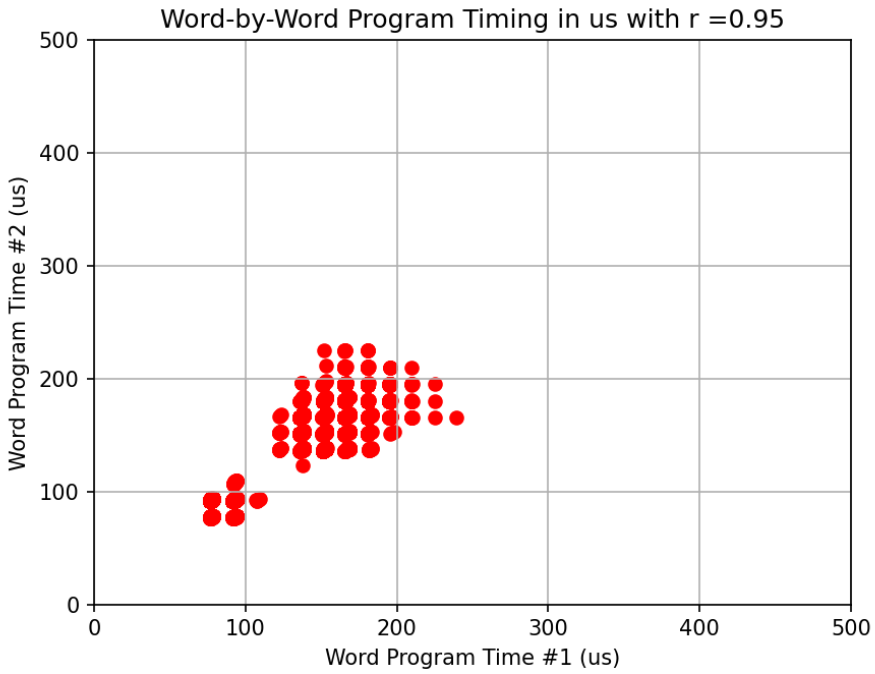


Figure 3.14: Correlation of word program time for different sectors

Additionally, Figure 3.14 presents a correlation plot of word program times between two different sectors. The latency measurements were taken from sectors with similar P/E cycles and programmed with similar data. The resulting correlation coefficient of 0.95 indicates a strong similarity in program latency across sectors, though not a perfect one-to-one correspondence.

3.4 Conclusion

In this chapter, we provide a detailed description of the experimental setup used for the research. The hardware setup consists of a custom-designed PCB that includes an Arduino Due

board and a 48-TSOP to 48-DIP socket, facilitating the interfacing of the flash memory chip with a host computer. This PCB serves as a robust and reliable platform for the experiments conducted. The firmware was developed using the Arduino programming environment and implements algorithms to control the flash memory chip, allowing for read, program, and erase operations. Additionally, it supports verification by accessing manufacturer and device-specific identification data. We evaluated four different flash memory chips: the Infineon S29GL064S70TFI040, Microchip SST39VF1601C, Macronix MX29GL640EBTI, and Micron M29F800FT. Data collection was conducted using a logic analyzer device, with data processing performed in a Python development environment. Finally, we performed timing characterizations for each chip, which included program time, erase time, and evaluations of partial program and partial erase operations.

Chapter 4

Reverse Engineering Memory Array Structure

4.1 Introduction

In commercial NOR flash memory devices, datasheets typically provide high-level specifications such as capacity, voltage requirements, and timing parameters. However, they often do not include detailed information about the internal memory array structure, such as the specific row and column organization within each sector. This lack of transparency creates a gap in understanding and modeling performance, as it limits the ability to explain certain low-level behaviors. For example, variations in program and erase times across different memory locations may be influenced by the underlying physical organization, which remains undocumented in publicly available resources.

The memory array of a NOR flash is divided into different sectors as shown in Figure 4.1. The sector architecture in flash memory can be uniform if the device contains sectors only of same size. All the flash parts chosen for the experiments have a non-uniform sector architecture. S29GL064S70TFI040 is a 64Mb boot sector model flash with one hundred twenty-seven 32-Kword (64-KB) sectors and eight-4 Kword (8-KB) boot sectors [59]. To illustrate the organization of the cell matrix, we refer to the S29GL064S70TFI040 NOR flash device as a representative example.

The target sector for the experiments consist of 32 Kwords or 64KB. The words are arranged in a sector in a 2D pattern with rows and columns. However, the number of rows and columns are unknown. Each words in the sector comprise of 16 bits. The device supports either word or byte configuration through the hardware pin $BYTE\#$. In this work, word configuration is chosen consistently for conducting all the evaluations.

The objective of this section is to determine the number of rows *m* and columns *n* in a sector, thereby inferring its physical layout. By reverse-engineering the physical organization based on

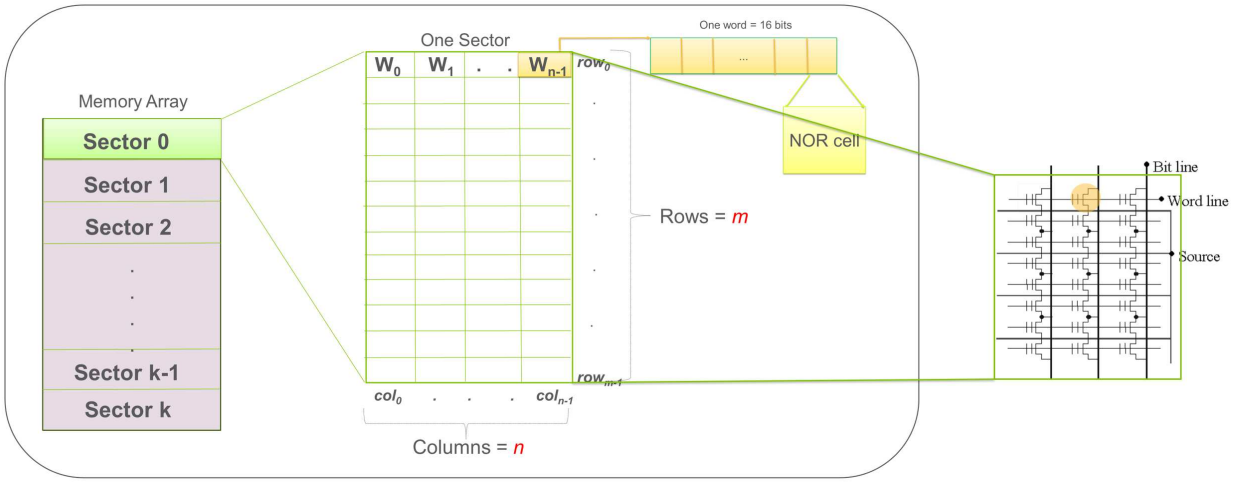


Figure 4.1: Memory organization of a NOR flash

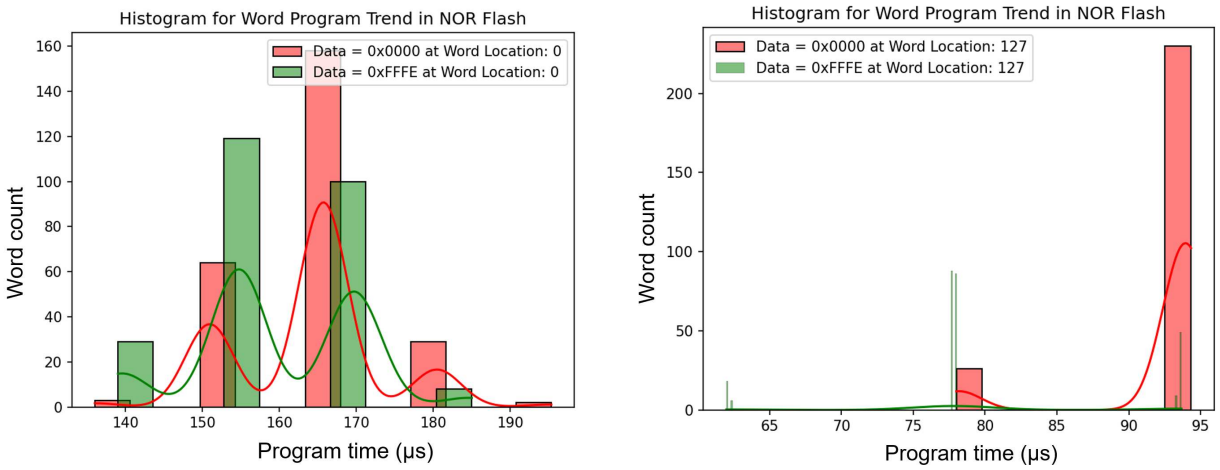
the timing characteristics of individual words, a hypothesized model of the memory array structure is developed and presented.

4.2 Motivation

During the timing characterization of word program time, as shown in Figure 3.13, it was observed that the program times exhibit significant variation across different memory words within a sector. This non-uniformity suggests an underlying physical layout affecting performance, motivating a detailed investigation into the memory array's structural organization.

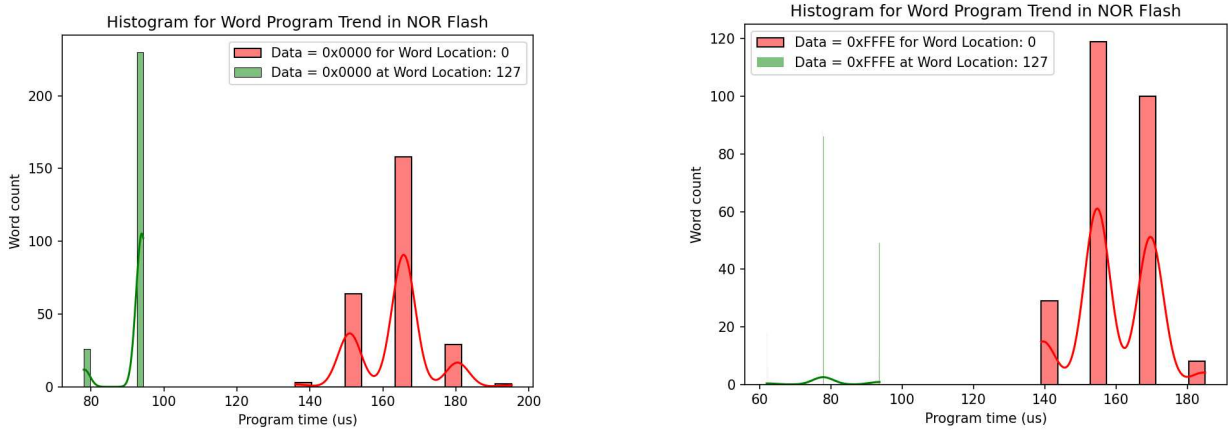
As an extension of the timing characterization, efforts were made to explore the relationship between the type of data written to memory and the corresponding program time. This analysis was performed using histograms and Cumulative Distribution Functions (CDFs). A histogram is a graphical representation that organizes data into bins, showing the frequency of data points within each range, while a CDF describes the probability that a variable takes a value less than or equal to a certain threshold [60].

Figure 4.2 and Figure 4.3 present the histogram plots analyzing the distribution of word counts and program times based on the programmed data values and word locations. Specifically, Figure 4.2a compares the word program times for writing data values 0x0000, 0x5555 and 0xFFFE at



(a) **(b)**

Figure 4.2: Histogram plot for analyzing the word program time distribution within the same column for data patterns 0x0000 and 0xFFFE (a) first word location in the sector array or Column 0 (b) last word location in the sector array or Column 127



(a) **(b)**

Figure 4.3: Histogram plot for analyzing the word program time distribution across the first (#0) and last (#127) column in the sector array for different data patterns (a) Data: 0x0000 (b) Data: 0xFFFE

word location 0 or the first column in the assumed memory array. Similarly, Figure 4.2b compares the word program times for writing the same set of data in word location 127 or the last column in the assumed memory array. Additionally, Figure 4.3 quantifies the timing distinction in writing the same value of data to different word locations. For example, it can be inferred from the Figure 4.3a that writing the data 0x0000 at word location 0 takes around 140 to 200 µs. Whereas, the same data can be written at word location 127 in less than 100 µs. A consistent pattern can be observed in Figure 4.3b as well. These results prove that data patterns and location in memory array influence the word program timing behavior.

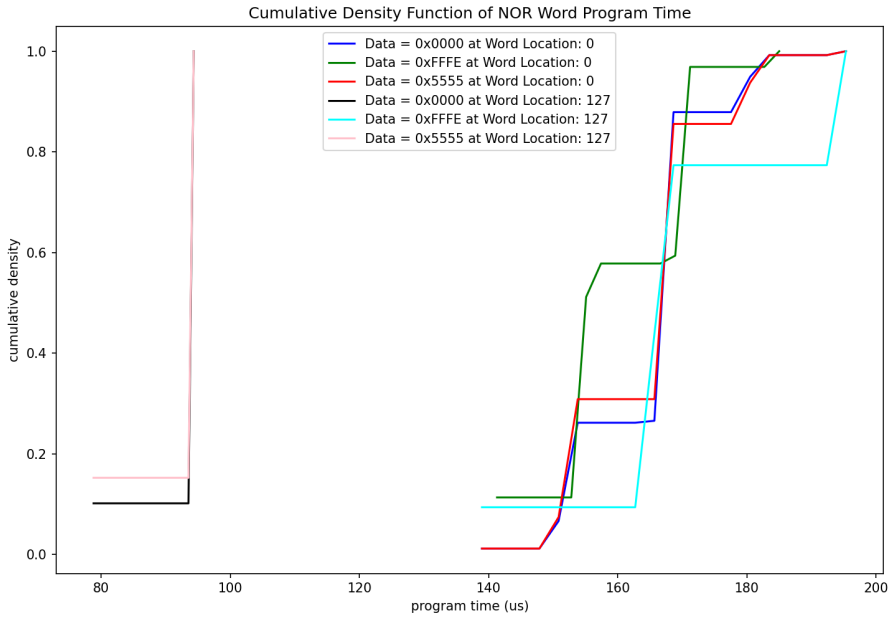


Figure 4.4: Cumulative Density Function for the word programming time taken for programming different data

Figure 4.4 illustrates how the distribution of programming time varies with the type of data being written to memory. It helps reveal patterns in timing behavior that are dependent on data value.

Moreover, as evident from Figure 4.3, a distinct grouping of word locations based on their program time can be observed. These groups can be broadly categorized into *fast-programming*

and *slow-programming* word locations. Identifying such locations enables the potential for adaptive programming strategies, where word allocation can be optimized depending on application requirements—such as prioritizing faster locations for time-critical data.

4.3 Methodology

The experimental procedure involves collecting program time data for each memory word and processing this data to generate a two-dimensional matrix plot, which is assumed to reflect the physical layout of the flash memory sector. The data collection can be done either through serial port or by exporting the timing information from logic analyzer capture to a text file. The data processing is completely performed in Python and is described in Algorithm 1.

Algorithm 1 Process to Map Program Time to Memory Matrix

1: Collect word program time data during P/E cycling of the device
2: Read program time per cycle into an array `progTimeArray`
3: Initialize `ROW_SIZE` and `COLUMN_SIZE` such that

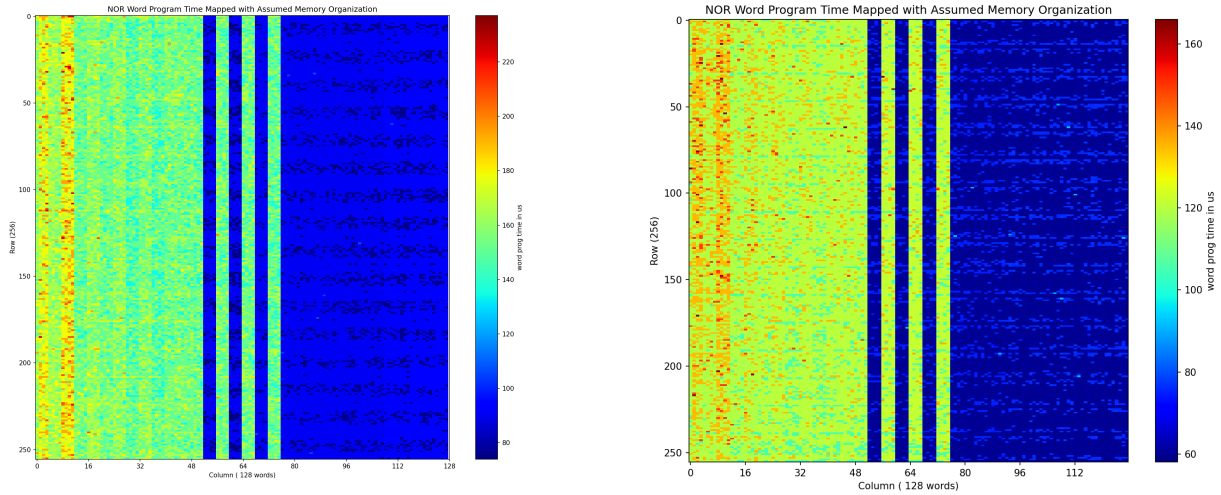
$$\texttt{ROW\_SIZE} \times \texttt{COLUMN\_SIZE} = 32768$$

4: Define matrix `progTimeMatrix` of size `[ROW_SIZE][COLUMN_SIZE]`
5: Initialize indices: `row` = 0, `col` = 0
6: **for** each element `t` in `progTimeArray[cycle]` **do**
7: `progTimeMatrix[row][col]` ← `t`
8: Increment `col` by 1
9: **if** `col` == `COLUMN_SIZE` **then**
10: Increment `row` by 1
11: Reset `col` to 0
12: **end if**
13: **end for**
14: Create a matrix plot of `progTimeMatrix` with x-axis as word index (columns) and y-axis as row number
15: Repeat with different `ROW_SIZE` - `COLUMN_SIZE` combinations

The product of the row size and column size should be $32,768$ (32 Kwords), which is equal to the size of the flash memory sector.

4.4 Memory Array Structure

This section presents an analysis of the internal memory array structure of flash devices, with a focus on understanding sector-level organization. The objective is to reverse-engineer the physical layout of sectors by examining word-level program timing behavior. Two classes of flash memory architectures are considered: MirrorBit™ and traditional floating-gate-based flash cells. The section is organized into two subsections: the first focuses on the structural inference for MirrorBit™ flash memory, where various row and column size combinations are explored to determine a configuration that best aligns with experimental observations. The second subsection discusses the timing characteristics of floating-gate memory, highlighting the limitations of applying the same structural analysis approach due to the inherently random nature of its program time distribution. Through comparative evaluation, the analysis aims to identify structural regularities unique to MirrorBit™ and establish a basis for further architectural modeling.

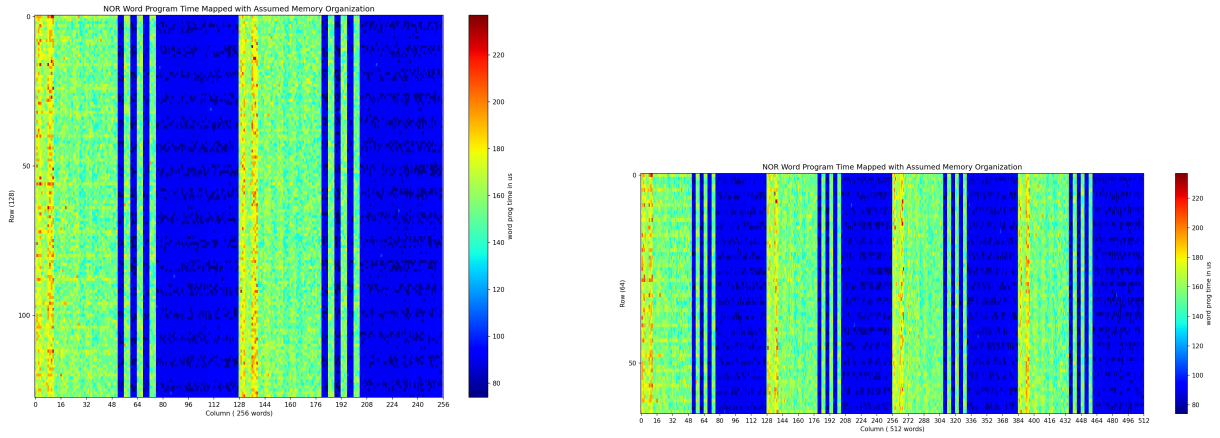


(a) **(b)**

Figure 4.5: Matrix plot of word programming time for assumed 256 x 128 words memory array in MirrorBit for (a) Fresh sector (b) 10k P/E cycled sector

4.4.1 MirrorBit™ Flash

The assumed sector array structure for S29GL064S70TFI040 device is provided in Figure 4.5, where Figure 4.5a is the program time collected for a non-cycled sector and Figure 4.5b is for a 10k P/E cycled sector. It can be observed that the columns in the first half of the sector consistently exhibit longer program times, while the latter half programs significantly faster. This indicates that the first half contains *slow-programming* word locations, whereas the second half comprises *fast-programming* word locations.

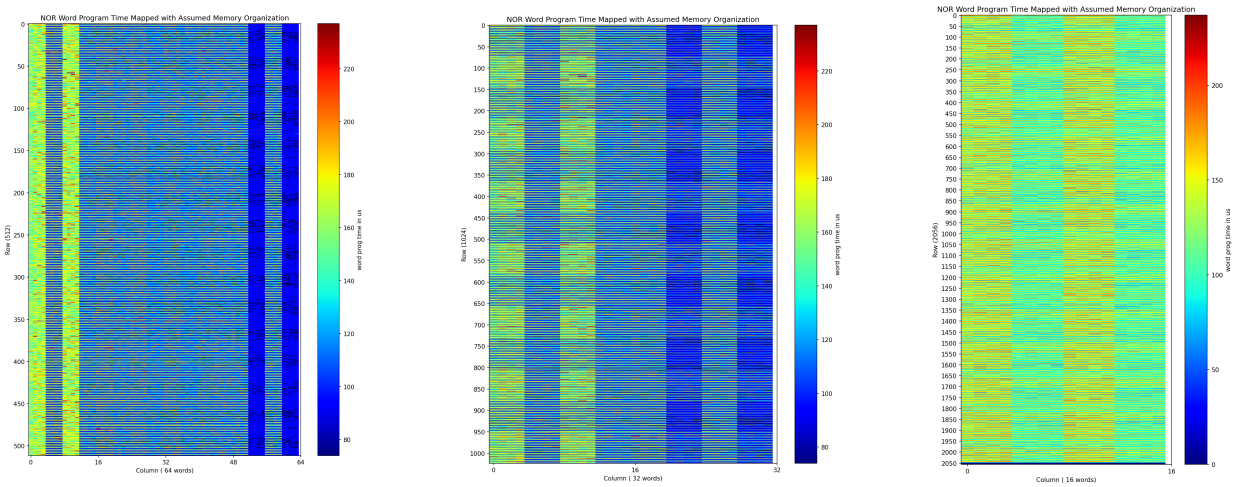


(a) **(b)**

Figure 4.6: Matrix plot of word programming time for assumed sector layout in MirrorBit (a) 128 x 256 words (b) 64 x 512 words

To validate the hypothesized physical layout of the flash memory sector in MirrorBit™ architecture, various combinations of row and column dimensions were systematically explored. The objective was to identify a matrix configuration that closely aligns with the observed programming time patterns. As shown in Figure 4.6, increasing the column size produces structures that repetitively mirror the pattern observed for the 256×128 word configuration presented in Figure 4.5. This redundancy suggests that such configurations do not provide new structural insights and are likely not representative of the actual physical layout.

Conversely, Figure 4.7 illustrates the effect of increasing the number of rows while reducing columns accordingly. However, these configurations—such as 512×64, 1024×32, and $2048 \times$



(a) **(b)** **(c)**

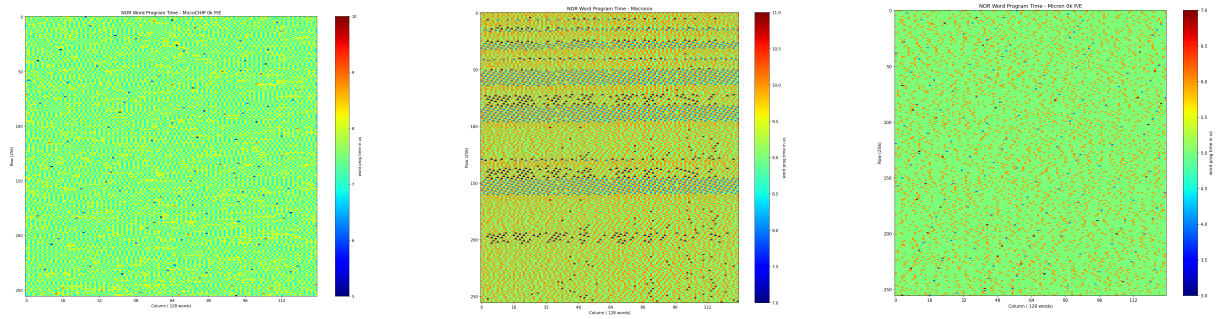
Figure 4.7: Matrix plot of word programming time for assumed sector layout in MirrorBit (a) 512 x 64 words (b) 1024 x 32 words (b) 2048 x 16 words

16—fail to capture the spatial differentiation in program time observed in Figure 4.5. Specifically, they do not exhibit the distinct separation between fast and slow programming regions, which is a key indicator of the underlying physical structure. Therefore, these alternative mappings were found to be inadequate in capturing the structural characteristics observed in the program timing data, reinforcing the validity of the 256×128 configuration as a more reasonable representation of the sector layout.

4.4.2 Floating Gate Flash

Apart from the MirrorBit™-based NOR flash memory, the sector layout was also investigated for floating-gate flash memory devices. This includes two conventional floating-gate flash memories and one SuperFlash® device, the latter employing a split-gate cell structure. Figure 4.8 presents the 2D matrix plots of word programming latency for each of these memory types. In contrast to the MirrorBit™ device—which exhibits program times in the range of 50–250 µs—all three floating-gate-based memories demonstrate significantly faster programming times, typically below 10 µs. This reduced timing range introduces challenges in achieving accurate characterization. Moreover, the measured data indicates that these devices do not exhibit the spatial program-

ming time patterns observed in the MirrorBit™ array. As a result, it is not possible to distinguish between fast- and slow-programming regions within a sector, thereby limiting the feasibility of reverse-engineering the physical sector layout in such architectures.



(a) (b) (c)

Figure 4.8: Matrix plot of word programming time for a 256 x 128 word sector layout in Floating Gate based memory flash (a) Microchip SST39VF1601C (b) Macronix MX29GL640EBTI (b) Micron M29F800FT

4.5 Conclusion

To summarize, the evaluation of various row and column configurations revealed that the 256×128 word structure most accurately reflects the internal organization of the MirrorBit™ flash memory sector. This configuration uniquely captures the spatial asymmetry in program timing, distinguishing fast- and slow-programming regions within the sector. Other configurations, such as those with increased column or row sizes, either resulted in redundant patterns or failed to preserve meaningful spatial differentiation. Additionally, this form of structural inference is not feasible in conventional floating-gate or split-gate flash memory devices, where program time distributions exhibit highly random and spatially uncorrelated behavior, preventing reliable reconstruction of the physical layout. These findings highlight the inherent architectural regularity of MirrorBit™ technology and provide a foundation for future investigations in memory array characterization and adaptive programming methodologies.

Chapter 5

Data Remanence in Sanitized NOR Flash

5.1 Introduction

Secure deletion of obsolete data from the storage medium is a fundamental requirement for safeguarding the privacy and security of data subjects. The Data Protection Act (DPA) 2018 requires that data erasure be both thorough and permanent, ensuring that deleted information is rendered completely unrecoverable.However, achieving true and complete data sanitization at the physical level is a non-trivial task. This challenge is largely attributable to the analog and device-specific characteristics of non-volatile memory technologies, which can significantly affect the efficacy of data removal techniques. As such, the secure deletion of data necessitates a nuanced understanding of the underlying storage medium to ensure compliance with legal and security standards.

Previous studies on secure deletion in NAND flash memory, such as [61] have demonstrated that data may still be recoverable even after the application of sanitization techniques. This recoverability arises from the inherent analog nature of flash memory cells, where variations in threshold voltage cause residual electrical states that can retain imprints of previously stored data. As a result, traces of the original data may persist and remain accessible through specialized recovery methods even after applying digital sanitization techniques. This vulnerability highlights the limitations of purely digital deletion approaches and underscores the necessity for analog-level sanitization techniques to ensure truly secure data erasure in flash memory systems. However, comparable investigations have yet to be conducted on NOR flash, leaving an important gap in the comprehensive understanding of secure deletion in this storage platform.

In this chapter, the principles and challenges associated with secure deletion in NOR flash-based non-volatile storage systems are examined.

5.2 Motivation

With the rapid proliferation of embedded and connected systems across domains such as automotive, industrial automation, and consumer electronics, the reliance on non-volatile memory for persistent data storage has become increasingly critical. Among various technologies, NOR flash memory has emerged as a preferred solution for applications requiring fast random access, code storage, and high reliability. However, as devices incorporating NOR flash approach obsolescence and enter the waste stream, concerns regarding residual data security and potential unauthorized access are rising. This thesis is motivated by the dual need to understand the growing role of NOR flash in embedded systems and to address the potential security vulnerabilities associated with data remanence in such memory devices.

5.2.1 NOR Flash Popularity

NOR flash memory is a mainstream technology for applications that require the storage of code and parameters such as encryption and decryption keys, and is more commonly used in embedded memories that have to provide random and fast memory access [7]. The demand for NOR flash has grown due to its fast random access speed, high reliability, and suitability for execute in place applications in Internet of Things, automotive, and industrial systems—especially where code storage and frequent updates, such as over-the-air programming, are required. The NOR flash market, valued at $2.9 billion in 2024, is projected to reach $3.82 billion by 2030, with a compound annual growth rate (CAGR) of 6.01% [2,3]. This growth is further supported by Macronix's three-dimensional NOR technology [4–6], which overcomes the scaling limitations of two-dimensional NOR and offers improved performance and cost efficiency in compact four gigabit single-chip solutions. NOR flash is increasingly used across sectors such as automotive, consumer electronics, and communications.

5.2.2 Device Obsolesce and E-Waste Data Breach Concerns

E-waste (electronic waste) refers to discarded electrical or electronic devices. It includes a wide range of electronic devices, including computers, mobile phones, electrical appliances, and other consumer electronics. The generation of e-waste has become a significant global concern due to the rapid advancement of technology and increasing consumer demand for electronics. According to recent statistics of [62], approximately 50 million tonnes of e-waste is generated annually, with the record high being approximately 57 million metric tons globally, in the year 2021. The global volume of e-waste is projected to reach 74 million metric tons by the year 2030, pointing towards a a continued upward trend in e-waste generation.

The wide adoption of NOR flash in control-critical real-time systems raises significant security concerns due to the risk of residual data exploitation after device end-of-life. In mobile phones, NOR flash is used to store the initial code (bootloader) that starts the phone when it's powered on, as well as the firmware that controls the phone's basic functions. Additionally, in laptops, NOR flash is used in the BIOS (Basic Input/Output System) or UEFI (Unified Extensible Firmware Interface) chip, which is responsible for initializing the hardware and starting the operating system [63]. The average lifespan of a mobile phone is around 2-3 years and it is estimated that over 150 million mobile phones are discarded annually in the United States. Similarly, the average lifespan of a computer is approximately 3-5 years. In the United States, over 40 million computers become obsolete each year and among them, 10-15% of e-waste from computers and laptops are recycled.

Recent studies [64–68] highlight the risk of sensitive data retrieval from recycled electronics including from e-waste and second-hand market. Figure 5.1 illustrates the process of data recovery from discarded electronic devices, including e-waste collection, NOR flash retrieval from circuit boards, and subsequent data extraction from the recovered flash memory. The recovered data may include highly sensitive and confidential information such as Personal Identification Information (PII), which can be used to distinguish or trace an individual's identity. [64, 67] demonstrate successful retrieval of previously stored information from recycled electronics sold as new devices in the market.

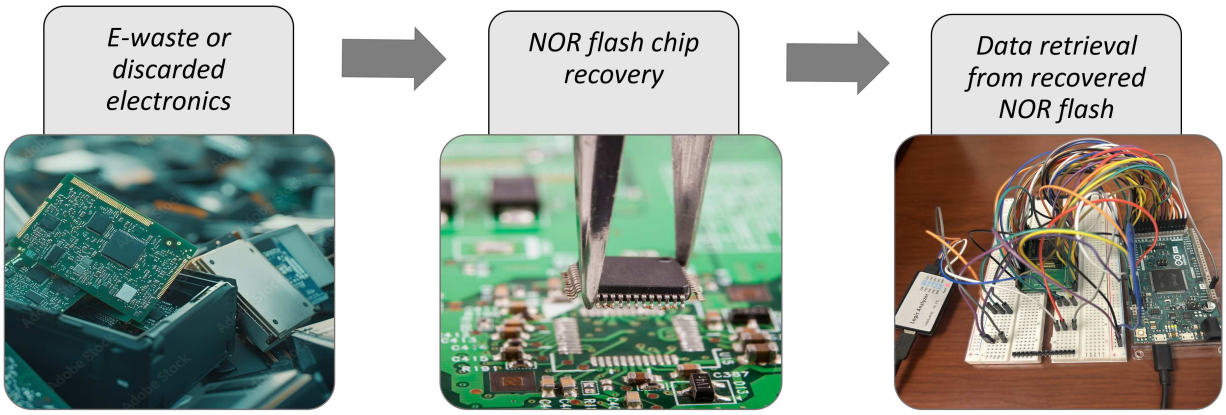


Figure 5.1: Overview of data recovery from e-waste

As embedded devices reach obsolescence and contribute to the growing volume of e-waste, improper data sanitization practices can leave sensitive data—such as firmware, security credentials, and configuration settings—exposed to unauthorized access [64, 65, 69]. This is particularly concerning for applications in automotive ECUs, IoT nodes, and industrial controllers, where persistent data may include proprietary algorithms, encryption keys, or system diagnostics. Unlike volatile memories, NOR flash retains data even after power is removed, and common erase operations may not fully eliminate stored charge, leading to potential data remanence [61, 70, 71]. As a result, discarded devices become attractive targets for adversaries seeking to extract valuable information from improperly sanitized NOR flash chips. Recent news articles reveal the extend of data theft from e-waste and their adverse effects [72–76]. These risks highlight the importance of characterizing and mitigating data remanence vulnerabilities in NOR flash. Therefore, proper sanitization practices and assurance of irreversible data destruction of sensitive information from electronic devices is of paramount importance before discarding them.

The majority of prior research concerning data remanence and unauthorized data retrieval from discarded devices has focused predominantly on NAND flash memory. However, NOR flash memory is equally vulnerable to these data theft threats, despite the fact that the potential for such attacks and the corresponding solutions to mitigate this vulnerability in NOR flash have not been

comprehensively explored. Therefore, this research is of considerable significance as it provides novel insights into data remanence in NOR flash memory and proposes recommendations to address this critical issue.

5.3 Related works

In this section, the prior research related to data remanence and sanitization techniques in flash memory are discussed.

5.3.1 Data Remanence in NAND Flash

Flash-based non-volatile memory systems exhibit unique challenges in ensuring secure data deletion due to their analog cell-level characteristics. Scrubbing is a sanitization technique which involves overwriting pages to logically delete content and it is widely deployed in NAND flash. However, recent studies reveal that these sanitization methods are insufficient for complete data erasure [61, 64–67, 71] since flash memory cells store information as analog threshold voltages.

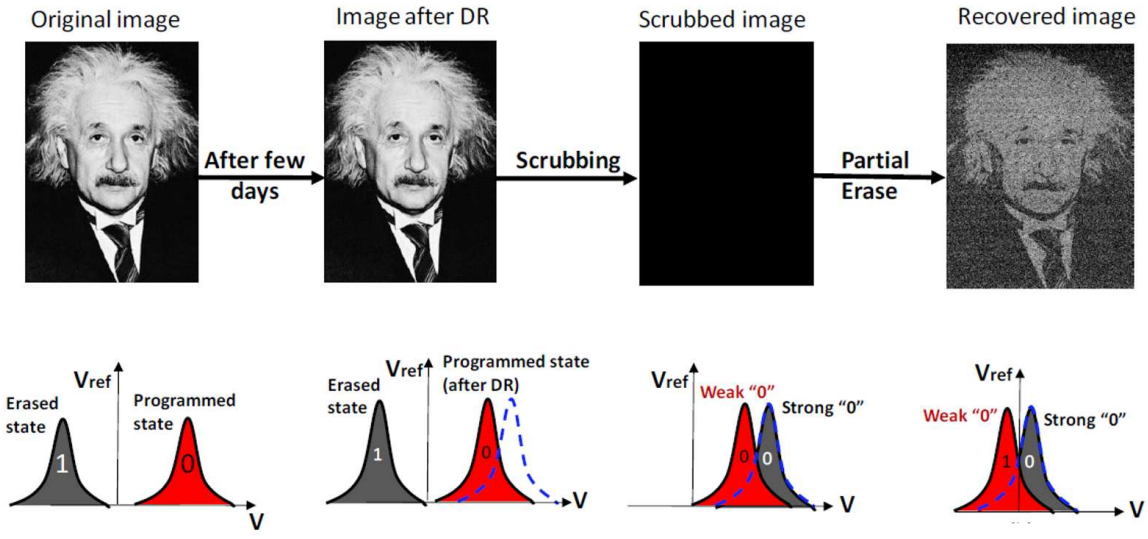


Figure 5.2: Data recovery in NAND flash presented in [61]

A study conducted by Hasan et al. [61] demonstrates that data can be partially or even fully recovered from sanitized NAND flash chips using conventional scrubbing techniques. Figure 5.2 shows the data recovery results presented by Hasan et al. The authors employed partial erase operations via standard digital interfaces to successfully extract deleted content from commercially available flash memory. This vulnerability arises due to the inherent data retention property of flash cells, which, while essential for non-volatility, inadvertently enables adversaries to recover sensitive data even after deletion. To address this issue, Hasan et al. propose analog scrubbing, a sanitization method that involves partial program operations tailored based on metadata, such as file creation time. This approach reduces remanent charge artifacts more effectively than conventional digital overwriting techniques by minimizing the distinguishability of previously programmed states at the analog level. This work highlights a critical insight that logical deletion does not guarantee physical deletion in flash memory systems. Secure sanitization thus requires a deep understanding of low-level memory behavior and the development of analog-aware techniques to ensure robust data privacy.

Garg et al. [71] investigate data deletion vulnerabilities in commercial NAND flash-based storage devices, highlighting security concerns in critical applications. Through extensive software-based recovery experiments on 8GB and 16GB NAND flash drives using differet file systems such as NTFS and FAT, the study demonstrates that standard delete and erase commands in Windows and Linux fail to effectively sanitize data, achieving 100% data recovery in multiple iterations. The paper attributes these vulnerabilities to inherent NAND flash management techniques like wear leveling and bad block management, which, while improving performance and endurance, complicate reliable data deletion.

While the residual voltage levels in partially programmed or erased cells after sanitization is studies and utilized for NAND flash and Solid State Drives (SSDs) to retain physical imprints of previous data, such a possibility remains unexplored in NOR flash memory. This gap in existing research further motivates this thesis.

5.3.2 State of the Art Sanitization Methods

This section provides a concise overview of current sanitization methods used in data storage systems. Non-volatile memory (NVM) devices are predominant in data storage solutions. In some cases, these devices are customized for storing sensitive documents and classified information. These devices can be vulnerable if firmware control is lacking, increasing the risk of data breaches. Unlike older data storage media such as Compact Discs and Floppies, modern flash storage faces serious drawback of data remanence, where residual data remains even after deletion, requiring complex sanitization procedures. Therefore, several sanitization methods have evolved overtime [77]. There are four popular sanitization techniques: analog sanitization, digital sanitization, logical sanitization and cryptographic sanitization [78–80].

- **Block / Sector Erasure:** Sector erase is a fundamental command that removes data from the target sector in a NOR flash either by removing the stored charge or by injecting holes in the memory cells, effectively erasing data physically. This command is primarily used during garbage collection when the storage is nearly full [81–83]. Some secure deletion methods also use block erasure to achieve data sanitization [84–86]. However, block erasure suffers from significant drawbacks including performance degradation due to the overhead of migrating valid data [86] and limited device endurance caused by frequent erase cycles. Therefore, block erasure is not an efficient solution for immediate or frequent sanitization. Although prior works demonstrating this technique focused on NAND flash, it can also be applied to NOR flash sectors.

- **Logical Sanitization:** To avoid the performance cost of physical erasure, NAND flash often uses logical sanitization by marking outdated pages as invalid and writing updates to new locations. However, the old data remains physically present and can potentially be recovered by adversaries. This behavior is specific to NAND flash and does not apply to NOR flash, which uses direct, byte-level access without page remapping.

- **Encryption Based Sanitization:** Recent work proposes encryption-based sanitization methods where user data is encrypted with a key stored separately in the flash memory [79,87–92]. Secure deletion is achieved by erasing the encryption key, rendering the encrypted data inaccessible. Different cryptographic methods are utilized for sanitization; for instance, SAFE [79] offers both rapid performance and verifiability. The most efficient approach to sanitization involves utilizing devices that encrypt their data while erasing the encryption keys from the key store. The success of various sanitization methods depends on the robustness of the encryption system employed [93], as well as the ability to mitigate "side channel" attacks that could enable an adversary to recover the classified information or circumvent the encryption. Although this method provides fast sanitization and requires less space for key storage, it faces limitations. First, weaknesses in key generation, such as poor randomness, can compromise security [94]. Second, all copies of the key and related metadata must be properly erased to prevent recovery. Third, this technique cannot be implemented in systems lacking built-in encryption support.

- **Data Scrubbing or Zero Overwrite:** Data scrubbing involves overwriting a page with a uniform pattern, such as all zeros, which effectively programs all cells in the page [78,86,95]. This technique offers an alternative to physical erase and enables page-level sanitization. However, Hasan et al. [61] demonstrate that scrubbed data can still be partially or fully recovered due to the analog behavior of flash cells, exposing vulnerabilities in this approach.

A study by Schneider et al., [64] found that data is recoverable from resold USB flash drives, with 12% of the 614 analyzed devices containing user data. This highlights poor sanitization practices in low-cost drives sold as new, likely due to chip reuse in the second-hand market, and raises serious concerns for data security and digital forensics.

Some recent works also explore physical destruction techniques to prevent unauthorized data access from obsolete flash media. Kumar A. et al., [69] present a novel approach for the permanent destruction of flash memory devices using thermal treatment. The method involves subjecting the chip to a high temperature of 800°C for 15 minutes, followed by examination of the flash array

using a Scanning Electron Microscope (SEM) and analysis through histograms to confirm the extent of thermal destruction. This technique resulted in complete and permanent damage to the flash device due to exposure to extreme heat.

5.4 Methodology

This section explains the experimental procedure for data recovery from a sanitized NOR flash memory. Before beginning the experiment, let's define the threat model and assumptions.

5.4.1 Threat Model

Adversary Model: We assume that the adversary has at least one-time physical access to the flash memory device. The adversary is capable of performing multiple read and erase operations on the flash. Furthermore, the adversary aims to extract sensitive information that is not accessible through legitimate system interfaces. We assume access to a raw NOR flash memory chip, enabling the adversary to perform low-level memory operations such as partial erase and partial program.
Assumption: We assume the adversary can read the data from the NOR flash without any error correction.

5.4.2 Process Overview

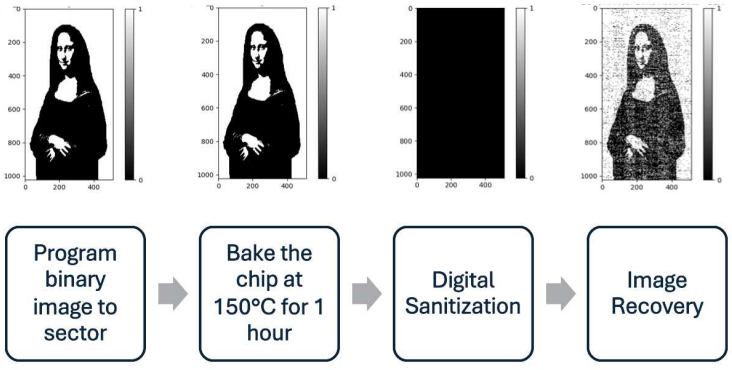


Figure 5.3: Overview of data recovery procedure in NOR flash

An overview of the steps involved in the data recovery procedure is provided in Figure 5.3. The first step in the process is programming input image to a sector in the flash memory. Since flash memory stores data in binary, the input image needs to be represented in binary format. This is achieved using the Python library OpenCV (cv2) through image thresholding. The steps involved in the input image processing is given in Figure 5.4. The input image is first converted to grayscale image, which represents each pixels in the image in a range of 0 to 255. Thresholding is then applied to convert all pixel values above the selected threshold (127) to 255 (white) and those below to 0 (black), resulting in a binary image [96].

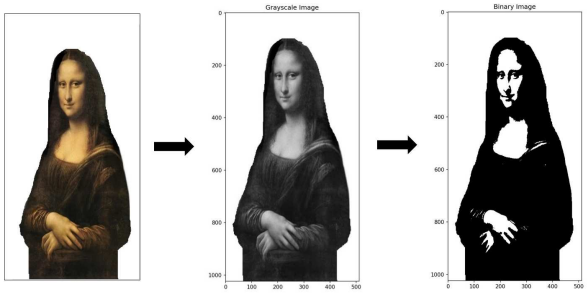


Figure 5.4: Image processing steps to generate binary input image

The binary image generated is converted to a hexadecimal word array, which is programmed to the flash memory. The input chosen here has a size of 32 Kword, which is the capacity of the sectors in the selected commercial flash parts. The dimensions of the image is 512x1024 bits. The image program is verified by reading back the sector after program and visualizing the sector contents using a matrix plot. The next important verification required prior to beginning the experiment is done by performing the partial erase and partial program operations in the flash. These two operations are performed for a range of delays to find the optimal window of delays for image recovery.

The second step in the process is baking the chip at 150 °C for 1 hour to perform temperature acceleration of data retention. This process is equivalent to data retention in the chip for a larger

time, may be years. The equivalent data retention period can be calculated using the equations provided in (5.1) and (5.2). Therefore, baking

Data retention of any flash is the ability to retain its programmed state. Flash data retention is known to degrade over temperature. Various tests and methods are in place to determine the reliability of flash.To test the flash data retention at various temperatures, accelerated tests are performed on the flash. These tests are based on Arrhenius law and equation [97]. The Arrhenius theory allows the test of any device under accelerated environments for short periods and predicts the behavior under normal conditions for longer periods [98]. The equation is given in (5.1).

$$A = \exp\left(\frac{E_a}{k_B}\left(\frac{1}{T_{\text{use}}} - \frac{1}{T_{\text{stress}}}\right)\right) \tag{5.1}$$

where,

A = Acceleration factor

E_a = Activation energy

k_B = Boltzmann's constant (8.623 × 10^{-5} eV/K)

$T_{\textbf{use}}$ = Normal operating temperature in Kelvin

$T_{\textbf{stress}}$ = Stress temperature in Kelvin

The data retention time can be calculated from the Acceleration Factor using the (5.2).

$$\mathbf{t}_{\text{use}} = \mathbf{A} \times \mathbf{t}_{\text{stress}} \tag{5.2}$$

Assuming E_a as 1 eV, the retention time t_use is calculated to be approximately 8.7 years for the given experiment conditions.

The baked chip is allowed to cool to room temperature before performing the digital sanitization and image recovery processes. The digital sanitization involves overwriting the baked image

with all-zeros. This is a popular technique practiced for ensuring security of obsolete data [61]. The image recovery is evaluated on the sanitized sector.

The procedure for preparing the chip for data retention acceleration is explained in Algorithm 2. Two sectors are chosen for basic evaluation - one fresh sector and a 10k P/E cycled sector. The binary image is written to these sectors and the chip is subjected to baking in a preheated oven.

Algorithm 2 Data Retention Acceleration

1: Perform 10k P/E cycles on `Sector1`
2: Program a fresh sector `Sector2` with the Monalisa image
3: Program `Sector1` with the Monalisa image
4: Bake at 150°C for 1 hour

5.4.3 Evaluation Metrics

In order to evaluate the efficiency of image recovery, we consider two metrics: Relative Bit Error Rate (RBER)% and Bit Accuracy%.

Relative Bit Error Rate (RBER%) is defined as the percentage of bits that have flipped from their original state in the recovered data. It quantifies the extent of corruption or deviation from the original image.

```
total_bits = 0  # Initialize total number of bits compared
change = 0      # Initialize count of bits flipped

# Iterate through each bit position in the matrices
for row in range(r): #r = number of rows in image
    for col in range(c): # c = number of columns in image
        if bmatrix[row, col] != omatrix[row, col]:
            # Increment flipped bits count if mismatch
            change+=1
        total_bits+=1

# Calculate percentage of flipped bits (BER%)
percentage = (change / total_bits) * 100
```

Figure 5.5: code snippet of Relative Bit Error Rate% estimation

The Python code snippet of RBER% calculation is provided in Figure 5.5 and it is calculated as:

$$\text{RBER } (\%) = \left(\frac{N_{\text{flipped}}}{N_{\text{total}}} \right) \times 100 \tag{5.3}$$

where N_{flipped} is the number of bits that are different between the recovered image and the original image and N_{total} is the total number of bits in the image.

A higher RBER% indicates greater deviation from the original data, whereas a lower RBER% signifies better data retention or more successful recovery.

Bit Accuracy (%) is defined as the percentage of bits in the recovered data that remain identical to the corresponding bits in the original data. It serves as a direct measure of how accurately the original information is preserved or reconstructed after the recovery process. Bit Accuracy is calculated as:

$$\text{Bit Accuracy } (\%) = \left(\frac{N_{\text{identical}}}{N_{\text{total}}} \right) \times 100 \tag{5.4}$$

where $N_{\text{identical}}$ is the number of bits that are identical between the recovered image and the original image.

Since every bit is either flipped or identical, Bit Accuracy and RBER are complementary:

$$\text{Bit Accuracy } (\%) = 100 - \text{RBER } (\%) \tag{5.5}$$

A higher Bit Accuracy value indicates better retention of the original data, with fewer errors or distortions introduced during the baking and recovery process.

The steps involved in image recovery are discussed in the next section.

5.5 Data Recovery

The evaluation of data recovery on sanitized NOR flash memory is performed on four different commercial NOR flash chips based on distinct cell structures such as the conventional floating gate, split-gate and charge trap. In S29GL064S70TFI040, which is a NOR flash memory based on

MirrorBit™ technology developed by Spansion, data recovery was achieved with partial program and partial erase. This flash has higher word program and sector erase time compared to the remaining chips. therefore, control of partial erase delay ($t_e{}^{\mathrm{d}}$) and partial program delay ($t_p{}^{\mathrm{d}}$) is easier.

5.5.1 Partial Program Recovery

As mentioned in Chapter 3, partial program is the premature termination of a word program operation. The partial program operation is performed in the flash memory to understand the range of delays at which at least 50 % of the bits are flipped to 0's (programmed state).

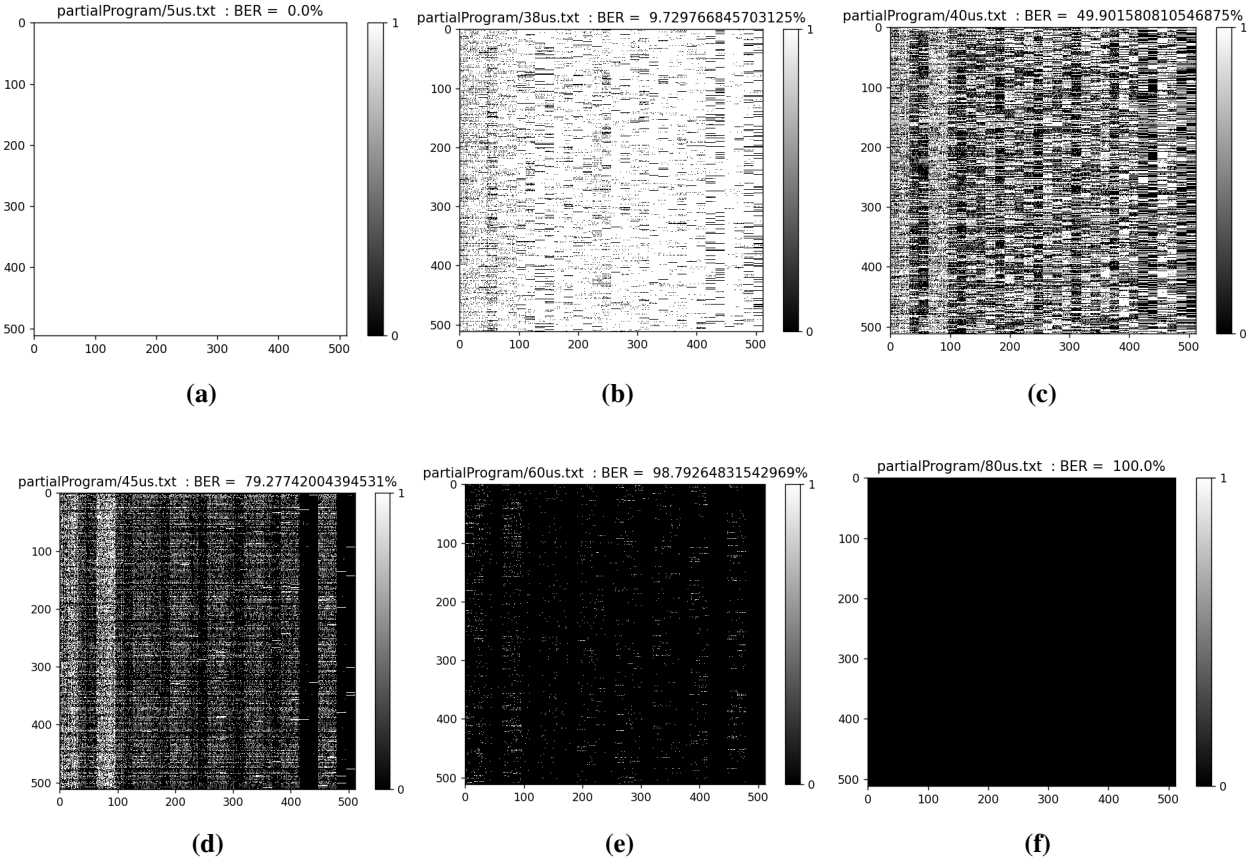


Figure 5.6: Data read from a flash sector at varying partial program time (tpp) (a) 5 µs (b) 38 µ s (c) 40 µ s (d) 45 µ s (e) 60 µ s (f) 80 µ s

Figure 5.6 shows the 2D matrix plot of partially programmed sector read data for varying partial program delay ($t_p{}^{\mathrm{d}}$) in a non-cycled device. The variation of the read data from the original or target data is measured in terms of Relative Bit Error Rate (BRER). Here, the RBER% is calculated with respect to an erased sector, which means all bits are one. Therefore, the RBER% for Figure 5.6a, which is partial program at 5 µs, is 0%, which means program operation hasn't started yet. As the ($t_p{}^{\mathrm{d}}$) increases, the RBER% also increases and finally it reaches the maximum value 100%, which means complete program - all bits are flipped to 0.

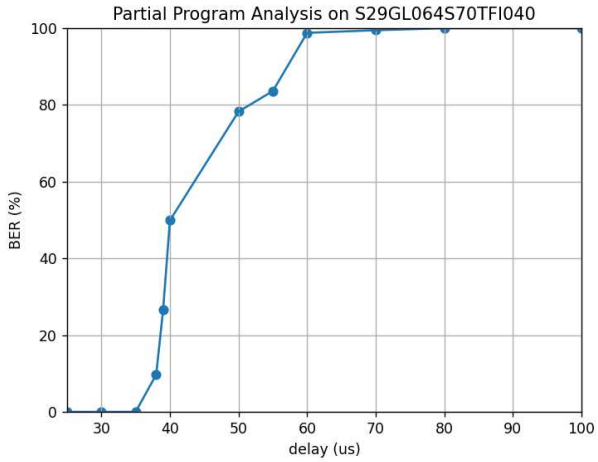


Figure 5.7: Partial program time vs Bit Error Rate

The RBER% vs ($t_p{}^{\mathrm{d}}$) plot is given in Figure 5.7. At 40 µs, the RBER is approximately 50%, which can be selected as the optimal ($t_p{}^{\mathrm{d}}$) value for data recovery.

Algorithm 3 Partial program for data recovery

Full Erase `Sector1`
for $i = 0$ to 32 kwords **do**
  i^{th} word $\leftarrow$ Img$[i]$
  Program word to `Sector1`
  Partial program delay $t_p^{\mathrm{d}} = 40$µs
  RESET
end for
Read `Sector1`

The implementation of data recovery through partial program is given in Algorithm 3. The image is written to a 10,000 P/E cycled sector and baked at 150°C for 1 hour, which simulates the data retention for approximately 8 years. The sector is fully erased after bake. Therefore, now all the cells in the sector are fully-erased. A word-program command is issued in the sector and after the specified delay, which in the initial case is 40µs, a hardware $RESET$ is issued. This involves writing a logic LOW pulse at the $RESET\#$ pin. This resets the device and thus the ongoing word program operation is terminated. This step is iterated for all words in the sector. The partially programmed sector is read to verify the data recovery. A device level explanation

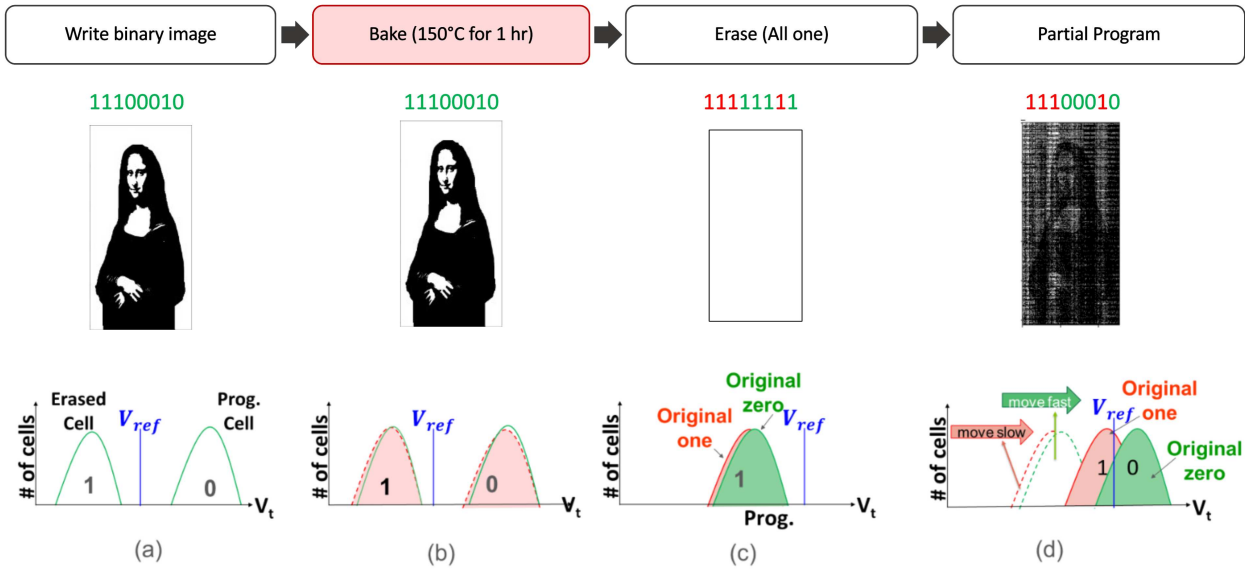


Figure 5.8: Image recovery through partial program operation

for this phenomenon in terms of threshold voltage (V_t) is provided in Figure 5.8. The threshold distribution during programming the image is given in Figure 5.8a. Programmed bits, which are represented with black color in the binary image, have higher voltage threshold and the erased bits (white) have lower threshold. After several years, a change in a V_t levels of these cells may occur, which is obtained here using baking. However, this shift is not major and the programmed distribution is still above V_{ref} and erase distribution is still below V_{ref}. Therefore, the image is same as the original image after baking, as can be seen in Figure 5.11b. However, due to the

nitride layer reliability issue involved in the charge trap cells [99], a mismatch in the program duration of original zero and original one cells occur, which is explained next in this section. Both original zero and original one cells are made 1 through full erase, as shown in Figure 5.11c.

The final step involved in image recovery is partial program. The partial program in terms of V_t level is bringing the voltage distributions of "original 0" and "original 1" (which are currently 1) towards (V_{ref}) in a controlled step-by step manner. The V_t levels above and below V_{ref} are read as 0 and 1 respectively. Therefore, the green and red distributions are brought between V_{ref} through partial program as shown in Figure 5.11d. It is found that there is now a small mismatch in the word program time for original zero and original one cells. Original zero, or the programmed cells in the original image are now faster to program compared to the original one or erased cells in the original image. Therefore, the original zero distribution (green) moves first to the V_{ref} than the original one (red). When the program operation is terminated at the point of original one < V_{ref} < original zero, the image is partially recovered since the distribution now slightly matches that of the original image which was baked.

Even though the data recovery is proved to be possible with partial program, more fine-tuning is required to obtain a better bit accuracy with the original image. This is because the current approach applies a constant (${t_p}^{\mathrm{d}}$) to all the words in the sectors. However, as discussed in Chapter 4, the program time for each words in a sector is non-uniform. Therefore, a dynamic (${t_p}^{\mathrm{d}}$) is required based on word-by-word program times, which can improve the quality of the data recovered.

5.5.2 Partial Erase Recovery

Partial erase is the premature termination of an erase operation. The partial erase operation is performed in the flash memory to understand the range of delays at which at least 50 % of the bits are flipped to 1's (erased state).

Figure 5.9 shows the 2D matrix plot of partially erased sector read data for varying partial erase delay (${t_e}^{\mathrm{d}}$) in a non-cycled device. The RBER% for partial erase recovery is calculated with respect to a fully programmed sector. Therefore, the RBER% for Figure 5.9a, which is partial erase at 110

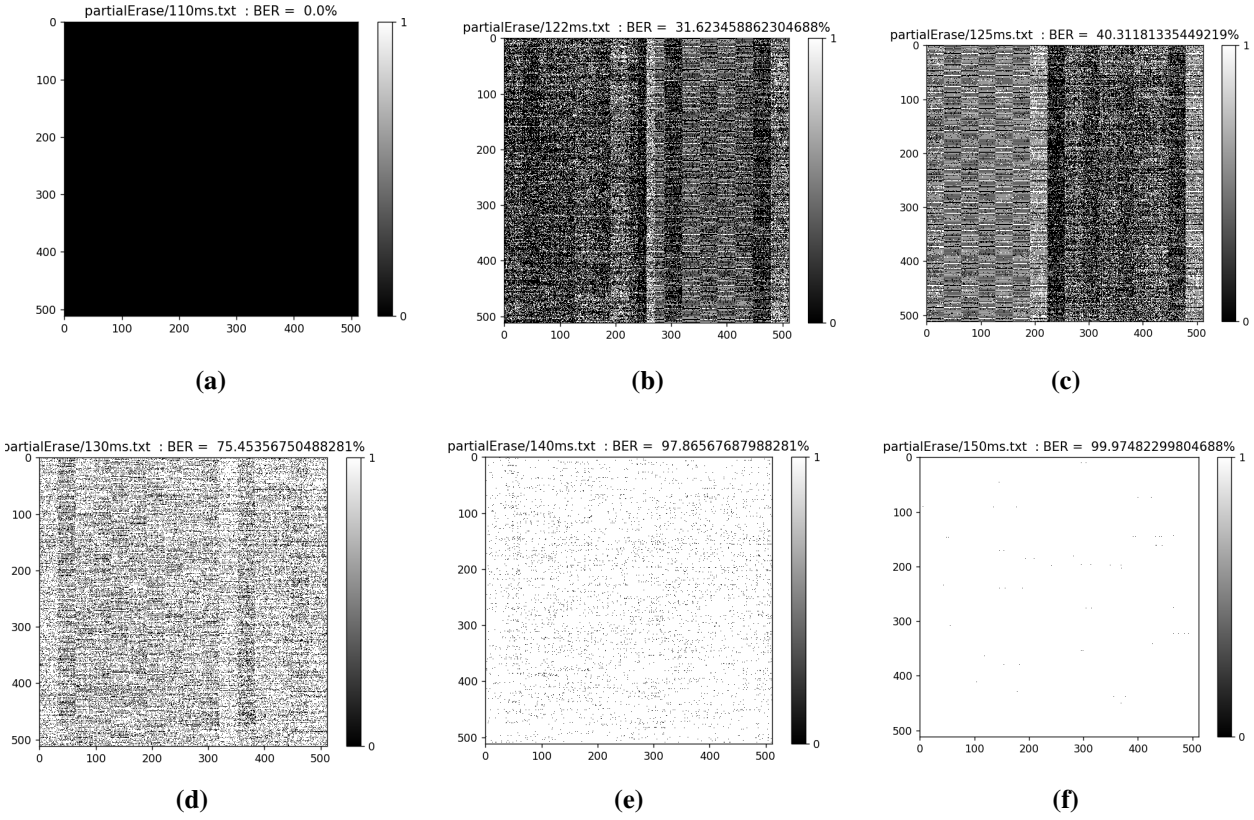


Figure 5.9: Data read from a flash sector at varying partial erase time (tpe) (a) 110 ms (b) 122 ms (c) 125 ms (d) 130 ms (e) 140 ms (f) 150 ms

ms, is 0%, which means erase operation hasn't started yet. As the ($t_e{}^{\mathrm{d}}$) increases, the RBER% also increases and finally it reaches the maximum value 100%, which means complete erase - all bits are flipped to 1.

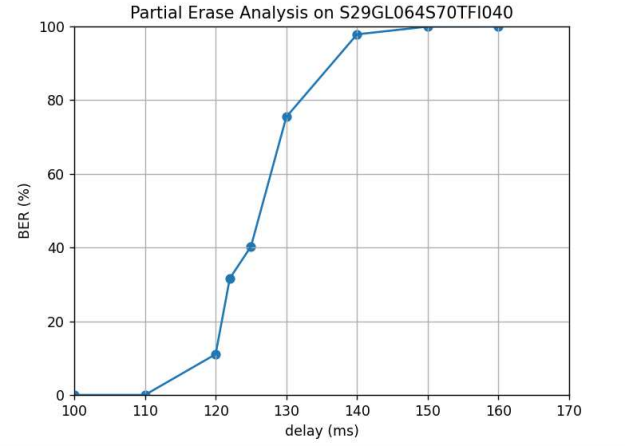


Figure 5.10: Partial erase time vs Bit Error Rate

The RBER% vs ($t_e{}^{\mathrm{d}}$) plot is given in Figure 5.10. At around 125 ms, the RBER is approximately 50%. However, this value cannot be arbitrarily chosen as the optimal ($t_e{}^{\mathrm{d}}$) value for data recovery because of the variation in erase time with cycling and baking. As demonstrated in [28, 40, 99, 100], the cycling and baking shift the program and erase voltage distributions and thus significantly varies the ($t_e{}^{\mathrm{d}}$) value that yields image recovery. Therefore, unlike ($t_p{}^{\mathrm{d}}$), ($t_e{}^{\mathrm{d}}$) is found by repeating the data recovery procedure given in Algorithm 4.

Algorithm 4 Partial Erase for Data Recovery

1: Write all-0s to `Sector1` and `Sector2` *(sanitization)*
2: Issue sector erase command to `Sector1` and `Sector2`
3: Partial erase delay $t_e^{\mathrm{d}} = 80$ ms
4: Send `RESET` command
5: Read data from `Sector1` and `Sector2`

The implementation of data recovery through partial erase is given in Algorithm 4. In order to evaluate the effect of number of P/E cycles on image recovery, the image is written to a fresh sector

and a 10,000 P/E cycled sector. After the device is baked at 150°C for 1 hour, which simulates the data retention for approximately 8 years, the sector is "scrubbed" [61] by overwriting all zeroes. Therefore, now all the cells in the sector are fully-programmed. A sector erase command is issued and after the specified delay, which in the initial case is 80ms, a hardware $RESET$ is issued. This involves writing a logic LOW pulse at the $RESET\#$ pin. This resets the device and thus the ongoing erase operation is terminated. The partially erased sector is read to verify the data recovery.

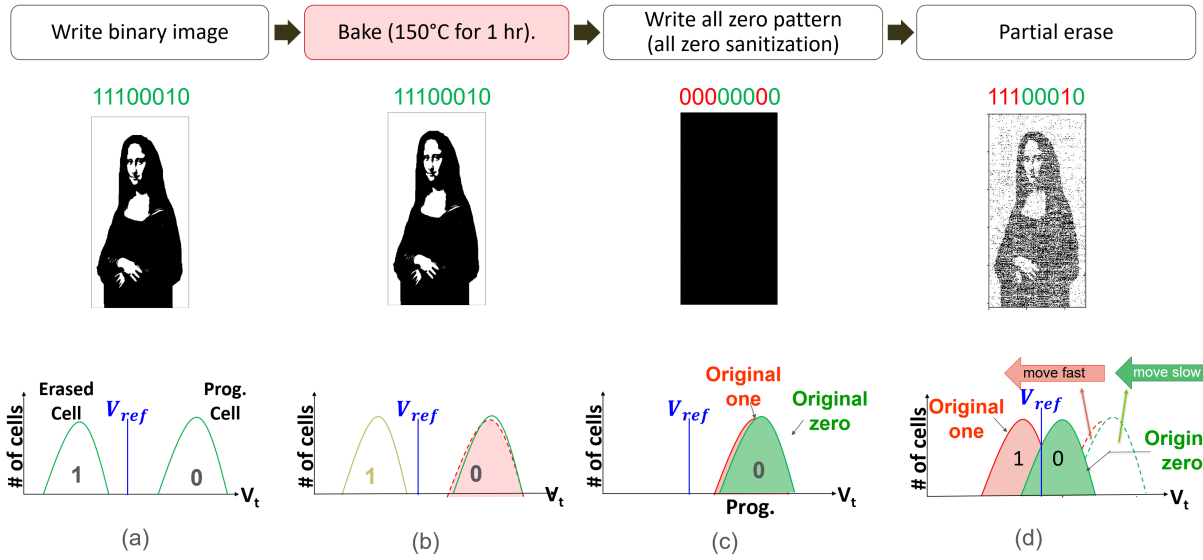


Figure 5.11: Image recovery through partial erase operation

A device level explanation for this phenomenon in terms of threshold voltage (V_t) is provided in Figure 5.11. The threshold distribution during programming the image is given in Figure 5.11a. Programmed bits, which are represented with black color in the binary image, have higher voltage threshold and the erased bits (white) have lower threshold. After several years, a change in a V_t levels of these cells may occur, which is obtained here using the temperature accelerated data retention or baking. However, this shift is not drastic and the programmed distribution is still above V_{ref} and erase distribution is still below V_{ref}. Therefore, the image is same as the original image after baking, as can be seen in Figure 5.11b. However, due to the nitride layer reliability issue

involved in the charge trap cells [99], a mismatch in the erase duration of original zero and original one cells occur, which is explained next in this section. Both original zero and original one cell are made 0 through all zero sanitization, as shown in Figure 5.11c.

The final step involved in image recovery is partial erase. The partial erase in terms of V_t level is bringing the voltage distributions of "original 0" and "original 1" towards (V_{ref}) in a controlled step-by step manner. The V_t levels above and below V_{ref} are read as 0 and 1 respectively. Therefore, the green and red distributions are brought between V_{ref} through partial erase as shown in Figure 5.11d. It is found that there is now a mismatch in the erase time of original zero and original one. Original zero, or the programmed cells in the original image are now slower to erase compared to the original one or erased cells in the original image. Therefore, the original one distribution (red) moves faster to the V_{ref} than the original zero (green). When the erase operation is terminated at the point of original one < V_{ref} < original zero, the image is recovered. Because, now most of the original ones are read as "1" and original zeroes as "0", closely matching the original image which was baked.

The detailed root cause of analysis of this discovered phenomenon of image recovery through partial program and partial erase, attributed to the nitride layer reliability, is provided in Section 5.7.

5.6 Permanent Imprinting

The data recoverability from MirrorBit™ is a repeatable phenomenon, which points towards a permanent imprinting of the data stored in the flash device for longer time. The data is recoverable after multiple P/E cycles after sanitization. At least a 100 P/E cycles were performed on the baked chips of Infineon S29GL064S70TFI040. However, the partial erase delay at which the data recovery was possible keeps decreasing with certain P/E cycles.

The experimental flow for validating permanent imprinting property of MirrorBit™ NOR flash is provided in Figure 5.12. The imprint state refers to the condition of the flash memory after it has undergone a controlled thermal aging process. In this process, a known image is first programmed

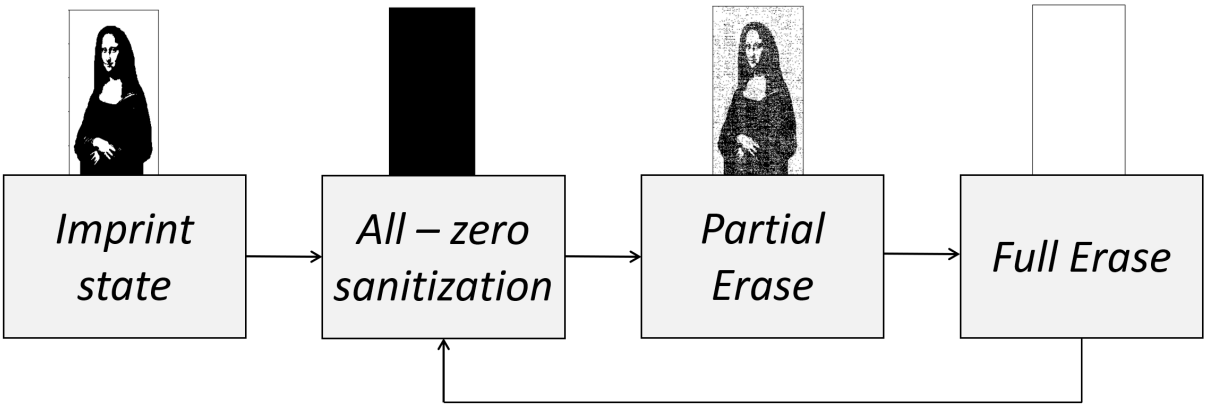


Figure 5.12: Block diagram of permanent imprinting validation

into the flash memory under standard operating conditions. The device is then subjected to high temperature exposure, commonly referred to as baking, in order to accelerate the effects of data retention and simulate extended use. After this process, the contents of the flash memory are read back for analysis. In our experiment, the state of the data retrieved after the baking process is referred to as the imprint state.

After the bake and read operation, the next step in validating permanent imprinting involves applying a sanitization technique. In this experiment, we employ an all-zero sanitization method or "scrubbing" [61], where a sector that previously stored the baked image is overwritten with all zeros. Following this, a partial erase is performed, which involves premature termination of an erase operation. This incomplete erase facilitates the partial recovery of the original image data.

Subsequently, a full erase command is issued to completely erase the sector, restoring it to a state of all ones. The all-zero sanitization is then applied again, and this cycle of all-zero sanitization, partial erase and full erase is repeated for multiple iterations.

It was observed that even after multiple iterations, the original image remained partially recoverable. Figure 5.13 shows the bit accuracy (%) plotted against number of iterations of the data recovery experiment provided in the block diagram Figure 5.12. It can be inferred from the plot that the bit accuracy decreases after the first iteration and remains stable, with the image still re-

Figure 5.13: Plot of number of iterations vs bit accuracy(%) to illustrate permanent imprinting effect

coverable. This persistent recoverability strongly supports the hypothesis of permanent imprinting behavior in MirrorBit™ NOR flash memory, highlighting a significant data remanence risk.

Figure 5.18 is obtained with repeated all-zero programming and partial erase on the same cycled and baked sector. It can be observed that the image is recoverable without deteriorating its quality at the optimal $t_e{}^{\mathrm{d}}$ value. This process was repeated multiple time and a consistent result was obtained.

5.7 Root Cause Analysis

In a cycled sector, electrons get deeply trapped in the nitride layer as shown in Figure 5.14. This results in a wider electron distribution with the tail representing the trapped electrons [99, 101].

However, a fresh sector has localized electron distribution due to less or no trapped electrons. Consequently, the hole distribution remains localized [38] in fresh as well as cycled cells, as can be seen in Figure 5.15.

The erase time increases in cycled cells due to the charge distribution mismatch of electrons and holes. The holes injected to the charge pockets do not migrate immediately to recombine with the trapped electrons [38, 99, 101] due to the insulating property of nitride layer. Therefore,

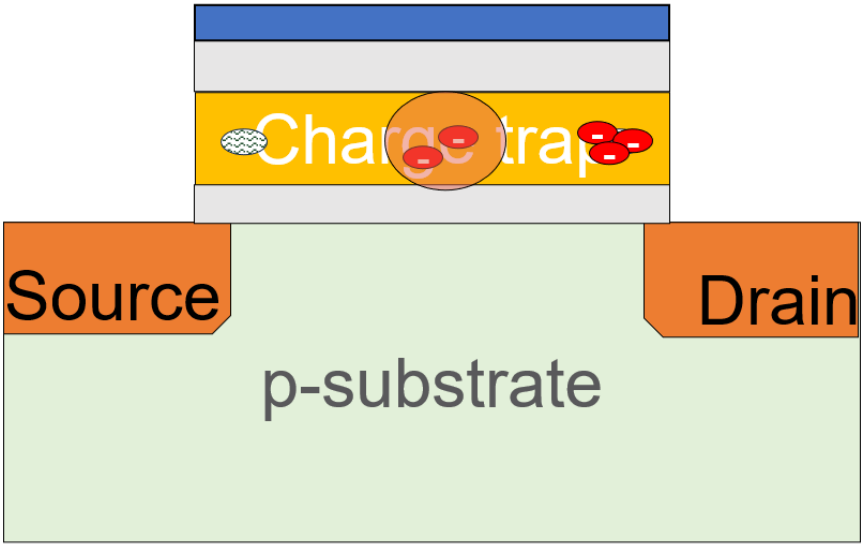


Figure 5.14: Cycling induced electron traps in Nitride layer

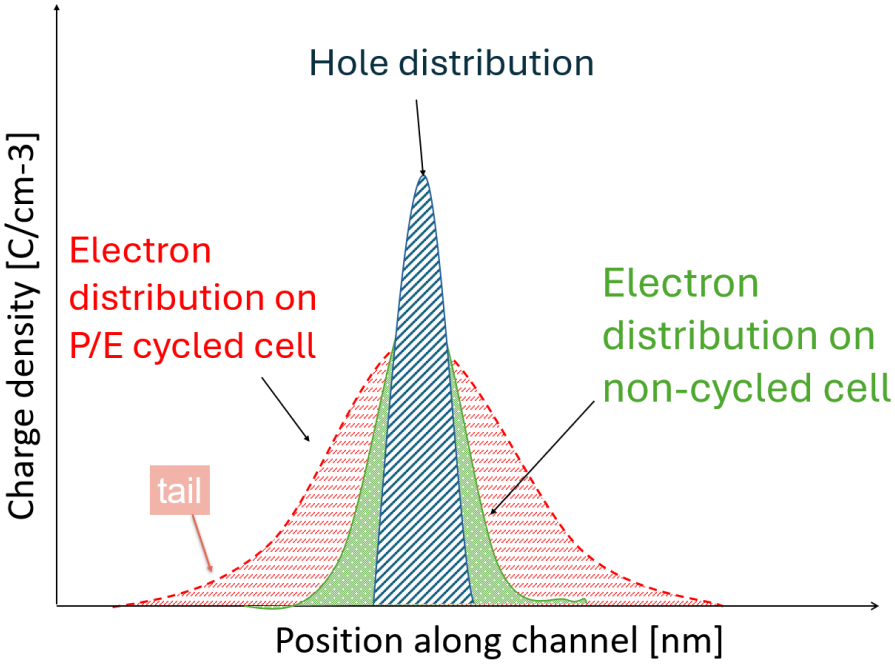


Figure 5.15: Electron-hole distribution in Nitride layer

more holes are required to neutralize the charge in the cycled cell to accomplish erase mechanism, thereby increasing the erase time.

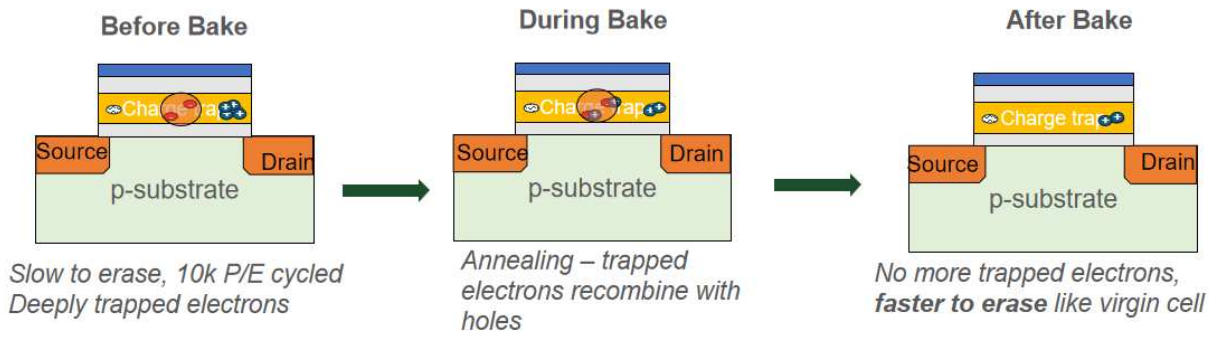


Figure 5.16: Root cause analysis of baking effect in an erased cell

The behavior of an erased cell under thermal stress can be understood in three stages: before bake, during bake, and after bake. These stages and the associated charge dynamics are illustrated in Figure 5.16. Before baking, the nitride layer of the erased cell contains a mixture of deeply trapped electrons due to 10k P/E cycles, and injected holes resulting from the BTBT hot hole injection erase mechanism. These charges are spatially distributed within the nitride, with electrons and holes occupying different localized trap states. During baking, thermally activated lateral redistribution of holes occurs through the Poole–Frenkel conduction mechanism, allowing holes to laterally migrate and recombine with nearby trapped electrons [102–104]. This recombination process progressively neutralizes the stored charge. After baking, the net trapped charge in the nitride is significantly reduced or eliminated, and the charge distribution becomes more localized and uniform—closely resembling that of a virgin cell. As a result, the erase time of the cell decreases substantially after baking and its timing characteristics resemble to that of a fresh cell. Conversely, the programmed cell holds trapped electrons along with the programmed electrons and doesn't hold any holes. Since there is very less or no holes to recombine with the trapped electrons, the charge dynamics of this cells remains the same throughout the three stages of baking experiment. Therefore, the erase time characteristics of this cell remains the same even after baking.

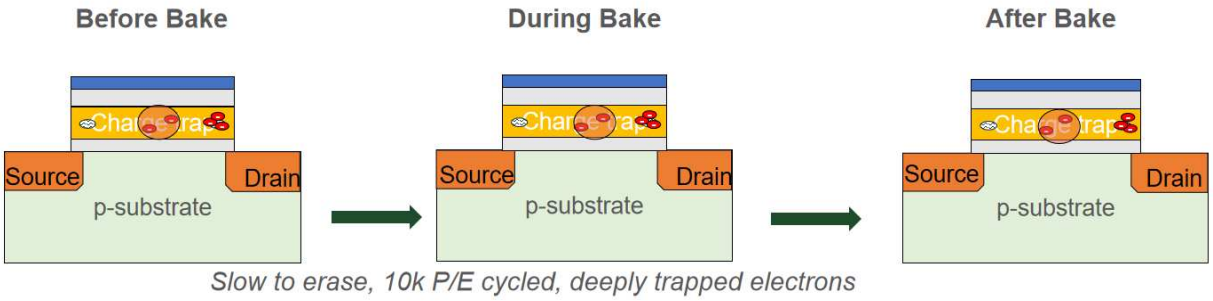


Figure 5.17: Root cause analysis of baking effect in a programmed cell

Since the erased cell (original one) and programmed cell (original zero) now has different erase time characteristics, they are erased fully erased at different durations. This spatial difference is exploited during partial erase to recover image from a MirrorBit™ flash.

From our experimental evaluation, it can be inferred that most of the original ones or erased cells are fully erased by 80 ms, which corresponds to the point where the maximum bit accuracy% is observed. However, most of the original zeros remain either not erased or only partially erased at this time, allowing partial recovery of the original image.

5.7.1 Factors Influencing Data Recovery

The data recovery results discussed earlier in this chapter are obtained by keeping the parameters such as the bake time, partial erase delay ($t_e{}^{\mathrm{d}}$) and number of P/E cycles on a sector constant. The quality of the image recovered can be optimized by fine-tuning these parameters. This section explores the dependency of the data recoverability from a sanitized NOR flash with duration of bake, number of P/E cycles and the delay at which partial erase is performed.

Partial Erase Delay ($t_e{}^{\mathrm{d}}$)

Figure 5.18 illustrates the relationship between $t_e{}^{\mathrm{d}}$ and bit accuracy%. The recovered image is also shown against the data points in the plot. The curve peaks at $t_e{}^{\mathrm{d}}$ is 75ms and the recovered image exhibits the best quality at this point. As $t_e{}^{\mathrm{d}}$ increases beyond this point, the bit accuracy% decreases and the image fades to an all-white image, which represents complete erase. This proves

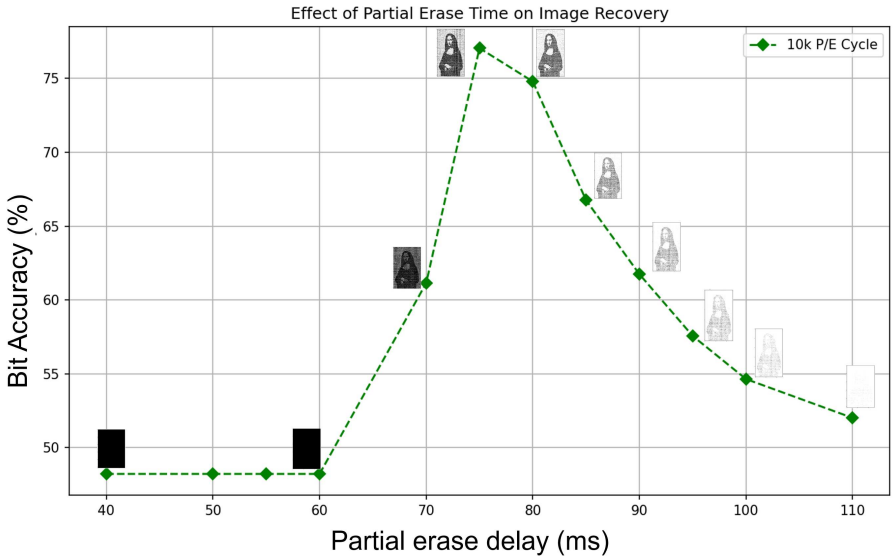


Figure 5.18: Effect of Partial Erase Delay on Data Recovery in S29GL064S70TFI040

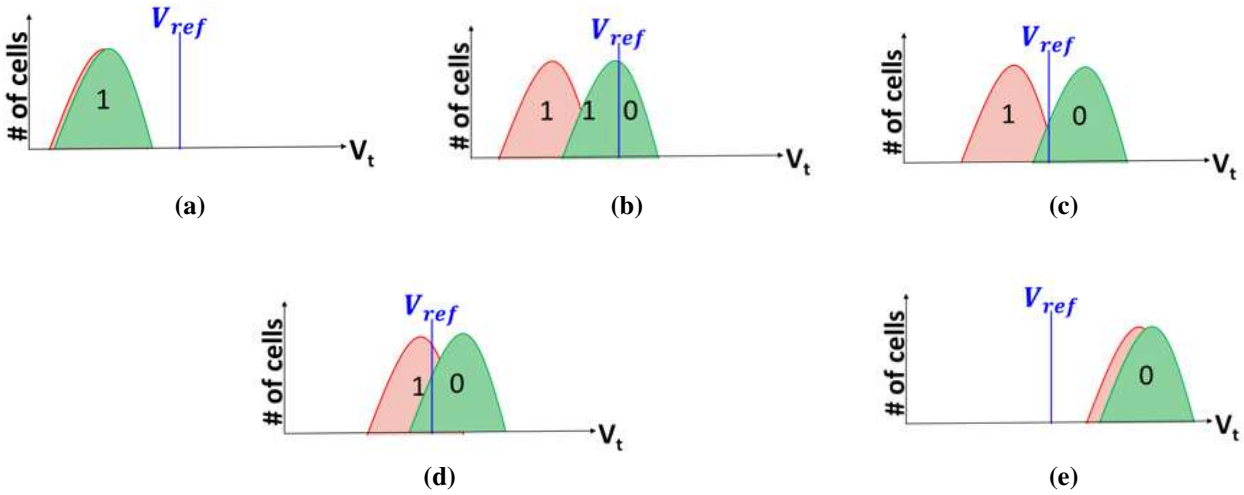


Figure 5.19: Threshold voltage distribution in a 10k P/E cycled and baked device at varying partial erase delay ($t_e{}^{\mathrm{d}}$) (a) 60 ms (b) 70 ms (c) 80 ms (d) 90 ms (e) 110 ms

the dependence of partial erase delay on image recovery and the importance of choosing the right $t_e{}^{\mathrm{d}}$ value for accurate recovery.

P/E Cycles

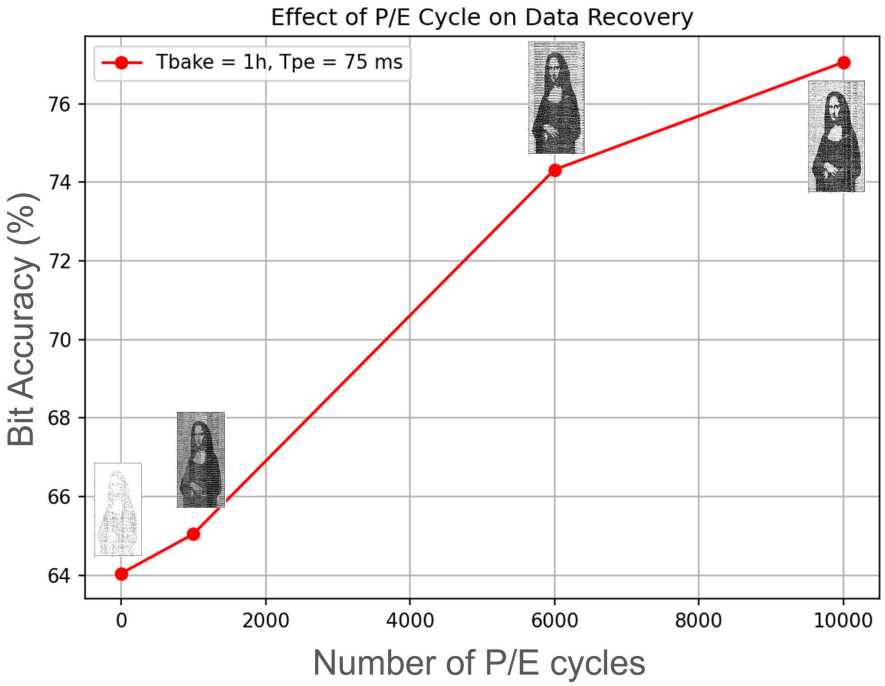


Figure 5.20: Effect of number of P/E Cycles on Data Recovery in S29GL064S70TFI040

The image recovery can be achieved on a cycled and non-cyled sectors. However, with cycling the image recovery is more pronounce and loud, as can be seen from the Figure 5.20. From the plot it is evident that bit accuracy % of recovered image increases with increase in number of P/E cycles performed on a sector.

The reason for improved image recovery with number of P/E cycles is due to the volume of trapped electrons in the nitride layer. Figure 5.21 shows the illustration of number of electron traps in a MirrorBit™ flash for different number of P/E cycles. The number of electrons trapped in the nitride layer increases with cycling [99–101]. More the number of trapped electrons, more number of holes are required to neutralize the charge and therefore, the erase duration keeps on increases. This aligns with the erase time characteristics we evaluated in Chapter 3.

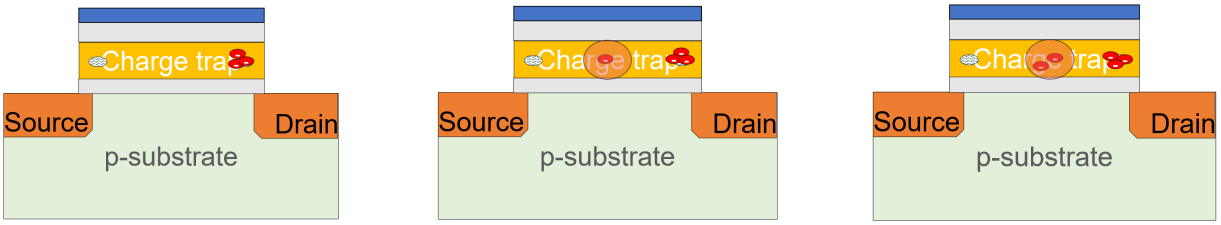


Figure 5.21: Representation of trapped electrons in the nitride layer for different P/E cycled cells (a) fresh cell (b) 1k P/E cycles (c) 10k P/E cycles

Bake Time

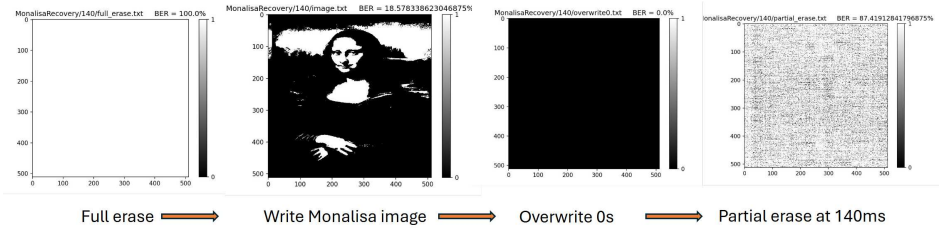


Figure 5.22: Partial erase operation on a sanitized sector without bake

The image recovery is achieved only in a baked device. The same procedure applied on a non-baked, but cycled chip yield no traces of the programmed image after recovery steps. Figure 5.22 shows the results of image recovery process applied on a 10k P/E cycled, sanitized, non-baked sector. It can be observed that the partial erase data shows no traces of the original image that was written before baking. This means that data retention or baking is an important factor for image recovery.

Table 5.1: Bake time and equivalent retention time

Bake Time (min)	Retention Time (years)
10	1.4
30	4.3
60	8.7

This inference can be explained in terms of the device physics using the nitride layer reliability. The recombination of holes with the trapped electrons take place only during data retention. Without baking, all cells - programmed and erased - are having the same mismatched electron-hole charge distribution. Therefore, both programmed and erased cells take similar time to erase. Hence, image recovery is not possible in a non-baked device due to the absence of spatial difference in the original programmed and erased cells.

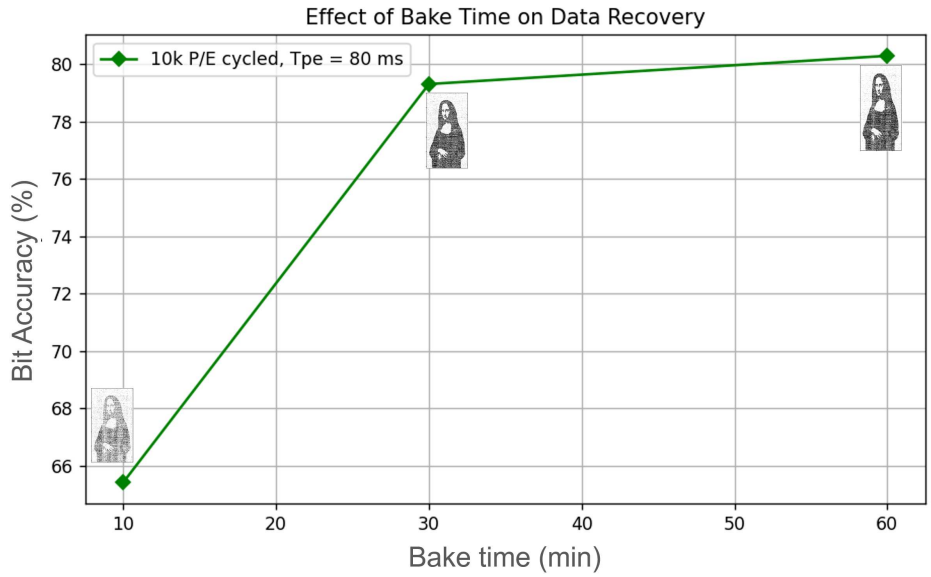


Figure 5.23: Effect of Bake Time on Data Recovery in S29GL064S70TFI040

Figure 5.23 shows the dependence of bake time with the bit accuracy % of recovered image. It is evident from the plot that bit accuracy % increases with increase in bake time. The equivalent retention duration of the bake time is provided in Table 5.1, calculated using the Arrhenius equation [97].

5.8 Performance Comparison

The data recovery results presented in this section correspond to the Infineon NOR flash memory device with part number S29GL064S70TFI040, which is based on a MirrorBit™ flash technology. Identical experiments were conducted on three additional NOR flash chips that use different

cell architectures—specifically, conventional floating gate (FG) and split-gate structures. These devices were subjected to identical experimental conditions, in which a known image was programmed to a sector and the device was baked at 150°C for one hour, followed by a sanitization and partial erase procedures as outlined in earlier sections.

However, no meaningful data recovery was achieved from any of the three FG based devices. Despite following the same sequence of baking, partial erase, and data extraction, these chips exhibited only random or fully erased sectors, with no evident traces of the original image. This contrast in behavior highlights the unique susceptibility of MirrorBit™ architecture to retain residual data through long term charge trapping, even after repeated sanitization, clearly indicating its vulnerability to data remanence during extended retention periods.

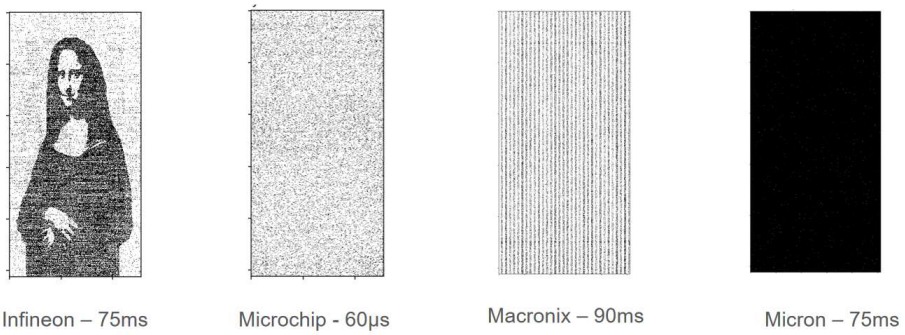


Figure 5.24: Image recovery results for different NOR flash memory devices

Figure 5.24 visually compares the image recovery outcomes across the four tested flash memory devices. As shown, only the Infineon MirrorBit™-based chip demonstrates successful image recovery, while the others display randomly erased sectors lacking any visible patterns of the originally stored image.

These results support the conclusion that the data recovery behavior observed in this study is specific to MirrorBit™-based NOR flash devices, at least under the tested thermal conditions of 150°C for 1 hour. This supports the hypothesis that permanent imprinting is influenced by the localized charge-trap structure of MirrorBit™ technology, making it more susceptible to residual charge retention even after repeated sanitization attempts.

5.9 Conclusion

In this chapter, the data remanence in a sanitized MirrorBit™ NOR flash is covered in detail. This research is motivated by the emerging popularity of NOR flash in applications such as automotive industry, consumer electronics and IoT devices for home and industry. Moreover, the contribution of these electronic devices to e-waste creates the risk of unauthorized access of these devices and potential data theft from the storage mediums in them. Since NOR flash is a popular storage medium for storing sensitive information in electronic devices, the paramount importance of this research is justified in this chapter. Furthermore, recent research on data remanence and associated sanitization techniques is reviewed to provide a comprehensive understanding of the current state of the field. This chapter demonstrates that residual data is recoverable on a baked NOR flash device. The data recovery is successful in both cycled and non-cycled sectors. The bit accuracy for recovered image increases with the bake time and number of P/E cycles performed on the sector, with 79.24% the best accuracy value. The optimal partial erase delay, ${t_e}^{\mathrm{d}}$ value obtained is 75ms. Additionally, it was found that image recovery is possible even with partial program operation. However, the accuracy of recovered image is lesser than partial-erase recovered image. The image recovery through partial program was best at a ${t_p}^{\mathrm{d}}$ of 40 µs. Moreover, it was observed that the image recovery is repeatable, implying towards a permanent change in the flash device during baking. Finally, the image recovery was found to be possible only on MirrorBit™ NOR flash and not on any of the FG-type devices for an experimental condition of bake at 150°C for 1 hour.

Chapter 6

Conclusion and Future Works

6.1 Conclusion

In this thesis, we presented a comprehensive analysis of the internal structure and data remanence behavior of MirrorBit™ NOR flash memory. Structural evaluation across various row and column configurations revealed that a 256×128 word array most accurately reflects the physical organization within a sector. This configuration uniquely exposes spatial asymmetries in program timing, effectively distinguishing between fast- and slow-programming regions. In contrast, other configurations either produced redundant information or failed to capture meaningful spatial variation. Such spatial programmability is characteristic of MirrorBit™ technology and is absent in conventional floating-gate or split-gate flash devices, where program latency is spatially random and uncorrelated.

Additionally, experiments on data remanence demonstrated that previously sanitized MirrorBit™ sectors exhibit partial data recovery when subjected to thermal baking. Notably, data remnants could be recovered from both cycled and non-cycled sectors, with bit-level image reconstruction accuracy improving as a function of bake duration and prior program/erase cycling. The highest observed recovery accuracy reached 79.24%, with an optimal partial erase delay (t_e^{d}) of 75 ms. Furthermore, partial program operations were also shown to allow recovery, albeit with lower accuracy, peaking at a partial program delay (t_p^{d}) of 40 µs. The repeatability of recovered images across baking trials indicates the likelihood of permanent structural changes within the device induced by thermal exposure.

Importantly, such recoverability was consistently observed only in MirrorBit™ NOR flash, with no corresponding data retrieval from floating-gate-type devices under identical experimental conditions (bake at 150°C for one hour). These findings underscore the architectural regularity and physical programmability of MirrorBit™ memory and highlight potential vulnerabilities to

data remanence, while also laying the groundwork for future studies in array-level characterization and adaptive memory system design.

6.2 Future Works

This work has been exclusively demonstrated on MirrorBit™ NOR flash memory. Given its unique charge-trap storage mechanism and spatial programming asymmetry, a natural extension would be to investigate whether similar behavior can be observed in other charge-trap memory technologies which are mentioned in Table 2.1. Future studies can explore data remanence and program-time asymmetries in alternative CT-based embedded flash memories such as SONOS, MONOS, or nanocrystal-based devices, to assess the generality of the structural inference and partial recovery techniques presented in this study.

While floating-gate NAND flash is known to exhibit data remanence effects, FG NOR flash did not demonstrate recoverable data under the experimental bake conditions employed here. A potential direction for future work involves systematically evaluating the bake temperature, duration, and cycling conditions to determine whether any residual data retention characteristics can be exploited in FG NOR devices.

Furthermore, partial program-based recovery showed reduced accuracy compared to partial erase-based recovery. To improve the efficacy of partial programming as a side-channel for data recovery, future work could focus on fine-tuning the partial program delay (${t_p}^{\mathrm{d}}$) in an adaptive manner. Specifically, incorporating knowledge of the nominal program time of individual words into the selection of dynamic ${t_p}^{\mathrm{d}}$ values for each words could enhance image reconstruction accuracy. Such adaptive techniques may enable more precise targeting of partially programmed states and provide further insight into the temporal characteristics of the programming operation.

This work can also be extended to the field of digital forensics, particularly in the context of autonomous driving systems, where NOR flash memory is commonly used to store critical information such as program code, system logs, sensor data, and calibration parameters. In the event of a vehicle malfunction or crash, recovering data from NOR flash can be vital for post-

incident analysis and determining root causes. The data recovery technique demonstrated in this research shows that even after logical erasure, remnants of data may persist due to the non-volatile nature of NOR flash and its charge-trapping mechanism.

Given that automotive systems are often exposed to elevated temperatures during operation, similar to the thermal stress used in the baking experiments, there is a reasonable assumption that such conditions could preserve charge states in a way that enables partial or full recovery of erased data. This insight opens up the possibility of applying controlled data recovery methods as a forensic tool, helping investigators extract meaningful information from devices presumed to be sanitized. Consequently, the methodology developed in this thesis could serve as a valuable asset for crash analysis, debugging safety-critical failures, or validating the software state of autonomous platforms post-incident.

Bibliography

[1] Global Market Insights. Online: https://www.gminsights.com/industry-analysis/nand-flash-market, 2025.

[2] market.us. Online: https://market.us/report/nor-flash-market/.

[3] Mordor Intelligence. Online: https://www.mordorintelligence.com/industry-reports/global-nor-flash-memory-market.

[4] Macronix. Online: https://www.eetimes.com/breaking-through-capacity-bottlenecks-macronix-leads-the-world-with-innovative-3d-nor-flash-technology/, 2025.

[5] Macronix. Online: https://www.electronicdesign.com/technologies/industrial/article/55288650/macronix-international-flash-memory-enters-a-new-age-maximizing-capacity-with-3d-nor, 2025.

[6] Macronix. Online: https://techovedas.com/macronix-unveils-worlds-first-3d-nor-flash-faster-stronger-smarter/, 2025.

[7] Giovanni Campardo, Rino Micheloni, David Novosel, et al. *VLSI-design of non-volatile memories*. Springer, 2005.

[8] L Crippa, R Micheloni, I Motta, and M Sangalli. Nonvolatile memories: Nor vs. nand architectures. In *Memories in Wireless Systems*, pages 29–53. Springer, 2008.

[9] F. Masuoka, M. Asano, H. Iwahashi, T. Komuro, and S. Tanaka. A new flash e2prom cell using triple polysilicon technology. In *1984 International Electron Devices Meeting*, pages 464–467, 1984.

[10] R Micheloni, A Marelli, R Ravasio, R Micheloni, A Marelli, and R Ravasio. Nor flash memories. *Error Correction Codes for Non-Volatile Memories*, pages 61–83, 2008.

[11] R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti. Introduction to flash memory. *Proceedings of the IEEE*, 91(4):489–502, 2003.

[12] F. Masuoka, M. Momodomi, Y. Iwata, and R. Shirota. New ultra high density eprom and flash eeprom with nand structure cell. In *1987 International Electron Devices Meeting*, pages 552–555, Dec 1987.

[13] R. Micheloni, A. Marelli, and S. Commodaro. *NAND overview: from memory to systems*, pages 19–53. Springer Netherlands, 2010.

[14] Krishna Parat and Chuck Dennison. A floating gate based 3d nand technology with cmos under array. In *2015 IEEE International Electron Devices Meeting (IEDM)*, pages 3.3.1–3.3.4, 2015.

[15] Christian Monzio Compagnoni, Akira Goda, Alessandro S. Spinelli, Peter Feeley, Andrea L. Lacaita, and Angelo Visconti. Reviewing the evolution of the nand flash technology. *Proceedings of the IEEE*, 105(9):1609–1633, 2017.

[16] Rino Micheloni, Luca Crippa, and Alessia Marelli. *Inside NAND flash memories*. Springer Science & Business Media, 2010.

[17] Hideto Hidaka. *Embedded Flash Memory for Embedded Systems: Technology, Design for Sub-systems, and Innovations*. Springer Publishing Company, Incorporated, 1st edition, 2017.

[18] Hideto Hidaka. Evolution of embedded flash memory technology for mcu. In *2011 IEEE International Conference on IC Design & Technology*, pages 1–4, 2011.

[19] Kelly Baker. Embedded nonvolatile memory technology. In *2009 IEEE International Conference on IC Design and Technology*, pages 185–189, 2009.

[20] P. Pavan, R. Bez, P. Olivo, and E. Zanoni. Flash memory cells-an overview. *Proceedings of the IEEE*, 85(8):1248–1271, 1997.

[21] Sohrab Kianian, Amitay Levi, Dana Lee, and Yaw-Wen Hu. A novel 3 volts-only, small sector erase, high density flash e/sup 2/prom. In *Proceedings of 1994 VLSI Technology Symposium*, pages 71–72. IEEE, 1994.

[22] Sohrab Kianian, Amitay Levi, Dana Lee, and Yaw-Wen Hu. A novel 3 volts-only, small sector erase, high density flash e/sup 2/prom. In *Proceedings of 1994 VLSI Technology Symposium*, pages 71–72. IEEE, 1994.

[23] Yong Kyu Lee, Jung Ho Moon, Young Ho Kim, Myung-Jo Chun, Soung-Youb Ha, Sunggon Choi, Hyunkye Yoo, Heeseog Jeon, Jaemin Yu, Jeong-Uk Han, et al. 2t-fn envm with 90 nm logic process for smart card. In *2008 Joint Non-Volatile Semiconductor Memory Workshop and International Conference on Memory Technology and Design*, pages 26–27. IEEE, 2008.

[24] Yong Kyu Lee, Jung Ho Moon, Young Ho Kim, Myung-Jo Chun, Soung-Youb Ha, Sunggon Choi, Hyunkye Yoo, Heeseog Jeon, Jaemin Yu, Jeong-Uk Han, et al. 2t-fn envm with 90 nm logic process for smart card. In *2008 Joint Non-Volatile Semiconductor Memory Workshop and International Conference on Memory Technology and Design*, pages 26–27. IEEE, 2008.

[25] Danny Shum, LQ Luo, YT Chow, F Zhang, JB Tan, XS Cai, ZQ Teo, N Do, JH Kim, P Ghazavi, et al. Functionality demonstration of a high-density 1.1 v self-aligned split-gate nvm cell embedded into lp 40 nm cmos for automotive and smart card applications. In *2015 IEEE International Memory Workshop (IMW)*, pages 1–4. IEEE, 2015.

[26] Boaz Eitan and Dov Frohman-Bentchkowsky. Hot-electron injection into the oxide in n-channel mos devices. *IEEE transactions on electron devices*, 28(3):328–340, 2005.

[27] Simon Tam, Ping-Keung Ko, and Chenming Hu. Lucky-electron model of channel hot-electron injection in mosfet's. *IEEE Transactions on Electron Devices*, 31(9):1116–1125, 1984.

[28] I Bloom, A Givant, E Lusky, A Shappir, M Janai, and B Eitan. Nrom/mirrorbit® technology for non-volatile memories. *Reference Module in Materials Science and Materials Engineering*, 2016.

[29] M Lenzlinger and EH Snow. Fowler-nordheim tunneling into thermally grown sio2. *Journal of Applied physics*, 40(1):278–283, 1969.

[30] M Kamiya, Y Kojima, Y Kato, K Tanaka, and Y Hayashi. Eprom cell with high gate injection efficiency. In *1982 International Electron Devices Meeting*, pages 741–744. IEEE, 1982.

[31] Kuo-Tung Chang, Wei-Ming Chen, Craig Swift, Jack M Higman, Wayne M Paulson, and Ko-Min Chang. A new sonos memory using source-side injection for programming. *IEEE Electron Device Letters*, 19(7):253–255, 1998.

[32] Bomy Chen. Highly reliable superflash® embedded memory scaling for low power soc. In *2007 International Symposium on VLSI Technology, Systems and Applications (VLSI-TSA)*, pages 1–2. IEEE, 2007.

[33] Ching-Shi Jenq. Single transistor non-volatile electrically alterable semiconductor memory device with a re-crystallized floating gate, November 19 1991. US Patent 5,067,108.

[34] Wei-Ming Chen, Kuo-tung Chang, et al. A novel flash memory device with s plit gate source side injection and 0n0 charge storage stack (spin). In *1997 Symposium on VLSI Technology*, pages 63–64. IEEE, 1997.

[35] Vei-Han Chan and D.K.Y. Liu. An enhanced erase mechanism during channel fowler-nordheim tunneling in flash eprom memory devices. *IEEE Electron Device Letters*, 20(3):140–142, 1999.

[36] S-I Minami and Yoshiaki Kamigaki. A novel monos nonvolatile memory device ensuring 10-year data retention after 10/sup 7/erase/write cycles. *IEEE transactions on Electron Devices*, 40(11):2011–2017, 2002.

[37] Hai-Ming Lee, Swee-Tuck Woo, Hsin-Ming Chen, Rick Shen, Chai-De Wang, Liang Choo Hsia, and CC-H Hsu. Neoflash-true logic single poly flash memory technology. In *2006 21st IEEE Non-Volatile Semiconductor Memory Workshop*, pages 15–16. IEEE, 2006.

[38] A. Padovani, Luca Larcher, and Paolo Pavan. Hole distributions in erased nrom devices: Profiling method and effects on reliability. *IEEE Transactions on Electron Devices*, 55:343–349, 2008.

[39] Meir Janai, Boaz Eitan, Assaf Shappir, Eli Lusky, Ilan Bloom, and Guy Cohen. Data retention reliability model of nrom nonvolatile memory products. *IEEE Transactions on Device and Materials Reliability*, 4(3):404–415, 2004.

[40] E. Lusky, Y. Shacham-Diamand, A. Shappir, I. Bloom, G. Cohen, and B. Eitan. Retention loss characteristics of localized charge-trapping devices. In *2004 IEEE International Reliability Physics Symposium. Proceedings*, pages 527–530, 2004.

[41] T. Ohnakado, K. Mitsunaga, M. Nunoshita, H. Onoda, K. Sakakibara, N. Tsuji, N. Ajika, M. Hatanaka, and H. Miyoshi. Novel electron injection method using band-to-band tunneling induced hot electrons (bbhe) for flash memory with a p-channel cell. In *Proceedings of International Electron Devices Meeting*, pages 279–282, 1995.

[42] Online: https://www.sec.gov/Archives/edgar/data/1297627/000117891307002125/exhibit_99-3.htm.

[43] Online: https://www.sec.gov/Archives/edgar/data/1297627/000117891308000691/exhibit_99-1.htm.

[44] D. Ielmini, A.S. Spinelli, A.L. Lacaita, and M.J. van Duuren. A comparative study of characterization techniques for oxide reliability in flash memories. *IEEE Transactions on Device and Materials Reliability*, 4(3):320–326, 2004.

[45] Yakov Roizin and Vladimir Gritsenko. Ono structures and oxynitrides in modern microelectronics: material science, characterization and application. *Dielectric Films for Advanced Microelectronics*, pages 251–295, 2007.

[46] Ilan Bloom, Eli Lusky, Assaf Shappir, Guy Cohen, Eduardo Maayan, Meir Janai, and Boaz Eitan. Nrom: Nitride-based nvm technology. In *Tutorial G, Science and Technology of Nonvolatile Memories, MRS Symposium*, 2006.

[47] Infineon Technologies. S29GL064S 64 Mb (8 MB) GL-S MIRRORBIT™ Flash Data Sheet, 2020. Datasheet for S29GL064S MirrorBit NOR Flash.

[48] Spansion Inc. Online: https://www.ciol.com/spansion-ships-industrys-65nm-nor-soln/.

[49] "allaboutcircuits". Online: https://www.allaboutcircuits.com/technical-articles/what-is-a-printed-circuit-board-pcb/.

[50] Online: https://flexpcb.org/pcb-creation-with-autodesk-eagle-step-by-step-tutorial/. Accessed: 2025-05-20.

[51] Michael Barr and Anthony Massa. *Programming embedded systems: with C and GNU development tools*. " O'Reilly Media, Inc.", 2006.

[52] Microchip Technology Inc. SST39VF1601C / SST39VF1602C Data Sheet – 16 Mbit (x16) Multi-Purpose Flash Plus, 2017. Datasheet for Microchip NOR Flash SST39VF160x series.

[53] Macronix International Co Ltd. MX29GL640E T/B, MX29GL640E H/L Data Sheet, 2018. Datasheet for Macronix MX29GL640E 64Mb NOR Flash.

[54] Micron Technology Inc. Micron Parallel NOR Flash Embedded Memory, 2019. Datasheet for Micron parallel NOR Flash products.

[55] Naoto Shibata and Atsushi Iwata. Initial programming and erase characteristics of flash eeprom cells. *IEEE Transactions on Electron Devices*, 50(3):793–799, 2003.

[56] Yiming Wang. Floating-gate memory cell technologies for high-density flash memories. *Microelectronics Reliability*, 46(3-4):421–427, 2006.

[57] R Rao and et al. Split-gate flash memory cell with self-aligned floating gate and control gate. *IEEE Transactions on Electron Devices*, 49(9):1647–1653, 2002.

[58] Jian Zhang et al. Reliability analysis of split-gate flash memory under program/erase stress. *IEEE Transactions on Device and Materials Reliability*, 10(2):190–196, 2010.

[59] Infineon Technologies. *S29GL064S 64 Mb (8M × 8-Bit/4M × 16-Bit) CMOS 3.0 Volt-only Page Mode Flash Memory*, 2017. Revision 14.

[60] Wikipedia. Online: https://en.wikipedia.org/wiki/Histogram.

[61] 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.

[62] market.us Scoop. Online: https://scoop.market.us/e-waste-statistics/, 2025.

[63] Giantec Semiconductor. Online: https://en.giantec-semi.com/blog/what-devices-use-nor-flash, 2025.

[64] Janine Schneider, Immanuel Lautner, Denise Moussa, Julian Wolf, Nicole Scheler, Felix Freiling, Jaap Haasnoot, Hans Henseler, Simon Malik, Holger Morgenstern, et al. In search of lost data: a study of flash sanitization practices. *arXiv preprint arXiv:2505.14067*, 2025.

[65] Janine Schneider, Aya Fukami, Immanuel Lautner, Maximilian Eichhorn, Denise Moussa, Julian Wolf, Nicole Scheler, Dominic Deuber, Felix Freiling, Jaap Haasnoot, et al. Poor sanitization practices and questionable digital evidence: a comprehensive study of scope and impact of recycled nand flash chips. *IEEE Transactions on Dependable and Secure Computing*, 2025.

[66] Aya Fukami, Francesco Regazzoni, and Zeno Geradts. Data sanitization on emmcs. In *Proceedings of the 28th Asia and South Pacific Design Automation Conference*, pages 455–460, 2023.

[67] Mahra Mohammed Alnaqbi, Shaikha Tareq Alblooshi, Elyazia Abdulla Alshamsi, Niyat Seghid, and Farkhund Iqbal. An empirical analysis of security and privacy issues associated with selling and purchasing of secondhand storage devices. In *2025 13th International Symposium on Digital Forensics and Security (ISDFS)*, pages 1–7. IEEE, 2025.

[68] Sanchari Das, AKM Salman Hosain, and Biswajit Debnath. A review of security threats from e-waste: Issues, challenges, and sustainability. *Development in E-waste Management*, pages 165–188, 2023.

[69] Amod Kumar, Manoj Malik, and Satya Jeet Singh. Flash based storage media secure sanitization using high temperature. In *2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*, pages 397–400. IEEE, 2018.

[70] Sergei Skorobogatov. Data remanence in flash memory devices. In *Proceedings of the 7th International Conference on Cryptographic Hardware and Embedded Systems*, pages 339–353, Berlin, Heidelberg, 2005.

[71] Abhilash Garg, Supriya Chakraborty, Manoj Malik, Devesh Kumar, Satyajeet Singh, and Manan Suri. Investigation of data deletion vulnerabilities in nand flash memory based storage. *arXiv preprint arXiv:2001.07424*, 2020.

[72] CNN. Online: https://www.cnn.com/2024/03/20/climate/electronic-waste-recycling-climate-un, 2024.

[73] United Nations Institute for Training and Research. Online: https://unitar.org/about/news-stories/press/global-e-waste-monitor-2024-electronic-waste-rising-five-times-faster-documented-e-waste-recycling, 2024.

[74] Creamer Media. Online: https://www.engineeringnews.co.za/article/e-waste-fastest-growing-stream-of-global-waste-needs-intervention-2024-04-19, 2024.

[75] CBS. Online: https://www.cbsnews.com/news/electronic-waste-ewaste-span-equator-growing/, 2024.

[76] MIT. Online: https://www.technologyreview.com/2024/10/28/1106316/ai-e-waste/, 2024.

[77] Peter Gutmann. Data remanence in semiconductor devices. In *USENIX Security Symposium*, 2001.

[78] M. Wei, L. M. Grupp, F. E. Spada, and S. Swanson. Reliably erasing data from flash-based solid state drives. In *Proceedings of the 9th USENIX Conference on File and Storage Technologies*, pages 8–8, Berkeley, CA, USA, 2011.

[79] Steven Swanson and Michael Wei. Safe: Fast, verifiable sanitization for ssds. 2011.

[80] M Abraham. Nand flash security. In *Special Pre-Conference Workshop on Flash Security*, 2010.

[81] L. Zuolo, C. Zambelli, R. Micheloni, and P. Olivo. Solid-state drives: Memory driven design methodologies for optimal performance. *Proceedings of the IEEE*, 105(9):1589–1608, Sep 2017.

[82] F. Chen, T. Zhang, and X. Zhang. Software support inside and outside solid-state devices for high performance and high efficiency. *Proceedings of the IEEE*, 105(9):1650–1665, Sep 2017.

[83] N. R. Mielke, R. E. Frickey, I. Kalastirsky, M. Quan, D. Ustinov, and V. J. Vasudevan. Reliability of solid-state drives based on nand flash memory. *Proceedings of the IEEE*, 105(9):1725–1750, Sep 2017.

[84] Johannes 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*, pages 63–64, New York, NY, USA, 2012.

[85] S. M. Diesburg, A. J. Long, M. Meyer, L. P. Tawalbeh, and A. C. Kocher. Trueerase: per-file secure deletion for the storage data path. In *Annual Computer Security Applications Conference (ACSAC)*, 2012.

[86] Kang Sun, Jaeyong Choi, Donghee Lee, and Sang Lyul Min. Models and design of an adaptive hybrid scheme for secure deletion of data in consumer electronics. *IEEE Transactions on Consumer Electronics*, 54, 2008.

[87] Johannes Reardon, Srdjan Capkun, and David Basin. Data node encrypted file system: Efficient secure deletion for flash. In *USENIX Security Symposium (USENIX Security 12)*, pages 333–348, 2012.

[88] Johannes Reardon, David Basin, and Srdjan Capkun. On secure data deletion. *IEEE Security & Privacy*, 12(3):37–44, May 2014.

[89] Johannes Reardon, David Basin, and Srdjan Capkun. Sok: Secure data deletion. In *2013 IEEE Symposium on Security and Privacy*, pages 301–315, 2013.

[90] Jaehoon Lee, Jaemyoung Heo, Yookun Cho, Jong Hyuk Hong, and Seung Yeob Shin. Secure deletion for nand flash file system. In *ACM Symposium on Applied Computing*, 2008.

[91] Shengjia Jia, Lei Xia, Binyi Chen, and Peng Liu. Deftl: Implementing plausibly deniable encryption in flash translation layer. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2217–2229, New York, NY, USA, 2017.

[92] Lifeng Yang, Tao Wei, Fengyuan Zhang, and Jian Ma. Sadus: Secure data deletion in user space for mobile devices. *Computers & Security*, 77:612–626, August 2018.

[93] Andrew Regenscheid, Larry Feldman, and Gregory Witte. Nist special publication 800-88, revision 1: Guidelines for media sanitization, 2015-02-05 2015.

[94] Thomas Ristenpart and Scott Yilek. When good randomness goes bad: Virtual machine reset vulnerabilities and hedging deployed cryptography. In *NDSS '10 (Network and Distributed System Security Symposium)*, 2010.

[95] Wei Wang, Chia-Lin Ho, Yu-Ting Chang, Tei-Wei Kuo, and Po-Chun 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, November 2018.

[96] Online: https://docs.opencv.org/3.4/d7/d4d/tutorial_py_thresholding.html. Accessed: 2025-05-27.

[97] David A. Baglee. Characteristics & reliability of 100Å oxides. In *22nd International Reliability Physics Symposium*, pages 152–155, 1984.

[98] Texas Instruments. Online: https://www.ti.com/lit/an/slaa392a/slaa392a.pdf.

[99] Meng Chuan Lee and Hin Yong Wong. Charge loss mechanisms of nitride-based charge trap flash memory devices. *IEEE Transactions on Electron Devices*, 60(10):3256–3264, 2013.

[100] Saied Tehrani, James Pak, Mark Randolph, Yu Sun, Sameer Haddad, Eduardo Maayan, and Yoram Betser. Advancement in charge-trap flash memory technology. In *2013 5th IEEE International Memory Workshop*, pages 9–12, 2013.

[101] Arnaud Furnemont, Maarten Rosmeulen, Koen van der Zanden, Jan Van Houdt, Kristin De Meyer, and Herman Maes. Root cause of charge loss in a nitride-based localized trapping memory cell. *IEEE Transactions on Electron Devices*, 54(6):1351–1359, 2007.

[102] PANG Huiqing, PAN Liyang, SUN Lei, WU Dong, and ZHU Jun. Lateral redistribution and interactive impacts of localized trapped charges during retention baking in sonos memory.

In *Extended abstracts of the... Conference on Solid State Devices and Materials*, volume 2006, pages 988–989, 2006.

[103] Alessandro Maconi, Antonio Arreghini, C Monzio Compagnoni, AS Spinelli, J Van Houdt, ANDREA LEONARDO Lacaita, et al. Impact of lateral charge migration on the retention performance of planar and 3d sonos devices. In *2011 Proceedings of the European Solid-State Device Research Conference (ESSDERC)*, pages 195–198. IEEE, 2011.

[104] Huiqing Pang, Liyang Pan, Lei Sun, Dong Wu, and Jun Zhu. Trapped charge distribution during the p/e cycling of sonos memory. In *2006 13th International Symposium on the Physical and Failure Analysis of Integrated Circuits*, pages 84–87. IEEE, 2006.