Università degli Studi di Ferrara

DOTTORATO DI RICERCA IN
"SCIENZE DELL’INGEGNERIA"

CICLO XXX

COORDINATORE Prof. Stefano Trillo

Memory-Driven Design Methodologies For Solid State Drives (SSDs)

Settore Scientifico Disciplinare ING-INF/01

Dottorando	**Tutore**
Dott. Micheloni Rino	Prof. Olivo Piero
(firma)	*(firma)*

Anni 2015/2017

Università degli Studi di Ferrara

DOTTORATO DI RICERCA IN
"SCIENZE DELL'INGEGNERIA"

CICLO XXX

COORDINATORE Prof. Stefano Trillo

Memory-Driven Design Methodologies For Solid State Drives (SSDs)

Settore Scientifico Disciplinare ING-INF/01

Dottorando	**Tutore**
Dott. Micheloni Rino	Prof. Olivo Piero
(firma)	*(firma)*

Anni 2015/2017

Acknowledgements

I want to thank Prof. Piero Olivo for his great support over the last two decades; it is always an immense pleasure to have a technical discussion with him and I sincerely appreciate the opportunity he gave me to work towards my PhD.

Special thanks to Dr. Zambelli for his continuous help (inside and outside the office). I definitely learned from him the difference between a technical report and a scientific paper. His tireless effort in being “scientific-correct” it is something that I will carry in my professional life.

Let me also thank the Microsemi team based in Vimercate for all the great inputs: Luca Crippa, Alessia Marelli, Antonio Aldarese, Salvatrice Scommegna, and Lorenzo Zuolo.

Writing a PhD thesis is a real challenge as there are so many details, diagrams, graphs, and numbers that it is very easy to have “bugs” (mistakes, errors), like in all engineering project. This is why I’m especially grateful to all the people who volunteered to review the chapters. Thank You All!!!

Rino Micheloni

Memoriae duplex virtus: facile percipere et fideliter continere.

(Quintiliano, Inst., 1, 3, 2, 33)

To the three women of my life:
my wife Sabrina,
and my daughters Laura and Greta

Table Of Contents

Introduction 1
1 Solid State Drive (SSD): a Non-Volatile Storage System 5
  1.1 Flash technology 6
  1.2 SSD’s block diagram 8
  1.3 Hybrid SSDs 13
2 Memory-driven design methodologies for high performance SSDs 15
  2.1 NAND Flash memory cells: basic operations and reliability 19
  2.2 The impact of ECC on SSD’s performances 26
  2.3 SSD controller design 37
    2.3.1 Efficient command management 39
    2.3.2 DRAM data caching 41
  2.4 Criteria for optimal host interface selection 46
  2.5 Research scenario opened by hardware-software co-design 48
    2.5.1 HB-FTL operations 49
    2.5.2 The Open-Channel architecture 50
3 3D NAND Flash memories 65
  3.1 3D Stack of Planar NAND Flash memories 67
  3.2 3D Charge Trap NAND Flash memories 68
  3.3 3D Floating Gate NAND Flash memories 78
  3.4 Advanced Layout Solutions 84
  3.5 Key challenges for 3D Flash development 90
    3.5.1 Number of layers 90
    3.5.2 Peripheral circuits under memory arrays 92
    3.5.3 Data retention 94
    3.5.4 3D Program disturb 95
  3.6 Future trend for 3D NAND Flash 97
4 Machine Learning applied to NAND Flash memories 105
  4.1 Data collection 106
  4.2 Data clustering results 108
  4.3 LDPC Code Rate optimization results 110

5 Impact of power-supply on the reliability of TLC memories............. 117
  5.1 Data collection 118
  5.2 Simulations of high voltage circuits 121
6 Uniform and Concentrated read disturb 133
  6.1 Data collection and analyses 134
  6.2 Implications of enterprise SSDs.. 137
7 RRAM-based SSDs 145
  7.1 All-RRAM SSD architecture 146
  7.2 Page size vs. queue depth... 147
  7.3 Design space exploration of All-RRAM SSDs............................ 156
8 MRAM-based NVRAM cards 165
  8.1 All-MRAM NVRAM cards 165
  8.2 Data correlation and simulation framework 167
  8.3 DRAM/Flash-based NVRAM vs. All-MRAM NVRAM............ 169
9 Modeling the reliability of Intra-Disk RAID solutions for SSDs........ 173
  9.1 Intra-disk RAID basic principles 175
  9.2 UBER ideal calculations using intra-disk RAID 176
    9.2.1 Inpact of the ECC correction strenght and stripe lenght ... 176
    9.2.2 Inpact of the SSD workload constraints 179
  9.3 Modeling intra-disk RAID failures... 183
    9.3.1 DMTC model.. 183
    9.3.2 Impact of DWPD and WA on data loss 185
    9.3.3 Impact of the channel failure rate on data loss 187
    9.3.4 Impact of the stripe length on cost and reliability 189
  Conclusions... 195
  Author's publications.. 199

Introduction

The unparalleled cost and form factor advantages of NAND flash memory has driven 35mm photographic film, floppy disks and one-inch hard drives to extinction. Due to its compelling price/performance characteristics, NAND Flash memory is now expanding its reach into the once-exclusive domain of hard disk drives and DRAM in the form of Solid State Drives (SSDs). Driven by the proliferation of thin and light mobile devices and the need for near-instantaneous accessing and sharing of content through the cloud, SSDs are becoming a permanent fixture in the computing infrastructure.

If we look at the DRAM history, DRAM data access speeds have increased at a faster rate than *Hard Disk Drives* (HDDs). The gap in read and write performances between DRAM and HDD has widened in the last years, leaving an opportunity for a new intermediate memory/storage technology between HDDs and DRAM: NAND Flash-based Solid State Drives (SSDs) can fill this performance gap, thus profoundly changing the traditional memory hierarchy below the microprocessor.

The basic architecture of SSDs is discussed in Chapter 1.

So far, the SSD design approach has been focused on the optimization of the Flash Translation Layer, i.e. the firmware required for the compatibility with traditional Hard Disk Drives. With hyperscaled SSDs this strategy is no longer valid since their performance and reliability are strictly linked to that of the NAND Flash memories that constitute the storage medium, in particular when the multilevel cell paradigm is considered. For this reason, the SSD design flow must follow a bottom-up approach that, starting from an accurate knowledge of the time and use dependent reliability of the NAND Flash memories, selects the most appropriate error correction strategy to extend the SSD's lifetime while reducing its performance degradation. Then the design flow moves to that of the SSD controller and of the interface towards the host where the application is running. Chapter 2 will thoroughly discuss this bottom-up approach and finally it will show how it is possible to leverage innovative approaches (e.g. the software defined storage system) that will be able to revolutionize the traditional computer/memory interaction, by exploiting a hardware/software co-design of the SSD controller architecture and of the host application.

To fuel the transition from HHD to SSD, NAND must remain very aggressive in terms of cost per bit. When approaching 10 nm technologies, planar NAND is running out of steam:

industry and academia have worked hard to fix this problem for more than a decade. 3D integration turned out to be the most promising alternative and it is now eventually reaching the market. Chapter 3 is about 3D NAND Flash memories and the related integration challenges. Charge Trap and Floating Gate 3D technologies will be discussed with the aid of several bird's-eye views. Advanced layout techniques will be analyzed and, finally, future scaling trends will be presented.

The advent of the 3D-NAND Flash memories has introduced significant issues in terms of characterization and system-level optimization that can be performed to increase the memory reliability over its lifetime. Indeed, the knobs that a system designer can leverage to this extent are many. In Chapter 4 we'll show that the application of machine learning algorithms like data clustering on a large characterization data set of TLC 3D-NAND Flash devices can help designers to optimize the countermeasures for improving the memory reliability, while reducing their implementation cost.

NAND Flash memories are complex systems that include many heterogeneous blocks that must work together to ensure a high reliability of the information storage. Many efforts in the reliability community are devoted to investigating the reliability-loss of this storage medium from a cell device physics point of view, whereas little importance is given to the other blocks that constitute such a system. In Chapter 5 we present a reliability threat related to NAND Flash memories that is present on the high voltage circuitry of the memory: the dependence from the power supply. Through the experimental characterization of TLC mid-1X samples, and thanks to the SPICE simulations of the high voltage blocks, we have investigated the possible sources of this new reliability issue.

The read disturb is another important problem related to TLC NAND Flash memories since their usage model is predominantly based on read-intensive applications. The state-of-the-art testing and qualification methods of Flash memories are performed by uniformly stressing the memory blocks with the same amount of reads. However, by analyzing several workloads, it appears that the read operations can also be concentrated in a specific address range. In Chapter 6, we'll show the different behavior of a mid-1X TLC NAND Flash under uniform and concentrated read disturb. The results are used to speculate the implications of the workload usage model on the reliability of enterprise Solid State Drives, when using different error correction strategies and data management policies.

Flash technology in not the only possible medium for SSDs because a lot of emerging memories are gaining more and more traction in the market. As a result, in recent years, both industry and academia have increased their research effort in the hybrid memory management space, developing a wide variety of systems. It is worth mentioning that "hybrid" is a generic term and it can have different meanings depending on the context. For instance, a storage system can be hybrid because it combines HDDs and SSDs; an SSD can be hybrid because it combines SLC (1 bit/cell) and TLC (3 bit/cell) Flash memories, or it combines different non-volatile memories like NAND and RRAM, MRAM, PCM, etc.

RRAM is perceived by the storage community as a reliable alternative to NAND Flash in SSDs for low latency applications. These emerging memories are non-volatile as NAND Flash, but with a lower read/write latency and a higher reliability. However, the relatively small storage capacity of RRAM memories integrated so far has limited their usage to specific applications such as saving critical data during power loss events or as a cache memory for fast data manipulation. In this case, RRAMs are combined with NAND Flash memories to minimize latency and to improve both the bandwidth and the reliability of the drive. In Chapter 7 a thorough design space exploration of a 512 GB All-RRAM SSD architecture is performed, with attention to architectural bottlenecks and inefficiencies, by using a custom developed simulator. We assumed a full compatibility of RRAM chips with typical NAND Flash interfaces, and hence a state-of-the-art SSD controller is embodied in the simulation environment. In light of these considerations, we leverage both the internal page architecture of a 1T-nR RRAM chip and the SSD's firmware to find the optimal configuration, thus enabling the adoption of the RRAM technology in high performance SSD applications. Collected results show that, in standard working condition (i.e., when 4 kB transactions are issued by the host system), All-RRAM SSDs can show extremely low latency only if a proper management of the operations is adopted.

PCIe DRAM/Flash-based NVRAM (Non-Volatile RAM) cards are gaining traction in the market because they can be used either as a very fast and secure synchronous write buffer, or to store both critical system data and user data in case of Power Failure. In a nutshell, the host sees the NVRAM card as a bunch of DRAM devices connected over a PCIe bus. If the power suddenly disappears, the on-board controller copies the DRAM content to a bank of Flash memories; during this copy operation, a super capacitor supplies the necessary energy. MRAM memories are now mature enough to offer a technically viable alternative to the combination of DRAM and Flash, thus removing the need for a super capacitor, because of the MRAM inherent non-volatility. In Chapter 8 we present an analysis of IOPS and latency (QoS) for both DRAM/Flash-based and All-MRAM NVRAM cards. Results of simulations indicate that MRAM NVRAM cards can compete with legacy DRAM/Flash cards, at least when looking at performance figures such as random read/write IOPS and latency.

Ensuring data protection in Solid State Drives is vital in enterprise applications. However, as the reliability of their storage medium, namely the NAND Flash, is decreasing at the same pace of the technology scaling, this activity is becoming non-trivial. The evaluation of different recovery strategies that employ both complex Error Correction Codes and second level error correction is becoming a more and more common approach. In Chapter 9 we model the endurance reliability of an advanced data protection methodology like the intra-disk Redundant Array of Independent Disks (RAID) applied to mid-1X Triple Level Cell NAND Flash-based SSD. The performed investigations include a parametric analysis of the Uncorrectable Bit Error Rate. By developing a dedicated discrete-time Markov-chain model of an SSD we evidenced that intra-disk RAID5 and RAID6 allows achieving an inherent reliability level compliant with the qualification target for enterprise SSDs. Finally, we'll provide a global picture of the disk economy when intra-disk RAID is implemented.

After studying all the above-mentioned topics, we can state that Solid State Drives are changing the way people store and process data, but SSDs are very complex systems to build because they require a sophisticated mix of hardware, software, and firmware. On top of that, non-volatile memories can be of several types, involving totally different storage mechanisms, each of them with its own reliability challenges. All the above considerations imply tens of billions of dollars spent in R&D worldwide each year, with engineers from all over the places scratching their heads to solve very complex problems: mathematics, physics, circuit design, process technology, manufacturing, lithography, signal processing, and testing techniques are all called to give their contribution to drive the evolution of SSDs even further.

Chapter 1

Solid State Drive (SSD): a Non-Volatile Storage System

Over the last 15 years, NAND Flash memories have changed our lives: Flash cards (mainly in the SD – Secure Digital - form factor) have almost completely replaced photographic films, and USB-keys have driven floppy disks to extinction. Lately, thanks to a great trade-off between cost and performance (i.e. write/read speed), NAND Flash technology has begun fighting against *Hard Disk Drives* (HDDs) in the form of *Solid State Drives* (SSDs).

In a nutshell, HDDs [1] can be seen as electro-mechanical devices because the information is stored on a spinning disk, covered with ferromagnetic material. A motor drives the spinning disk while a moving actuator arm has to tightly control the position of the magnetic head in charge of writing and reading to/from the storage media. The simple fact that there is a rotating disk implies that random access is limited by the mechanical movement of the disk; reaching a different area of the spinning plate in less than a millisecond is definitely tough. Modern applications like financial transactions, data mining, machine learning, and cloud computing need very fast access to stored data and HDDs aren't the best fit for them. Moreover, the mechanical parts pose a major constraint on reducing the HDD form factor and they also represent a major source of power consumption.

Smartphones and tablets have played a key role in looking for something different from HDDs because portable applications absolutely need less power-hungry and lighter storage devices. But this is not the only reason. Historically, if we focus only on access speed to stored data, DRAMs have greatly outpaced HDDs, thus creating a big gap in the so-called *memory hierarchy*, which is shown in Figure 1.1. It was exactly this gap that opened the door to new comers in the storage infrastructure, which, in the old days, was an exclusive domain of HDDs (and tapes). The abovementioned gap in read and write performances is now so big that even NAND Flash memories cannot fill it entirely. The gap between DRAM and NAND is supposed to be covered by a new class of memories called SCMs, which stands for *Storage Class Memories*. Both industry and academia are placing a lot of effort in identifying and developing these new memories. MRAM (Magnetic RAM), ReRAM (Resistive RAM), Carbon Nanotubes, and 3D XPoint are some of the leading SCM candidates.

1.1 Flash technology

Let’s now take a closer look at NAND technology [2]. Flash memories are solid-state devices; in other words, they are “simple” pieces of silicon without any moving mechanical parts. Just because of that, there is no need for a motor, which greatly improves access speed to stored data by itself. As a rule of thumb, a NAND memory can be read in less than 100 µs. More importantly, there is no difference between sequential and random access as there is no sensing head that needs to move across the silicon die. Being much faster than HDDs, SSDs have moved the speed bottleneck from the storage side to the Host side. Legacy storage interfaces like SATA (Serial ATA) and SAS (Serial Attached SCSI) are now running out of steam and this is why faster “computing” interfaces like PCI Express (PCIe) are gaining momentum in the storage market.

In terms of capacity, a single NAND die can now store up to 512 Gbit and a single package can contain up to 16 dies in a 12 mm x 18 mm footprint. At this point, SSDs can really challenge HDDs in most of the applications. This massive storage density improvement has been enabled by two main technologies:

- 3D (vertical) integration (Figure 1.2) [3]. NAND memory cells can be vertically stacked to form multiple memory layers within the same silicon die. The most recent devices have 64 layers but memories with more than 100 layers are expected to come in the near future. 3D Flash memories are reviewed in more details in Chapter 3.
- Multi-level storage (Figure 1.3). Flash storage is built around the ability of trapping and de-trapping electrons inside a MOS transistor. In practice, the population of trapped electrons acts as an electrostatic shield and it ends up modifying the transistor’s threshold voltage. By carefully modulating the number of electrons, multiple threshold voltages can be generated and translated into the digital domain.

For instance, 8 voltage values will result in 3 bits of digital information. Based on the number of voltage levels, NAND memories can be classified as follows:

SLC: 2 threshold voltages, 1 bit per memory cell
MLC: 4 threshold voltages, 2 bits per memory cell
TLC: 8 threshold voltages, 3 bits per memory cell
QLC: 16 threshold voltages, 4 bits per memory cell

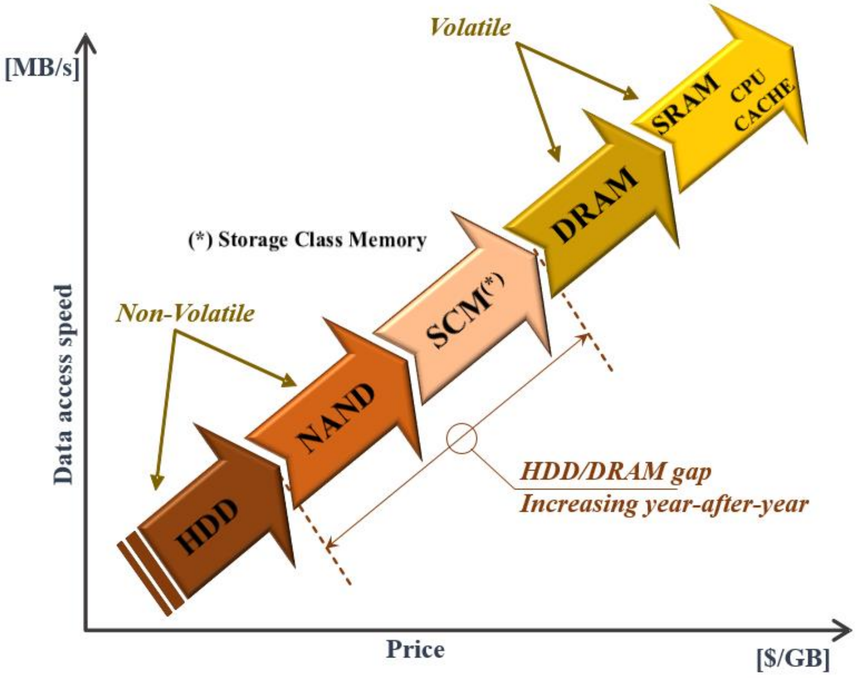


Figure 1.1 Memory hierarchy

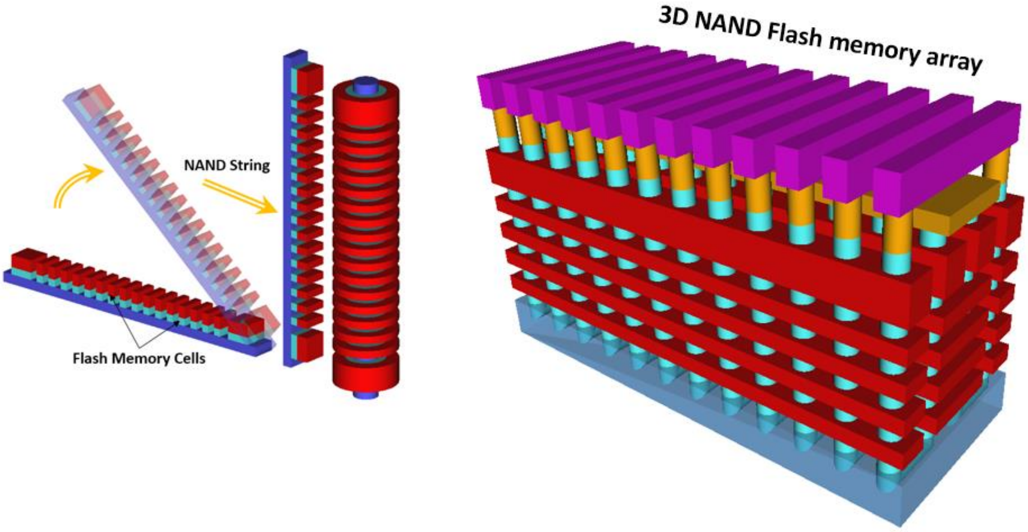


Figure 1.2 3D NAND Flash Memory Array: NAND strings go from planar (left) to vertical (right)

The abovementioned 512 Gbit devices are based on TLC storage but QLC devices are under development and they are expected to reach the market in the coming few years.

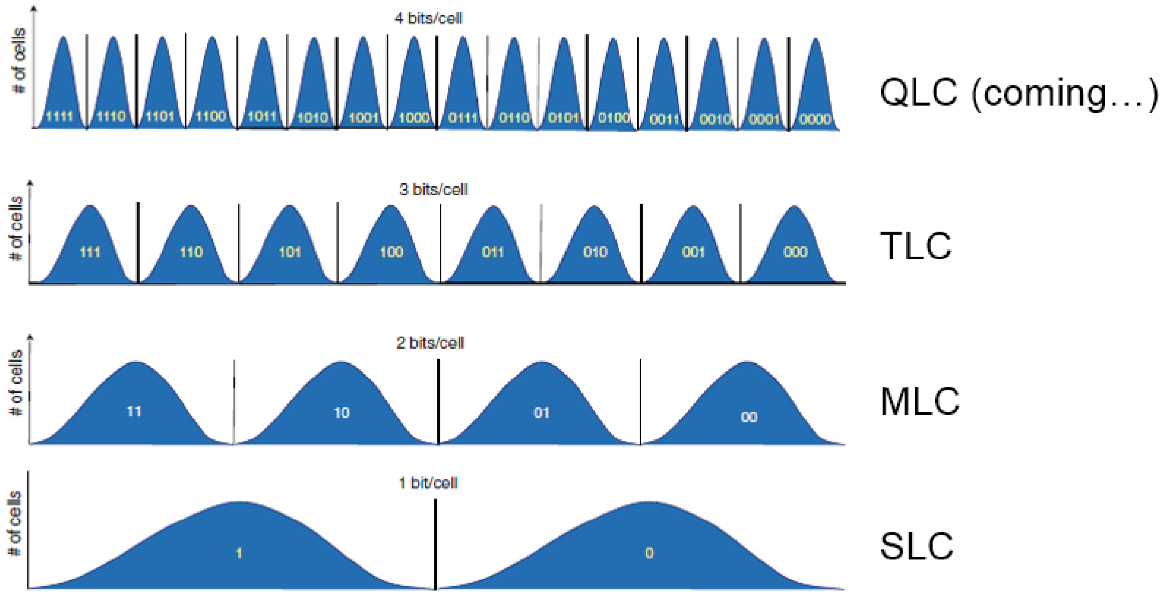


Figure 1.3 NAND classification based on how many bits are stored per physical memory cell

1.2 SSD's block diagram

When we open the case of an SSD, we find a complete system inside; human eyes can just see part of it, the hardware (HW) one, but the firmware (FW) part is as important. Let's start from what we can immediately see. A simplified block diagram of a typical SSD's HW is shown in Figure 1.4. Of course, there are plenty of NAND Flash memories, but the Microcontroller is definitely the brain of the system. It is also common to find other components like:

- DC-DC converters to derive all the necessary internal power supplies;
- quartz crystals for high precision clocks;
- filter capacitors for filtering power supplies;
- a network of temperature sensors for power management (for instance, if the temperature becomes too high, performances can be throttled not to exceed SSD's power budget).

- fast DRAM components are used for data caching: when the System Host issues a write operation to the drive, data are actually first cached to reduce the transfer time seen by the Host, and then copied to the Flash sub-system.

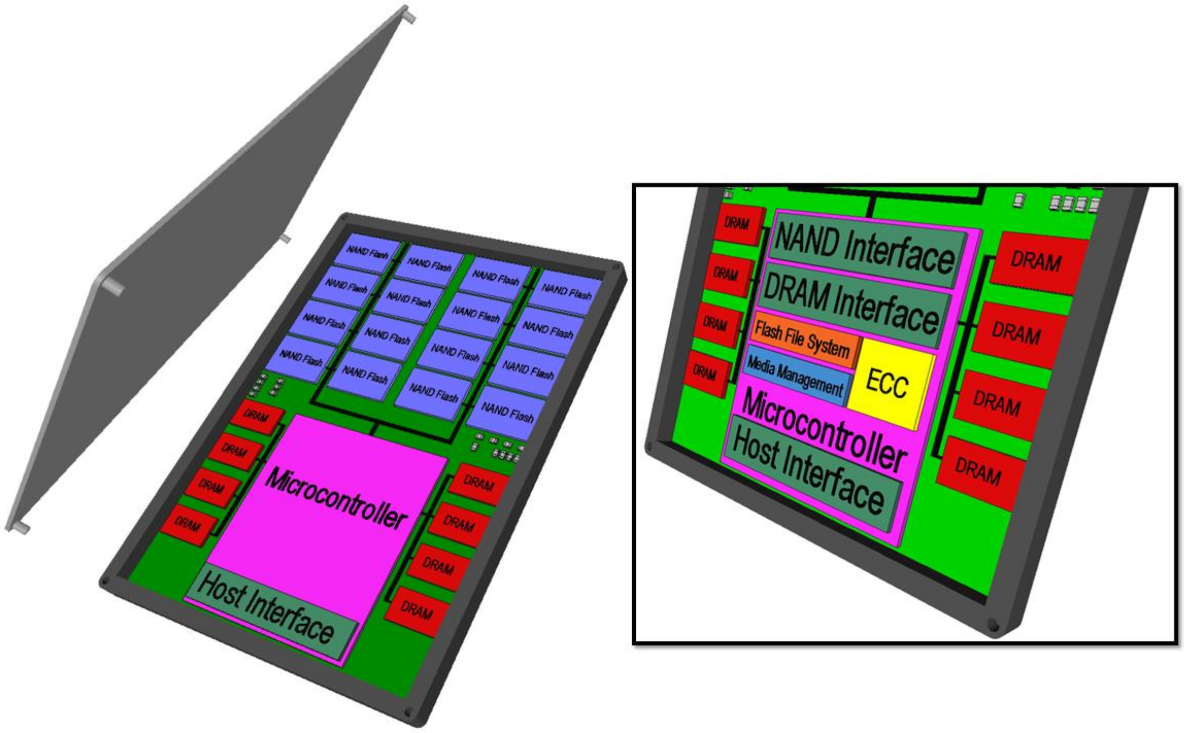


Figure 1.4 Solid State Drive – Block Diagram

At a very high level, the SSD’s microcontroller (or simply Flash Controller) needs to take care of the following tasks [4]:

- communication to/from the System Host;
- communication to/from the Flash sub-system by using the selected electrical interface and protocol (e.g. ONFI or Toggle);
- communication to/from DRAM sub-system;
- read/write performances;
- data integrity during all data transfers, and retention of the stored non-volatile information (which is very sensitive to temperature).

Generally speaking, activities of Flash controllers can be grouped into six modules, which can be implemented either in hardware or in firmware, depending on design choices and target performances (Figure 1.5).

The first module connects the drive to the Host System (*Host Interface* in the block diagram of Figure 1.5). In other words, it enables the physical connection between Host and SSD based on the selected protocol (e.g. PCIe, SAS, SATA, etc.), thus ensuring both logical and electrical interoperability. Usually, this block is made of HW (e.g. buffers, drivers, etc.) and FW (e.g. one of the Cores is used to decode the command sent by the Host). When Host commands are decoded, the second module, the *Flash Interface,* kicks in. In essence, this second module translates all the decoded commands into low-level instructions for the NAND sub-system. Again, the controller needs to guarantee the electrical interoperability with NAND devices.

The two most popular commercial NAND protocols are called ONFI and Toggle, and they are capable of transferring data in DDR mode up to 800 MB/s (1 GB/s and beyond might be possible in future generations). Another electrical interface (the third module, *DRAM Interface*) that needs to be handled by the Flash controller is the one towards DRAM components which is mainly used for data caching and for storing the mapping tables required by the FTL (see below).

The fourth module is the *Flash File System* (FFS) [5]; the main goal here is to make an SSD look like a standard HDD to the Host, main reason being the possibility of re-using all the existing applications, not necessarily developed having in mind the specific properties of the solid-state storage. Typical FFS implementation is FW based, as sketched in Figure 1.5. There are four main FW layers: *Flash Translation Layer* (FTL), *Wear leveling*, *Garbage Collection*, and *Bad Block Management*.

In order to understand why there is a need for such a complex FW infrastructure, we first need to dig a little bit deeper in how Flash memories store data. Flash arrays start from the so-called “erased” state where all bits are set to “1”. Write (a.k.a. Program) operations can change the state of each single bit to “0”; in other words, writing is selective at the bit level. Unfortunately, we can’t state the same for the “erase” operation, i.e. the operation that brings the digital value back from “0” to “1”: erase can only act on group of cells called “blocks”. The whole Flash memory array is split in thousands of blocks, and each of them is made by hundreds (or thousand) of pages. Nowadays, each page is 16 kB long: read and write operations work on pages, in the sense that the user read and write data patters of 16 kB in parallel.

Because of this very unique storage functionality, a “simple” page update is actually not that simple. In fact, a page update implies changing some of the bits from “0” to “1” and this operation, in Flash terms, means erasing. The point is that a single erase operation involves multiple pages and it can take several milliseconds to complete. Because it would take too long, what actually happens is that the updated page gets written to a different memory location and the original page (i.e. the page that needed to be updated) gets invalidated. As a consequence, there is a mismatch between physical and logical page addresses. This misalignment can be fixed by using tables to store the logical-to-physical mapping, and this is the so-called *Flash Translation Layer* or FTL. The number of tables can be huge and it can have a significant impact on the effective SSD's storage capacity, if it is not carefully designed.

Operation after operation (especially write and erase operations), NAND Flash memories wear out in the sense that it becomes more and more difficult to precisely control the number of electrons trapped inside memory cells; in other words, it becomes harder to generate the number of threshold voltages required for multi-level storage (MLC, TLC or QLC). Therefore, it is critical to spread operations across the memory array as much as possible. Wear Leveling algorithms are designed to accomplish this goal by leveraging the above-described concept of logical-to-physical translation.

When the System Host wants to update a specific page within a specific block, the Flash controller dynamically maps the new content to a different block. Wear leveling algorithms are in charge of deciding which of the available blocks to pick. There are two possible strategies. *Dynamic* wear leveling looks for the block with the lowest erase count, while *Static* wear leveling choses among blocks whose erase count deviates from the average, even if they have been recently erased. Wear leveling needs a pool of "free" (i.e. erased) blocks. When the population of this pool goes below a threshold, a FW layer called *Garbage Collection* takes over.

This algorithm selects the block that needs to be erased based on a pre-defined cost function; it copies the entire content to a different block, and then it triggers the erase operation such that the block can be moved to the list of available blocks. Usually, Garbage collection is a background operation to avoid any performance (throughput and latency) hit; in other words, write and especially read operations have higher priorities compared to erase operations, which take a much longer time to complete. Of course, given the same SSD's workload, the bigger the memory capacity, the lower the number of operations that each cell has to experience.

The fourth FW layer, the *Bad Block Management* (BBM) takes care of the so-called *Bad Blocks* (BB): these blocks contain unreliable cells and, therefore, can't be used to store data. NAND Flash devices contain BBs when they are shipped from the factory, but new bad blocks can pop up during SSD's lifetime as a result of failures during either programming or erasing. BBM keeps track of BBs in a dedicated list, which has to be non-volatile. In fact, this list has to be retrieved at every boot of the drive (power-up) to avoid storing user data inside unreliable memory cells.

Let's now go back to Figure 1.4. The fifth module inside the Flash controller is the one performing the error recovery, i.e. the *Error Correction Code* (ECC) [6]. Historically, BCH (Bose-Chaudhuri-Hocquenghem) code has been the code used to enhance NAND reliability, mainly because its HW implementation is relatively simple. Most recently, LDPC (*Low density Parity Check*) codes [7] have caught a lot of attention because they can get much closer to the Shannon limit.

With Flash technology moving to high 3D stacks and QLC coming in few years, the possibility of correcting more errors becomes very attractive. LDPC codes leverage complex algorithms, require more logic gates and, therefore, consume more power. Therefore, there is a big research activity in this space trying to find the right trade-off between correction performances and HW cost.

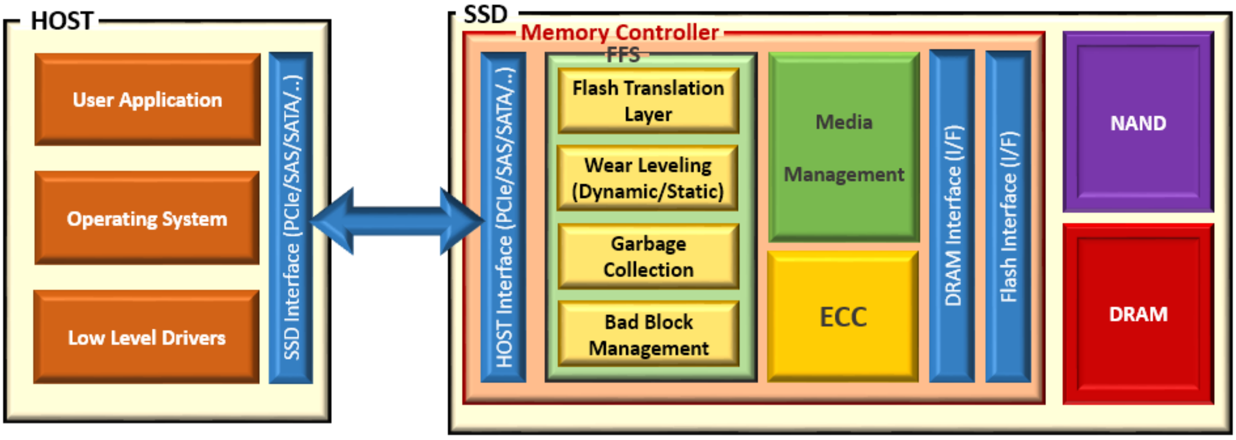


Figure 1.5 Functional view of a Flash controller

Last but not least, in Figure 1.4 we have the *Media Management* module. By *media* we mean either NAND Flash or any other emerging non-volatile memories (e.g. ReRAM, MRAM, etc.). Given the fact that LDPC codes are approaching the Shannon limit, there is not so much space left for improving the SSD's lifetime by "simply" increasing the number of errors that the Flash controller can recover. As a result, there is the need for reducing the

BER growth rate such that the ECC maximum correction capability is reached at a higher count of Program/Erase cycles, as sketched in Figure 1.6.

When looking at Flash technology, all the techniques used to mitigate the NAND raw BER fall under the term *Flash Signal Processing* (FSP) [8]: data randomization and read oversampling (a.k.a Read Retry) are popular examples of these techniques. Therefore, the Media Management module is in charge of executing FSP.

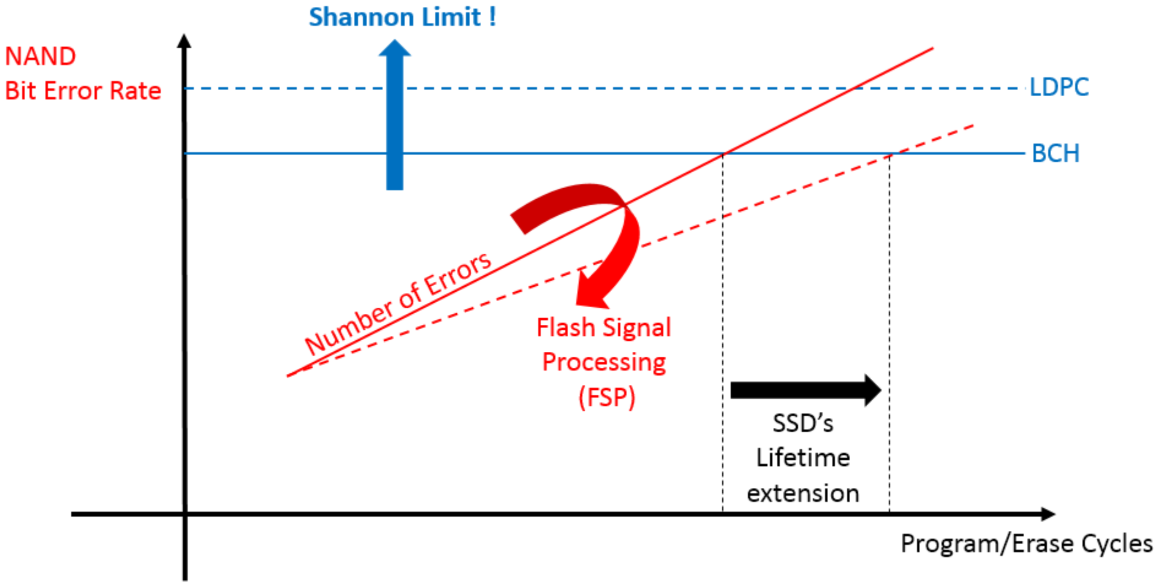


Figure 1.6 Flash Signal Processing is used to mitigate NAND BER growth

1.3 Hybrid SSDs

Most recently, both industry and academia have increased their research effort in the hybrid memory management space, developing a wide variety of systems. Actually, “hybrid” is a generic term because it can have different meanings depending on the context. For instance, at the system level, storage can be hybrid because it combines HDDs and SSDs together. A single SSD can be hybrid because of two reasons:

- it embeds different types of NAND memories: SLC and TLC, SLC and QLC, etc.;
- it combines different non-volatile memories like NAND and ReRAM, MRAM, PCM, etc.

Of course, the combination of different memories in the same system boosts the complexity to a completely different level, both in terms of firmware and data management. Indeed, in order to exploit all the benefits of the different memories, applications running on the System Host side have to carefully decide where it is more convenient to store a particular set of data. This opens the box to the concept of “data temperature”: data are defined as “hot”, “warm”, and “cold” depending on how frequently they are updated and accessed.

Solid State Drives are changing the way people store and process data, but SSDs are very complex systems to build because they require a sophisticated mix of hardware, software, and firmware. On top of that, non-volatile memories can be of different types, involving totally different storage mechanisms, each of them with its own reliability challenges. All of the above considerations imply tens of billions of dollars spent in R&D worldwide each year, with engineers from all over the places scratching their heads to solve very complex problems: mathematics, physics, circuit design, process technology, manufacturing, lithography, signal processing, and testing techniques are all called to give their contribution to drive the evolution of SSDs even further.

In the next chapter we’ll address the topic of how to design high performance SSDs, as this an area where a lot of innovation is required.

Bibliography

[1] T. Zhang, G. Mathew, H. Zhong, R. Micheloni, “Modern Hard Disk Drive Systems: Fundamentals and Future Trends”, Chapter 4 in *Memory Mass Storage* (G. Campardo, F. Tiziani, M. Iaculo, Eds.), Springer, 2011.

[2] R. Micheloni, L. Crippa, A. Marelli, *Inside NAND Flash Memories*, Springer, 2010.

[3] R. Micheloni (Ed.), *3D Flash Memories*, Springer, 2016.

[4] R. Micheloni, A. Marelli, K. Eshghi, *Inside Solid State Drives (SSDs)*, Springer, 2013.

[5] R. Micheloni, M. Picca, S. Amato, H. Schwalm, M. Scheppler and S. Commodaro, "Non-Volatile Memories for Removable Media," in *Proceedings of the IEEE*, vol. 97, no. 1, pp. 148-160, Jan. 2009.

[6] R. Micheloni, A. Marelli, R. Ravasio, *Error Correction Codes for Non-Volatile Memories*, Springer, 2008.

[7] N. Xie, W. Xu, T. Zhang, E. F. Haratsch, and J. Moon, Concatenated LDPC and BCH Coding System for Magnetic Recording Read Channel with 4K-Byte Sector Format, *IEEE Transactions on Magnetics*, vol. 44, no. 12, pp. 4784-4789, Dec. 2008.

[8] B. Shin, C. Seol, J. S. Chung and J. J. Kong, "Error control coding and signal processing for flash memories," *2012 IEEE International Symposium on Circuits and Systems*, Seoul, 2012, pp. 409-412.

Chapter 2

Memory-driven design methodologies for high performance SSDs

Solid State Drives (SSDs) are one of the electronic systems with the higher development rate in the last decade: they are widely used in hyperscale systems such as cloud computing and big data servers where performance is a constraint, as well as in consumer electronics as a replacement for traditional hard-disk drives (HDDs) [1].

SSDs' design, in the last 5 years, faced an extraordinary evolution caused by the continuous development of NAND Flash memories which are used as the storage medium [2]. Indeed, as shown in Fig. 2.1, NAND Flash memories have completely transformed the way information is processed and stored. Starting as film and tape replacement for cameras and voice recorders, NAND Flash memories rapidly surpassed traditional magnetic storage supports and now they represent an obliged choice for high-performance storage solutions. The availability of NAND Flash-based SSDs also materialized as an astonishing proliferation of global-scaled corporations whose commercial strength is tightly coupled to the availability of SSDs engineered for big data centers and cloud computing. The previous developing strategy of SSDs, in fact, was based on a full compatibility with HDDs and therefore the SSDs' performance optimization was focused on that of the Flash Translation Layer (FTL), the firmware managing the basic memory operations [3, 4, 5]. As mentioned in Chapter 1, FTL is responsible for a plug-and-play connection between the host system, where the

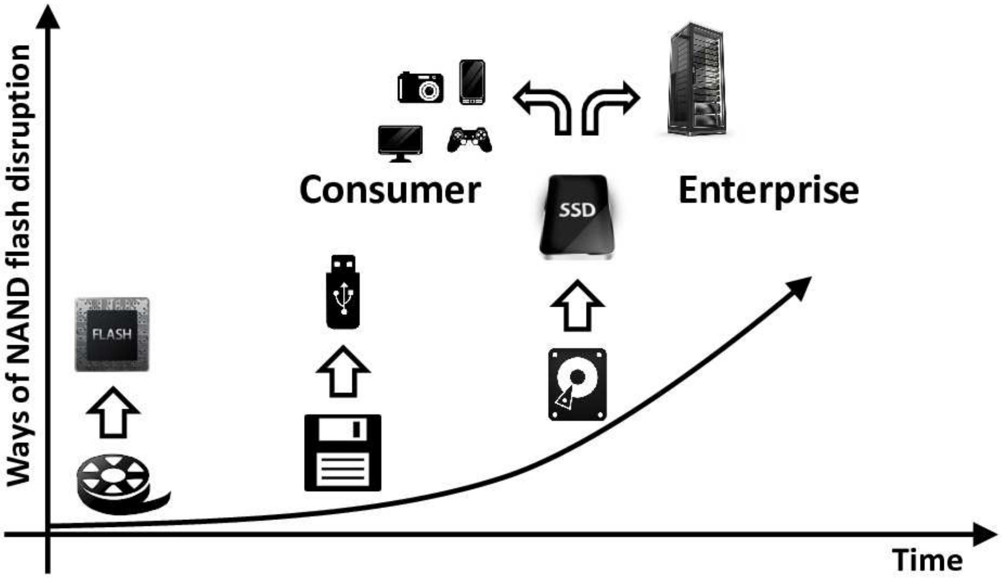


Figure 2.1: Evolution of NAND Flash-based systems: from tape, film and floppy disk replacement to the explosive SSDs applications for cloud computing and big data centers

application is running, and the SSD. It must also be considered that in the last 4 decades user applications have been designed to work with traditional magnetic HDDs, which are conceptually different from SSDs. Therefore, rather than redesigning the whole architecture of the application, it is more convenient to leverage a command translation layer.

The development of SSDs was made possible by the use of sufficiently reliable Single Level Cells (SLC) NAND Flash memories [6], storing a single bit per cell in the traditional 0/1 digital paradigm with a low read error probability, thus requiring the design of simple engines for Error Correction Codes (ECC) [7]. The SATA protocol [8] sitting between the memory system and the host was sufficient to guarantee the requested Quality of Service (QoS), that is the ability of keeping a sustained performance over time within a defined threshold [9, 10]. As a whole, the SSD architecture optimization and the development of dedicated CAD tools for the exploration of the SSD design space were FTL-oriented, in a top-down approach.

In the last few years, the need for SSDs with higher storage capacities and performance combined with the availability of high density NAND Flash memories able to store 2, 3 or even 4 bits in a single cell [11], moved the design paradigm from a Top-Down to a Bottom-Up approach, where the performance and the reliability of the storage medium dictate the design constraints [12]. The reliability of NAND Flash memories with scaled technologies, in fact, suffers from several physical mechanisms. Reliability's key metrics are: *i*) *Endurance*, that is the maximum number of Program/Erase (P/E) operations that the memory can withstand before leading to a failure; *ii*) *Data Retention*, i.e. the ability of a memory to keep a stored information over time without power supply; *iii*) the immunity to *Read Disturbs*, which represents the stress suffered by a memory cell when neighbor cells are read [13, 14, 15].

In NAND Flash memories, the stored information is associated to the amount of charge present in the storage layer. P/E (Program/Erase) operations rely on charge transport through a thin oxide via Fowler-Nordheim (FN) tunneling into/from the storage layer [16]. Electron tunneling is responsible for a slow but continuous oxide wear out, thus causing undesired charge flowing into/from the storage layer. As the number of P/E cycles increases, this effect strongly impacts the writing operation. To deal with endurance effects, sophisticated (but slow and power hungry) algorithms are adopted to tightly control the amount of charge transferred into/from the storage layer [17]. However, the relentless oxide degradation strongly affects the ability of keeping unaltered the charge content into the storage layer for long times, a mandatory requirement to fulfill the nonvolatile paradigm. These reliability issues become more and more significant in Multi-Level Cells (MLC) [18], Triple-Level Cells (TLC) [19] and Quadruple-Level Cells (QLC) [20] storing 2, 3, and 4 bits per cell, respectively, where the undesired transfer of few electrons into/from the storage layer may significantly alter the memory information content. Hereafter, MLC, TLC, and QLC architectures will be generically denoted as *multilevel cells*.

The key metric describing the NAND Flash memory reliability is the Raw Bit Error Rate (RBER), which represents the fraction of erroneous bits retrieved during a read operation [15]. The RBER value increases with: technology scaling, the number of bits that a cell can store, the number of P/E operations, the time elapsed between two successive read operations, and the number of repeated read operations on the same memory location. As a matter of fact, RBER is the new driver for architectural and software design of present SSDs [21].

Multilevel NAND Flash memories require the availability of an ECC able to correct the errors detected when reading the memory. The choice of the ECC code together with its hardware design represent the key point for present SSDs design since they must be carefully calibrated with respect to the figures of merit of the selected nonvolatile memories. Indeed, a too simple ECC scheme may not be able to guarantee a suitable reliability, whereas a too complex one may reduce severely the read bandwidth because of the time required for error correction, with a consequent impact on the system power consumption [22]. On the basis of the selected ECC code and of the designed ECC engine, an optimal error reduction algorithm for the memory read operation can be identified. The selection of the appropriate NAND Flash memories and the identification of the adequate ECC scheme represent the key point to guarantee a high QoS for the SSD to be designed.

Once the ECC scheme has been designed, the Bottom-Up design flow rises to the memory controller, representing the interface towards the ECC engine and the memory storage system. The bandwidth provided by the ECC block must be guaranteed by the controller, to avoid that the design efforts devoted to optimize the ECC scheme vanish. With this respect, the SSD controller must be designed in order to manage a sufficient amount of commands to fully exploit the bandwidth of the underlying storage system. Similarly, also the interface towards the host must be able to guarantee the expected bandwidth. For this reason, SATA protocol is no longer able to deal with the performance made available by the other blocks in the SSD architecture [23] so that SAS [24] and PCI-Express [25] are adopted for enterprise environments.

If this new bottom-up approach in the SSDs design flow is the way to go, then CAD tools for SSD design must change accordingly, thus reducing the effort previously spent on FTL design [26].

In this chapter, starting from a review of the basic reliability issues in multilevel NAND Flash memories, several aspects related to the design of an SSD architecture will be presented. Emphasis will be given to the choice of the appropriate ECC code, the design constraints of the ECC engine able to guarantee the optimal trade-off between performance and reliability [27], the controller design, and the selection of the host interface protocol able to sustain the bandwidth provided by the storage system. We will explain why the SSD performance rapidly decreases with use and time and why a different design approach allows fully exploiting the NAND Flash features, thus extending the SSD's lifetime.

This chapter is organized as follows: in Section 2.1 multilevel NAND Flash operations and reliability are analyzed with emphasis on how oxide ageing impacts on endurance, data retention, and read disturb. Section 2.2 is devoted to ECC and the impact of the decoding time on data read throughput. Section 2.3 deals with the advantages introduced by dedicated command queueing strategies, and by the adoption of DRAM-caching [28]. In Section 2.4, the criteria for the optimal host interface selection are addressed, focusing on the trade-off between cost and perfomances, on the relationship between queue depth and bandwidth, and on the host payload co-design for optimal performance exploitation. Finally, in Section 2.5, the chapter speculates on future research opportunities made possible by high-performance SSDs with multi-core Flash controllers, such as *software defined storage systems* [29].

2.1 NAND Flash memory cells: basic operations and reliability

The most common Flash memory cell is a metal-oxide-semiconductor device with an electrically isolated floating gate (FG). The insulation is achieved by a tunnel oxide and an interpoly oxide (see Fig. 2.2) [30]. The former oxide plays a basic role for the control of the device threshold voltage V_T whose value represents, from a physical point of view, the stored information. In quiescent conditions, thanks to the two oxides, the charge stored into the FG does not leak away, thus granting the nonvolatile paradigm fulfillment.

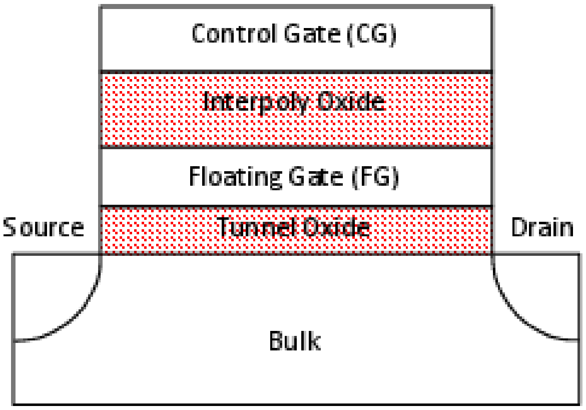


Figure 2.2: Standard floating gate memory cell used in NAND architectures

By referring, for the sake of simplicity, to a SLC architecture, programming is performed by injecting electrons within the FG, whereas erasing is performed by removing

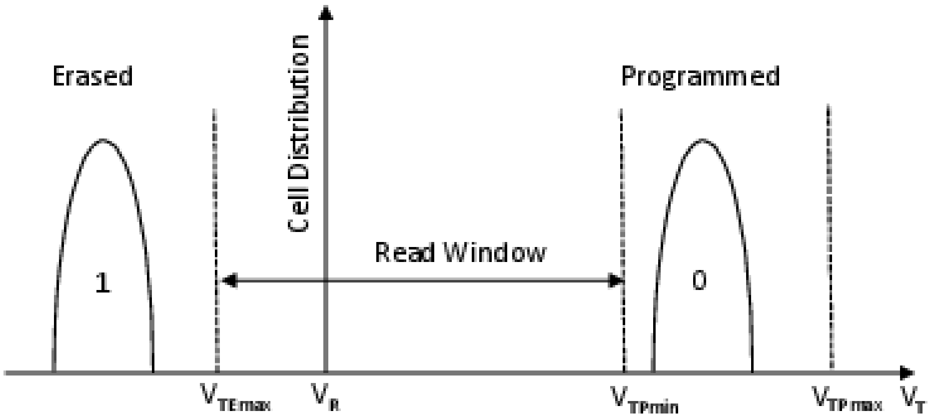


Figure 2.3: Threshold voltage distributions in SLC cells. V_{TPminx} **and** V_{TPmax} **represent the minimum and the maximum target** V_T **for a programmed cell, respectively.** V_{TEmax} **represents the maximum** V_T **for an erased cell while** V_R **denotes the read voltage.**

that charge from the FG [31]. The charge within the FG modifies substantially the cell's threshold voltage V_T and, consequently, the voltage to be applied to the Control Gate (CG) to switch the cell ON, as well as the current flowing through the device when a fixed voltage V_{CG} is applied to the CG [32]. Cell writing occurs thanks to the FN tunneling [16]: by applying high electric field to the tunnel oxide, it is possible to transfer charge to/from the FG. This operation requires an accurate control of both V_{CG} and the pulse duration t_p, since V_T must be placed in a well defined interval $[V_{TPmin}, V_{TPmax}]$ (see Fig. 2.3, where the V_T distributions of a cell array are shown). Using $V_T < V_{TPmin}$ would reduce the read margin, whereas $V_T > V_{TPmax}$ could provoke read errors in other cells of the array due to the over-programming phenomenon [33, 34].

During a cell programming, the charge injected within the FG reduces the electric field applied to the oxide. Therefore, to avoid a reduction of the program efficiency, this operation is accomplished by applying to the CG a sequence of pulses with duration t_p and increasing amplitude. Each pulse is followed by a verify operation [35] that ends the program operation when the target V_T interval has been reached, thus realizing the so-called Incremental Step Pulse Programming (ISPP) algorithm [17, 36]. It can be demonstrated that the amplitude increment ΔV_{CG} almost coincides with the threshold shift ΔV_T produced by the pulse itself [37]. The choice of the two parameters, ΔV_{CG} and t_p, allows controlling the overall programming time and the accuracy

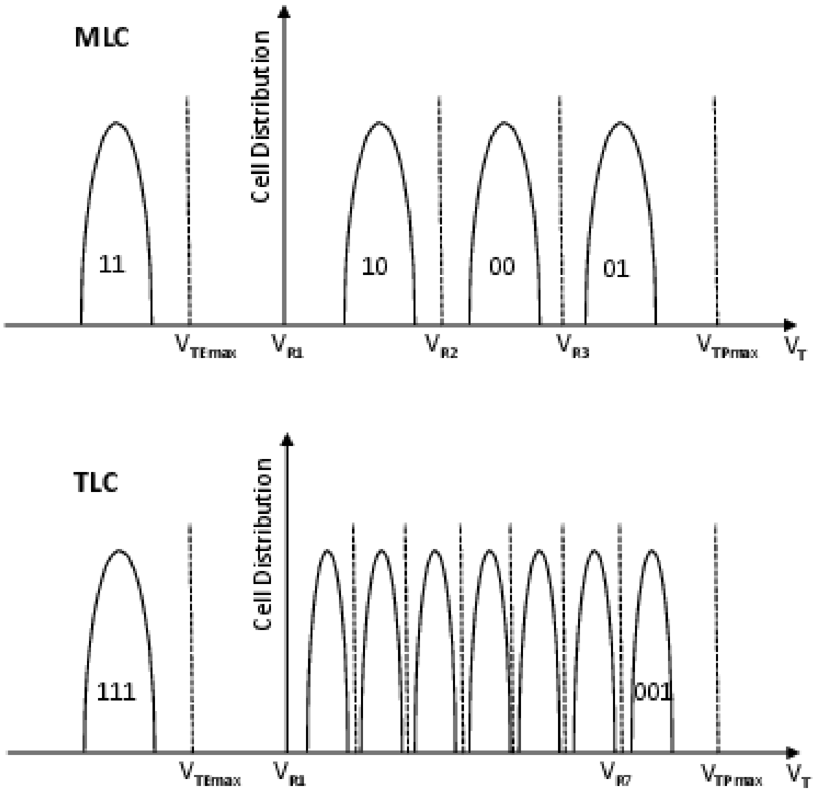


Figure 2.4: Threshold voltage distributions in MLC and TLC cells. For the MLC case the 3 reference voltages V_{R1}, V_{R2}, and V_{R3} are shown, whereas for the TLC case only 2 out of 7 reference voltages are shown.

of the placement of the cell V_T within the target interval. Long pulses and/or high ΔV_{CG} reduce the programming time but it becomes more difficult to control the cell's final V_T, whereas short pulses and/or reduced ΔV_{CG} increase the programming time but allow a tighter control of the number of electrons transferred to the FG [37, 38].

Read operation is performed by evaluating the current flowing through the cell when a fixed reference voltage V_R is applied to CG (see Fig. 2.3) [30, 39]. In a programmed cell (high V_T) the current is "low" and the read circuitry outputs a bit equal to "0", whereas in an erased cell (negative V_T) the current is higher and it is interpreted as a "1".

With the introduction of multilevel architectures (MLC, TLC, QLC), programming and reading operations become much more complex [18, 19, 20]. Since V_{TPmax} cannot be increased because of reliability constrains [40], 3, 7 or even 15 different threshold intervals must be allocated within the same voltage range, each one corresponding to a different set of 2, 3 or 4 bits stored within the cell (see Fig. 2.4).

The amplitude reduction of each interval calls for a very tight control of the charge injected within the FG. Since the relationship $\Delta V_{CG} \simeq \Delta V_T$ is still valid [37], the ΔV_T reduction forces the overall program time to increase with the number of bits stored in a cell. In a similar way, a read operation requires longer times since successive read procedures with different threshold voltage references must be considered [19, 18]. In addition, the reduced distance between adjacent intervals may trigger read errors.

The erase operation brings the cells back to the logic "1" state and it acts simultaneously on all the cells belonging to the same "block". Cells sharing the same Source line belong to the same memory block [41, 42].

The operations of Flash memory cells described so far refer to an ideal case. In the real world, tunnel oxides face a continuous wear-out, thus reducing the FN efficiency and triggering long-term reliability effects; the charge stored in the FG is not stable but leaks away producing read errors; cell's dimensions are so scaled that cell-to-cell variability must be taken into account too [43]; the number of electrons injected in the FG is so small that statistical effects during programming may produce errors. Finally, even an ideal cell is embedded in a complex array architecture so that write and read operations performed on neighbor cells may alter its stored content.

Damages in the tunnel oxide represent the main reason for reliability degradation in Flash memories. Because of the continuous charge transport through the insulator, traps can be created at the SiO_2 interfaces or within the oxide, thus modifying the FN tunneling dynamics [13, 40, 44]. The ability of tightly controlling threshold voltage distributions decreases with the number of Program/Erase (P/E) operations, and this fact impacts memory endurance [14, 15]. Fig. 2.5a sketches the effects of a reduced ability in producing tight distributions as the number of P/E cycles increases. The *Program & Verify* approach stops the program operation of a cell when the target threshold interval has been reached [35]. However, because of the tunnel oxide wear-out, some cells can be slightly over-programmed and their thresholds could end in an adjacent interval [33, 40]. As a consequence of this distribution broadening, read errors are produced. Fig. 2.6 shows the RBER measured in a TLC NAND Flash manufactured in the 1x-nm planar technology node as a function of the number of P/E cycles, evidencing a reliability reduction induced by successive write operations.

Oxide ageing and traps creation also reduce the data retention feature, that is the ability of keeping unaltered the charge within the FG when the cell is in a quiescent

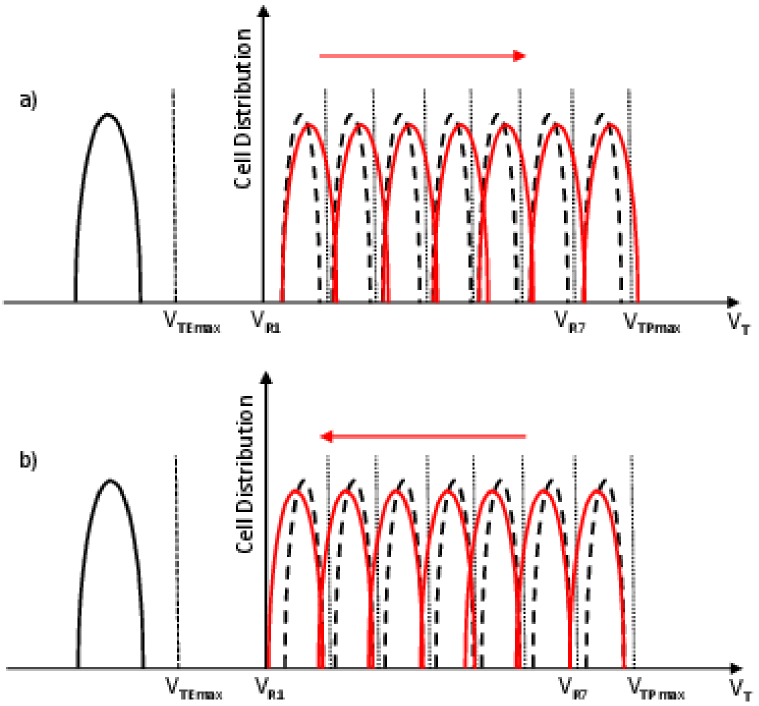


Figure 2.5: Shifts of the threshold voltage distributions in TLC cells caused by oxide ageing (dashed line: virgin samples; full line: ageing effects). Shifts towards higher intervals are caused by endurance effects (a), since the correct placement of the threshold voltage in a given interval becomes more difficult, whereas shifts towards lower intervals are due to electrons escaping from the FG causing a reduced data retention (b).

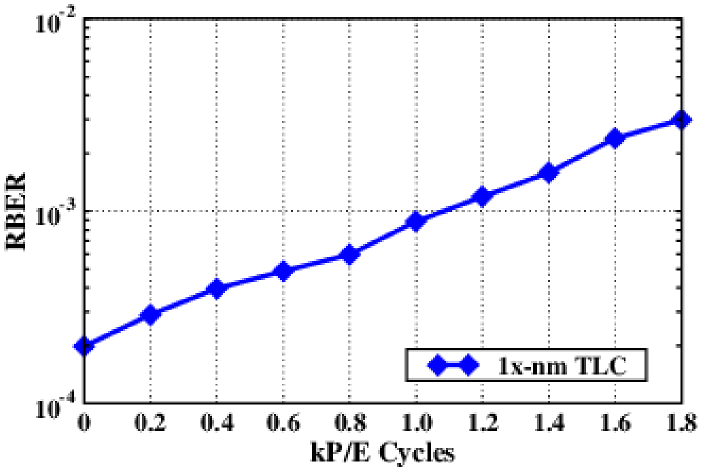


Figure 2.6: RBER measured in a 128 Gb TLC NAND Flash die manufactured in the 1x-nm planar technology node as a function of the number of P/E cycles, up to twice the rated endurance (900 P/E cycles).

state. Electrons may escape from the FG because of trap-assisted tunneling or Stress-Induced-Leakage-Current (SILC) effects [45, 46, 47, 48, 49, 50]. Fig. 2.5b shows the threshold distribution shifts during retention. The risk that the threshold of a cell shifts to an adjacent interval increases significantly with the number of bits stored in a single cell. It is worth pointing out that in a MLC or TLC architecture the number of electrons differentiating two adjacent intervals is in order of few tens, whereas in QLC cells it is sufficient that one or two electrons escape from the FG to produce a read error [51].

Besides the degradation mechanisms related to oxide wear-out described so far, other effects may worsen the ability of controlling the correct number of electrons to be transferred in the FG during a single programming pulse. Among them, the Random Telegraph Noise (RTN) related to filling/empting of tunnel oxide traps affects the V_T distributions stability few microseconds after the application of the programming pulse, creating distribution tails below the target verification level [52, 53, 54, 55]. Additionally, positive trapped charge in the tunnel oxide during cycling results in a modified FN tunnel dynamics that may trigger erratic effects [33, 56, 57, 58]. These sporadic mechanisms, that may potentially affect any cell in the array, have a random and transient nature; they can occur during any programming pulse and they may produce threshold shifts larger than expected, with the risk of programming some cells with a threshold voltage larger than the desired one. The limited number of electrons discriminating between adjacent intervals makes the programming operation discrete [38, 59, 60].

Fig. 2.7 shows the schematic of a typical memory array. Cells are organized in strings, which are the minimum read unit. Read and program operations are performed *page-wise*, by reading/programming simultaneously 8 kB or 16 kB cells belonging to the same word line [32].

Architectural solutions for memory operations may also affect the overall reliability, by producing errors and even cell failures. The most common effects are the so called disturbs, that can be interpreted as the influence of an operation performed on a cell (Read or Write) on the charge content of a different cell. Read disturbs are the most frequent source of disturbs in NAND architectures [32, 61, 62]. This kind of disturb may occur when reading many times the same cells without any erase operation of the entire block they belong to. All the cells belonging to the same string

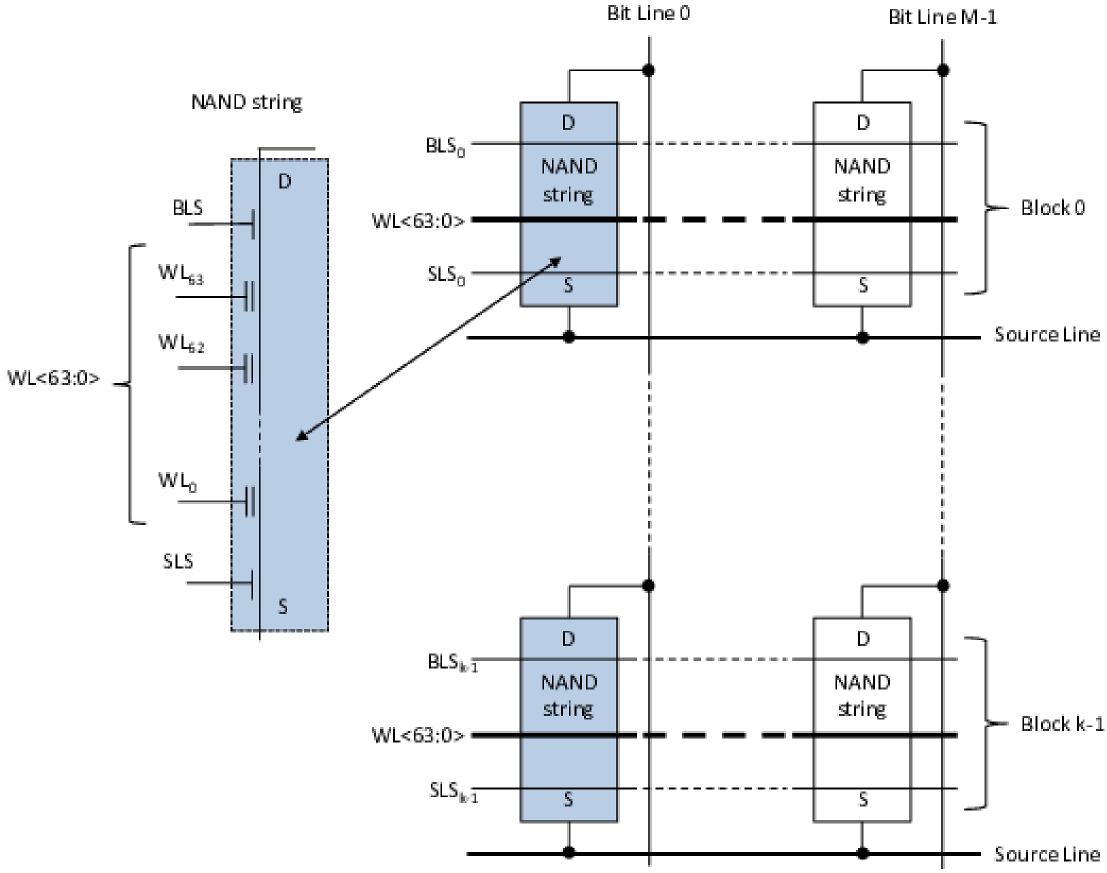


Figure 2.7: Schematic organization of a NAND flash array. Each cell string is connects to a Bit line and a Source line through two select transistors (BLS and SLS, respectively).

of the cell to be read must be driven in an ON state, independently of their stored charge (see Fig. 2.8). The relatively high $V_{PASS} > V_{TPmax}$ applied to the CG of the unselected cells to turn on their conduction and the sequence of pulses applied during successive read operations may induce a charge gain due to SILC effects [61] or hot carrier effects [62]. These cells suffer a threshold voltage shift that may lead to read errors, when addressed. The probability of suffering from read disturb increases with the P/E number (i.e., towards the end of the memory useful lifetime) and it is higher in damaged cells. Read disturbs do not provoke permanent oxide damages: if erased and then reprogrammed, the correct charge content will be present within the FG.

The NAND Flash technology scaling has introduced additional disturbance mechanisms affecting the array reliability: the cell-to-cell interference [63, 64, 65, 66] and the Gate Induced Drain Leakage (GIDL) [67, 68]. The former issue is mainly caused by the FG coupling due to parasitic capacitances between cells, thus it is greatly af-

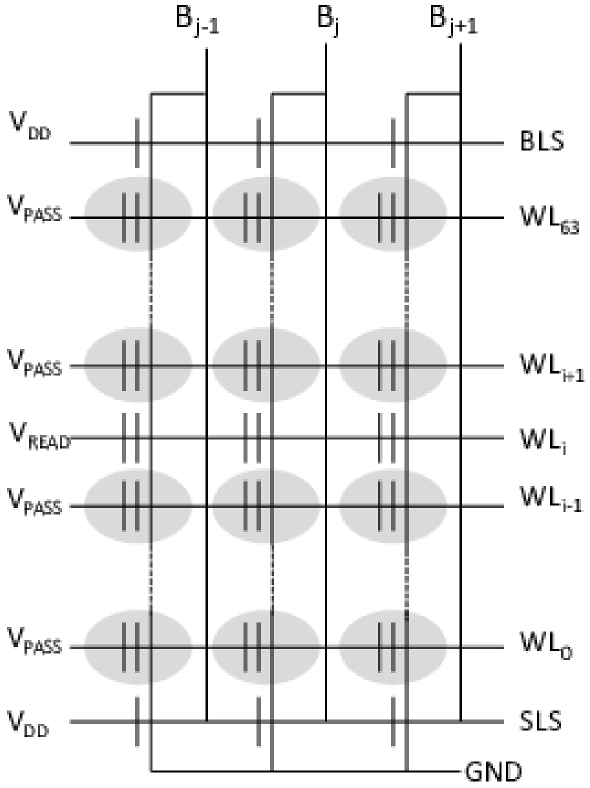


Figure 2.8: Representation of read disturb in a NAND Flash array when reading cells in the WL$_i$ word line. All cells sharing the same strings (marked in gray) are potentially affected by the read disturb.

fected by cell scaling, and is well known to widen the V_T distributions by producing read errors. The latter effect is due to the usage of the self-boosting technique to inhibit unselected cells during programming [69]. An electron-hole pair generation mechanism triggered by high electric fields during the program operation leads to the generation of charge in the region between the Source Line Selector (SLS) and the WL_0 that can be injected as hot electrons in the floating gate of cells belonging to WL_0 [67]. To avoid this effect, dummy word-lines need to be integrated in the array.

2.2 The impact of ECC on SSD's performances

As summarized in the previous section, because of endurance problems, poor data retention or read disturbs, the actual threshold voltage read in a cell may be different from the programmed one [15]. Therefore, when a page is read, some cells may return a wrong value, thus producing read errors. To overcome these problems, data encoding guaranteeing a reconstruction of the correct read page data is mandatory in electronic systems using NAND Flash memories.

The correction capability of the code to be adopted is strictly related to the error probability. For a given technology node, since physical degrading mechanisms are the same independently of the different storage paradigms (SLC, $\cdots$, QLC), the error probability increases with the number of bits stored in a single cell since the smaller the number of electrons associated to each data pattern, the higher the probability of having a V_T different from the expected one.

In the first SLC memories, thanks to the large V_T gap between the two threshold voltage distributions, the error probability was very low, so that Bose-Chaudhuri-Hocquengham (BCH) codes able to correct few tens of bits in a 1 kB or 2kB page were sufficient. With limited number of errors to be corrected, the correction time was not an issue and the read bandwidth and latency were marginally affected by the use of ECCs [70]. Read bandwidth is the number of read operations sustained in a given time, whereas latency is the time elapsed between a read command submission and its completion. Fig. 2.9a shows the typical blocks for ECC engines based on BCH codes: a high-speed encoder is connected to each one of the N_c SSD channels (that is a bus used to communicate with an array of N_d memory dies), whereas a reconfigurable parallel decoder (i.e. a multi-engine decoder) is shared among the channels [71]. The structure of the decoder is represented in Fig. 2.9b, where the *Syndrome block* determines whether an error is present, the *Berlekamp-Massey block* calculates the coefficients of the error locator polynomial, and the *Chien machine* locate the errors [70].

In multilevel architectures the number of errors to be corrected increases by an order of magnitude for any further bit stored in a single cell. Although ECC engines based on BCH codes are still used thanks to their simple hardware implementation, high numbers of bits to be corrected may impact significantly on the overall read time. As a consequence, the correction time may become the bottleneck of the entire read procedure [21]. In addition, because of the high number of errors, the probability of having uncorrectable pages (that are pages read with a number of wrong bits higher than the ECC correction capabilities) increases [72]. When a page is marked as uncorrectable, the read operation fails and the page content is irremediably lost. The adoption of parallel decoding architectures can reduce the bandwidth and latency

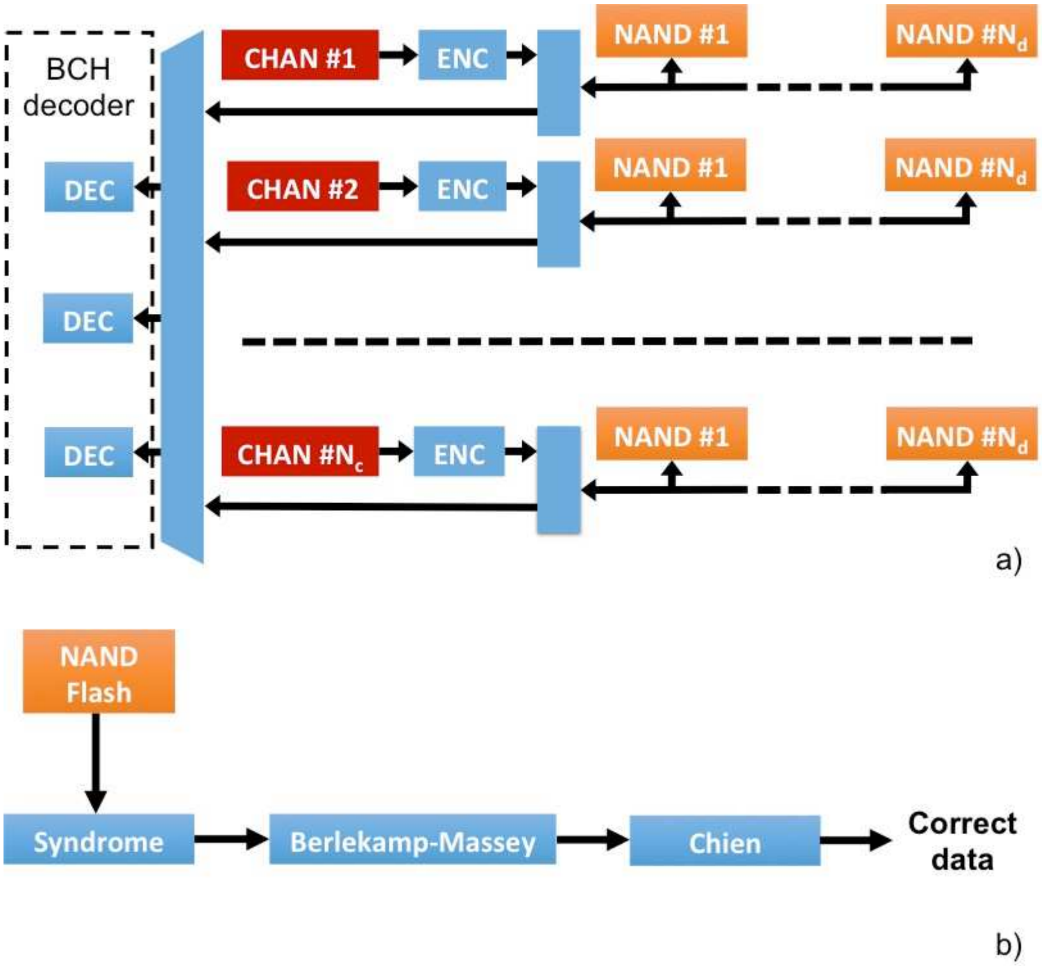


Figure 2.9: a): schematic representation of an ECC architecture based on BCH codes. A high-speed encoder is connected to each SSD channel whereas a a reconfigurable parallel decoder is shared among the N_c channels. b): schematic representation of the BCH decoder.

degradation (at the expenses, however, of both area occupation and power consumption) but it cannot solve the problems caused by uncorrectable pages.

To deal with the presence of uncorrectable pages, two alternatives exist: *i)* keep BCH codes and their ease of implementation while defining sophisticated read algorithms in order to reduce the number of errors [73, 74]; *ii)* develop ECC solutions based on different coding concepts, like Low Density Parity Check (LDPC) codes [75]. In the former case, the basic idea in the presence of uncorrectable pages consists in re-read the page with different read reference voltages, in the attempt of tracking the shift of the threshold voltage distributions. Such a solution led to the development of different read algorithms, generally defined as *read retry* [73]: they are automatically

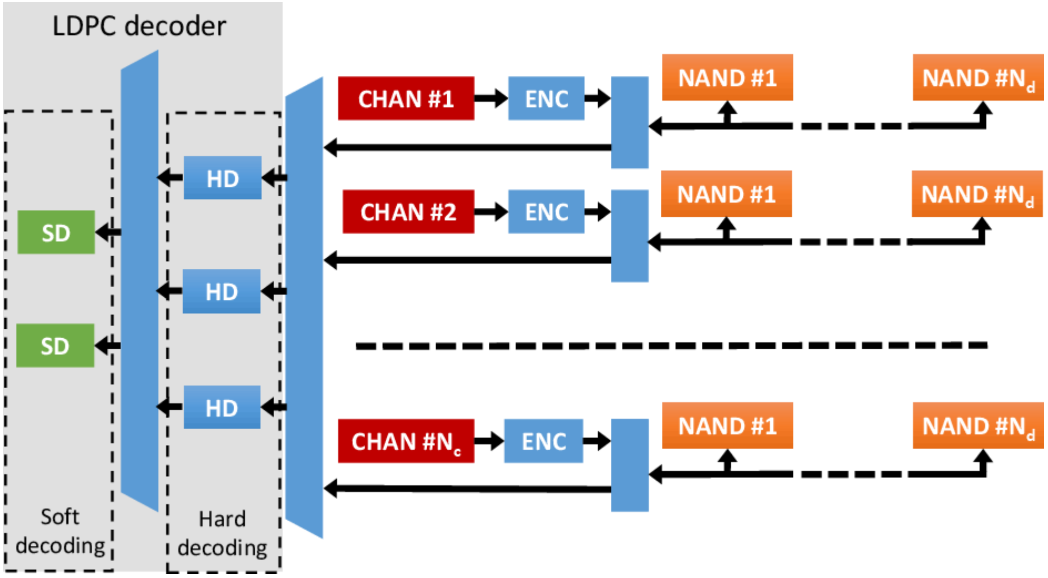


Figure 2.10: Schematic representation of an ECC architecture based on LDPC codes. The decoding path is composed by two main blocks: the hard decoding, whose architecture is similar to that designed for BCH engines and the soft-level decoding.

managed by the ECC engine and they call for (at least) a page re-reading with the unavoidable degradation of the read bandwidth. The latter solution adopts LDPC codes that, differently from BCH codes, present a much higher correction capability [75]. Fig. 2.10 shows the typical blocks for ECC engines based on LDPC codes: the decoding engine is composed by two main blocks: the Hard Decoding (HD) and the Soft Decoding (SD).

From an operative point of view, LDPC decoding works as follows. As shown in Fig. 2.4, multilvel NAND Flash memories are read page-wise by using a set of read reference voltages, hereafter denoted as HD_0 (see Fig. 2.11a showing the read reference discriminating between two adjacent threshold voltage distributions). Cells are read as 1 or 0 depending on their threshold voltage V_T with respect to HD_0. If during the ECC decoding phase the page is evaluated as uncorrectable, the LDPC decoding algorithm can be retried with the SD. To accomplish this second step, more information about the actual position of the NAND Flash threshold voltage distributions must be collected. Basically, the algorithm moves sequentially the internal read references to SD_{10} and SD_{11} (Fig. 2.11b) thus reading the page twice. Data are transferred to the LDPC decoder and then they are bit-wise combined with those previously read with

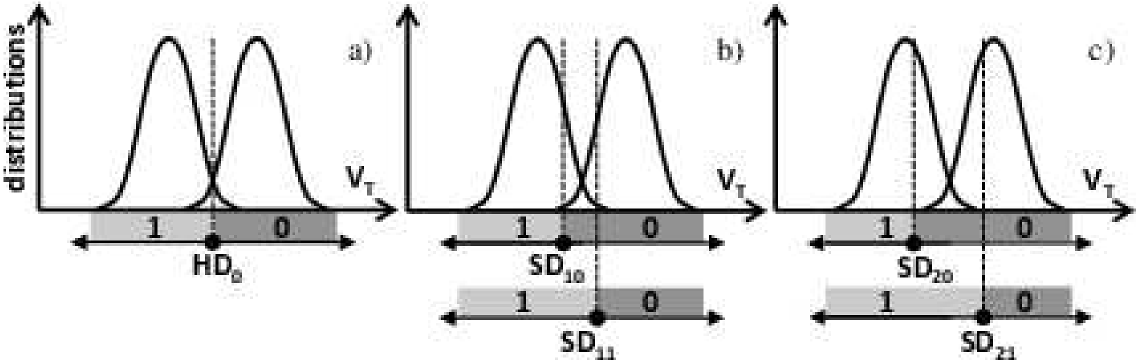


Figure 2.11: NAND Flash read references used in the two levels LDPC sensing scheme to discriminate between two adjacent threshold voltage distributions. A memory page is read by setting the read voltage at HD_0 and determining, for each bit, whether $\mathrm{V}_T < \mathrm{HD}_0$ or $\mathrm{V}_T > \mathrm{HD}_0$ (a). If the ECC engine is not able to correct possible read errors, the soft decision algorithm starts and the page is read twice by moving the read references around HD_0, to SD_{10} and SD_{11} (b). If the page is still marked as uncorrectable, the page is read again with the SD_{20} and SD_{21} references (c). Reprinted with permission from [76].

HD_0. This step is possible because during the whole SD process the data read with the HD_0 reference are stored in a dedicated buffer inside the SSD controller and used as a reference.

Thanks to this multiple read operation it is possible to calculate the information needed by the SD: the Log-Likelihood Ratios (LLRs) [7]. The calculated numbers are used as input for the soft decoder and are defined as follows:

$$LLR(y_i) = ln\frac{P(x=0|y_i)}{P(x=1|y_i)} = ln\frac{P(y_i|x=0)}{P(y_i|x=1)} \tag{2.1}$$

where P is the probability, whereas x and y_i represent the transmitted (i.e., the programmed value) and the received (i.e., the read bit) symbols, respectively [77]. As a matter of fact, when a set of read reference voltages is used (SD_{10} and SD_{11}), Eq. (2.1) defines that the LLRs can be viewed as the probability of reading a 0 or a 1 given the value of a specific programmed bit [7]. In other words, the higher the absolute value of the LLR is, the higher the confidence that the read bit is correct. [78].

An example of the result of the SD process is sketched in Fig. 2.12. As it can be

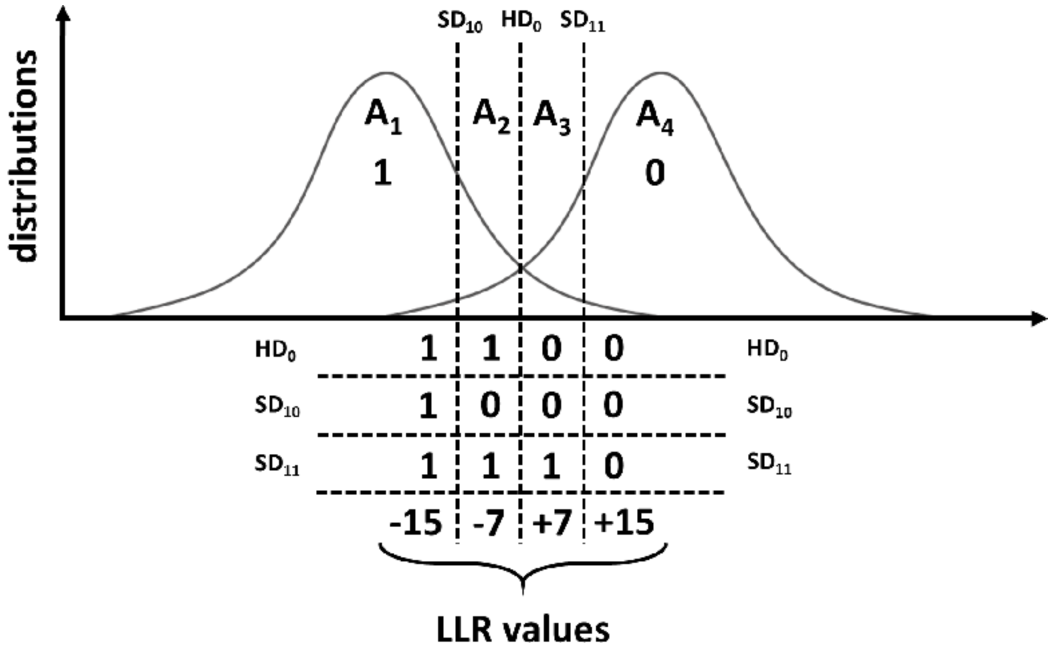


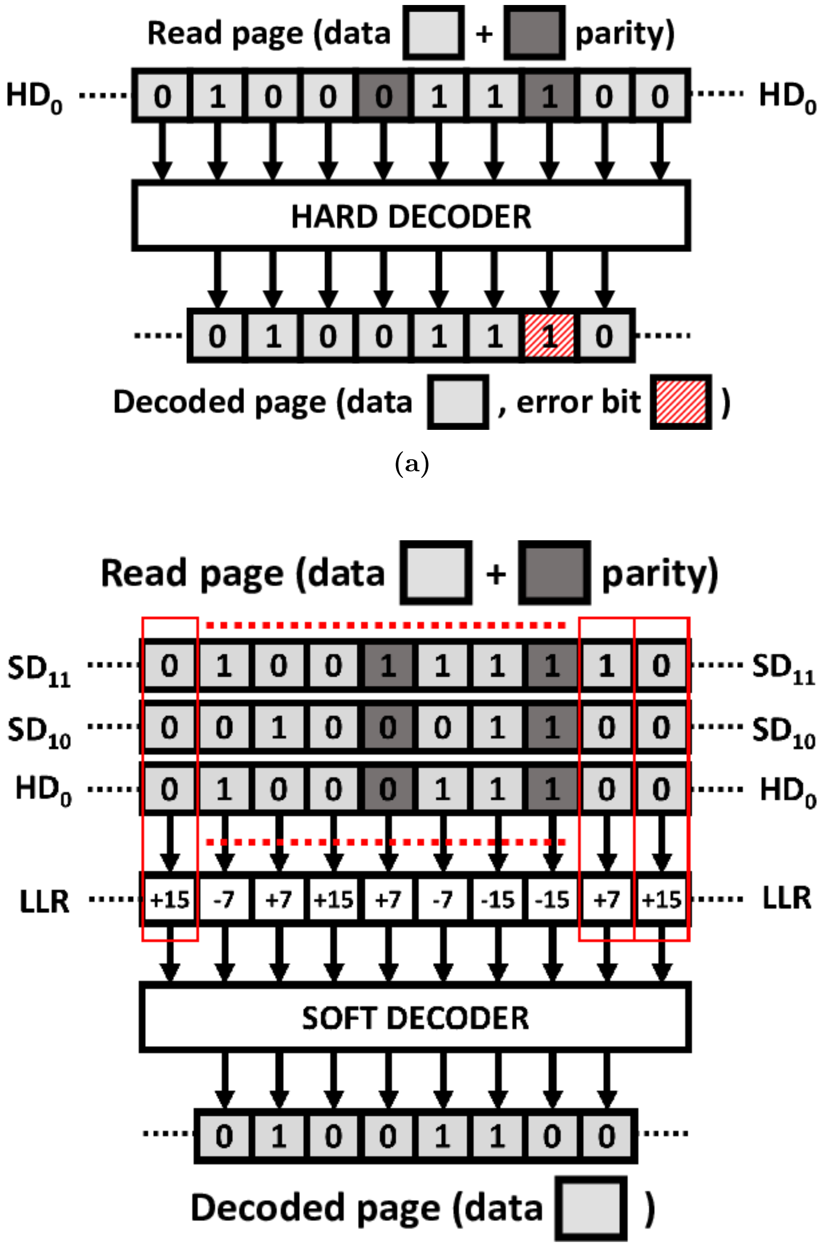
Figure 2.12: Voltage threshold partitioning performed during a one Soft Decoding (SD). Four regions (A1, A2, A3, and A4) are marked by HD$_0$, SD$_{10}$, and SD$_{11}$.

seen, the bit-wise combination of the data read from the NAND flash memory defines four different regions of the threshold voltage distributions. These represent *de facto* a probability density function of the programmed cells: the probability that a bit belongs to one of the areas (A_i with $i = 1, ..., 4$ in the example of Fig. 2.12) identified by the hard and the soft references is defined as follows:

$$P(X \in A_i) = \int_{A_i} p_X(x)dx \tag{2.2}$$

where X represents the programmed bit, and $p_X(x)$ is the actual threshold voltage distribution. At this point, it is clear that to extrapolate the LLRs expressed in Eq. (2.1) it is sufficient to calculate a bounded logarithmic ratio between the number of cells read as 0 and those read as 1.

Once the LLRs are calculated for all the regions, instead of using the raw bits coming from the NAND flash memory (HD decoding sketched in Fig. 2.13a), the SD decoder translates the bit-wise combination of the data read with HD$_0$, SD$_{10}$, and SD$_{11}$ with the corresponding LLR values, and it starts the decoding procedure (see Fig. 2.13b). At this point, a purely probabilistic decoding process is triggered.



(a)

(b)

Figure 2.13: Difference between the HD and the SD decoding phase. When HD is considered, raw bits coming from the NAND flash memory are used as input of the decoder. In this example after the decoding step one bit is still in error, therefore SD is required. When SD is considered, LLRs computed by the bit-wise combination of data read with HD_0, SD_{10}, and SD_{11} are used as input (see Fig. 2.12).

If the decoding process still fails, a second iteration is performed by moving the read references to SD_{20} and SD_{21} (Fig. 2.11c) and comparing the new read data

Table 2.1: LDPC and BCH features benchmark (data from [82]).

	BCH	**LDPC**
Decoding Algorithm	Algebraic-based	Probability-based
Guaranteed correction	Yes	No
Soft Bit Decoding	Hard (Read Retry)	Easy
Hard Decoding Performance	Code dependent	Similar to BCH
Soft Decoding Performance	-	2X-3X
Decoding complexity	Low	High
Power consumption	Medium	High
Cost	Low	High

with those previously analyzed and stored in the dedicated buffer. In this case the number of regions defined by the read voltage references switch from 4 to 6, therefore the LLRs values must be computed again by the decoder. The algorithm continues this process until the page is correctly read or the maximum number of soft-levels is reached and the page is marked as uncorrectable [22]. Finally, since LDPC codes provide a probabilistic correction, they are not immune from errors like false-decoding that occurs when the ECC performs erroneous correction while declaring successful decoding [79]. The presence of false-decoding errors is strictly related to the LDPCs mathematical characteristics and, therefore, it is essential to identify *a priori* the algorithm minimizing these errors [70].

LDPC codes, although presenting much higher correction capabilities with respect to BCH, can still fail the correction process in presence of pages with large numbers of errors. Also in these cases there exist re-reading algorithms (for instance the *multiple soft decision*) that can correct pages initially marked as uncorrectable at the expense of the overall reading time [80, 81, 76]. Table 2.1 summarizes the features of LDPC and BCH described in this section.

To evaluate the optimal ECC engine design in terms of HD and SD implementation, the knowledge of the actual memory RBER is mandatory. With this respect, it is usual to leverage a *worst-case* design methodology where the correction strength figure of the HD is compared with the maximum percentage of uncorrectable pages measured at the end of the memory's lifetime. Fig. 2.14 shows this process when a

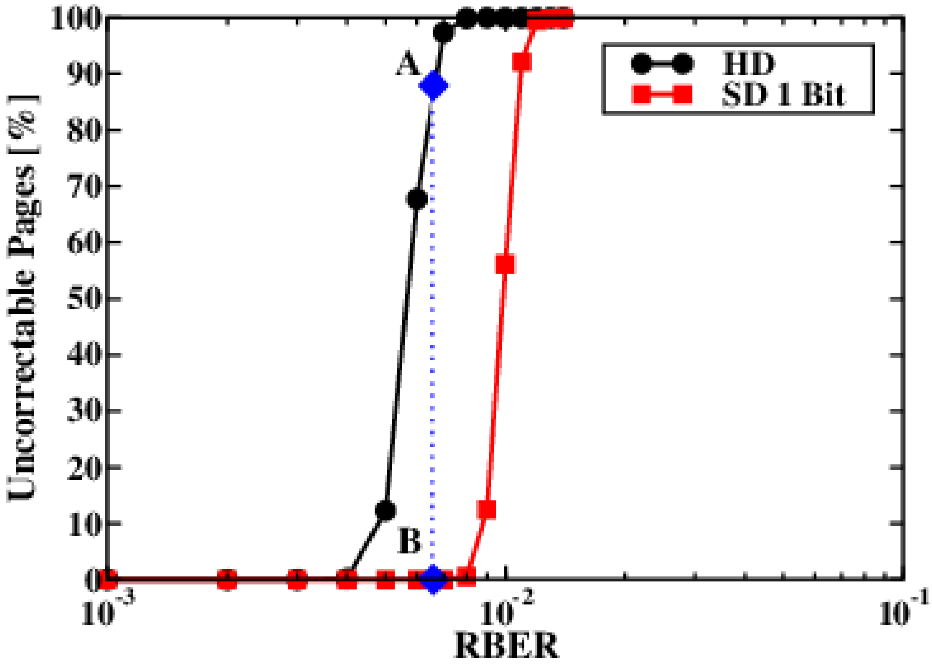


Figure 2.14: Correction strength of both HD and SD when a LDPC able to correct up to 100 Bits in a 4320 Bytes codeword is considered for a 128-Gbits TLC NAND Flash memory manufactured in a planar 1x-nm technology node. Points A and B represent the maximum measured percentage of uncorrectable pages at the end of the memory lifetime, when HD and SD are used, respectively.

LDPC able to correct up to 100 bits in a 4320 Bytes codeword is considered for a TLC NAND Flash memory manufactured in a planar 1x-nm technology node. Point A marks the maximum percentage of uncorrectable pages measured at the end of the memory's lifetime. As it can be seen, in this case switching from the HD to a one bit SD is sufficient to correct all the errors (point B). Other correction strategies like a two bits SD, become an over-design.

The above considerations are mandatory when it is required to design the optimum LDPC architecture (both in terms of correction strength and correction bandwidth) for the target SSD. In fact, since the SD directly impacts the drive's bandwidth, once the correction strategy is defined (a one bit SD rather then a two bits SD) and the decoder's bandwidth is fixed, it is important to find the right balance between the number of HD and SD decoders. Fig. 2.15 shows the read bandwidth obtained, for different HD implementations, in a 2 TB SSD featuring 16 channels each one

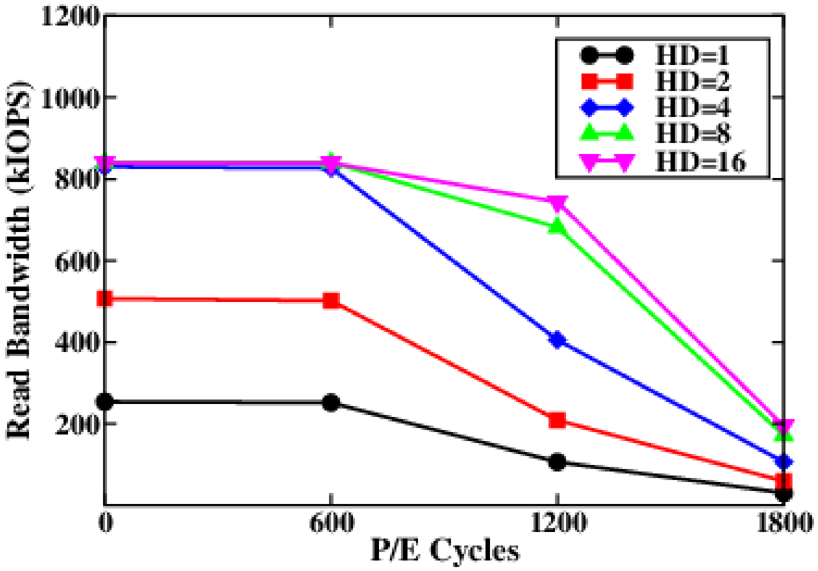


Figure 2.15: Read bandwidth evolution as a function of the number of P/E cycles in a 2 TB SSD featuring a PCI-Express GEN3x4 host interface and 16 channels each connected to eight 128-Gb TLC NAND Flash dies manufactured in a planar 1x-nm technology node with a rated endurance of 900 P/E cycles. The ECC engine is composed by a variable pool of HD decoders and a single SD decoder. Each hard decoder has a decoding bandwidth of about 1.2 GB/s.

connected to eight 128-Gbits TLC NAND Flash dies manufactured in a planar 1x-nm technology node, as a function of the number of P/E cycles. The correction strategy used in this example is the same sketched in Fig. 2.14, therefore, a 1 Bit SD has been used. All results have been obtained by using the SSDExplorer simulator [26]. Since each hard decoder has a bandwidth of 1.2 GB/s and the SSD host interface is a PCI-Express GEN3x4 [25] with a maximum bandwidth of 4 GB/s, it is clear that a coarse design choice (that neglects the actual RBER evolution) requires 4 HD decoders and any higher number would result in a cost ineffective overdesign.

However, since RBER increases with the number of P/E cycles (see Fig. 2.6), the percentage of uncorrectable pages detected by the HD increases as well. As a consequence SD is triggered and the read bandwidth rapidly decreases when the memory rated endurance (P/E = 900) is approached. To guarantee the expected performance and to extend the SSD working window, it is necessary to increase the number of HD decoders (see Fig. 2.15) as well as that of SD decoders. Fig. 2.16a shows

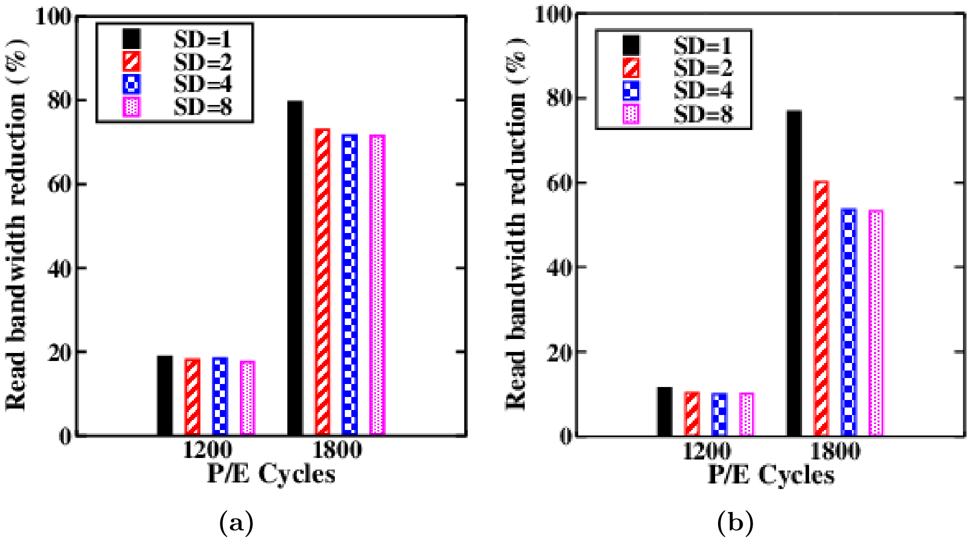


Figure 2.16: Read bandwidth degradation with respect to the beginning of life at P/E = 1200 and P/E = 1800 (*i.e.* at twice the rated endurance) considering different SD levels. 8 and 16 HD decoders have been considered in Fig. a) and b), respectively.

the calculated read bandwidth degradation with respect to the beginning of life at P/E = 1200 and P/E = 1800 (i.e., at twice the rated endurance) by implementing 8 HD decoders and different numbers of SD decoders. As it can be seen, to reduce the read bandwidth degradation at twice the rated endurance, 2 SD decoders can be used, while any larger number of decoders would result in an overdesign. Fig. 2.16b shows the results obtained by using 16 HD decoders and different numbers of SD decoders, showing a significant performance improvement thanks to a much higher hardware cost. From a designer point of view, an accurate trade-off evaluation between performance (*i.e.* read bandwidth reduction) and hardware cost must be based on the actual knowledge of the memory RBER evolution.

By summarizing the previous reasonings, in multilevel Flash memories the use of sophisticated ECC architecture is mandatory in order to efficiently correct a number of errors that increases with the memory endurance and with the time elapsed between two successive read operations of the same page. These ECC engines, however, strongly impact on the read bandwidth and latency. This holds true, in particular, when uncorrectable pages are detected, since advanced read algorithms are required. Therefore, the choice of the ECC code to be implemented and of its correction capability, the design of the ECC engine architecture, and the identification of the most effective re-reading algorithm depend on the memory reliability and, in particular, on the BER whose value grows with the memory wear-out.

The optimal design of the reading path for a *delay insensitive* SSD must be based on the accurate knowledge of the performance and reliability of the selected memories and, therefore, on a careful pre-characterization of the memories themselves in order to estimate their BER [83].

2.3 SSD controller design

The main block diagram of an SSD controller is shown in Fig. 2.17. Once the SSD's specifications have been fixed, and hence the maximum device bandwidth has been defined, the SSD controller design follows a simple rule of thumb to calculate N_c and N_d needed to meet the requirements. Basically, to calculate the actual controller bandwidth B_{cont}, it is sufficient to sum the bandwidth contributions B_{ch} of each channel:

$$B_{cont} = \sum_{i=1}^{N_c} B_{ch_i} \ . \tag{2.3}$$

The maximum channel bandwitdth $B_{ch_i}^{max}$ is obtained under the assumption that all the memory dies connected to channel i are addressed at the same time. By defining B_d as the bandwidth of each memory die, the theoretical controller bandwidth B_{cont}^{th} is given by:

$$B_{cont}^{th} = \sum_{i=1}^{N_c} B_{ch_i}^{max} = \sum_{i=1}^{N_c} N_{d_i} B_d \ . \tag{2.4}$$

Eq. (2.4) represents, however, the theoretical condition under the hypothesis that all single dies can communicate simultaneously with the controller and, therefore, it represents the maximum achievable value. Unfortunately, for several reasons (e.g., access request to the same die, die's response time slowed down by a read retry operation, die busy for a program operation whose latency is much higher with respect to read latency, etc.), the probability that all dies can communicate simultaneously with the controller is generally < 1. Taking into account that a number n of dies in a channel cannot serve new requests since they are processing other commands, the

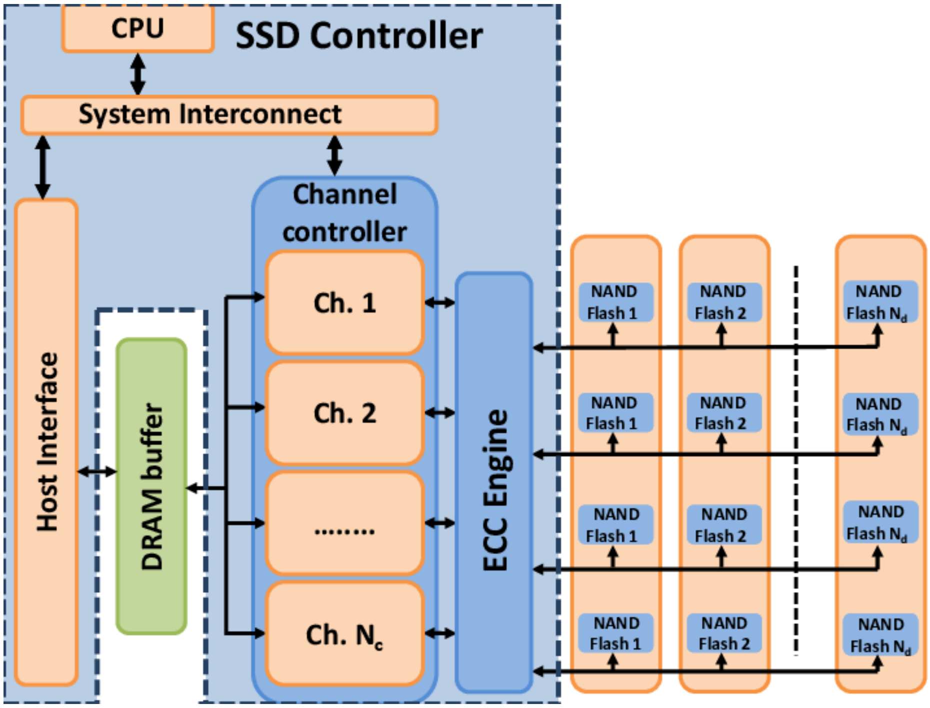


Figure 2.17: Schematic representation of the SSD controller, considering N_c channels and N_d memory dies connected to each channel.

actual controller bandwidth is given by:

$$B_{cont} = \sum_{i=1}^{N_c} (N_{d_i} - n_i) B_d \leq B_{cont}^{th} \ . \tag{2.5}$$

The above equation calculates the controller bandwidth in a fresh condition (i.e., at the beginning of the drive's lifetime). However, as previously shown in Section 2.2, the actual performance of the SSD is strongly affected by the reliability phenomena associated with the storage layer. As a consequence, to take into account these effects, Eq. 2.5 can be modified as follows:

$$\begin{gathered} B_{cont}(P/E, T, RD, WAF) = \\ = \sum_{i=1}^{N_c} (N_{d_i} - n_i(P/E, T, RD, WAF)) B_d \leq B_{cont}^{th} \end{gathered} \tag{2.6}$$

where P/E, T, RD and WAF are the current Program/Erase cycle number of the drive, the working Temperature, the Read Disturb level of the memories, and the Write Amplification Factor, respectively. The WAF factor is defined as

$$WAF = \frac{data\ written\ to\ the\ NAND\ flash}{data\ written\ by\ the\ host} \geq 1 \ ; \tag{2.7}$$

it has been accurately described in [84] and it depends on several factors ascribed to the FTL implementation including Wear Leveling, Garbage Collection, and Bad Block management algorithms. Along with WAF, P/E, T, and RD introduce hard-to-model effects that complicate the description of the controller's bandwidth in a *closed form*. Therefore, to help SSD designers to calculate the actual performance and latency of a target SSD over time and use, the adoption of sophisticated simulation tools like SSDExplorer is mandatory [26].

Overall, what ultimately stands out from both Eq. (2.5) and Eq. (2.6) is that, to approach as much as possible the ideal controller bandwidth, it is necessary to:

- reduce the probability that a command addresses a busy die (i.e., a die already scheduled by another operation);
- maximize the number of dies that can process a new command.

This can be accomplished: *i*) by increasing the number N_d of dies connected to each channel, which however impacts on the SSD cost; *ii*) with an effective command management performed by the FTL; *iii*) by using a DRAM as a data buffer.

2.3.1 Efficient command management

In nowadays SSDs, to efficiently manage the commands issued by the host, it is possible to leverage the *Command Queue* (CQ) concept [85]. This resource is usually implemented as a software routine shared between the host interface, which pushes host commands inside the CQ, and the SSD controller which manages the requested operations and pulls out the commands from the CQ.

Fig. 2.18 shows the queuing hierarchy usually implemented in traditional SSD controllers [86]. Besides the external host CQ, it is common to have a dedicated small command queue for each NAND Flash memory die: the *Target Command Queue* (TCQ). Basically, thanks to the TCQ, the host can continue to issue commands even when it tries to read or program a die which is in the busy state. In fact, when this condition is verified, the command is simply queued in the TCQ and the SSD controller can continue to fetch other commands from the host CQ. This technique allows maximizing B_{cont} since TCQs keep always busy all the NAND Flash dies. It is thus clear that the main parameters controlling B_{cont} are the parallelism (i.e., N_c

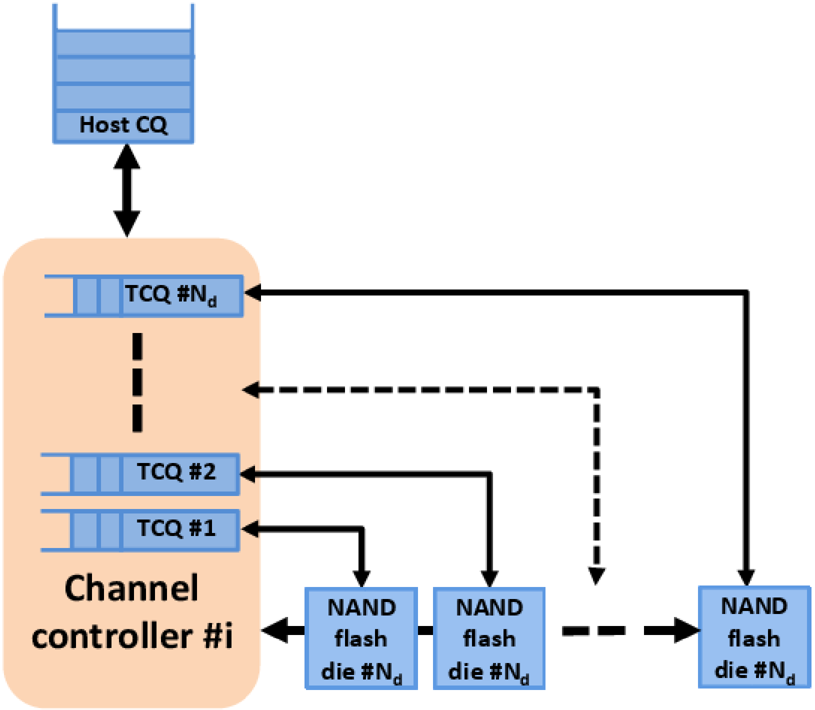


Figure 2.18: Queueing hierarchy implemented inside the SSD controller for a generic channel

and N_d) and the queue depth (QD), that is the number of commands that the host interface can store.

The attempt of approaching the ideal performance in terms of bandwidth by increasing QD presents an unavoidable disadvantage: the increase of the service time (*i.e.* the time elapsed between the issue and the execution of a command) and, consequently, of the SSD latency. Therefore QD has a severe impact on QoS, that basically defines the maximum acceptable latency of the drive and it is calculated as the 99.99-th percentile of the SSD latencies cumulative distribution. To this extent, QoS is used to quantify how the SSD behaves in the worst-case conditions [9]. By using this metric it is possible to understand if the target SSD architecture is suitable for a specific application, such as real-time and safety-critical systems [87]. Fig. 2.19 shows an example of how B_{cont} and QoS scale with the host QD. As expected, both B_{cont} and QoS increase with QD. This behavior, however, is in contrast with the requirements of high performance SSDs, which ask for achieving the target bandwidth with the lowest QoS. In fact, state-of-the-art user applications such as financial transactions or cloud platforms [88, 89] are designed to work with storage devices which have to serve an I/O operation within a specific time-frame which is usually upper-bounded by the QoS requirement.

To deal with this requirement it is possible to use the *Head-of-Line* (HoL) blocking concept, whose effect is to limit the number of outstanding commands inside the SSD, thus partially solving the latency issue [90]. The HoL blocking is managed by the controller firmware implementing a FIFO stack whose dimensions can be dynamically defined. When the number of commands queued in a TCQ exceeds a predefined threshold, it is possible to trigger a blocking state inside the SSD controller which stops the submission of a new command from the host CQ. In such a way, depending on the HoL threshold value, it is possible to avoid long command queues inside the TCQs and, hence, the device QoS can be limited within a defined window.

Figure 2.20 shows the effectiveness of the HoL blocking for the case analyzed in Fig. 2.19. As soon as the target performance of 300 kIOPS is reached (QD = 64), the HoL blocking effect starts keeping the QoS below the target requirements even when long QDs, such as QD = 128 and QD = 256, are considered.

The fine-grained QoS calibration made available by the HoL blocking, however, does not come for free. If, besides B_{cont} and QoS, the average SSD latency is taken into account, it is clear that the HoL blocking effect has to be wisely used (see Fig. 2.21). When the HoL blocking is triggered, it trades the QoS reduction with an increase of the average latency. Moreover, this behavior becomes more pronounced when high QDs are used, *i.e.* when a higher QoS reduction is required.

Summing up, it becomes clear that the performance optimization process that has to be followed by SSD designers must involve the optimization of the bandwidth, the average and the maximum latency, the length of the command queue, the command management policy, the head of line blocking, all considered at the same time.

2.3.2 DRAM data caching

To increase the controller bandwidth and to approach as much as possible the theoretical bandwidth B_{cont}^{th}, it is possible to use a DRAM as data cache buffer [28]. As shown in Fig. 2.17, this block is located between the host interface and the channel controller. Standard data caching algorithms can be adopted, such as Least Recently Used (LRU) or Least Frequently Used (LFU) [91], to decrease the number of accesses to the Flash memories. Since data are addressed in a much faster memory, the access

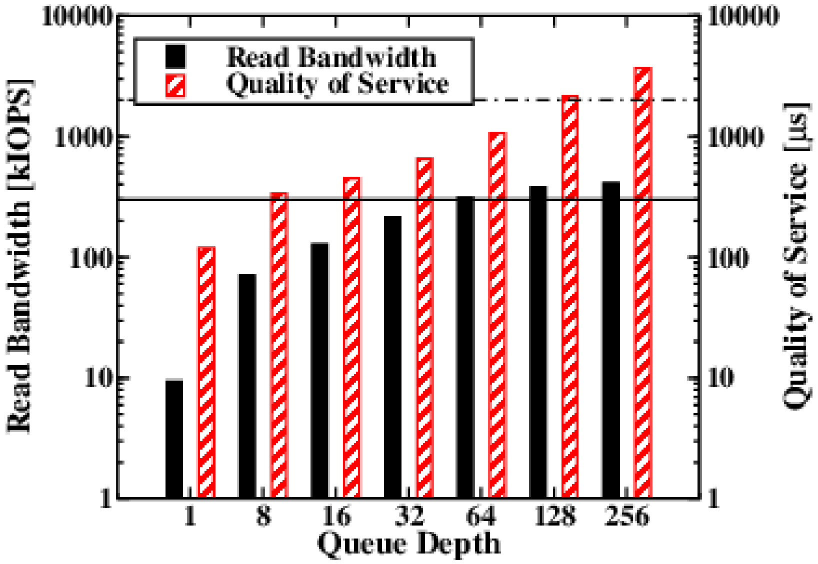


Figure 2.19: B_{cont} and QoS as a function of the host Queue Depth. The full line and the dashed-dotted line represent the target B_{cont} and the target QoS, respectively. Simulations refer to an SSD featuring $N_c = 8$ and $N_d = 8$ TLC NAND Flash manufactured in a planar 1x technology node. Average read time is 86 μs and workload is 100% 4 kB random read.

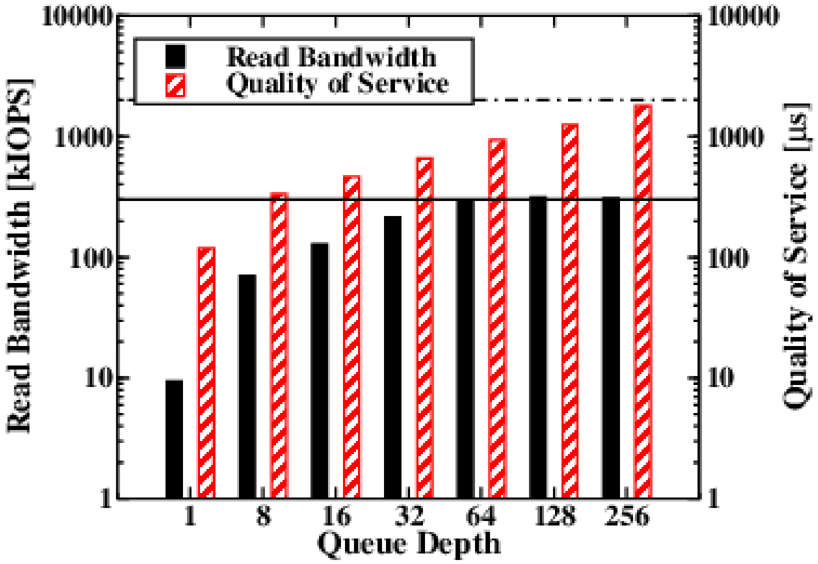


Figure 2.20: SSD bandwidth and QoS for the same case of Fig 2.19 as a function of the host queue depth when HoL blocking is used. The full line and the dashed-dotted line represent the target B_{cont} and the target QoS, respectively.

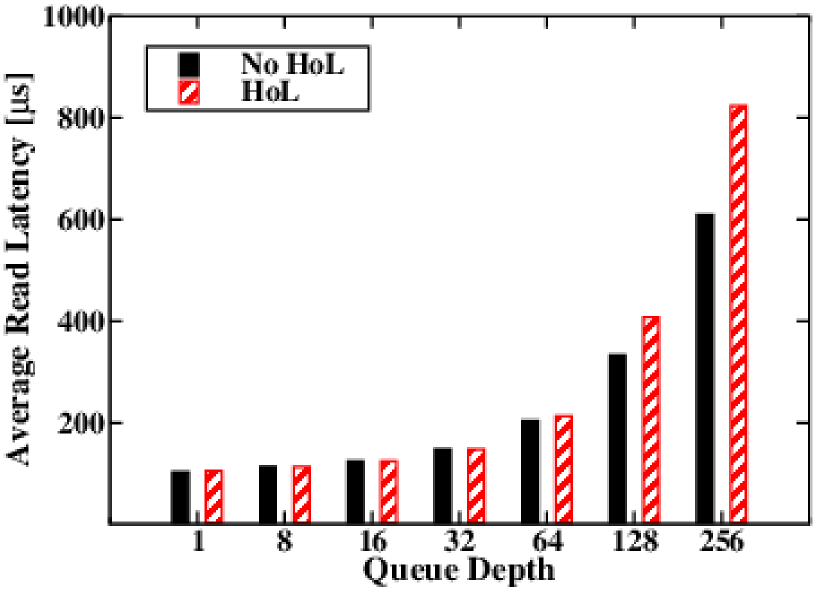


Figure 2.21: Average SSD latency evaluated as a function of the host queue depth, for the same case of Fig 2.19, with and without the HoL blocking.

time can be reduced with respect to a standard NAND Flash read/program operation. In addition, since part of the data to be read/written are stored in the DRAM buffer, the number of accesses to the NAND Flash dies are reduced, thus limiting the number of busy dies.

These effects positively impact the SSD bandwidth and the average latency. Moreover, the reduction of the number of accesses to the NAND Flash dies increases their reliability. This point is strictly related to the smaller number of write operations, thus limiting endurance effects and, possibly, leading to a reduced read disturb issue (see Section 2.1).

Table 2.2 shows the cache hit probability, the read bandwidth, the average latency, and the QoS calculated for the "no cache" case (i.e., a case where the DRAM data cache buffer is not present, assumed as reference) and for different ratios between the total NAND and the DRAM sizes. The number of cache hits (*i.e.* the percentage of memory accesses to the DRAM buffer with respect to the total number of data accesses) depends on the probability of addressing any single nonvolatile memory page. All data have been collected considering a uniformly distributed Logical Block Address (LBA) space of the SSD and a LRU eviction policy is used as caching algorithm.

As it can be seen, the perfomance metrics of the simulated drive are not significantly influenced by the DRAM size. This is due to the fact that the LBA space is

Table 2.2: NAND/DRAM size ratio and SSD performance for the same configuration of Fig. 2.19 considering an uniformly distributed LBA space.

NAND/DRAM size ratio	No cache	256	50	15
Cache hit probability [%]	0	0.6	2.7	8.2
Read Bandwidth [kIOPS]	301	312	318	337
Average latency [μs]	206	204	200	189
QoS [ms]	1.07	1.19	1.13	1.03

uniformly distributed across all the SSD pages, therefore all data locations have the same probability to be addressed.

An uniformly distributed LBA space, however, represents the worst-case condition for the assessment of the benefits materialized by a caching algorithm. In general real user workloads tend to follow different LBA distributions which are more similar to a Gaussian or a Log-Normal with a mode around a specific address. As a consequence, if the I/O address profile of the target application is known, it is possible to optimize the DRAM cache size depending on the statistical parameters presented by the LBA profile itself.

Fig. 2.22 shows three examples of Gaussian workloads spanning across the whole LBA space of the drive. By considering a $\pm\sigma$ deviation around the average of the total SSD LBA address space, it is possible to design the proper DRAM size ratio in two different ways:

- reducing the DRAM capacity while keeping the same cache hit probability and drive performance;
- increasing the DRAM capacity maximizing the number of cache hits and, therefore, boosting the drive performance.

Table 2.3 shows, for the three cases of Fig. 2.22, the NAND/DRAM size ratio, the cache hit probability, the read bandwidth, the average latency, and the QoS of the target SSD architecture. As it can be seen, the performance metrics are almost similar with a significant reduction of the DRAM size for the tightest workload distribution of Fig. 2.22.

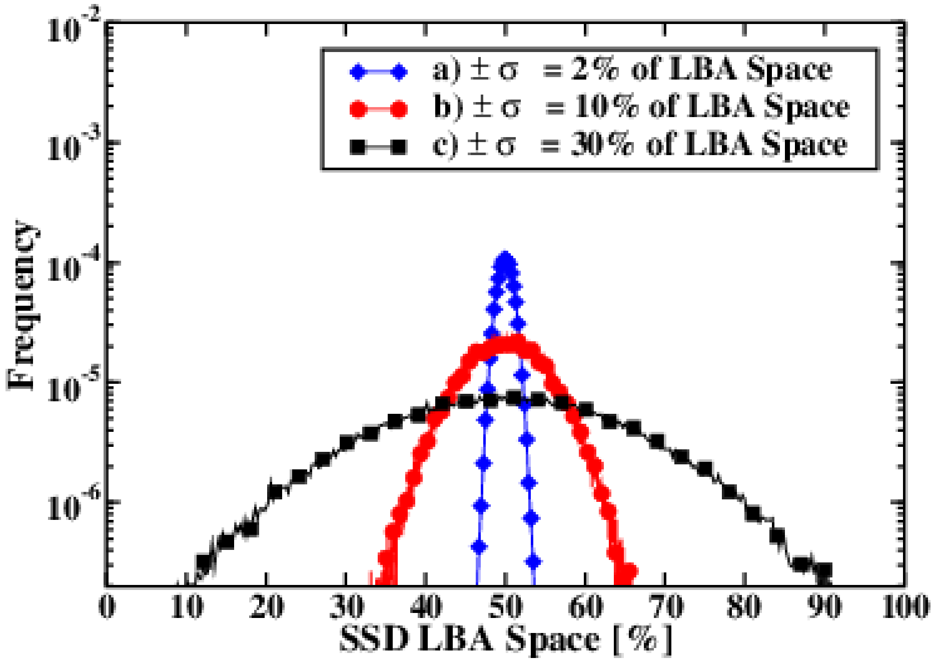


Figure 2.22: Examples of three gaussian distributions of the I/O addressing space. The median of the distributions is placed at the the 50% of the SSD LBA addressing space in all cases.

Table 2.4 shows, for case b) shown in Fig. 2.22, the NAND/DRAM size ratio, the cache hit probability, and the performance metrics of the target SSD architecture. With respect to case b) of Table 2.3 the NAND/DRAM size ratio has been reduced from 50 to 15. As it can be seen, it is possible to almost triplicate the cache hit probability thus increasing the read bandwidth while reducing the average latency. It is worth to highlight that this performance improvement marginally impacts the QoS, since it is related to the worst case (usually a read operation performed on a NAND Flash die.)

Summing up, the use of a DRAM cache offers advantages in terms of bandwidth, latency, and reliability. The design of an application specific SSD, in addition, can be optimized if the LBA space distribution is known, in order to reduce the DRAM size. Therefore, the drive design must be done concurrently with the application for which it represents the storage element. This concept, leading to the development of Software Defined Flash (SDF, [29]), will be extensively treated in Section 2.5.

Table 2.3: NAND/DRAM size ratio and SSD performance for the same configuration of Fig. 2.19 as a function of the LBA space distributions of Fig. 2.22.

Case	a)	b)	c)
NAND/DRAM size ratio	256	50	15
Cache hit probability [%]	15.3	15.3	15.3
Read Bandwidth [kIOPS]	367	364	365
Average latency [μs]	173	175	175
QoS [ms]	0.98	1.27	1.29

Table 2.4: NAND/DRAM size ratio and SSD performance for the same configuration of Fig. 2.19 as a function of the LBA space distribution of the case b) of Fig. 2.22.

NAND/DRAM size ratio	50	15
Cache hit probability [%]	15.3	42.1
Read Bandwidth [kIOPS]	364	536
Average latency [μs]	175	118
QoS [ms]	1.27	1.19

2.4 Criteria for optimal host interface selection

The host interface represents the link between the SSD controller and the host where the application is running. Differently from the SSD controller that is fully customized, the physical structure of the communication interface follows consolidated standards. At the moment, the used interfaces are SATA [8] (mainly for consumer applications), SAS [24], and PCIe [25] (for enterprise environments).

The correct choice of the host interface represents a crucial aspect along the drive design phase since it allows guaranteeing that the SSD controller is used in optimal conditions. In a traditional design approach for general purpose SSDs, where both controller and host interface are chosen separately without any knowledge of the final application, the constraint of selecting a host interface able to guarantee a bandwidth

$B_{hi} \geq B_{cont}$ (where B_{hi} is the maximum bandwidth of the host interface) at the lowest cost represents the standard approach, whereas a host interface whose $B_{hi} < B_{cont}$ would act as a bottleneck limiting the SSD performance. A detailed analysis of the impact of the host interface on the SSD's performance has been presented in [26].

If the application to be run on the host is known, a different approach can be adopted. It must be taken into account that the design of a fully customized SSD controller is much more expensive with respect to that of the host interface, which follows well defined standards [86]. By considering this economic aspect, it is convenient to design an SSD controller with top performance (rather than a family of controllers with different quality metrics) and to operate at the host interface level to satisfy the application requirements. As an example, if the controller has been designed to sustain a certain B_{cont}^{th} and the application requires a lower bandwidth B_{app}, an interface satisfying the condition

$$B_{app} \leq B_{hi} \leq B_{cont}^{th} \tag{2.8}$$

can be selected, confirming that the ideal host interface must be chosen on the basis of the application and, therefore, on the drive use. In such a way, with a single SSD controller design, different application requirements can be satisfied by using different host interfaces. Such methodology allows reducing the controller bandwidth to match that of the application and lowering the design cost of the SSD controller. In addition, it allows also reducing the drive power consumption since, operating at a lower throughput, a lower number of NAND Flash dies are activated simultaneously.

An evolution of this design methodology, envisaging a single controller associated to different interfaces as a function of the application, considers an unique combination of SSD controller and host interface. In this case, each block is able to provide the maximum theoretical performance. The effective performance, however, can be tuned dynamically at software level by acting on the SSD's firmware and especially on the command queue depths, which can be modified during the normal execution. An example of this methodology can be found in [92, 93] where the SSD controller is able to automatically limit the performance of the drive depending on the allowed power consumption or on the thermal dissipation level. Such an approach, that calls for the design of a single block embedding the SSD controller and the host interface, however, implies a higher design cost for the development of a controller whose hardware resources can be programmed by the user.

2.5 Research scenario opened by hardware-software co-design

In the last 40 years all software applications and Operating Systems (OS) which make use of persistent storage architectures have been designed to work with HDDs [1]. However, SSDs are physically and architecturally different from HDDs so that they need to execute the FTL algorithm to translate host commands [3, 4, 5]. Basically, the main role of FTL is to mimic the behavior of a traditional HDD and to enable the usage of SSDs in any electronic system without acting on the software stack. Besides this translation operation, SSD controllers have to run garbage collection, command scheduling algorithms, data placement schemes, wear-leveling, and errors correction. All these routines, even if on the one hand allow a ”plug and play” connection of the SSD with traditional hardware and software, on the other hand they limit actual SSD performance. The main drawback of FTL is the *Garbage Collection* (GC), that is performed when valid pages belonging to a block to be erased are read and written in a different block. Such an operation, that is time and power consuming, reduces both drive bandwidth and NAND Flash reliability [84]. In the enterprise market and hyperscale data centers, performance and reliability losses induced by GC are not tolerable.

To deal with the above mentioned challenges, software developers of hyperscale data centers have shown, in the past few years, a growing interest for *Software-Defined Flash* (SDF) [29]. In this kind of environments the driving forces in the design of computational nodes are reliability and high performance: therefore, even the I/O management has to be re-architected. SDF leverages a new SSD design approach called *Host-Based* FTL (HB-FTL) which allows the host system to:

- optimize the host payload, i.e., the amount of data read/written with a single command and hence relieve the SSD from any host command translation or manipulation;
- remove the GC related to FTL execution;
- execute the FTL directly on top of its computational node (*Open-Channel* architecture [94]).

2.5.1 HB-FTL operations

HB-FTL considers the migration of all FTL routines from the SSD to a more powerful processor located outside the SSD. To this purpose, the processor must be able to issue commands to be interpreted directly by the NAND Flash dies, such as read, program and, especially, erase [95]. In this context, a new protocol called *Light NVME* (LNVME) [96] allows a native communication between NAND memories and the external processor. Thanks to this protocol, the FTL can be implemented and executed by the external processor such as the host where the application is running.

A first advantage provided by this approach concerns the optimization of the host payload. With this respect, since ECC coding/decoding operate on an entire memory page, read/write operations on a NAND Flash page must follow the constrains imposed by the ECC itself. As an example, consider a NAND Flash memory whose page size is 4 kB and a host reading/writing data on a 512 B basis.

Write operations are performed on the NAND memories only when eight 512 B data chunks have been transferred by the host. However, the host considers as accomplished a write operation when the SSD has acknowledged the data acquisition. If a power fail occurs between the data load and the effective storage in the nonvolatile layer, data are considered as lost. To avoid this occurrence, dedicated solutions such as super-capacitors [97] or the introduction of emerging nonvolatile technologies, such as MRAM [98], replacing DRAM buffers can be adopted [99, 100]. On the contrary, a NAND memory page is read every time the host requires even a single chunk. Therefore, even if only 512 B are requested by the host, the entire 4 kB page is read and decoded by the ECC. It is clear that, in this case, the SSD is operating at 1/8 of its theoretical read bandwidth.

To improve the SSD performance and to better exploit its internal resources, it is convenient to co-design the application payload with the ECC engine. The optimal solution is achieved by data chuncks that are an integer multiple of the actual ECC codeword.

A more powerful approach takes into account that in HB-FTL-based SDF both the application and the FTL are processed in the same software environment [101]. Therefore, they can be co-designed in order to optimize the access pattern to the nonvolatile memory. As an example, the application can be designed to perform only sequential accesses to the storage medium, respecting the physical *in-order-program* of NAND Flash memories [102]. By following this approach, the actual access to the NAND Flash dies is *block-based* rather than *page-based* which is typical of random write accesses. By moving the write granularity from pages to blocks, GC is no longer necessary. In addition, by serializing the write traffic to the NAND Flash memories, the write bandwidth is maximized.

2.5.2 The Open-Channel architecture

The Open-Channel architecture [94, 101] allows implementing the management of HB-FTL-based SDF.

Fig. 2.23 sketches a template architecture that can be modeled by Open-Channel. Basically, thanks to the PCI-Express interconnection and the LNVME protocol, a bunch of NAND Flash cards can establish a peer-to-peer communication with the host processor without requesting any specific management to the SSD controller [103]. In

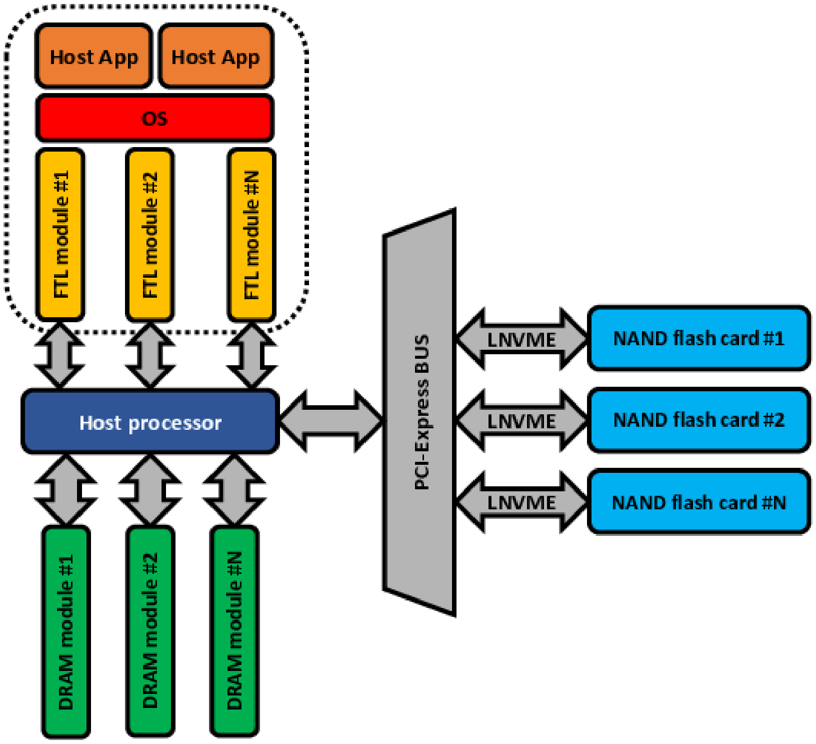


Figure 2.23: Reference architecture modeled by the Open-Channel storage layer when the host processor is used for HB-FTL execution. More than one NAND Flash card are connected to the PCI-Express bus. Different FTL modules are executed by the host processor.

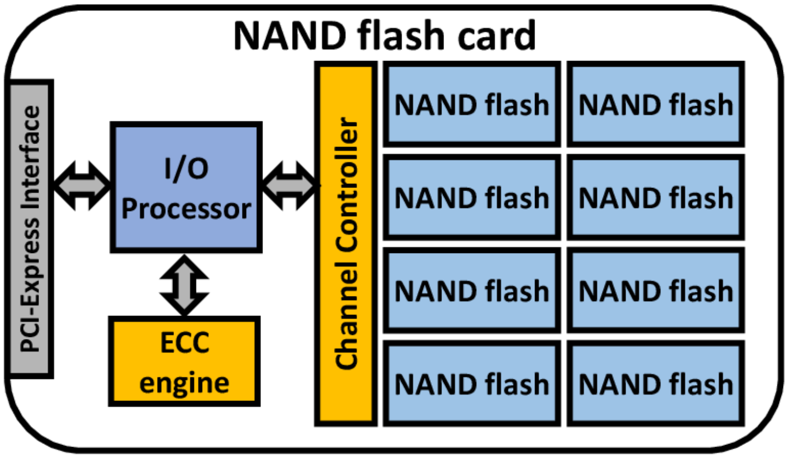


Figure 2.24: Schematic of a NAND Flash card used in the Open-Channel storage system.

Table 2.5: HGST SN150 Utrastar configuration.

Parameter	Configuration
Channels	16
Dies per channel	16
SSD Capacity	3.2 TB
NAND Flash die	128 Gb Toshiba A19 eMLC
Host interface	PCI-Express GEN3x4

this architecture ”NAND Flash cards” are not standard SSDs because, besides a simple I/O processor, a channel controller for NAND addressing and an ECC engine, they do not embody any complex processor, DRAM or even FTL (see Fig. 2.24). As a consequence, data read/write from/to these cards have to be considered as the raw output/input of NAND memories without any further manipulation.

Fig. 2.25 shows the effectiveness of HB-FTL with respect to a standard FTL in increasing the SSD performance. To this purpose the HGST SN150 Ultrastar SSD [104], whose configuration is reported in Table 2.5, has been compared with a simulated drive feauturing a HB-FTL approach and the same SSD configuration.

The comparison has been performed for different mixed workloads, from a 100% 4 kB random read, 0 % random write to a 0 % random read, 100 % 4kB random write.

All results show that in a standard FTL-based SSD performance decreases with the

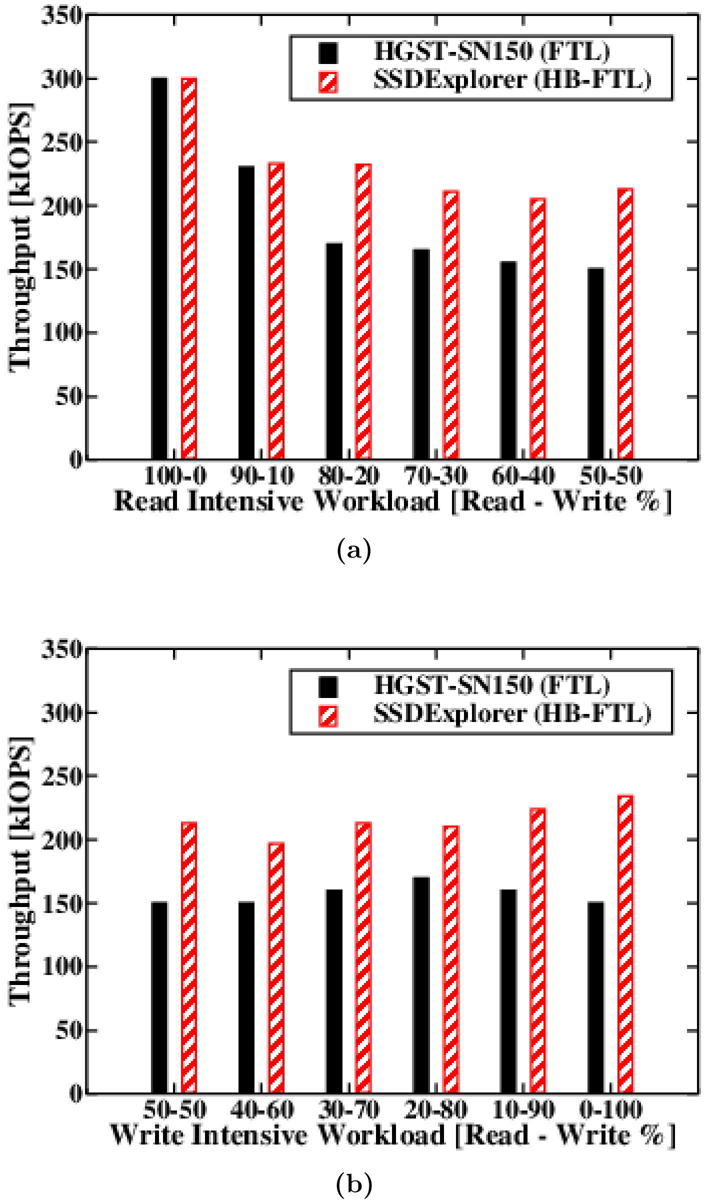


Figure 2.25: Throughput (expressed in kIOPS) of HGST SN150 Ultrastar SSD architecture compared to that of a simulated HB-FTL-based drive with the same configuration: (a) read intensive and (b) write intensive workloads. A queue depth of 32 commands is used. Simulations have been performed with SSDExplorer [26].

write percentage, whereas in a HB-FTL-based SSD performance is mostly independent from the write percentage. This result is due to the absence of the GC algorithm that strongly affects standard FTL-based SSDs.

Another architecture that can fully exploit the Open-Channel concept and the LNVME protocol relies on the usage of a dedicated accelerator in the form of a *Multi-Purpose Processing Array* (MPPA) [105, 106], as shown in Fig. 2.26. This solution allows the reduction of the host I/O command submission/completion timings.

These delays are strictly related to the host's processing capabilities and they represent the time spent by the host to execute the LNVME driver and the OS file system for each submitted/completed I/O. It has been demonstrated that the performance of nowadays SSDs is heavily affected by the I/O submission/completions timings [107]. Moreover, in most recent architectures like the one based on the 3D Xpoint technology [108], these delays can even represent the actual bottleneck of the whole storage layer, whose IOPS are limited by the host system itself. As a consequence, reducing these timings is the key for designing ultra-high performance storage systems.

A possible solution to this problem is to switch the LNVME protocol from an interrupt-driven I/O completion mechanism to a polling-driven approach. Basically, in standard SSDs, when an I/O is completed, the Flash controller sends an interrupt to the host notifying that the transaction is ready to be transferred/processed. After that, the host can submit another command to the drive because the submission of an I/O is driven by a completion event. In theory this approach requires that the host takes action only when I/Os are submitted/completed, but in practice it introduces long processing delays because of the OS interrupt service routines [107]. Polling the I/O completion events, on the contrary, can minimize the above mentioned processing timings. It requires, however, that the host system monitors continuously the I/Os, thus wasting part of its processing capabilities. In light of all these considerations, moving the whole submission/completion process to a dedicated MPPA represents a good solution which can offload the host system and, at the same time, exploit the full performance of the NAND Flash cards.

Fig. 2.27 shows the bandwidth comparison among the HGST SN150 Ultrastar SSD [104] and two simulated drives with the same architectural configuration, the former executing the FTL on the host (HB-FTL), the latter on a dedicated MPPA (HB-

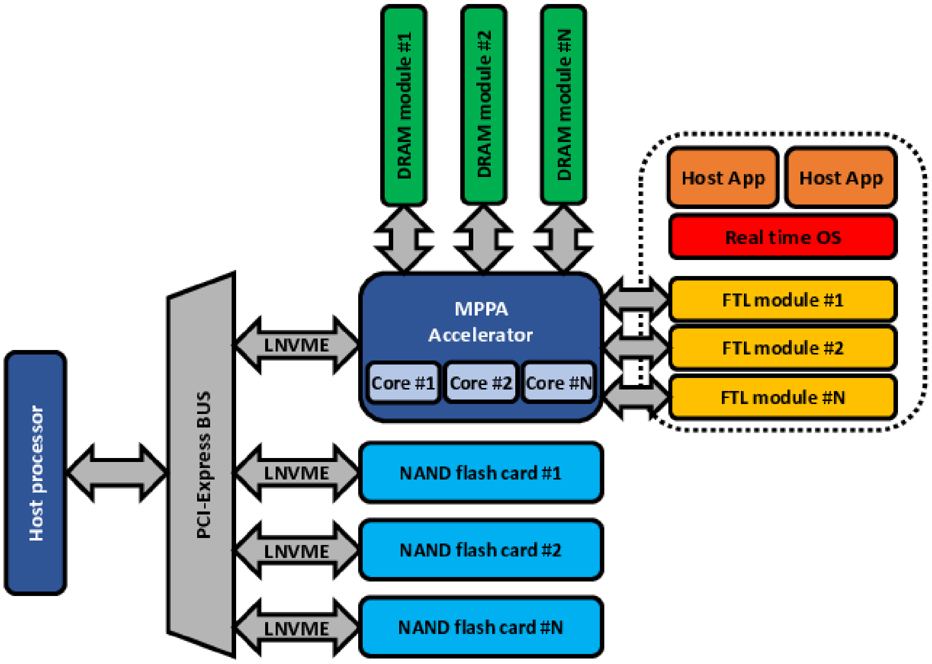


Figure 2.26: Reference architecture modeled by the Open-Channel storage layer when a MPPA is used for HB-FTL execution. Besides the NAND Flash cards, the PCI-Express bus is connected to a MPPA accelerator executing different FTL modules.

FTL-MPPA). Five different MPPA acceleration levels have been considered, ranging from a 0% speed-up of the host up to the 95%. The maximum I/O acceleration was imposed by the hardware limitations introduced by the PCI-Express bus.

As it can be seen the HB-FTL-MPPA is able to heavily improve performance in all the tested conditions, but it is extremely effective when write intensive workloads are considered. This phenomenon is related to the fact that program operations on NAND flash cards still follow a *Write-Through* (WT) [109] caching policy; therefore, once the data payload is transferred to the target card, a completion packet goes immediately back to the MPPA. At this point it is clear that, since the access time of WT buffers is in the order of a few μs, the reduction of the I/O submission/completion timings impacts the overall transfer time of the payload. This is also true for read operations, but because of the pipelining and queuing effects of the NAND flash cards, the overall improvement is not so evident.

These considerations push towards a new SSD design methodology: a complete virtualization of the storage backbone. In fact, both HB-FTL and Open-Channel

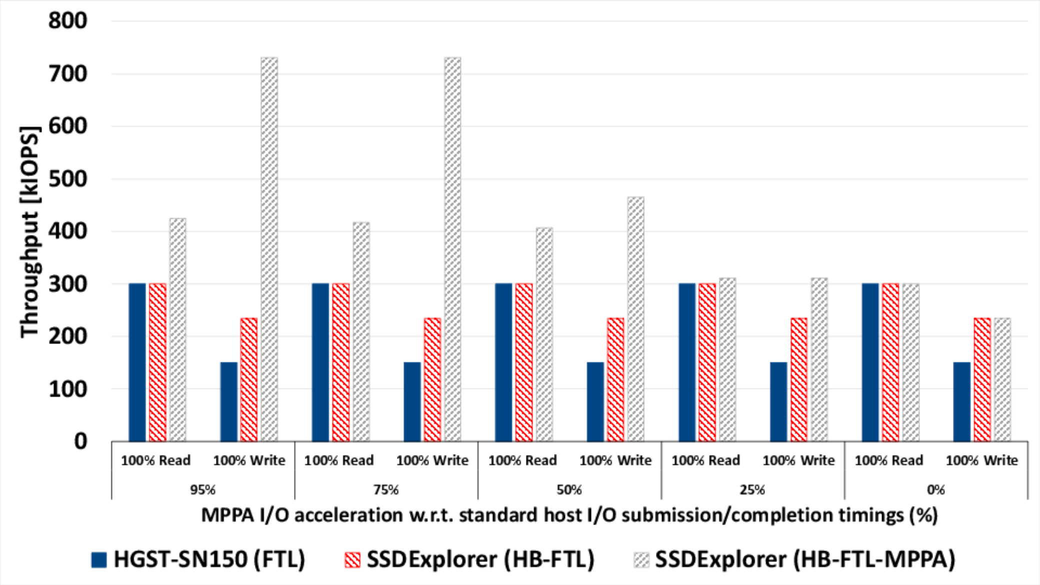


Figure 2.27: Throughput (expressed in kIOPS) of HGST SN150 Ultrastar SSD compared with the simulated HB-FTL or HB-FTL-MPPA for different percentages of host I/O submission/completion timings accelerations. Command queue depth is 32. Simulations have been performed with SSDExplorer [26].

allow to virtually separating the internal resources of the SSD (like channels and targets), providing a clear and straight path to OS data partitioning.

To sum up, an SSD design aimed at optimizing performances must follow a Bottom-Up approach; indeed, most of the design constraints are strongly related to the performance and reliability of the nonvolatile storage medium. A detailed knowledge of the memory behavior (i.e. endurance, data retention and read disturb) is mandatory to efficiently design the whole SSD architecture. RBER represents the main figure of merit driving designer's choices. The knowledge of RBER and, in particular, of its dependency from time and workload, allows selecting the most effective architecture to extend the memory's lifetime as much as possible. Since RBER increases with technology scaling, the use of LDPC codes represents the solution of choice for the most advanced ECC engines. Once the NAND Flash memories (together with the knowledge of their RBER) and the most appropriate ECC algorithm (together with either read retry techniques or LDPC soft decisions) have been selected, the design of the SSD controller must be based on multiple aspects:

- the ECC architecture, as a trade-off between performance (bandwidth, latency, power consumption) and area occupation;

- the number of memory channels, as a trade-off again between performance and area occupation;

- the number of memory dies per channel, that is generally a power of 2;

- the appropriate command management, maximizing the number of active dies and hence the SSD bandwidth, whereas limiting as much as possible the maximum latency (*i.e.* the QoS) by leveraging the head of line blocking concept;

- the introduction of a DRAM data cache buffer able to reduce the number of access operations to the NAND Flash memories, thus increasing SSD bandwidth while reducing NAND Flash degradation effects;

- the choice of the most suitable host interface able to guarantee the performance requested by the host applications.

To further improve the performance of next generation SSDs to be used in hyper-scaled environments it is possible to leverage new approaches, like SDF, exploiting hardware/software co-design of the SSD controller architecture and of the host applications.

Bibliography

[1] G. Wong, “SSD Market Overview,” in *Inside Solid State Drives (SSDs)*, R. Micheloni, A. Marelli, and K. Eshghi, Ed. Springer, 2012, pp. 1–17.

[2] Semiconductor Industry Association, “International Technology Roadmap for Semiconductors,” [Online]. Available: {http://www.semiconductors.org/main/2015_international_technology_roadmap_for_semiconductors_itrs/}, 2015.

[3] D. Liu, Y. Wang, Z. Qin, Z. Shao, and Y. Guan, “A Space Reuse Strategy for Flash Translation Layers in SLC NAND Flash Memory Storage Systems,” *IEEE Trans. on VLSI systems*, vol. 20, no. 6, pp. 1094–1107, June 2012.

[4] T. Wang, D. Liu, Y. Wang, and Z. Shao, “FTL2: A Hybrid Flash Translation Layer with Logging for Write Reduction in Flash Memory,” *ACM SIGPLAN Not.*, vol. 48, no. 5, pp. 91–100, 2013.

[5] Y. H. Chang, P. C. Huang, P. H. Hsu, L. J. Lee, T. W. Kuo, and D. Du, “Reliability Enhancement of Flash-Memory Storage Systems: An Efficient Version-Based Design,” *IEEE Trans. on Computers*, vol. 62, no. 12, pp. 2503–2515, 2013.

[6] JEDEC Org., “JESD 22- A 117 document,” Oct. 2011.

[7] R. Micheloni, A. Marelli and R. Ravasio, “Basic Coding Theory,” in *Error Correction Codes for Non-Volatile Memories*, R. Micheloni, A. Marelli, and R. Ravasio, Ed. Springer, 2008, pp. 1–33.

[8] Serial ATA International Organization, “SATA Revision 3.0 Specifications,” [Online]. Available: www.sata-io.org.

[9] Intel Inc., “Intel Solid-State Drive DC S3500 Series Quality of Service,” [Online]. Available: {http://www.intel.com/content/www/us/en/solid-state-drives/ssd-dc-s3500-spec.html}, p. 9, 2013.

[10] A. Grossi, L. Zuolo, F. Restuccia, C. Zambelli, and P. Olivo, “Quality-of-Service Implications of Enhanced Program Algorithms for Charge-Trapping NAND in Future Solid-State Drives,” *IEEE Trans. on Devices and Materials Reliability*, vol. 15, no. 3, pp. 363–369, 2015.

[11] S. Aritome, *NAND Flash Memory Technologies.* Wiley-IEEE Press, 2016.

[12] C. Zambelli and P. Olivo, “SSD Reliability,” in *Inside Solid State Drives (SSDs)*, R. Micheloni, A. Marelli, and K. Eshghi, Ed. Springer, 2013, pp. 203–231.

[13] J. D. Lee, J. H. Choi, D. Park, and K. Kim, “Degradation of Tunnel Oxide by FN Current Stress and its Effects on Data Retention Characteristics of 90 nm NAND Flash Memory Cells,” in *Proc. Int. Rel. Phys. Symp.*, Mar. 2003, pp. 497–501.

[14] N. Mielke, H. Belgal, I. Kalastirsky, P. Kalavade, A. Kurtz, Q. Meng, N. Righos, and J. Wu, “Flash EEPROM Threshold Instabilities due to Charge Trapping during Program/Erase Cycling,” *IEEE Trans. on Devices and Materials Reliability*, vol. 4, no. 3, pp. 335–344, 2004.

[15] N. Mielke, T. Marquart, N. Wu, J. Kessenich, H. Belgal, E. Schares, F. Trivedi, E. Goodness, and L. R. Nevill, “Bit Error Rate in NAND Flash Memories,” in *Proc. Int. Rel. Phys. Symp.*, Apr. 2008, pp. 9–19.

[16] M. Lenzlinger and E. H. Snow, "FowlerNordheim Tunneling into Thermally Grown SiO_2," *J. Appl. Phys.*, vol. 40, pp. 278 – 283, 1969.

[17] K. D. Suh, B. H. Suh, Y. H. Um, J. K. Kim, Y. J. Choi, Y. N. Koh, S. S. Lee, S. C. Kwon, B. S. Choi, J. S. Yum, J. H. Choi, J. R. Kim, and H. K. Lim, "A 3.3 V 32 Mb NAND Flash Memory with Incremental Step Pulse Programming Scheme," in *IEEE Int. Solid-State Circuits Conf.*, Feb. 1995, pp. 128–129.

[18] K. Fukuda, Y. Watanabe, E. Makino, K. Kawakami, J. Sato, T. Takagiwa, N. Kanagawa, H. Shiga, N. Tokiwa, Y. Shindo, T. Ogawa, T. Edahiro, M. Iwai, O. Nagao, J. Musha, T. Minamoto, Y. Furuta, K. Yanagidaira, Y. Suzuki, D. Nakamura, Y. Hosomura, R. Tanaka, H. Komai, M. Muramoto, G. Shikata, A. Yuminaka, K. Sakurai, M. Sakai, H. Ding, M. Watanabe, Y. Kato, T. Miwa, A. Mak, M. Nakamichi, G. Hemink, D. Lee, M. Higashitani, B. Murphy, B. Lei, Y. Matsunaga, K. Naruke, and T. Hara, "A 151-mm^2 64-Gb 2 Bit/Cell NAND Flash Memory in 24-nm CMOS Technology," *IEEE J. of Solid State Circuit*, vol. 47, no. 1, pp. 75–84, 2012.

[19] K. T. Park, O. Kwon, S. Yoon, M. H. Choi, I. M. Kim, B. G. Kim, M. S. Kim, Y. H. Choi, S. H. Shin, Y. Song, J. Y. Park, J. E. Lee, C. G. Eun, H. C. Lee, H. J. Kim, J. H. Lee, J. Y. Kim, T. M. Kweon, H. J. Yoon, T. Kim, D. K. Shim, J. Sel, J. Y. Shin, P. Kwak, J. M. Han, K. S. Kim, S. Lee, Y. H. Lim, and T. S. Jung, "A 7MB/s 64Gb 3-Bit/Cell DDR NAND Flash Memory in 20nm-Node Technology," in *IEEE Int. Solid-State Circuits Conf.*, Feb. 2011, pp. 212–213.

[20] C. Trinh, N. Shibata, T. Nakano, M. Ogawa, J. Sato, Y. Takeyama, K. Isobe, B. Le, F. Moogat, N. Mokhlesi, K. Kozakai, P. Hong, T. Kamei, K. Iwasa, J. Nakai, T. Shimizu, M. Honma, S. Sakai, T. Kawaai, S. Hoshi, J. Yuh, C. Hsu, T. Tseng, J. Li, J. Hu, M. Liu, S. Khalid, J. Chen, M. Watanabe, H. Lin, J. Yang, K. McKay, K. Nguyen, T. Pham, Y. Matsuda, K. Nakamura, K. Kanebako, S. Yoshikawa, W. Igarashi, A. Inoue, T. Takahashi, Y. Komatsu, C. Suzuki, K. Kanazawa, M. Higashitani, S. Lee, T. Murai, K. Nguyen, J. Lan, S. Huynh, M. Murin, M. Shlick, M. Lasser, R. Cernea, M. Mofidi, K. Schuegraf, and K. Quader, "A 5.6MB/s 64Gb 4b/Cell NAND Flash Memory in 43nm CMOS," in *IEEE Int. Solid-State Circuits Conf.*, Feb. 2009, pp. 246–247.

[21] L. Zuolo, C. Zambelli, R. Micheloni, D. Bertozzi, and P. Olivo, "Analysis of Reliability/Performance Trade-off in Solid State Drives," in *Proc. Int. Rel. Phys. Symp.*, June 2014, pp. 4B.3.1–4B.3.5.

[22] K. Zhao, W. Zhao, H. Sun, X. Zhang, N. Zheng, and T. Zhang, "LDPC-in-SSD: Making Advanced Error Correction Codes Work Effectively in Solid State Drives," in *USENIX Conf. on File and Storage Technologies*, 2013, pp. 243–256.

[23] L. Zuolo, C. Zambelli, R. Micheloni, S. Galfano, M. Indaco, S. D. Carlo, P. Prinetto, P. Olivo, and D. Bertozzi, "SSDExplorer: a Virtual Platform for Fine-Grained Design Space Exploration of Solid State Drives," in *Proc. of IEEE Eur. Design Automation and Test Conf.*, Mar. 2014, pp. 1–6.

[24] Seagate Technology LLC, "Serial Attached SCSI (SAS)," [Online]. Available: http://www.seagate.com/staticfiles/support/disc/manuals/Interface%20manuals/100293071c.pdf, 2009.

[25] PCI-SIG Ass., "PCI Express Base 3.0 Specification," [Online]. Available: {http://www.pcisig.com/specifications/pciexpress/base3/}, 2013.

[26] L. Zuolo, C. Zambelli, R. Micheloni, M. Indaco, S. Di Carlo, P. Prinetto, D. Bertozzi, and P. Olivo, "SSDExplorer: A Virtual Platform for Performance/Reliability-Oriented Fine-Grained Design Space Exploration of Solid State Drives," *IEEE Trans. Computer-Aided Design*, vol. 34, no. 10, pp. 1627–1638, 2015.

[27] C. Zambelli, M. Indaco, M. Fabiano, S. D. Carlo, P. Prinetto, P. Olivo, and D. Bertozzi, "A Cross-Layer Approach for new Reliability-Performance Trade-offs in MLC NAND Flash Memories," in *Proc. of IEEE Eur. Design Automation and Test Conf.*, Mar. 2012, pp. 881–886.

[28] INTEL inc., "Intel X18-M X25-M SATA Solid State Drive. Enterprise Server/Storage Applications," [Online]. Available: {http://cache-www.intel.com/cd/00/00/42/52/425265_425265.pdf}.

[29] J. Ouyang, S. Lin, S. Jiang, Z. Hou, Y. Wang, and Y. Wang, "SDF: Software-defined Flash for Web-scale Internet Storage Systems," in *Proc. ACM Int. Conf. on Architectural Support for Programming Languages and Operating Systems*, Mar. 2014, pp. 471–484.

[30] F. Masuoka, M. Momodomi, Y. Iwata, and R. Shirota, "New Ultra High Density EPROM and Flash EEPROM with NAND Structure Cell," in *IEDM Tech. Dig.*, Dec. 1987, pp. 552–555.

[31] D. Kahng and S. Sze, "A Floating Gate and its Application to Memory Devices," *Bell Syst. Tech. J.*, vol. 46, p. 1288, 1967.

[32] C. Zambelli, A. Chimenton, and P. Olivo, "Reliability Issues of NAND Flash Memories," in *Inside NAND Flash Memories*, R. Micheloni, L. Crippa, and A. Marelli, Ed. Springer, 2010, pp. 89–113.

[33] A. Chimenton, C. Zambelli, and P. Olivo, "A Statistical Model of Erratic Behaviors in Flash Memory Arrays," *IEEE Trans. Electron Devices*, vol. 58, no. 11, pp. 3707–3711, 2011.

[34] A. Torsi, Y. Zhao, H. Liu, T. Tanzawa, A. Goda, P. Kalavade, and K. Parat, "A Program Disturb Model and Channel Leakage Current Study for sub-20 nm NAND Flash Cells," *IEEE Trans. Electron Devices*, vol. 58, no. 1, pp. 11–16, 2011.

[35] M. Momodomi, T. Tanaka, Y. Iwata, Y. Tanaka, H. Oodaira, Y. Itoh, R. Shirota, K. Ohuchi, and F. Masuoka, "A 4 Mb NAND EEPROM with Tight Programmed Vt Distribution," *IEEE J. of Solid State Circuit*, vol. 26, no. 4, pp. 492–496, 1991.

[36] G. J. Hemink, T. Tanaka, T. Endoh, S. Aritome, and R. Shirota, "Fast and Accurate Programming Method for Multi-Level NAND EEPROMs," in *VLSI Symp. on Tech.*, Jun. 1995, pp. 129–130.

[37] C. Monzio Compagnoni, A. S. Spinelli, R. Gusmeroli, S. Beltrami, A. Ghetti, and A. Visconti, "Ultimate Accuracy for the NAND Flash Program Algorithm Due to the Electron Injection Statistics," *IEEE Trans. Electron Devices*, vol. 55, no. 10, pp. 2695–2702, 2008.

[38] C. Monzio Compagnoni, R. Gusmeroli, A. S. Spinelli, and A. Visconti, "Analytical Model for the Electron-Injection Statistics During Programming of Nanoscale NAND Flash Memories," *IEEE Trans. Electron Devices*, vol. 55, no. 11, pp. 3192–3199, 2008.

[39] M. Momodomi, Y. Itoh, R. Shirota, Y. Iwata, R. Nakayama, R. Kirisawa, T. Tanaka, S. Aritome, T. Endoh, K. Ohuchi, and F. Masuoka, "An Experimental 4-Mbit CMOS EEPROM with a NAND-Structured Cell," *IEEE J. of Solid State Circuit*, vol. 24, no. 5, pp. 1238–1243, 1989.

[40] A. Chimenton, P. Pellati, and P. Olivo, "Analysis of Erratic Bits in Flash Memories," *IEEE Trans. on Devices and Materials Reliability*, vol. 1, no. 4, pp. 179–184, 2001.

[41] S. Aritome, R. Kirisawa, T. Endoh, R. Nakayama, R. Shirota, K. Sakui, K. Ohuchi, and F. Masuoka, "Extended Data Retention Characteristics after more than 10^4 Write and Erase Cycles in EEPROMs," in *Proc. Int. Rel. Phys. Symp.*, Mar. 1990, pp. 259–264.

[42] R. Shirota, R. Nakayama, R. Kirisawa, M. Momodomi, K. Sakui, Y. Itoh, S. Aritome, T. Endoh, F. Hatori, and F. Masuoka, "A 2.3 μm^2 Memory Cell Structure for 16 Mb NAND EEPROMs," in *IEDM Tech. Dig.*, Dec. 1990, pp. 103–106.

[43] E. I. Vatajelu, H. Aziza, and C. Zambelli, "Nonvolatile Memories: Present and Future Challenges," in *Int. Design and Test Symposium*, Dec. 2014, pp. 61–66.

[44] Y. Park and D. K. Schroder, "Degradation of Thin Tunnel Gate Oxide under Constant Fowler-Nordheim Current Stress for a Flash EEPROM," *IEEE Trans. Electron Devices*, vol. 45, no. 6, pp. 1361–1368, 1998.

[45] T. N. Nguyen, P. Olivo, and B. Riccò, "A New Failure Mode of Very Thin ($< 50\mathring{A}$) Thermal SiO_2 Films," in *Proc. Int. Rel. Phys. Symp.*, Apr. 1987, pp. 66 – 71.

[46] P. Olivo, T. N. Nguyen, and B. Riccò, "High-Field-Induced Degradation in Ultra Thin SiO_2 Films," *IEEE Trans. Electron Devices*, vol. 351, no. 12, pp. 2259–2267, 1988.

[47] G. J. Hemink, K. Shimizu, S. Aritome, and R. Shirota, "Trapped Hole Enhanced Stress Induced Leakage Currents in NAND EEPROM Tunnel Oxides," in *Proc. Int. Rel. Phys. Symp.*, Apr. 1996, pp. 117–121.

[48] R. Yamada, Y. Mori, Y. Okuyama, J. Yugami, T. Nishimoto, and H. Kume, "Analysis of Detrap Current due to Oxide Traps to Improve Flash Memory Retention," in *Proc. Int. Rel. Phys. Symp.*, Apr. 2000, pp. 200–204.

[49] J. Lee, J. Choi, D. Park, and K. Kim, "Effects of Interface Trap Generation and Annihilation on the Data Retention Characteristics of Flash Memory Cells," *IEEE Trans. on Devices and Materials Reliability*, vol. 4, no. 1, pp. 110–117, 2004.

[50] K. Lee, M. Kang, S. Seo, D. Kang, D. H. Li, Y. Hwang, and H. Shin, "Separation of Corner Component in TAT Mechanism in Retention Characteristics of Sub 20-nm NAND Flash Memory," *IEEE Electron Device Lett.*, vol. 35, no. 1, pp. 51–53, 2014.

[51] G. Molas, D. Deleruyelle, B. De Salvo, G. Ghibaudo, M. Gely, S. Jacob, D. Lafond, and S. Deleonibus, "Impact of Few Electron Phenomena on Floating-Gate Memory Reliability," in *IEDM Tech. Dig.*, Dec. 2004, pp. 877–880.

[52] R. Gusmeroli, C. Monzio Compagnoni, A. Riva, A. S. Spinelli, A. L. Lacaita, M. Bonanomi, and A. Visconti, "Defects Spectroscopy in SiO_2 by Statistical Random Telegraph Noise Analysis," in *IEDM Tech. Dig.*, Dec. 2006, pp. 1–4.

[53] K. Fukuda, Y. Shimizu, K. Amemiya, M. Kamoshida, and C. Hu, "Random Telegraph Noise in Flash Memories - Model and Technology Scaling," in *IEDM Tech. Dig.*, Dec. 2007, pp. 169–172.

[54] C. Monzio Compagnoni, A. Spinelli, S. Beltrami, M. Bonanomi, and A. Visconti, "Cycling Effect on the Random Telegraph Noise Instabilities of NOR and NAND Flash Arrays," *IEEE Electron Device Lett.*, vol. 29, pp. 941–943, 2008.

[55] A. Chimenton, C. Zambelli, and P. Olivo, "A New Methodology for Two-Level Random-Telegraph-Noise Identification and Statistical Analysis," *IEEE Electron Device Lett.*, vol. 31, no. 6, pp. 612–614, 2010.

[56] ——, "A Statistical Model of Erratic Erase Based on an Automated Random Telegraph Signal Characterization Technique," in *Proc. Int. Rel. Phys. Symp.*, Apr. 2009, pp. 896–901.

[57] C. Zambelli, G. Koebernik, R. Ullmann, M. Bauer, G. Tempel, F. D. Tano, M. Atti, F. P. Pistone, A. Siviero, and P. Olivo, "Exposing Reliability/Performance Tradeoff in Non-Volatile Memories Through Erratic Bits Signature Classification," *IEEE Trans. on Devices and Materials Reliability*, vol. 14, no. 1, pp. 66–73, 2014.

[58] C. Zambelli, T. Vincenzi, and P. Olivo, "A Compact Model for Erratic Event Simulation in Flash Memory Arrays," *IEEE Trans. Electron Devices*, vol. 61, no. 11, pp. 3716–3722, 2014.

[59] C. Monzio Compagnoni, A. S. Spinelli, R. Gusmeroli, A. L. Lacaita, S. Beltrami, A. Ghetti, and A. Visconti, "First Evidence for Injection Statistics Accuracy Limitations in NAND Flash Constant-Current Fowler-Nordheim Programming," in *IEDM Tech. Dig.*, Dec. 2007, pp. 165–168.

[60] C. Friederich, J. Hayek, A. Kux, T. Muller, N. Chan, G. Kobernik, M. Specht, D. Richter, and D. Schmitt-Landsiedel, "Novel Model for Cell - System Interaction (MCSI) in NAND Flash," in *IEDM Tech. Dig.*, Dec. 2008, pp. 1–4.

[61] S. Satoh, G. Hemink, K. Hatakeyama, and S. Aritome, "Stress-Induced Leakage Current of Tunnel Oxide derived from Flash Memory Read-Disturb Characteristics," *IEEE Trans. Electron Devices*, vol. 45, no. 2, pp. 482–486, 1998.

[62] H. H. Wang, P. S. Shieh, C. T. Huang, K. Tokami, R. Kuo, S. H. Chen, H. C. Wei, S. Pittikoun, and S. Aritome, "A New Read-Disturb Failure Mechanism Caused by Boosting Hot-Carrier Injection Effect in MLC NAND Flash Memory," in *IEEE Int. Memory Workshop*, May 2009, pp. 1–2.

[63] J. Lee, S. Hur, and J. Choi, "Effects of Floating-Gate Interference on NAND Flash Memory Cell Operation," *IEEE Electron Device Lett.*, vol. 23, no. 5, pp. 264–266, 2002.

[64] K. T. Park, M. Kang, D. Kim, S. W. Hwang, B. Y. Choi, Y. T. Lee, C. Kim, and K. Kim, "A Zeroing Cell-to-Cell Interference Page Architecture With Temporary LSB Storing and Parallel MSB Program Scheme for MLC NAND Flash Memories," *IEEE J. of Solid State Circuit*, vol. 43, no. 4, pp. 919–928, 2008.

[65] S. Seo, H. Kim, S. Park, S. Lee, S. Aritome, and S. Hong, "Novel Negative Vt Shift Program Disturb Phenomena in 2X-3X nm NAND Flash Memory Cells," in *Proc. Int. Rel. Phys. Symp.*, 2011, pp. 6B.2.1–6B.2.4.

[66] C. Zambelli, F. Andrian, S. Aritome, and P. Olivo, "Compact Modeling of Negative Shift Disturb in NAND Flash Memories," *IEEE Trans. Electron Devices*, vol. 63, no. 4, pp. 1516–1523, 2016.

[67] I. Park, W.-G. Hahn, K.-W. Song, K. Choi, H.-K. Choi, S. Lee, C.-S. Lee, J. Song, J. Han, K. Kyoung, and Y.-H. Jun, "A New GIDL Phenomenon by Feld Effect of Neighboring Cell Transistors and its Control Solutions in sub-30 nm NAND Flash Devices," in *VLSI Symp. on Tech.*, June 2012, pp. 23–24.

[68] J. Lee, C. Lee, M. Lee, H. Kim, K. Park, and W. Lee, "A New Programming Disturbance Phenomenon in NAND Flash Memory By Source/Drain Hot-Electrons Generated By GIDL Current," in *Non-Volatile Semiconductor Memory Workshop*, Feb. 2006, pp. 31–33.

[69] S. Satoh, H. Hagiwara, T. Tanzawa, K. Takeuchi, and R. Shirota, "A Novel Isolation-Scaling Technology for NAND EEPROMs with the Minimized Program Disturbance," in *IEDM Tech. Dig.*, Dec. 1997, pp. 291–294.

[70] R. Micheloni, A. Marelli and R. Ravasio, "Cyclic Codes for Non Volatile Storage," in *Error Correction Codes for Non-Volatile Memories*, R. Micheloni, A. Marelli, and R. Ravasio, Ed. Springer, 2008, pp. 167–198.

[71] Y. Lee, H. Yoo, I. Yoo, and I.-C. Park, "6.4Gb/s Multi-Threaded BCH Encoder and Decoder for Multi-Channel SSD Controllers," in *IEEE Int. Solid-State Circuits Conf.*, Feb 2012, pp. 426–428.

[72] R. Micheloni, A. Marelli and R. Ravasio, "BCH Hardware Implementation in NAND Flash Memories," in *Error Correction Codes for Non-Volatile Memories*, R. Micheloni, A. Marelli, and R. Ravasio, Ed. Springer, 2008, pp. 199–247.

[73] S. M. Jeff Yang, "High-Efficiency SSD for Reliable Data Storage Systems," in *Flash Memory Summit*, 2012.

[74] A. Cometti, L. Huang, and A. Melik-Martirosian, "Apparatus and Method for Determining a Read Level of a Flash Memory after an Inactive Period of Time," Feb. 4 2014, US Patent 8,644,099.

[75] X. Wang, G. Dong, L. Pan, and R. Zhou, "Error Correction Codes and Signal Processing in Flash Memory," in *Flash Memories*, I. Stievano, Ed., 2011, pp. 57–82.

[76] L. Zuolo, C. Zambelli, A. Marelli, R. Micheloni, and P. Olivo, "LDPC Soft Decoding with Improved Performance in 1X-2X MLC and TLC NAND Flash-Based Solid State Drives," *IEEE Trans. on Emerging Topics in Computing*, in press, 2017.

[77] S. N. R. Motwani, Z. Kwok, "Low Density Parity Check (LDPC) Codes and the Need for Stronger ECC ," in *Flash Memory Summit*, Aug. 2011.

[78] M. Ivkovic, S. K. Chilappagari, and B. Vasic, "Eliminating trapping sets in low-density parity-check codes by using Tanner graph covers," *IEEE Trans. on Information Theory*, vol. 54, no. 8, pp. 3763–3768, 2008.

[79] A. Marelli, "False Decoding Probability (Detection) of BCH and LDPC Codes," in *Flash Memory Summit*, 2016.

[80] Y. Yamaga, C. Matsui, S. Hachiya, and K. Takeuchi, "Application Optimized Adaptive ECC with Advanced LDPCs to Resolve Trade-Off among Reliability, Performance, and Cost of Solid-State Drives," in *IEEE Int. Memory Workshop*, May 2016, pp. 1–4.

[81] L. Zuolo, C. Zambelli, P. Olivo, R. Micheloni, and A. Marelli, "LDPC Soft Decoding with Reduced Power and Latency in 1X-2X NAND Flash-Based Solid State Drives," in *IEEE Int. Memory Workshop*, May 2015, pp. 1–4.

[82] W. Lin, "Advanced Controller Technology for 3D NAND Flash," in *Flash Memory Summit*, Aug. 2016.

[83] R. Micheloni, A. Marelli, L. Crippa, A. Aldarese, L. Zuolo, C. Zambelli, and P. Olivo, "Fully Integrated SSD-NAND Characterization Flow," in *Flash Memory Summit*, 2015.

[84] X. Hu, E. Eleftheriou, R. Haas, I. Iliadis, and R. Pletka, "Write Amplification Analysis in Flash-based Solid State Drives," in *Proc. ACM Int. Systems and Storage Conf.*, May 2009, pp. 10:1–10:9.

[85] D. Rollins, "Best Practices for SSD Performance Measurement," in *Micron Technology, Inc., Technical Marketing Brief*, 2011. [Online]. Available: https://www.micron.com/~/media/documents/products/technical-marketing-brief/brief_ssd_performance_measure.pdf

[86] K. Eshghi and R. Micheloni, "SSD Architecture and PCI Express Interface," in *Inside Solid State Drives (SSDs)*, R. Micheloni, A. Marelli, and K. Eshghi, Ed. Springer, 2012, pp. 19–45.

[87] L. M. Grupp, J. D. Davis, and S. Swanson, "The Bleak Future of NAND Flash Memory," in *Proc. Usenix Int. Conference on File and Storage Technologies*, 2012, pp. 1–8.

[88] Avago Tech., "Accelerating Financial Applications Using Solid State Storage," [Online]. Available: http://docs.avagotech.com/docs/12353095, 2011.

[89] Amazon Inc., "New SSD-Backed Elastic Block Storage," [Online]. Available: https://aws.amazon.com/it/blogs/aws/new-ssd-backed-elastic-block-storage/, 2014.

[90] M. Karol, M. Hluchyj, and S. Morgan, "Input Versus Output Queueing on a Space-Division Packet Switch," *IEEE Trans. on Communications*, vol. 35, no. 12, pp. 1347–1356, 1987.

[91] E. G. Coffman Jr. and P. J. Denning, *Operating Systems Theory.* Prentice Hall Professional Technical Reference, 1973.

[92] S. Lee, T. Kim, K. Kim, and J. Kim, "Lifetime Management of Flash-based SSDs Using Recovery-aware Dynamic Throttling," in *Proc. Usenix Int. Conference on File and Storage Technologies*, 2012.

[93] R.-S. Liu, C.-L. Yang, and W. Wu, "Optimizing NAND Flash-Based SSDs via Retention Relaxation," in *Proc. Usenix Int. Conference on File and Storage Technologies*, 2012.

[94] "Open-Channel Solid State Drives," [Online]. Available: http://openchannelssd.readthedocs.org/en/latest/, 2016.

[95] A. Batwara, "Leveraging Host based Flash Translation Layer for Application Acceleration," in *Flash Memory Summit*, Aug. 2012.

[96] "Open Channel Solid State Drives NVMe Specification," [Online]. Available: http://bit.ly/2gfidpQ, 2016.

[97] Samsung Electronics Co., "Power loss Protection (PLP) - Protect your Data Against Sudden Power Loss," [Online]. Available: http://www.samsung.com/semiconductor/minisite/ssd/downloads/document/Samsung_SSD_845DC_05_Power_loss_protection_PLP.pdf, 2014.

[98] D. Apalkov, B. Dieny, and J. M. Slaughter, "Magnetoresistive Random Access Memory," *Proc. IEEE*, vol. 104, no. 10, pp. 1796–1830, 2016.

[99] Xilinx Inc., "Everspins NVMe Storage Accelerator mixes MRAM, UltraScale FPGA, delivers 1.5M IOPS," [Online]. Available: https://forums.xilinx.com/t5/Xcell-Daily-Blog/Everspin-s-NVMe-Storage-Accelerator-mixes-MRAM-UltraScale-FPGA/ba-p/733781, 2016.

[100] G. Sun, Y. Joo, Y. Chen, D. Niu, Y. Xie, Y. Chen, and H. Li, "A Hybrid Solid-State Storage Architecture for the Performance, Energy Consumption, and Lifetime Improvement," in *IEEE Int. Symposium on High-Performance Computer Architecture*, 2010, pp. 1–12.

[101] J. Gonzalez, M. Bjrling, S. Lee, C. Dong, and Y. R. Huang, "Application-Driven Flash Translation Layers on Open-Channel SSDs," in *Non Volatile Memory Workshop*, Mar. 2016, pp. 1–2.

[102] C. Friederich, "Program and Erase of NAND Memory Arrays," in *Inside NAND Flash Memories*, R. Micheloni, L. Crippa, and A. Marelli, Ed. Springer, 2010, pp. 55–88.

[103] S. Bates, "Accelerating Data Centers Using NVMe and CUDA," in *Flash Memory Summit*, Aug. 2014.

[104] HGST Inc., "Ultrastar SN150 Series NVMe PCIe x4 Lane Half-Height Half-Length CardSolid-State Drive Product Manual," [Online]. Available: https://www.hgst.com/sites/default/files/resources/US_SN150_ProdManual.pdf.

[105] Karlay Inc., "The KalRay Multi-Purpose-Processing-Array (MPPA)," [Online]. Available: http://www.kalrayinc.com/kalray/products/#processors, 2016.

[106] P. Couvert, "High Speed IO Processor for NVMe over Fabric (NVMeoF)," in *Flash Memory Summit*, Aug. 2016.

[107] J. Yang, D. B. Minturn, and F. Hady, "When Polling is Better than Interrupt," in *USENIX Conf. on File and Storage Technologies*, Feb. 2012.

[108] F. Hady, "Wicked Fast Storage and Beyond," in *Non Volatile Memory Workshop*, Mar. 2016.

[109] J. L. Hennessy, D. A. Patterson, "B.2 Cache Performance," in *Computer Architecture: A Quantitative Approach*, Morgan Kaufmann, Ed. Elsevier, 2011.

Chapter 3

3D NAND Flash memories

Nowadays, Solid State Drives consume an enormous amount of NAND Flash memories [1] causing a restless pressure on increasing the number of stored bits per mm^2. Planar memory cells have been scaled for decades by improving process technology, circuit design, programming algorithms [2], and lithography.

Unfortunately, when approaching a minimum feature size of 1x-nm, more challenges pop up: doping concentration in the channel region becomes difficult to control [3], RTN [4] and electron injection statistics [5] widen threshold distributions, thus causing a significant hit to both endurance and retention. Furthermore, by reducing the distance between memory cells, the intra-wordline electric field becomes higher, pushing the bit error rate to an even higher level.

3D arrays can definitely be considered as a breakthrough for fueling a further increase of the bit density. Identifying the right way for going 3D is not so easy though.

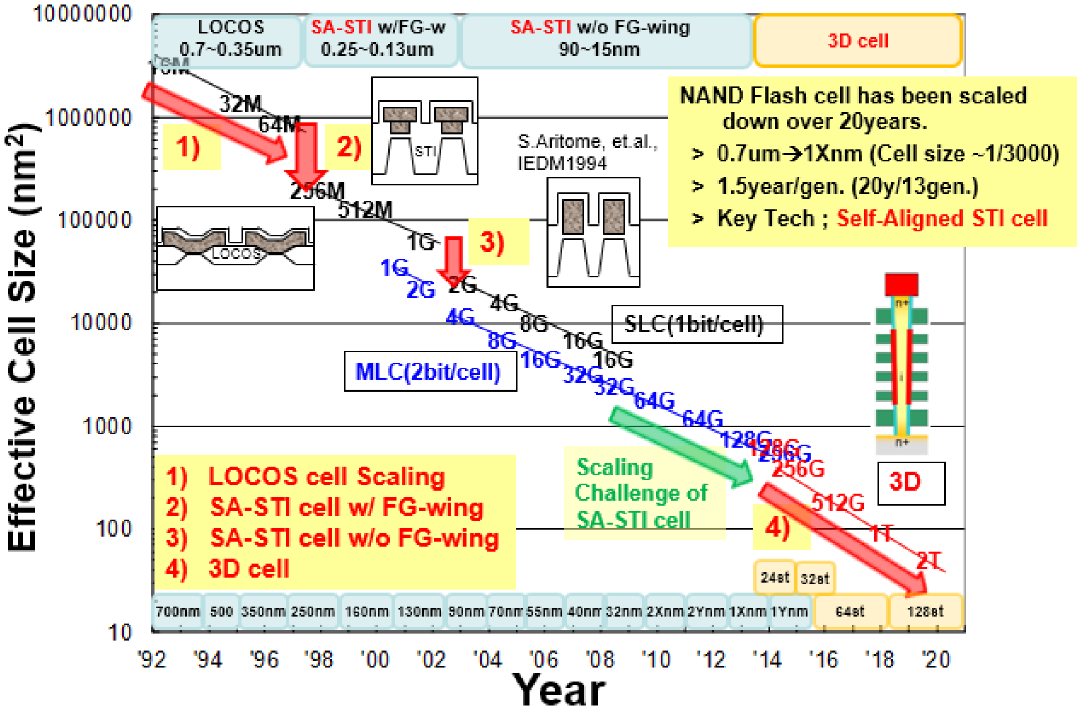
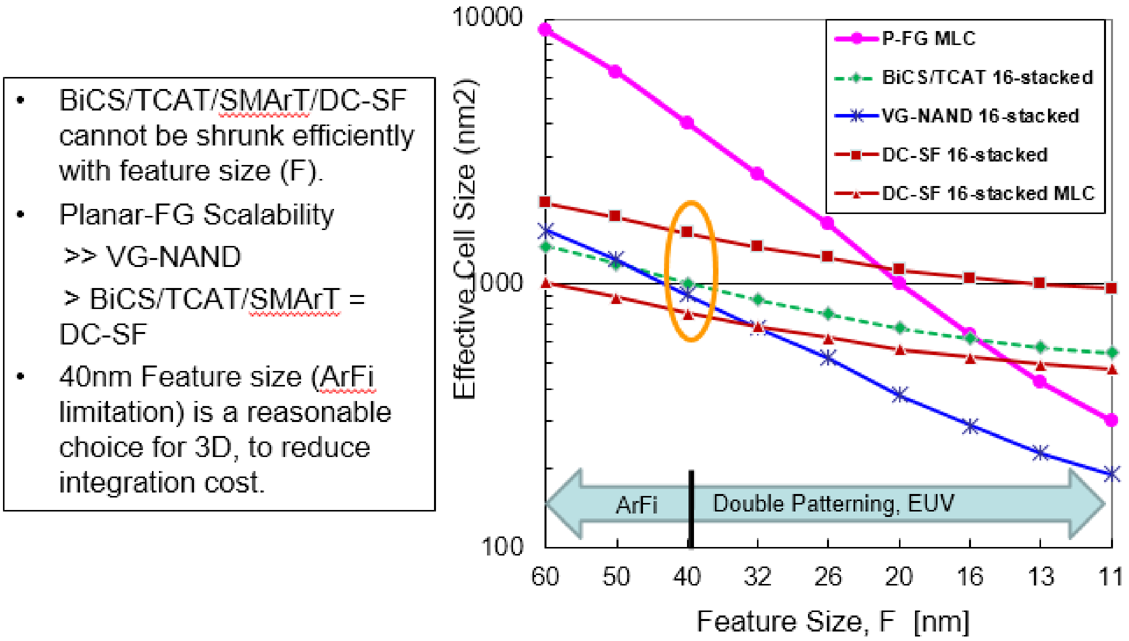


Figure 3.1 NAND Flash Memory Technology Roadmap [6]



- BiCS/TCAT/SMArT/DC-SF cannot be shrunk efficiently with feature size (F).
- Planar-FG Scalability
  >> VG-NAND
  > BiCS/TCAT/SMArT = DC-SF
- 40nm Feature size (ArFi limitation) is a reasonable choice for 3D, to reduce integration cost.

Figure 3.2 3D NAND Flash scaling [7]

Historically, Flash memory manufacturers have leveraged lithography to shrink the 2-dimensional (2D) memory cell (Figure 3.1) [6].

However, with 3D architectures, the "simple" reduction of the minimum feature size is running out of steam, as shown in Figure 3.2 [7]: a higher number of stacked cells is the only hope for dramatically reducing the real estate of a stored bit.

3D arrays can leverage either *Floating Gate* (FG) or *Charge Trapping* (CT) technologies [8]. As a matter of fact, the vast majority of 3D architectures published to date are built with CT cells, mainly because of the simpler fabrication process. Nevertheless, Floating Gate is still around and there are commercial products who managed to integrate FG into a 3D array.

This chapter is organized as follows. In Section 3.1 the simplest (from a conceptual point of view) 3D architecture will be described: basically, a stack of planar memories. In Section 3.2 we will introduce the most popular 3D arrays based on CT, from BiCS to P-BiCS, from TCAT to V-NAND. Section 3.3 is devoted to 3D memories based on Floating Gate cells: we will provide an overview of the FG cells that have developed to date, together with some details of the most recent commercial products. Section 3.4 introduces the most advanced layout solutions for 3D NAND memories with vertical channel; in particular, we will analyze staggering techniques for vertical pillars and bitline contacts. Integration of an increasing number of cells along the vertical direction causes a lot of technological problems which are very tough to solve. Section 3.5 highlights the main challenges in this space, including data retention and program disturb. A specific sub-section describes memory devices with circuits underneath the array: this kind of solutions are gaining traction as they can save a significant portion of the chip area (i.e. cost). Finally, in Section 3.6 we'll summarize scaling trends for the effective cell's size.

3.1 3D Stack of Planar NAND Flash Memories

The easiest 3D architecture that one can think of is a pile of planar arrays, one on top of each other, as shown in Figure 3.3. All NAND strings share drain contacts and bitlines, while source line, source and drain selectors, and wordlines have to be decoded on a layer basis.

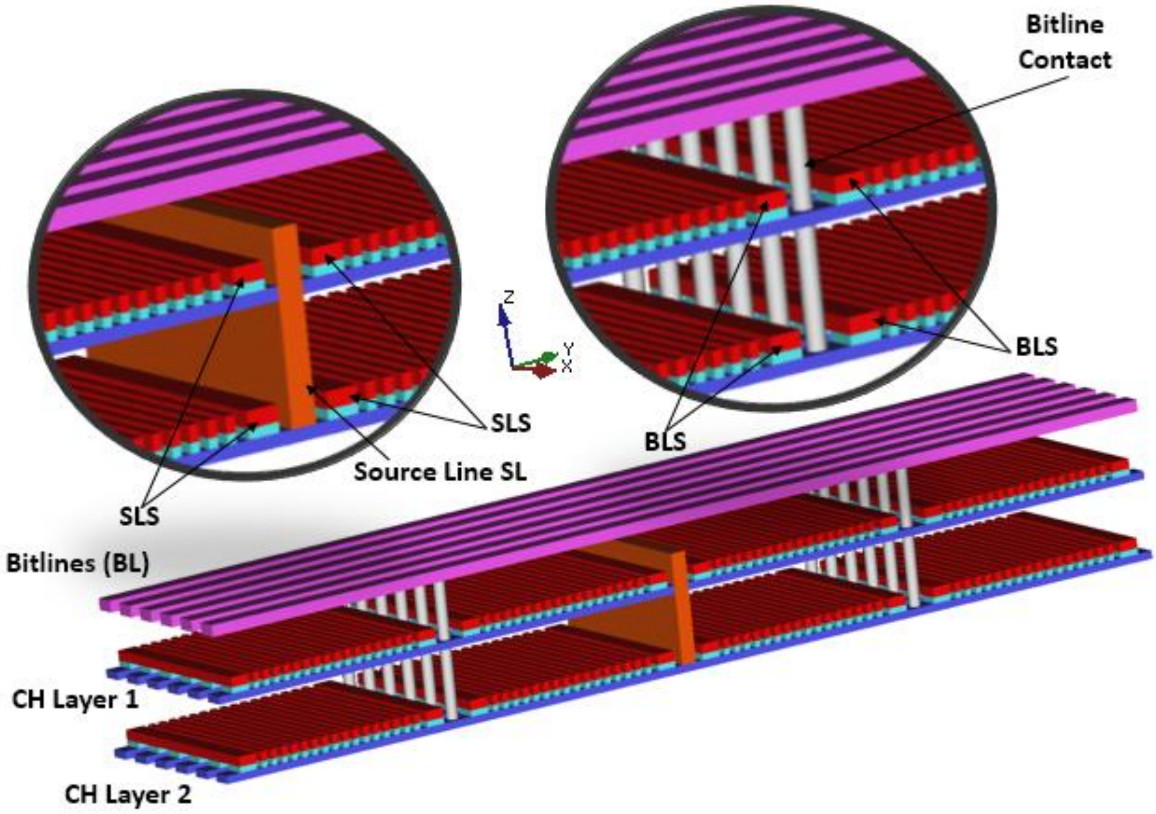


Figure 3.3 3D Flash as a simple stack of planar arrays

It is worth highlighting that the thermal budget necessary for growing additional silicon layers represents the biggest challenge as it can cause a non-uniform behavior of the memory cells belonging to different layers. Another important point to consider is the floating substrate (Figure 3.3), which significantly impacts the erase operation. As already mentioned, economics is the key driver for going 3D and this approach multiplies the cost of a single planar array by the number of layers, therefore, being no so effective. Indeed, each array layer ends up asking for 3 critical process modules, i.e. bitlines, wordlines, and contacts. The only cost saving comes from the fact that circuits and metal interconnections can be shared. Re-use of existing know-how probably explains the remarkable activity done in this area [9,10].

3.2 3D Charge Trap NAND Flash Memories

A much more effective 3D array can be built by vertically rotating the planar NAND Flash string of Figure 3.4(a), as displayed in Figure 3.4(b). The solution of choice is a conduction channel completely surrounded by the gate (Figure 3.4 (c) and (d)) [11]: indeed, the curvature effect helps increasing the electric field E_t across the tunnel oxide, and reduces the electric field E_b across the blocking oxide [12,13], and this has a positive impact on oxide reliability and overall power consumption.

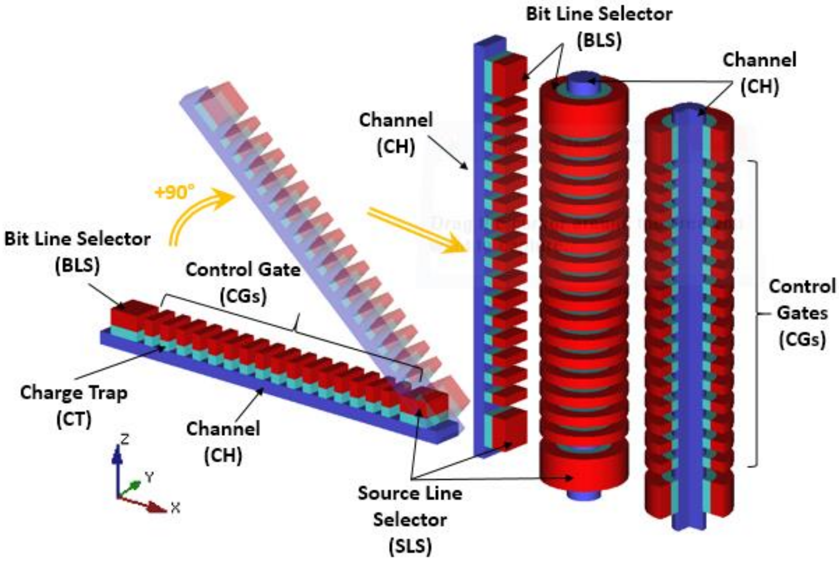


Figure 3.4 The NAND Flash string goes vertical

Vertical channel arrays have been historically driven by architectures known as BiCS, which stands for *Bit Cost Scalable* [14,15] and P-BiCS, acronym for *Pipe-Shaped BiCS* [16,17,18], which are both leveraging CT cells. Let's get started with BiCS, which is sketched in Figure 3.5 and Figure 3.6 [19]. There is a stack of *Control Gates* (CGs), the lowest being the one of the *Source Line Selector* (SLS). The whole vertical stack is punched through and the resulting holes are filled with poly-silicon; each filled hole (a.k.a. pillar) forms a series of memory cells vertically connected in a NAND fashion. *Bit Line Selectors* (BLS's) and *Bitlines* (BLs) are formed at the top of the structure [20].

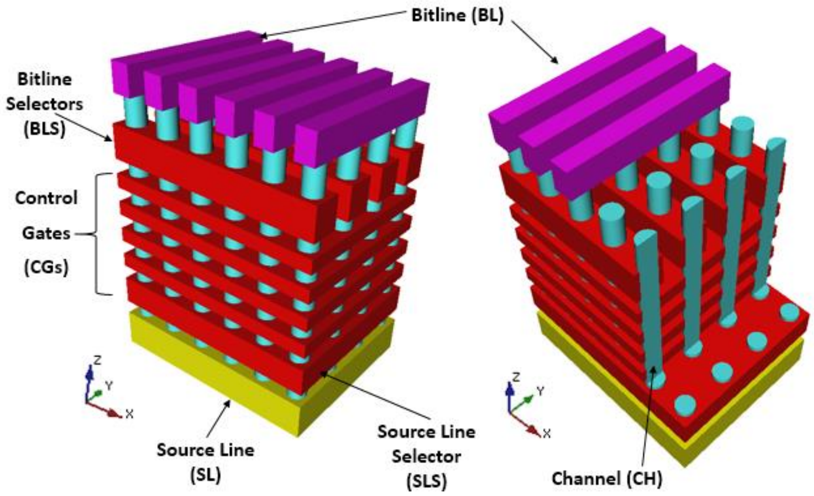


Figure 3.5 BiCS architecture

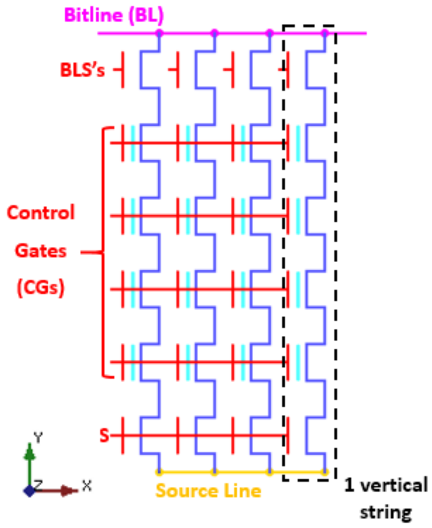


Figure 3.6 Equivalent circuit of a BiCS array

The poly-silicon body of memory cells is not doped or lightly doped [12,13]; indeed, considering the bad aspect ratio of the vertical polysilicon plug, p-n junctions cannot be easily realized by either diffusion or implantation in a trench structure. As usual, a select transistor (BLS) is used to connect each NAND string to a bitline; there is also another select transistor (SLS), which connects the other side of the string to the common source diffusion.

It is important to highlight that the number of critical and expensive lithography steps does not depend on the number of control gate plates because the whole 3D stack is drilled at one [21,22].

As sketched in Figure 3.7, vertical transistor have polysilicon body and this fact turned out to be one of the critical cornerstone of the 3D foundation. From a manufacturing perspective, the density of the traps at the grain boundary is very difficult to control, with such a vertical shape: the bad thing is that this poor control induces significant fluctuations of the characteristics of vertical transistors.

The recipe for fixing the trap density fluctuation problem is to manufacture a polysilicon body much thinner than the depletion width. In other words, by shrinking the polysilicon volume, the total number of traps goes down (Figure 3.8). This particular structure is usually referred to as *Macaroni Body* [15]. A *filler layer* (i.e. a dielectric film) is used in the central part of the macaroni structure, essentially because it makes the manufacturing process easier.

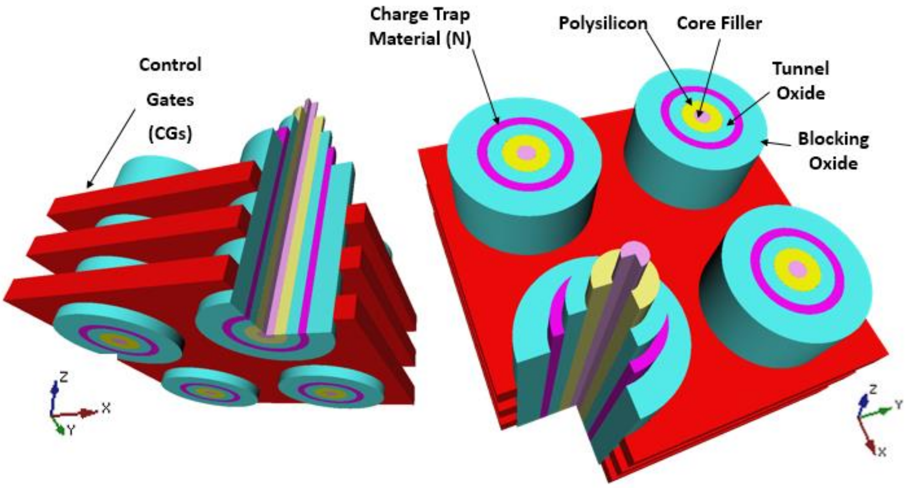


Figure 3.7 BiCS memory cells

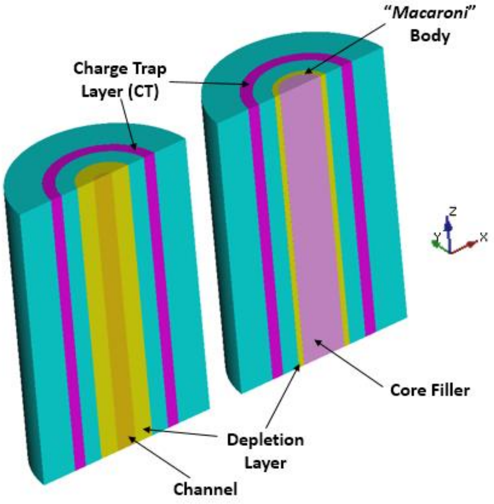


Figure 3.8 A vertical transistor (right) modified with *Macaroni* body (left)

The fabrication sequence of the BiCS array [23] starts from building the layers for control gates and selectors. Then, BLS stripes are defined. After forming pillars, bitlines are laid out by using a metal layer.

Control gate edges are extended to form a ladder to connect to the fan-out region, as sketched in Figure 3.9 [14,15,23,24]. Actually, there are 2 ladders: one of the 2 can't be used because it is masked by the metals biasing the bitline selectors.

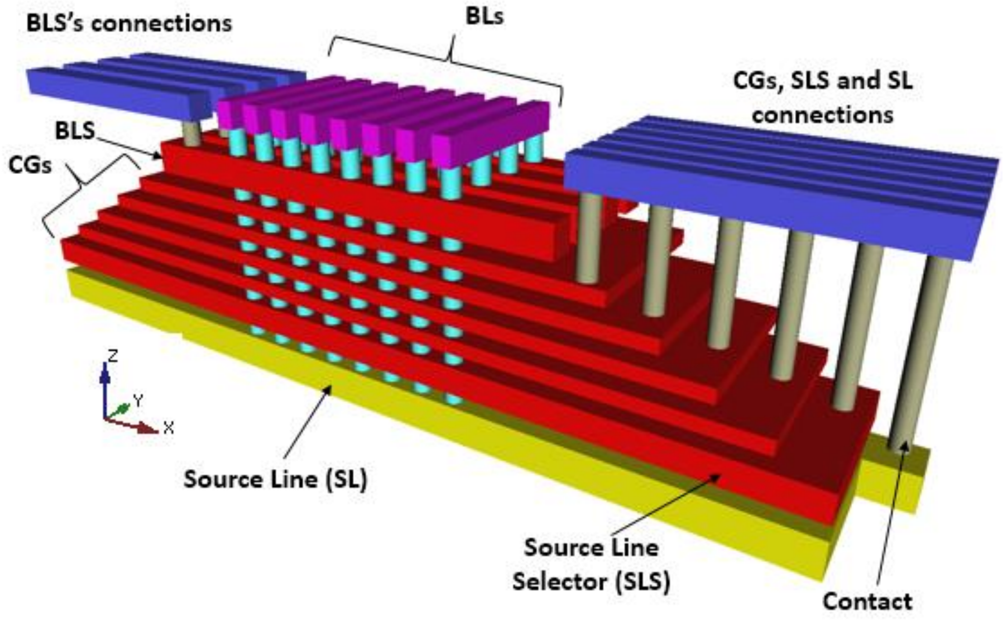


Figure 3.9 Fan-out of the BiCS array

Over time BiCS became P-BiCS, mainly to improve the Source Line resistance [25,26]. In a nutshell, two vertical NAND strings are shorted together at the bottom of the 3D structure: in this way, they form a single NAND string and the 2 edges are connected to the bitline and to the Source Line, respectively (Figure 3.10). Thanks to its U-shape, P-BiCS has few advantages over BiCS:

- retention is better because manufacturing creates less damages in the tunnel oxide;
- being at the top, the Source Line can be connected to a metal mesh, thus lowering its parasitic resistance;
- Source Line and bitline selectors are at the same height of the stack and, therefore, they can be equally optimized and controlled, thus obtaining a better string functionality.

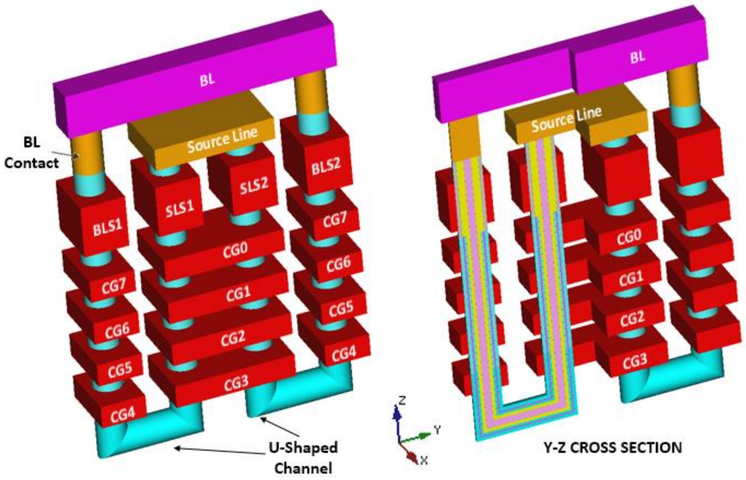


Figure 3.10 P-BICS NAND strings

Figure 3.11 shows a P-BiCS array [27].

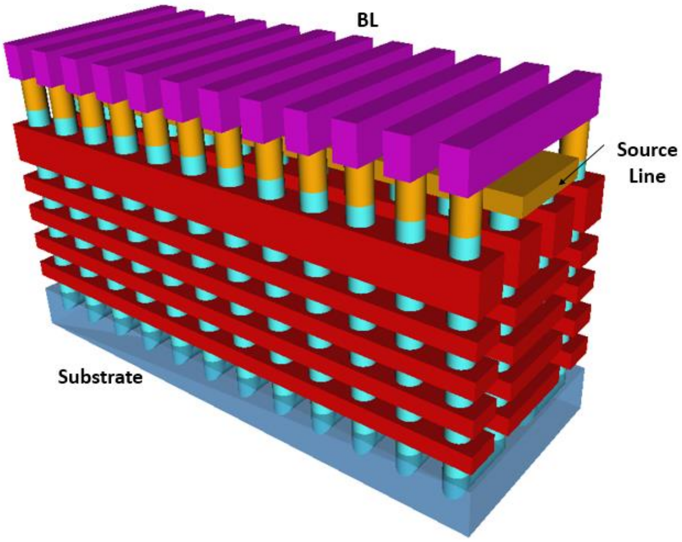


Figure 3.11 P-BICS NAND Flash array

One of the biggest drawbacks of P-BiCS is the fact that at the same height of the stack there are two different control gates which, of course, can’t be biased together; therefore, the two layers can’t be simply shorted together. As a result, compared to BiCS, a totally different and more complex fan-out is required [27], as displayed in Figure 3.12: basically, a fork-shaped gate is adopted, such that each branch acts on two NAND pages.

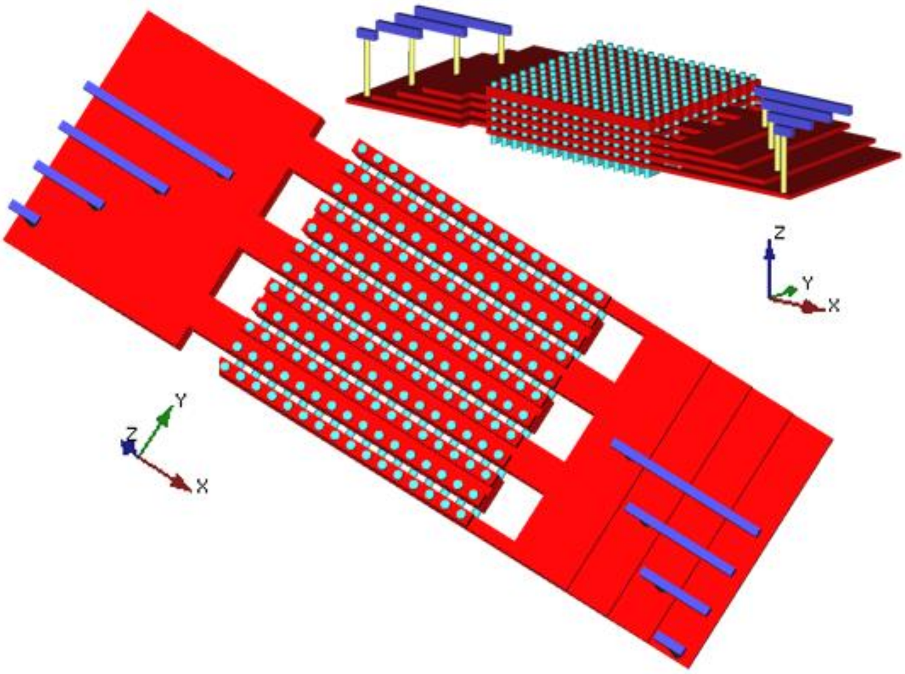


Figure 3.12 Fork-shaped fan-out

A major advantage is the easier connection of the source line [16] through the “Top Level Source Line” of Figure 3.13. This additional metal mesh guarantees a much better noise immunity for circuits.

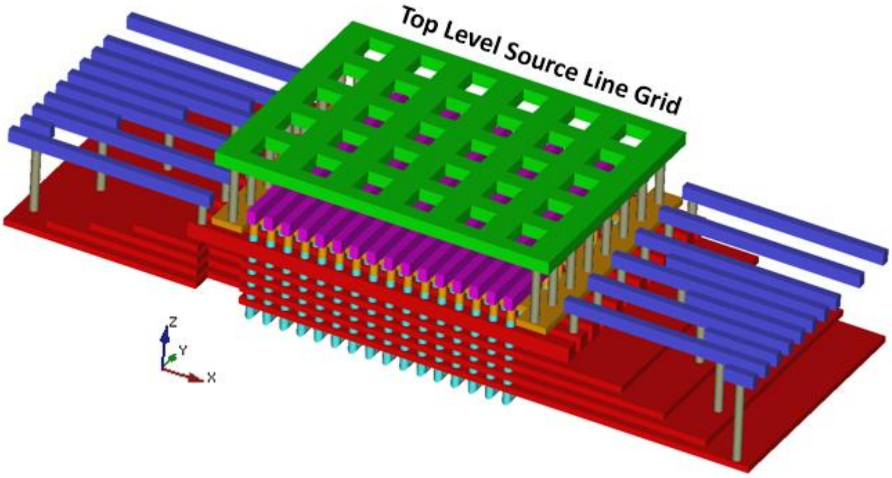


Figure 3.13 P-BiCS: Source Line metal mesh

Besides BiCS and P-BiCS, many other approaches were tried, including VRAT (*Vertical Recess Array Transistor*) [28], Z-VRAT (*Zigzag* VRAT) [28], and VSAT (*Vertical Stacked*

Array Transistor) [29], and 3D-VG (*Vertical Gate*) NAND [78] which is a unique architecture where the channel runs along the horizontal direction.

TCAT (*Terabit Cell Array Transistor*) was disclosed in 2009 [30] and it was the foundation for V-NAND (Figure 3.14), which is the first 3D memory device who reached the market. Except for SL+ regions which are n+ diffusions, the equivalent circuit of TCAT is the same of BiCS (Figure 3.6). All SL+ lines are connected together to form the common Source Line (Figure 3.15). There are 2 metal layers for decoding wordlines and NAND strings, respectively.

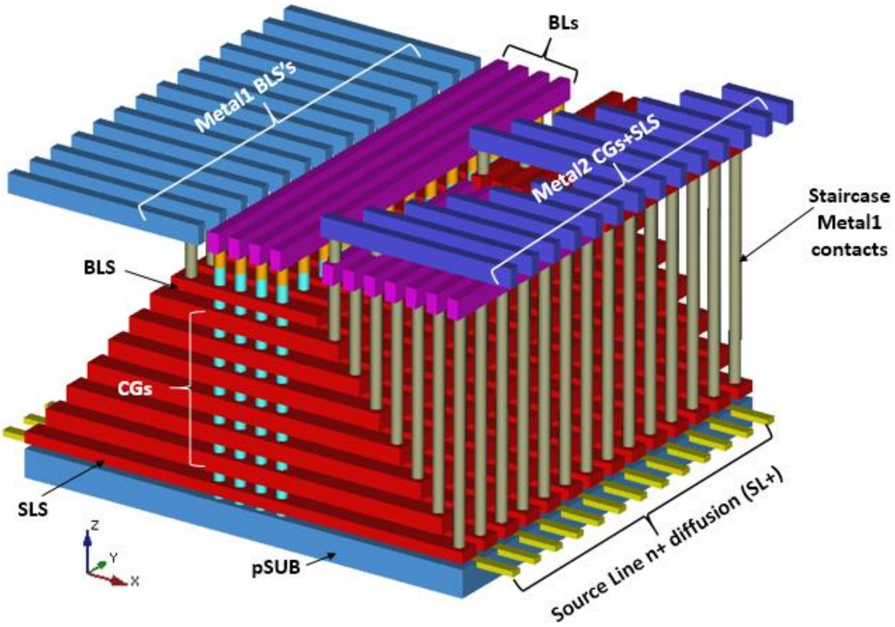


Figure 3.14 TCAT NAND Flash array

TCAT is based on *gate-replacement* [30], whereas BiCS is *gate-first*. Gate-replacement begins with the deposition of multiple oxide/nitride layers. After the stack formation, nitride is removed through an etching process. Afterwards, tungsten metal gates are deposited and, finally, gates are separated by using another etching step. Metal gates translate into a lower wordline parasitic resistance, resulting in faster programming and reading operations.

The bulk erase operation is another significant difference compared to BiCS. Because NAND strings are close to n+ areas, during erasing, holes can come straight from the substrate, thus avoiding the GIDL (*Gate Induced Drain leakage*) on the source side, which is a well-known problem for BiCS.

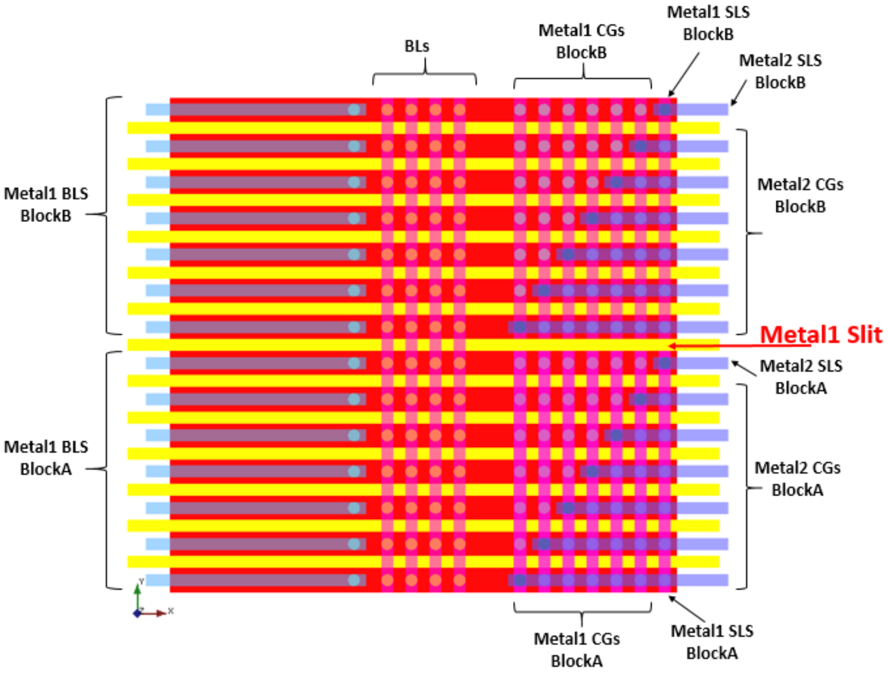


Figure 3.15 Top view of Figure 3.14

BiCS and TCAT are compared in Figure 3.16 [31]. Being TCAT based on a gate-last process, the charge trap layer is biconcave, and thanks to this particular shape it is much harder for charges to spread out. On the contrary, BiCS is characterized by a charge trapping layer going through all gate plates, thus acting as a charge spreading path: of course, the main consequence of this layout is a degradation of data retention.

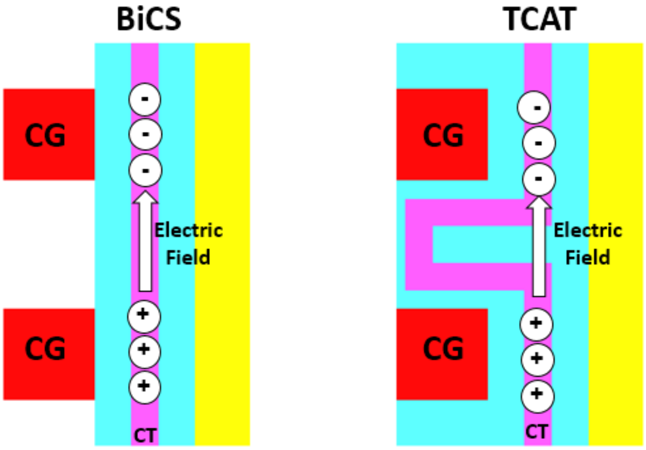


Figure 3.16 BiCS vs. TCAT

TCAT evolved into another architecture called V-NAND [32]. As depicted in Figure 3.17, the first generation, V-NAND Gen1, had 24 wordline layers, plus additional dummy wordline layers (dummy CG) [33-35].

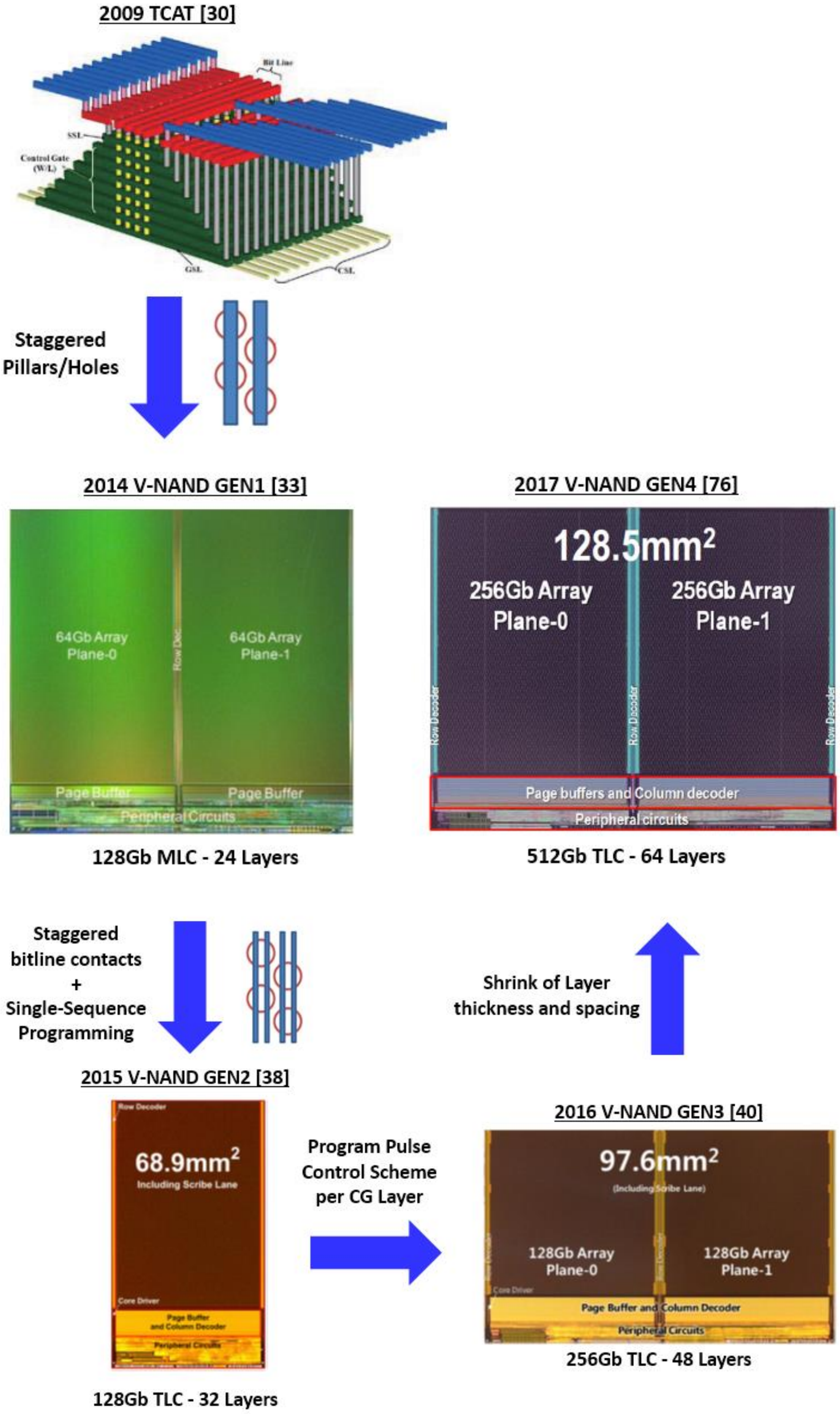


Figure 3.17 Evolution from TCAT to V-NAND (reproduced with permission from [30,33,38,40,76])

Why dummy layers? Mainly because of the floating body of the memory cells with vertical channel. In fact, during the programming operations, hot carriers are generated by the high lateral electric field located at the edge of the NAND string. Therefore, these hot carriers keep the voltage on the channel low during the programming operation of the first wordline (i.e. Program Disturb). Dummy wordlines before the first WL are an effective and simple solution to this problem [36,37].

To reduce the area of the memory array, V-NAND Gen1 adopts a *Staggered Pillars* layout. In addition to that, V-NAND Gen2 introduced a new layout for bitline contacts, known as *Staggered Bitline Contacts* [38]. More details about these architectures in Section 3.5.

A 128 Gb TLC (3 bit/cell) device manufactured by using V-NAND Gen2 was published in 2015 [38,39]. Gen2 had 32 memory layers instead of the previous 24 and introduced the concept of Single-Sequence Programming. Conventional (mainly 2D) TLC programming techniques go through the programming sequence multiple times. To be more specific, each wordline is programmed 3 times, such that V_{TH} distributions can be progressively tightened. Because of the smaller cell-to-cell interference (compared to FG), CT cells exhibit an intrinsic narrower native V_{TH} distribution. As a result, V-NAND Gen2 could write 3 pages of logic data in a single programming sequence. There are 2 benefits to this approach: reduced power consumption and faster programming.

V-NAND Gen3 appeared in 2016 [40], in the form of a 48 layer TLC device. With such a high number of gate layers, the very high aspect ratio of the pillar becomes a serious challenge for the etching technology. To mitigate this problem, the easiest solution is to shrink the thickness of gate layers. The downside of this approach is that the parasitic RC of the wordline gets higher, thus slowing access operations to the memory array. Moreover, channel’s size fluctuations become critical. Indeed, pillars are holes drilled in the gate layer and they represent a barrier for charges flowing along the wordline: in essence, a distribution of the holes diameters generates a distribution of the parasitic resistances of gate layers. In addition, pillars, once manufactured, have the conic shape sketched in Figure 3.18. The overall result is that the same voltage applied to different gate layers translates into a waveform per layer. An adaptive program pulse scheme can fix the problem. In a nutshell, the program pulse duration has to be tailored to the characteristics of the wordline layer. As the number of layers increases, the pillar becomes longer with a negative impact on the aspect ratio of the pillar. To compensate for that, V-NAND Gen4 [76], which is built on a stack of 64 layers, had to shrink both the layer thickness and the intra-layer distance (spacing). The downside is an increased wordline parasitic capacitance which adversely affects cell's reliability and timings. Improved circuits and programming algorithms can be used to tackle this problem [76].

As discussed, both BiCS [77] and V-NAND use CT cells, but Floating Gate still exists, as explained in the next Section.

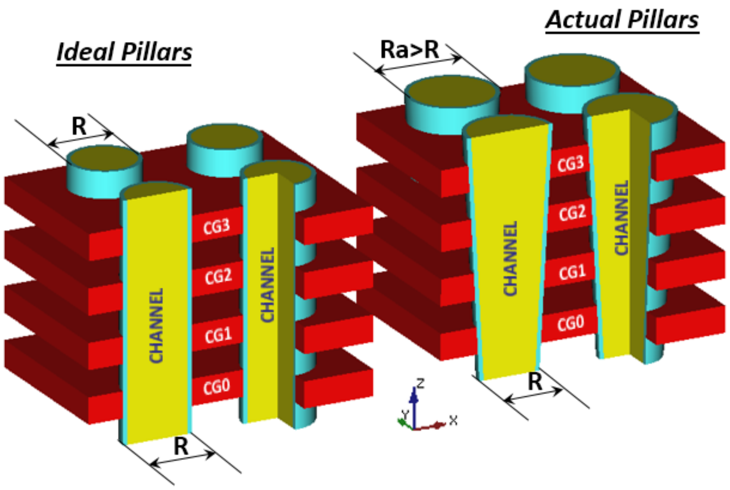


Figure 3.18 Ideal vs. actual shape of pillars

3.3 3D Floating Gate NAND Flash Memories

2D NAND Flash memories use FG cells which have been, improved and optimized for decades. Of course, there have been many attempts to reuse this know-how in 3D.

The first 3D attempt is known as *3D Conventional FG* (C-FG) or S-SGT (*Stacked-Surrounding Gate Transistor*) [41-43], and it is sketched in Figure 3.19.

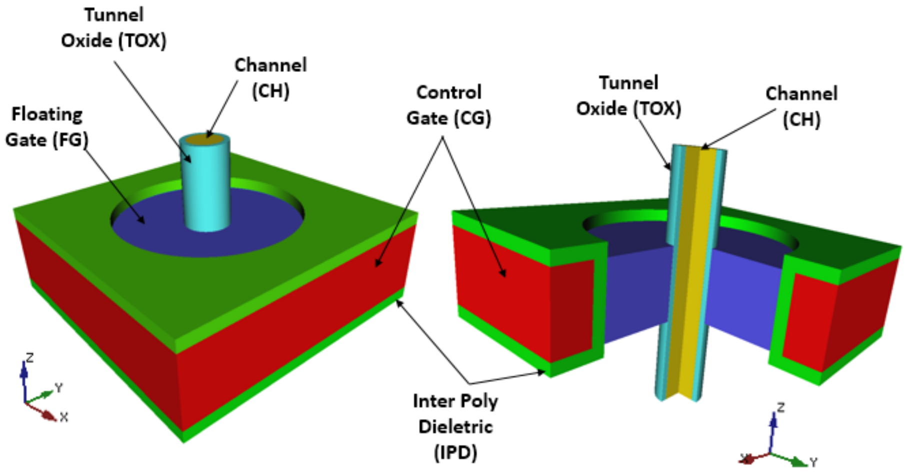


Figure 3.19 3D C-FG cell

A C-FG NAND string is shown in Figure 3.20, including select transistors. Please note that both string selectors are manufactured as standard transistors, i.e. they haven't any floating gate. Figure 3.21 shows a C-FG array and Figure 3.22 adds the fan-out region. While all wordlines at the same height of the stack are connected, BLS lines can't, because they need to be page selective per each CG layer. On the contrary, SLS transistors can be shorted together, thus saving both power and silicon area.

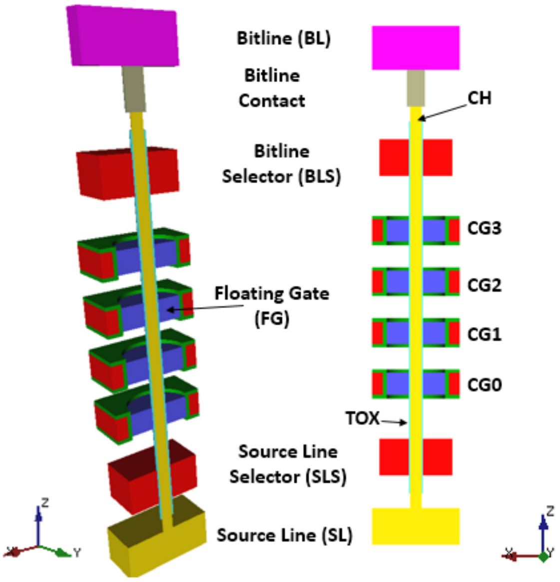


Figure 3.20 C-FG NAND Flash string

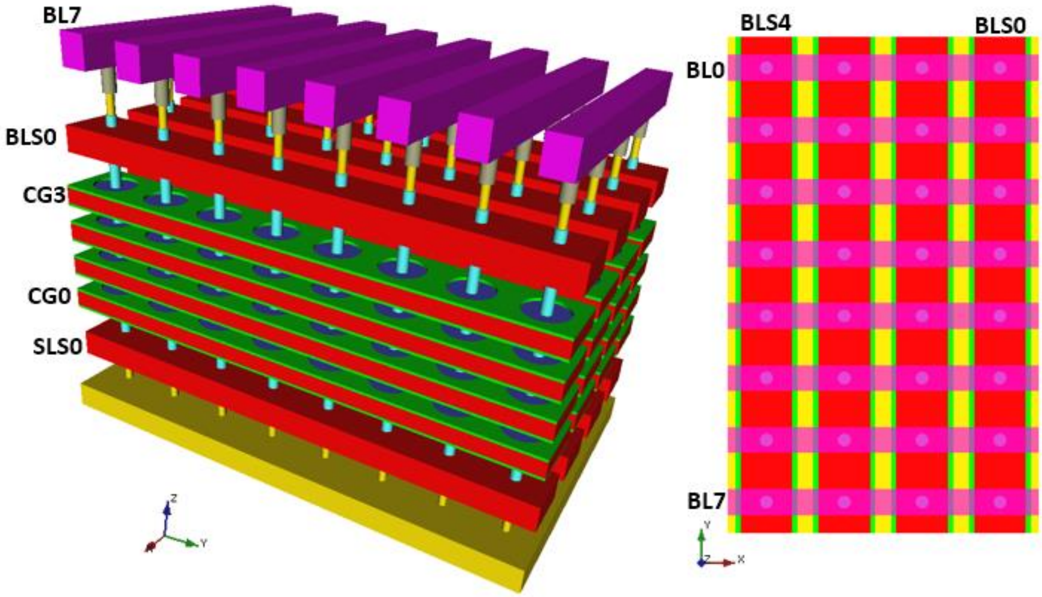


Figure 3.21 C-FG NAND Flash array

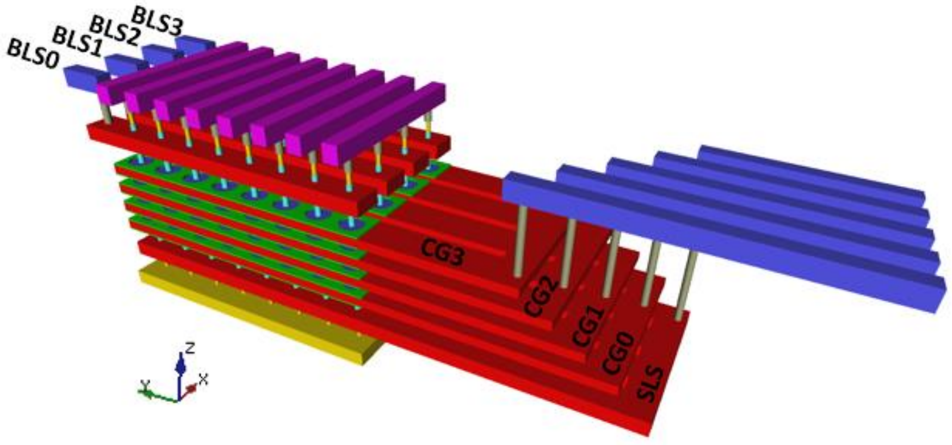


Figure 3.22 C-FG NAND Flash array with fan-out

As already discussed in the previous Section, the Source Line is the *local ground* of memory cells. A big single Source Line plate laid out at the bottom of the stack, with a limited number of contacts, simply doesn't work: when tens of thousands of cells sink current, managing the voltage on the source side becomes a real challenge. Having more contacts to the Source plate is not an option. The *Source Line Metal Grid* sketched in Figure 3.23 fixes this problem.

As already discussed, slits between NAND blocks are the most common way for reducing program/read disturbs and parasitic loads. Of course, there is no need to cut bitlines and Top Source Lines. This is fundamentally the same approach adopted in BiCS.

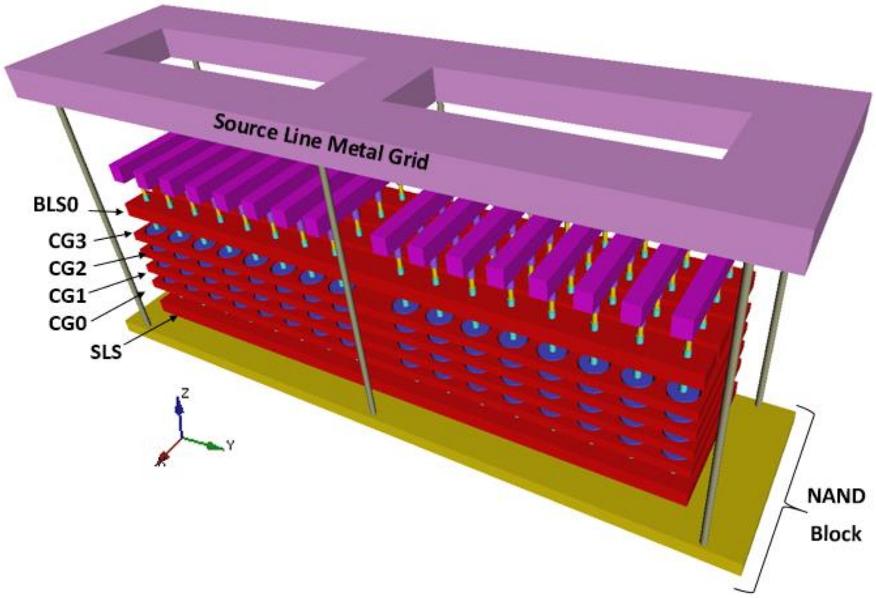


Figure 3.23 C-FG array with Source Line Metal Grid

Because we are talking about FG cells, FG coupling between neighboring cells is the main hurdle for vertical scaling. With enhancement-mode operations, the high resistance of source/drain (S/D) regions should also be carefully considered. In fact, these regions need high-doping and this is not very easy to accomplish when the conduction channel is made of polysilicon. The solution to this problem is to electrically invert the S/D layer by using higher voltages during read. This simple solution is hardly manageable by C-FG cells because of the thin FG.

The *Extended Sidewall Control Gate* (ESCG) structure, Figure 3.24 [44], is another FG option and it was developed to contain the interference effect. Moreover, by applying a positive voltage to the ESCG structure, density of electrons on the surface of the pillar can be much higher than C-FG (even one order of magnitude): a highly inverted electrical source/drain can significantly lower the S/D resistance.

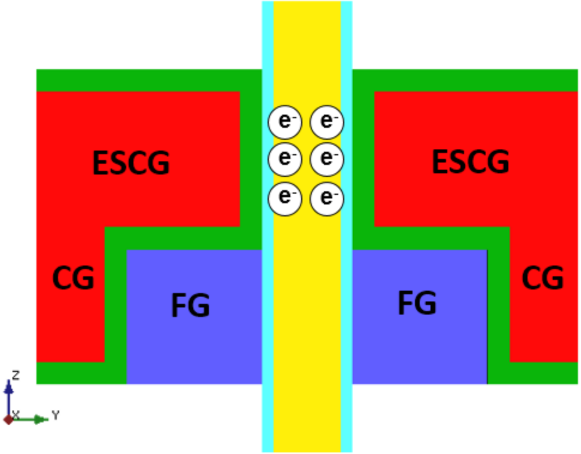


Figure 3.24 ESCG NAND Flash cell

In addition, the ESCG shielding structure reduces the FG–FG coupling capacitance: the ESCG region is biased as CG, and the CG coupling capacitance (C_{CG}) is significantly increased because of the increased overlap area between CG and FG. A higher CG coupling ratio is one of the key ingredients for achieving effective NAND Flash operations [45].

Another FG cell is DC-SF (*Dual Control-Gate with Surrounding Floating Gate,* Figure 3.25) [46]. This time FG is controlled by two CGs. The impact on the FG/CG coupling ratio is remarkable, thanks to the enlargement of the FG/CG overlap area. Another positive aspect is the reduction of the voltages required for programming and erasing. DC-SF eliminates the FG-FG interference because the CG between two adjacent FGs plays the role of an electrostatic shield [47].

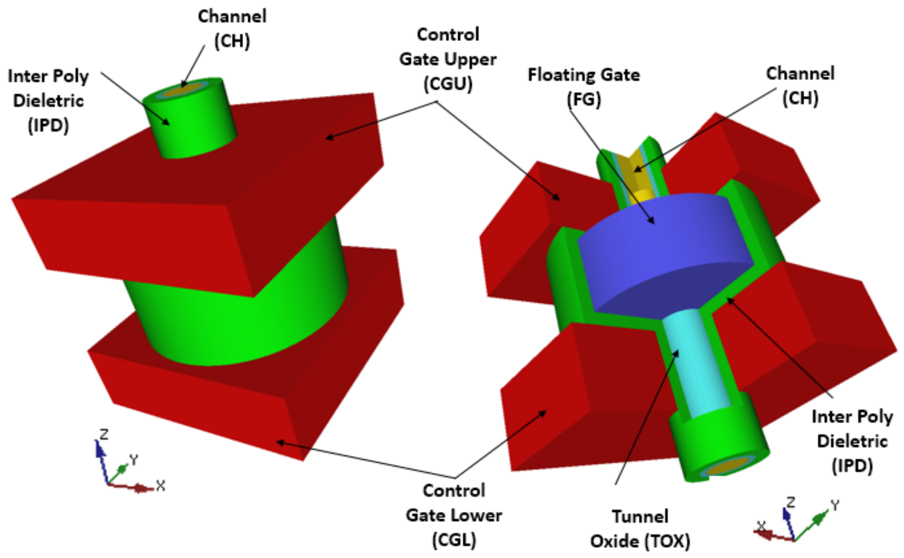


Figure 3.25 DC-SF NAND Flash cell

FG is fully isolated by IPD (*Inter Poly Dielectric*) and capacitive coupled to upper and lower control gates, CGU and CGL, respectively. The tunnel oxide is located between the channel CH and FG, while IPD is on the sidewall of the CG. In this way, free charges cannot tunnel to the control gates.

BiCS and DC-SF NAND strings are sketched in Figure 3.26. In BiCS the nitride layer, going across all gates, makes the cell prone to data retention issues. On the contrary, the surrounding FG is totally isolated: it is much easier for DC-SF to retain electrons [48,49]. Of course, the downside of DC-SF is the fact there are two gate layers instead of one, coupled with much more complex biasing schemes [50,51].

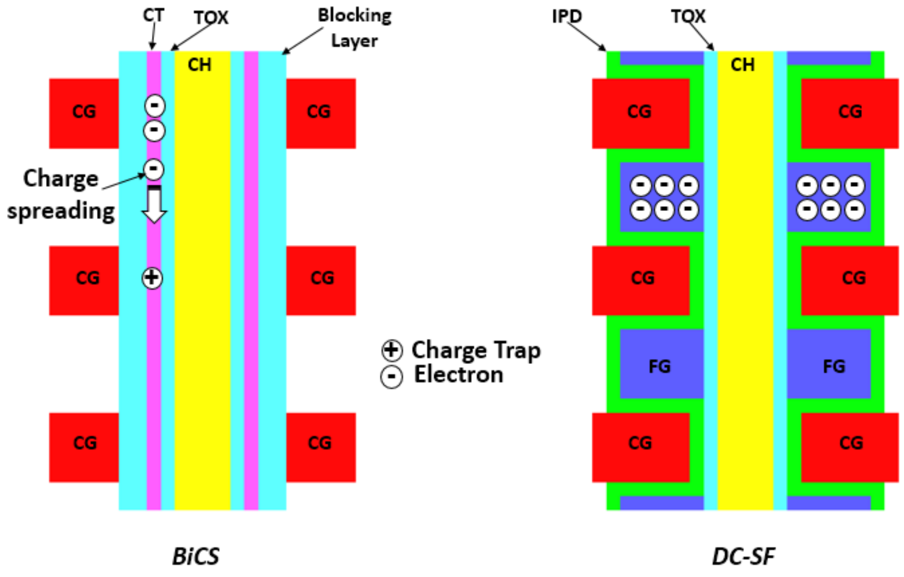


Figure 3.26 BiCS vs. DC-SF

The *Separated Sidewall Control Gate* (S-SCG) Flash cell [52] displayed in Figure 3.27 is another 3D FG option developed around the sidewall concept.

One of major drawbacks of this cell is the "direct" disturb to the neighboring passing cells, caused by the high SCG/FG coupling capacitance. We define it as "direct" because the sidewall CG is shared between adjacent cells: as a matter of fact, biasing SCG means biasing both FGs.

To minimize the decoding complexity, all SCGs belonging to one block adopt a common SCG scheme; besides their electrostatic shield functionality, sidewall gates can help all memory operations [53]. For instance, the common SCG is biased at 1V during read operations, thus electrically inverting the channel (same as ESCG). Compared to ESCG, the electrical inversion happens simultaneously on source and drain, exactly because of the sidewall gates (Figure 3.28). Same thing happens during programming: the common SCG is biased at a medium voltage to improve the channel boosting efficiency.

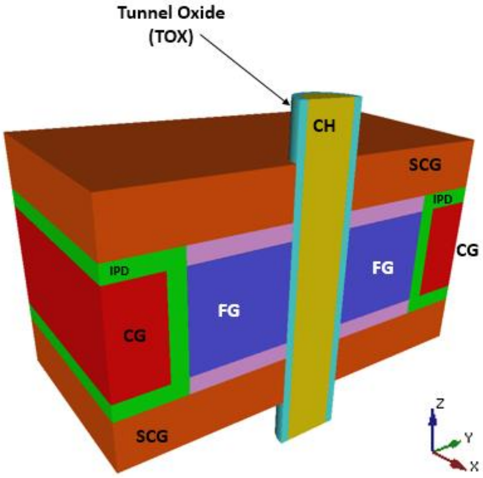


Figure 3.27 S-SCG NAND Flash cell

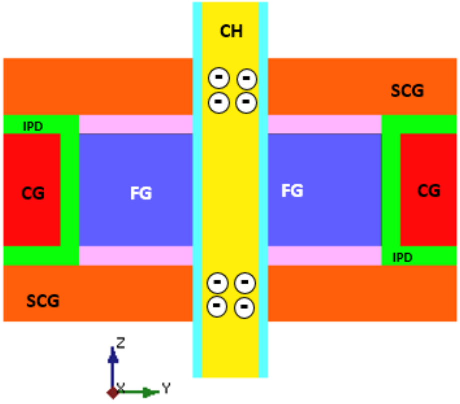


Figure 3.28 Common SCG approach to enable Source/Drain inversion

Besides the direct disturb, another problem of Sidewall Gates is the limitation of vertical scaling to around 30nm; indeed, the thicknesses of SCG and IPD can't be scaled too much, otherwise they would breakdown when voltages are applied.

Let's now take a look at examples of 3D FG NAND memory arrays of hundreds of Gb. As shown in Figure 3.29, the first 3D FG device was published in 2015 [54], in the form of a 384Gb TLC NAND based on C-FG. This memory device was built on stack of 32 (+ dummy) memory layers.

A 768Gb 3D FG NAND became public in the following year [55]. What is unique in this case is the fact that the area underneath the array was used for circuitry. More details about this approach are provided in Section 3.6.

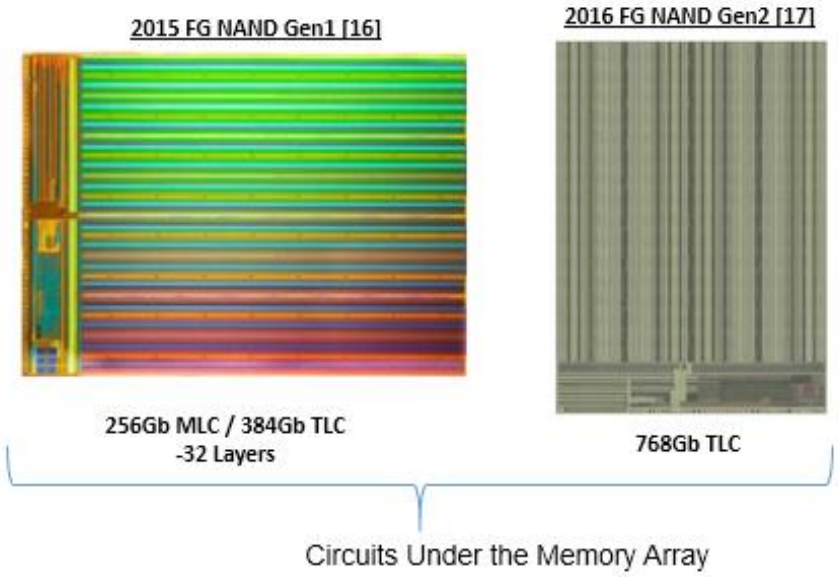


Figure 3.29 3D FG NAND devices [16,17]

The next Section deals with the most advanced array architectures; all these solutions can be integrated independently from the storage technology (FG or CT).

3.4 Advanced Layout Solutions

As discussed in the first part of this chapter, the big push for 3D is coming from the need of increasing the *Bit_Density* [56], which can be defined as follows.

$$Bit\_Density = \frac{Die\_Capacity}{Die\_Size} \qquad (3.1)$$

If we define the area of the memory matrix as A_{MAT}, and the area of the peripheral circuits as A_{PERI} we can then write

$$Die\_Size = A_{MAT} + A_{PERI} \qquad (3.2)$$

For a planar memory, A_{MAT} is:

$$A_{MAT} = \frac{Die\_Capacity \cdot A_{CELL}}{n_{bitpercell}} \qquad (3.3)$$

where A_{CELL} is the cell's area, and $n_{bitpercell}$ is the number of logic bits stored in a single physical cell.

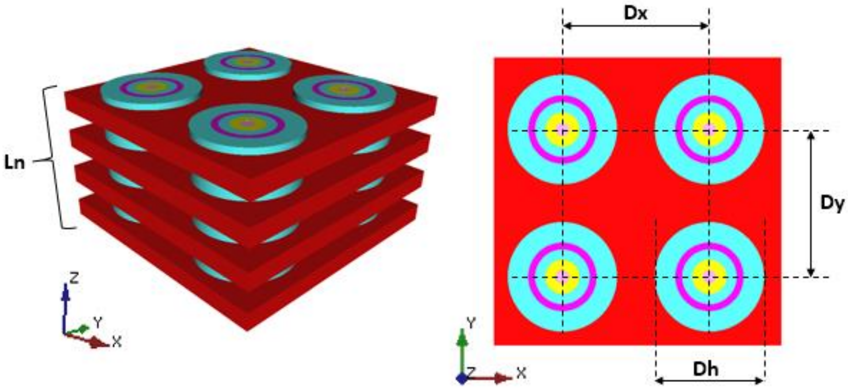


Figure 3.30 3D array

If we consider a 3D array, Figure 3.30, Equation (3.3) becomes

$$A_{MAT} = \frac{Die\_Capacity \cdot A_{CELL}}{Ln \cdot n_{bitpercell}} \qquad (3.4)$$

where Ln is the number of gate layers, and A_{CELL} is

$$A_{cell} = Dx \cdot Dy \qquad (3.5)$$

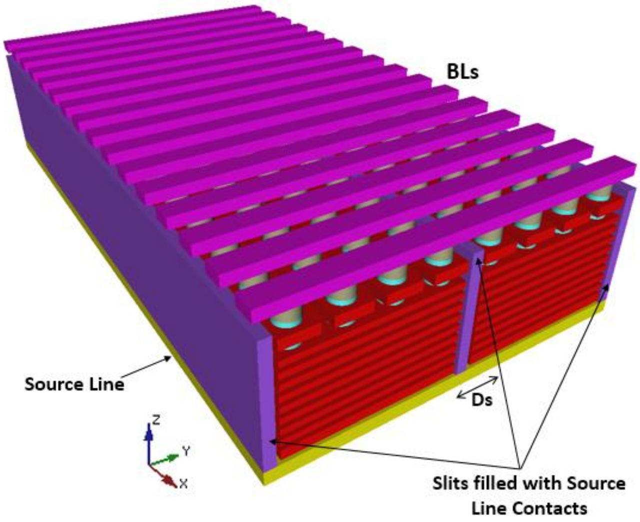


Figure 3.31 Array slits

Equation (3.4) assumes that cells are simply laid out along X and Y directions, without any other elements. Actually, this is not true. 3D NAND suffers from Program Disturb more than 2D, because several NAND pages belong to the same physical layer [57, 58]. Simply put, Program Disturb is comes from the X direction in 2D, but it involves Y when looking at 3D arrays.

The only way to fix this problem is the reduction of the size of CG layers, and this is accomplished by placing slits at a regular frequency, as displayed in Figure 3.31. A smaller gate layer means also a smaller logic NAND block, and this is extremely valuable for the data management algorithms running within a Solid State Drive [59].

The slit overhead Doh per NAND block can be written as

$$Doh = Ds - Dy \qquad (3.6)$$

where Ds is the distance between two pillars when there is a slit between them.

In each block there are p pillars, and the overhead per memory cell $Doheff$ becomes

$$Doheff = \frac{Ds - Dy}{p} \qquad (3.7)$$

Therefore, the effective cell size A_{cell_eff} is

$$A_{cell\_eff} = Dx \cdot (Dy + Doheff) \qquad (3.8)$$

A_{cell_eff} can replace A_{cell} in Eq. (3.4). Of course, the higher the number of pillars is, the higher the bit density is, but the worse the Program and Read Disturbs are.

One way for compensating the impact of slits on the array size is the adoption of the so-called *staggered pillars* layout [60-62].

To illustrate the concept we can assume *Dx* = *Dy* = *D* and re-draw Figure 3.30 as shown in Figure 3.32. We start drawing a circle centered in “O” with radius r = *D*, and then we move from point “A” to point “B” along the circle, having $\overline{BC} = D$. With this simple geometrical trick, the distance between pillars is still *D* (and this is true for all pillars), but the Y pitch got reduced by *Δy*. Along the orthogonal direction, the matrix is enlarged by *Δx*, but X corresponds to the wordline direction (Figure 3.31): in modern Flash devices each wordline is made of 16k bytes and, therefore, *Δx* is actually negligible.

At this point we can re-write Equation (3.5) as follows

$$A_{cell} = D \cdot (D - \Delta y) \qquad (3.9)$$

with

$$\Delta y = D - D\cos\alpha = D(1 - \cos\frac{\pi}{6}) \qquad (3.10)$$

All in all, the above described rotation can save as much as 13.5% of the matrix size along one direction.

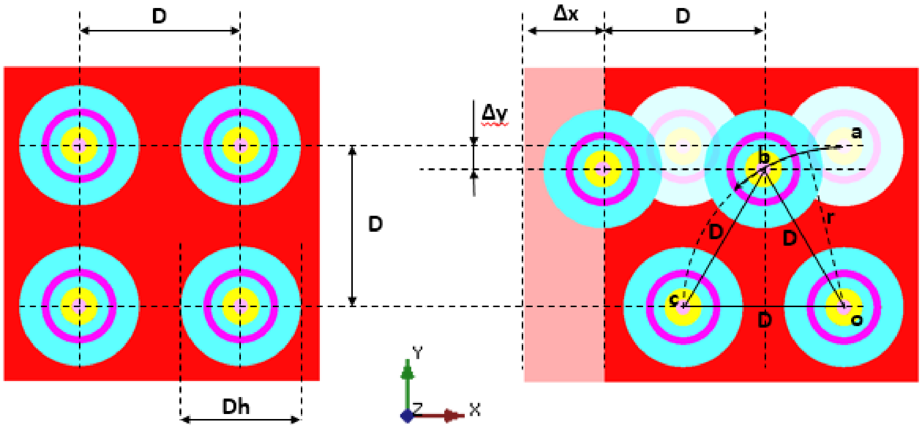


Figure 3.32 Conventional (left) and staggered (right) pillar layout

While increasing *Bit_Density*, the staggering technique changes the bitline density, as sketched in Figure 3.33. In fact, even and odd rows of pillars along the X axis (wordline direction) are not aligned anymore.

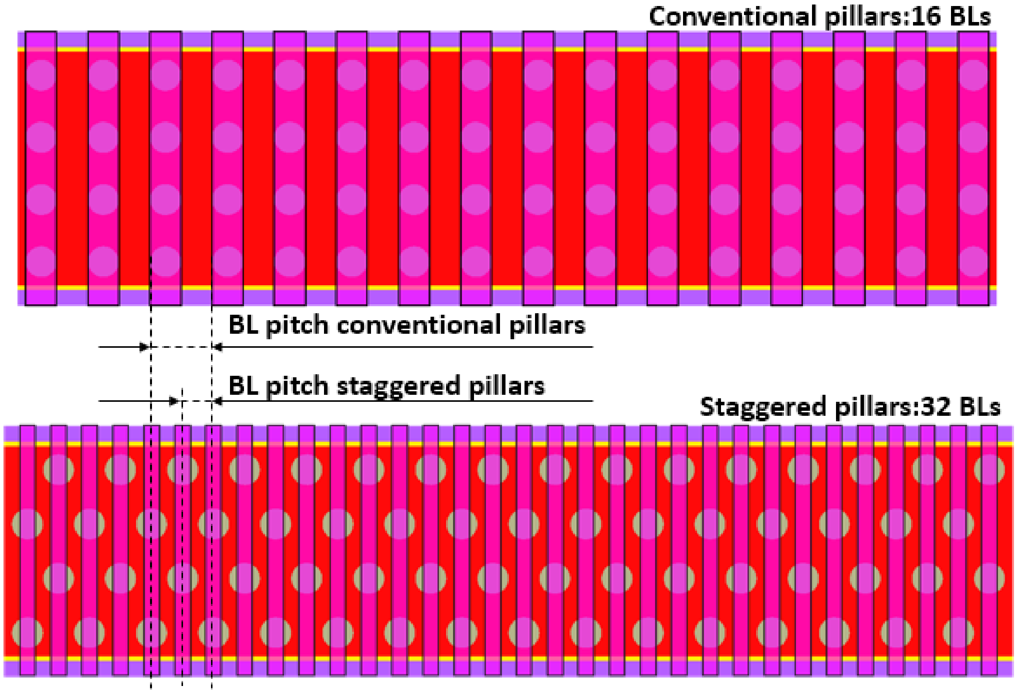


Figure 3.33 Bitline density with (bottom) and without (up) staggering of pillars

After staggering the pillars along the wordline direction, the NAND array looks like Figure 3.34. Two zoom boxes have been added to Figure 3.34 for a better understanding. All 3D NAND with vertical channels can be modified to accommodate the staggering of pillars.

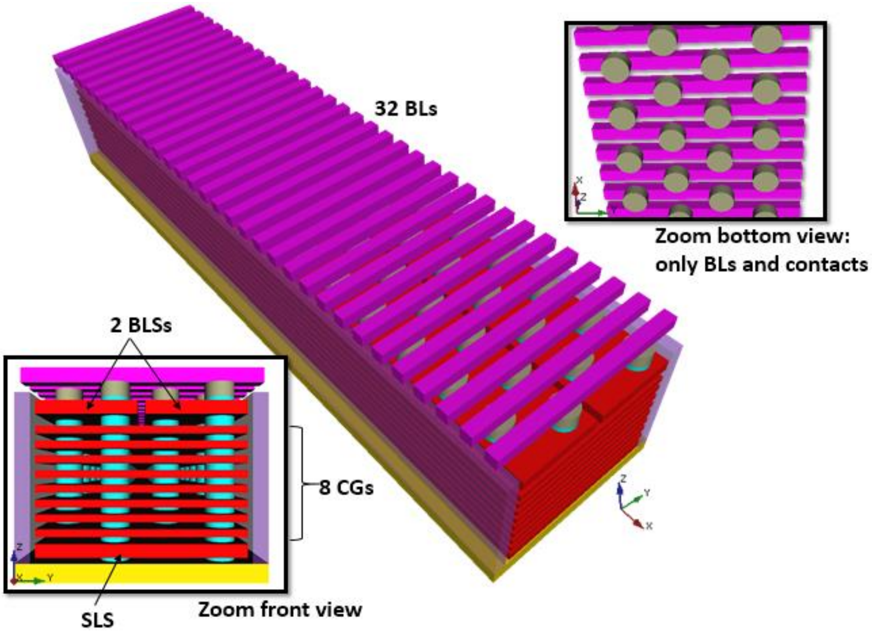


Figure 3.34 3D NAND Flash matrix with staggered pillars

The staggering technique can also be applied to the contacts between strings and bitlines. Figure 3.35 shows a NAND page spread across four rows of pillars. So far we always had one bitline per column of pillars, but now we have two: one bitline is connected to the even row of pair1, while the other one is for the even row of pair0 (Figure 3.35). Same applies to odd rows.

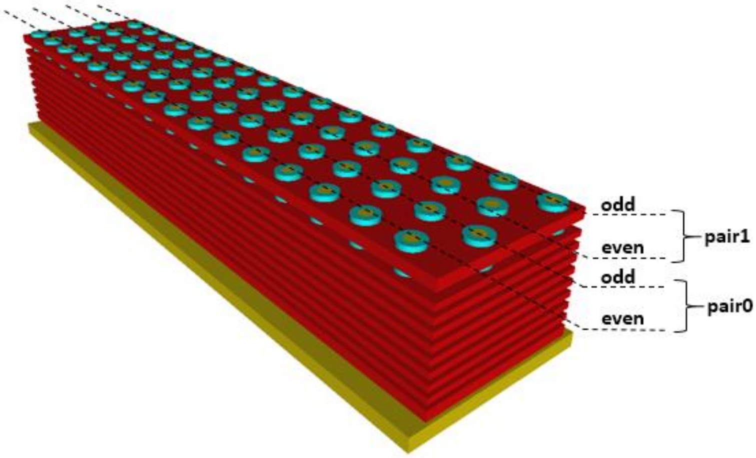


Figure 3.35 Single NAND page made of four rows of pillars

Figure 3.36 should help to understand the concept even better. At the top of the figure we have Figure 3.35 seen from the top. The middle section adds the bitlines and it highlights that two bitlines need to fit into pitch of pillars. At the bottom of Figure 3.36 bitlines are made transparent to show bitline contacts and pillar contacts.

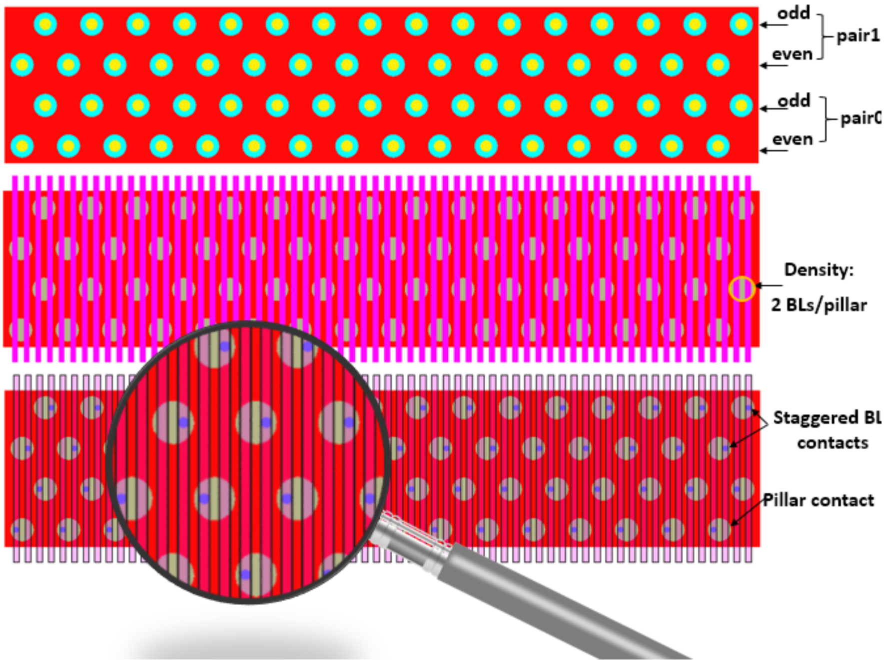


Figure 3.36 Staggered bitline contacts

Bitline contacts can be staggered in several ways; Figure 3.37 shows the so-called bracket-shaped approach [63, 64]. Each bracket connects a pair of pillars to the same bitline. The actual bitline contact is indicated with a circle.

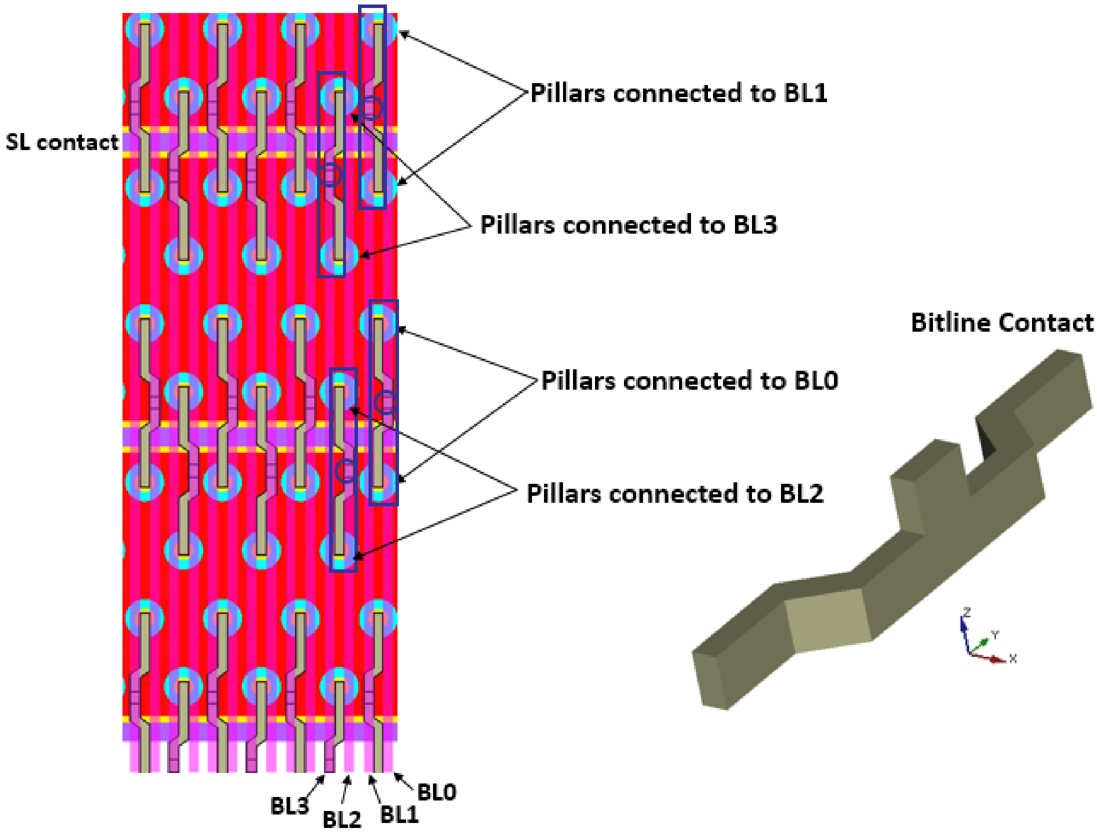


Figure 3.37 Staggering of BLs with bracket-shaped contacts

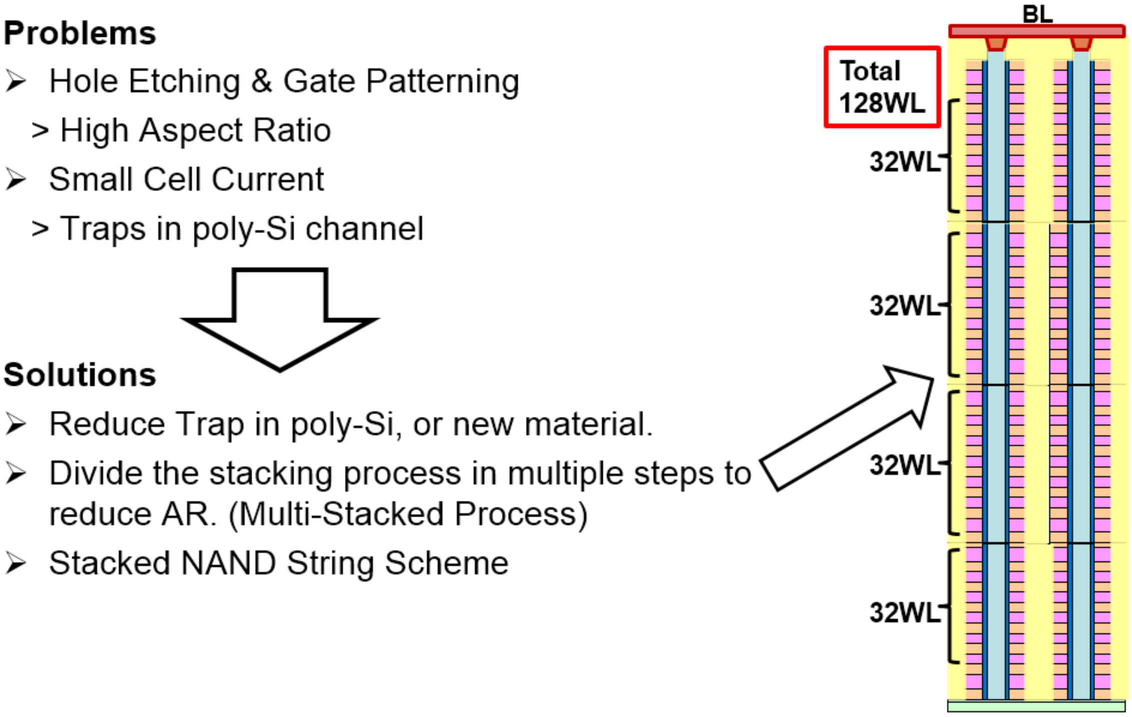
To summarize this Section, we can state that the area penalty caused by the need of frequently contacting the Source Line layer can be offset by adopting staggering techniques.

3.5 Key Challenges for 3D Flash development

In this Section we cover some of the key challenges that technologists and designers are facing to push 3D memories even further.

3.5.1 Number of Layers

To reduce the bit size, the number of stacked cells needs to go up, but this causes a bunch of problems hard to solve, as shown in Figure 3.38 [6].



Problems

- Hole Etching & Gate Patterning
  - > High Aspect Ratio
- Small Cell Current
  - > Traps in poly-Si channel

Solutions

- Reduce Trap in poly-Si, or new material.
- Divide the stacking process in multiple steps to reduce AR. (Multi-Stacked Process)
- Stacked NAND String Scheme

Figure 3.38 Challenges for increasing the number of 3D layers [6]

Pillar's *Aspect Ratio* (AR) is definitely the first challenge to overcome; in a stack of 32 cells AR can already be as high as 30. In this context, hole etching and gate patterning are extremely difficult, but of paramount importance.

A possible solution to this problem is to divide the stacking process in more steps to reduce the corresponding AR. For example, a NAND string made of 128 cells can be divided in 4 groups of 32 cells each, as shown in Figure 3.38. The downside of this solution is the cost of the stacking process (in this example, 4 times higher than the cost of the plain solution).

Second problem is the small cell current [65]. With 2D sensing schemes, a 200 nA/cell saturation current is considered the right value because it gives a reasonable sensing margin. Unfortunately, as shown in Figure 3.39, already with a stack of 24 layers, the cell current is just ~20% of FG cell. And it becomes lower and lower as the number of cells in the vertical stack increases. There are a couple of possible paths to solve this problem: sensing schemes with higher sensitivity, and the introduction of new materials enabling a higher cell mobility in the poly-Si channel (i.e. a higher current) [80,81,82,83].

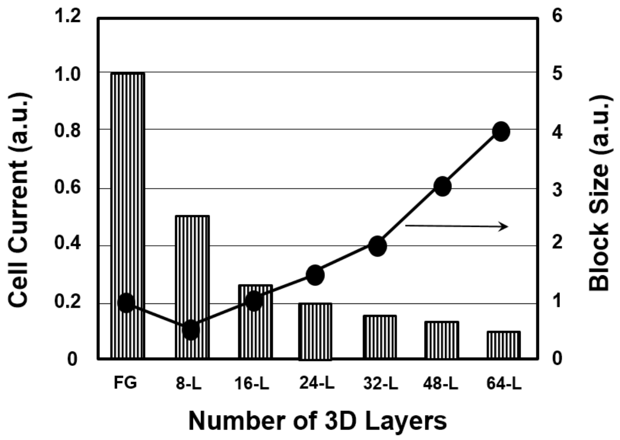


Figure 3.39 Cell current and block size vs. the number of 3D layers [65]

All the above mentioned problems can be fixed if NAND strings could be stacked as depicted in Figure 3.40 [6][66][67]. In this case, either bitlines or source lines are fabricated between NAND strings. This special architecture can simultaneously reduce the aspect ratio and increase the sensing current at same time.

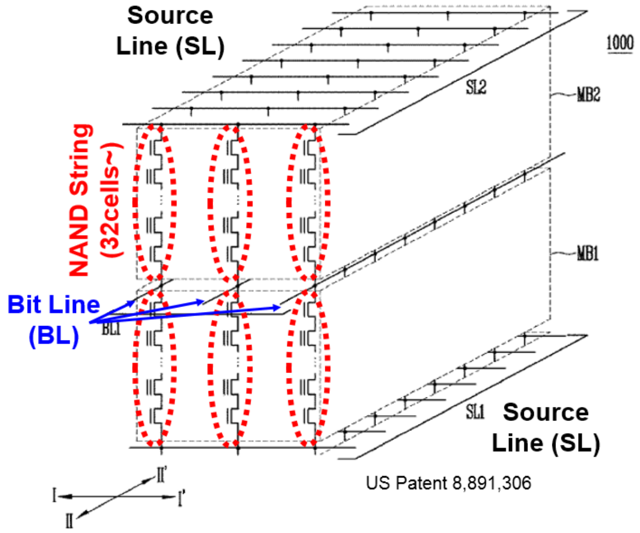
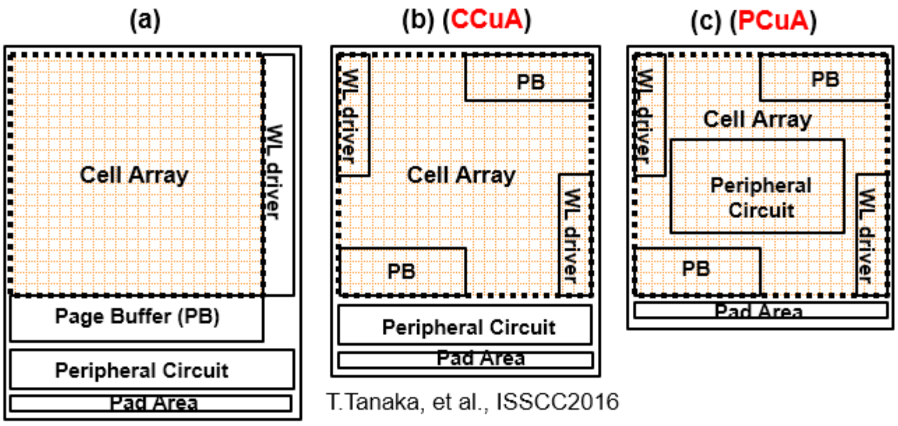


Figure 3.40 The stacked NAND String scheme

3.5.2 Peripheral Circuits under memory arrays

In the first 3D generations [68,69], peripheral circuits (charge pumps, logic, etc.) and core circuits (like Page Buffers and Row decoders) are located outside the memory matrix, like in a conventional 2D chip floorplan, as sketched in Figure 3.41(a). However, 3D memory cells are vertically stacked: in other words, memory transistors are not formed on the Si substrate; on the contrary, they are built around a deposited poly-Si (vertical pillar). Therefore, 3D architectures allow placing some circuits directly on the Si substrate under the memory array. Of course, this solution offers a significant reduction of the chip size.



T.Tanaka, et al., ISSCC2016

Figure 3.41 3D NAND Flash memory layout: (a) conventional, (b) CCuA, and (c) PCuA [70].

Figure 3.41(b) shows a layout of a Flash memory with *Core Circuits Under the Array* (CCuA) [70]; in addition, Figure 3.41(c) displays the case where both *Core* and *Peripheral Circuits* are manufactured on the Si substrate *under the Array* (PCuA) [66].

Figure 3.42 compares cell array efficiency of 2D, 3D in conventional layout, CCuA, and PCuA [66]. Efficiency of 2D and conventional 3D are between 60% and 81%. If CCuA is used, then the cell efficiency can be as high as 85%. In the extreme case, when both peripheral and core circuits sits under the memory matrix (PCuA), the cell efficiency can reach around 95%, because peripheral circuits usually occupy more than 10% of the whole chip.

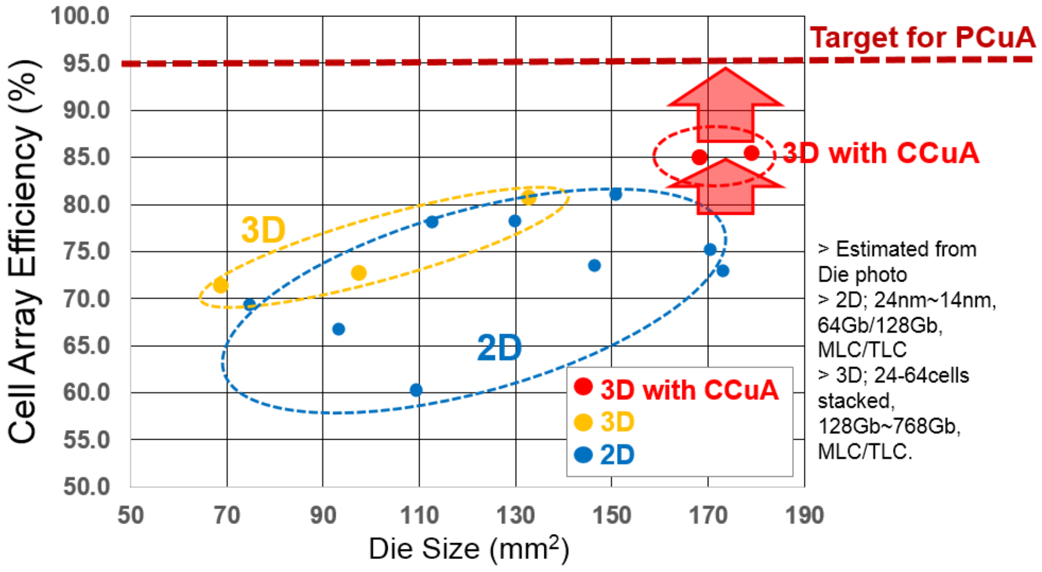


Figure 3.42 Cell Array Efficiency [66]

This big area saving doesn't come for free. The most important challenge is manufacturing low resistance metal layers under the array: this is absolutely critical for a reliable circuit functionality. Usually, metal layers used in 2D NAND flash memories are made of Cu. However, when circuits are under the array, the high temperature processes (i.e. > 800°C) that 3D requires can seriously degrade the resistance of metal layers. Therefore, circuits under the array require 3D "low" temperature fabrication processes.

3.5.3 Data Retention

3D CT cells and 2D FG cells are completely different in terms of data retention properties. Generally speaking, 2D SONOS (*Silicon-Oxide-Nitride-Oxide-Silicon*, which is one variant of CT) cells exhibit larger V_{TH} shifts than 2D FG cells: this is caused by a fast charge detrapping through the tunnel oxide [71]. Figure 3.43 compares data retention of two different cells: (a) 3D SONOS cell and (b) 2D 2y-nm FG cell [65]. Both cells have been cycled 3,000 times. After 3k cycles 3D SONOS has a V_{TH} distribution width narrower than 2D FG; however, after baking at *High Temperature* (HT), the V_{TH} distribution becomes wider, and it has a bigger V_{TH} shift. For 3D SONOS cells, data retention is definitely one of the hottest topics.

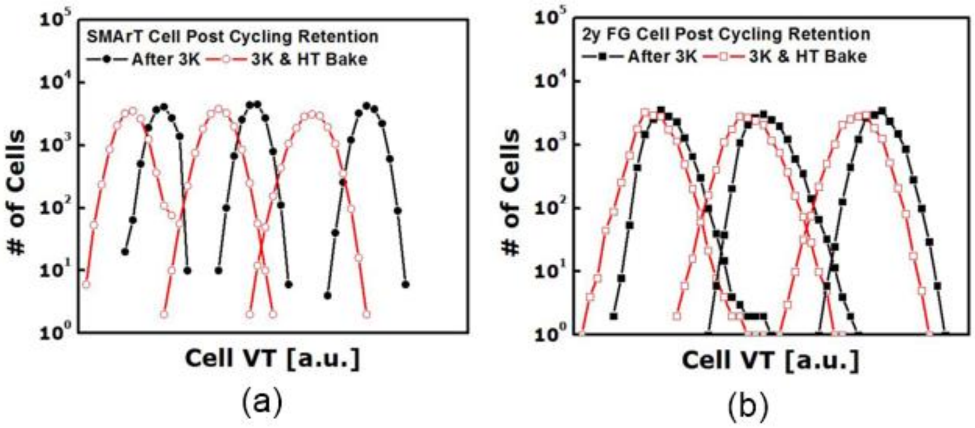


Figure 3.43 Vth distribution of cycled cells after high temperature retention for (a) 3D SMArT cell and (b) 2D 2y-nm FG cell [65].

Another important retention issue for 3D SONOS is the fact that the relationship between charge loss and temperature is different from 2D FG, as shown in Figure 3.44 [65], thus impacting the way accelerated tests should be performed. For 2D FG cells V_{TH} shift is linearly dependent upon the bake temperature, which says that the mechanism governing data loss remains constant. However, in 3D SONOS cell V_{TH} shift exhibits a non-linear relationship with respect to the bake temperature; in other words, the data loss mechanism changes from low to high temperature. The data loss mechanisms are dominated by band-to-band tunneling at low temperature and by thermal emission at high temperature [65]. As a consequence, simple temperature accelerated tests, which have been used for decades, should be used very carefully: retention below 90 °C has to be evaluated by extrapolating from data collected over at least 3 weeks at relatively low temperatures. It is worth highlighting that there multiple variations of CT cells; for example, BE-SONOS (*Bandgap Engineered*) can be used to optimize the bandgap structure of the SONOS cell [79].

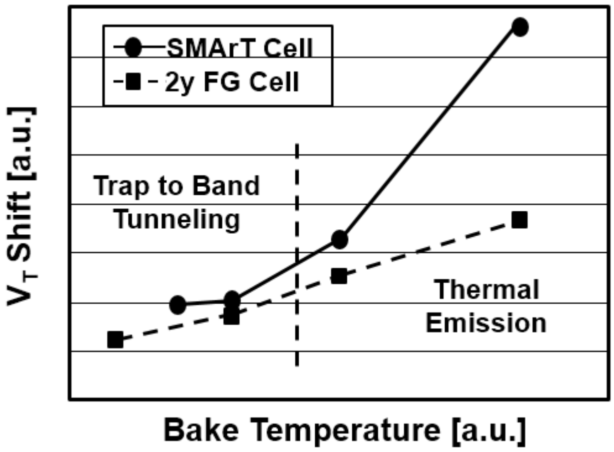
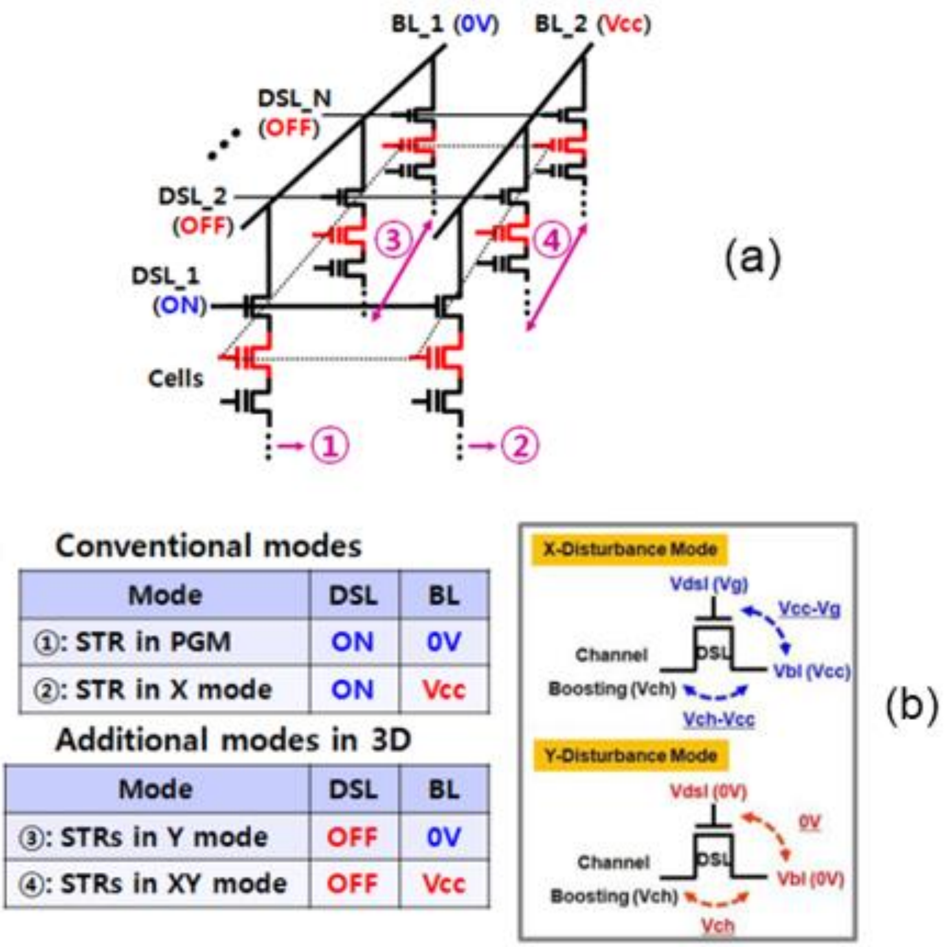


Figure 3.44 V_{TH} shift vs. Bake Temperature for 3D SONOS cells and 2D 2y-nm FG cells [65].

3.5.4 3D Program Disturb

Figure 3.45(a) shows one 3D NAND block [72][73]. In each block, N strings are connected to the same bitline by means of N select transistors, namely DSL_1 to DSL_N. In a 2D NAND block, there is a 1:1 correspondence between strings and bitlines. As a matter of fact, 3D architectures introduce new program disturb modes, as sketched in Figure 3.45(b).

When DSL_1 is activated, strings (STRs) along DSL_1 are either being programmed or they suffer “X” disturb, depending on the BL bias. When we look at “X” disturb, bitlines are biased at Vcc and there is no difference with respect to 2D NAND. But in 3D, DSL_2 to DSL_N are turned off. We can distinguish two different situations, which we call “Y” and “XY” program disturbs. In the “Y” case bitlines are biased at ground and drain select transistors (DSL) are off; for “XY” we have bitlines at Vcc and DSL off.



Conventional modes

Mode	DSL	BL
①: STR in PGM	ON	0V
②: STR in X mode	ON	Vcc

Additional modes in 3D

Mode	DSL	BL
③: STRs in Y mode	OFF	0V
④: STRs in XY mode	OFF	Vcc

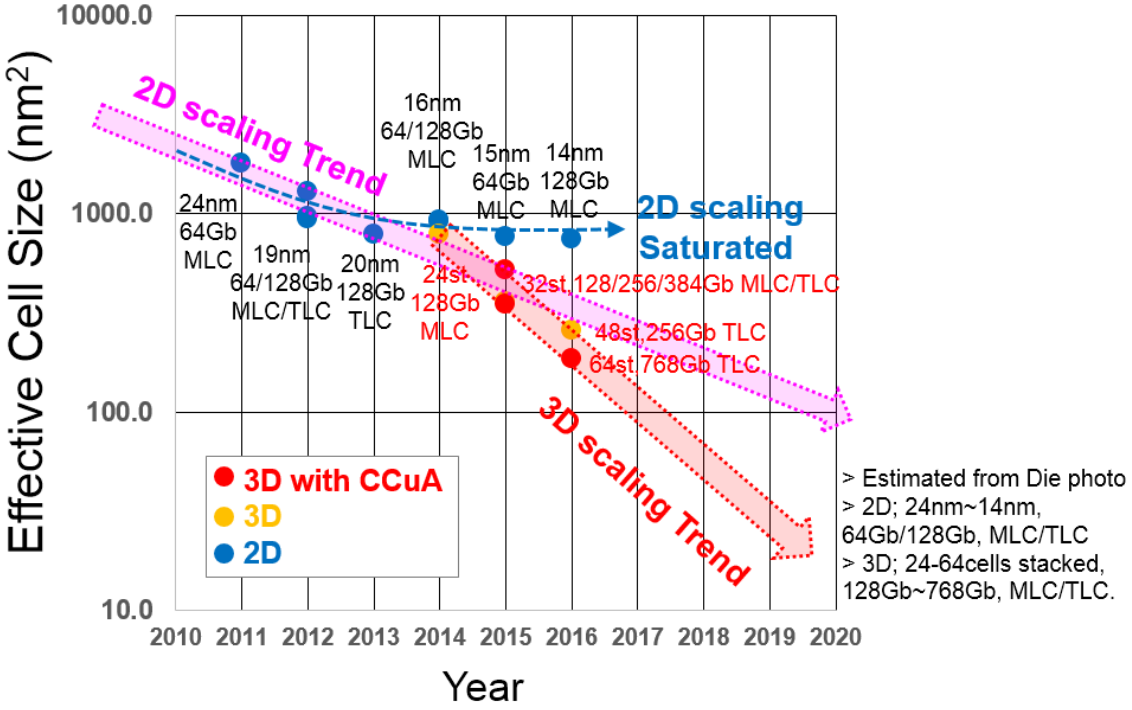
Figure 3.45 (a) Program disturb in a 3D NAND array. (b) 3D introduces two new program disturbs, Y and XY [73].

"XY" disturb mode is not severer than "X" mode. Being DSL off and BL at Vcc, the self-boosting voltage cannot cause a leakage current through DSL. On the contrary, in the "Y" mode BL is at ground, thus open the door to a possible leakage through DSL. In addition, DSL of 3D NAND shows a larger leakage current compared to 2D NAND [65][73]. Moreover, in 2D the leakage current through DSL is prevented by the fact that V_{TH} of DSL becomes higher during programming thanks to a strong body effect. This is not the case with 3D NAND. Several approaches to suppress the above mentioned leakage current have been proposed over time [72]. These include: (1) DSL with high V_{TH}, (2) DSL negative bias, and (3) dummy wordlines between DSL and edge wordlines. Dummy wordlines can reduce the voltage drop going from the self-boosting voltage to the voltage applied to the DSL; on top of that, they are helpful for inhibiting the hot carrier generation that might take place on the edge wordline (in practice, they reduce the lateral electric field). Indeed, dummy WLs have to be carefully designed (biasing, V_{TH}, number of wordlines) given all the above mentioned functions. A detailed analysis of 3D program disturb mechanism can be found in [73].

3.6 Future Trend for 3D NAND Flash

Figure 3.46 shows cell's size scaling trend, based on published die photographs. 2D became flat below 20 nm, while 3D cell showed a significant reduction going from 24 to 64 layers. This 3D scaling speed will continue by increasing the height of the memory stack, and exploiting technological innovations like Multi-stacked and Stacked NAND string, as shown in Section 3.6.1.

3D NAND arrays based on CT vertical channel were selected for volume production because the fabrication process is simpler than other 3D architectures. Volume production of 3D NAND Flash started in late 2013 with a 24 layer MLC (2 bit/cell) V-NAND [68][74]. Year after year, the number of stacked cells grew up, as shown in Figure 3.47 [7][69][75], thus reducing the cost per bit and fueling an even more pronounced diffusion of Solid State Drives. The scaling trend shows that a monolithic 1 Tera bit NAND Flash memory will be doable in the coming years.



> Estimated from Die photo
> 2D; 24nm~14nm, 64Gb/128Gb, MLC/TLC
> 3D; 24-64cells stacked, 128Gb~768Gb, MLC/TLC.

Figure 3.46 Effective cell size trend.

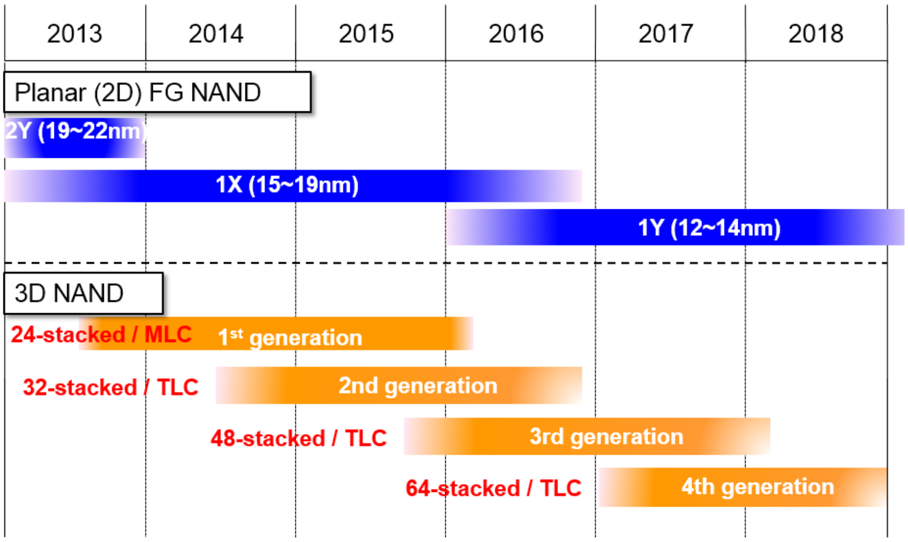


Figure 3.47 Transition from 2D NAND to 3D NAND [66].

In this chapter we have presented many architectural options for building a 3D NAND array, including some of the latest and greatest layout options, but the 3D evolution is just at the beginning. In fact, two fundamentally different technologies, Floating and Charge Trap, are fighting each other, trying to prove that they can win in the long run, i.e. when scaling will be pushed to the limit. Flash manufactures are already shooting for 100 vertical layers with multi-level capabilities, including 4 bit/cell. No doubt that we'll see a lot of innovations in the near future: engineers and scientists are called to give their best effort to make this vertical evolution happen.

Bibliography

[1] Masuoka, F.; Momodomi, M.; Iwata, Y.; Shirota, R.; "New ultra high density EPROM and flash EEPROM with NAND structure cell," Electron Devices Meeting, 1987 International, vol.33, pp. 552- 555.

[2] R. Micheloni, L. Crippa, A. Marelli, "Inside NAND Flash Memories", Chap. 6 (Springer, 2010).

[3] T. Mizuno et al., "Experimental study of threshold voltage fluctuation due to statistical variation of channel dopant number in MOSFET's." IEEE Trans. Electron Devices 41(11), 2216–2221, 1994.

[4] H. Kurata et al., "The impact of random telegraph signals on the scaling of multilevel flash memories, in Symposium on VLSI Technology", 2006.

[5] C.M. Compagnoni et al., "Ultimate accuracy for the NAND flash program algorithm due to the electron injection statistics." IEEE Trans. Electron Devices 55(10), 2695–2702 (2008)

[6] Seiichi Aritome, "NAND Flash Memory Technologies", IEEE Press Series on Microelectronics System, Wiley-IEEE Press, Published on December 2015

[7] S. Aritome, "3D Flash Memories", International Memory Workshop 2011 (IMW 2011), short course.

[8] R. Micheloni, L. Crippa, A. Marelli, "Inside NAND Flash Memories", Chap. 5 (Springer, 2010)

[9] S.M. Jung et al., “Three dimensionally stacked NAND flash memory technology using stacking single crystal Si layers on ILD and TANOS structure for beyond 30 nm node”, in IEDM Technical Digest (2006).

[10] E.K. Lai et al., “A multi-layer stackable thin-film transistor (TFT) NAND-type flash memory”, in IEDM Technical Digest (2006)

[11]http://www.samsung.com/us/business/oem-solutions/pdfs/VNAND_technology_WP.pdf. Samsung V-NAND technology, White Paper, Sept 2014

[12] R. Micheloni, L. Crippa, Chapter 3, “Multi-bit NAND flash memories for ultra high density storage devices”, in Advances in Non-volatile Memory and Storage Technology, ed. by Y. Nishi (Woodhead Publishing, Sawston, 2014)

[13] R. Micheloni et al., Chapter 7, “High-capacity NAND flash memories: XLC storage and single-die 3D”, in Memory Mass Storage, by G. Campardo et al. (Springer, 2011)

[14] H. Tanaka et al., “Bit cost scalable technology with punch and plug process for ultra high density flash memory”, in VLSI Symposium Technical Digest (2007), pp. 14–15

[15] Y. Fukuzumi et al., “Optimal integration and characteristics of vertical array devices for ultra-high density, bit-cost scalable flash memory”, in IEDM Technical. Digest (2007), pp. 449–452

[16] M. Ishiduki et al., “Optimal device structure for pipe-shaped BiCS flash memory for ultra high density storage device with excellent performance and reliability”, in IEDM Technical Digest (2009), pp. 625–628

[17] T. Maeda et al., “Multi-stacked 1G cell/layer pipe-shaped BiCS flash memory”, in Digest Symposium on VLSI Circuits, June 2009, pp. 22–23

[18] R. Katsumata et al., “Pipe-shaped BiCS flash memory with 16 stacked layers and multi-level-cell operation for ultra high density storage devices”, in 2009 Symposium on VLSI Technology (2009), pp. 136–137

[19] Y. Fukuzumi et al., “Optimal integration and characteristics of vertical array devices for ultra-high density, bit-cost scalable flash memory”, in IEDM Technical. Digest (2007), pp. 449–452

[20] H. Aochi, “BiCS flash as a future 3-D non-volatile memory technology for ultra high density storage devices”, in Proceedings of International Memory Workshop (2009), pp. 1–2

[21] Y. Yanagihara et al., “Control gate length, spacing and stacked layers number design for 3D-Stackable NAND flash memory2, in IEEE IMW (2012), pp. 84–87

[22] K. Takeuchi, “Scaling challenges of NAND flash memory and hybrid memory system with storage class memory and NAND flash memory”, in IEEE Custom Integrated Circuits Conference (CICC) (2013), pp. 1–6

[23] A. Nitayama et al., “Bit Cost Scalable (BiCS) flash technology for future ultra high density storage devices", in 2010 International Symposium on VLSI Technology Systems and Applications (VLSI TSA), Apr. 2010, pp. 130–131

[24] Y. Komori et al., “Disturbless flash memory due to high boost efficiency on BiCS structure and optimal memory film stack for ultra high density storage device”, in IEDM Technical Digest (2008), pp. 851–854

[25] M. Ishiduki et al., “Optimal device structure for pipe-shaped BiCS flash memory for ultra high density storage device with excellent performance and reliability”, in IEDM Technical Digest (2009), pp. 625–628

[26] T. Maeda et al., “Multi-stacked 1G cell/layer pipe-shaped BiCS flash memory”, in Digest Symposium on VLSI Circuits, June 2009, pp. 22–23

[27] R. Katsumata et al., Pipe-shaped BiCS flash memory with 16 stacked layers and multi-level-cell operation for ultra high density storage devices, in 2009 Symposium on VLSI Technology (2009), pp. 136–137

[28] J. Kim et al., “Novel 3-D structure for ultra high density flash memory with VRAT (vertical-recess-array-transistor) and PIPE (planarized integration on the same plane)”, in 2008 IEEE Symposium on VLSI Technology (2008)

[29] J. Kim et al., “Novel vertical-stacked-array-transistor (VSAT) for ultra-high-density and cost-effective NAND flash memory devices and SSD (solid state drive)”, in 2009 IEEE Symposium on VLSI Technology (2009)

[30] J. Jang et al., “Vertical cell array using TCAT (terabit cell array transistor) technology for ultra high density NAND flash memory”, in 2009 IEEE Symposium on VLSI Technology (2009)

[31] W. Cho et al., “Highly reliable vertical NAND technology with biconcave shaped storage layer and leakage controllable offset structure”, in 2010 Symposium on VLSI Technology (VLSIT) (2010), pp. 173–174

[32] J. Elliott, E.S. Jung, “Ushering in the 3D memory era with V-NAND”, in Proceedings of Flash Memory Summit, www.flashmemorysummit.com, Santa Clara, CA, Aug 2013

[33] K.-T. Park, “Three-dimensional 128 Gb MLC vertical NAND flash memory with 24-WL stacked layers and 50 MB/s high-speed programming”, in IEEE ISSCC, Digest Technical Papers, Feb 2014, pp. 334–335

[34] K.-T. Park, “Three-dimensional 128 Gb MLC vertical NAND flash memory with 24-WL stacked layers and 50 MB/s high-speed programming”. IEEE J. Solid-State Circ. 50(1), (2015)

[35] K.T. Park, “A world’s first product of three-dimensional vertical NAND flash memory and beyond”, in NVMTS, 27–29 Oct 2014

[36] E. Choi et al., “Device considerations for high density and highly reliable 3D NAND flash cell in near future”, in IEEE International Electron Devices Meeting (2012), pp. 211–214

[37] K. Shim et al., “Inherent issues and challenges of program disturbance of 3D NAND flash cell”, in IEEE International Memory Workshop (2012), pp. 95–98

[38] J.-W. Im, “128 Gb 3b/cell V-NAND flash memory with 1 Gb/s I/O rate”, in IEEE International Solid-State Circuits Conference, Feb 2015, pp. 130–131

[39] J.-W. Im, “128 Gb 3b/cell V-NAND flash memory with 1 Gb/s I/O rate”. J. Solid-State Circ. 51(1) (2016)

[40] D. Kang et al., “256 Gb 3b/Cell V-NAND flash memory with 48 stacked WL layers”, in IEEE International Solid-State Circuits Conference (ISSCC), Digest Technical Papers, Feb 2016, pp. 130–131

[41] T. Endoh et al., “Novel ultra high density flash memory with a stacked-surrounding gate transistor (S-SGT) structured cell.” IEDM Tech. Dig. pp. 33–36 (2001)

[42] T. Endoh et al., “Novel ultra high density flash memory with a stacked-surrounding gate transistor (S-SGT) structured cell.2 IEEE Trans. Electron Devices 50(4), 945–951 (2003)

[43] T. Endoh et al., “Floating channel type SGT flash memory.” in The 1999 Joint International Meeting, Hawaii, vol. 99–2, Abstract No. 1323, 17–22 Oct 1999

[44] M.S. Seo et al., “The 3-dimensional vertical FG nand flash memory cell arrays with the novel electrical S/D technique using the extended sidewall control gate (ESCG).” in Proceedings of IEEE International Memory Workshop (2010), pp. 1–4

[45] M.S. Seo et al., “3-D Vertical FG NAND flash memory with a novel electrical S/D technique using the extended sidewall control gate.” IEEE Trans. Electron Devices 58(9) (2011)

[46] S. Whang et al., "Novel 3-dimensional dual control gate with surrounding floating-gate (DC-SF) NAND flash cell for 1 Tb file storage application." in Proceedings of International Electron Devices Meeting (IEDM) (2010), pp. 668–671

[47] Y. Noh et al., "A new metal control gate last process (MCGL process) for high performance DC-SF (dual control gate with surrounding floating gate)" 3D NAND flash memory in Symposium on VLSI Technology (2012), pp. 19–20

[48] R. Micheloni, L. Crippa, "Multi-bit NAND flash memories for ultra high density storage devices (chapter 3)." in Y. Nishi (ed.) Advances in Non-volatile Memory and Storage Technology (Woodhead Publishing, 2014)

[49] R. Micheloni et al., "High-capacity NAND flash memories: XLC storage and single-die 3D" (chapter 7) in G. Campardo et al. (eds.) Memory Mass Storage (Springer, 2011)

[50] H. Yoo et al., "New read scheme of variable Vpass-read for dual control gate with surrounding floating gate (DC-SF) NAND flash cell." in Proceedings of 3rd IEEE International Memory Workshop (2011), pp. 1–4

[51] S. Aritome et al., "Advanced DC-SF cell technology for 3-D NAND flash" IEEE Trans. Electron Devices 60(4), 1327–1333 (2013)

[52] M.S. Seo et al., "A novel 3-D vertical FG nand flash memory cell arrays using the separated sidewall control gate (S-SCG) for highly reliable MLC operation" in Proceedings of 3rd IEEE International Memory Workshop (IMW) (2011), pp. 1–4

[53] M.S. Seo et al., "Novel concept of the three-dimensional vertical FG nand flash memory using the separated-sidewall control gate" IEEE Trans. Electron Devices 59(8), 2078–2084 (2012)

[54] K. Parat, C. Dennison, "A floating gate based 3D NAND technology with CMOS under array." in Conference on International Electron Devices Meeting (IEDM), San Francisco (USA), Dec 2015

[55] T. Tanaka et al., "A 768 Gb 3 b/cell 3D-floating-gate NAND flash memory." in 2016 IEEE International Solid-State Circuits Conference (ISSCC), Digest of Technical Papers, (San Francisco, USA, 2016), pp. 142–143

[56] R. Micheloni, L. Crippa, A. Marelli, "Inside NAND Flash Memories" (Springer, 2010)

[57] K. Parat, C. Dennison, "A floating gate based 3D NAND technology with CMOS under array," in IEDM, 7 Dec 2015

[58] Y. Komori et al., "Disturbless flash memory due to high boost efficiency on BiCS structure and optimal memory film stack for ultra high density storage device", in IEDM Technical Digest (2008), pp. 851–854

[59] R. Micheloni, A. Marelli, K. Eshghi, "Inside Solid State Drives (SSDs)" (Springer, 2013)

[60] K.-T. Park, "Three-dimensional 128 Gb MLC vertical NAND flash memory with 24-WL stacked layers and 50 MB/s high-speed programming," in IEEE ISSCC Digest Technical Papers, pp. 334–335, 2014

[61] K.-T. Park, "Three-dimensional 128 Gb MLC vertical NAND flash memory with 24-WL stacked layers and 50 MB/s high-speed programming." IEEE J. Solid-State Circ. 50(1) (2015)

[62] K.T. Park, "A World's First Product of Three-Dimensional Vertical NAND Flash Memory and Beyond," NVMTS 27–29 Oct 2014

[63] D.-H. Lee, "A new cell-type string select transistor in NAND flash memories for under 20 nm node," in 4th IEEE International Memory Workshop (IMW), Milan, May 2012, pp. 1–3

[64] J.-W. Im, "128 Gb 3b/cell V-NAND flash memory with 1 Gb/s I/O rate," in 2015 IEEE International Solid-State Circuits Conference (2015) pp. 130–131

[65] Eun-Seok Choi; Sung-Kye Park, "Device considerations for high density and highly reliable 3D NAND flash cell in near future," Electron Devices Meeting (IEDM), 2012 IEEE International, vol., no., pp.9.4.1-,9.4.4, 10-13 Dec. 2012.

[66] S. Aritome, "NAND Flash Memory Revolution," 2016 IEEE 8th International Memory Workshop (IMW), Paris, 2016, pp. 1-4.

[67] S. Aritome, US Patent 8,891,306.

[68] Ki-Tae Park et al., "Three-Dimensional 128 Gb MLC Vertical nand Flash Memory With 24-WL Stacked Layers and 50 MB/s High-Speed Programming," Solid-State Circuits, IEEE Journal of , vol.50, no.1, pp.204,213, Jan. 2015.

[69] Jae-woo Im, et al "A 128Gb 3b/cell V-NAND Flash Memory with 1Gb/s I/O rate," Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2015 IEEE International , vol., no., pp.,, 23-25 Feb. 2015.

[70] T. Tanaka et al., "7.7 A 768Gb 3b/cell 3D-floating-gate NAND flash memory," 2016 IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA, 2016, pp. 142-144.

[71] Chih-Ping Chen et al."Study of fast initial charge loss and its impact on the programmed states Vt distribution of charge-trapping NAND Flash," Electron Devices Meeting (IEDM), 2010 IEEE International , vol., no., pp.5.6.1,5.6.4, 6-8 Dec. 2010

[72] Keon-Soo Shim et al. "Inherent Issues and Challenges of Program Disturbance of 3D NAND Flash Cell," Memory Workshop (IMW), 2012 4th IEEE International, vol., no., pp.1,4, 20-23 May 2012

[73] HyunSeung Yoo et al."Modeling and optimization of the chip level program disturbance of 3D NAND Flash memory," Memory Workshop (IMW), 2013 5th IEEE International , vol., no., pp.147,150, 26-29 May 2013.

[74] Ki-Tae Park et al. "19.5 Three-dimensional 128Gb MLC vertical NAND Flash-memory with 24-WL stacked layers and 50MB/s high-speed programming," Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International, pp.334-335,Feb 9-13, 2014.

[75] S. Aritome, Joint Rump session in VLSI Symposium 2012, “Scaling challenges beyond 1Xnm DRAM and NAND Flash.”

[76] C. Kim et al. "A 512 Gb 3b/cell 64-Stacked WL 3D V-NAND Flash Memory", Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2017 IEEE International, pp.202-203, February, 2017.

[77] R. Yamashita et al. "A 512 Gb 3b/cell Flash Memory on 64-Word-Line-Layer BiCS Technology" Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2017 IEEE International, pp.196-197, February, 2017.

[78] H. T. Lue, T. H. Hsu et al, “A Highly Scalable 8-Layer 3D Vertical-Gate (VG) TFT NAND Flash Using Junction-Free Buried Channel BE-SONOS Device”, VLSI Symposia on Technology, 2010.

[79] H.-T. Lue, S.-Y. Wang, E.-K. Lai, K.-Y. Hsieh, R. Liu, C. Y. LuA, “BESONOS (Bandgap Engineered SONOS) NAND for Post-Floating Gate Era Flash Memory”, Symposium on VLSI Technology, 2007.

[80] Subirats et al. “Impact of discrete trapping in high pressure deuterium annealed and doped poly-Si channel 3D NAND macaroni” 2017 IEEE International Reliability Physics Symposium (IRPS)

[81] L. Breuil “Improvement of poly-Si channel vertical charge trapping NAND devices characteristics by high pressure D2/H2 annealing” 2016 IEEE 8th International Memory Workshop (IMW)

[82] E. Capogreco et al. “MOVPE In1-xGaxAs high mobility channel for 3-D NAND Memory” 2015 IEEE International Electron Devices Meeting (IEDM)

[83] J. G. Lisoni et al. “Laser Thermal Anneal of polysilicon channel to boost 3D memory performance”, 2014 Symposium on VLSI Technology (VLSI-Technology), Digest of Technical Papers

Chapter 4

Machine Learning applied to NAND Flash memories

NAND Flash memories are an ubiquitous storage media found in many applications like portable devices, smart-phones, and Solid State Drives (SSDs). Recently, as we have seen in the previous chapter, the scaling path of the NAND Flash technology evolved from a planar integration concept towards a three dimensional process (i.e., 3D NAND), in the attempt of breaking the 1Tb/in^2 storage density barrier [1] [2].

However, such a paradigm shift introduced substantial issues in the characterization activities of the memory reliability. The typical characterization flow of NAND Flash products must follow strict guidelines in order to measure the reliability metrics as a function of the memory lifetime [3]. Such experimental activity allows developing proper error correction and management solutions to cope with the many physical mechanisms that impacts the reliability [4,5]. Nevertheless, there is the need of a thorough exploration of all the possible operative parameters (e.g., programming voltages, timings, etc.) of a NAND Flash chip to identify the optimal set of parameters that minimizes the RBER. The number of parameter combinations that needs to be explored in different scenarios (i.e., cycling and temperature range) going from planar to 3D devices increased to a point that it is now almost impossible to find the best working conditions by simply using a ”judge-by-eye” approach. Moreover, the variety of Error Correction Codes (ECCs) strategies that can be adopted [6], together with the number of secondary correction mechanisms like the Read Retry or the V_T shift [7] further enlarged the problem space. To complete the picture, in dense architectures, like Triple Level Cell (TLC) NAND Flash, there is a large variability of the reliability figures within the same memory, which makes a single solution that fits all very hard to find.

In this context, the data clustering algorithms typical of the machine learning discipline could help. Previous works dealt with machine learning for NAND Flash architectures [8–10]. The goal of those works was to provide a predictive solution for memory reliability rather than considering memory optimization.

In this chapter we exploit a data clustering algorithm to find homogeneous areas in terms of endurance reliability within a TLC 3D NAND Flash product. By analyzing a large dataset, obtained by an extensive characterization campaign of different devices under different working conditions, we found that the clustering helps in identifying, through a semi-supervised learning approach, peculiar behaviors of different memory regions. The results of the learning process are exploited in the optimization of the ECC strategies to be adopted by the system in order to find the optimal code rate for a Low-Density Parity-Check (LDPC) code that balances memory reliability and implementation cost.

4.1 Data collection

The electrical characterization of 3D NAND Flash memory devices has been performed with the test equipment shown in Fig. 4.1. The system is an advanced version of that already presented in [11] and is composed by a state-of-the-art ASIC PCIe Gen3 NVMe memory controller used for SSDs [12] dealing with NAND Flash commands for accessing the devices, a DRAM buffer for temporary data storage, and a set of SO-DIMM sockets for 3D-NAND Flash interfacing. The board hosts up to 8 SO-DIMMs each one populated with 8 3D-NAND Flash chip. A single chip contains 8 memory dies. The supply voltages are provided by an external regulated power supply. The characterization system communicates through a PCIe interface with an x86-PC where the data are collected for post-processing with machine learning algorithms.

The data clustering process requires a large amount of data. Since the goal is to find a general clustering rule for 3D NAND Flash under different endurance experiments, we tested multiple memory devices mounted on different SO-DIMMs each one with a proper testing sequence (i.e., different temperature, cycling time, etc.). The memories under test are TLC 3D-NAND Flash whose structure is depicted in Fig. 4.2. A single memory block is composed by an arrangement of wordlines, bitlines, and layers.

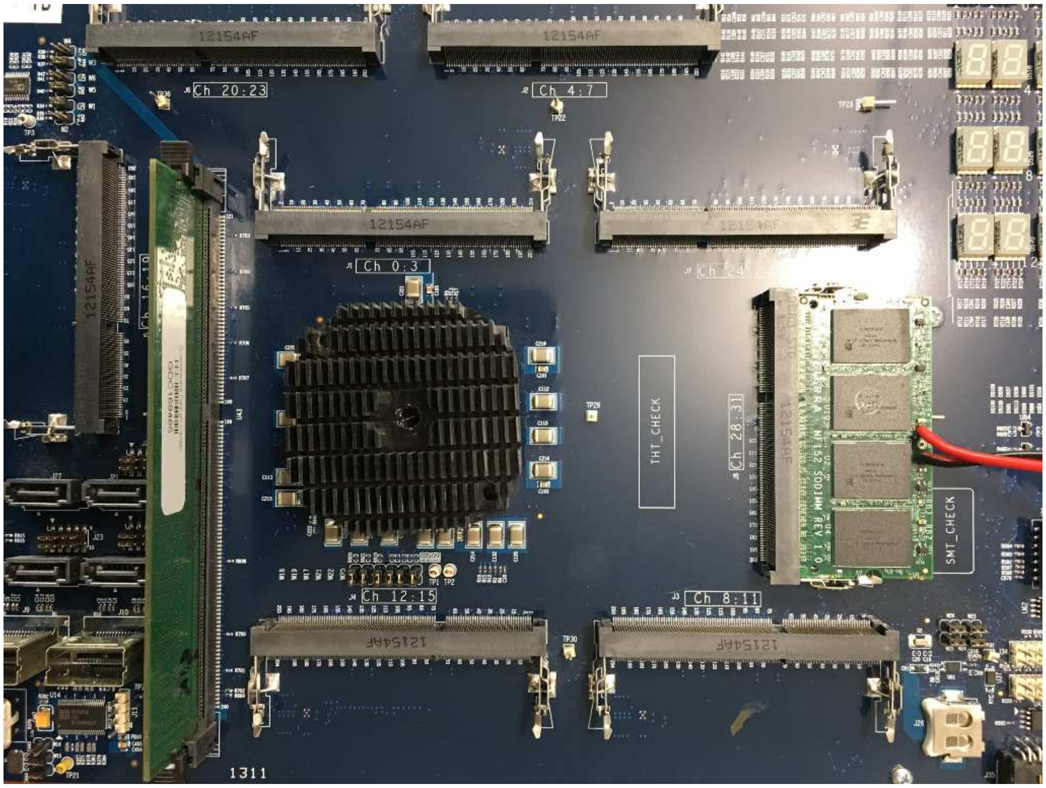


Figure 4.1: Test equipment used in the characterization of 3D NAND Flash memory devices.

The data analysis is performed on all the WL wordlines within a block considering lower, center, and upper page types. The size of a page is 16 kB divided to 4kB chunks, which are the minimum unit read during tests by the characterization system. Data clustering was performed on more than 800 different memory blocks belonging to different devices coming from multiple lots in order to improve the consistency of the classification algorithms that are presented in the next section. The overall data collection took several months to complete. The characterization data are produced by repeatedly writing and erasing (i.e., P/E cycle) the memory blocks with a random pattern and then reading out their contents to verify the number of erroneous bits. From the characterization standpoint, this activity is very important since it allows understanding the lifetime features of a memory while providing a starting point for the design of ECC strategies through the calculation of the Bit Error Rate (BER). In particular, since the system designers mostly tailor the ECC correction capability on the worst reliability case for the memory, we have evaluated the BER at the end of each endurance experiment, namely after 3k P/E cycles.

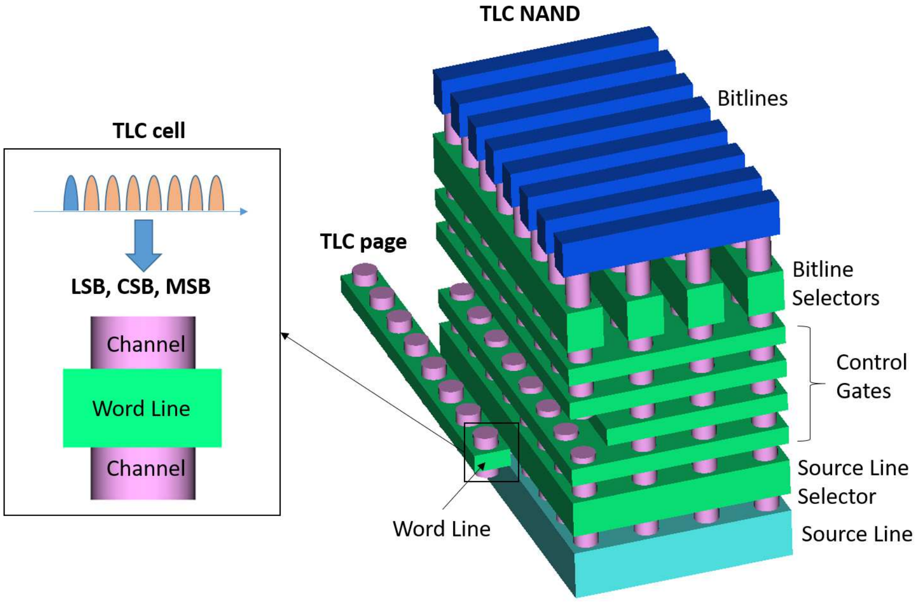


Figure 4.2: Architecture of a 3D NAND Flash memory showing the different topological elements (i.e., wordlines and bitlines on different layers) and TLC cell structure with its storage paradigm.

4.2 Data clustering results

The characterization data have been used as inputs to the well-known *k-means* data clustering algorithm [13]. The goal of the *k-means* is to partition the input data set (i.e., the BER of the different TLC pages in a 3D NAND Flash block) in k different clusters. The algorithm repeatedly computes the centroid of spherical clusters until each of the data points are assigned to a cluster. The number of clusters is specified *a priori* and depends on different factors like the complexity of the algorithm execution and the computational capabilities of the system. Fig. 4.3 shows an example of the *k-means* application on the input data set when k is equal to 5. As it can be seen from the figure, the clusters are not clearly separated, although the algorithm seems to identify specific regions of the Flash block that behave similarly in terms of BER. To ease the data analysis process we have separated lower, center, and upper TLC pages as their behavior is known to be different in terms of endurance reliability.

For a system designer, it becomes difficult to leverage the results of Fig. 4.3 because there is not a simple correlation between clusters and physical wordlines. As such, we have modified the *k-means* algorithm by applying a constrained clustering rule; this approach is also known as *”semi-supervised learning”*. The rule is that all the ECC chunks (or

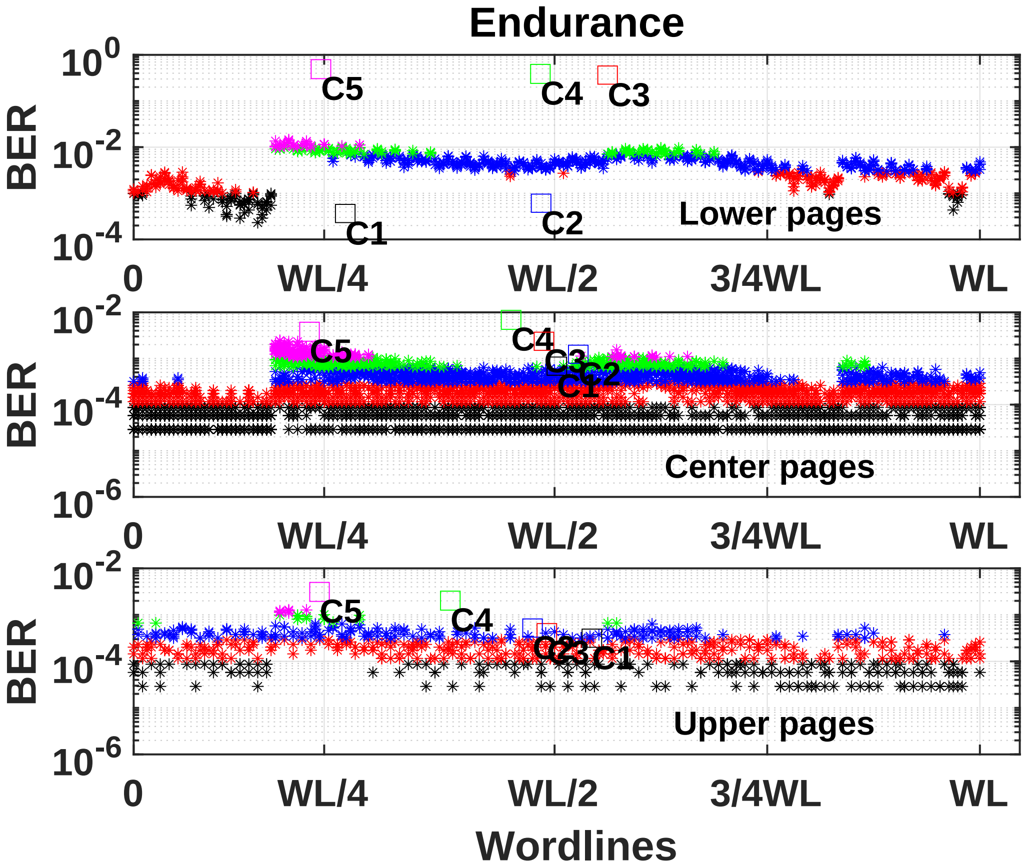


Figure 4.3: Data clustering performed through $k - means$ algorithm. The cluster centroids are indicated in the plot.

codewords) of a TLC page must belong to the same cluster. In addition to that, we require that the TLC pages are as contiguous as possible (in terms of logical address) and, in order to ease the firmware implementation, that lower, center, and upper pages share the same clusters. By analyzing the output of the constrained clustering (see Fig. 4.4) it is possible to appreciate that the clusters are more separated and easier to understand. The algorithm clustered the pages on the WL of a 3D NAND Flash in 6 different clusters, where a single cluster includes the three TLC page types. This result can be translated into 6 different reliability constraints for the 6 specific areas of a block; these constraints can be leveraged by both ECC design and advanced NAND Flash data management algorithms. We evaluated the consistency of clusters over different endurance testing conditions and we always found clusters similar to those represented in Fig. 4.4. Such a validation allows speculating that the cluster identification can be performed only once, at the end of the memory characterization.

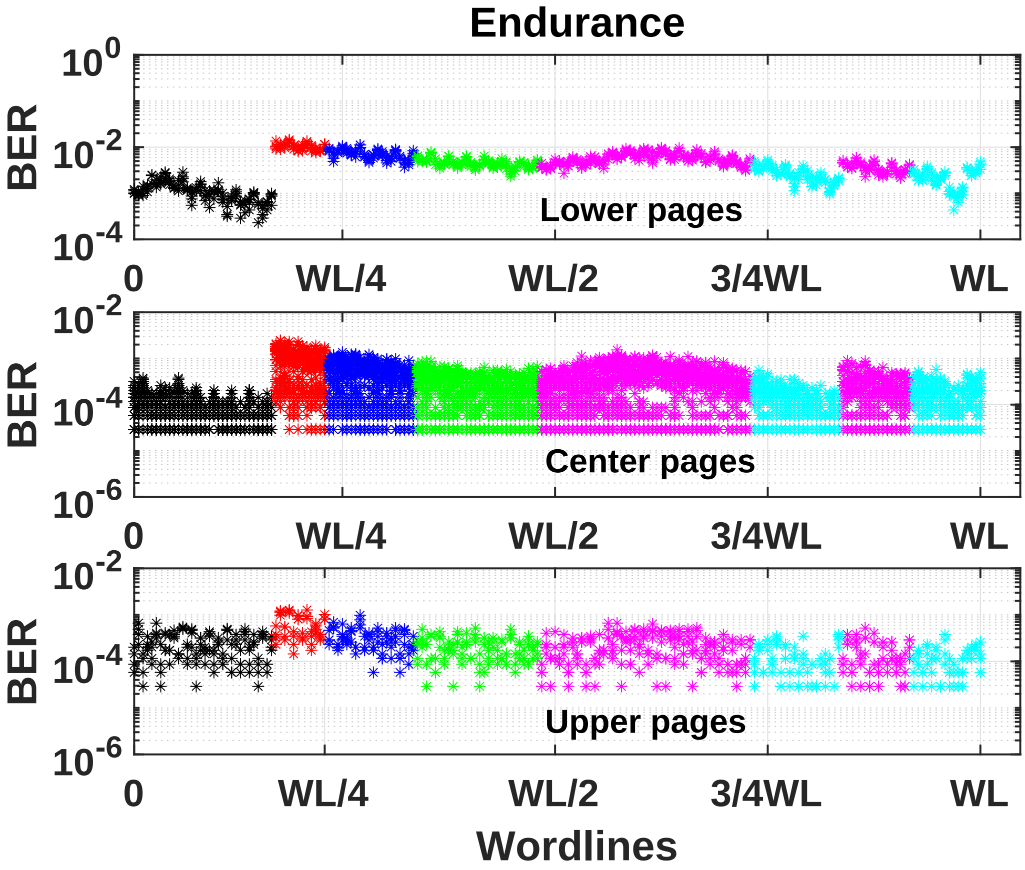


Figure 4.4: Constrained clustering results on 3D-NAND Flash. Six different clusters have been identified in the data-set.

Table 4.1: Worst case BER and associated LDPC Code Rate after data clustering

	Worst BER L	CR L	Worst BER C	CR C	Worst BER U	CR U
Cluster 1	2.8e-3	0.935	3.8e-4	0.97	6.7e-4	0.97
Cluster 2	1.5e-2	0.75	2.5e-3	0.935	1.3e-3	0.935
Cluster 3	1.2e-2	0.79	1.4e-3	0.935	9.9e-4	0.97
Cluster 4	7.8e-3	0.8475	8.7e-4	0.97	5.2e-4	0.97
Cluster 5	9.7e-3	0.81	1.5e-3	0.935	6.7e-4	0.97
Cluster 6	4.8e-3	0.9	5.8e-4	0.97	3.5e-4	0.97

4.3 LDPC Code Rate optimization results

LDPC codes are known as capacity approaching codes, in other words they are a category of codes that are able to reach a Frame Error Rate (FER) value close to the Shannon limit (for an infinite lenght of the ECC codeword) [14]. The FER is defined as the NAND Flash

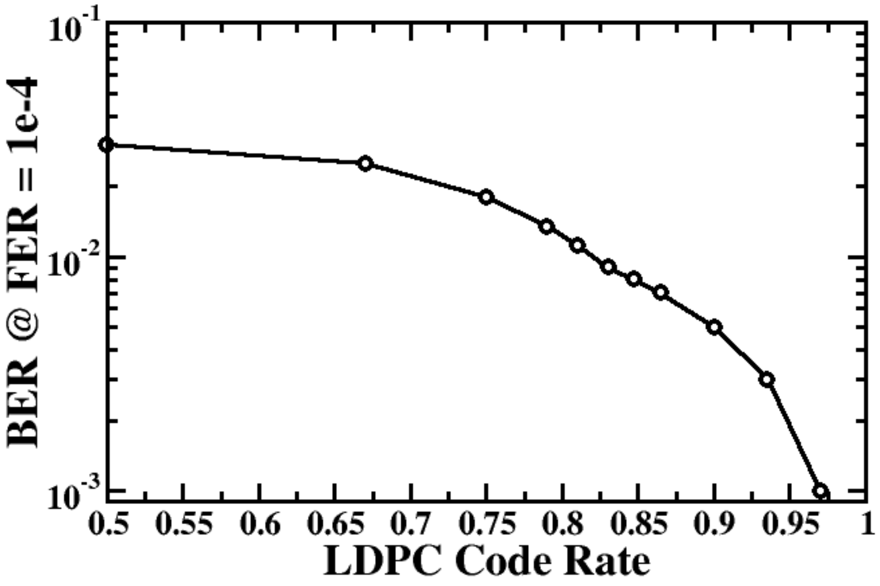


Figure 4.5: Correction capabilities of a generic LDPC code [14] using different code rates measured as a function of the sustainable BER at FER = 1e-4.

error rate after the application of the ECC; therefore, this metric quantifies the performance of a code.

LDPC are block linear codes defined with a very sparse parity check matrix H. To be a good candidate for 3D NAND Flash memories, LDPC codes must not only achieve excellent decoding performance, but they also need to be suitable for high-speed VLSI implementation with minimal silicon and energy cost. Among all the available options, it has been shown that quasi-cyclic (QC) LDPC codes are the most HW friendly. In a nutshell, the H matrix of a QC-LDPC code is an array of circulants. A circulant is a matrix in which each row is the cyclic shift of the row above it, and the first row is the cyclic shift of the last row. The parity check matrix H of a QC-LDPC code can be written as:

$$H = \begin{pmatrix} H_{1,1} & H_{1,2} & \cdots & H_{1,n} \\ H_{2,1} & H_{2,2} & \cdots & H_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ H_{m,1} & H_{m,2} & \cdots & H_{m,n} \end{pmatrix} \tag{4.1}$$

where each sub-matrix $H_{i,j}$ is a binary circulant. The LDPC encoder exploits the regular structure of the H matrix, so that it can be implemented in a modular way. For the same reason, the decoding (through belief propagation or an approximation of it) can be made in a layered way exploiting the modularity of the circulants [14].

Data storage systems such as Flash memories and hard disk drives typically demand very high code rates (e.g., 0.89 and higher). The code rate is the ratio between the user data and the total stored codeword (user data plus parity bits). A high code rate means a small overhead in terms of parity bits but less correction capability, while a low code rate means good correction capability but large overhead in terms of parity bits and, therefore, a higher waste of memory user capacity, which translates into higher costs of the system (see Fig. 4.5). The LDPC modular structure described above makes LDPC ideal for adaptive code rate implementations [15]. In particular, for 3D NAND Flash where BER changes during the device lifetime are strongly affected by the TLC page topology, it is key to have an ECC that has the ability to change the code rate without changing all the hardware underneath.

The LDPC code rate optimization in 3D NAND Flash can be performed by evaluating the worst case BER for each identified cluster along with specific ECC design considerations. In NAND Flash system design, it is important to guarantee that the ECC will be able to sustain a defined FER at the specified endurance conditions. Table 4.1 shows the measured worst case BER after 3k P/E per identified cluster and the correspondent LDPC code rate requested to sustain at least a FER = 1e-4 [16].

The code rate chosen by the designer, when no data clustering is performed and no diversification of lower, center, and upper page is present, should consider the worst BER of the whole device. In our experiments, this value is equal to 0.75. Such a code rate is extremely conservative and forces user space waste for parity purpose in locations where the reliability is higher. By splitting the three different TLC pages we gain in terms of code rate since the lower pages will be associated with the lowest code rate, while center and upper pages can be associated with 0.935, respectively. By calculating the equivalent code rate as the average of the page associated rates we obtain 0.87. This materializes in a 16% gain of memory space that can be allocated for user data. However, the best code rate optimization is achieved in combination with data clustering is used. When the TLC page types are split and grouped in clusters, the equivalent code rate can be computed as follows:

$$\frac{\sum_{i\in[L,C,U]}\sum_{j=1}^{6} n_j * CR_{i,j}}{3 * WL} \tag{4.2}$$

where n_j is the dimension of the $j-th$ cluster (i.e., the number of pages it contains), and $CR_{i,j}$ is the associated code rate for the $i-th$ page type of the $j-th$ cluster. In this case we obtained an equivalent code rate equal to 0.93, that allows a 24% gain on the memory user addressable space.

In this chapter we have demonstrated the benefits of machine learning in 3D NAND Flash characterization through the application of data clustering algorithms. The characterization data set has been obtained by an extensive testing campaign of 3D-NAND Flash devices under different operating conditions. By developing a semi-supervised learning methodology we have been able to optimize the LDPC code rate dedicated to ECC, resulting in a 24% gain of the memory space addressable by the user. Such activity paves the way for further applications in the memory characterization context.

Bibliography

[1] R. Micheloni, L. Crippa, and A. Marelli, *Inside NAND Flash memories*. Springer-Verlag, 2010.

[2] W. Jeong, J. w. Im, D. H. Kim, S. W. Nam, D. K. Shim, M. H. Choi, H. J. Yoon, D. H. Kim, Y. S. Kim, H. W. Park, D. H. Kwak, S. W. Park, S. M. Yoon, W. G. Hahn, J. H. Ryu, S. W. Shim, K. T. Kang, J. D. Ihm, I. M. Kim, D. S. Lee, J. H. Cho, M. S. Kim, J. H. Jang, S. W. Hwang, D. S. Byeon, H. J. Yang, K. Park, K. H. Kyung, and J. H. Choi, "A 128 Gb 3b/cell V-NAND Flash Memory With 1 Gb/s I/O Rate," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 1, pp. 204–212, Jan. 2016.

[3] JEDEC, "JESD22-A117 document," Oct. 2011.

[4] Y. Park, J. Lee, S. S. Cho, G. Jin, and E. Jung, "Scaling and reliability of nand flash devices," in *IEEE International Reliability Physics Symposium (IRPS)*, 2014, pp. 2E.1.1–2E.1.4.

[5] E. Vatajelu, H. Aziza, and C. Zambelli, "Nonvolatile memories: Present and future challenges," in *9th International Design Test Symposium (IDT)*, 2014, pp. 61–66.

[6] L. Zuolo, C. Zambelli, P. Olivo, R. Micheloni, and A. Marelli, "LDPC Soft Decoding with Reduced Power and Latency in 1X-2X NAND Flash-Based Solid State Drives," in *IEEE International Memory Workshop (IMW)*, May 2015, pp. 1–4.

[7] J. Yang, "High-Efficiency SSD for Reliable Data Storage Systems," in *Proc. Flash Memory Summit*, 2012.

[8] D. Hogan, T. Arbuckle, and C. Ryan, "Estimating MLC NAND Flash Endurance: A Genetic Programming Based Symbolic Regression Application," in *Proc. Conf. on Genetic and Evolutionary Computation*, 2013, pp. 1285–1292.

[9] T. Arbuckle, D. Hogan, and C. Ryan, "Learning Predictors for Flash Memory Endurance: A Comparative Study of Alternative Classification Methods," *Int. J. Comput. Intell. Stud.*, vol. 3, no. 1, pp. 18–39, 2014.

[10] Y. Nakamura, T. Iwasaki, and K. Takeuchi, "Machine learning-based proactive data retention error screening in 1Xnm TLC NAND flash," in *International Reliability Physics Symposium (IRPS)*, Apr. 2016, pp. PR–3–1–PR–3–4.

[11] C. Zambelli, P. King, P. Olivo, L. Crippa, and R. Micheloni, "Power-supply impact on the reliability of mid-1X TLC NAND flash memories," in *International Reliability Physics Symposium (IRPS)*, Apr. 2016, pp. 2B–3–1–2B–3–6.

[12] Microsemi Corp., "Flashtec nvme controllers," [Online] - Available: http://www.microsemi.com/products/storage/flashtec-nvme-controllers/flashtec-nvme-controllers, 2017.

[13] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. of Berkeley Symp. on Math. Statistics and Prob.*

[14] A. Marelli and R. Micheloni, *BCH and LDPC Error Correction Codes for NAND Flash Memories*. Springer Netherlands, 2016, pp. 281–320.

[15] X. Hu, "LDPC Codes for Flash Channels," in *Proc. Flash Memory Summit*, 2012.

[16] A. I. V. Casado, W. Y. Weng, S. Valle, and R. D. Wesel, "Multiple-rate low-density parity-check codes with constant blocklength," *IEEE Transactions on Communications*, vol. 57, no. 1, pp. 75–83, 2009.

Chapter 5

Impact of power-supply on the reliability of TLC NAND Flash memories

NAND Flash memories are now an ubiquitous storage medium commonly integrated in mobile, embedded and solid-state disks (SSDs) solutions [1]. The storage density scaling requires a continuous effort because of the reliability degradation induced by several physical effects (Chapter 2 and Chapter 3) [2–7]. Data management algorithms and strong error correction codes try to mitigate those effects [8–10], whereas little or no importance is given to other sources of reliability-loss and this is what we address in the following. A NAND Flash is composed by several macro blocks: the memory array, the data path circuitry that controls the input/output towards the external world, the decoders which select individual groups of cells in the array, and the high-voltage (HV) circuitry for read/write/erase operations. This latter sub-system plays an important role on the reliability since its design affects sensitive analog circuits that control the behavior of the memory cells during read and write operations. This is achieved by using a large set of voltages provided to the memory with a defined precision, timing and granularity. On top of that, many voltages have a value greater than the NAND power supply, asking for an on-chip charge pump. Fig. 5.1 shows the main circuits inside the HV domain of a NAND Flash: the charge pump, the oscillator, the voltage regulators, and the wordline (WL) switch. All these circuits share the same power supply source, namely V_{CC}, which Flash vendors advice in a working range usually between 2.7 V and 3.6 V [11]. However, even if V_{CC} is in such a safe operative region, a different behavior of the memory reliability during its entire lifetime can be observed depending on the chosen power supply.

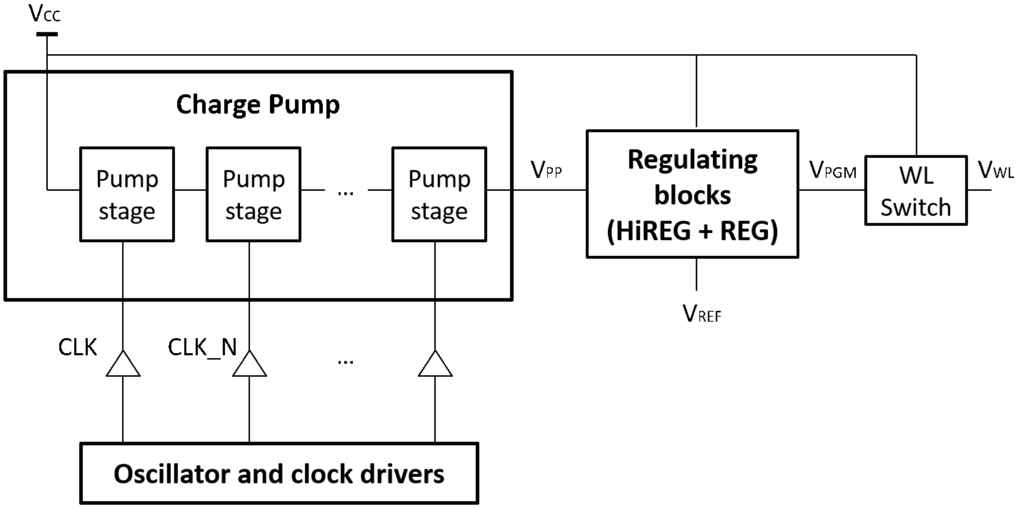


Figure 5.1: Typical architecture of the HV circuitry in a NAND Flash memory.

In this work we evaluate the impact of the supply voltage on the number of errors produced in write operations during endurance stress [12] performed on mid-1X Triple-Level Cell (TLC) NAND Flash samples. The experimental characterization of this technology, with a dedicated test environment, is benchmarked with SPICE simulations of the NAND Flash HV circuitry and of the memory cells to expose the culprits of the different reliability figures produced during endurance tests. This work will help system designers to understand how much they can leverage on the power supply to reduce the Flash power consumption while trading this feature with reliability.

5.1 Data collection

The analyses have been performed by means of a dedicated NAND Flash memory characterization system which collects the number of errors per page retrieved after a readout operation on an entire memory block. Fig. 5.2 shows the test equipment at a glance. It is composed by a programmable FPGA, a DRAM buffer for temporary data storage, an interface with a x86-PC, and a dedicated socket for NAND Flash memory interfacing. The power supply can be changed externally through a regulated power-supply unit. In this work we have considered two typical values used in NAND Flash design for SSD applications: 2.7 V, which is close to specification boundaries provided by the memory vendor, and an intermediate value like 3 V. The former value is generally used for power consumption lowering and to enable performance/reliability trade-offs [13].

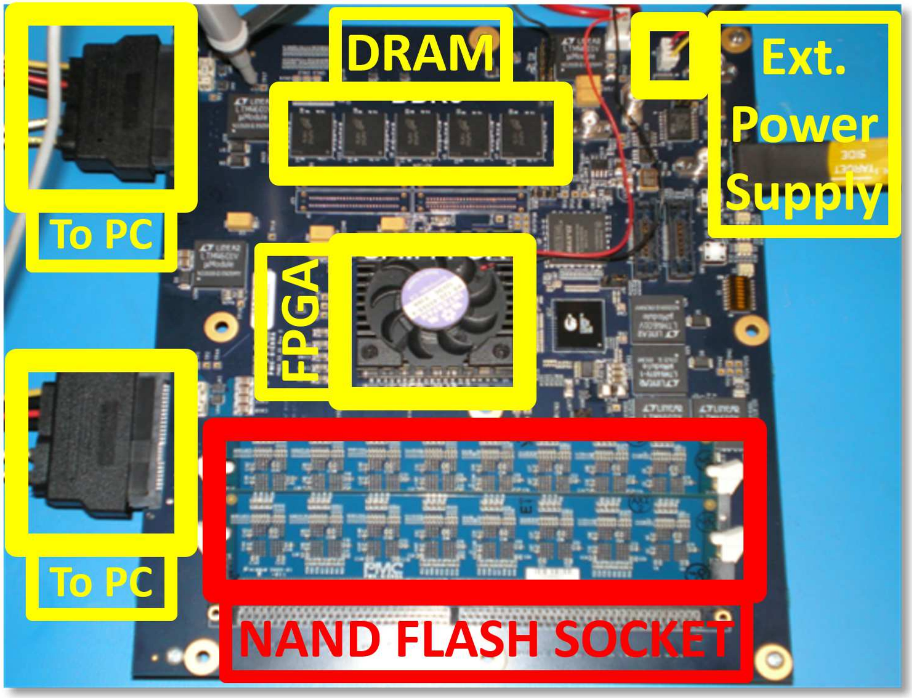


Figure 5.2: Test equipment used in NAND Flash characterization

Each tested device was a mid-1X TLC NAND Flash that has been stressed with random data patterns to emulate real memory cycling. The tests followed the guidelines provided in [12]. All TLC page types (i.e., lower, middle, and upper pages) have been considered in the analysis to help the data interpretation. The page size of the tested NAND Flash memories is 16 KB.

In TLC NAND Flash it is possible to store three different bits on a single cell by controlling the placement of the cells threshold voltage with the Three Steps Programming (TSP) algorithm [14], that results into a separation of the cells in 8 different voltage distributions, each representing three stored bits being the lower, middle, or upper page content during reading (see Fig. 5.3). The total width of the threshold voltage distributions ΔV can be approximately expressed as:

$$\Delta V \approx \Delta V_{PGM} + R_{CCC} + N \tag{5.1}$$

where ΔV_{PGM} is the used step size increment in the staircase programming phases of the TSP [15], R_{CCC} is the residual cell-to-cell coupling [16], and N is the contribution of the noise sources during programming [14]. To reduce the number of bit errors it is required a tight control of ΔV, which is achieved by an accurate voltage control exerted by the HV

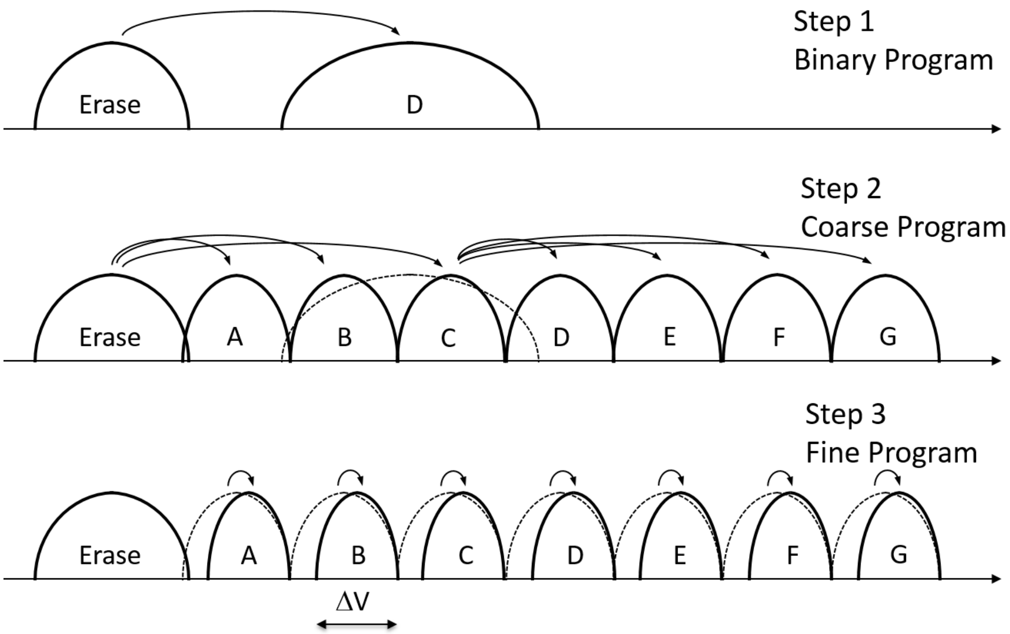


Figure 5.3: Phases of the TSP algorithm: binary program, coarse program, and fine program.

circuitry on the program step ΔV_{PGM}, being this latter factor the dominant term in eq.(5.1).

By monitoring different pages of a NAND Flash block during the memory endurance, measured in terms of sustained Program/Erase (P/E) cycles, it is possible to observe that the number of errors increases with endurance stress as expected. Fig. 5.4 shows the cumulative bit errors distribution for a lower page, exhibiting a clear trend in the median distribution value. Similar results are obtained for middle and upper pages except for the bit errors magnitude.

However, by investigating the number of errors measured at each readout cycle of the endurance test it is observed a strong dependence from the power supply voltage. Fig. 5.5 shows that using lower V_{CC} values generally yields to a higher number of errors and to a higher page-to-page variability. Moreover, it is observed that lower pages are usually more affected by the power supply dependence with respect to middle and upper pages. Errors page-to-page variability is analyzed through the σ parameter and the coefficient of variation of the Gaussian fitting applied to the errors distributions in lower, middle, and upper pages. Although at the memory rated endurance (usually from 300 to 500 P/E cycles in TLC NAND Flash) the variability is barely perceivable, as soon as the lifetime increases beyond this limit there is a maximum difference in the σ coefficient for the lower pages'

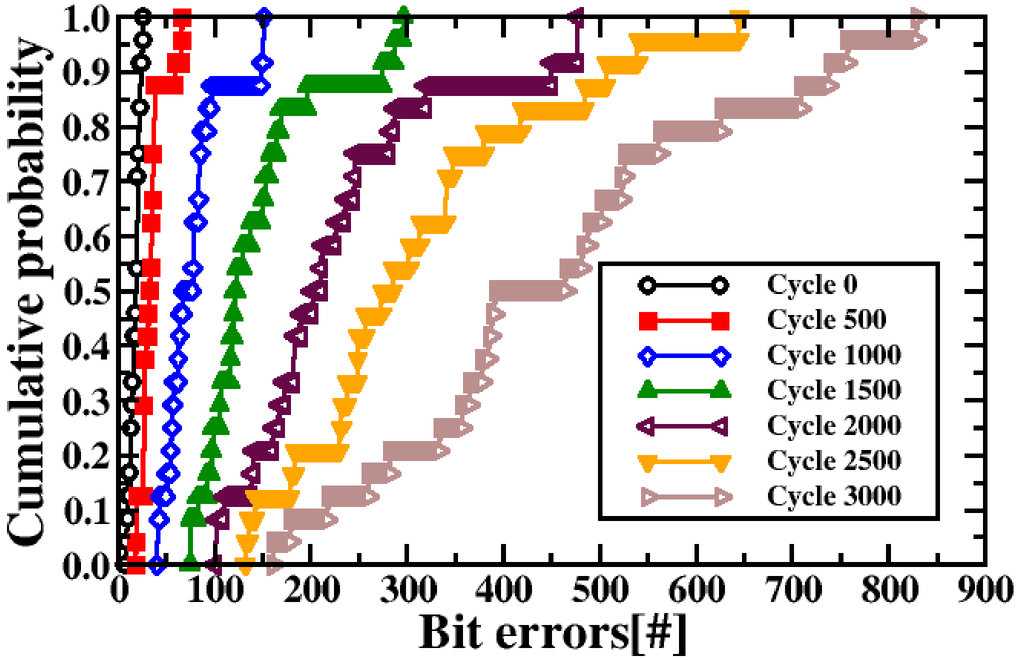


Figure 5.4: Evolution of the cumulative bit errors distributions at each readout cycle during endurance stress. The data are shown for lower pages with V_{CC} = 2.7 V.

errors up to 5.2 (see Fig. 5.6) by operating the memory either with a V_{CC} equal to 2.7 V or 3 V. Similar considerations can be derived by analyzing the coefficient of dispersion. These results indicates that the HV circuitry becomes less effective in controlling the threshold voltage distributions, especially for lower pages, both as a function of P/E cycles and V_{CC}.

5.2 Simulations of high voltage circuits

The previous experimental results call for an investigation of the power supply dependence of RBER in NAND Flash technology. To understand this phenomenon we have initially simulated the circuits in the HV blocks responsible for the sole generation and the control of the programming voltage V_{PGM} [1, 17]: the charge pumps in the pump stages and the voltage regulators to achieve a defined ΔV_{PGM} (see Fig. 5.7). All the simulations have been performed using SPICE with a HV technology process library typical for NAND Flash applications (i.e., gate length equal to 0.35 μm) for the transistors and the capacitors in the HV domain.

In the NAND Flash HV environment, one of the most used type of charge pumps in the pumping stages is the voltage doubler [18]. The basic circuit is shown in Fig. 5.8.

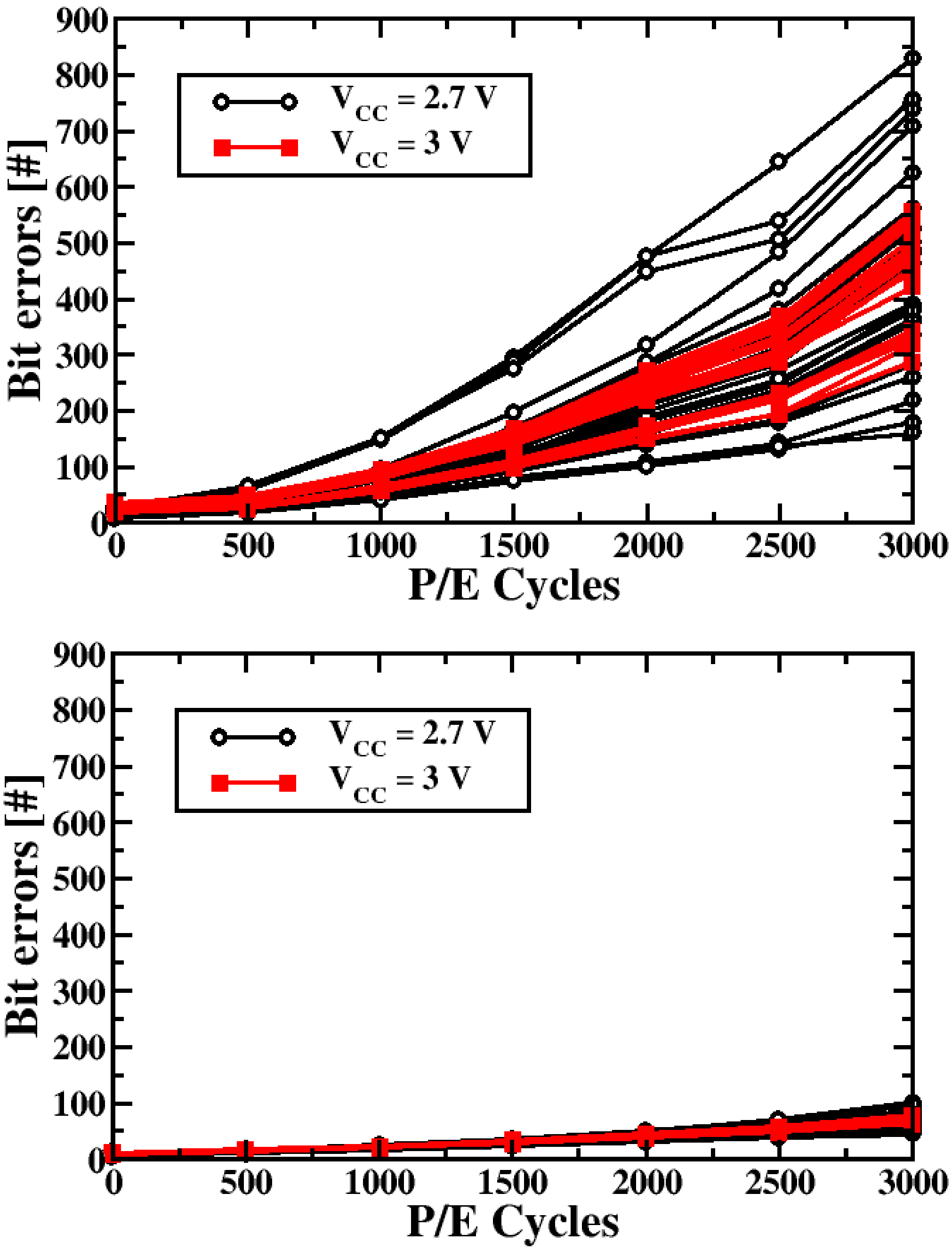


Figure 5.5: Errors retrieved during the endurance stress performed on lower pages (top) and middle/upper pages (bottom) in a block of a mid-1X TLC NAND Flash.

It is a feedback system that can boost the input voltage and, essentially, it is made up by two n-channel transistors (MN1, MN2), two p-channel transistors (MP1, MP2) and two capacitors (C1, C2) of the same size. Since NAND Flash requires, during the TSP algorithm, a range of voltages spanning from 12 V up to 23 V, multiple pumping stages

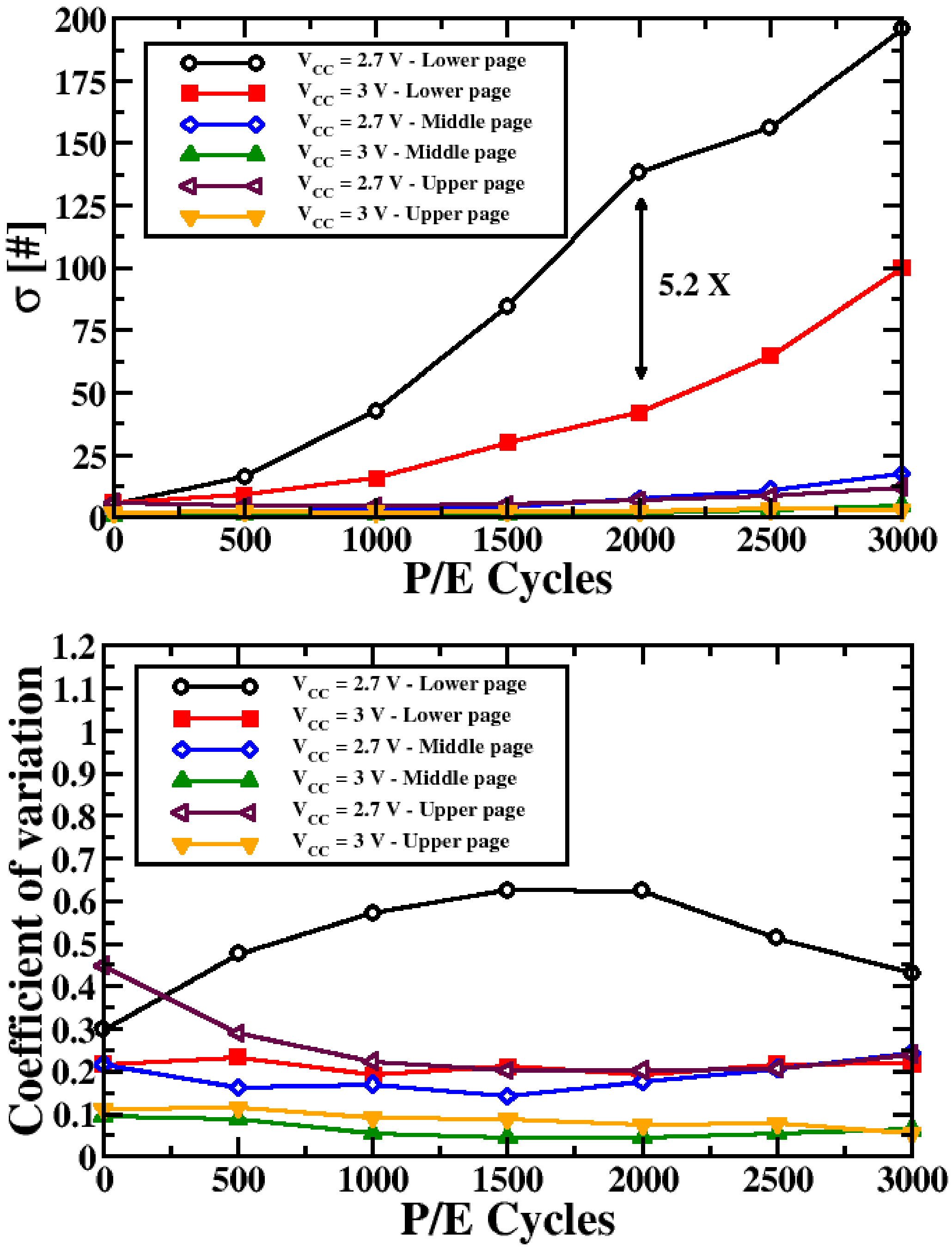


Figure 5.6: Variability (σ) and coefficient of dispersion of the errors distribution extracted at each readout cycle as a function of the page type and of the power-supply voltage.

driven by complementary clocks are required [19]. The output voltage V_{PP} of a N stages charge pump module can be expressed as:

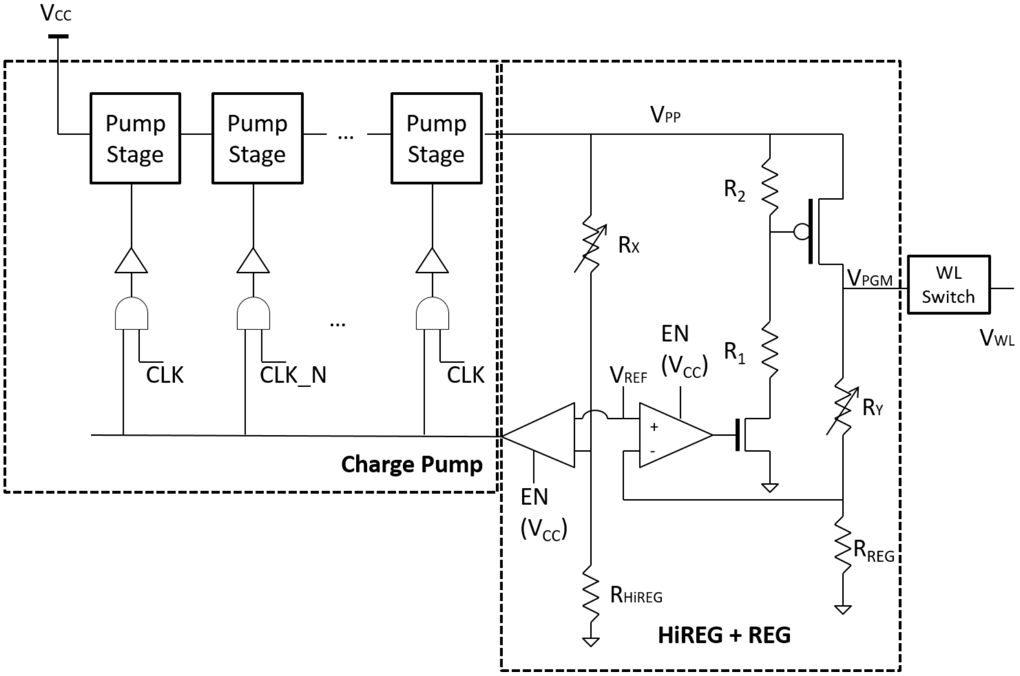


Figure 5.7: Simulated circuits in the HV blocks of a mid-1X TLC NAND Flash.

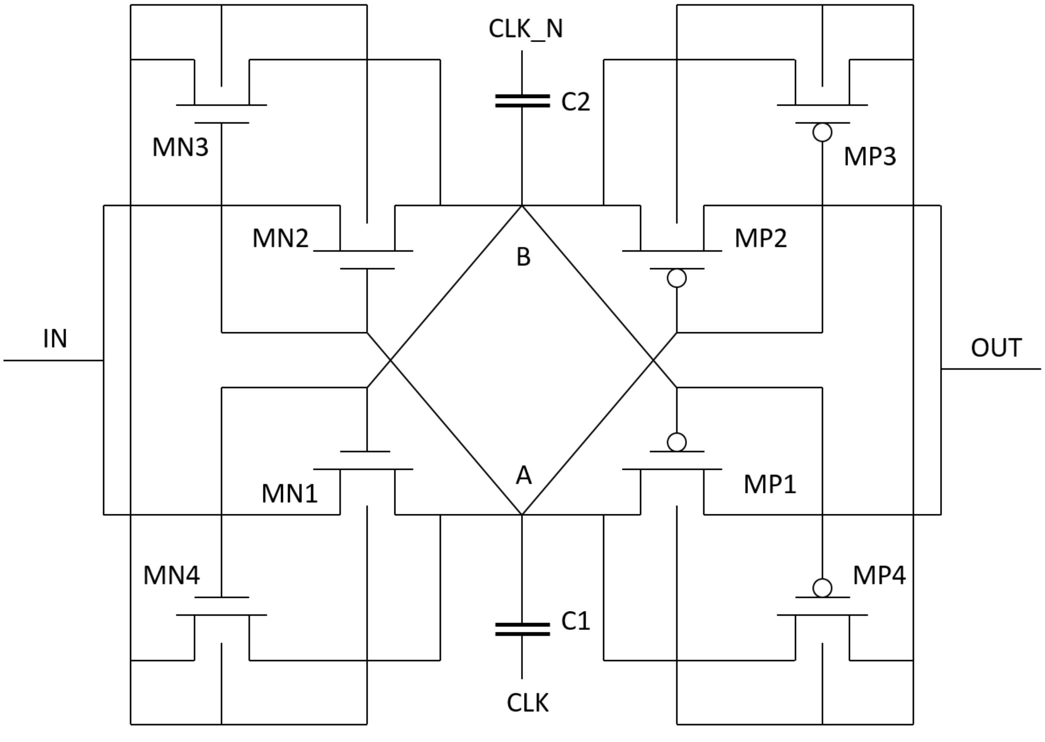


Figure 5.8: Voltage doubler circuit used in the charge pump unit stage of a mid-1X TLC NAND Flash.

$$V_{PP} = (N+1)V_g - \frac{NI_{OUT}}{f(C+C_s)} \tag{5.2}$$

where V_g is the maximum gain voltage per stage, I_{OUT} is the output current provided by the charge pump, f is the driving clock frequency, C is the equivalent stage capacitance, and C_s is the parasitic capacitance per stage, respectively. The output of the charge pump must be regulated to avoid disturbs and spurious glitches due to the parasitic capacitances through a high voltage regulator (HiREG). According to the schematics in Fig. 5.7:

$$V_{PP} = V_{REF}\left(1 + \frac{R_X}{R_{HiREG}}\right) \tag{5.3}$$

where V_{REF} is a reference low voltage used for the comparator, R_x is a programmable resistor to vary the output of the charge pump during the TSP algorithm, and R_{HiREG} is a constant resistor. The choice of a programmable resistor in the HiREG stage is mandatory in mid-1X TLC NAND Flash to reduce the power consumption of the programming algorithm while ensuring a good reliability of the HV transistors in this block that could prematurely enter into a breakdown condition. The linear low-dropout regulator (REG) generates the program voltage V_{PGM} to be applied to the memory cells selected for programming. The staircase program voltage required in the TSP phases is obtained by increasing the resistance R_Y (i.e., the value assumed at the first programming step) by a fixed amount of a programmable resistance ΔR_Y. The determined voltage step ΔV_{PGM} is equal to:

$$\Delta V_{PGM} = V_{REF}\frac{\Delta R_Y}{R_{REG}} \tag{5.4}$$

where R_{REG} is a constant resistance used for the voltage divider in the feedback path.

To ensure high reliability it is requested a high linearity of the V_{PGM} characteristics as a function of time, yielding to a constant ΔV_{PGM} during all the algorithm steps. SPICE simulations have been performed by considering the entire V_{PGM} range used in the TSP algorithm with a desired ΔV_{PGM} = 200 mV. The results evidenced in Fig. 5.9 show that by lowering the V_{CC} from 3 V to 2.7 V leads to a broadening of the ΔV_{PGM} distribution. This will reflect on a broadening of the cells threshold voltage distributions exposing the memory to a higher count of bit errors. The failure in complying with the staircase linearity for lower V_{CC} values can be explained as a sum of two problems occurring in the HiREG and REG blocks in the HV circuitry [20]: *i)* the regulators output voltage overshoots the target V_{PGM} due to a significant RC delay on the differential amplifier sensing path that

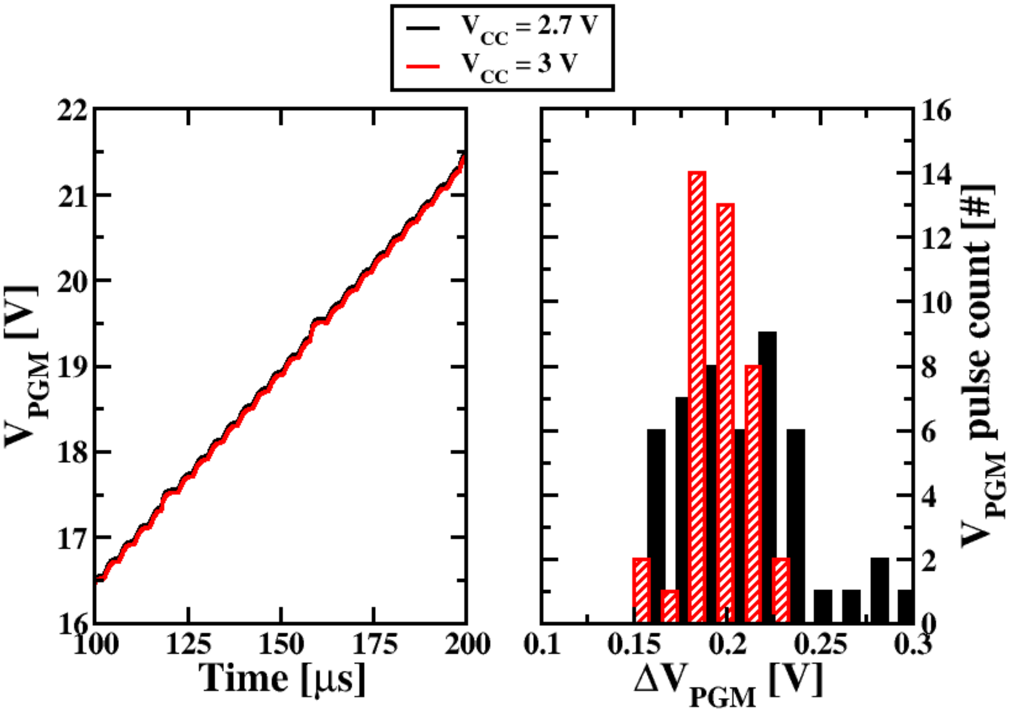


Figure 5.9: Linearity check of the V_{PGM} (insight) at different supply voltages (left) and ΔV_{PGM} distribution during the simulation of a TSP (right).

yields to a high output ripple; *ii)* the V_{PGM} changes with the supply voltage due to the power supply sensitivity of the trip point of the differential amplifiers in the regulators.

To accurately account for all the possible sources of reliability-loss induced by the HV circuitry, all the phases of a programming pulse in the TSP algorithm must be simulated. A single programming step is constituted by three phases: the self-boosting phase to inhibit the unselected cells from actual programming; the programming pulse at V_{PGM} for the selected cells to program; the read verify to check whether or not a selected cell is correctly programmed [15]. The switching from one phase to another must be timed accurately. The circuit block that manages all these operations is the wordline (WL) switch, that is constituted by several high voltage switches connected as show in Fig. 5.10. The WL switch is an analog multiplexer that takes as input several voltages generated by different charge pumps and regulators (i.e., the one for the inhibit voltage, the one for V_{PGM}, etc.) and outputs one voltage (V_{WL}) on the memory cells in a wordline.

The goal of the high voltage switches inside the WL switch is to transfer the voltages generated from the charge pumps and regulated by the HiREG and REG blocks whenever required, ensuring the minimal voltage degradation during the transfer. The circuit used in

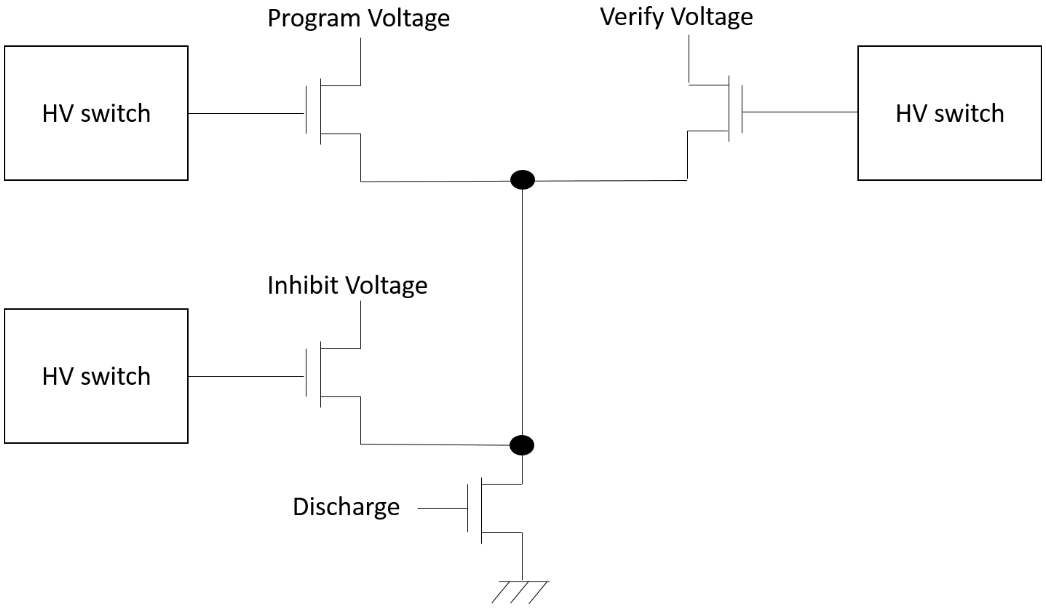


Figure 5.10: Simplified structure of a WL switch.

simulations is similar to the voltage doubler element used in the charge pump unit stages [1].

Several SPICE simulations have been performed by considering consecutive programming pulses at a ΔV_{PGM} distance. The inhibit voltage for the self-boosting phase has been assumed equal to 9 V. Typical pulse durations have been considered in the simulations [1]. For the sake of simulation speed the read verify operation is here not considered. When V_{CC} is lowered there are visible glitches during the transitions between the stages (see Fig. 5.11). This could cause additional disturbs in the programming due to a sub-optimal inhibit operation resulting again in distribution broadening. A ΔV_{pulse} difference in the generated V_{WL} between the two V_{CC} values due to the WL switch sensitivity to the power supply can be appreciated.

Finally, even if the supply voltage is considerably into the boundaries of the safe operating region, there could be some external AC and DC noise sources coupling with the NAND Flash power supply. Page buffers switching activity and high frequency signals on the data path of the NAND Flash system can be sources of disturb. To understand this issue SPICE simulations of the HV circuitry were performed by using either a stable V_{CC} or a V_{CC} with a superimposed sinusoidal AC noise at two different frequencies (i.e. 40 KHz and 100 KHz). To remain in the safe operating region a 3 V value was considered for the V_{CC} with a maximum noise of 100 mV peak-to-peak. The simulations demonstrates (see Fig. 5.12) the impact of different AC noise frequencies on the broadening of ΔV_{PGM}

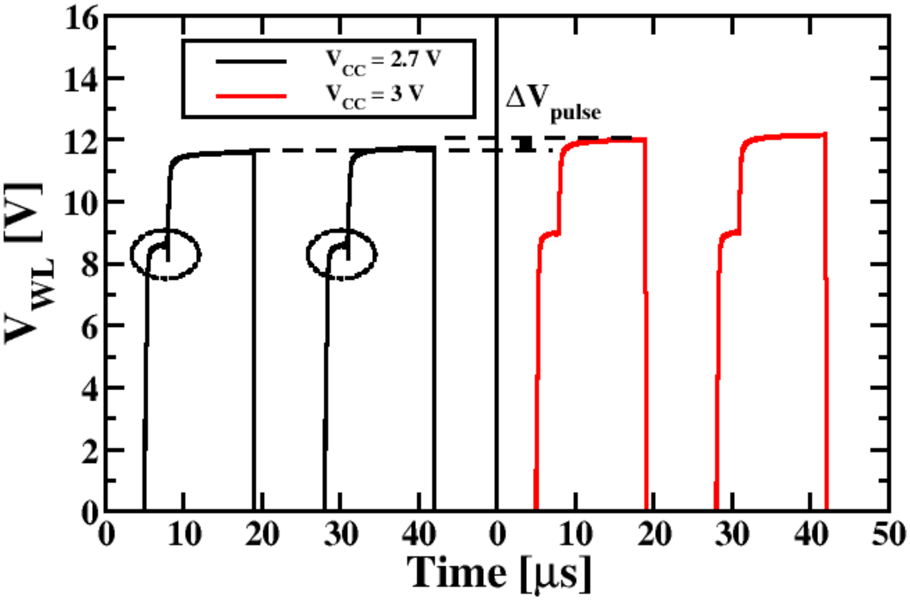


Figure 5.11: Output of the WL switch (self-boosting + programming) for two consecutive V_{PGM} pulses at a ΔV_{PGM} distance in a stage of a TSP algorithm.

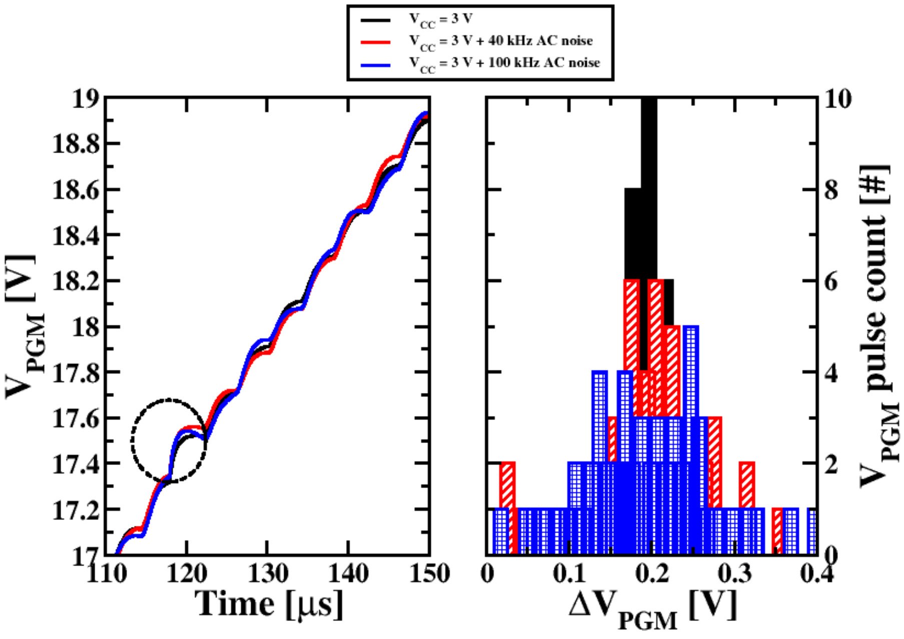


Figure 5.12: Linearity check of the VPGM (insight) at VCC = 3 V when AC noise is present (left) and VPGM distribution during the TSP (right).

distribution and therefore on the cells threshold voltage distributions.

In this chapter we have analyzed the impact of the power supply voltage on the reliability of a TLC mid-1X NAND Flash memory. Through an experimental characterization performed during endurance stress it was observed that the number of errors and the page-to-page errors variability strongly depend on the power supply. By simulating the different blocks of the high voltage circuitry in the NAND Flash system through a SPICE model we identified some of the possible culprits of this dependence, namely the regulators controlling the generation of the program voltage in the TSP algorithm and the wordline switch. Finally, we have also investigated the possible side effects of the coupled noise sources with the high voltage NAND Flash subsystem, evidencing that even if the power supply is chosen in a safe operating region, it is not immune from errors.

Bibliography

[1] R. Micheloni, L. Crippa, and A. Marelli, *Inside NAND Flash memories.* Springer-Verlag, 2010.

[2] A. Modelli, A. Visconti, and R. Bez, "Advanced flash memory reliability," in *IEEE International Conference on Integrated Circuit Design and Technology*, 2004, pp. 211–218.

[3] J.-D. Lee, C.-K. Lee, M.-W. Lee, H.-S. Kim, K.-C. Park, and W.-S. Lee, "A new programming disturbance phenomenon in NAND flash memory by source/drain hot-electrons generated by GIDL current," in *IEEE International Reliability Physics Symposium (IRPS)*, 2013, pp. 2E.3.1–2E.3.7.

[4] C. Compagnoni, R. Gusmeroli, A. Spinelli, and A. Visconti, "Analytical model for the electron-injection statistics during programming of nanoscale nand flash memories," *IEEE Transactions on Electron Devices*, vol. 55, pp. 3192–3199, 2008.

[5] A. Chimenton, C. Zambelli, and P. Olivo, "A statistical model of erratic behaviors in flash memory arrays," *IEEE Transactions on Electron Devices*, vol. 58, no. 11, pp. 3707–3711, 2011.

[6] Y. Park, J. Lee, S. S. Cho, G. Jin, and E. Jung, "Scaling and reliability of nand flash devices," in *IEEE International Reliability Physics Symposium (IRPS)*, 2014, pp. 2E.1.1–2E.1.4.

[7] E. Vatajelu, H. Aziza, and C. Zambelli, "Nonvolatile memories: Present and future challenges," in *9th International Design Test Symposium (IDT)*, 2014, pp. 61–66.

[8] S. Tanakamaru, Y. Yanagihara, and K. Takeuchi, "Over-10x-extended-lifetime 76%-reduced-error solid-state drives (SSDs) with error-prediction LDPC architecture and error-recovery scheme," in *IEEE International Solid-State Circuits Conference (ISSCC)*, Feb. 2012, pp. 424–426.

[9] H. Shim, S.-S. Lee, B. Kim, N. Lee, D. Kim, H. Kim, B. Ahn, Y. Hwang, H. Lee, J. Kim, Y. Lee, H. Lee, J. Lee, S. Chang, J. Yang, S. Park, S. Aritome, S. Lee, K.-O. Ahn, G. Bae, and Y. Yang, "Highly reliable 26nm 64Gb MLC E2NAND (Embedded-ECC & Enhanced-efficiency) flash memory with MSP (Memory Signal Processing) controller," in *Symposium on VLSI Technology (VLSIT)*, Jun. 2011, pp. 216–217.

[10] L. Zuolo, C. Zambelli, P. Olivo, R. Micheloni, and A. Marelli, "LDPC Soft Decoding with Reduced Power and Latency in 1X-2X NAND Flash-Based Solid State Drives," in *IEEE International Memory Workshop (IMW)*, May 2015, pp. 1–4.

[11] Micron, "MT29F512G08EMCBBJ5-6 TLC NAND Flash Data sheet," 2015.

[12] JEDEC, "JESD22-A117 document," Oct. 2011.

[13] L. Zuolo, C. Zambelli, R. Micheloni, D. Bertozzi, and P. Olivo, "Analysis of reliability/performance trade-off in solid state drives," in *IEEE International Reliability Physics Symposium (IRPS)*, Jun. 2014, pp. 4B.3.1–4B.3.5.

[14] Y. Li, "3 Bit Per Cell NAND Flash Memory on 19nm Technology," Flash Memory Summit, Aug. 2012.

[15] K.-D. Suh, B.-H. Suh, Y.-H. Um, J.-K. Kim, Y.-J. Choi, Y.-N. Koh, S.-S. Lee, S.-C. Kwon, B.-S. Choi, J.-S. Yum, J.-H. Choi, J.-R. Kim, and H.-K. Lim, "A 3.3 V 32 Mb NAND flash memory with incremental step pulse programming scheme," in *IEEE International Solid-State Circuits Conference (ISSCC)*, Feb. 1995, pp. 128–129.

[16] S.-G. Jung, K.-W. Lee, K.-S. Kim, S.-W. Shin, S.-S. Lee, J.-C. Om, G.-H. Bae, and J.-H. Lee, "Modeling of v_{th} shift in nand flash-memory cell device considering crosstalk and short-channel effects," *IEEE Transactions on Electron Devices*, vol. 55, no. 4, pp. 1020–1026, 2008.

[17] S. K. Won, Y. Noh, H. Cho, J. Ryu, S. Choi, S. Choi, D. Kim, J. Chung, B. Han, and E. Y. Chung, "High-voltage wordline generator for low-power program operation in NAND flash memories," in *IEEE Asian Solid State Circuits Conference (A-SSCC)*, Nov. 2011, pp. 169–172.

[18] G. Campardo, R. Micheloni, and D. Novosel, *VLSI-Design of Non-Volatile Memories*. Springer-Verlag, 2005.

[19] G. Palumbo, D. Pappalardo, and M. Gaibotti, "Charge-pump circuits: power-consumption optimization," *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 49, no. 11, pp. 1535–1542, 2002.

[20] Y. H. Kang, J. K. Kim, S. W. Hwang, J. Y. Kwak, J. Y. Park, D. Kim, C. H. Kim, J. Y. Park, Y. T. Jeong, J. N. Baek, S. C. Jeon, P. Jang, S. H. Lee, Y. S. Lee, M. S. Kim, J. Y. Lee, and Y. H. Choi, "High-Voltage Analog System for a Mobile NAND Flash," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 2, pp. 507–517, 2008.

Chapter 6

Uniform and concentrated Read Disturb effects in TLC NAND Flash memories

Triple Level Cell (TLC) NAND Flash memories are largely exploited in enterprise Solid State Drives (SSD) thanks to their high storage density and low cost per bit [1]. However, these storage architectures are suitable mostly for read-intensive applications since their endurance in terms of sustainable program/erase (P/E) cycles is quite low compared to other NAND Flash storage paradigms. Such a usage constraint exacerbates a reliability issue that was almost negligible for previous NAND Flash generations, namely the read disturb. The typical read disturb configuration is the one described in Fig. 6.1. All the cells belonging to the same string of the cell to be read in a wordline must be driven with a V_{read}, independently of their stored charge. The relatively high V_{pass} bias applied to the control gate of neighbor cells may trigger several effects due to hot carrier degradation, stress-induced leakage currents, and charge loss that result in a perturbation of the threshold voltage distributions of a programmed block (see Fig. 6.2), yielding in turn to read errors [2,3]. These errors are corrected by dedicated Error Correction Code (ECC) modules in the SSD or by secondary correction mechanisms on the NAND Flash like the read retry. Several techniques, either at device or at system level, have been proposed to reduce the occurrence of the disturb for a given application, although no clear indications were provided on the read access model of the memory leading to its insurgence [4, 5]. Some concerns about the application-specific read usage models in Flash memories were debated in [6], although never targeting neither NAND Flash nor SSD.

In this chapter we will show, through an extensive read disturb characterization on mid-

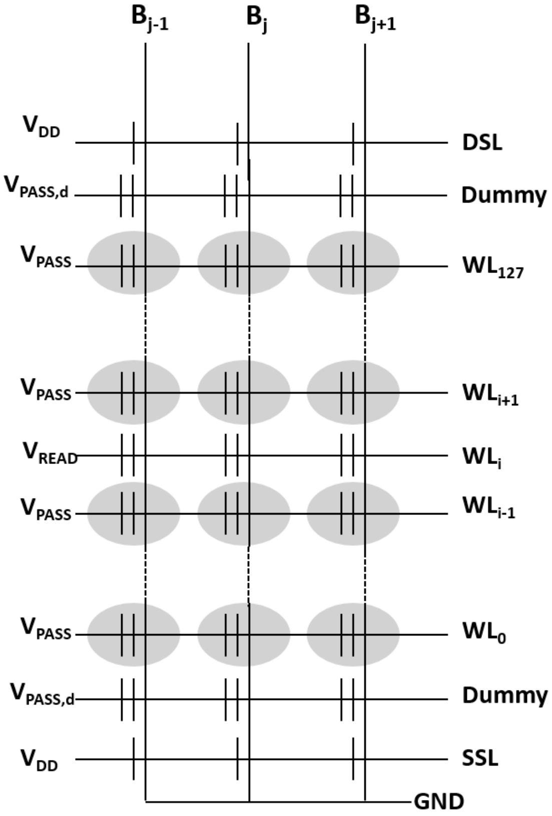


Figure 6.1: Bias configuration for a read operation applied on a NAND Flash block. The cells in gray are those suffering the read disturb.

1X TLC NAND Flash memories, that different read usage models of NAND Flash blocks (i.e., uniform versus concentrated) in an SSD lead to different constraints and guard band strategies against the disturb. The variability of the disturb entity among different pages and wordlines of a block are presented and related to specific properties. This work is particularly useful to SSD controller designers that need developing firmware strategies to counteract the read disturb for a given workload profile.

6.1 Data collection and analyses

The read disturb characterization of several NAND Flash memory devices has been performed with the test equipment shown in Fig. 6.3. The system is an advanced version of that already presented in [7] and is composed by a state-of-the-art ASIC PCIe Gen3 NVMe memory controller used for SSDs [8] dealing with NAND Flash commands for accessing the devices, a DRAM buffer for temporary data storage, and a set of SO-DIMM sockets

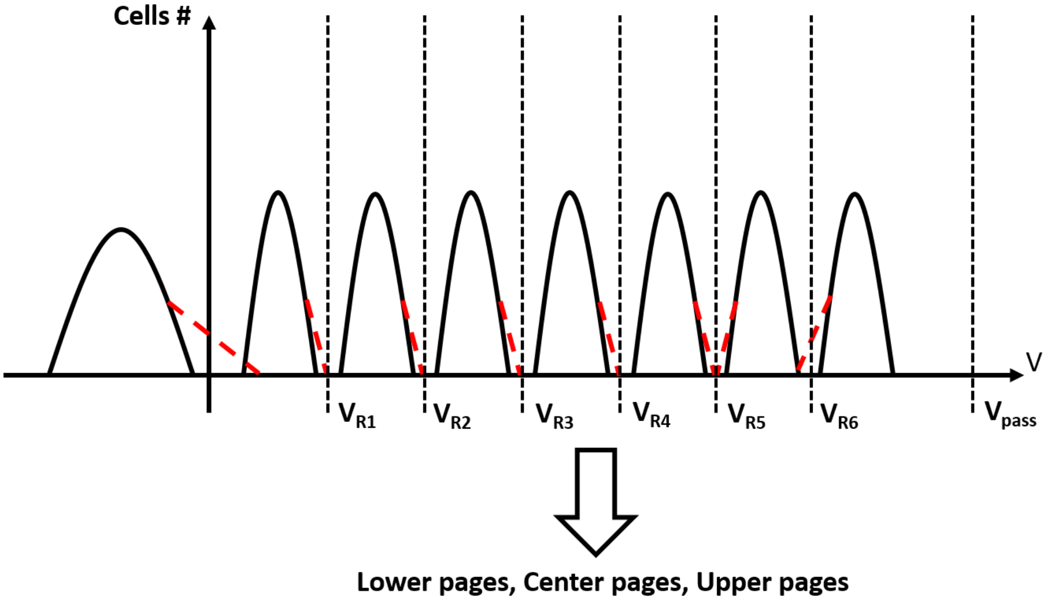


Figure 6.2: Threshold voltage distribution and TLC page coding. Red dashed lines indicate the effect of the read disturb on the distributions.

for mid-1X TLC NAND Flash interfacing. The board hosts up to 8 SO-DIMMs each one populated with 8 NAND Flash chip. A single chip contains 8 memory dies. The supply voltages are provided by an external regulated power supply. The characterization system communicates through a PCIe interface with an x86-PC where the data are collected for post-processing purpose.

In TLC NAND Flash the state-of-the-art read disturb testing is performed by cycling all the pages within the memory blocks up to a given P/E for lower, center, and upper pages and then consecutively reading the content of that block following the programming sequence [9]. The cycling has been performed up to 3k P/E with minimal dwell time. Both cycling and read disturb were performed at room temperature. Fig. 6.4 shows that the maximum number of errors retrieved for all the wordlines in a block and for different page types increases with the number of block reads by following a power law. The disturb growth error rate heavily depends on the P/E cycle count (see Fig. 6.5). By breaking down the wordline contribution to the read disturb (see Fig. 6.6) for each page type it is possible to appreciate a reproducible signature: the wordlines close to the drain selector are those heavily affected by the disturb in center and upper pages, whereas in lower pages the trend is inverted toward the source selector. This is related both to the different electric fields near the Dummy wordlines and to the effective number of experienced V_{pass}.

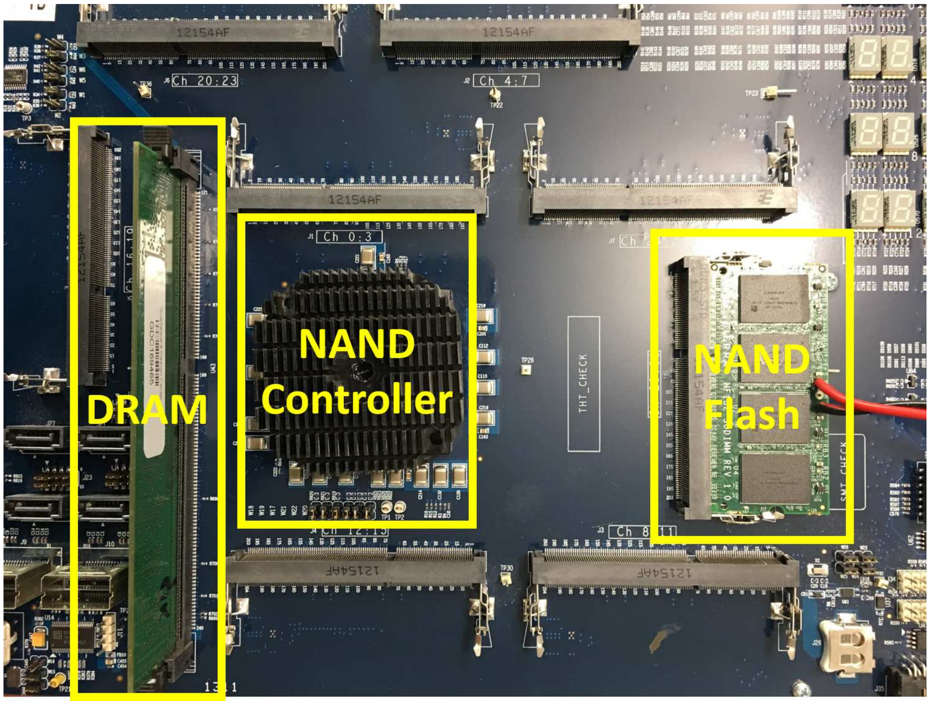


Figure 6.3: Threshold voltage distribution and TLC page coding. Red dashed lines indicate the effect of the read disturb on the distributions.

As said, the read disturb testing is performed with a sequential read access usage model (i.e., from the first to the last word-line) up to 3300 block reads. Considering a block composed by 128 wordlines (i.e., 384 pages), the total page read count is 384*3300=1267200 per block. The page read count is the number tracked by many algorithms in the SSD firmware coping with read disturb to understand its criticality for the disk reliability. With a uniform read access of the NAND Flash blocks every page gets the same amount of accesses. However, there are some read-intensive applications where the number of reads applied to the pages is concentrated in some regions of the block (see Fig. 6.7) [10].

Taking the analysis to extreme conditions we show the impact of concentrating all the page reads on a single page of a block, therefore we applied 1267200 page reads after 3k P/E either on a page in the center or in the last wordline of the block. As shown in Fig. 6.8, when the reads are concentrated on a single page, the errors profile on the wordlines is the same as for the uniformly distributed read disturb except for the two neighbor wordlines to the continuously read one. In fact, these two wordlines suffer from hot carrier degradation effects that heavily change their threshold voltage distribution. The worst case for concentrated read disturb is found to be in the penultimate wordline where the error count, especially for center pages, is the highest. Spreading the 1267200 reads on two pages in

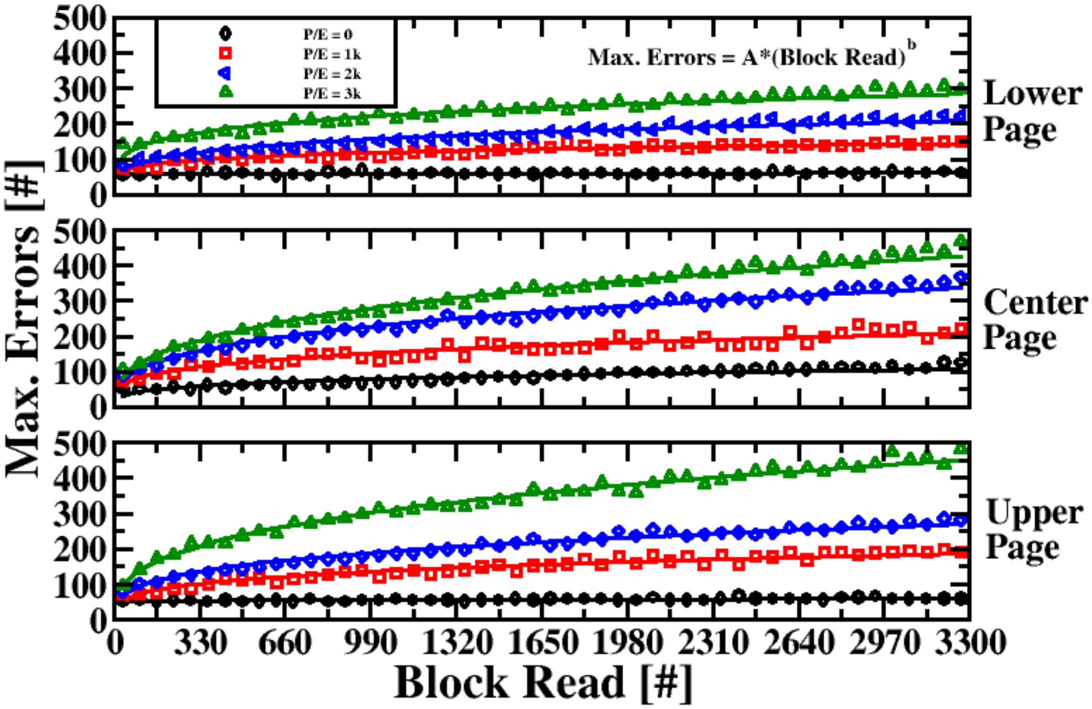


Figure 6.4: Maximum errors number retrieved in all the wordlines of a specific NAND Flash page type at different P/E cycles. Data are interpolated through the power-law indicated in figure.

the same block reduces the number of errors, but in this case four wordlines (i.e., 12 pages) become corrupted.

6.2 Implications on enterprise SSDs

The read disturb handling strategies in enterprise SSD use the ECC engine to understand whether the disturb effect is becoming critical for the reliability. When the number of errors in a page due to read disturb reaches a threshold that is defined by the SSD firmware, the entire block where the page belongs is relocated on another available in the SSD, and then erased to reset the disturb effect. The relocation is a critical operation since it triggers an additional P/E cycle for the block and therefore must be carefully handled to avoid lifetime limitation of the disk. Fig.6.9 shows the achievable page reads count on a block before its relocation after 3k P/E cycles by considering a 220b/4320B ECC. Different error thresholds (i.e., the 80% and the 90% of the error correction capacity) were considered as well as lifetime enhancement strategies like the read retry technique that lower the errors number

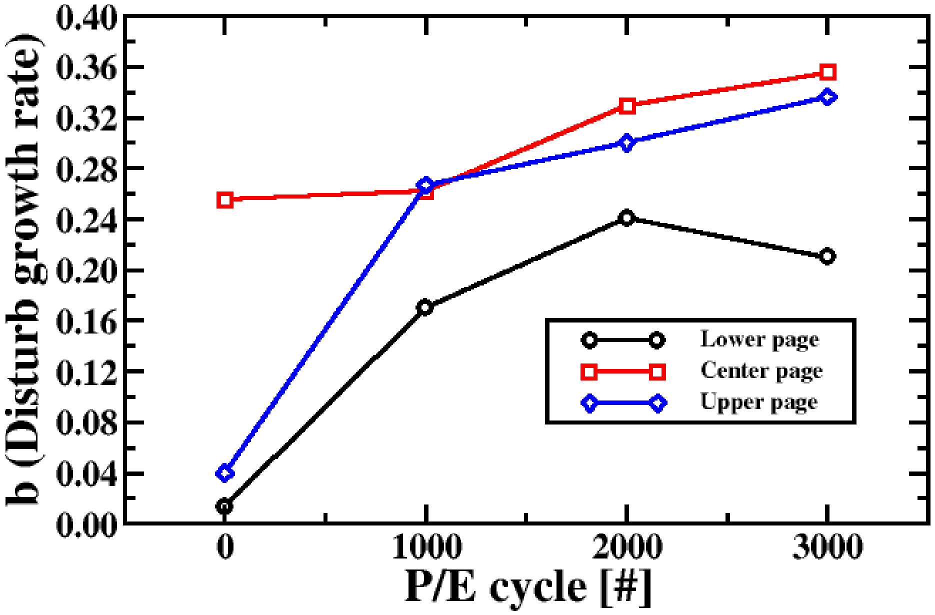


Figure 6.5: Dependency of the disturb growth rate (fitting coefficient b of the power-law) on the P/E cycles.

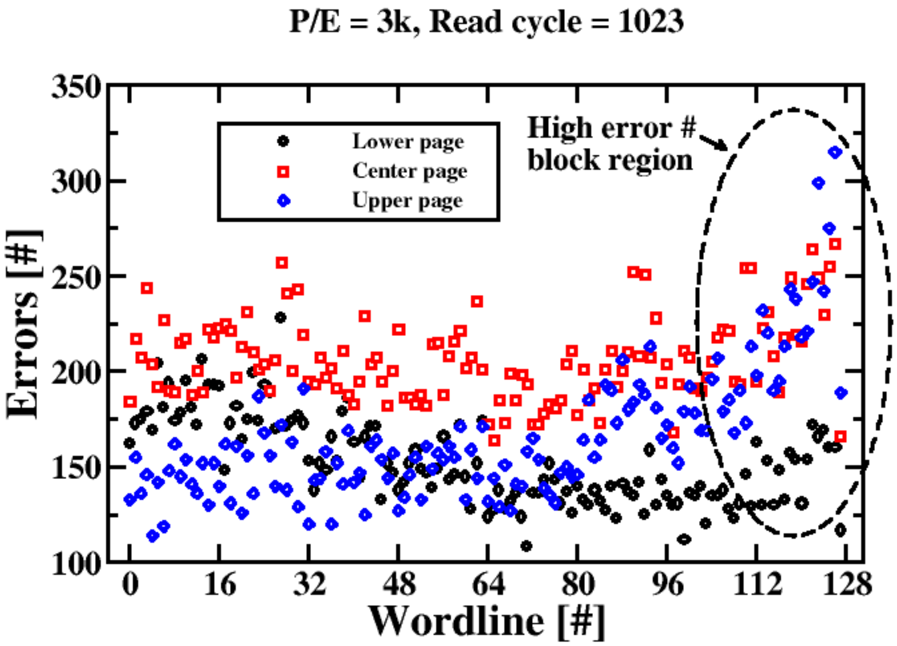


Figure 6.6: Wordline contribution breakdown on the read disturb errors. The region with higher error count is evidenced.

by shifting the V_{Ri} read reference voltages indicated in Fig.6.2 [11].

Results prove that for the worst case access mode (i.e., concentrated page reads on the

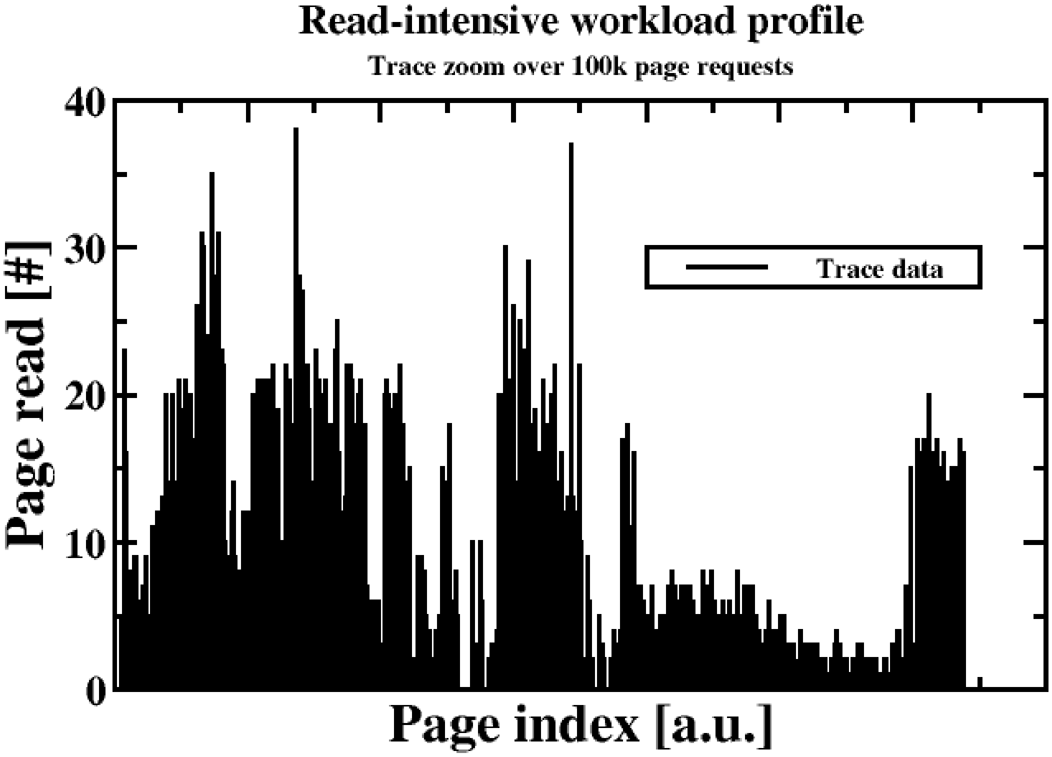


Figure 6.7: Example of a read-intensive workload profile (websearch [9]) retrieved for 100k page reads.

penultimate wordline of the block) there is up to a 22% loss in sustainable read operations on the same block. It is worth to mention that this is an extreme operating condition, therefore we expect that less aggressive read disturb usage models will be applied to the memory, thus leading to a reduction of that factor. However, even if that value would be reduced there will still be an impact on the disk lifetime.

To better assess the impact of the previous considerations on enterprise SSD we have considered different read-intensive applications by extracting the number of requested page reads on a block per day [10]. The applications indicated in Table 6.1 are: a high performance computing facility, an online web search service, an OnLine Transaction Processing (OLTP) system, and a proxy web server. Fig.6.10 shows the number of relocations per day spent to counter the read disturb. A block relocation is triggered whenever the block read count exceeds the achievable page read counts according to the ECC strategy adopted with or without the read retry feature. Such a number impacts the write amplification factor of the disk since internal data movement is required at the expense of reducing the P/E amount that the disk can sustain. The higher is the value, the higher is the impact on the reliability,

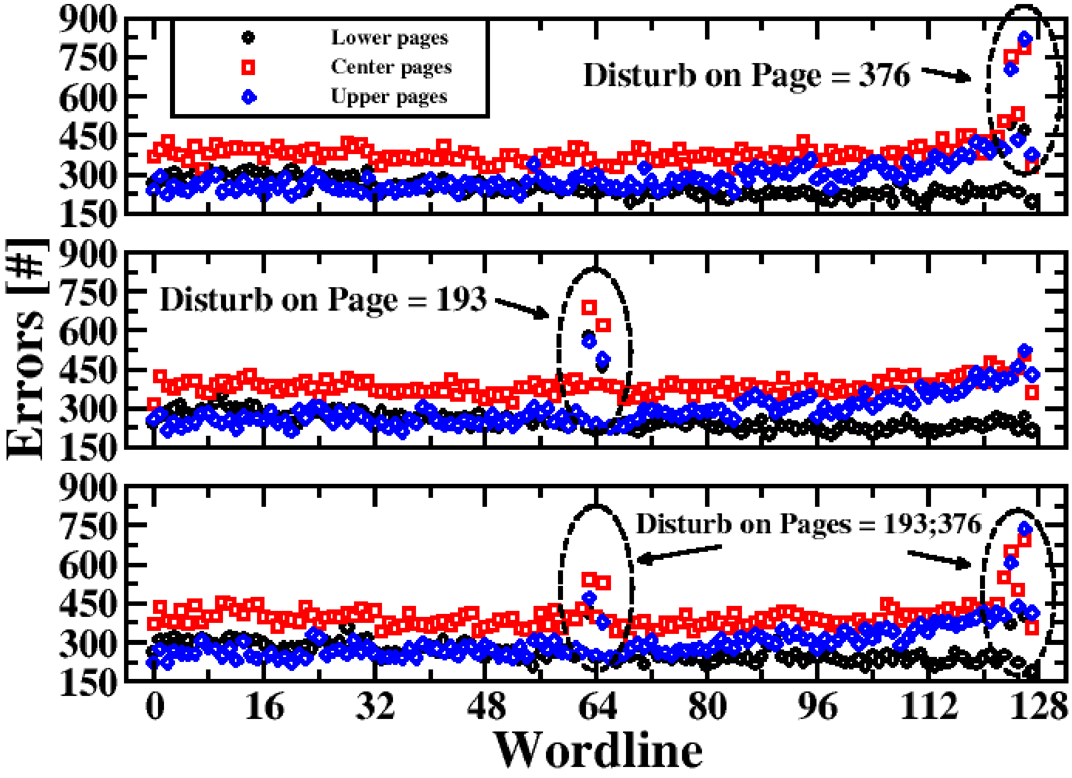


Figure 6.8: Concentrated read disturb effect when different disturb configurations are applied (i.e., one page in worst location, one page in best location, two pages).

Table 6.1: Read-intensive applications for enterprise SSD

Trace	Source	Daily block read count
cello99 (HPC)	HP Labs	51879
websearch	UMass	87405
financial (OLTP)	UMass	247004
prxy (Server)	Microsoft Research	421456

on the lifetime of the disk, and on the overall power consumption overhead caused by the additionally needed SSD operations. Applications like OLTP or web servers triggers in the worst concentrated read disturb case up to 1.9 block relocations per day, therefore particular care must be taken both in data storage phase and on the data access policies.

In this chapter we have investigated the differences between uniform and concentrated read disturb effects in mid-1X TLC NAND Flash memories. The characterization showed that a uniform read access of NAND Flash blocks yields to a reproducible signature of the

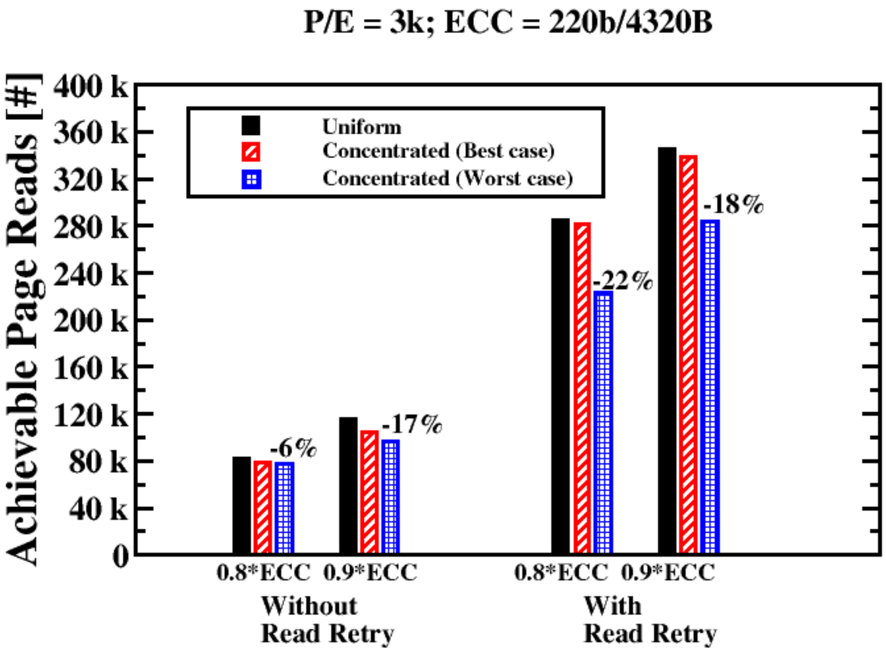


Figure 6.9: Achievable page reads count on a mid-1X TLC NAND Flash block as a function of the read disturb model (i.e., uniform versus concentrated) for different SSD disturb handling strategies.

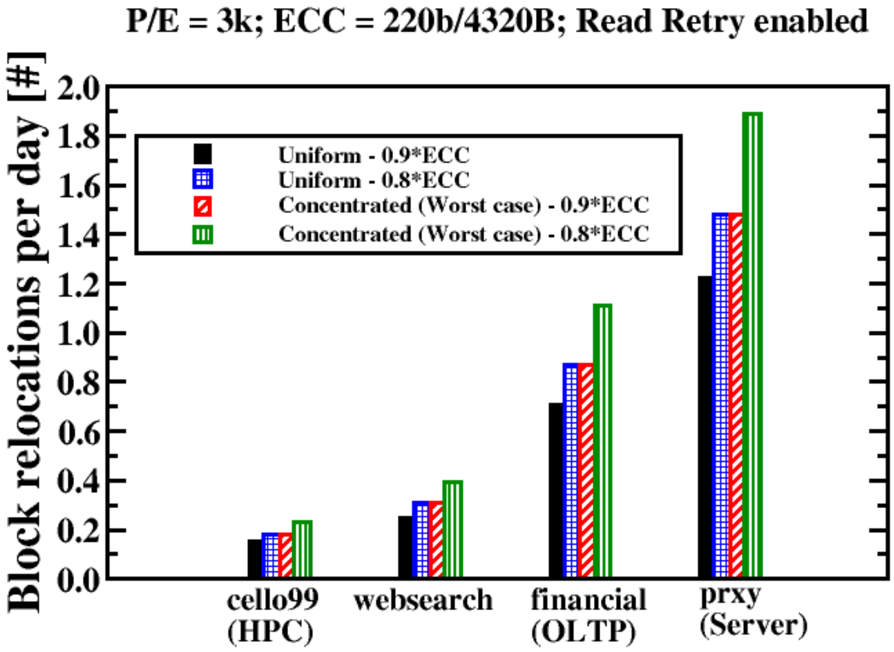


Figure 6.10: Calculated number of block relocations per day as a function of the workload supplied to the SSD.

disturb. The wordlines close to the drain selector are those heavily affected by the phenomenon. By characterizing the concentrated read access mode, better mimicking the real workloads that a memory will sustain throughout its lifetime, it is appreciable that the errors profile on the wordlines is similar to the uniform case except for the two neighbors closer to the one where the read accesses are concentrated. The implications on the enterprise SSD are evident: when a concentrated read access is performed there is up to a 22% achievable page reads count by a block before requesting the intervention of the ECC or other data management policies (e.g., scrubbing). For server and OLTP application this could represent a limitation in terms of reliability, endurance, and power consumption of the drive.

Bibliography

[1] R. Micheloni, L. Crippa, and A. Marelli, *Inside NAND Flash memories*. Springer-Verlag, 2010.

[2] S. Satoh, G. Hemink, K. Hatakeyama, and S. Aritome, "Stress-induced leakage current of tunnel oxide derived from flash memory read-disturb characteristics," *IEEE Transactions on Electron Devices*, vol. 45, no. 2, pp. 482–486, 1998.

[3] H. H. Wang, P. S. Shieh, C. T. Huang, K. Tokami, R. Kuo, S. H. Chen, H. C. Wei, S. Pittikoun, and S. Aritome, "A New Read-Disturb Failure Mechanism Caused by Boosting Hot-Carrier Injection Effect in MLC NAND Flash Memory," in *IEEE International Memory Workshop*, May 2009, pp. 1–2.

[4] A. Kobayashi, T. Tokutomi, and K. Takeuchi, "Versatile TLC NAND flash memory control to reduce read disturb errors by 85% and extend read cycles by 6.7-times of Read-Hot and Cold data for cloud data centers," in *IEEE Symposium on VLSI Circuits*, Jun. 2016, pp. 1–2.

[5] D. J. Post and H. Thio, "Read disturb scorecard," US patent 8,503,257 B2, Aug. 2013.

[6] T. S. Harp, P. J. Kuhn, J. M. Higman, R. E. Paulsen, and B. E. Hornung, "An application-specific usage model for flash memory read disturb reliability," in *International Reliability Physics Symposium (IRPS)*, Apr. 2001, pp. 280–282.

[7] C. Zambelli, P. King, P. Olivo, L. Crippa, and R. Micheloni, "Power-supply impact on the reliability of mid-1X TLC NAND flash memories," in *International Reliability Physics Symposium (IRPS)*, Apr. 2016, pp. 2B–3–1–2B–3–6.

[8] Microsemi Corp., "Flashtec nvme controllers," [Online] - Available: http://www.microsemi.com/products/storage/flashtec-nvme-controllers/flashtec-nvme-controllers, 2017.

[9] N. Papandreou, T. Parnell, T. Mittelholzer, H. Pozidis, T. Griffin, G. Tressler, T. Fisher, and C. Camp, "Effect of Read Disturb on Incomplete Blocks in MLC NAND Flash Arrays," in *IEEE International Memory Workshop*, May 2016, pp. 1–4.

[10] Y. Cai, Y. Luo, S. Ghose, and O. Mutlu, "Read Disturb Errors in MLC NAND Flash Memory: Characterization, Mitigation, and Recovery," in *IEEE/IFIP International Conference on Dependable Systems and Networks*, Jun. 2015, pp. 438–449.

[11] L. Zuolo, C. Zambelli, P. Olivo, R. Micheloni, and A. Marelli, “LDPC Soft Decoding with Reduced Power and Latency in 1X-2X NAND Flash-Based Solid State Drives,” in *IEEE International Memory Workshop (IMW)*, May 2015, pp. 1–4.

Chapter 7

RRAM-based SSDs

SSDs are the most effective solution for both consumer applications and large enterprise environments when high performance storage devices are required [1]. To cope with the increasing request of data storage, especially for large computing facilities, there is a call for a continuous expansion of the bit density in the SSD storage medium, namely the NAND Flash. This is generally achieved through either a technology shrink or a multi-bit per cell storage or both; in all cases, it implies a significant degradation of memory speed and reliability, thus impacting the main figures of merit of an SSD (i.e., latency and bandwidth) [2].

Resistive RAM (RRAM) is perceived by the storage community as a reliable alternative to NAND Flash in SSDs for low latency applications [3]. These emerging memories are non-volatile as NAND flash, but with a lower read/write latency and a higher reliability. However, the relatively small storage capacity of RRAM memories integrated so far [4, 5] has limited their usage to specific applications such as saving critical data during power loss events or as a cache memory for fast data manipulation, like in the hybrid system described in [6]. In this case, RRAMs are combined with NAND flash memories to minimize latency and to improve both the bandwidth and the reliability of the drive.

To increase the density of RRAM memory arrays, several researchers started to consider advanced 3D architectures. Among them, the 1T-nR approach seems the easiest to integrate by stacking multiple RRAM planes, each one selected by a proper decoding structure [7]. The 1T-nR arrays often utilize cross-point core architectures for higher density and one transistor that drives n RRAM devices. Those arrays are forecasted to be fully compatible with the state-of-the-art NAND Flash interface [8, 9], paving the way to innovative "All-RRAM" SSD's architectures. In these systems, NAND flash memories are completely replaced by RRAM devices offering a highly reliable and extremely faster storage medium.

In this chapter, a thorough design space exploration of a 512 GB All-RRAM SSD architecture is performed, with particular attention to architectural bottlenecks and inefficiencies, by using the SSDExplorer co-simulator [10]. We assumed a full compatibility of RRAM chips with typical NAND flash interfaces [11, 12], and hence a state-of-the-art SSD controller is embodied in the simulation environment. In light of these considerations, we leverage both the internal page architecture of a 1T-nR RRAM chip [8] and the SSD's firmware to find the optimal configuration, thus enabling the adoption of the RRAM technology in high performance SSD applications. Collected results show that, in standard working condition (i.e., when 4 kB transactions are issued by the host system), All-RRAM SSDs are able to show extremely low latency only if a proper management of the operations is adopted.

7.1 All-RRAM SSD architecture

The RRAM chip considered in the simulated SSD architecture is a configurable 16 planes 32 Gbits memory module with a 8 bit ONFI 2.0/Toggle Mode interface, capable of 200 MB/s [11, 12] (see Fig. 7.1). Each plane is a 2 Gbit RRAM array with a page size of 256 B [8]. The RRAM chip features an internal memory controller that can work either with a native addressing mode (i.e. 256 byte-wide page) or in a multi-plane emulated addressing mode, which allows accessing from 512 B up to 4 kB within a single operation. A read operation takes 1μs per page. The main array characteristics of this technology are summarized in Table 7.1.

The simulated SSD configuration (sketched in Fig. 7.2) is a 512 GB drive made of 16 channels; each channel is populated with 8 RRAM targets. SSD controller, Error Correction Code modules and DRAM buffers are included to keep the compatibility with state-of-the-art NAND-Flash based SSDs [10]. The drive host interface is a PCI-Express Gen2 with 8 lanes adopting the NVM-Express protocols [13], which is typical in enterprise-class SSDs. Different 100% random read workloads were used to test the 2 addressing modes described above, by aligning the logical block address of the drive with the effective RRAM page size. Write workloads are not described in this chapter since the target drive architecture makes use of DRAM buffers where write operations are cached; therefore they do not represent a significant threat for the latency and bandwidth figures.

Table 7.1: Main characteristics of the simulated RRAM devices

Chip parameter	Configuration
IO-Bus interface	ONFI 2.0 / Toggle Mode
IO-Bus speed	200 MB/s
Native Page Size	256 B
Emulated Page Size	512-1024-4096 B
t_{READ} per Page	1 μs

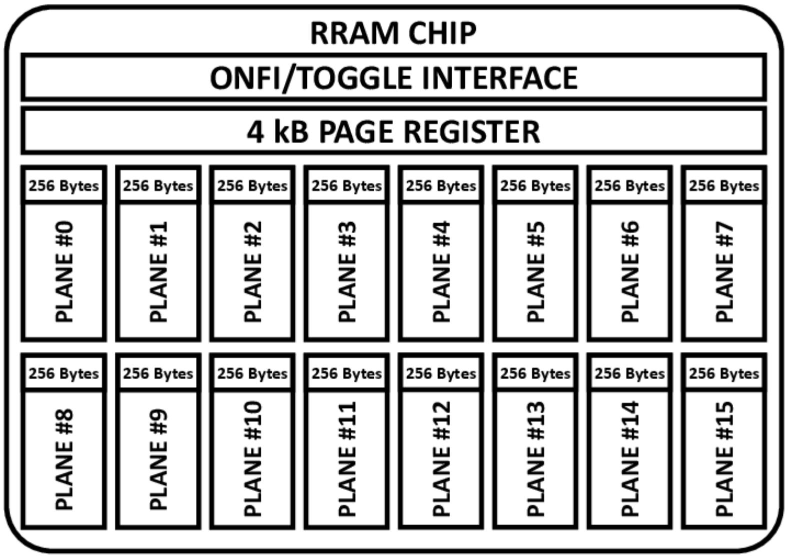


Figure 7.1: 32 Gbit RRAM memory module architecture.

7.2 Page size vs. queue depth

A statistical assessment of the All-RRAM SSD read latency and bandwidth figures was performed by simulating 500,000 random read operations, with different RRAM page sizes and queue depths. As shown in Figs. 7.3 and 7.4, by setting a fixed queue depth of 16 read commands, the read bandwidth of the SSD increases proportionally

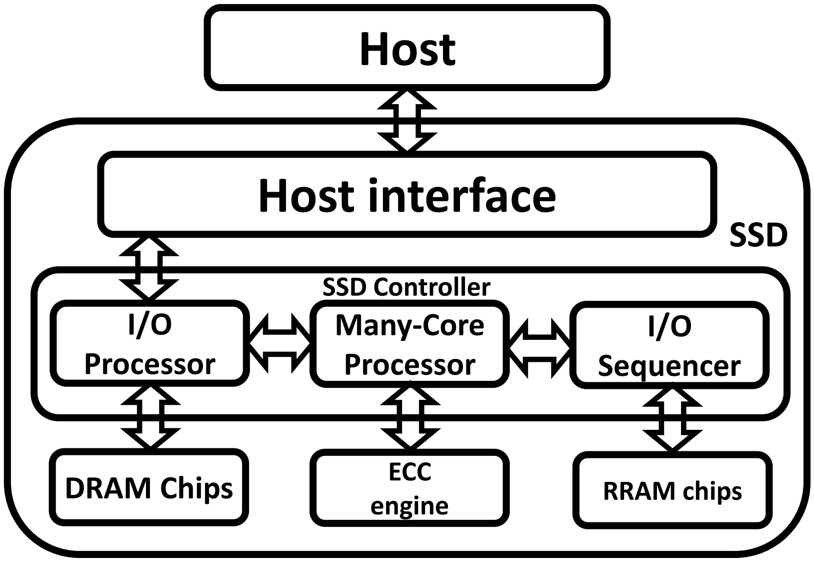


Figure 7.2: Block diagram of the simulated All-RRAM 512 GB SSD architecture.

with the memory page size, while the latency remains almost constant. A different behavior is observed when the RRAM page size is fixed and the read commands queue depth is varied from 1 to 32. The bandwidth increases proportionally with the queue depth up to a saturation level, which depends on the RRAM page size (see Fig. 7.5). For latency, trend is the same but the saturation happens for lower queue depth values (see Fig. 7.6).

In SSD architectures, especially those for enterprise environments, it is extremely important to analyze the *Quality of Service* (QoS), with special focus on read. A slower QoS of the read operation corresponds to a longer response time of the drive when the host wants to read data for subsequent manipulation [14]. The *Cumulative Distribution Function* (CDF) and the *Probability Density Function* (PDF) of the latency are very useful tools for this kind of analyses. Fig. 7.7 shows CDF and PDF for the All-RRAM SSD. When user transactions match the RRAM chip page size (i.e. 256 B), and only one operation is served at a time, latency gets extremely low, in the range of tens of microseconds. The longest read response time (i.e., 99.99 percentile of the CDF) is around 16 μs, which is well below the few hundreds of microseconds offered by NAND Flash-based SSDs. However, such queue depth and RRAM page size do not reflect the workload conditions of the state-of-the-art host platforms and file-systems, which are designed to issue multiple read operations with a fixed payload of 4 kB. Looking at Fig. 7.8, when the host interface queue depth is fixed to 32 commands and user operations match the native 256 B RRAM addressing mode, the median latency rapidly increases up to 66 μs. Eventually, with a 4 kB page All-RRAM SSD, read response times become very similar to those of a simulated 1x-nm MLC NAND Flash-based SSD (Fig. 7.9).

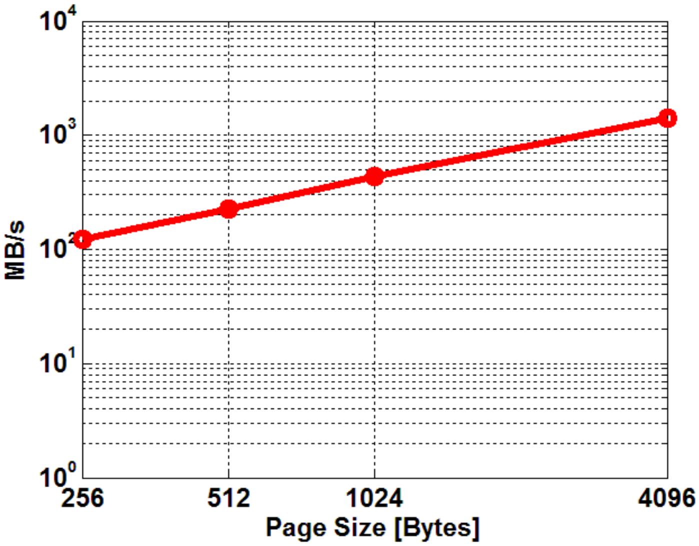


Figure 7.3: Simulated SSD's average read bandwidth as a function of the RRAM page size, with a queue depth of 16 commands.

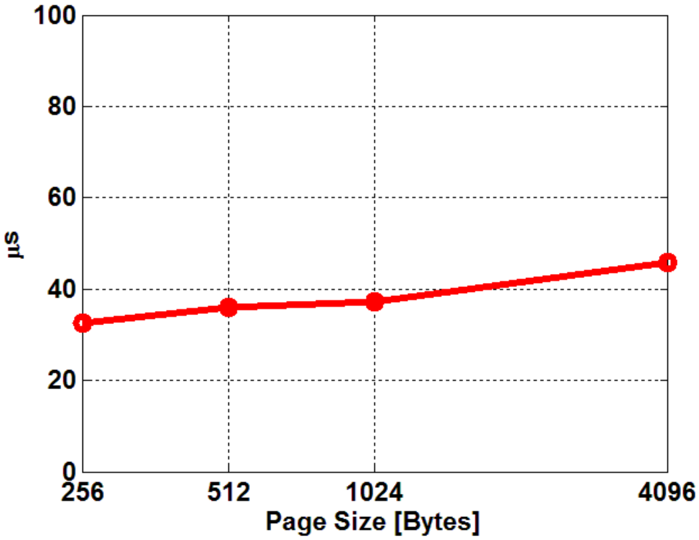


Figure 7.4: Simulated SSD's average read latency as a function of the RRAM page size, with a queue depth of 16 commands.

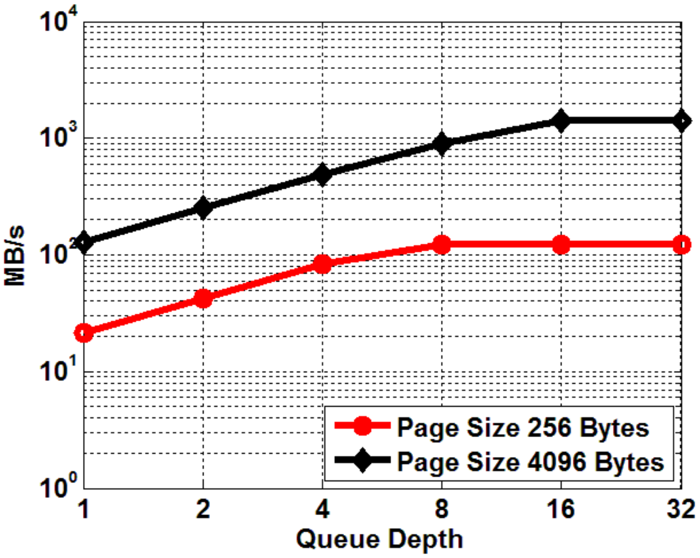


Figure 7.5: Simulated SSD's average read bandwidth as a function of the host interface queue depth. Native 256 B and 4 kB multi-plane RRAM addressing modes are considered.

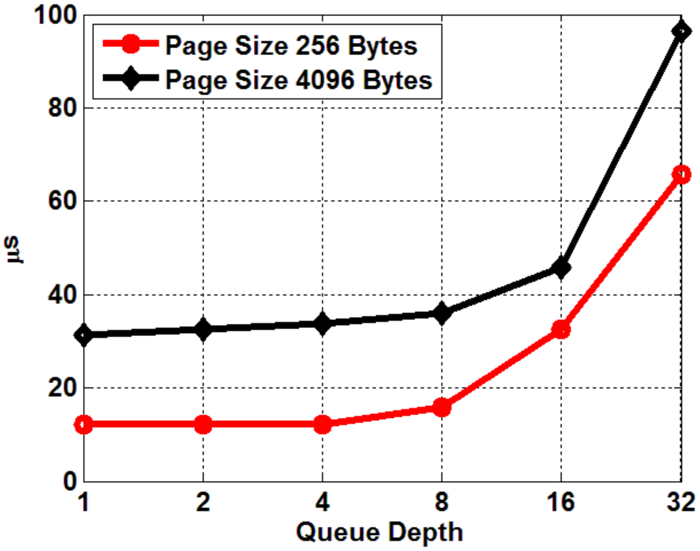
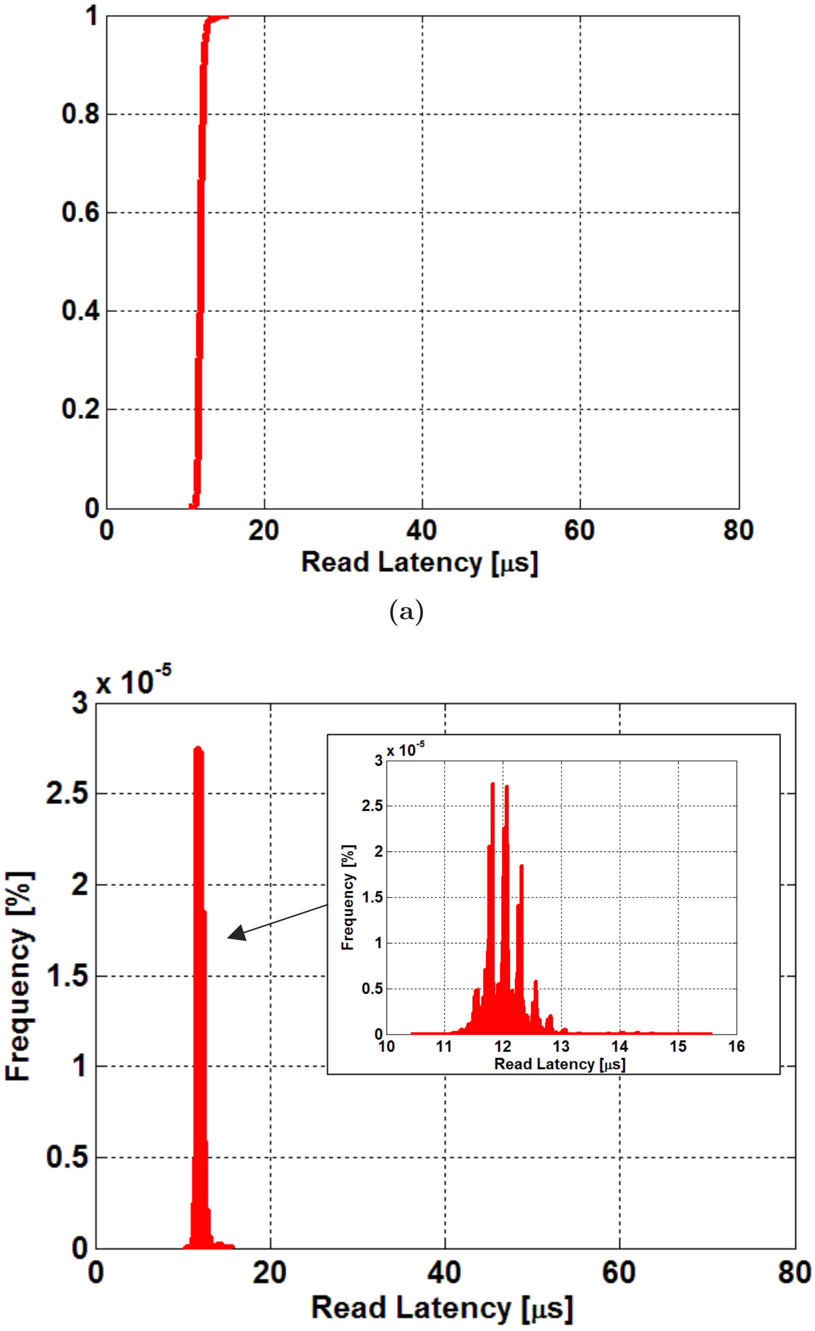


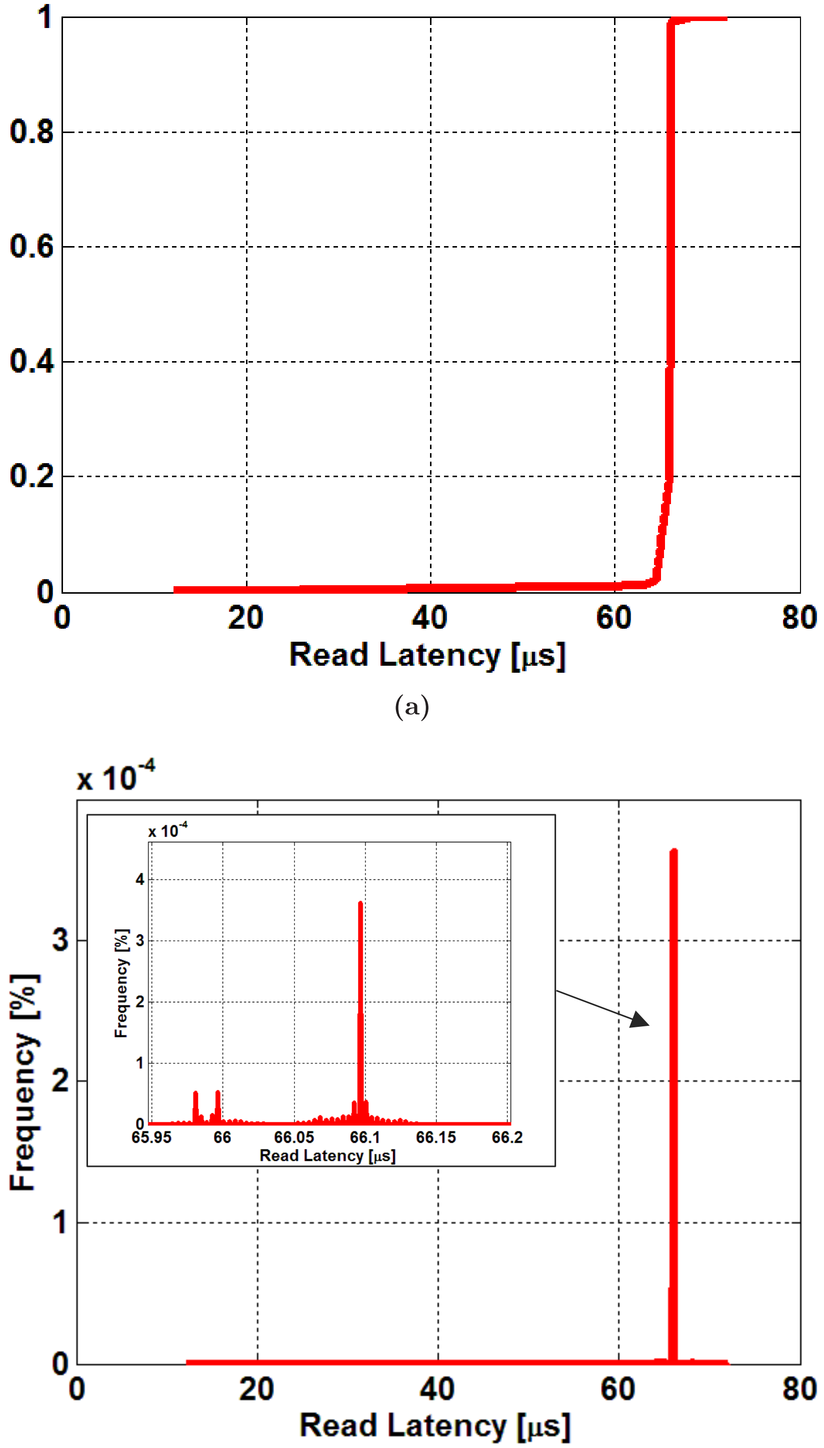
Figure 7.6: Simulated SSD's average read latency as a function of the host interface queue depth. Native 256 B and 4 kB multi-plane RRAM addressing modes are considered.



(a)

(b)

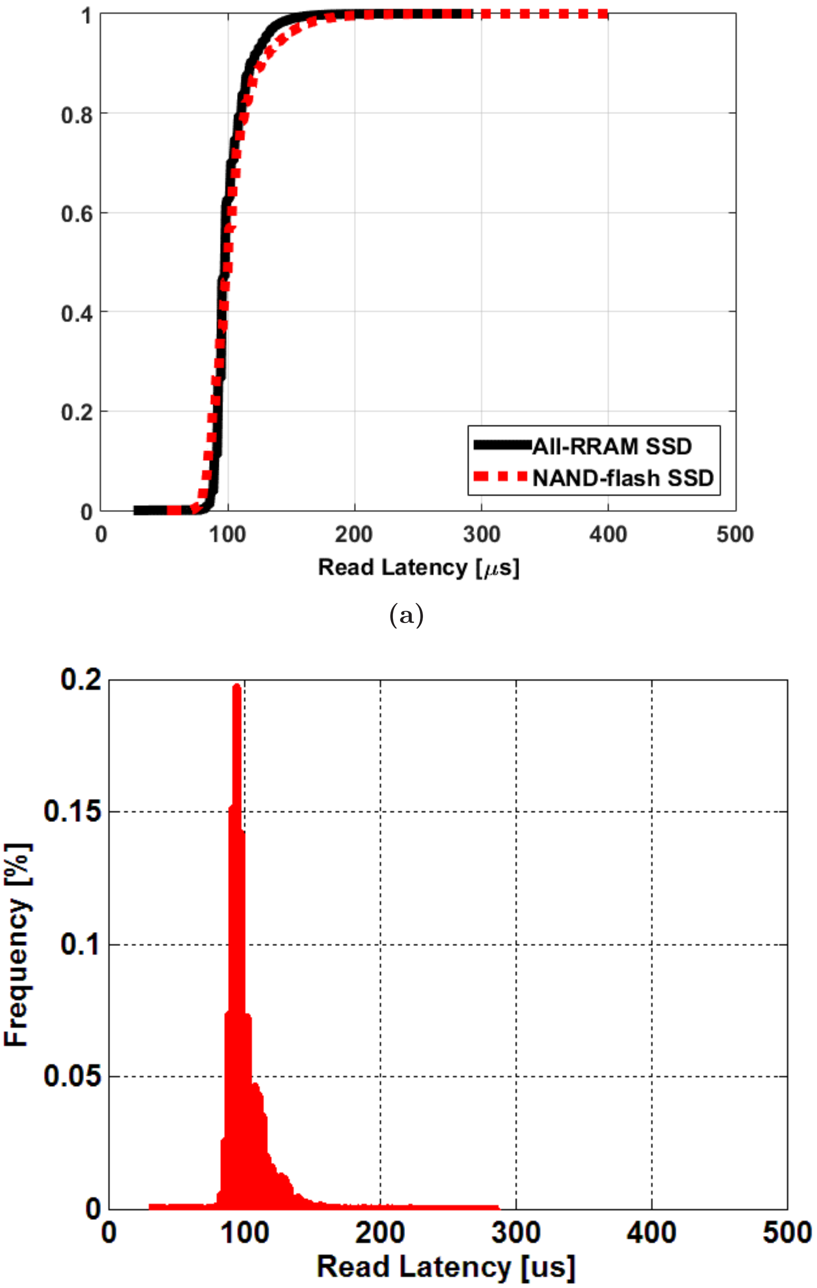
Figure 7.7: CDF (a) and PDF (b) of the simulated SSD's read latency with a queue depth of 1 command and a native 256 B RRAM page size.



(a)

(b)

Figure 7.8: CDF (a) and PDF (b) of the simulated SSD's read latency with a queue depth of 32 commands and a native 256 B RRAM page size.



(a)

(b)

Figure 7.9: CDF (a) and PDF (b) of the simulated SSD's read latency with a queue depth of 32 commands and the emulated 4 kB RRAM page size. A comparison with a state-of-the-art NAND Flash SSD is provided.

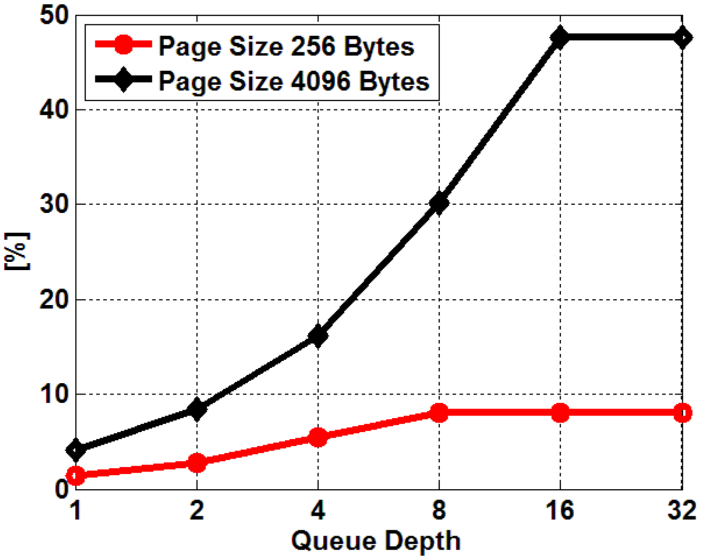


Figure 7.10: Average RRAM I/O bus interface usage as a function of the host interface queue depth. Native 256 B and 4 kB multi-plane RRAM addressing modes are considered.

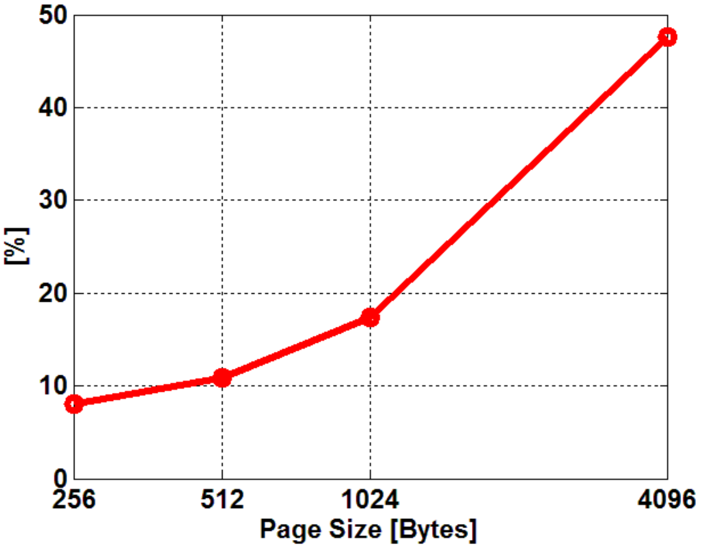


Figure 7.11: Average RRAM I/O bus interface usage as a function of the RRAM page size when a queue depth of 16 commands is fixed.

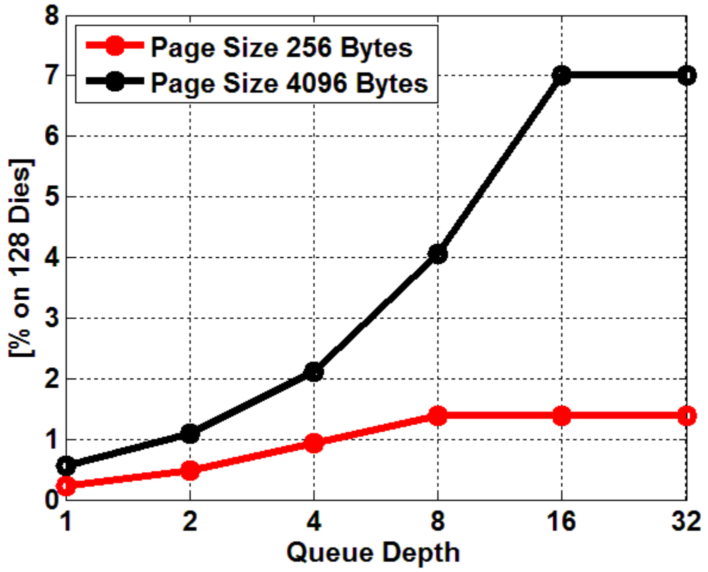


Figure 7.12: Percentage of active RRAM dies as a function of the host interface queue depth. Native 256 B and 4 kB multi-plane RRAM addressing modes are considered.

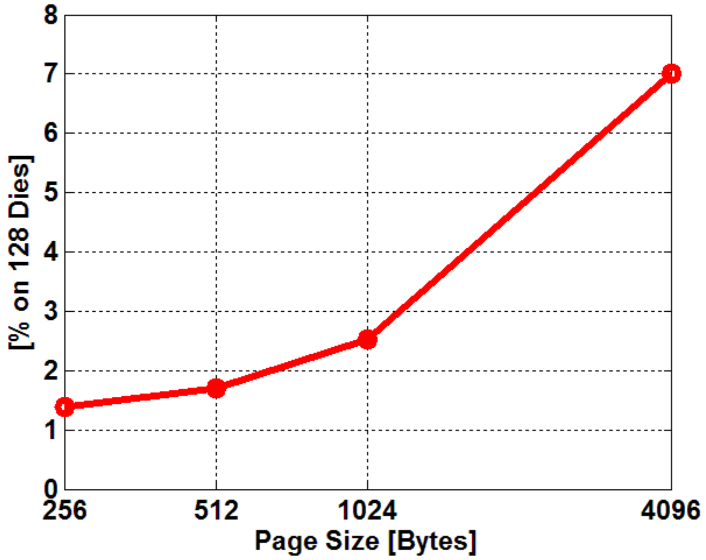


Figure 7.13: Percentage of active RRAM dies as a function of the RRAM page size when a queue depth of 16 commands is fixed.

In order to explain the above mentioned results, we observed the RRAM I/O bus interface utilization and the percentage of active RRAM dies, as a function of the RRAM page size and drive's commands queue depth. When the payload of read transactions increases and the queue depth is large enough to serve multiple commands, more data have to be transferred from the memories to the SSD controller. This condition yields to a massive overhead in terms of data transfers, thus impacting the percentage of the I/O memory bus usage. This metric rapidly grows up, reaching 48% when 4 kB transactions are served with a queue depth of 16, as shown in Figs. 7.10 and 7.11. As a consequence, the overall SSD latency is impacted and the performance advantages of RRAMs partially vanish. Another important consideration can be made by observing the average percentage of active RRAM memories under a 100% random read workload. With reference to Figs. 7.12 and 7.13, even considering 4 kB transactions, this percentage remains far below 10%. These results clearly denote a high under-utilization of SSD resources. In fact, as previously described, the analyzed All-RRAM SSD is based on a controller designed for NAND flash memories. This basic approach is cost-effective but, on the other hand, it does not permit to properly use the underlying storage medium, which is completely different from NAND.

7.3 Design space exploration of All-RRAM SSDs

Figs. 7.14 and 7.15 show a breakdown of the latency, considering a 200 MB/s DDR I/O bus frequency. It is clear that, compared to NAND flash memories, the I/O bus transfer time is the dominant factor when RRAMs are used.

Thanks to the advent of extremely fast storage media such as RRAMs, memory vendors are now investing to push the I/O frequency to 400 MHz and beyond [15].

In order to understand how next generation SSD controllers could improve performances of an All-RRAM SSD, a complete design space exploration was performed, considering a 800 MB/s I/O bus transfer rate and 5 different RRAM page sizes: 256 B, 512 B, 1 kB, 2 kB, and 4 kB. For these simulations, the RRAM characteristics summarized in Table. 7.1 were kept unaltered, and only the IO-Bus speed was increased to exploit the capabilities of the latest standard [15]. Drive bandwidth, average latency and Quality of Service (QoS) [14] were simulated for several page size configurations. The bandwidth is the average number of read commands completed in a second; the

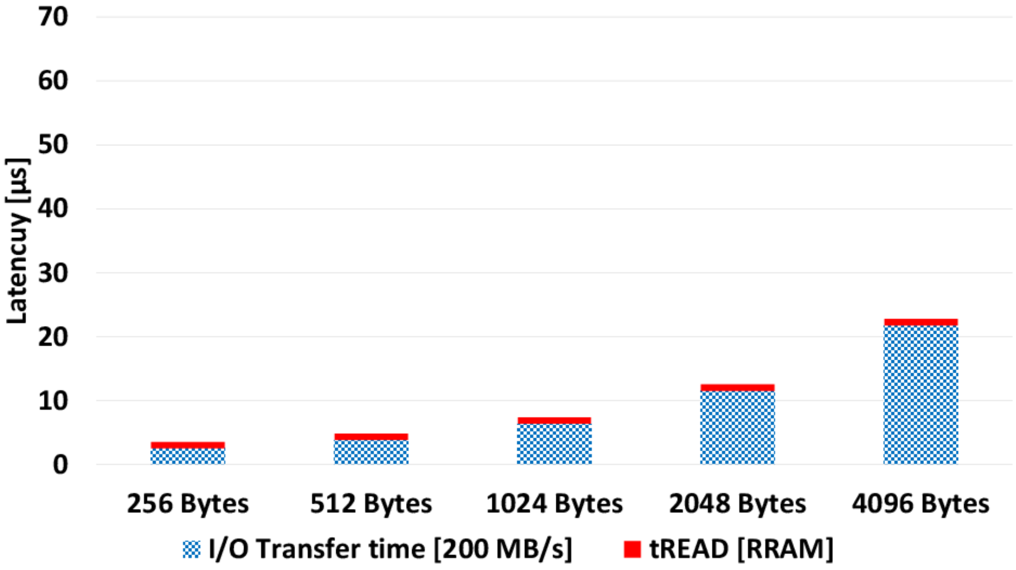


Figure 7.14: Breakdown of the storage latency when a RRAM and a 200 MB/s I/O bus are used.

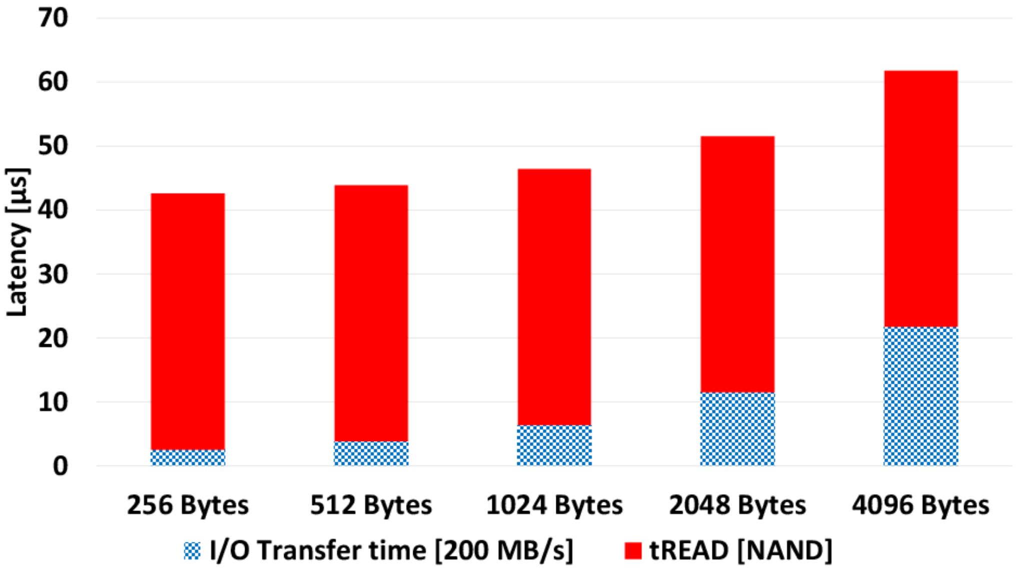


Figure 7.15: Breakdown of the storage latency when a 1X-MLC NAND flash memory and a 200 MB/s I/O bus are used.

average latency is the average time elapsed between a read command submission and its completion; the QoS is computed as the 99.99 percentile of the SSD's latency distribution. To provide a complete performance exploration of the SSD's architecture, data were collected for different host *Queue Depths* (QD), ranging from 1 to 32 commands [13].

NAND-like mode: RRAM with 4 kB page size

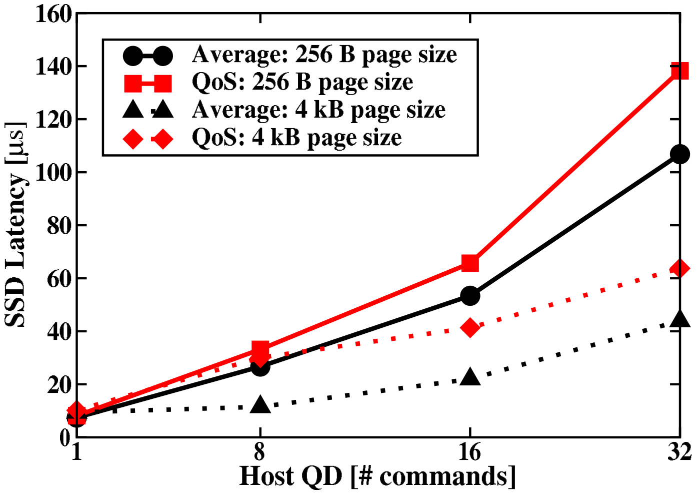


Figure 7.16: Average latency and QoS of the simulated All-RRAM SSD with page sizes of 4 kB and 256 B.

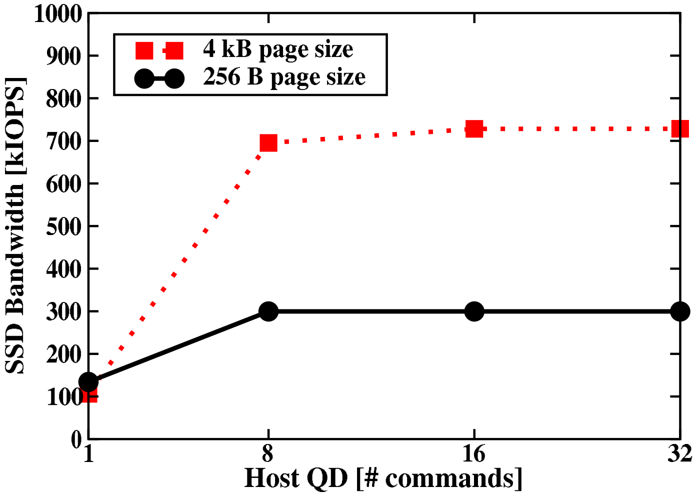


Figure 7.17: Average bandwidth of the simulated All-RRAM SSD with page sizes of 4 kB and 256 B.

This case study corresponds to the simple replacement of a NAND Flash memory with a RRAM chip in a user-transparent mode, and it will be used as a baseline for comparison. Therefore, as already presented in Section 7.2, in order to provide a full compatibility with NAND, the RRAM die must operate in 16-plane mode. Figs. 7.16 and 7.17 (dashed lines) show average latency, QoS, and bandwidth increase with respect to QD. In particular, the bandwidth saturates for a QD equal to 8 commands, showing that the SSD controller has reached its maximum performance. As sketched in Fig. 7.14, the average read latency of the SSD's storage layer depends on two factors: memory t_{READ} and data transfer time from the memory to the SSD controller. As displayed in Fig. 7.16, at QD = 1 (i.e. one command issued at a time), average latency is 9.4 μs, which is almost 4 times shorter than the one observed in Fig. 7.6. This improvement is mainly due to the faster I/O bus transfer time (i.e., the 800 MB/s memory interface). In fact, in this case the transfer of a 4 kB page takes only 6 μs instead of the 21 μs taken by the legacy 200 MB/s interface. To be fair, it must be highlighted that, compared to the memory t_{READ} time, the I/O bus transfer contribution still dominates the overall SSD's latency.

Although the achieved latency is far below the typical values of NAND-based SSDs [8], RRAMs can be further optimized to reach even higher performances. For example, one area of improvement is the partitioning of the 4 kB transactions coming from the host into smaller chunks. The goal of this approach is to reduce both the data transfer time by selecting the right number of planes to be simultaneously accessed (i.e. optimal memory page size), and the number of internal read commands to be handled by the SSD firmware.

Single-plane RRAM

The minimum read granularity allowed by RRAMs is a single plane 256 B page read operation, which could potentially reduce both the transfer time and the SSD latency. However, as shown in Fig. 7.18, since the host works with 4 kB transactions, it is necessary to split the host operations in 16 chunks of 256 B each. The firmware running inside the SSD performs this operation: by using a 16 channels architecture and DRAM buffers, firmware reads in parallel all the addressed 256 B chunks, and rebuilds the 4 kB transaction before sending the data back to the host. Figs. 7.16 and 7.17 (solid lines) show bandwidth, average latency and QoS achieved by the

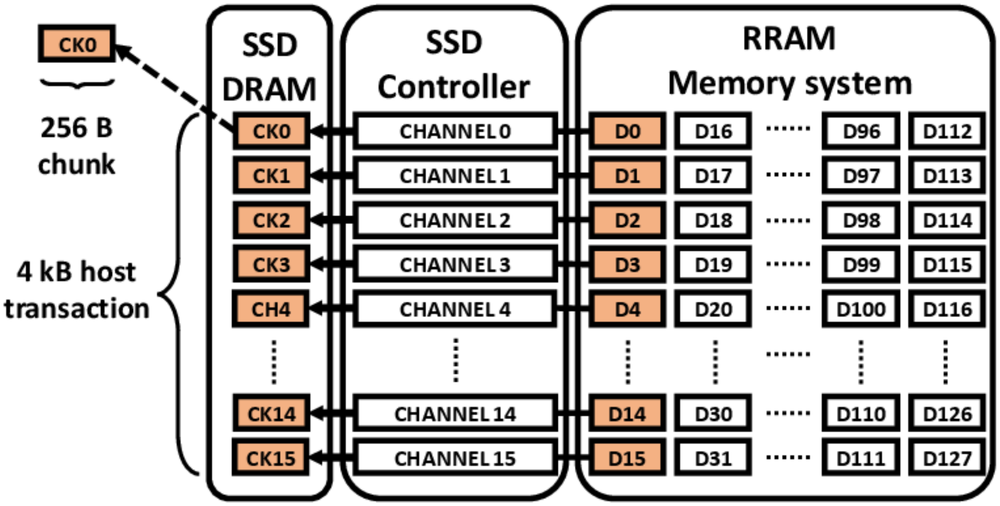


Figure 7.18: A single 4 kB host transaction is split across multiple memory channels.

aforementioned approach. Looking at the results of the simulations performed with QD = 1, the straightforward conclusion would be that the 256 B page size reduces SSD latency, increases the bandwidth, and improves the QoS. However, for QD > 1 these considerations do not hold true anymore, since the number of operations internally handled by the SSD controller increases by a factor 16, leading to a saturation of its processing capabilities. This turns into a dramatic performance degradation, also considering that all data chunks must be temporarily stored inside the DRAM buffer, whose access is contended by all the SSD channels, thus causing resources starvation.

Multi-plane RRAM

To reduce both the amount of commands processed by the SSD controller and the number of accesses to the internal DRAM , different RRAM page sizes were considered: 512 B, 1 kB, and 2 kB. To keep the payload of the 4 kB host transactions constant, SSD's firmware and RRAM memories were co-designed to work in multiplane mode. In other words, when a n * 256 B RRAM page size is used, being $n = [2, 4, 8]$, the SSD's firmware is configured to read $16/n$ chunks of n * 256 B each from $16/n$ parallel channels. Fig. 7.19 shows the cumulative latency distributions and the QoS of the simulated All-RRAM SSD as a function of the page size, when a host QD = 1 is selected. The optimal drive latency is achieved neither with the

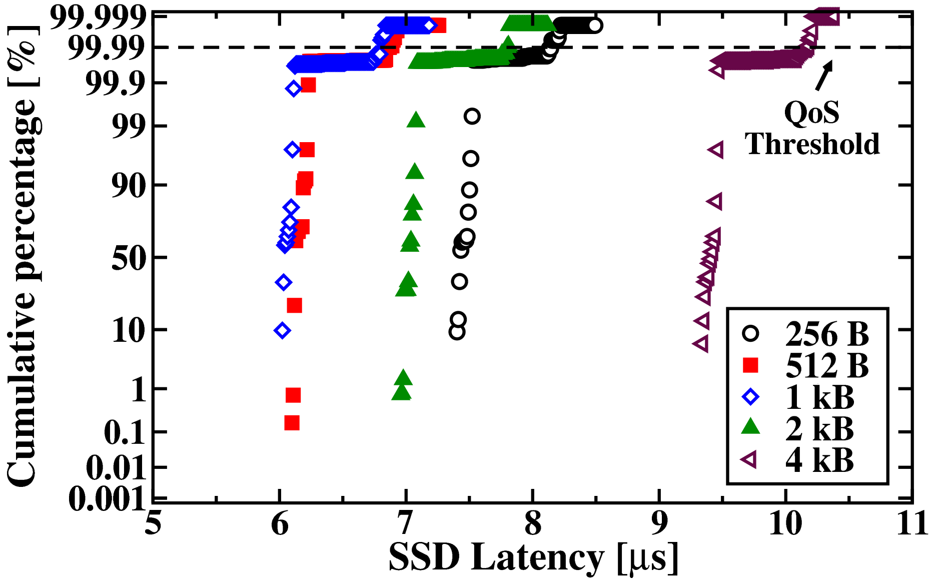
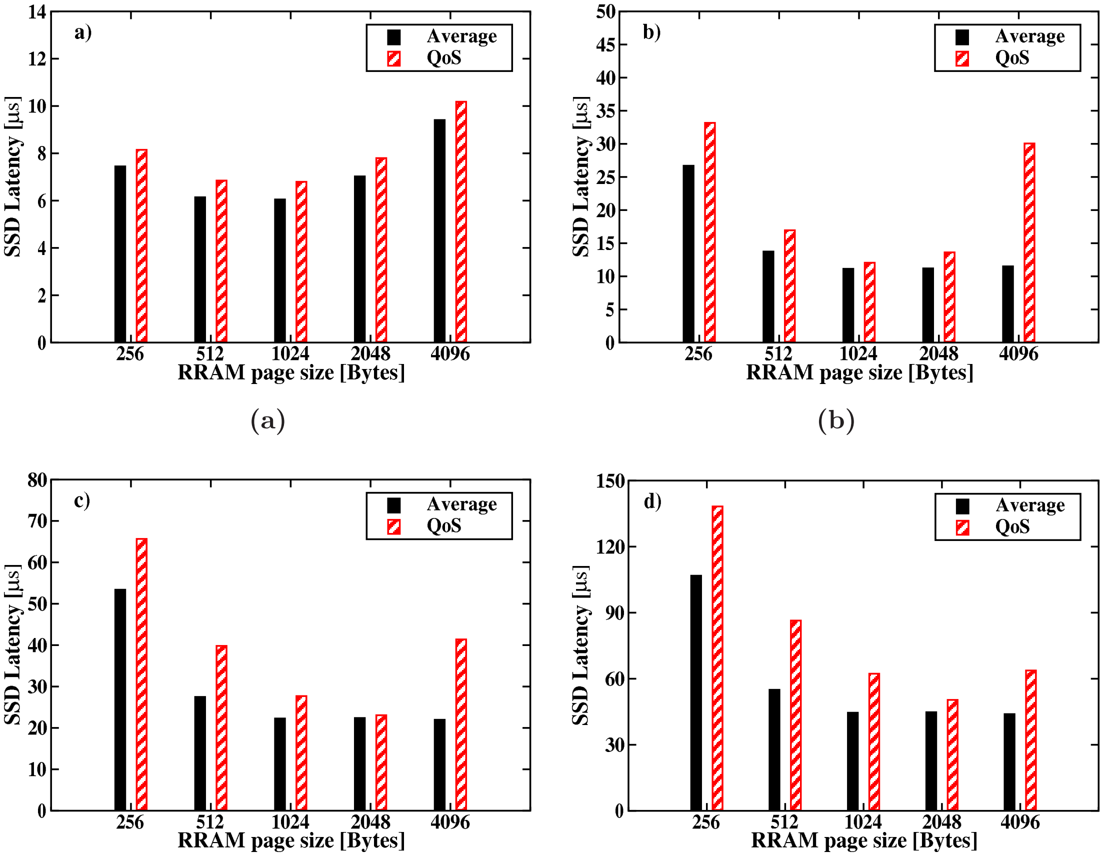


Figure 7.19: Normal probability paper of the latency of the simulated All-RRAM SSD for QD = 1 and different page sizes.

standard 256 B page size nor with the 4 kB NAND-like mode, but rather with a 1 kB multi-plane page configuration. Same considerations apply to other host QD values, as shown in Figs. 7.20 a-d. there are two hot spots for the page size: when QD < 8 the value is 1 kB, whereas it becomes 2 kB for QD > 8. Fig. 7.21 shows the bandwidth achieved by the All-RRAM SSD as a function of both the RRAM page size and the host QD. When QD $= 1$, the maximum bandwidth is with a page size of 1 kB; for QD > 16 the maximum bandwidth is with 4 kB.

These results proved that emerging memories such RRAMs have to be wisely designed when they are used as the main storage media in SSDs. In this regard, replacing NAND flash memories with RRAMs in a “plug and play” fashion is not the best way to reach high performances and low-latency. Moreover, even in the best working conditions (i.e., co-designing the memories characteristics together with the whole SSD architecture), it has been shown that the optimum design point of the All-RRAM SSD is still affected by the host configuration and its requirements. This is in agreement with today’s trend of developing specific SSD architectures for specific host applications [16]; the downside of this approach is that it leads to extremely complex SSD designs with hundreds of parameters to explore. SSDExplorer can definitely help to address the above mentioned problems, allowing a better understanding of where, in All-RRAM SSDs, the co-design activity is more effective.



(a) (b)

(c) (d)

Figure 7.20: SSD average latency and QoS for host QD = 1 (a), QD= 8 (b), QD =16 (c), and QD = 32 (d).

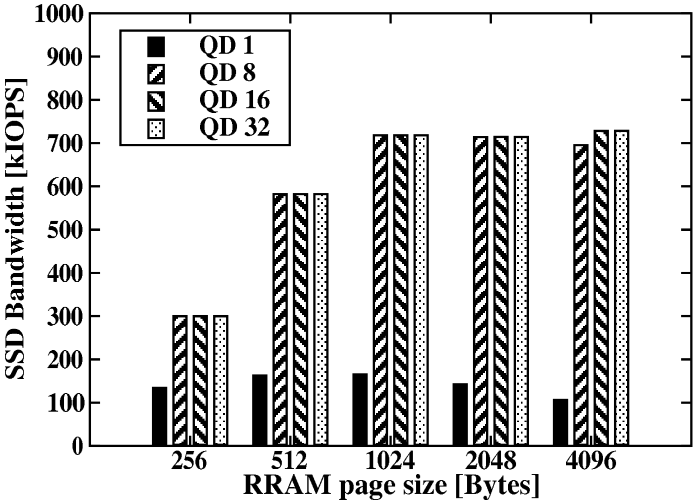


Figure 7.21: SSD's bandwidth as a function of the RRAM page size and the host QD.

References

[1] R. Micheloni, A. Marelli, and K. Eshghi. *Inside Solid State Drives (SSDs)*. Springer, 2012.

[2] L. Zuolo, C. Zambelli, R. Micheloni, D. Bertozzi, and P. Olivo. Analysis of reliability/performance trade-off in solid state drives. In *IEEE International Reliability Physics Symposium*, pages 4B.3.1–4B.3.5, June 2014.

[3] E.I. Vatajelu, H. Aziza, and C. Zambelli. Nonvolatile memories: Present and future challenges. In *International Design Test Symposium (IDT)*, pages 61–66, Dec. 2014.

[4] C. Zambelli, A. Grossi, D. Walczyk, T. Bertaud, B. Tillack, T. Schroeder, V. Stikanov, P. Olivo, and C. Walczyk. Statistical analysis of resistive switching characteristics in ReRAM test arrays. In *IEEE Int. Conf. on Microelectronics Test Structures (ICMTS)*, pages 27–31, Mar. 2014.

[5] X. Y. Xue, W. X. Jian, J. G. Yang, F. J. Xiao, G. Chen, X. L. Xu, Y. F. Xie, Y. Y. Lin, R. Huang, Q. T. Zhou, and J. G. Wu. A 0.13 μm 8mb logic based cuxsiyo resistive memory with self-adaptive yield enhancement and operation power reduction. In *Symposium on VLSI Circuits (VLSIC)*, pages 42–43, June 2012.

[6] K. Takeuchi. Hybrid solid-state storage system with storage class memory and nand flash memory for big-data application. In *IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1046–1049, Jun. 2014.

[7] S.H. Jo. Recent progress in rram materials and devices. In *SEMICON Korea*, 2015.

[8] S. Dubois. Crossbar Resistive RAM (RRAM): The Future Technology for Data Storage. In *SNIA Data Storage Innovation Conference*, Apr. 2014.

[9] S Bates, M Asnaashari, and L. Zuolo. Modelling a High-Performance NVMe SSD constructed from ReRAM. In *Proc. of Flash Memory Summit*, Aug. 2015.

[10] L. Zuolo, C. Zambelli, R. Micheloni, M. Indaco, S. Di Carlo, P. Prinetto, D. Bertozzi, and P. Olivo. Ssdexplorer: A virtual platform for performance/reliability-oriented fine-grained design space exploration of solid state drives. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(10):1627–1638, 2015.

[11] Open Nand Flash Interface (ONFI). `http://www.onfi.org`.

[12] Ha Ryong (Harry) Yoon. Toggle-Mode NAND to Fill Growing Need for Higher Performance. In *Proc. of Flash Memory Summit*, Aug. 2009.

[13] Nvm express 1.1 specification, 2013. `http://nvmexpress.org/wp-content/uploads/2013/05/NVM_Express_1_1.pdf`.

[14] Intel solid-state drive dc s3700 series – quality of service., 2013. `http://www.intel.com/content/www/us/en/solid-state-drives/ssd-dc-s3700-quality-service-tech-brief.html`.

[15] Open Nand Flash Interface (ONFI) revision 4.0. `www.onfi.org/~/media/onfi/specs/onfi_4_0-gold.pdf?la=en`.

[16] Jian Ouyang, Shiding Lin, Song Jiang, Zhenyu Hou, Yong Wang, and Yuanzheng Wang. Sdf: Software-defined flash for web-scale internet storage systems. In *Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS, pages 471–484, 2014.

Chapter 8

MRAM-based NVRAM cards

Non-Volatile RAM (NVRAM) cards address the need of persistency for small, frequent and/or transient data. For example, they can be used as a very fast and secure synchronous write buffer, which can quickly acknowledge synchronous writes, without compromising data integrity/persistency. Another typical use case for NVRAM cards is the storage of critical system data and user data in case of Power Failure: amongst the others, File System Metadata, Transaction Logs, and Cache Index Tables.

The most common architecture of existing NVRAM cards is shown in Fig. 8.1a [1]. The card is connected to the Host via a PCIe interface and it contains DRAM and NAND Flash memories [2], together with a controller and a super capacitor. DRAM is used as a primary storage, in the sense that it is directly exposed to the Host via PCIe.

On the contrary, NAND Flash memories are hidden to the Host and they are activated only in case of abrupt power down. When this happens, the on-board controller copies the entire DRAM content into the Flash array, thus making data non-volatile. Given the fact that the external power supply has been shut down, the energy required for reading from the DRAM and writing to the NAND is supplied by a super capacitor. As a matter of fact, NAND Flash devices, most part of the controller, and the super capacitor are there only because DRAMs are volatile, and all these additional components have an impact on cost, power and reliability. Therefore, there is a continuous research for a non-volatile technology that can simplify the overall design of NVRAM cards.

8.1 All MRAM NVRAM cards

Given the most recent advancements [3], MRAM technology seems to be a technically viable alternative for building simplified NVRAM cards, which would look like Fig. 8.1b. Because high performance is the main value proposition of NVRAM cards, it is key to understand how the number of read/write random IOPS and latency figures (i.e QoS, Quality of Service) would be impacted by replacing DRAM with MRAM. In this work we present a detailed IOPS/QoS analysis based on a commercial DRAM/Flash NVRAM card [1] combined with a 256 Mbit perpendicular Spin-Transfer-Torque (STT) MRAM [4].

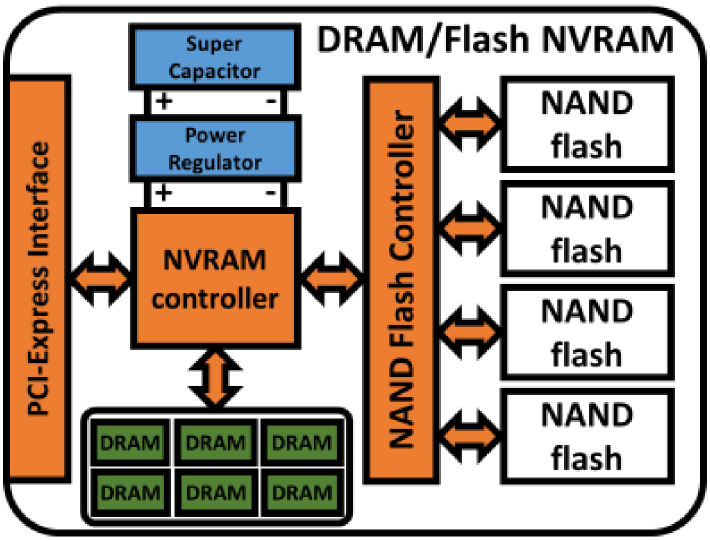


Figure 8.1a. Block diagram of a DRAM/Flash NVRAM card

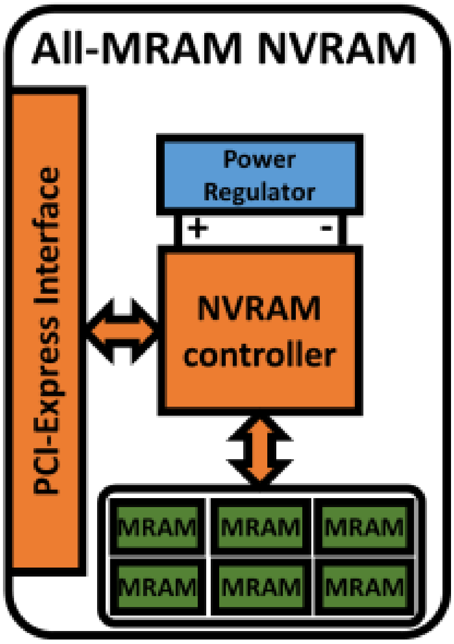


Figure 8.1b. Block diagram of an All-MRAM card

To get started, we successfully proved the interoperability between the on-board controller and the selected MRAM device, over the DDR3 bus. Figure 8.2 shows the validation board of the on-board controller that was used for this test: MRAM devices fully populate a UDIMM which is vertically mounted on the validation board.

A custom GUI, shown in Fig. 8.3, was also developed to check interoperability under different workload conditions (i.e. different combinations of read/write operations).

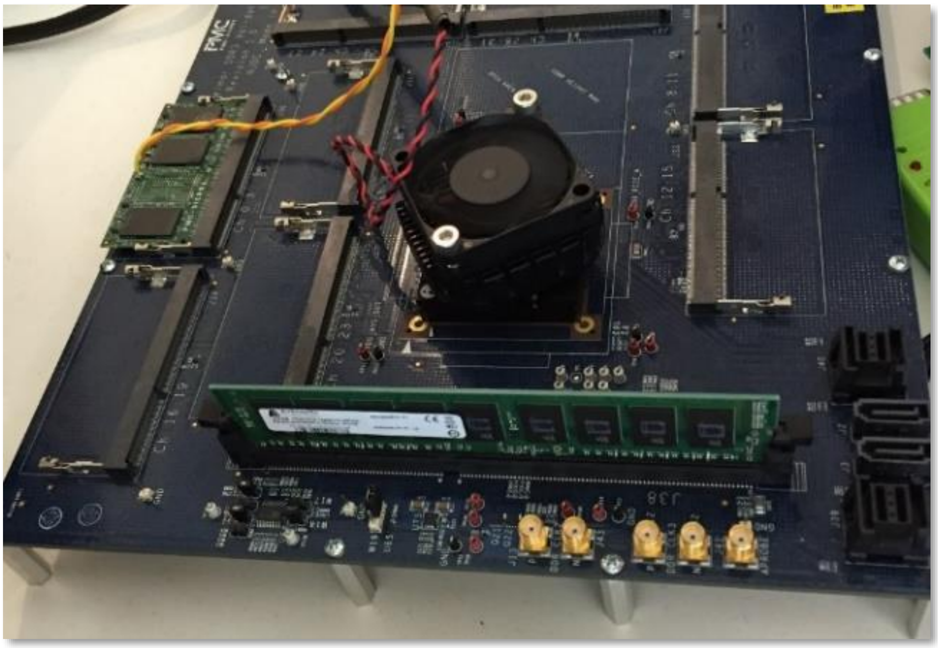
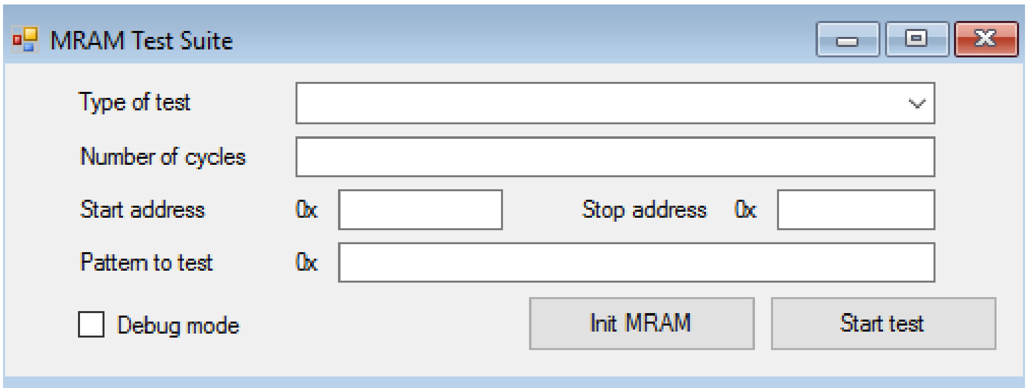


Figure 8.2. SSD controller evaluation board used for interoperability tests. Eight standard SO-DIMM sockets can accommodate NAND flash memory cards, while one standard DDR UDIMM socket can host either DRAMs or MRAM memories.



MRAM Test Suite

Type of test

Number of cycles

Start address 0x

Stop address 0x

Pattern to test 0x

☐ Debug mode

Figure 8.3. Custom developed Graphical User Interface (GUI) for MRAM testing. This GUI helps programming the SSD controller to issue single/multiple read/write operations to/from the MRAM DIMM.

8.2 Data correlation and Simulation Framework

Besides the experimental set up of Fig. 8.2 we wanted to develop a simulation platform to enable the design of the new generation NVRAM cards based on the All-MRAM approach. The selected simulation framework is SSDExplorer [5] because it is a Fine-Grained Design Space Exploration (FGDSE) tool, well suited for evaluating the impact of micro-architectural design choices on performances.

Figure 8.4 sketches the architecture of the adopted simulation framework, while Table 8.1 summarizes the main characteristics of the simulated host system and NVRAM cards.

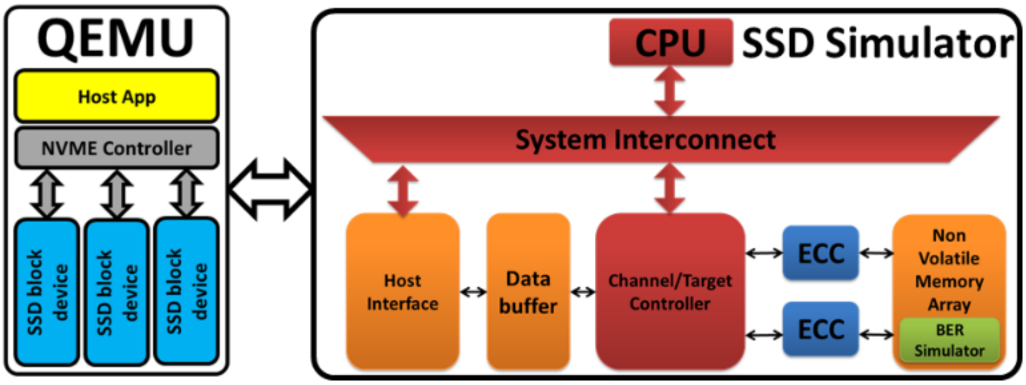


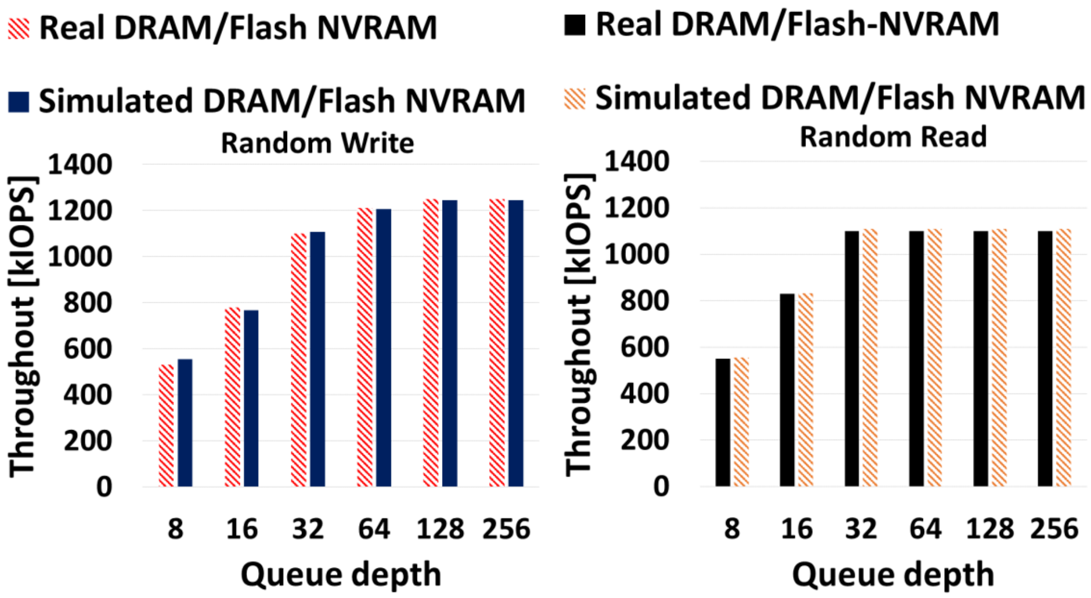
Figure 8.4. Architecture of the simulation framework used to test the performance and the latency of both the DRAM/Flash and the All-MRAM NVRAM cards. The parameters of the SSD simulator can be tuned to simulate a wide variety of SSD architectures and memories. Qemu is used as a workload generator.

Table 8.1. Main characteristics of the host system and the simulated NVRAM cards

NVRAM card parameter	**Configuration**
Host Interface	PCI-Express Gen3 x8
Host protocol	NVMe 1.1
DRAM/MRAM size	1 GByte
DRAM/MRAM controller	Single channel
Host System	**Configuration**
Intel S2600GZ server	Dual Xeon E5-2680 v2 128 GBytes DRAM

Figure 8.5 shows the comparison between actual performances [6] and SSDExplorer simulation results in terms of IOPS: there is a great matching for both Random Read and

Random Write workloads. It is worth highlighting how the matching is consistent across different queue depths.



(a) (b)

Figure 8.5. Performance comparison between a real (red dashed columns) and a simulated (blue solid columns) DRAM/Flash-based card when 100% 4 kBytes random write (a) and 100% 4 kBytes random read (b) are considered, respectively. Different host interface queue depths are considered.

8.3 DRAM/Flash-based NVRAM vs. All-MRAM NVRAM

Figure 8.6 shows a direct IOPS comparison between the 2 architectures described in Fig. 8.1.

Also in this case we considered random write and random read workloads and different queue depths. As a matter of fact, the All-MRAM architecture can keep up with the requested number of transactions without any significant performance degradation. While being counter-intuitive, this result can be explained by noting that the simulated NVRAM architecture sits beyond a PCIe interface, which turns out to be the bottleneck of the system.

Last but not least, let’s take a look at latencies for both random read and write workloads. Indeed, these latencies are becoming more and more critical, especially when looking at applications where fast response is critical, such as financial trading and ecommerce. Delays introduced by host systems could mask actual card's performance; therefore, latency has been evaluated as the raw latency introduced by the card only.

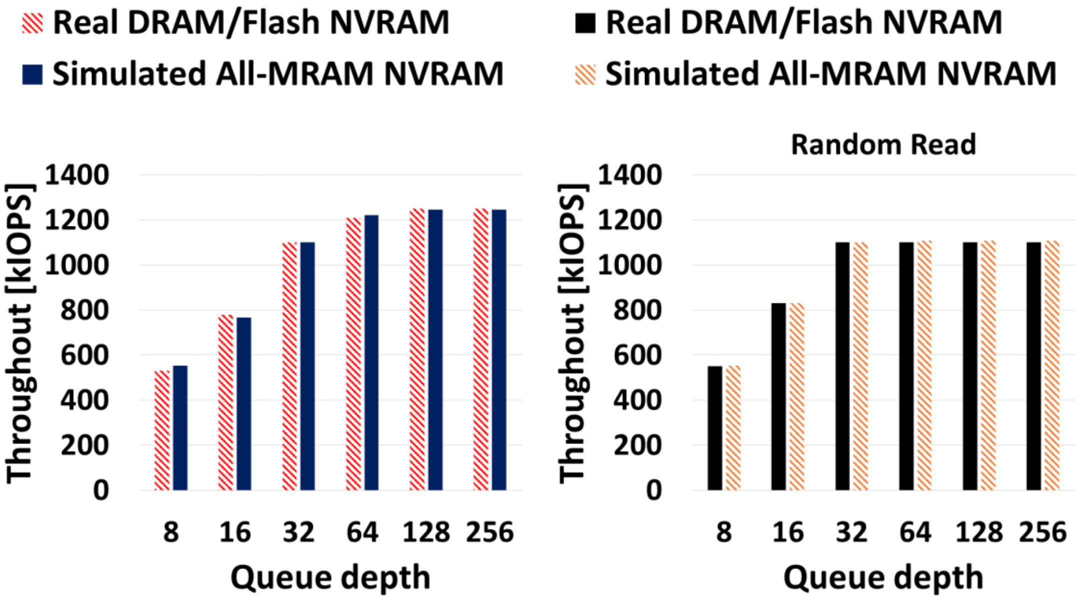


Figure 8.6. Performance comparison between a real DRAM/Flash-based card (red dashed columns) and a simulated All-MRAM card (blue solid columns) when 100% 4 kBytes random write (a) and 100% 4 kBytes random read (b) are considered, respectively. Different host queue depths are considered.

Fig. 8.7 and Fig. 8.8 show the Cumulative Distribution Function (CDF) of the latencies for queue depths of 8 and 256, under 100% 4 kByte random write and 100% 4 kByte random read workloads, respectively.

Similarly, to what we have seen for IOPS, All-MRAM NVRAM cards can respond as quickly as legacy cards, and this is true also when looking at the upper part of the CDF. It is worth highlighting that small differences in the latency profile are negligible when adding the overhead introduced by the application software.

In this chapter, we have shown that MRAM is a viable alternative for replacing DRAM inside NVRAM cards. Number of IOPS and latency figures have been extensively analyzed under different workload conditions and queue depths. In all cases, we haven't detected any significant performance degradation with respect to the DRAM/Flash legacy solutions. MRAM-based architectures can definitely simplify the card design by removing the need for Flash memories and the super-capacitor. Looking forward, the overall cost and power of the NVRAM card need to be assessed, especially considering that MRAM density is expected to reach 4Gbit/die in the coming years.

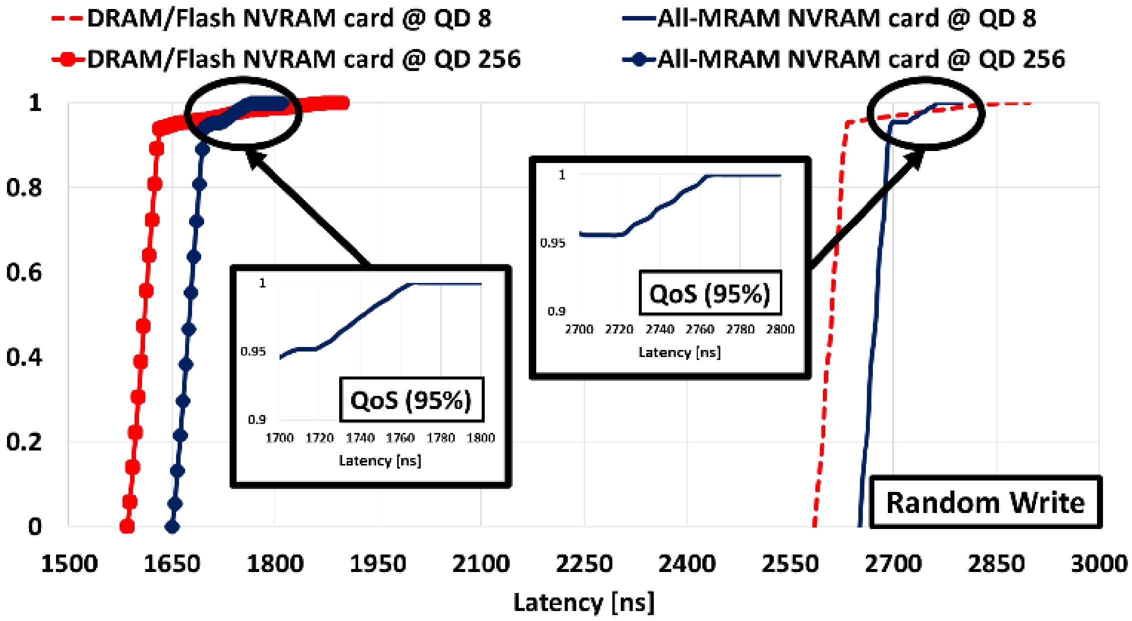


Figure 8.7. Latency cumulative distribution functions of the NVRAM cards when the standard DRAM/Flash architecture and the All-MRAM configurations are considered, respectively. A queue depth (QD) of 8 and 256 commands have been used. Simulated workload is 100% 4 kByte random write.

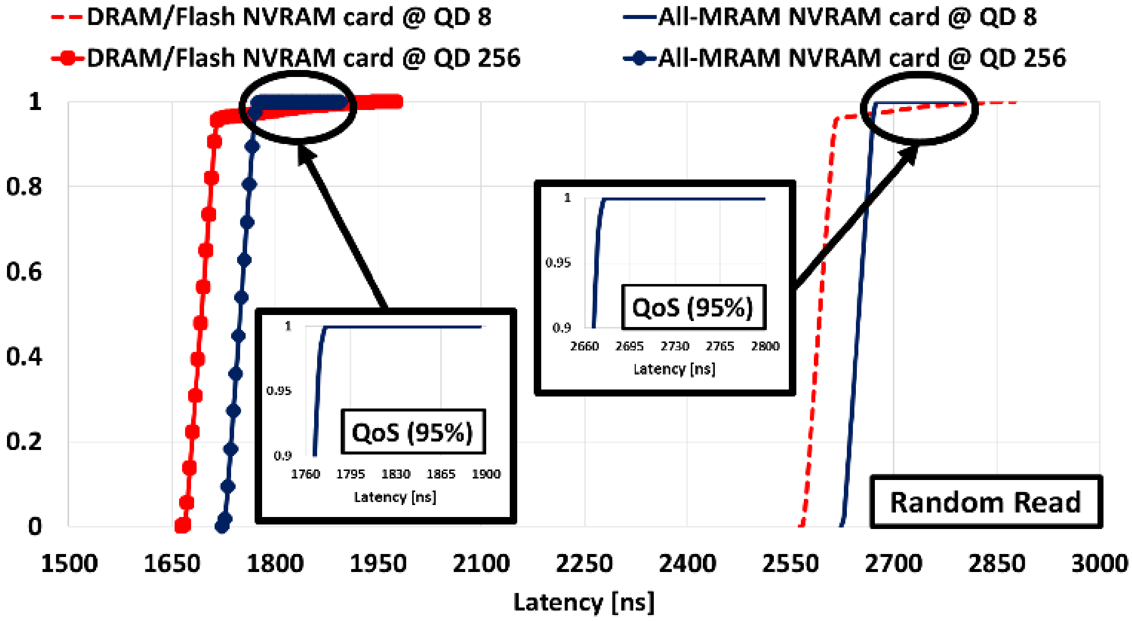


Figure 8.8. Latency cumulative distribution functions of the NVRAM cards when the standard DRAM/Flash architecture and the All-MRAM configurations are considered, respectively. A queue depth (QD) of 8 and 256 commands have been used. Simulated workload is 100% 4 kByte random read.

Bibliography

[1] FlashTec NVRAM Drives, Microsemi. 2016. [Online]. Available: http://www.microsemi.com/products/storage/flashtec-nvram-drives/flashtec-nvram-drives

[2] R. Micheloni et al., Inside NAND Flash Memories, Springer, 2010.

[3] D. Apalkov et al., Magnetoresistive Random Access Memory, Proceedings of the IEEE, pp. 1796-1830, Vol. 10, October 2016.

[4] Everspin 256 Mb STT-MRAM. 2016. [Online]. Available: https://www.everspin.com/file/965/download

[5] L. Zuolo et al., IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2015. Vol. 43, no. 10, pp. 1627-1638.

[6] PMC-Sierra Flashtec NVRAM Drive Review. [Online]. Available: http://www.tomsitpro.com/articles/pmc-sierra-flashtec-nvram-drive,2-954.html

Chapter 9

Modeling the reliability of Intra-Disk RAID solutions for SSDs

Solid State Drives (SSD) in enterprise and data center environment are calling for denser non-volatile memories to increase the storage capacity. This request is fulfilled by Triple Level Cell (TLC) NAND Flash memories, that offer a reasonable cost-per-bit feature while enabling high storage density for SSD applications [1]. This technology, compared either to Multi Level Cell (MLC) or Single Level Cell (SLC) paradigms, comes with evident reliability and performance drawbacks that must be handled by complex SSD controllers to ensure data protection and a reasonable Quality of Service (QoS) [2].

The state-of-the-art in SSD data protection is to integrate an Error Correction Code (ECC) engine in the SSD controller to handle the NAND Flash bit errors occurring throughout the entire lifetime of the disk, trying to marginally impact the disk performance [3–5]. The ECCs are usually designed to provide an Uncorrectable Bit Error Rate (UBER) lower than 10^{-16} to consistently cope with their mission in a time lapse of at least five years [6]. This latter metric is generally used to qualify an SSD to be ”safe” against data corruption [7]. However, as the scaling of NAND Flash proceeds in the deca-nanometer range involving many reliability threats, the ECC might be insufficient to provide that sought protection [8–12].

Furthermore, as the memory devices become more and more complex by integrating structures for optimal data handling and performance guard-banding [13, 14], their defectivity rate increases at a fast pace. In both cases the SSD UBER specifications are jeopardized. Recently, an additional level of error correction in multiple SSDs was proposed by

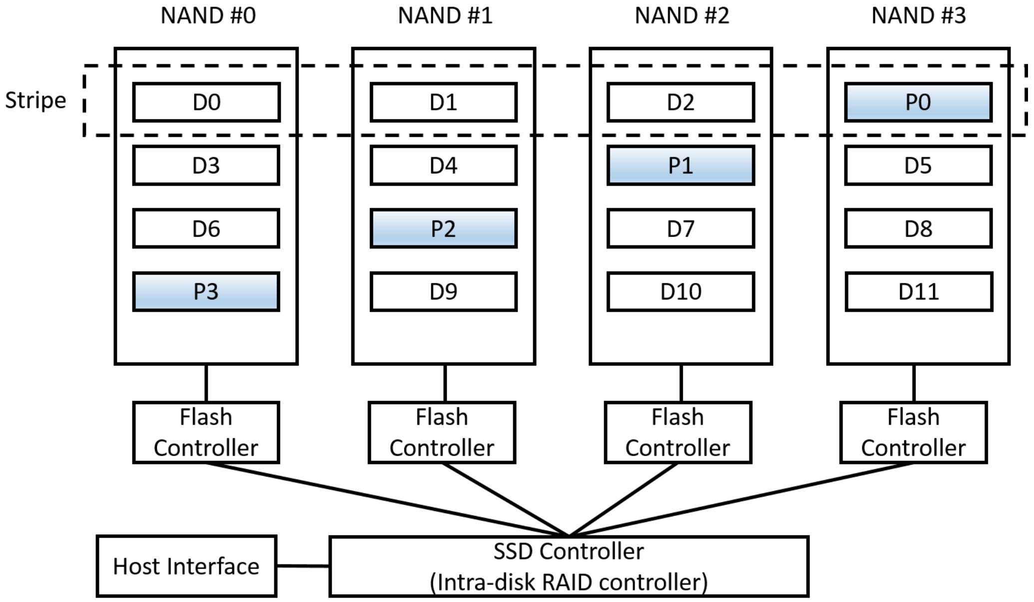


Figure 9.1: Architecture example of an intra-disk RAID-5 approach for SSD based on four NAND Flash devices.

exploiting the Redundant Array of Independent Disks (RAID) concept. Many studies in literature tackled the analysis of this solution in terms of reliability by proving its effectiveness in data protection, although pointing out the performance drawbacks due to the long RAID rebuild times [15–17]. The application of intra-disk RAID techniques then came into play [18–21]. Typical SSD already employ the RAID0 architecture as they stripe data over multiple NAND Flash, while relying on ECC for error recovery. By transforming RAID-0 SSD to higher RAID level SSDs can greatly enhances error recovery while limiting the performance drawbacks.

In this work, we analyze two different intra-disk RAID techniques on mid-1X TLC NAND Flash SSD for enterprise applications, namely the RAID-5 and RAID-6. Their reliability is evaluated in terms of UBER through a parametric analysis as a function of the ECC correction strength applied by the SSD controller, stripe size, disk and workload constraints. Moreover, a Discrete Time Markov chain (DTMC) model is developed to provide an estimate of the data loss probability (PDL) when die-level or channel-level failures are considered, thus providing awareness of the achievable reliability to the SSD users.

9.1 Intra-disk RAID basic principles

The architecture of an SSD that supports intra-disk RAID is shown in Fig. 9.1. In this example we assume that the SSD is composed by four NAND Flash chips, each one managed by its own Flash controller constituting an entity called *channel*. An SSD controller will supervise the data transfers from/to the host interface and will manage the data recovery procedure in case of a fault. Generally, in RAID architectures the data are arranged following a specific pattern that mixes data and parity, where the latter represents the additional data required for the SSD to recover any of the actual blocks of stored information. The minimum granularity for data in RAID is called a *sector*. In the SSD scenario this is related to the minimum amount of data that can be accessed on a NAND Flash, usually representing a page or a portion of thereof. A *stripe* is an ensemble of data and parity sectors representing the minimum unit for data reconstruction [22]. The stripe length expresses how many user data elements are associated with parity elements. In case of intra-disk RAID-5, as shown in Fig. 9.1, the stripe length refers to a single parity element in a stripe (i.e., the notation N-to-1 is also used), whereas in intra-disk RAID-6 refers to double parity elements in a stripe (i.e., N-to-2). The choice of the stripe length and of the RAID level depends on a reliability/performance trade-off that an SSD wants to leverage on [16]. In this paper we assume that the maximum stripe length is equal to the number of channels present in the SSD. Let us assume the 3-to-1 intra-disk RAID-5 configuration shown in Fig. 9.1 by considering the $(D0; D1; D2; P0)$ stripe. If one of the data sectors Di with $i = 0..2$ in the stripe fails either due to unexpected NAND Flash/channel failures or due to the impossibility to correct the data using the ECC, the intra-disk RAID recovers the faulty sector via the parity $P0$ by applying a XOR algorithm [18]. The recovered data are then written on another sector of the SSD and the faulty one is marked as invalid and retired process handled by the SSD controller. This recovery comes with additional data movement in the SSD that causes write amplification, therefore both the stripe length and the data movement strategy after recovery should be carefully planned in advance. Similar considerations apply to intra-disk RAID-6 except that in this case the fault tolerance is increased up to two data sectors.

9.2 Uncorrectable Bit Error Rate ideal calculations using intra-disk RAID

The reliability metric used in NAND Flash memories to design both the ECC and the additional error recovery options like the intra-disk RAID is the Raw Bit Error Rate (RBER) [23]. The RBER represents the probability to observe an erroneous bit before applying the ECC, and is usually calculated on a sector basis. This value is used to define the UBER after the correction procedures. The failure modes that contributes to the RBER in a NAND Flash are: endurance errors, retention errors, and read disturb errors. Since in this work we focus the analysis on the lifetime reliability of the SSD we consider only the endurance errors, that represent the effect of the lifetime degradation on the NAND Flash memories. Other error sources impact can be derived by this starting point.

Fig. 9.2 shows the RBER retrieved from different 4 KB sectors of several 16 KB pages of a mid-1X TLC NAND Flash during an endurance experiment to check its evolution over the memory lifetime. The test equipment was the same described in [24] and was used to stress the memory with random data patterns to emulate real cycling conditions as indicated in [25]. The cycling was performed at a 55°C temperature according to [6, 25], which represents the typical condition for an enterprise environment. All TLC page types (i.e., lower bits, center bits, and upper bits) have been considered in the analysis to increase the statistical consistency.

9.2.1 Impact of the ECC correction strength and stripe length

To calculate the UBER from the RBER data as a function of the intra-disk RAID approach it is mandatory to derive the so-called codeword failure probability (i.e., F_{CW}). The equations presented in this section to this purpose basically derive from the binomial probability distribution. If there are n independent trials of an experiment that can have either an outcome 1 with probability p or an outcome 2 with probability $(1-p)$, the probability to have k outcomes of type 1 is given by $\binom{n}{k}p^k * (1-p)^{n-k}$.

In NAND Flash the ECC supplements the user data in a sector with some parity bytes to reconstruct the information in case of bit corruption [23]. The sum of the user data plus the parity constitutes a codeword, that for the tested memory is equal to 4320 bytes (4096 bytes for data plus 224 bytes for parity). Given an ECC that can correct up to k bits in a

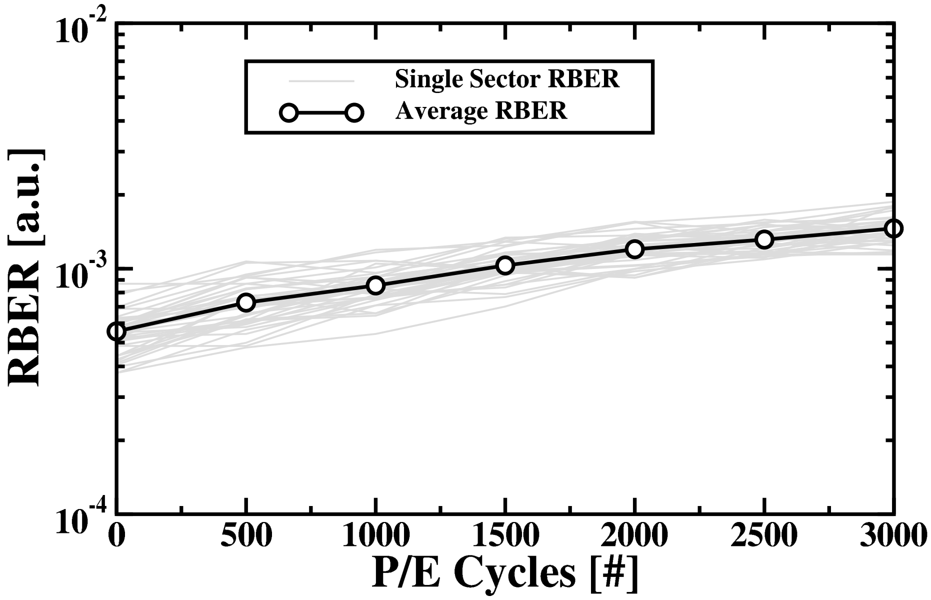


Figure 9.2: Worst RBER characteristics of different 4 KB sectors of a mid-1X TLC NAND Flash as a function of the endurance. The average value is reported for further analysis and discussions.

codeword we can calculate the codeword failure probability as [23]:

$$F_{CW} = 1 - \sum_{i=0}^{k} \binom{n}{i} \cdot RBER^i \cdot (1 - RBER)^{n-i} \tag{9.1}$$

where n is the number of bits in the codeword. By considering an intra-disk RAID-5 there are two specifical cases where errors can be corrected in a stripe: *i)* all the sectors in a stripe with length N have less than or equal to k errors; *ii)* $N-1$ sectors in a stripe have less than or equal to k errors and one sector has more than k errors. We can write the Correctable Stripe Error Rate ($CSER_{R5}$) as:

$$\begin{aligned} CSER_{R5}(N) = (1 - F_{CW})^N + N(1 - F_{CW})^{N-1} \\ \times \sum_{i=k+1}^{n} \binom{n}{i} RBER^i (1 - RBER)^{n-i} \end{aligned} \tag{9.2}$$

The intra-disk RAID-6 tolerates an additional sector failure with respect to RAID-5, therefore the $CSER_{R6}$ equation must include eq.(9.2) summed with an additional term that consider the occurrence of $N-2$ sectors in a stripe that have less than or equal to k errors and two sectors have more than k errors.

$$CSER_{R6}(N) = CSER_{R5}(N) + \binom{N}{2}(1 - F_{CW})^{N-2} \times \left[\sum_{i=k+1}^{n} \binom{n}{i} RBER^{i}(1 - RBER)^{n-i}\right]^{2} \tag{9.3}$$

The UBER for RAID-5 and RAID-6 is then derived as:

$$UBER = \frac{1 - CSER(N)}{n_{data} \cdot N} \begin{cases} CSER = CSER_{R5} & \text{RAID-5} \\ CSER = CSER_{R6} & \text{RAID-6} \end{cases} \tag{9.4}$$

where n_{data} is the number of user data bits in the codeword (i.e., 4096 bytes). Eqs. (9.2) and (9.3) base on the assumption that all the sectors within a stripe come from the same RBER distribution. Such a statement is equivalent to assume that each of the N NAND Flash dies in an SSD feature the same RBER, therefore negleting the die-to-die variations. Although this is not a realistic condition, in this paper we consider that the RBER NAND Flash characteristics as those shown in Fig. 9.2 comes from the worst NAND Flash die (i.e., the die with highest RBER). In this case, all the considerations on the intra-disk RAID are tuned to the worst-case condition, that usually represents the reliability reference point for the design of secondary error correction mechanisms.

Figs. 9.3 and 9.4 shows the UBER of both intra-disk RAID-5 an RAID-6 as a function of the ECC correction capability and of the stripe length. In the calculations we consider correction strengths of 40, 60, 80, and 100 bits on a 4320B codeword, and stripe lengths equal to 8, 16, 32, and 64, respectively. By considering the enterprise SSD qualification specifications provided by [6], it is possible to appreciate that the 10^{-16} target is quite easy to reach even with poor ECC strength (i.e., 60 bits), proving superior reliability for RAID-6 as expected from theoretical foundations [22]. However, it must be reminded that such a target represents ideal calculations that must be complemented with additional thoughts shown in the forthcoming sections of this work. A difference in the UBER up to 25 orders of magnitude can be appreciated for a 100 bits ECC by considering the lower bound of the NAND Flash RBER region for a mid-1X TLC Flash that is $5 * 10^{-4}$. The stripe length has a minor impact on the UBER compared with that of the ECC correction capability, although this effect will turn to have a dramatic impact when complete SSD simulations are performed, as shown in the next sections of the paper.

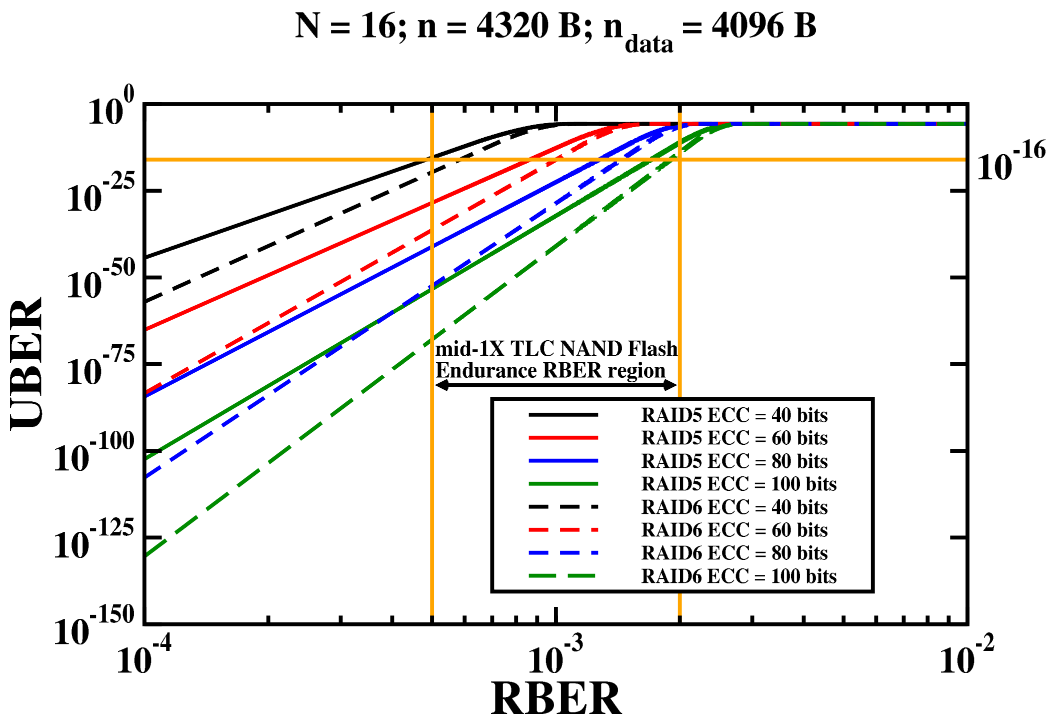


Figure 9.3: UBER characteristics for intra-disk RAID-5 and RAID-6 as a function of the ECC correction capability. The mid-1X TLC NAND Flash RBER working region and the 10^{-16} target UBER are highlighted.

Another important point to consider in modeling the UBER with previous equations is related to the nature of the errors considered in the calculation of F_{CW}. In reality, there can be defect issues that cause, for example, an entire page to catastrophically fail. Under this condition, the actual F_{CW} is the sum of the F_{CW} from RBER calculated with eq.(9.1) and the F_{CW} from defects. This latter component is orders of magnitude lower than the RBER related F_{CW}, but is found to be dominating in the UBER contribution when RBER values are low. The defects cause mainly a slope change of the UBER characteristic dependently on the magnitude of the F_{CW} from defects. However, the RBER values considered in the analysis of this work are sufficiently high to neglect the defects contribution on the overall UBER calculation.

9.2.2 Impact of the SSD workload constraints

The UBER calculated with eq.(9.4) is a static metric, since it is independent from the time variations of RBER in the NAND Flash sectors that occur during the SSD lifetime. Further, that equation represents an instantaneous calculation that does not include any usage model

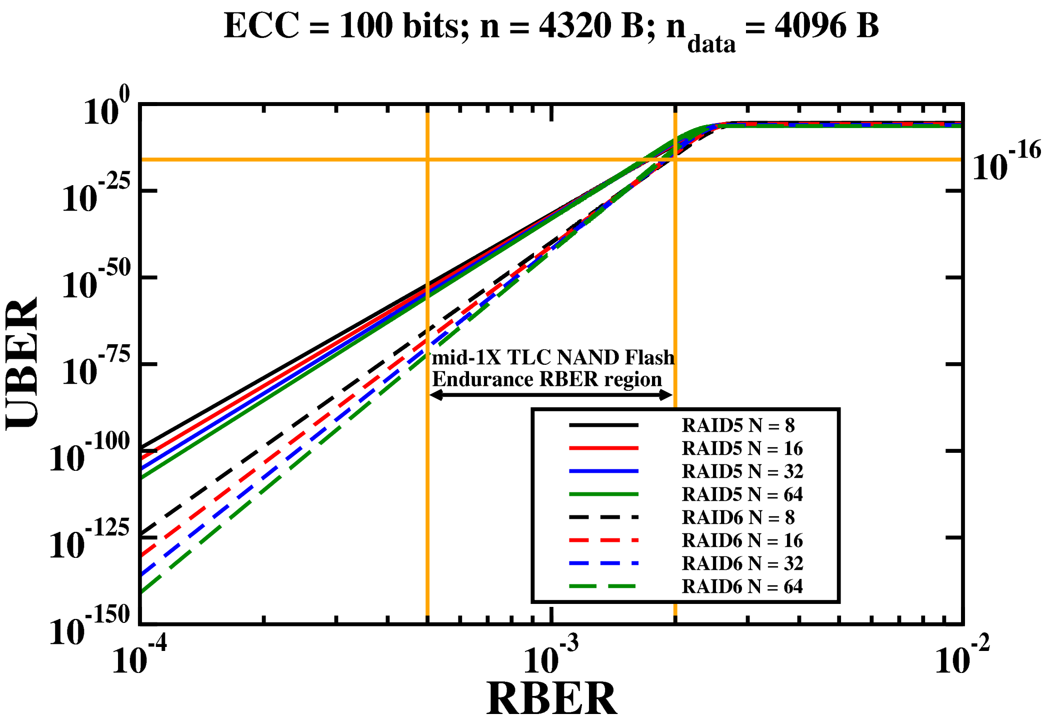


Figure 9.4: UBER characteristics for intra-disk RAID-5 and RAID-6 as a function of the stripe length. The mid-1X TLC NAND Flash RBER working region and the 10^{-16} target UBER are highlighted.

of the memory [7]. The JESD22-A117C document defines the UBER as [25]:

$$UBER = \frac{CDE}{\sum_{BS} (PE_{cycles} \cdot RPC + RAC)} \tag{9.5}$$

where CDE is the cumulative number of data errors during cycling, BS is the number of bits in a tested sample, RPC is the number of reads per cycle, and RAC is the number of reads after cycling, respectively. To account for these considerations the eq.(9.4) needs to be modified as follows:

$$UBER(t) = \frac{1 - \int_0^t CSER(N, x)dt}{n_{data} \cdot N} \cdot \frac{1}{\frac{PE_{cycles}}{WA} + 1} \tag{9.6}$$

where the integral calculated on $CSER(N, x)$ represents the cumulative CSER calculated as a function of the time and equal either to $CSER_{R5}$ for RAID-5 or $CSER_{R6}$ for RAID-6, PE_{cycles} is the actual number of sustained program/erase cycles by the NAND Flash memories in the SSD, and WA is the disk write amplification factor, respectively. The basis of the time dependency in previous equations is related to the constraints applied to guardband the SSD reliability during a defined mission time. To limit the wear of the

NAND Flash in an SSD, especially in enterprise applications, a maximum number of Disk Writes Per Day (DWPD), a target Over-Provisioning (OP) level for the garbage collection operations, and finally a WA, is fixed. However, these definitions apply to the PE cycles domain that is not of particular interest in reliability estimations. A linear transformation of the NAND Flash PE cycles into SSD working hours can be performed to calculate the hours spent per PE cycle by using the formerly defined parameters as follows:

$$h_{PE} = \frac{24}{PE_{day}} = \frac{24}{DWPD/(1+OP) * WA} \tag{9.7}$$

where PE_{day} is the actual number of NAND Flash PE cycles per day that considers both user writes and internal SSD data movements accounted by the write amplification. With this approach the variable t becomes discrete, thus we can calculate $UBER(t)$ up to the SSD mission time using time steps equal to h_{PE}. A calculation issue could arise when the DWPD is large enough to achieve a number of PE cycles within the mission time higher than what has been measured in our samples and shown in Fig. 9.2. In this case the problem is solved by extrapolating the missing RBER values through an exponential fit of the NAND Flash RBER characteristics following the formula $RBER(t) = RBER_0 * exp^{mt}$, where $RBER_0$ is the value at beginning of the memory lifetime, and m is a fitting coefficient, respectively.

Fig. 9.5 shows the $UBER(t)$ calculated for different DWPD values using both intra-disk RAID-5 and RAID-6. The values considered span from 0.1 up to 5, representing typical read intensive (i.e., low DWPD) or write intensive (i.e., high DWPD) scenarios [26]. As it can be seen, both intra-disk RAID approaches can guarantee a UBER lower than 10^{-16} for DWPD values below one during a typical enterprise SSD mission time of 5 years. This is in line with the expectations for TLC-based SSD since TLC NAND Flash are mostly suitable for data archiving and write-once-read-many applications due to their reduced endurance compared either to MLC or SLC. Fig. 9.6 shows the $UBER(t)$ dependency on the WA. The WA parameter depends on many parameters like the workload type (e.g., sequential or random), the write transaction size, the amount of OP, and on the compressibility of the data (i.e., the data entropy). When the sum of those effects results in a low value both intra-disk RAID approaches are able to guarantee the target UBER, additionally proving that the RAID-6 approach allows achieving lower UBER for the entire mission time. Once again, we have to stress that the results extracted with these analysis are heavily idealized and must be complemented with additional information on the NAND Flash and SSD

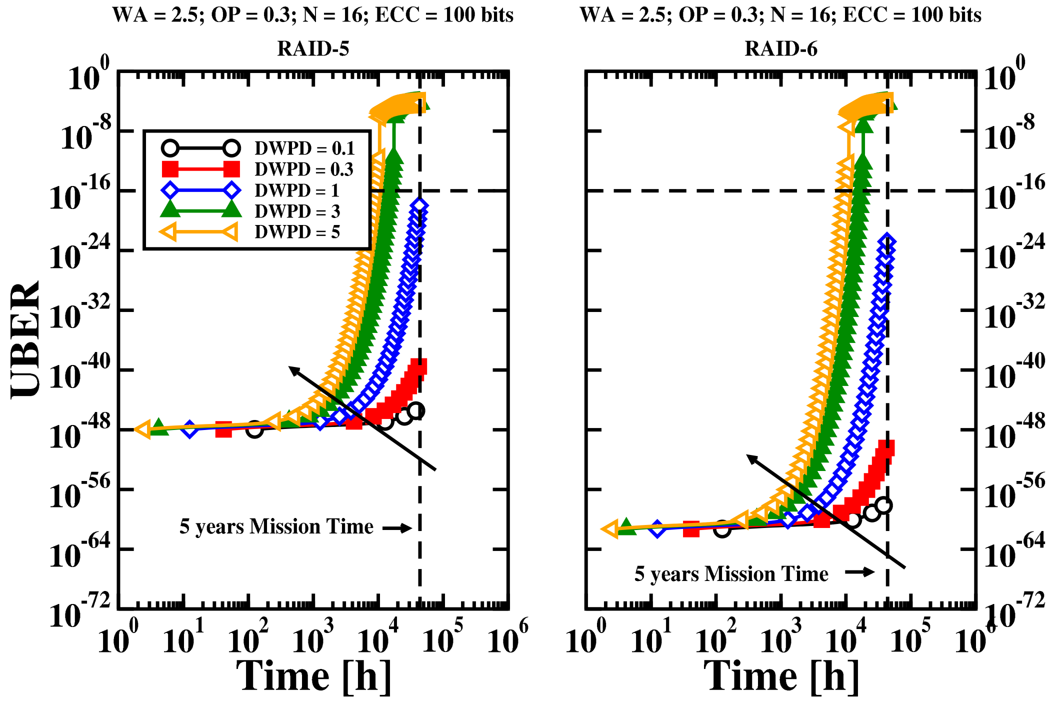


Figure 9.5: $UBER(t)$ **characteristics calculated for intra-disk RAID-5 and RAID-6 with different DWPD constraints.**

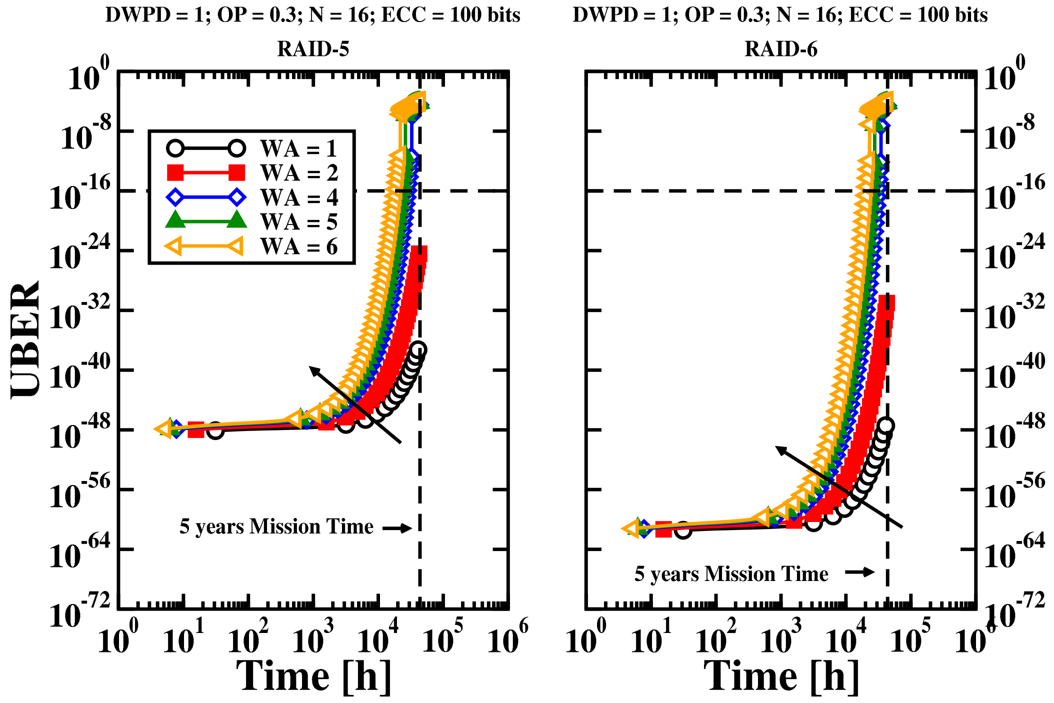


Figure 9.6: $UBER(t)$ **characteristics calculated for intra-disk RAID-5 and RAID-6 with different WA constraints.**

Table 9.1: SSD parameters considered in the simulations

Disk size	512 GB
Channels	32 (16 GB per channel)
NAND Flash technology	mid-1X TLC
NAND Flash page	16 KB+parity (4 sectors 4320B)
NAND Flash rated endurance	300 PE
ECC strength	up to 100 bits per 4320B
OP	30%
WA	from 1 up to 6
Assumed cycling temperature	55°C [6]

controller reliability, as shown in the next section of this work.

9.3 Modeling intra-disk RAID failures

The previous section of this paper considered the reliability of the intra-disk RAID by studying the UBER characteristics calculated for different SSD constraints throughout a typical SSD mission time. However, that approach did not include all the potential failure patterns that could be experienced by an SSD yielding to possible data losses. In this section we model the failure behaviors of the RAID-5 and RAID-6 approaches through DTMC simulations of a 512GB enterprise SSD whose parameters are indicated in Table 9.1.

9.3.1 DTMC model

The most common metric devised to quantify the reliability of a RAID system is the Mean Time To Data Loss. However, its interpretation can be quite difficult to forecast real world applications, turning to be misleading most of the time [27]. A more useful metric is the PDL, that measures the risk of losing data within a specified time. To evaluate this metric we made use of the DTMC representation of the RAID-5 and RAID-6 approaches by modifying the state transition probabilities to fit the intra-disk RAID study case. The HFRS Markov chain solver [22] has been exploited for the simulations by modifying its code to accommodate both discrete time events and the time-varying nature of the F_{CW} in NAND

Flash sectors. All the simulations consider a five years SSD mission time that is discretized in time steps of 8.76 hours (i.e., 5000 simulation points). The choice of this value is sufficient to capture the F_{CW} variations in time according to the different workload constraints studied in this paper, thus ensuring the DTMC convergence in one of the chain's states. Each time step is simulated 10^5 times with an importance sampling algorithm applied on the exponentially distributed state transition probabilities to increase statistical consistency while providing rare-events observability [22]. The F_{CW} variability shown in Fig.9.2 is considered in all the simulations and the worst-case assumptions on die-to-die variability are the same applied for the ideal calculations in section III.

Let us consider the RAID-5 case whose model is depicted in Fig. 9.7. The simulation of the SSD always starts in state 0 which represents the *normal mode*. At this point the SSD could move into *degraded mode* by two failure types: *i)* a sector cannot be recovered by the ECC and requires the reconstruction using the RAID approach; *ii)* an entire channel connected to the NAND Flash chips fails. While the first failure pattern is modeled by taking into account the F_{CW} at a defined time step, the second failure pattern requires some discussion. An SSD channel may fail due to different causes, the most common are: defective Flash controllers integrated in the SSD controller, interconnection or solder failures on the circuit board hosting the controller and the NAND Flash, and defective NAND Flash dice. Each one of these causes contribute to the channel failure rate λ used in the DTMC. When the SSD is in degraded mode it can be brought back to normal mode only if the failure was due by an uncorrectable sector. In this case the recovery rate μ models this transition probability. The time requested for a sector recovery via stripe reconstruction has been evaluated in the range of tens of milliseconds through an accurate SSD performance co-simulator [28]. If the SSD is in degraded mode due to a channel failure there is no possibility to return in normal mode. In this case the SSD will apply some restrictions to the workload (i.e., limits the write operations), but still allows for data reconstruction of the failed channel. This is the case used for data backup on another device. The *data loss* state (i.e., RAID failure) can be reached by these conditions:

- Two uncorrectable sector errors in the same stripe
- Any uncorrectable sector error and a channel failure
- A channel failure and any uncorrectable sector error in another channel
- Two channel failures

Similar considerations apply for intra-disk RAID-6 modeled as in Fig. 9.7b. In this case the model complexity increases to include the additional *degraded mode* state typical of RAID-6 systems, and the additional failure patterns to reach the data loss state. Another important consideration for DTMC simulations is related to the storage efficiency of the two intra-disk RAID approaches that is calculated as [29]:

$$S = 1 - \frac{p}{N} \begin{cases} p = 1 & \text{RAID-5} \\ p = 2 & \text{RAID-6} \end{cases} \tag{9.8}$$

where N is the stripe length. The S parameter contributes to the channel capacity available for SSD user operation C_{ch} that is used in the state transition probabilities calculation as:

$$C_{ch} = S \cdot (1 - OP) \cdot 16\text{GB} \tag{9.9}$$

9.3.2 Impact of DWPD and WA on data loss

The DWPD and WA parameters play a major role in the SSD UBER determination, as seen in the previous section of this paper. Here we apply the DTMC model of intra-disk RAID-5 and RAID-6 to evaluate the data loss probability over different SSD workload conditions. The channel failure probability λ in the DTMC is set in all the simulations to 32.5 FIT, that is assumed by considering the NAND Flash die failure rate (0.25 FIT) [30], and the typical failure rate of an SSD controller manufactured with state-of-the-art Application Specific Integrated Circuit guidelines [31]. The data loss probability is evaluated by considering a stripe length equal to the SSD channel number, namely 32. Fig. 9.8 shows the evolution of the PDL and of the UBER during a 5 years SSD mission time. In that figure it is possible to extract information on the dominant failure mechanism. If we consider the case of DWPD = 3 for RAID-5, it is appreciable that in the first part of the SSD mission time the PDL is dominated by the channel failure probability. Then, as soon the F_{CW} starts to increase during the mission time, the dominant failure mechanism is that of a single sector failure plus a channel failure. In the remainder of SSD mission time, the dominant failure mechanism is that of two sectors failures in a stripe, since F_{CW} is very high. For DWPD values below unity, the dominant failulre mechanism at 5 years mission time is generally that of channel failures. The relationship between PDL and UBER has been derived by the following equation [32]:

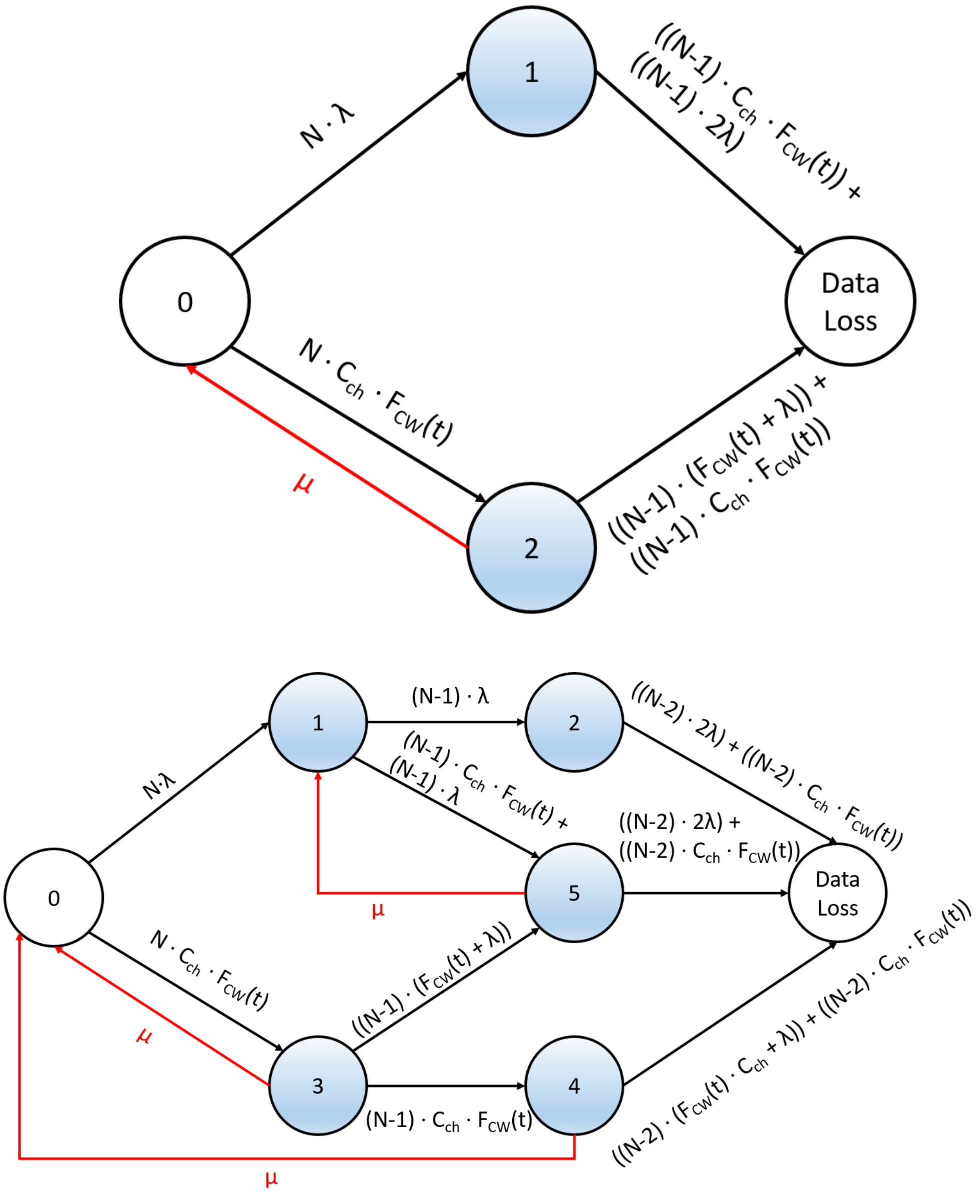


Figure 9.7: DTMC for intra-disk RAID-5 (top) and RAID-6 (bottom) simulations. The RAID degraded modes are highlighted.

$$UBER = \frac{PDL}{(N-x) \cdot C_{ch}} \begin{cases} x = 1 & \text{RAID-5} \\ x = 2 & \text{RAID-6} \end{cases} \tag{9.10}$$

In Table 9.2 are reported the PDL and the correspondent UBER at 5 years by considering a disk WA equal to 2. It can be observed that the RAID-6 approach always outperforms RAID-5 in terms of PDL and UBER when the DWPD conditions are those representing

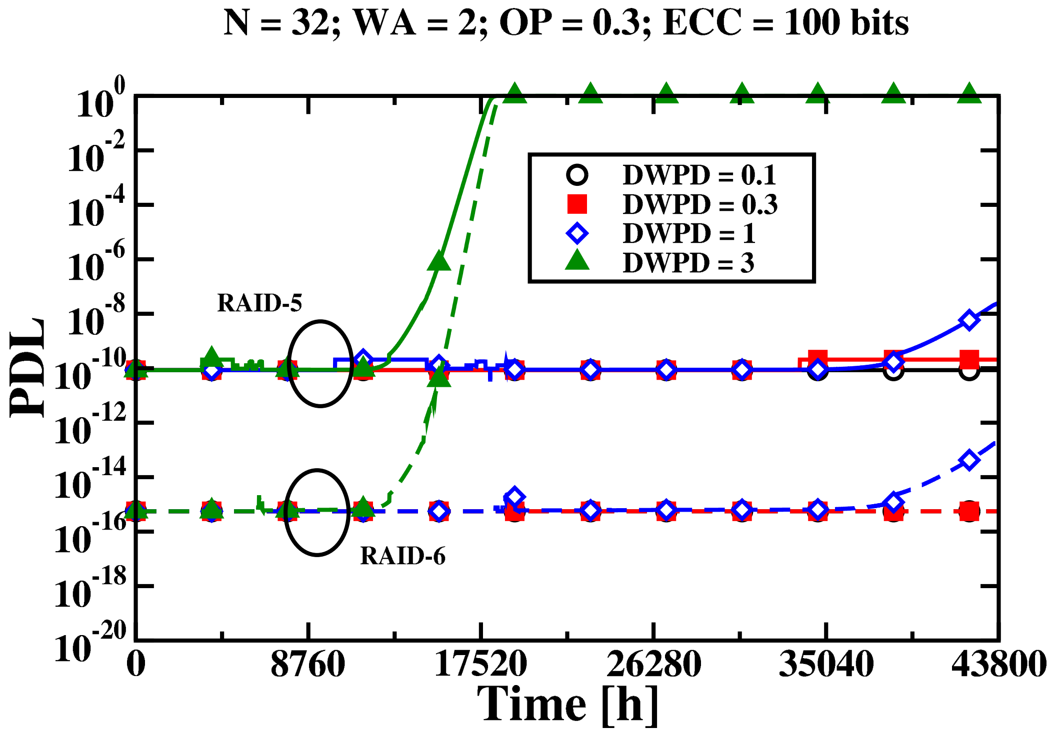


Figure 9.8: PDL evolution during time for intra-disk RAID-5 and RAID-6 calculated by DTMC simulations.

the typical usage model of TLC NAND Flash memories, namely read-most workloads with DWPD below unity. When the workload becomes write-most there is no difference between RAID-5 and RAID-6 since both approaches cannot keep the UBER below the 10^{-16} target in 5 years.

In Table 9.3 is reported the role of the WA on the data loss by considering a DWPD equal to 0.3. The choice of analyzing the WA only for a read-most workload lies on the fact that the major contributor on the disk reliability is the DWPD. Therefore, this approach will help understanding whether or not the WA has some macroscopical effects. The PDL and the UBER data at 5 years show that high WA values contribute to decrease the SSD reliability. However, it must be noted that both approaches materialize into a significant margin with respect to the SSD reliability target throughout the entire mission time.

9.3.3 Impact of the channel failure rate on data loss

Most of the investigations on intra-disk RAID for SSD evaluate the reliability by explicitly neglecting the channel failure rate in the analysis [20, 21, 32]. In the DTMC model this parameter (i.e., λ) is taken into account, therefore a sensitivity analysis of its impact on the

Table 9.2: PDL and UBER retrieved at 5 years SSD mission time as a function of the DWPD for different intra-disk RAID approaches

DWPD	RAID	PDL	UBER
0.1	5	8.53e-11	2.95e-23
	6	5.59e-16	2.07e-28
0.3	5	2.08e-10	7.20e-23
	6	5.59e-16	2.07e-28
1	5	2.30e-8	7.96e-21
	6	1.95e-13	7.21e-26
3	5	1	3.46e-13
	6	1	3.70e-13

Table 9.3: PDL and UBER retrieved at 5 years SSD mission time as a function of the WA for different intra-disk RAID approaches

WA	RAID	PDL	UBER
1	5	8.53e-11	2.95e-23
	6	5.59e-16	2.07e-28
2	5	2.08e-10	7.20e-23
	6	5.59e-16	2.07e-28
4	5	3.08e-10	1.07e-22
	6	6.24e-16	2.31e-28
6	5	5.09e-10	1.76e-22
	6	3.69e-15	1.36e-27

PDL and on the UBER is mandatory. Since we previously observed that TLC NAND Flash-based SSD works better with low DWPD constraints we have considered the following SSD simulation scenario [33]: *i)* read-most workload with low WA (DWPD = 0.3, WA = 2) mimicking sequential video-on-demand traffic; *ii)* read-most workload with high WA (DWPD = 0.3, WA = 6) typical of highly random applications like file servers. This will return PDL and UBER values at 5 years mission time that are not heavily affected by write operation degradation (i.e., PDL = 1). In the analysis we have considered three different channel failure rates: 10 FIT, 32.5 FIT, and 90 FIT. Fig. 9.9 shows that the channel failure

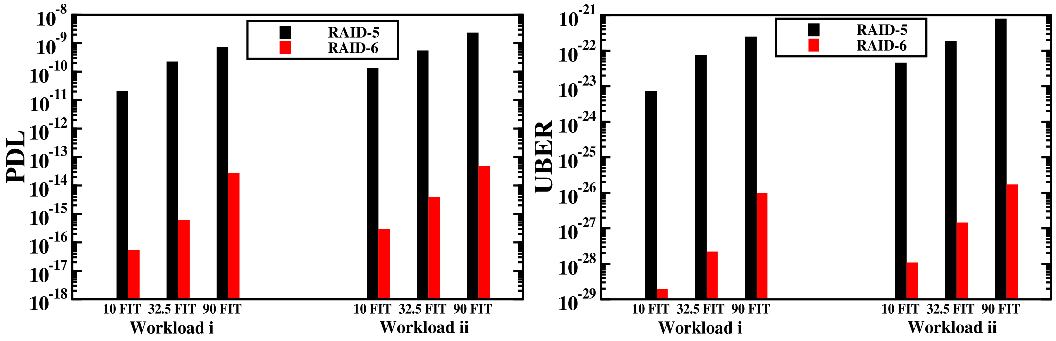


Figure 9.9: PDL (left) and UBER (right) dependency on the channel failure rate for different workloads on intra-disk RAID-5 and RAID-6.

rate impacts the reliability of the SSD at 5 years mission time as expected. Indeed, the PDL and the UBER will increase by increasing that value, although showing significant reliability margin.

The dominant failure mechanism in these simulations for RAID-5 and RAID-6 is that of channel failures due to the low DWPD value exploited, as previously explained. Such result is confirmed by the fact that PDL and UBER scales as the square of the FIT value.

9.3.4 Impact of the stripe length on cost and reliability

The last parameter impacting the intra-disk RAID behavior is the stripe length. This factor expose a trade-off in the SSD since it affects both reliability and performance. The larger is the stripe the lower is the level of protection that can be achieved against channel or sector failures [22]. However, using very short stripe lengths may lead to severe performance loss caused by the reduced disk storage efficiency, as evidenced by eq.(9.8). A trade-off needs to be exercised in this context. We considered in the DMTC simulations the same workload as for the analysis on the channel failure rate PDL and UBER sensitivity by varying the stripe length N from 4 up to 32. The channel failure rate is fixed to 32.5 FIT. Fig. 9.10 shows that the higher is the stripe length the higher will be the PDL and the UBER as expected, although for the simulated workload conditions every intra-disk RAID configuration is well below the reliability limits. In this case, it is important to evaluate the economical impact of the RAID, as shown in Fig. 9.11. Assume an SSD manufactured with an addressable NAND Flash capacity of 512GB. The available user capacity reduces around 350GB with an OP around 30% (i.e., typical in enterprise SSD). Shorter stripe lengths will result in a drastic reduction of the user capacity, especially for the RAID-6 approach. While it is generally

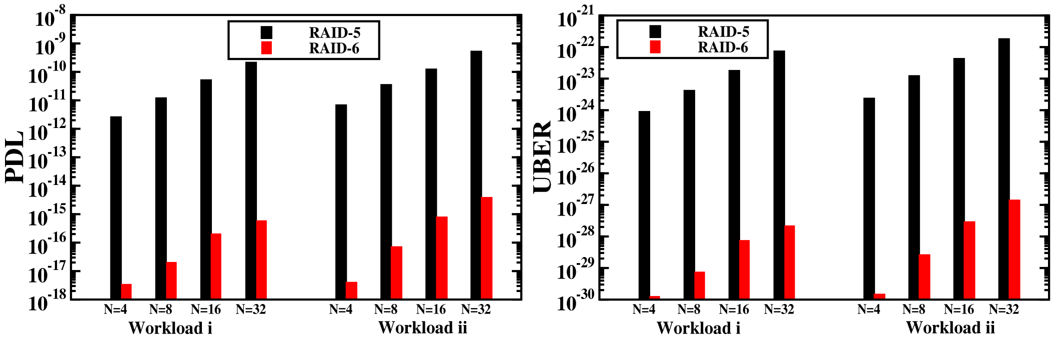


Figure 9.10: PDL (left) and UBER (right) dependency on the stripe length for different workloads on intra-disk RAID-5 and RAID-6.

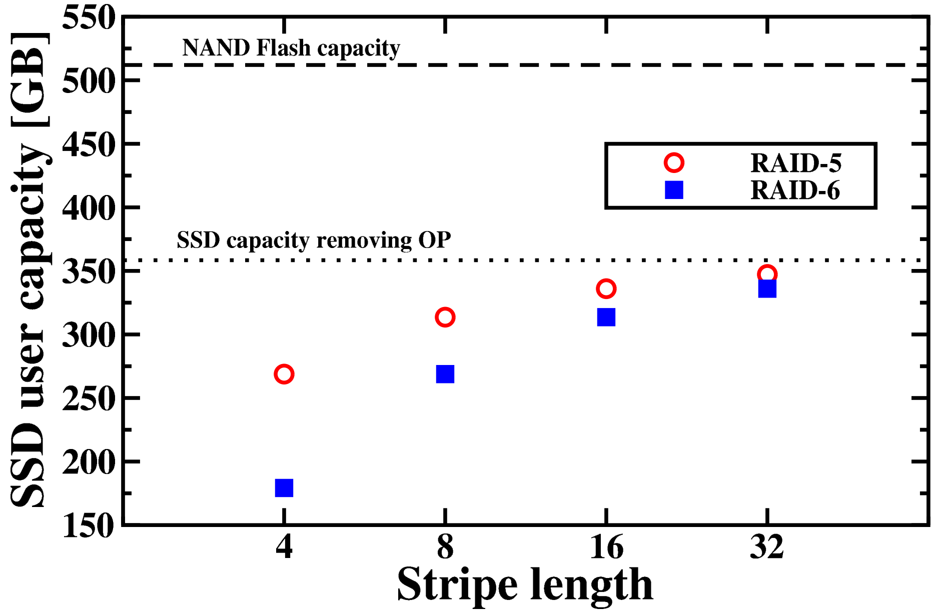


Figure 9.11: Stripe length impact on the overall SSD capacity when either intra-disk RAID-5 or RAID-6 is exploited.

true that shorter stripe length will provide higher reliability against data corruption it is not economical wasting more than 50% of the user capacity for RAID (i.e, storing the stripes parity), considering the fact that achievable UBER is significantly below the 10^{-16} reliability bound. Moreover, a shorter stripe will severely impact the DWPD constraint due to the increased number of parity-write operations on the disk, thus requiring proper management burdening on the overall SSD performance [15]. The minimum stripe length that considers all the aforementioned thoughts should be in our opinion between 8 and 16 for RAID-5 and between 16 and 32 for RAID-6 if additional data protection is needed.

In this work we have investigated the reliability of two different intra-disk RAID approaches (RAID-5 and RAID-6) for SSD when the considered storage medium is an ultra-scaled TLC NAND Flash technology. The simulation results on the UBER indicated that RAID-6 offers superior data protection with respect to RAID-5 as expected (up to five orders of magnitude), but at the cost of an increased disk capacity utilization. Different workload constraints have been analyzed to prove this assumption on a broad scale of cases, showing that intra-disk RAID becomes ineffective in applications far from the TLC NAND Flash typical usage model. Through the development of a DTMC model that included the channel failure rate we have been able, for the first time, to evaluate the impact of the hardware failures on the SSD reliability, while at the same time quantifying the data loss probability.

Bibliography

[1] R. Micheloni, L. Crippa, and A. Marelli, Eds., *Inside NAND Flash memories*. Springer-Verlag, 2010, DOI: 10.1007/978-90-481-9431-5.

[2] A. Grossi, L. Zuolo, F. Restuccia, C. Zambelli, and P. Olivo, "Quality-of-Service Implications of Enhanced Program Algorithms for Charge-Trapping NAND in Future Solid-State Drives," *IEEE Transactions on Device and Materials Reliability*, vol. 15, no. 3, pp. 363–369, Sep. 2015, DOI: 10.1109/TDMR.2015.2448108.

[3] S. Tanakamaru, Y. Yanagihara, and K. Takeuchi, "Over-10x-extended-lifetime 76%-reduced-error solid-state drives (SSDs) with error-prediction LDPC architecture and error-recovery scheme," in *IEEE International Solid-State Circuits Conference (ISSCC)*, Feb. 2012, pp. 424–426, DOI: 10.1109/ISSCC.2012.6177074.

[4] Y. Cai, G. Yalcin, O. Mutlu, E. F. Haratsch, A. Cristal, O. S. Unsal, and K. Mai, "Error analysis and retention-aware error management for NAND flash memory," *Intel Technology Journal*, vol. 17, no. 1, p. 140, May 2013.

[5] L. Zuolo, C. Zambelli, P. Olivo, R. Micheloni, and A. Marelli, "LDPC Soft Decoding with Reduced Power and Latency in 1X-2X NAND Flash-Based Solid State Drives," in *IEEE International Memory Workshop (IMW)*, May 2015, pp. 1–4, DOI: 10.1109/IMW.2015.7150293.

[6] JEDEC, "JESD218B Solid-State Drive (SSD) Requirements and Endurance Test Method," Jun. 2016.

[7] T. Marquart, "Solid-State-Drive qualification and reliability strategy," in *IEEE International Integrated Reliability Workshop (IIRW)*, Oct. 2015, pp. 3–6, DOI: 10.1109/IIRW.2015.7437056.

[8] J.-D. Lee, C.-K. Lee, M.-W. Lee, H.-S. Kim, K.-C. Park, and W.-S. Lee, "A new programming disturbance phenomenon in NAND flash memory by source/drain hot-electrons generated by GIDL current," in *IEEE Non-Volatile Semiconductor Memory Workshop*, Feb. 2006, pp. 31–33, DOI: 10.1109/.2006.1629481.

[9] C. Compagnoni, R. Gusmeroli, A. Spinelli, and A. Visconti, "Analytical Model for the Electron-Injection Statistics During Programming of Nanoscale nand Flash Memories," *IEEE Transactions on Electron Devices*, vol. 55, pp. 3192–3199, Nov. 2008, DOI: 10.1109/TED.2008.2003332.

[10] A. Chimenton, C. Zambelli, and P. Olivo, "A Statistical Model of Erratic Behaviors in Flash Memory Arrays," *IEEE Transactions on Electron Devices*, vol. 58, no. 11, pp. 3707–3711, Nov. 2011, DOI: 10.1109/TED.2011.2165722.

[11] Y. Park, J. Lee, S. S. Cho, G. Jin, and E. Jung, “Scaling and reliability of NAND flash devices,” in *IEEE International Reliability Physics Symposium (IRPS)*, Jun. 2014, pp. 2E.1.1–2E.1.4, DOI: 10.1109/IRPS.2014.6860599.

[12] E. Vatajelu, H. Aziza, and C. Zambelli, “Nonvolatile memories: Present and future challenges,” in *9th International Design Test Symposium (IDT)*, Dec. 2014, pp. 61–66, DOI: 10.1109/IDT.2014.7038588.

[13] H. Shim, S.-S. Lee, B. Kim, N. Lee, D. Kim, H. Kim, B. Ahn, Y. Hwang, H. Lee, J. Kim, Y. Lee, H. Lee, J. Lee, S. Chang, J. Yang, S. Park, S. Aritome, S. Lee, K.-O. Ahn, G. Bae, and Y. Yang, “Highly reliable 26nm 64Gb MLC E2NAND (Embedded-ECC & Enhanced-efficiency) flash memory with MSP (Memory Signal Processing) controller,” in *Symposium on VLSI Technology (VLSIT)*, Jun. 2011, pp. 216–217.

[14] Y. Li, “3 Bit Per Cell NAND Flash Memory on 19nm Technology,” Flash Memory Summit, Aug. 2012.

[15] S. Lee, B. Lee, K. Koh, and H. Bahn, “A Lifespan-aware Reliability Scheme for RAID-based Flash Storage,” in *Proceedings of the 2011 ACM Symposium on Applied Computing*, Mar. 2011, pp. 374–379, DOI: 10.1145/1982185.1982266.

[16] D. Yimo, L. Fang, C. Zhiguang, and M. Xin, *WeLe-RAID: A SSD-Based RAID for System Endurance and Performance*. Berlin, Heidelberg: Springer Berlin Heidelberg, Oct. 2011, pp. 248–262, DOI: 10.1007/978-3-642-24403-2_20.

[17] Y. Li, P. P. C. Lee, and J. C. S. Lui, “Analysis of Reliability Dynamics of SSD RAID,” *IEEE Transactions on Computers*, vol. 65, no. 4, pp. 1131–1144, Apr. 2016, DOI: 10.1109/TC.2014.2349505.

[18] Micron Technology Inc., “NAND Flash Media Management Through RAIN,” [Online] https://www.micron.com/~/media/documents/products/technical-marketing-brief/brief_ssd_rain.pdf, 2011.

[19] LSI Corporation, “LSI SandForce SF3700 Flash Controller,” [Online] http://www.storagesearch.com/lsi-3rdgenerationssdcontroller.pdf, 2013.

[20] S. Im and D. Shin, “Flash-Aware RAID Techniques for Dependable and High-Performance Flash Memory SSD,” *IEEE Transactions on Computers*, vol. 60, no. 1, pp. 80–92, Jan. 2011, DOI: 10.1109/TC.2010.197.

[21] J. Kim, E. Lee, J. Choi, D. Lee, and S. H. Noh, “Chip-Level RAID with Flexible Stripe Size and Parity Placement for Enhanced SSD Reliability,” *IEEE Transactions on Computers*, vol. 65, no. 4, pp. 1116–1130, Apr. 2016, DOI: 10.1109/TC.2014.2375179.

[22] K. Greenan, “Reliability and Power-Efficiency in Erasure-Coded Storage Systems,” Ph.D. dissertation, University of California, Santa Cruz, Dec. 2009.

[23] N. Mielke, T. Marquart, N. Wu, J. Kessenich, H. Belgal, E. Schares, F. Trivedi, E. Goodness, and L. R. Nevill, “Bit error rate in NAND Flash memories,” in *IEEE International Reliability Physics Symposium (IRPS)*, Apr. 2008, pp. 9–19, DOI: 10.1109/RELPHY.2008.4558857.

[24] C. Zambelli, P. King, P. Olivo, L. Crippa, and R. Micheloni, "Power-Supply Impact on the Reliability of mid-1X TLC NAND Flash Memories," in *IEEE International Reliability Physics Symposium (IRPS)*, Apr. 2016, pp. 2B.3.1–2B.3.6, DOI: 10.1109/IRPS.2016.7574509.

[25] JEDEC, "JESD22-A117 document," Oct. 2011.

[26] R. Micheloni, A. Marelli, and K. Eshghi, Eds., *Inside Solid State Drives (SSDs)*. Springer Netherlands, 2013, DOI: 10.1007/978-94-007-5146-0.

[27] K. M. Greenan, J. S. Plank, and J. J. Wylie, "Mean time to meaningless: MTTDL, Markov models and storage system reliability," in *HotStorage '10: 2nd Workshop on Hot Topics in Storage and File Systems*. USENIX, Jun. 2010.

[28] L. Zuolo, C. Zambelli, R. Micheloni, M. Indaco, S. D. Carlo, P. Prinetto, D. Bertozzi, and P. Olivo, "SSDExplorer: A Virtual Platform for Performance/Reliability-Oriented Fine-Grained Design Space Exploration of Solid State Drives," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1627–1638, Oct. 2015, DOI: 10.1109/TCAD.2015.2422834.

[29] I. Iliadis, R. Haas, X.-Y. Hu, and E. Eleftheriou, "Disk Scrubbing Versus Intradisk Redundancy for RAID Storage Systems," *Trans. Storage*, vol. 7, no. 2, pp. 5:1–5:42, Jul. 2011, DOI: 10.1145/1970348.1970350.

[30] B. Schroeder, R. Lagisetty, and A. Merchant, "Flash Reliability in Production: The Expected and the Unexpected," in *USENIX Conference on File and Storage Technologies*, Feb. 2016, pp. 67–80.

[31] Altera, "Reliability report," [Online] https://www.altera.com/en_US/pdfs/literature/rr/rr.pdf, 2015.

[32] W. Yi, H. Xu, Q. Xie, and N. Li, "A Flash-aware Intra-disk Redundancy scheme for high reliable All Flash Array," *IEICE Electronics Express*, vol. 12, no. 13, pp. 20 150 295–20 150 295, May 2015, DOI: 10.1587/elex.12.20150295.

[33] Samsung, "Optimized solid-state drives ideal for data center environments," [Online] http://www.samsung.com/semiconductor/global/file/insight/2015/08/PM863_White_Paper-0.pdf, 2015.

Conclusions

In Chapter 1 we have shown how Solid State Drives are changing the way people store and process data. SSDs are very complex systems to build because they require a sophisticated mix of hardware, software, and firmware. On top of that, non-volatile memories can be of different types, involving totally different storage mechanisms, each of them with its own reliability challenges. All of the above considerations imply tens of billions of dollars spent in R&D worldwide each year, with engineers from all over the places scratching their heads to solve very complex problems: mathematics, physics, circuit design, process technology, manufacturing, lithography, signal processing, and testing techniques are all called to give their contribution to drive the evolution of SSDs even further.

In Chapter 2 we have demonstrated why an SSD design aimed at optimizing performances must follow a Bottom-Up approach; indeed, most of the design constraints are strongly related to the performance and reliability of the nonvolatile storage medium.

A detailed knowledge of the memory behavior (i.e. endurance, data retention and read disturb) is mandatory to efficiently design the whole SSD architecture. RBER represents the main figure of merit driving designer's choices. The knowledge of RBER and, in particular, of its dependency from time and workload, allows selecting the most effective architecture to extend the memory's lifetime as much as possible. Since RBER increases with technology scaling, the use of LDPC codes represents the solution of choice for the most advanced ECC engines.

Once the NAND Flash memories (together with the knowledge of their RBER) and the most appropriate ECC algorithm (together with either read retry techniques or LDPC soft decisions) have been selected, the design of the SSD controller must be based on multiple aspects:

- the ECC architecture, as a trade-off between performance (bandwidth, latency, power consumption) and area occupation;
- the number of memory channels, as a trade-off again between performance and area occupation;
- the number of memory dies per channel, that is generally a power of 2;

- the appropriate command management, maximizing the number of active dies and hence the SSD bandwidth, whereas limiting as much as possible the maximum latency ({\it i.e.} the QoS) by leveraging the head of line blocking concept;
- the introduction of a DRAM data cache buffer able to reduce the number of access operations to the NAND Flash memories, thus increasing SSD bandwidth while reducing NAND Flash degradation effects;
- the choice of the most suitable host interface able to guarantee the performance requested by the host applications.

To further improve the performance of next generation SSDs to be used in hyperscaled environments it is possible to leverage new approaches, like SDF, exploiting hardware/software co-design of the SSD controller architecture and of the host applications.

In Chapter 3 we have presented many architectural options for building a 3D NAND array, including some of the latest and greatest layout options, but the 3D evolution is just at the beginning. In fact, two fundamentally different technologies, Floating and Charge Trap, are fighting each other, trying to prove that they can win in the long run, i.e. when scaling will be pushed to the limit. Flash manufactures are already shooting for 100 vertical layers with multi-level capabilities, including 4 bit/cell. No doubt that we'll see a lot of innovations in the near future: engineers and scientists are called to give their best effort to make this vertical evolution happen.

In Chapter 4 we have demonstrated the benefits of machine learning in 3D NAND Flash characterization through the application of data clustering algorithms. The characterization data set has been obtained by an extensive testing campaign of 3D-NAND Flash devices under different operating conditions. By developing a semi-supervised learning methodology we have been able to optimize the LDPC code rate dedicated to ECC, resulting in a 24\% gain of the memory space addressable by the user. Such activity paves the way for further applications in the memory characterization context.

In Chapter 5 we have analyzed the impact of the power supply voltage on the reliability of a TLC mid-1X NAND Flash memory. Through an experimental characterization performed during endurance stress it was observed that the number of errors and the page-to-page errors variability strongly depend on the power supply. By simulating the different blocks of the high voltage circuitry in the NAND Flash system through a SPICE model we identified some of the possible culprits of this dependence, namely the regulators controlling the generation of the program voltage in the TSP algorithm and the wordline switch. Finally, we have also investigated the possible side effects of the coupled noise sources with the high voltage NAND Flash subsystem, evidencing that even if the power supply is chosen in a safe operating region, it is not immune from errors.

In Chapter 6 we have investigated the differences between uniform and concentrated read disturb effects in mid-1X TLC NAND Flash memories. The characterization showed that a uniform read access of NAND Flash blocks yields to a reproducible signature of the disturb. The wordlines close to the drain selector are those heavily affected by the phenomenon. By characterizing the concentrated read access mode, better mimicking the real workloads that a memory will sustain throughout its lifetime, it is appreciable that the errors profile on the wordlines is similar to the uniform case except for the two neighbors closer to the one where the read accesses are concentrated. The implications on the enterprise SSD are evident: when a concentrated read access is performed there is up to a 22\% achievable page reads count by a block before requesting the intervention of the ECC or other data management policies (e.g., scrubbing). For server and OLTP application this could represent a limitation in terms of reliability, endurance, and power consumption of the drive.

In Chapter 7 we have shown how emerging memories such RRAMs have to be wisely designed when they are used as the main storage media in SSDs. Indeed, replacing NAND flash memories with RRAMs in a “plug and play” fashion is not the best way to reach high performances and low-latency. Moreover, even in the best working conditions (i.e., co-designing the memories characteristics together with the whole SSD architecture), it has been shown that the optimum design point of the All-RRAM SSD is still affected by the host configuration and its requirements. This is in agreement with today's trend of developing specific SSD architectures for specific host applications \cite{SDF}; the downside of this approach is that it leads to extremely complex SSD designs with hundreds of parameters to explore. SSDExplorer can definitely help to address the above mentioned problems, allowing a better understanding of where, in All-RRAM SSDs, the co-design activity is more effective.

In Chapter 8 we have shown that MRAM is a viable alternative for replacing DRAM inside NVRAM cards. Number of IOPS and latency figures have been extensively analyzed under different workload conditions and queue depths. In all cases, we haven’t detected any significant performance degradation with respect to the DRAM/Flash legacy solu-tions. MRAM-based architectures can definitely simplify the card design by removing the need for Flash memories and the super-capacitor. Looking forward, the overall cost and power of the NVRAM card need to be assessed, especially considering that MRAM den-sity is expected to reach 4Gbit/die in the coming years.

In Chapter 9 we have investigated the reliability of two different intra-disk RAID approaches (RAID-5 and RAID-6) for SSD when the considered storage medium is an ultra-scaled TLC NAND Flash technology. The simulation results on the UBER indicated that RAID-6 offers superior data protection with respect to RAID-5 as expected (up to five orders of magnitude), but at the cost of an increased disk capacity utilization. Different workload constraints have been analyzed to prove this assumption on a broad scale of cases, showing that intra-disk RAID becomes ineffective in applications far from the TLC NAND Flash typical usage model. Through the development of a DTMC model that included the channel failure rate we have been able, for the first time, to evaluate the impact of the hardware failures on the SSD reliability, while at the same time quantifying the data loss probability.

Author’s Publications

Conference Papers

[1] R. Micheloni, " Flash Controllers for SSD’s Lifetime Extension through Machine Learning", Proceedings of the 15th International System-on-Chip (SoC) Conference, Irvine, CA, USA, Oct. 18-19, 2017.

[2] (Invited) R. Micheloni, "3D Nand Flash Memories: Array Architectures and Scaling/Reliability Challenges", Proceedings of the Non-Volatile Memory Technology Symposium (NVMTS), Aachen, Germany, Aug. 30 -Sept. 1, 2017.

[3] C. Zambelli, L. Zuolo, P. Olivo, L. Crippa, A. Marelli, R. Micheloni, "Characterization of Uniform and Concentrated Read Disturb Effect in Mid-1X TLC NAND Flash Memories", Proceedings of the Flash Memory Summit, www.flashmemorysummit.com, Santa Clara, CA, USA, Aug. 8-10, 2017.

[4] L. Zuolo, C. Zambelli, T. Hulett, B. Cooke, R. Micheloni, P. Olivo, "IOPS and QoS Analysis of DRAM-based and MRAM-based NVRAM Cards", Proceedings of the Flash Memory Summit, www.flashmemorysummit.com, Santa Clara, CA, USA, Aug. 8-10, 2017.

[5] R. Micheloni, A. Marelli, "SSD Lifetime Extension using Multi-Code-Rate LDPC with Multi-Dimensional LLR", Proceedings of the Flash Memory Summit, www.flashmemorysummit.com, Santa Clara, CA, USA, Aug. 8-10, 2017.

[6] C. Zambelli, G. Cancelliere, F. Riguzzi, E. Lamma, P. Olivo, A. Marelli, and R. Micheloni, "Characterization of TLC 3D-NAND Flash Endurance through Machine Learning for LDPC Code Rate Optimization", Proceedings of the IEEE International Memory Workshop (IMW), Monterey, CA, USA, May 14-17, 2017.

[7] C. Zambelli, P. Olivo, L. Crippa, A. Marelli, and R. Micheloni, "Uniform and Concentrated Read Disturb Effects in Mid-1X TLC NAND Flash Memories for Enterprise Solid State Drives", Proceedings of the IEEE International Reliability Physics Symposium (IRPS), Monterey, CA, USA, Apr. 2-6, 2017.

[8] L. Zuolo, C. Zambelli, T. Hulett, B. Cooke, R. Micheloni, P. Olivo, " IOPS and QoS Analysis of DRAM/Flash-based and All-MRAM based NVRAM cards", Non-Volatile Memory Workshop (NVMW), San Diego, CA, USA, March 12-14, 2017.

[9] R. Micheloni, "Impact of 3D Flash Memories on SSD's Controller Design", Proceedings of the 14th International System-on-Chip (SoC) Conference, Irvine, CA, USA, Oct. 19-20, 2016.

[10] L. Zuolo, M. Cirella, C. Zambelli, R. Micheloni, P. Olivo, "Performance Assessment of an All-RRAM Solid State Drive Through a Cloud-Based Simulation Framework", Proceedings of the Flash Memory Summit, www.flashmemorysummit.com, Santa Clara, CA, USA, Aug. 8-11, 2016.

[11] A. Marelli, R. Micheloni, "False Decoding Probability (Detection) of BCH and LDPC Codes", Proceedings of the Flash Memory Summit, www.flashmemorysummit.com, Santa Clara, CA, USA, Aug. 8-11, 2016.

[12] L. Zuolo, C. Zambelli, A. Grossi, P. Olivo, R. Micheloni, S. Bates, "Memory System Architecture Optimization for Enterprise All-RRAM Solid State Drives", Proceedings of the IEEE International Memory Workshop (IMW), Paris, France, May 15-18, 2016.

[13] C. Zambelli, P. King, P. Olivo, L. Crippa and R. Micheloni, "Power-Supply Impact on the Reliability of mid-1X TLC NAND Flash Memories", Proceedings of the 2016 IEEE International Reliability Physics Symposium (IRPS), Pasadena, CA, USA, Apr. 17-21, 2016.

[14] L. Zuolo, C. Zambelli, R. Micheloni, S. Bates, P. Olivo, "Design Space Exploration of Latency and Bandwidth in RRAM-based Solid State Drives ", Poster Session, Non-Volatile Memory Technology Symposium (NVMTS), Tsinghua University, China, Oct. 11-14, 2015.

[15] R. Micheloni, A. Marelli, L. Crippa, A. Aldarese, "Fully Integrated NAND-SSD Characterization Flow", Proceedings of the Flash Memory Summit, www.flashmemorysummit.com, Santa Clara, CA, USA, Aug. 11-13, 2015.

[16] K. Zhao, R. Micheloni, T. Zhang, "Safely Overclocking Flash I/O in SSDs", Proceedings of the Flash Memory Summit, www.flashmemorysummit.com, Santa Clara, CA, USA, Aug. 11-13, 2015.

[17] L. Zuolo, C. Zambelli, R. Micheloni, P. Olivo, "SSDExplorer: A virtual platform for SSD simulation", Proceedings of the Flash Memory Summit, www.flashmemorysummit.com, Santa Clara, CA, USA, Aug. 11-13, 2015.

[18] L. Zuolo, C. Zambelli, P. Olivo, R. Micheloni, A. Marelli, "LDPC Soft Decoding with Reduced Power and Latency in 1X-2X NAND Flash-Based Solid State Drives", Proceedings of the International Memory Workshop (IMW), Monterey, CA, USA, April, 2015.

Journal Papers

[19] C. Zambelli, A. Marelli, R. Micheloni, P. Olivo, "Modeling the Endurance Reliability of Intra-disk RAID Solutions for mid-1X TLC NAND Flash Solid State Drives”, IEEE Transactions on Device and Materials Reliability, to be published 2017.

[20] R. Micheloni, P. Olivo, “Solid-State Drives (SSDs) [Scanning the Issue]”, Proceedings of the IEEE, Volume 105, Issue 9, pp. 1586-1588, September, 2017.

[21] (Invited) L. Zuolo, C. Zambelli, R. Micheloni, P. Olivo, “Solid-State Drives: Memory Driven design Methodologies for Optimal Performance”, Proceedings of the IEEE, Volume 105, Issue 9, pp. 1589-1608, September, 2017.

[22] (Invited) R. Micheloni, S. Aritome, L. Crippa, “Array architectures for 3-D NAND Flash Memories”, Proceedings of the IEEE, Volume 105, Issue 9, pp. 1634-1649, September, 2017.

[23] R. Micheloni, L. Crippa, C. Zambelli, P. Olivo, "Architectural and Integration Options for 3D NAND Flash Memories”, MDPI Computers, Special Issue on “3D Flash Memories”, 6(3), 27, doi: 10.3390/computers6030027, 2017.

[24] L. Zuolo, C. Zambelli, A. Marelli, R. Micheloni, P. Olivo, "LDPC Soft Decoding with Improved Performance in 1X-2X MLC and TLC NAND Flash-Based Solid State Drives”, IEEE Transactions on Emerging Topics in Computing, to be published 2017.

[25] R. Micheloni, "Solid-State Drive (SSD): A Nonvolatile Storage System”, Point Of View, Proceedings of the IEEE, Volume 105, Issue 4, pp. 583-588, April, 2017.

[26] L. Zuolo, C. Zambelli, R. Micheloni, S. Galfano, M. Indaco, S. Di Carlo, P. Prinetto, P. Olivo, D. Bertozzi, "SSDExplorer: a Virtual Platform for Fine-Grained Design Space Exploration of Solid State Drives", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, April, 2015.

Posters

[27] L. Zuolo, M. Cirella, C. Zambelli, R. Micheloni, P. Olivo, "Simulation platform for sub-10us RRAM-based Solid State Drives", Poster Session, Open Power Summit Europe, Barcelona, Spain, Oct. 26-28, 2016.

[28] L. Zuolo, M. Cirella, C. Zambelli, R. Micheloni, S. Bates, P. Olivo, "Performance Assessment of an All-RRAM Solid State Drive through a Cloud-Based Simulation Framework", Poster Session, Non-Volatile Memory Workshop (NVMW), San Diego, CA, USA, March 6-8, 2016.

[29] L. Zuolo, C. Zambelli, R. Micheloni, S. Bates, P. Olivo, "Quality of Service Implications of the Error Correction Techniques in Solid State Drives", Poster Session, Non-Volatile Memory Workshop (NVMW), San Diego, CA, USA, March 1-3, 2015.

Contributed Chapters

[30] R. Micheloni, L. Crippa, “Solid State Drives (SSDs)”, Chapter 1 in “Solid-State-Drives (SSDs) Modeling”, R. Micheloni (Ed.), ISBN 978-3-319-51734-6, pp. 1-17, Springer, 2017.

[31] R. Micheloni, L. Crippa, “NAND Flash Memories”, Chapter 2 in “Solid-State-Drives (SSDs) Modeling”, R. Micheloni (Ed.), ISBN 978-3-319-51734-6, pp. 19-39, Springer, 2017.

[32] L. Zuolo, C. Zambelli, R. Micheloni, P. Olivo, “SSDExplorer: A Virtual Platform for SSD Simulations”, Chapter 3 in “Solid-State-Drives (SSDs) Modeling”, R. Micheloni (Ed.), ISBN 978-3-319-51734-6, pp. 41-65, Springer, 2017.

[33] L. Zuolo, C. Zambelli, A. Marelli, R. Micheloni, P. Olivo, “Design Trade-Offs for NAND Flash-Based SSDs”, Chapter 4 in “Solid-State-Drives (SSDs) Modeling”, R. Micheloni (Ed.), ISBN 978-3-319-51734-6, pp. 67-97, Springer, 2017.

[34] L. Zuolo, C. Zambelli, R. Micheloni, P. Olivo, “Simulations of RRAM-Based SSDs”, Chapter 6 in “Solid-State-Drives (SSDs) Modeling”, R. Micheloni (Ed.), ISBN 978-3-319-51734-6, pp. 123-138, Springer, 2017.

[35] L. Zuolo, C. Zambelli, L. Crippa, R. Micheloni, P. Olivo, “Simulations of SSD’s Power Consumption”, Chapter 7 in “Solid-State-Drives (SSDs) Modeling”, R. Micheloni (Ed.), ISBN 978-3-319-51734-6, pp. 139-151, Springer, 2017.

[36] L. Zuolo, C. Zambelli, R. Micheloni, P. Olivo, “Simulations of Software-Defined Flash”, Chapter 8 in “Solid-State-Drives (SSDs) Modeling”, R. Micheloni (Ed.), ISBN 978-3-319-51734-6, pp. 153-165, Springer, 2017.

[37] R. Micheloni, L. Crippa, “3D Stacked NAND Flash Memories”, Chapter 3 in “3D Flash Memories”, R. Micheloni (Ed.), ISBN 978-94-017-7510-6, pp. 63-84, Springer, 2016.

[38] L. Crippa, R. Micheloni, “3D Charge Trap NAND Flash Memories”, Chapter 4 in “3D Flash Memories”, R. Micheloni (Ed.), ISBN 978-94-017-7510-6, pp. 85-128, Springer, 2016.

[39] R. Micheloni, L. Crippa, “3D Floating Gate NAND Flash Memories”, Chapter 5 in “3D Flash Memories”, R. Micheloni (Ed.), ISBN 978-94-017-7510-6, pp. 129-166, Springer, 2016.

[40] L. Crippa, R. Micheloni, “Advanced Architectures for 3D NAND Flash Memories with Vertical Channel”, Chapter 6 in “3D Flash Memories”, R. Micheloni (Ed.), ISBN 978-94-017-7510-6, pp. 168-196, Springer, 2016.

[41] H. Huang, R. Micheloni, “3D Multi-chip Integration and Packaging Technology for NAND Flash Memories”, Chapter 9 in “3D Flash Memories”, R. Micheloni (Ed.), ISBN 978-94-017-7510-6, pp. 261-280, Springer, 2016.

[42] A. Marelli, R. Micheloni, “BCH and LDPC Error Correction Codes for NAND Flash Memories”, Chapter 10 in “3D Flash Memories”, R. Micheloni (Ed.), ISBN 978-94-017-7510-6, pp. 281-320, Springer, 2016.

Dottorati di ricerca

Il tuo indirizzo e-mail

rino.micheloni@studentunife.it

Oggetto:

Dichiarazione di conformità della tesi di Dottorato

Io sottoscritto Dott. (Cognome e Nome)

Micheloni Rino

Nato a:

San Marino

Provincia:

San Marino

il giorno:

29/08/1969

Avendo frequentato il Dottorato di Ricerca in:

Scienze dell'Ingegneria

Ciclo di Dottorato

30

Titolo della tesi:

Memory-Driven Design Methodologies Far Solid State Drives (SSDs)

Titolo della tesi (traduzione):

Tutore: Prof. (Cognome e Nome)

Olivo Piero

Settore Scientifico Disciplinare (S.S.D.)

Ing-inf/01

Parole chiave della tesi (max 10):

SSD, Flash memories, MRAM, ReRAM

Consapevole, dichiara

CONSAPEVOLE: (1) del fatto che in caso di dichiarazioni mendaci, oltre alle sanzioni previste dal codice penale e dalle Leggi speciali per l'ipotesi di falsità in atti ed uso di atti falsi, decade fin dall'inizio e senza necessità di alcuna formalità dai benefici conseguenti al provvedimento emanato sulla base di tali dichiarazioni (2) dell'obbligo per l'Università di provvedere al deposito di legge delle tesi di dottorato al fine di assicurarne la conservazione e la consultabilità da parte di terzi; (3) della procedura adottata dall'Università di Ferrara ove si richiede che la tesi sia consegnata dal dottorando in 2 copie di cui una in formato cartaceo e una in formato pdf non modificabile su idonei supporti (CD-ROM, DVD) secondo le istruzioni pubblicate sul sito: http://www.unife.it/studenti/dottoratoalla voce ESAME FINALE - disposizioni e modulistica; (4) del fatto che l'Università, sulla base dei dati forniti, archivierà e renderà consultabile in rete il testo completo della tesi di dottorato di cui alla presente dichiarazione attraverso l'Archivio istituzionale ad accesso aperto "EPRINTS.unife.it" oltre che attraverso i Cataloghi delle Biblioteche Nazionali Centrali di Roma e Firenze; DICHIARO SOTTO LA MIA RESPONSABILITA': (1) che la copia della tesi depositata presso l'Università di Ferrara in formato cartaceo è del tutto identica a quella presentata in formato elettronico (CD-ROM, DVD), a quelle da inviare ai Commissari di esame finale e alla copia che produrrò in seduta d'esame finale. Di conseguenza va esclusa qualsiasi responsabilità dell'Ateneo stesso per quanto riguarda eventuali errori, imprecisioni o omissioni nei contenuti della tesi; (2) di prendere atto che la tesi in formato cartaceo è l'unica alla quale farà riferimento l'Università per rilasciare. a mia richiesta. la dichiarazione di conformità di eventuali copie; (3) che il contenuto e l'organizzazione della tesi è opera originale da me realizzata e non compromette in alcun modo i diritti di terzi, ivi compresi quelli relativi alla sicurezza dei dati personali; che pertanto l'Università è in ogni caso esente da responsabilità di qualsivoglia natura civile, amministrativa o penale e sarà da me tenuta indenne da qualsiasi richiesta o rivendicazione da parte di terzi; (4) che la tesi di dottorato non è il risultato di attività rientranti nella normativa sulla proprietà industriale, non è stata prodotta nell'ambito di progetti finanziati da soggetti pubblici o privati con vincoli alla divulgazione dei risultati, non è oggetto di eventuali registrazioni di tipo brevettale o di tutela. PER ACCETI AZIONE DI QUANTO SOPRA RIPORTATO

Firma del dottorando

Ferrara, li 09/02/2018 (data) Firma del Dottorando

Michelon

Firma del Tutore

Visto: Il Tutore Si approva Firma del Tutore ____________________