Notes of Solid-State Drive

Chapter 1 Introduction

As modern computing systems (e.g., enterprise servers, data center storage, and consumer devices) process a large amount of data at an unprecedented scale, **storage devices must meet high requirements for storage capacity and I/O performance**. While *electromechanical disk drives* (i.e., hard-disk drives) have continuously ramped in capacity, the rotating-storage technology does not provide the access time or transfer-rate performance required in demanding enterprise applications, including online transaction processing, data mining, and cloud computing. Client applications also require an alternative to electromechanical disk drives that can deliver faster response times, consume less power, and fit into smaller mobile form factors.

A solid-state drive (SSD) has two main advantages compared to a traditional hard-disk drive (HDD): speed and reliability, both of which stem from the absence of mechanical moving parts, as shown in Figure 1.*1*. There is no need to wait for the drive's head to move to the correct spot to read or write data, which improves speeds both in general operation and at the device's start-up. Also, there is no need to worry about the head mechanism wearing out.

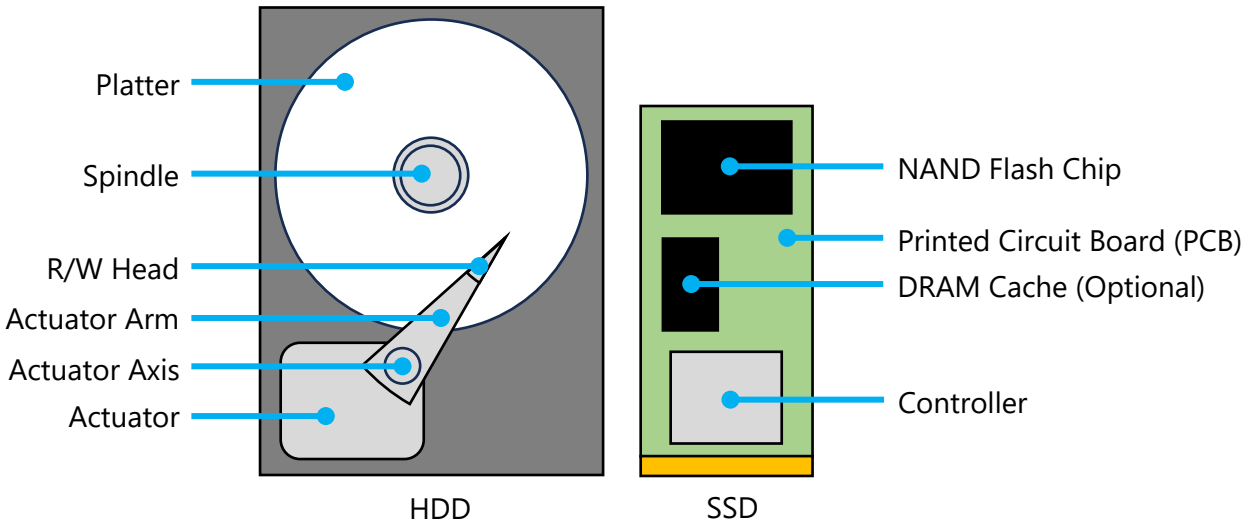


Figure 1.1 Overview of HDD and SSD.

A NAND flash-based solid-state drive is a storage device that utilizes NAND flash memory to store user data. NAND flash memory is a type of non-volatile solid-state storage that persistently stores and retrieves data. It is non-volatile memory, as it retains data even when power is not applied, and has become the de facto standard for architecting storage devices in modern computing systems. **NAND flash memory offers higher performance** (i.e., lower latency and higher bandwidth) compared to magnetic storage, which utilizes rigid and rapidly rotating platters and read/write heads. Additionally, its capacity has increased continuously, and its costs have decreased over the decades. Given these advantages, NAND flash-based SSDs can provide orders-of-magnitude higher I/O performance (i.e., lower read & write access time and higher random-access input/output operations) compared to traditional hard-disk drives (HDDs), (also, reliability because of resilience to physical shock, a small form factor, consuming less static power) with a much lower cost-per-bit value over SSDs based on emerging non-volatile memory (NVM)

technologies. Thus, SSDs have now reached a point where they can serve as replacements for traditional rotating storage.

NAND flash memory has several unique characteristics, such as the erase-before-write property (i.e., a flash cell needs to be first erased before programming it), limited lifetime (i.e., a cell cannot reliably store data after experiencing a certain number of program/erase (P/E) cycles), and large operation units (e.g., modern NAND flash memory typically reads/writes data in a page (e.g., 16 KiB) granularity). To achieve high performance and large capacity of the storage system while hiding the unique characteristics of NAND flash memory, it is critical to design efficient SSD firmware, commonly called Flash-Translation Layer (FTL). An FTL is responsible for many critical management tasks, such as address translation, garbage collection, wear leveling, and I/O scheduling, which significantly affect the performance, reliability, and lifetime of the SSD.

In a typical flash memory-based storage system, the applications from the host issue I/O requests by calling file system APIs (e.g., fread/fwrite). File systems typically manage the storage capacity as a linear array of fixed-size blocks. Therefore, a file access is converted to many block-level I/O requests by the file system. Each block-level I/O request contains a specific logical block address (LBA) and a data block length. Block-level I/O requests will be processed through the FTL and translated into specific commands provided by the flash memory. The commands will be converted into control signals to access the raw flash memory media.

Modern solid-state drives can be abstracted into three levels: (1) NAND flash memory, (2) integrated circuit architecture, and (3) SSD firmware, as shown in Figure 1.2. At the lowest level of abstraction is the NAND flash memory level, which describes the motion of electrons for the basic operations (i.e., program, read, and erase) for a single NAND flash memory cell. Above the device level is engineering level: the integrated circuit architecture and SSD firmware levels. (mark) .

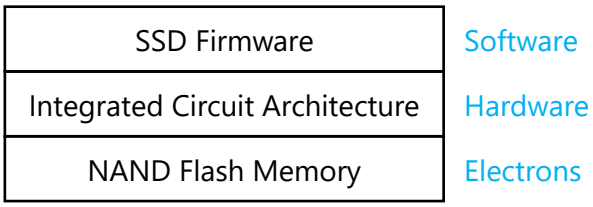


Figure 1.2 Abstraction Levels of Modern Solid-State Drive

In all SSDs, a flash microcontroller sits between one or multiple hosts (i.e., CPUs) and NAND flash memories, and on each side, there are a lot of challenges that designers need to overcome. Moreover, a single controller can have multiple cores, with all the complexity associated with developing a multi-threaded firmware. (mark) As usual, simulation speed and precision do not go hand in hand, so it is important to understand when to simulate what.

We will first dive into the lowest level, NAND flash physics, to understand the characteristics of NAND Flash since its unique properties decide the upper levels' structures.

Note the NAND flash's unique properties are including:

- Large operation units
- Erase-before-write property
- Asymmetry in operation units
- Limited endurance
- Various error sources
- Asymmetry in operation latencies

(retention loss, schematic, wear leveling->for endurance, Error Correction Code->for reliability, soft decoding, randomization, read retry)

Chapter 2 NAND Flash Memories

Benefiting from two key trends: (1) effective process technology scaling and (2) multi-level (e.g., MLC, TLC) cell data coding, NAND flash memory becomes ubiquitous in everyday life today (e.g., flash cards, USB keys, and solid-state drives). Unfortunately, the reliability of raw data stored in flash memory has also become more challenging to ensure due to these two trends. Manufacturing process scaling, which has increased the number of flash memory cells within a fixed area, makes fewer electrons in the flash memory cell floating gate to represent the data. Multi-level cell data coding, which represents more than one bit of digital data in a single floating-gate transistor, results in larger cell-to-cell interference and disturbance effects. Without mitigation, worsening reliability can reduce the lifetime of NAND flash memory.

To develop mitigation mechanisms, we first need to understand the inside of NAND flash memory.

2.1. NAND Flash Cell

The basic building block of NAND flash memory is the NAND flash cell. NAND flash memory stores information (i.e., data) as the number of electrons, which produces a threshold voltage (Turn-on voltage) variation of each NAND flash cell. A NAND flash cell is a special type of nMOS transistor based on the Floating Gate (FG) technology, whose cross-section and symbol (i.e., diagram and schematic) are shown in *Figure 2.1*. This *floating-gate transistor* is built with two overlapping gates rather than a single one: the first one is contacted to form the gate terminal, while the second one is completely surrounded by oxide. On top of the isolated gate is an interpoly oxide layer, and at the bottom is a tunnel oxide layer. Oxide layers insulate the floating gate (storage layer), preventing electrons from leaking out of it. As a result, **this isolated gate constitutes an excellent "trap" for electrons**, which ideally do not discharge even when flash memory is powered off (i.e., without the connection of a supply voltage) over years, ensuring charge retention and guaranteeing years of operation.

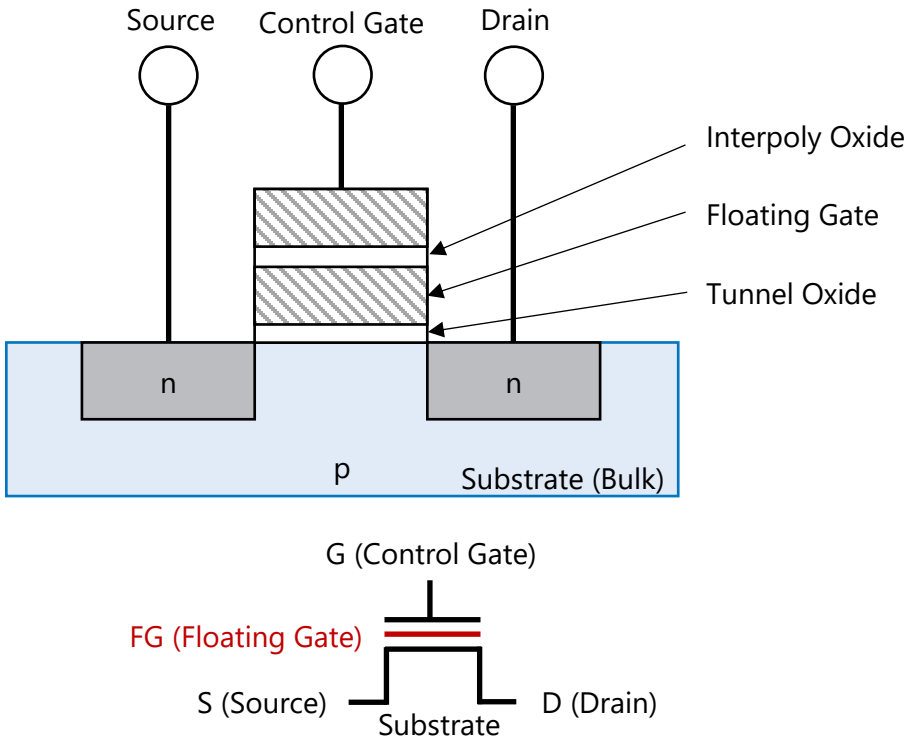
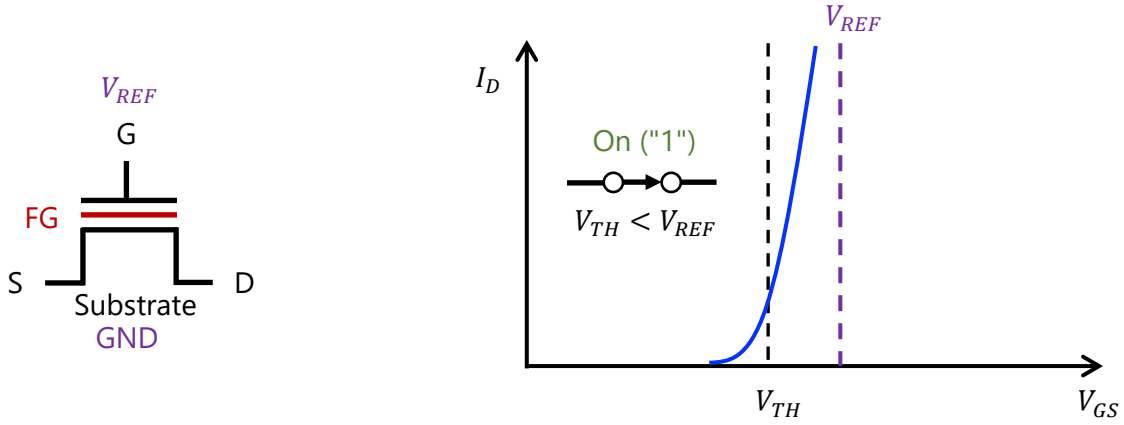


Figure 2.1 Diagram and Schematic of Floating Gate Memory Cell.

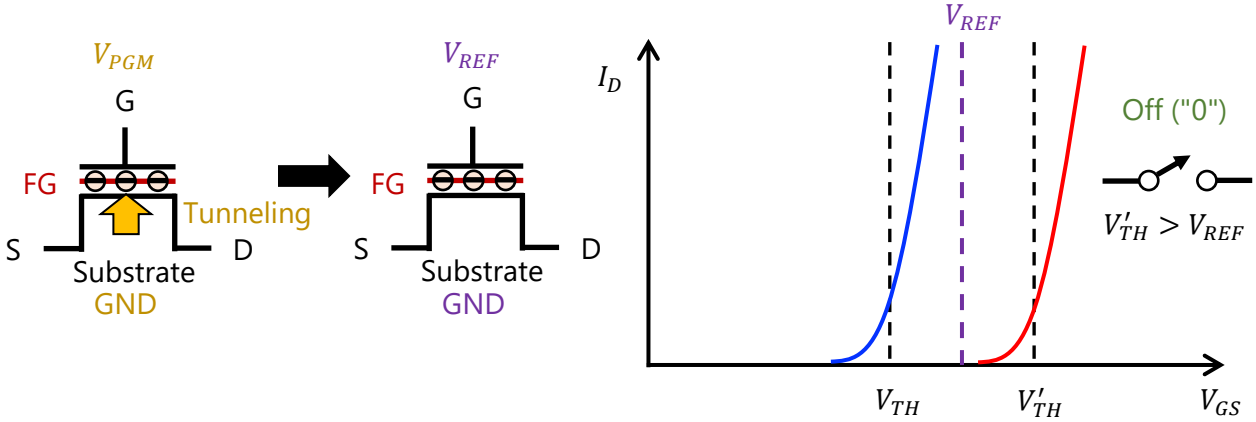
Program, Read and Erase of a Single Floating Gate Cell

For NAND flash cells, data are stored through program/erase operations and accessed via read operation. When we conduct the program, erase or read operations, electrons must be injected, removed, and counted from the isolated floating gate, respectively. The number of electrons stored in that gate is associated with the threshold voltage of the NAND flash cell. In *Figure 2.2*(a), given a fixed gate voltage (V_{REF}), the cell current (I_D) is a function of its threshold voltage (V_{TH}) because flash cells act like usual MOS transistors. Therefore, it is possible to understand the threshold voltage distribution to which the memory cell belongs through a current measure.

Program and erase operations modify the threshold voltage V_{TH} of the memory cell based on the *Fowler-Nordheim* (FN) *tunneling mechanism*. Earlier NAND flash chips stored a single bit of data in each cell, which was referred to as **single-level cell (SLC)** NAND flash. By applying a high program voltage (V_{PGM}, e.g., 8/18/26 V) to the cell's control gate and keeping the bulk terminal at ground potential (0 V), the **program operation** induces a large current transfer through the whole FG cell stack. This operation can set the transistor to a specific threshold voltage within a fixed range of voltages (*negative charge*), as shown in *Figure 2.2*(b). SLC NAND flash divided this fixed range into two voltage windows: one window represents the bit value 0 and the other represents the bit value 1, which we call the ER (erase) and P1 (program 1) states. The **erase operation** works principally in the same way but with control gate voltages negative with respect to the cell channel region (*positive charge*). Note that programming a cell cannot change the ER state to the P1 state. Thus, NAND flash cells have the erase-before-write property.



(**a**) Applying V_{REF} to Flash Cell with No Charge, and Its Current-Voltage (I-V) Characteristics



(**b**) Applying V_{PGM} to Charge Cell, and Its Current-Voltage (I-V) Characteristics

Figure 2.2 Read (**a**) and Program (**b**) Operation of SLC NAND Flash Cell.

A **read operation** is performed by comparing the cell's threshold voltage with a reference voltage V_{REF}. By applying a reference voltage V_{REF} to the cell's terminals, it allows discrimination between two storage levels: when the gate voltage is higher than the cell's V_{TH}, the cell is on ("1"), otherwise it is off ("0"). In other words, each state representing a different value is assigned to a voltage window within the range of all possible threshold voltages.

The floating gate cell can encode one or more bits of digital data. Specifically, cells containing n bit of information have $2n$ different levels of V_{TH}. **Multi-level cell (MLC)** NAND flash divides the flash voltage range into four voltage windows, each representing a possible 2-bit value (00, 01, 10, and 11), which we refer to as ER, P1, P2, and P3 states. Each voltage window in MLC NAND flash is therefore much smaller than a voltage window in SLC NAND flash. This smaller voltage window makes it more challenging to identify the value stored in a cell. Moreover, **triple-level cell (TLC)** NAND flash further divides the range, providing eight voltage windows to represent a 3-bit value, which we refer to as ER, P1-P7 states. And, Quadruple-level cell (QLC) NAND flash stores a 4-bit value per cell. The benefits of multi-bit per cell storage are the additional capacity

of the SSD without increasing the chip size, while it also decreases reliability by making the cells more difficult to store and read bits correctly.

Due to the process variation (i.e., cell's structural characteristics), the threshold voltage of flash cells programmed or erased to the same state varies across the voltage window, as shown in *Figure 2.3*. A program verify (PV) level confirms that a memory cell has been successfully programmed to its target state, preventing over-programming. An erase verify (EV) level ensures that the cell has been completely erased to its default low-voltage state, preventing over-erasing or leaving data in an intermediate, un-erased state. These verify levels are crucial for maintaining data reliability by ensuring cells do not become stuck in an intermediate state, preventing errors and data loss, particularly in multi-bit per cell devices.

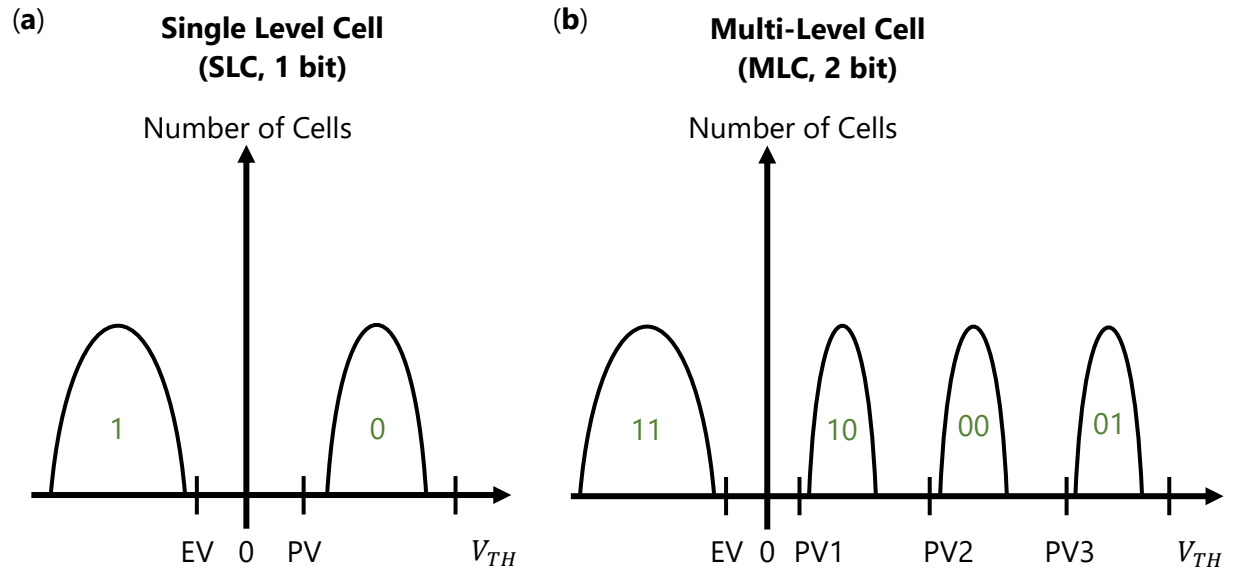


Figure 2.3 Memory Cell Threshold Voltage Distributions for Single Level Cells (**a**) and Multi-Level Cells (**b**).

Given a fixed program voltage, some cells might be successfully programmed to a specific V_{TH} state (easy-to-program cells) while others do not (hard-to-program cells). Thus, programming a floating gate cell to a specific V_{TH} state is typically accomplished by the so-called "incremental step pulse programming" (ISPP) scheme. ISPP applies several high program voltage pulses to the control gate. During each pulse, electrons move from the substrate to the floating gate. After every pulse, a read-verify is performed. This programming technique repeats its sequence multiple times until the verification passes on all programmed cells. For example, each word line is programmed 3 times, such that V_{TH} distributions can be progressively tightened.

Reliability of Floating Gate Cell

The reliability of FG NAND flash memory is one of the most important criteria. Typically 10 years of charge retention and 1-100 k program/erase cycles need to be guaranteed for a NAND flash product chip.

a) Charge Retention

Charge retention (or data retention) refers to the ability of a memory to retain stored information over time without any biases being applied. In *Figure 2.4*, a typical charge retention requirement is shown. $\Delta V_{TH,PGM}$ is the programmed threshold voltage shift, and UV stands for ultraviolet. UV erasure process for erasing data stored in NAND flash memory involves exposing the memory chip to UV light, which causes the floating gate to return to a default state, effectively erasing the data. This process is distinct from the programming or writing of data, which is done electronically. It needs to be guaranteed that, for a successful read-out of the stored information, the programmed V_{TH} (above the PV level) will not decrease more than 10% over the product's relevant time period of 10 years. In principle, there are multiple leakage paths which can lead to a loss of the programmed floating gate electron charges. The electrons can be lost through the interpoly oxide towards the control gate, leak through the cell side wall oxide to the cell junction area, or through the tunnel oxide towards the substrate, which is the most severe charge loss.

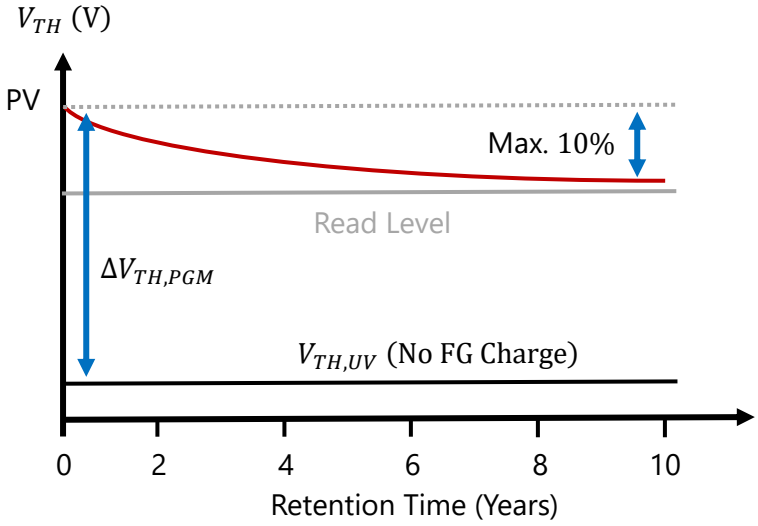


Figure 2.4 Charge Retention of an FG Cell. A Certain Amount of Charge Loss Needs To Be Tolerated (e.g., 10% V_{TH} Loss Over the Time Period of 10 Years). UV Stands for Ultraviolet.

We explain the electron leakage through tunnel oxide (TOX, refer to *Figure 2.1*) since it is the main factor in the wear of the FG cells. This type of electron leakage is not only because the TOX is physically the thinnest dielectric layer that holds the electrons on the floating gate but also because of the change in the density of traps in the tunnel oxide (or trap generation). The charge transfer during the program and erase operation generates electric states in the TOX, called **oxide traps** (the TOX should be the only dielectric where the electron charge is transferred). These traps are broken bonds of the atoms in the oxide matrix due to the electron tunneling processes. The density of traps in the tunnel oxide consequently increases with the number of program/erase cycles, which causes so-called **oxide stress**. The traps in the TOX barrier can act as stepping stones when floating gate electrons leak via a trap-assisted tunneling process toward the cell channel region. As the density of traps increases, the probability of this trap-to-trap tunneling (called stress-induced leakage current, SILC) becomes much higher than a direct tunneling process (for program or erase) through the whole TOX thickness. It means the *TOX is damaged*. The reason is that the effective tunnel distance of each tunneling step is significantly reduced for the SILC.

b) Endurance

Endurance is the maximum number of program/erase (P/E) operations (or P/E cycles) that the memory can withstand before it fails (i.e., life span). Figure 2.5 shows the endurance of FG cells in a 48 nm NAND technology. All program and erase cycles were carried out with unchanged program and erase voltages of $V_{PGM} = 23\ \text{V}$ and $V_{Erase} = -19\ \text{V}$ for the indicated pulse times (program period and erase period). The V_{TH} window has a decent size for low cycle numbers, whereas the V_{TH} window shrinks for higher cycle numbers above 300 cycles. Furthermore, a general V_{TH} upward shift is visible as the cycle number increases. The V_{TH} window closing and V_{TH} upward shift behaviors are because, at higher cycle numbers, the fixed negative charges which are generated in the TOX (trap generation) generally increase the cell's V_{TH}. In the case shown in Figure 2.5, the erased cell's V_{TH} is shifted by one volt after 10 k program/erase cycles. Besides the increased retention problem for higher cycle numbers due to trap generation, the window closing and the general V_{TH} upward shift will increase pulse voltages for electron charge, especially for the erase operation.

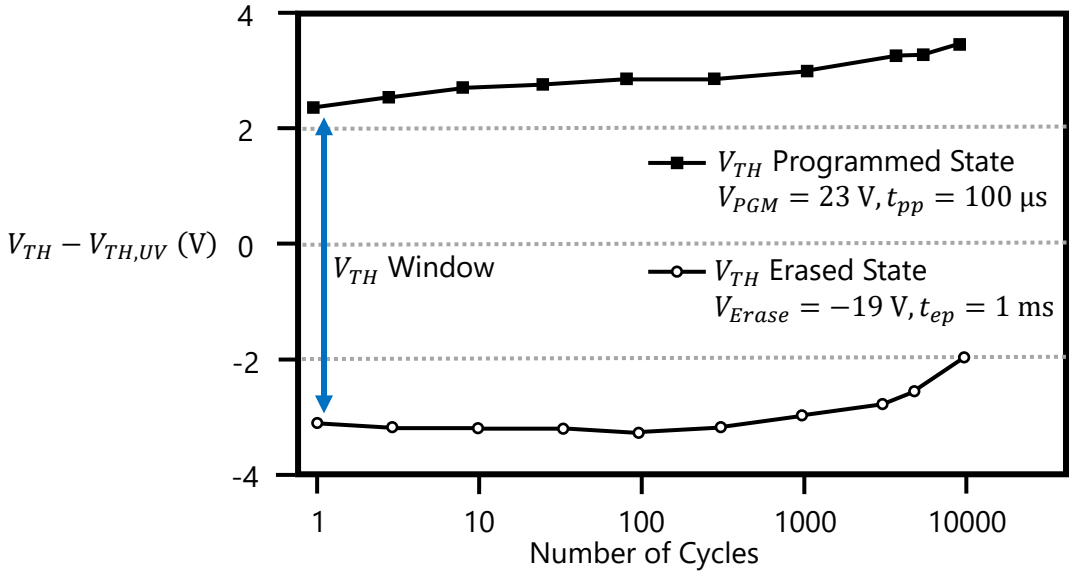


Figure 2.5 Program/Erase Cycling Endurance of an FG Cell in a 48 nm NAND Technology.

c) Threshold Voltage Shift

When the dimensions of floating gate cells are scaled down, the number of floating gate electrons needed for a certain threshold voltage shift ΔV_{TH} is also reduced. This reduced number of stored floating gate electrons is critical for reliability because the loss of one electron increases the cell V_{TH} loss. On the other hand, the charge granularity of single electrons affects, at a certain stage, the ability to program narrow V_{TH} distributions. This effect becomes more critical in TLC or later technologies with narrow V_{TH} distributions if one electron causes a significant threshold voltage shift.

(Todo)

Charge trapping (CT) cell, a 3D NAND flash memory cell, is the planar memory cell replacement of the conventional floating gate cell. At first glance, the construction of CT memory cells for

NAND application is not very different from the floating gate NAND cell construction. The major difference is that the charge is stored in a non-conducting dielectric layer with high trap density instead of the conducting floating gate.

The cylindrical shape of the memory cells in 3D cell approaches have one major advantage over fully planar memory cells, namely the electric field enhancement in the TOX and the field reduction in the inter gate dielectric (IGD)
(the cylindrical cell geometry is used in most of the 3D NAND Flash memory arrays)

(Todo)

Encoding more bits per cell increases the capacity of the SSD without increasing the chip size, yet it also decreases reliability by making it more difficult to correctly store and read the bits.

It is worth mentioning that, due to floating gate scalability reasons, charge trap memories are gaining more and more attention, together with their 3D evolution.

(charge trap transistors for 3D)

2.2. NAND Flash Array
We will use schematic diagrams to delineate the NAND flash array from here. **In schematic diagrams, wires are always joined at three-way junctions**. They are joined at four-way junctions only if a dot is shown. **The slash across the input wire indicates that the gate may receive multiple inputs**.

Flash memory for non-volatile data storage was introduced commercially in the mid-1980s. Since then, common ground **NOR and NAND architecture** have become the most common memory array architectures. Traditionally, *NOR flash is used for code storage due to faster memory cell access*. *NAND flash is used for mass data storage because of its higher memory density, enabling higher storage capacities*.

The difference in memory cell area is evident from the schematic diagrams of the NOR and NAND arrays in *Figure 2.6*. In the NOR array, two memory cells share one contact with the ground (SL: Source Line) and one contact with the bit line (BL), as shown in *Figure 2.6*(a). This results in an effective memory cell area of about $10\ \mathrm{F}^2$ for 6 cells. In the NAND array, two or more memory cells are connected in series to form a string, known as the NAND string, which is a relatively compact structure. *Figure 2.6*(b) illustrates the so-called NAND string, comprising up to $m + 1$ memory cells connected in a column. For the NAND string operation, two additional select transistor devices need to be added: (1) Bit Line Selector (BLS), and (2) Source Line Selector (SLS). These additional structures cause the effective cell area consumption to be slightly higher than $4\ \mathrm{F}^2$ for 64 cells, the theoretically smallest effective cell size. However, accessing one individual cell requires going through the other cells in its bit line. This limitation introduces significant noise to the reading process. It also presents challenges to writing, as the adjacent cells in the line should

not be disturbed. For erasing, all cells on the same bit line have to be erased. Note that only NAND flash is a viable option for SSD applications due to the required high memory capacity and bit cost structure.

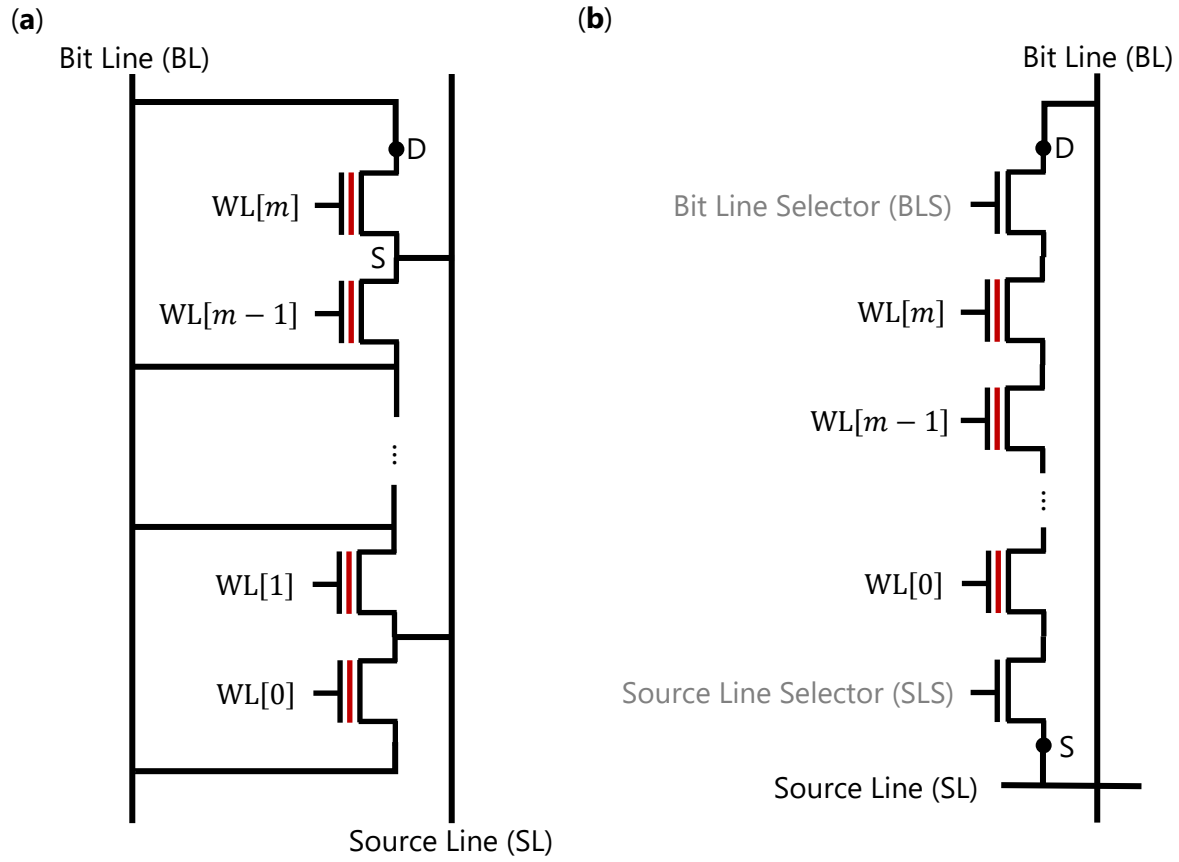


Figure 2.6 Schematic Memory Cell Organization of the NOR Array (**a**) and the NAND Array (**b**). The Word Lines (WL) Run Perpendicular to the Bit Lines (BL).

The basic element of a NAND flash memory, **NAND string** as shown in Figure 2.7(a), has multiple cells (e.g., 128) connected serially, and two selection transistors placed at the edges of the string. BLS connects the NAND string to a bit line, and SLS connects the other side of that string to a source line. The drain select line (or string select line) controls the BLS, and the source select line (or ground select line) controls the SLS. The cell's control gates are connected through the word lines (WLs). The **NAND array** is composed of multiple NAND strings. Figure 2.7(b) shows how the matrix array (i.e., NAND array) is built starting from the basic string. In the WL direction, the adjacent NAND strings share the same WL, DSL (drain select line), SSL (source select line), and SL (a single source line acts as the *common source diffusion*). In the BL direction, two consecutive strings share the bit line contact.

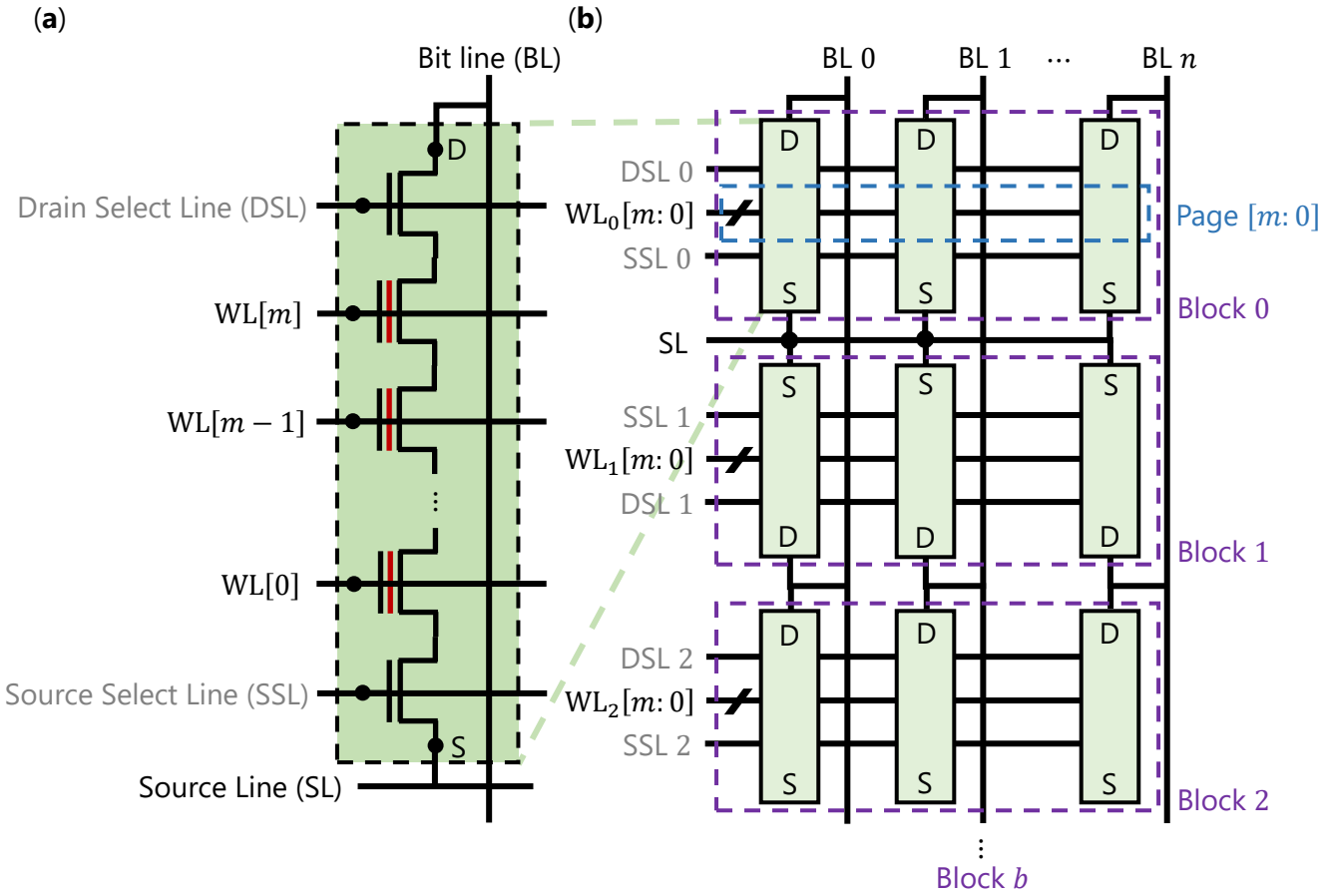


Figure 2.7 NAND String (**a**) and NAND Array (**b**).

All the NAND strings sharing the same group of word lines form a **block**. In Figure 2.7(b), each block is made up of WL$[m: 0]$. A physical NAND **page** is a group of NAND flash cells belonging to the same word line of the same block, which share, horizontally, the same control gate. Thus, a NAND block is composed of several pages. *The number of pages per WL is related to the storage capabilities of the memory cell*. Depending on the number of storage levels, NAND flash memories are referred to in different ways:

- SLC memories store 1 bit per cell.
- MLC memories store 2 bits per cell.
- TLC memories store 3 bits per cell.
- QLC memories store 4 bits per cell.

The number of pages along a word line depends on the storage levels. For example, a WL of TLC memories contains 3 pages.

Supplement materials

The schematic NOR array diagram can also be drawn as Figure 2.*8* shows:

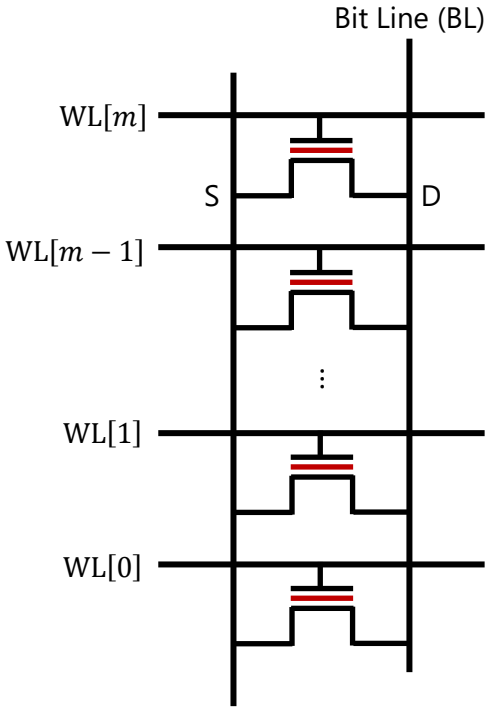


Figure 2.8 NOR Flash Array.

Program, Erase, and Read of Flash Cells in the NAND String

When a large number of floating gate cells need to be operated in the NAND array, it is essential to consider that one floating gate cell is located at every intersection of bit lines and word lines. In particular, the memory cells in the NAND array can no longer be operated independently of each other. In the word line direction (depending on the page size), a couple of thousand FG cells are controlled by the same word line. In bit line direction, the only way to access an individual cell for either reading or writing is through the other cells in its bit line (i.e., the string size (e.g., 64-66 cells) defines the number of cells that cannot be operated independently). This string structure adds significant noise to the read process, and also requires care during the writing process to ensure that adjacent cells in the string are not disturbed. During erasure, in contrast, all cells on the same bit string are erased. Consequently, it is crucial to consider the impact on all neighboring cells when a single cell is treated.

Following *Figure 2.3*, the threshold voltage of each memory cell is carefully adjusted, as shown for SLC and MLC cells in *Figure 2.9*.

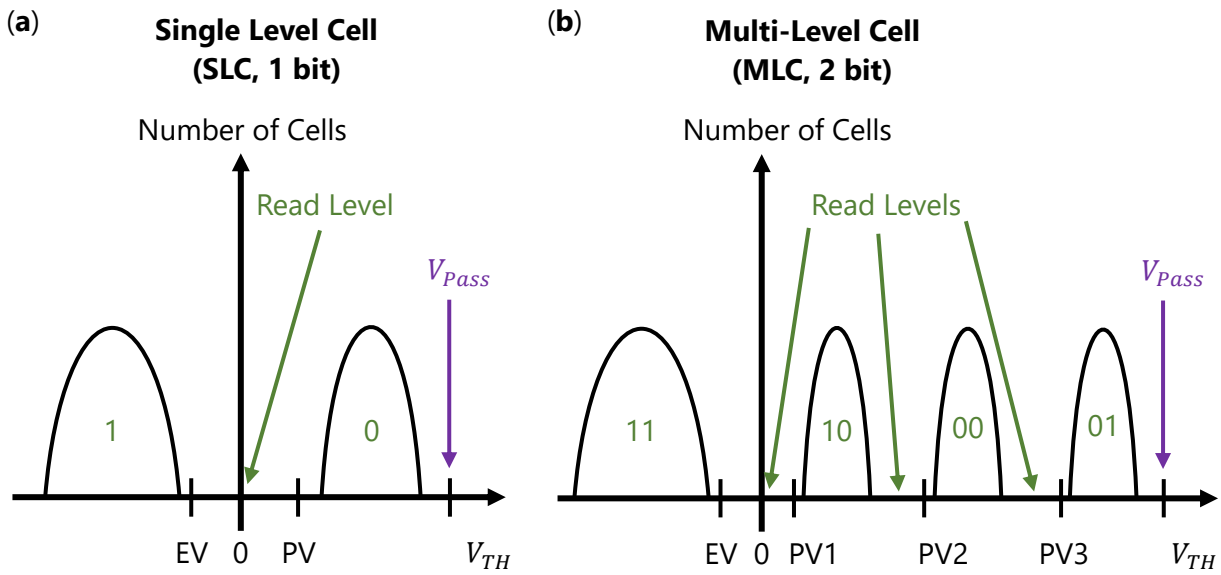


Figure 2.9 Memory Cell Threshold Voltage Distributions for One Bit Per Cell (SLC) Data Storage (**a**) and Two Bit Per Cell (MLC) Data Storage (**b**) in a NAND Flash Array.

The V_{TH} distribution of erased cells is placed at negative V_{TH} values. In an ISPP-like sequence, the erase voltage is increased until all cells are erased below the Erase Verify (EV) level. **The V_{TH} distributions of programmed cells** are placed in the positive V_{TH} range. For a single-level cell (SLC), the ISPP programming is continued until all cells designated for programming are above the Program Verify (PV) level. Consequently, in the case of multi-level cells (MLC), there are three program verify levels (PV1, PV2, and PV3). In addition, the margins between the different programmed V_{TH} distributions must be guaranteed to be large enough to place the read levels (V_{REF}) and have sufficient space for charge/retention loss-caused V_{TH} reductions. A specific distribution shaping algorithm with a small program step increase in certain stages of ISPP programming is necessary to obtain these narrow cell V_{TH}. The pass voltage (V_{Pass}), higher than the maximum possible V_{TH}, makes the cell behave as a pass-transistor.

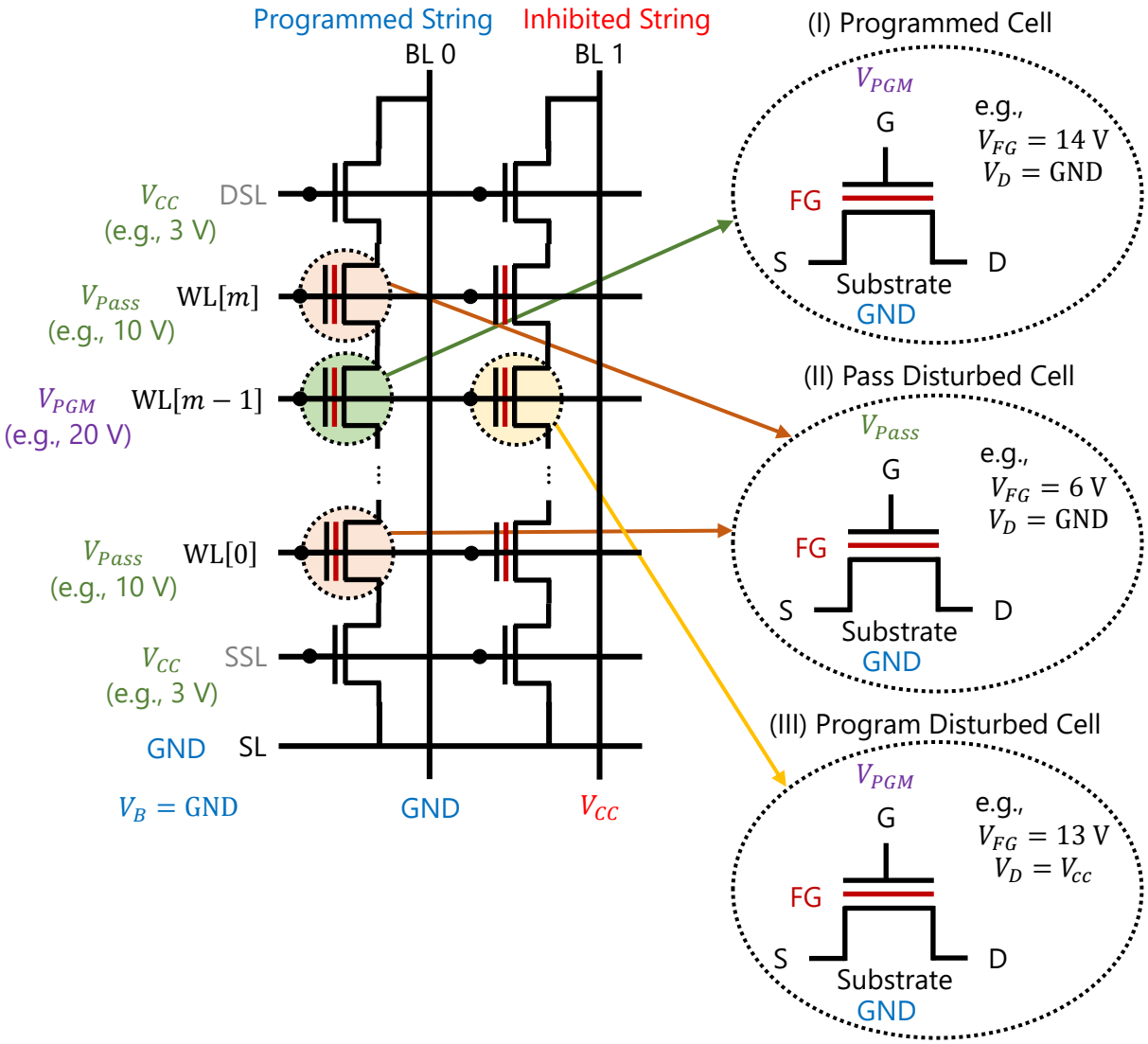


Figure 2.10 Voltage Conditions During Program Operation in the NAND Array. The Memory Cell at the Crossing Point of WL[$m-1$] and BL 0 is Programmed; Several Other Cells are Disturbed by Either Pass Disturb or Program Disturb.

Figure 2.10 shows the voltage condition in the NAND array when the FG cell (selected) at WL[m − 1] and BL 0 is programmed (program operation). For this purpose, a program pulse with the pulse amplitude of V_{PGM} (e.g., 20 V) is applied to WL[m − 1]. To conduct a successful program, it is also required to transfer 0 V (GND) to the channel region (V_D) of the programmed cell as shown *Figure 2.10*(I). Consequently, the 0 V potential is applied to BL 0 and then needs to be transferred to the whole string including the programmed cell at WL[m − 1]. This is done by applying the pass voltage V_{Pass} (e.g., 10 V) to all other word lines so that all other cells (unselected) operate as a resistance.

In principle, **all cells addressed by the same word line could be programmed by this means simultaneously**. However, the programming of arbitrary information requires that specific

memory cells at the word line are excluded from programming. In this example, the cell at the crossing point of BL 1 and $\mathrm{WL}[\mathrm{m}-1]$ represents the cells that should be prevented from programming (program-inhibited cell in *Figure 2.10*). In former FG NAND generations, programming in certain NAND strings was avoided by actively applying a positive voltage to the corresponding bit lines. As a result, the voltage difference between the channel and the control gate was not high enough for programming in these strings. This procedure was complicated, and the voltage pumps used for this purpose required additional power and chip area. Therefore, the so-called "Self-Boosted Program Inhibit" (SBPI) scheme was introduced in recent generations. The principle of the SBPI scheme is that the channel potential in the inhibited strings is not actively raised by applying a voltage but capacitively raised.

Pass disturb and program disturb result in a threshold voltage increase during program operation. The pass cells, located in a string with a memory cell dedicated to programming, experience soft programming when the pass voltage is increased beyond a certain limit (pass disturbed cell in *Figure 2.10*(II)). The program cells, located in a word line with one or more memory cells dedicated to programming, also experience soft programming when the program voltage is high enough to weaken the inhibiting effect.

Recalled that the NAND flash cell has to be erased before writing data to it (Section 2.1), also known as **erasing before writing**. The NAND flash erase operation erases an entire block at once to reduce chip area and lower costs. The voltage conditions during the erase operation are shown in *Figure 2.11*. All word lines are at ground potential ($V_{CG} = 0\ \mathrm{V}$), and the erase voltage is applied to the well of that block. The select transistors, the bit line, and the source line must be left floating during the erase operation. For this purpose, the usually grounded source line needs to be disconnected from the ground potential. By this means, the source line and the bit line, and to a certain extent, the select transistors, can follow the bulk potential to avoid large currents flowing into the source line and the bit line. Due to the improved coupling, the voltage difference between the control gate and the channel required for erasure (e.g., $V_B = 18\ \mathrm{V}$) is lower than the programming voltage when the same voltage is applied to all cells. The erase operation is successful when all cells in the erase block are erased below the EV level, as described above.

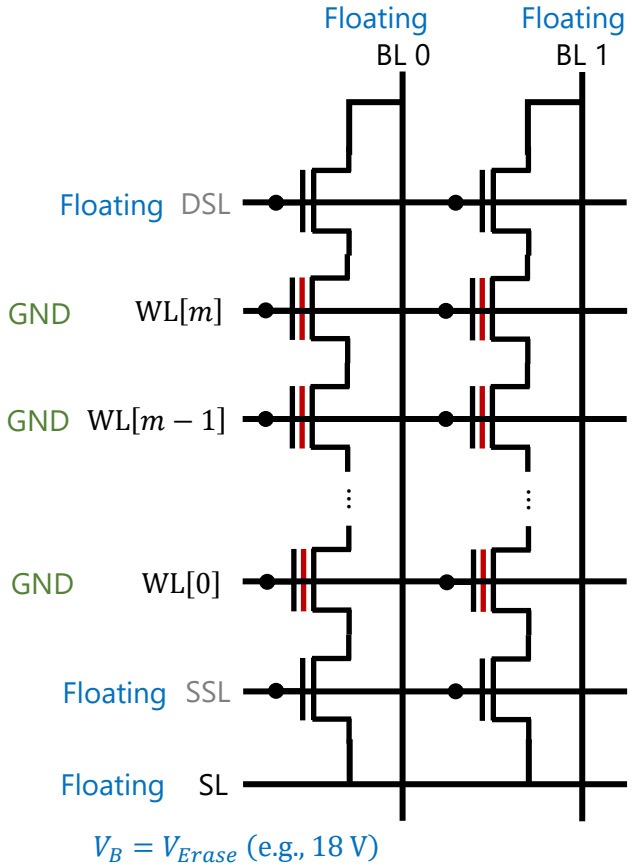


Figure 2.11 The Erase of Floating Gate Cells in the NAND Array is Carried Out Electrically in Separated Erase Sectors. By Applying a Positive Voltage to the Well of the Erase Sector, All Cells are Erased at the Same Time.

Because the granularity of erase operation is usually larger than the program operation (block > page), the in-place write on a page is very inefficient. As a result, SSD does **out-of-place write** and conducts garbage collection to recover free blocks (we will mention it later).

The read operation addresses specific memory cells within the array at a word line unit. For a current sensing read scheme, the bit line selected for a read operation (BL 1 in *Figure 2.12*) is set to the read voltage (e.g., V_{CC}), the word line at the read cell is set to the V_{REF}, and all other word lines are set to the pass voltage. If the cell at $\mathrm{WL[m-1]}$ in the string of BL 2 is erased, the reading current can flow through all cells in that string, and vice versa. The recent sensing technique integrates the cell current on a dedicated capacitor to sense the current flow. We will explain the basic structure later. Note that only one cell can be read at a time.

The pass voltage can also change the threshold voltage (read disturb) when the number of read operations is high enough. *Figure 2.12*(II) shows that the cells of a string in BL 0 are under soft programming due to read disturb. For SLC FG NAND cells, it is assumed that 10^6 read operations with each 15 μs duration need to be guaranteed without read fails. This results in a total disturbance of about 15 seconds.

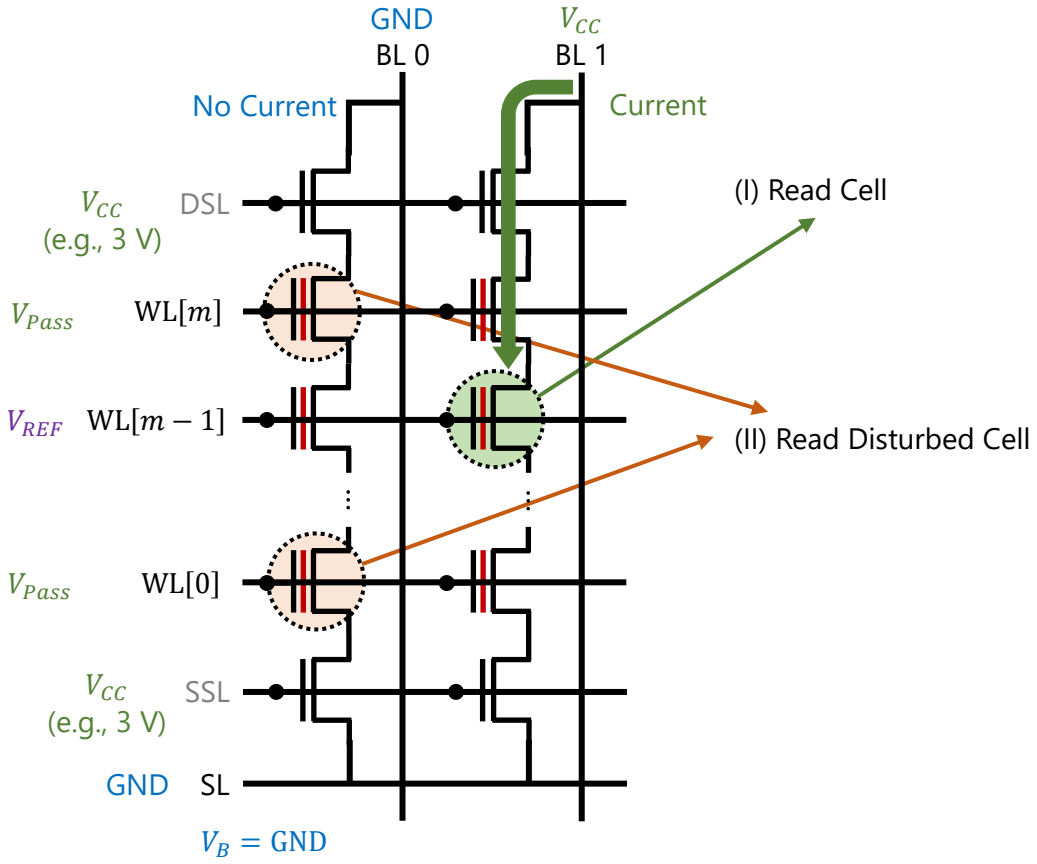


Figure 2.12 Read Operation in the NAND Flash Array.

Figure 2.*13* shows the basic structure of the recent current sensing technique and its related timing diagram. It integrates the cell current on a sense-out (SO) capacitor and utilizes the sense amplifier to sense the current flow in a fixed time (i.e., read the analog information stored in the memory cell). The read mechanism consists of three phases: 1) precharge, 2) evaluation, and 3) discharge. In the precharge phase ($T_{Precharge}$), a NAND flash chip charges each target BL and its sense-out (SO) capacitor C_{SO} to a specific voltage (e.g., $V_{PRE} = V_{CC}$) by turning on the precharge (PCH) transistor. The chip also applies the read-reference voltage V_{REF} to the target cell at the same time mentioned before, which enables the BL to sink current (I_{Cell}) through the NAND string depending on the cell's V_{TH} level (i.e., the BL can sink current when $V_{TH} < V_{REF}$). The chip then enters the evaluation phase ($T_{Evaluation}$), disconnecting the BL from the device's power supply voltage by turning off the PCH transistor and enabling the sense amplifier (SA). If the target cell had been programmed (i.e., $V_{TH} > V_{REF}$), the capacitance of C_{SO} would hardly have changed as the BL could not sink current. In contrast, if the cell had been erased, the charge in C_{SO} quickly flows through the BL as the cell current (I_{Cell}), which rapidly decreases the SO-node voltage below the SA's reference voltage V_{Sense}. As a result, the SA can tell whether the V_{PRE} is smaller or larger than the V_{Sense} via the voltage comparator and output the result through the latch (or flip-flop). Finally, the chip discharges the BL to return to the initial state for future operations in the discharge phase ($T_{Discharge}$).

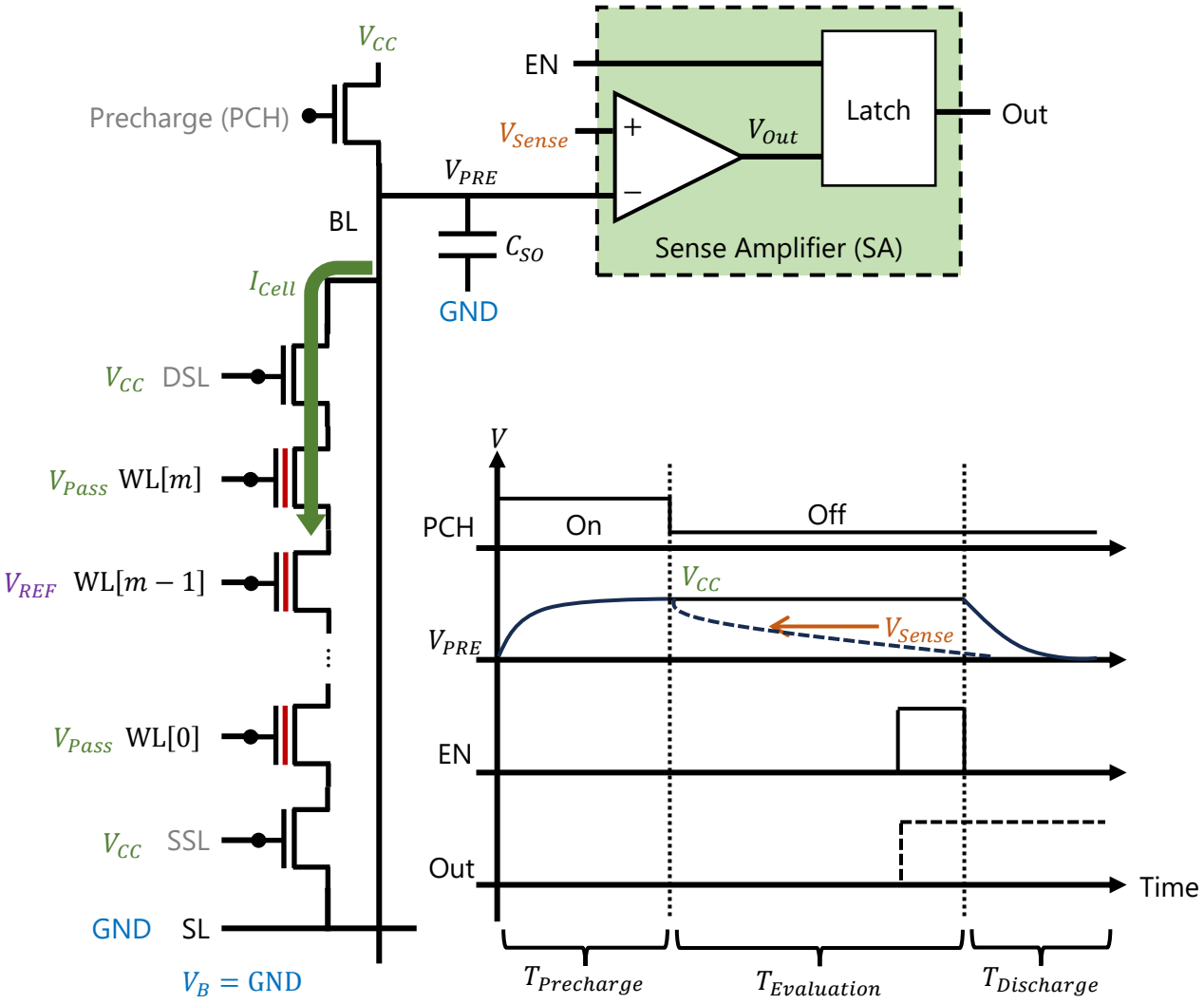


Figure 2.13 Basic Structure of Recent Current Sensing Technique and Its Related Timing Diagram.

Put it all together, the unit sizes of write/read operation and erase operation are different (i.e., asymmetry in operation units): a physical page is the smallest addressable unit for reading and writing, and a physical block is the smallest erasable unit in NAND flash memories. A block must be erased before writing data to it (i.e., erase-before-write property).

Scaling Challenge of Floating Gate NAND Memory Cells

The NAND flash memory scaling of the last 15 years (from 2018) was accomplished by reducing the cell dimensions, whereas the cell construction principle was unchanged. The effective cell size of NAND Flash in 1995 was in the range of 1 μm^2, which resulted in a product chip memory capacity of 32 Mb. In 2010, the cell size was reduced to 0.0028 μm^2 with a chip capacity of 64 Gb. This substantial reduction of cell geometry leads to scaling issues, which are discussed below.

a) Scaling Limitation of the Floating Gate Cell Geometry

For the programmability of floating gate cells, it is important to have an enhanced control gate in the floating gate area, where a control gate is wrapped around the floating gate. A space between adjacent floating gates is required to build an enhanced control gate.

b) Cell-to-Cell Interference

A general problem for floating gate NAND cells in technology generations below 50 nm is the **cell-to-cell cross-coupling**, also called floating gate cross-coupling or floating gate interference. Figure 2.*14* shows this effect: the direct coupling from one floating gate to the nearest neighboring floating gates. This direct coupling increases as NAND flash memory cells are scaled down (or reduced dimensions) because the cells move closer together, increasing the relative coupling capacitance (i.e., cell-to-cell parasitic capacitance). The most significant part is the FG-to-FG coupling in the direction along the bit lines (y-direction in Figure 2.*14*). The reason is that the floating gates directly face each other with the full FG height and full FG width in this direction. Consequently, $C_{FG,y}$ is the largest of the FG-to-FG coupling capacitance terms. In the direction along the word lines (x-direction), parts of the FG-to-FG coupling are screened by the control gate, and therefore, $C_{FG,x}$ is typically smaller than $C_{FG,y}$. The diagonal coupling components $C_{FG,xy}$ and $C_{FG,yx}$ are typically the smallest ones.

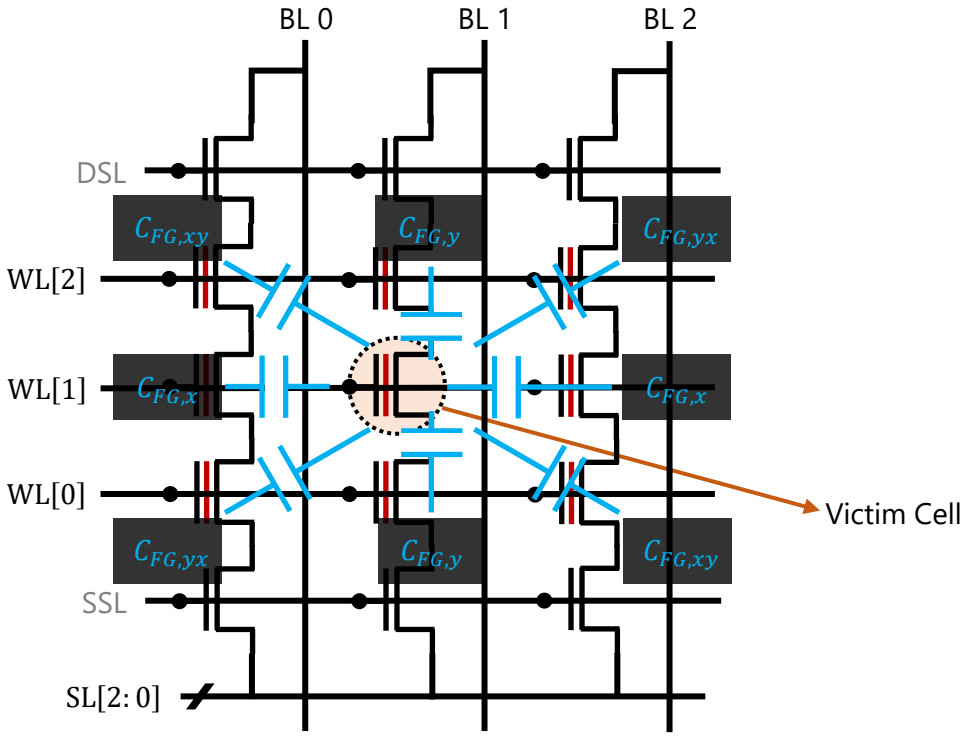


Figure 2.14 Floating Gate Cross-Coupling in Scaled NAND Flash Technologies.

Cell-to-cell interference causes threshold voltage variation (ΔV_{TH}) of a victim cell during the read operation. The reason is that the programming of a cell can change the threshold voltage of a directly neighboring cell (i.e., victim cell), which was already programmed or erased. When a neighboring cell changes its V_{TH} (e.g., during programming or erasing), the capacitive coupling

causes a shift in the V_{TH} of the victim cell (or interfered cell). This shift can affect the victim cell's ability to read correctly or lead to data corruption.

Therefore, when shrinking the FG NAND flash dimensions, the program algorithm needs to take care of the floating gate cross-coupling issue at a certain point. The strategy is to reduce the number of neighboring programmed cells after reaching the final programming target V_{TH} of each cell, in combination with reducing the amount that these neighboring cells increase their V_{TH}. For example, we can improve a standard program algorithm to program cells in a particular order. This algorithm involves writing pages within a block sequentially or incrementally, a concept known as **sequentiality of programming** (SOP). In a multi-level cell flash memory, two bits are programmed in one cell. Among the bits, a lower bit is denoted as the least significant bit (LSB), and an upper bit is denoted as the most significant bit (MSB). The LSB pages within a block should be programmed before the MSB pages. Random page address programming is prohibited.

Cell-to-cell interference during the erase operation is uncritical because all cells (in a block) are erased simultaneously and therefore all cells return to the ER state.

c) Word Line to Word Line Leakage Current
During a program operation, the reduced cell-to-cell distances with scaled dimensions cause strongly increased electric fields between neighboring word lines. Besides, high WL voltage differences during program operation become critical since the programming voltage does not scale but increases slightly. As a result, electrons can tunnel from a programmed floating gate to the control gate on the high program voltage V_{PGM} or generally introduce WL-to-WL leakage currents. Note that the WL-to-WL voltages during erase are uncritical because all cells are erased at the same time, and therefore, all word lines are at the same potential.

Options to reduce WL-to-WL leakage using a special program algorithm include limiting the difference voltage between adjacent word lines.

d) Random Telegraph Noise
Different types of field effect devices have observed random telegraph noise (RTN). It can be explained by electron capture and emission processes in oxide traps close to the channel of a MOSFET device. RTN in the string current results in a threshold voltage shift ΔV_{TH}, which can cause read failures.

3D NAND Flash Memories
Planar memory cells have been scaled for decades to achieve larger capacity by improving process technology, circuit design, programming algorithms, and lithography. However, when approaching a minimum feature size of 1x-nm (Feature size refers to the smallest physical dimension of a component during manufacturing, measured in nanometers), more challenges pop up: doping concentration in the channel region becomes difficult to control, RTN (Random telegraph noise mentioned in 2.2 (d)) and electron injection statistics widen threshold distributions, thus causing a significant hit to both endurance and retention. Furthermore, by reducing the distance between memory cells, the intra-word line electric field becomes higher (2.2 (c)), pushing

the bit error rate to an even higher level. 3D arrays leverage either floating gate (FG) or charge trapping (CT) technologies to fuel a further increase in the bit density. In fact, the vast majority of 3D architectures published to date are built with CT cells, primarily due to the simpler fabrication process. Nevertheless, the floating gate remains, and commercial products have successfully integrated it into a 3D array.

Identifying the right way for going 3D is not so easy though. With 3D architectures, the "simple" reduction of the minimum feature size is running out of steam, as shown in Figure 2.*15*: a higher number of stacked cells is the only hope for dramatically reducing the real estate of a stored bit.

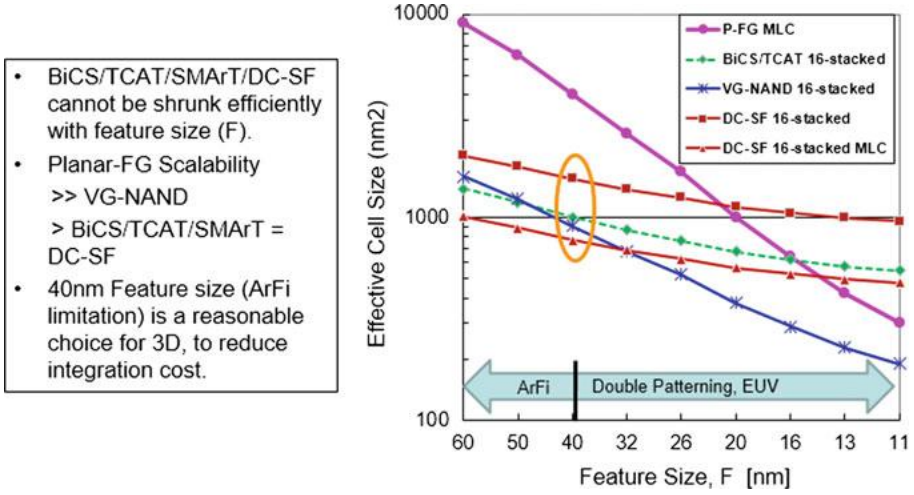


Figure 2.15 3D NAND Flash Scaling.

3D arrays can be efficiently built by vertically rotating the planar NAND flash string shown in Figure 2.*16*. The solution of choice is a conduction channel completely surrounded by the gate: indeed, the curvature effect helps increase the electric field E_t across the tunnel oxide, and reduces the electric field E_b across the blocking oxide, and this has a positive impact on oxide reliability and overall power consumption.

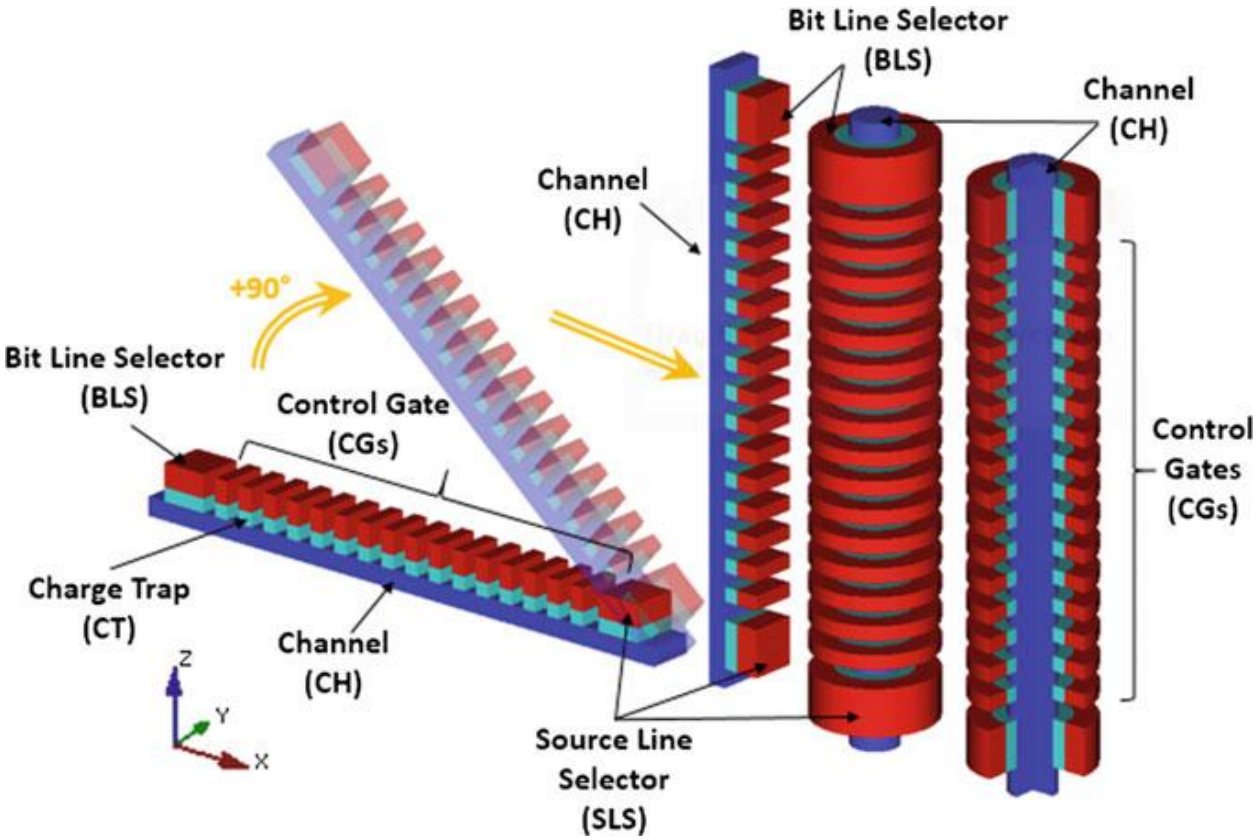


Figure 2.16 NAND Flash String Goes Vertical.

Vertical channel arrays have been historically driven by architectures known as **BiCS**, which stands for *Bit Cost Scalable*, which leverages CT cells. Let us begin with BiCS, which is illustrated in Figure 2.*17* and Figure 2.*18*. There is a stack of control gates (CGs), the lowest being the one of the source line selectors (SLSs). The whole vertical stack is punched through, and the resulting holes are filled with polysilicon; each filled hole (a.k.a. pillar) forms a series of memory cells vertically connected in a NAND fashion. Bit line selectors (BLSs) and bit lines (BLs) are formed at the top of the structure. As usual, a select transistor (BLS) is used to connect each NAND string to a bit line; there is also another select transistor (SLS), which connects the other side of the string to the common source diffusion.

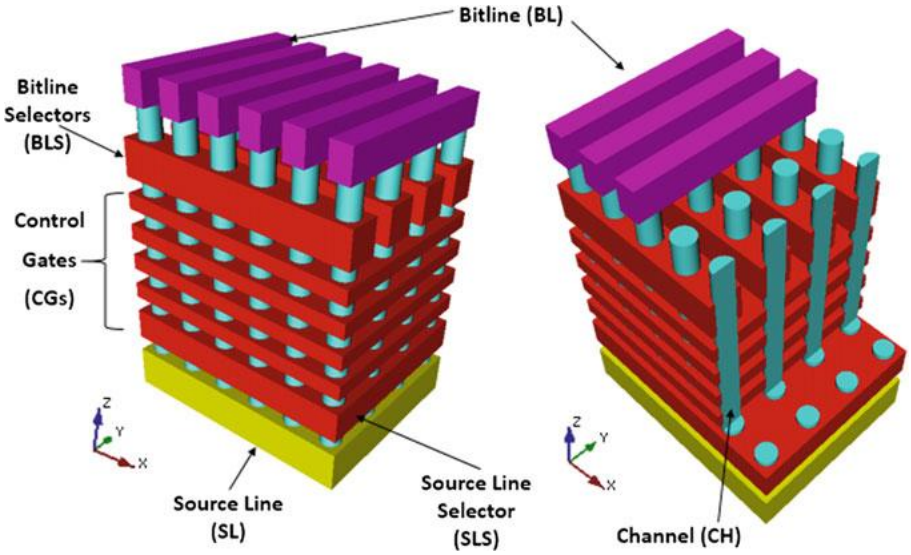


Figure 2.17 BiCS Architecture.

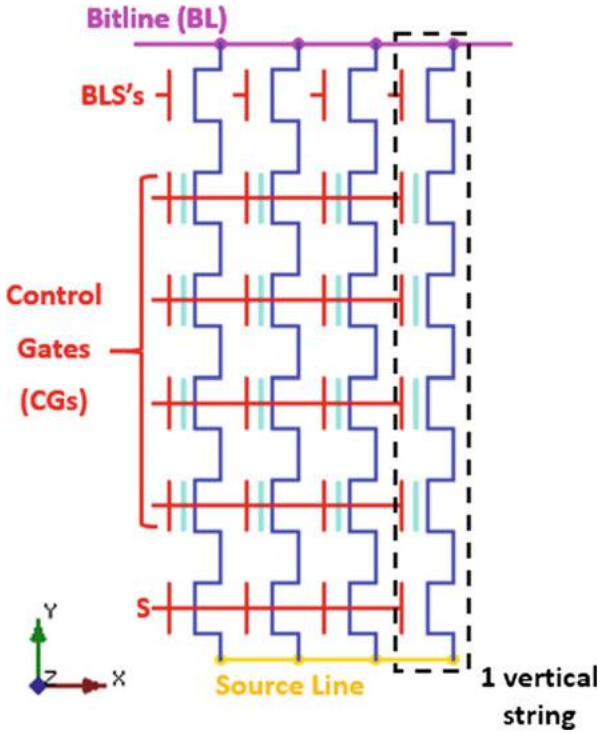


Figure 2.18 Equivalent Circuit of a BiCS Array.

As sketched in Figure 2.*19*, vertical transistors have a poly-silicon body, and this fact turned out to be one of the critical cornerstones of the 3D foundation. From a manufacturing perspective, the density of the traps at the grain boundary is difficult to control due to its vertical shape. The poor control of this density induces significant fluctuations in the characteristics of vertical transistors.

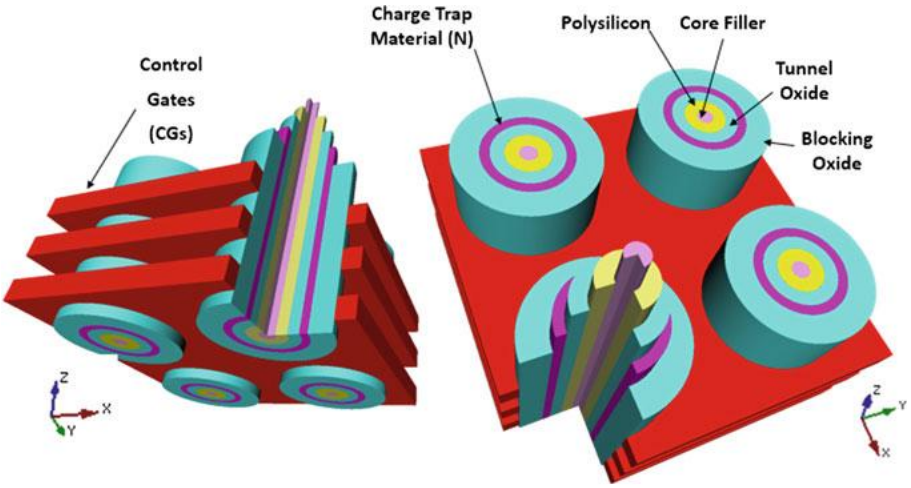


Figure 2.19 BiCS Memory Cells.

The recipe for fixing the trap density fluctuation problem is to manufacture a poly-silicon body much thinner than the depletion width. In other words, by shrinking the poly-silicon volume, the total number of traps decreases (Figure 2.*20*). This particular structure is typically referred to as a *Macaroni Body*. A filler layer (i.e., a dielectric film) is used in the central part of the *Macaroni* structure because it makes the manufacturing process easier.

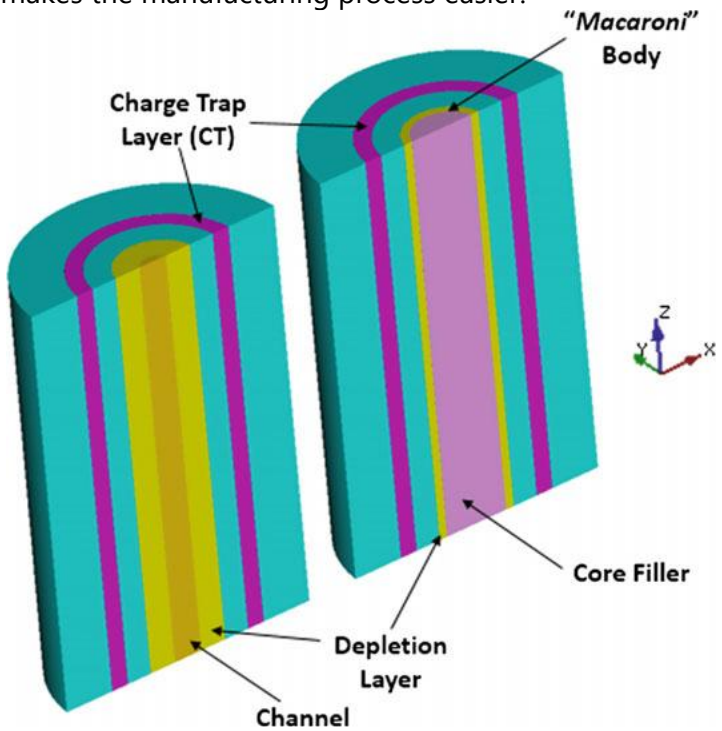


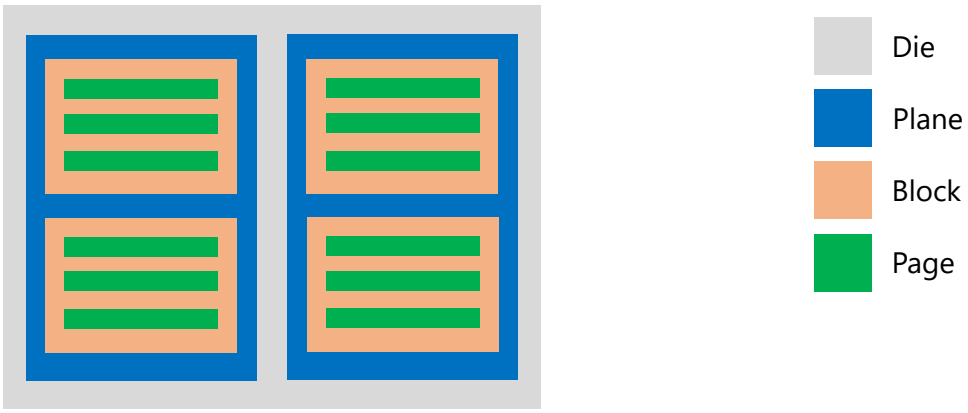
Figure 2.20 A Vertical Transistor (right) Modified with *Macaroni* Body (left).

Besides BiCS, many other approaches were tried, including P-BiCS, VRAT (Vertical Recess Array Transistor), Z-VRAT (Zigzag VRAT), VSAT (Vertical Stacked Array Transistor), TCAT (Terabit Cell Array Transistor), V-NAND, and 3D-VG (Vertical Gate) NAND which is a unique architecture where the channel runs along the horizontal direction.

(Note a 3D layer means gate layer in 3D NAND flash memories)

Memory Logic Organization

Recall that a NAND block contains several pages, and pages from different word lines share the same bit lines. It means that only pages on the same word line can be accessed simultaneously. To achieve higher performance, several NAND blocks form a **plane** where cells operate together. NAND flash devices may contain independent flash planes with their own buffer to hold data, allowing simultaneous operations for higher performance (i.e., multi-plane parallelism). This technique has limited effectiveness due to the constraints of the same word line address for required operations. To tackle that problem, planes form a **die**. A die is the minimum unit that can independently execute commands and report its status. It contains at least one page buffer and a flash array (The number of page buffers is dependent on the number of plane operations supported for the die). Multiple dies can exist within a single NAND flash chip, allowing for increased parallelism and concurrency by enabling simultaneous operations on different dies (although they compete for the NAND flash chip pins). Figure 2.*21* illustrates the NAND flash layout of SLC and MLC memories. A page is the smallest unit that can be programmed (written to) and read, typically ranging in size from 8 to 16 KiB. A block is the smallest NAND die portion that can be erased.



(**a**) SLC Memory

(**b**) MLC Memory

Figure 2.21 NAND Flash Layout of SLC (**a**) and MLC (**b**) Memories.

Furthermore, a **die** within a NAND flash device can also be referred to as a **logical unit number** (LUN). For example, it is permissible to start a program operation on LUN 0 and then, prior to the operation's completion, to start a read operation on LUN 1.

Each page has a fixed-size main data area and a spare data area, as shown in Figure 2.*22*. The data area can be 4, 8, or 16 KiB and is used for storing data (e.g., user data). On the other hand, the spare area is in the order of hundreds of bytes for every main data area and is used for storing ECC (Error Correction Code).

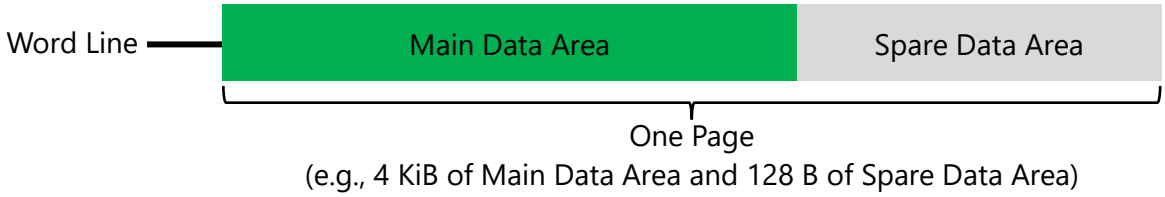


Figure 2.22 Logic Organization of a Page.

(todo)

2.3. Reliability of NAND Flash Memory

The basic parameter characterizing the NAND flash memory reliability is the **raw bit error rate** (RBER), representing the fraction of erroneous bits retrieved during a read operation (todo).

Error Mitigation

Because of poor data retention, endurance problems or read disturbs, the actual threshold voltage read in a cell may be different from the programmed one. 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.

NAND flash memories utilize an ECC (Error Correction Code) scheme to correct errors detected when reading the memory. 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 (e.g., SLC, …, QLC), the error probability increases with the number of bits stored in a single cell.

In the first SLC memories, thanks to the large gap between the program and the erase 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 or 2 KiB 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.

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 affect significantly on the overall read time. Consequently, the correction time may become the bottleneck of the entire read procedure. 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. 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 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; (ii) develop ECC solutions based on different coding concepts, like Low Density Parity Check (LDPC) codes. In the former case, the basic idea in the presence of uncorrectable pages consists in re-reading 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: the ECC engine automatically manages them 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 [28]. Figure 7.3 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).

(todo).

2.4. Integrated Circuit Architecture

The operations of NAND flash memories require a mix of analog and digital circuits that need to be properly and timely driven. Starting from a generic floorplan of a NAND memory, we guide the reader through the main building blocks.

With few exceptions, today's NAND flash chips correspond to the block diagram in Figure 2.*23*. I/O (Input/Output) refers to the shared pins and signals used to communicate with the NAND flash controller (or external host), transferring commands, addresses, and data. High-speed NAND introduced a double data rate (DDR) interface in 2008. Now, two major solutions are available in the market. One is the ONFI interface introduced by the ONFI (Open NAND Flash Interface) organization, including SK Hynix, Intel, Micron, SanDisk, Phison, Sony, and others; another one is toggle interface (i.e., toggle-mode DDR interface) introduced by Samsung and Toshiba. Note that, as usual, JEDEC (Joint Electron Device Engineering Council) is working on combining the above interfaces in a single standard.

The data path is a path through which data flows through the chip. To reduce transmission time on the data path, the NAND flash chip uses a pipeline for data transfer. Specifically, data transfer consists of two parts. The first part is transferring data over the external bus to or from the page buffer. The second part is between the page buffer and the flash arrays. The page buffer is a

temporary storage area (SRAM) that holds data during read and write operations. It often consists of a page register (for transferring data to and from the flash array) and a cache register (for transferring data to and from the host and ECC processing). The page register temporarily stores data to be written to the flash array or receives data read from it. At the same time, the page buffer, as a whole, manages the page-based read/write flow, often employing techniques such as internal ECC processing to improve performance.

The row decoder is the block responsible for addressing and biasing each word line, while the column decoder selects the correct page buffers (Note the byte location within the page buffer is referred to as the column). Finally, the brain of the memory is the control and address logic, which has multiple functionalities to orchestrate the communication between the external host and the data inside the device:

- Microcontroller. It stores and executes all the internal algorithms for NAND flash memory, including read, program, erase, and test mode operations.
- Command Interface (CI). It is the interface that understands legal or illegal command sequences between the NAND flash memory and the external user.
- Test Interface (TI). It is used when we want to test specific features that are usually not accessible during normal operations (User mode).

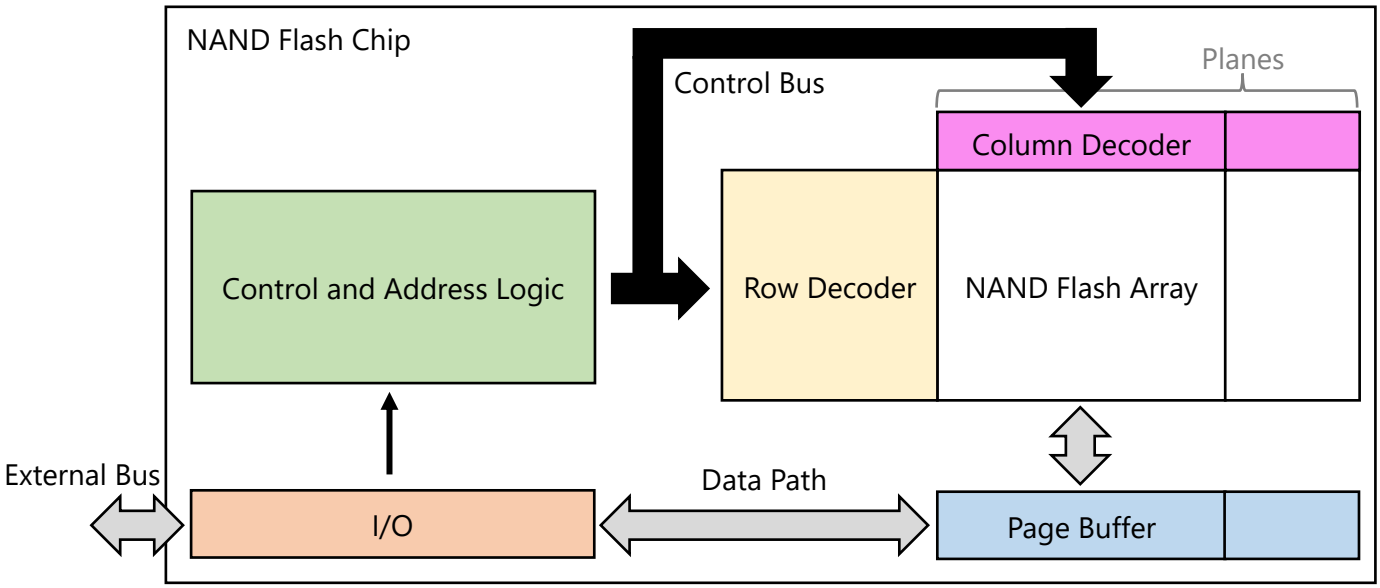


Figure 2.23 Typical Flash Chip Architecture.

Note that NAND flash package (or device) is a structure that can combine NAND chips, controllers, and passive components for miniaturization and simplified system design.

NAND Interface

A standardized NAND flash device interface provides a means for a system to be designed that supports a range of NAND flash devices without direct design pre-association. To enable the broad use of NAND in today's performance-demanding applications, the NAND flash interface

specification also needs to provide a means for a system to seamlessly use new NAND devices that may not have existed when the system was designed.

Three NAND flash interfaces are available for NAND devices. They are the Asynchronous SDR (Single Data Rate), Synchronous DDR (Double Data Rate), and Toggle DDR. The Asynchronous SDR is the first NAND flash device interface, also known as Legacy, which uses an asynchronous data transfer scheme (i.e., the timing is not controlled by a clock). This interface utilizes an 8-bit(b) bidirectional data bus for I/O signaling (DQ) to deliver a byte (B) within a transfer time of 25 nanoseconds (ns). The asynchronous interface limits data transfer to 40 MB/s. NAND read throughput is determined by flash array access time and data transfer across the DQ bus. As technology shrinks, page size increases, and data transfer takes longer, NAND read throughput decreases, thereby unbalancing the ratio between array access time and data transfer on the DQ bus. Therefore, DDR-like interfaces have been introduced to balance this ratio.

The Synchronous DDR (Double Data Rate), also known as the Open NAND Flash Interface (ONFI), is developed and maintained by the ONFI Working Group, including SK Hynix, Intel, Micron, SanDisk, Phison, Sony, and others. On the other hand, the Toggle DDR is published by Samsung Semiconductors and Toshiba Memory Corporation (TMC). They use a synchronous data transfer scheme to synchronize signals with a continuous clock signal. While distinct differences exist between the two standards, a well-recognized "common ground" is the JESD230 standard, published by the Joint Electron Device Engineering Council (JEDEC), which provides interoperability for a system to be designed that can support the Asynchronous SDR, Synchronous DDR, and Toggle DDR.

Common terminology of the NAND flash device interface standards is listed below (Table 2.*1*). The default explanation is referenced from the JESD230 standard.

Table 2.1 Common Terminology of NAND Flash Interfaces.

Active-low signals:	While the preferred method for indicating a signal that is active when it is low is to use the overbar, as in $\overline{CE}$, the difficulty in producing this format has led to several alternatives intended to be equivalent. These are the use of a CE reverse solidus (\) or the trailing underscore (_) following the signal name, as in CE\ and CE_. In this publication, "_n" is used to indicate an active low signal (i.e., an inverted logic sense).
NAND non-volatile memory device	The packaged NAND non-volatile memory unit containing one or more NAND targets.
Target (chip):	A non-volatile memory component with a unique chip enable (CE_n) select pin. A device (or package) contains one or more targets. A target is controlled by one CE_n signal. A target is organized into one or more logical units (LUNs).
LUN (logical unit number):	The minimum memory array size that can independently execute commands and report status.

Address:	A character or group of characters that identifies a register, a particular part of storage, or some other data source or destination. The address is comprised of a row address and a column address. The **row address** identifies the page, block, and LUN to be accessed. The **column address** identifies a bit (x1 devices), byte (x8 devices), word (x16 devices), or Dword (x32 devices) location within a page to access. The least significant bit (LSB) of the column address shall always be zero (nature of double data rate) for the source synchronous data interface (NV-DDR, NV-DDR2, NV-DDR3, or NV-LPDDR4). The least significant bit of the column address in a DDR memory system is zero because a DDR transfer sends two bytes of data simultaneously. Making the LSB of the column address zero ensures that the burst transfer always starts at an even address, allowing the system to read two bytes (or multiples of two) at once. This synchronization between the address and the double data rate transfer is what improves performance and efficiency.
Block:	A continuous range of memory addresses. A block consists of multiple pages and is the smallest addressable unit for the erase operation.
Page:	The smallest non-volatile memory array segment, within a device, that can be addressed for read or program operations.
Page register:	A register used to transfer data from a page in the memory array for a read operation or to transfer data to a page in the memory array for a program operation. The number of page registers is dependent on the number of plane operations supported for the LUN.
Status register (SR[x]):	A register within a particular LUN containing status information about that LUN. SR[x] refers to bit "x" within the status register.
Word (x16):	A sequence of 16 bits that is stored, addressed, transmitted, and operated on as a unit within a computing system.
Dword (x32):	A sequence of 32 bits that is stored, addressed, transmitted, and operated on as a unit within a computing system. A Dword may be represented as 32 bits, as two adjacent words, or as four adjacent bytes. When shown as bits, the least significant bit is bit 0 and the most significant bit is bit 31; the most significant bit is shown on the left. When shown as words, the least significant word (lower) is word 0, and the most significant (upper) word is word 1. When shown as bytes, the least significant byte is byte 0 and the most significant byte is byte 3.
DDR:	Acronym for "double data rate".

Read request:	For NAND non-volatile memory, a read request is a data output cycle initiated by the host, resulting in data transfer from the device to the host.
Source synchronous:	For NAND non-volatile memory, source synchronous is the operation in which the **strobe signal (DQS)** is transmitted with the data to indicate when the data should be latched. The strobe signal (DQS) is similar in concept to an additional data bus bit.
Latching edge:	The rising or falling edge of a waveform that initiates a latch operation. For a NAND non-volatile memory, the latching edge is the edge of the CK or DQS signal at which the contents of the data bus are latched for the source synchronous data interface. For NAND non-volatile **data cycle**, the latching edge is both the rising and falling edges of the DQS signal. For a NAND non-volatile **command and address cycles**, the latching edge for the source synchronous interface is the rising edge of the CK signal.
NAND defect area:	A portion of either the first page and/or the last page of the marked defect block, and the defect area per page is defined as (# of data bytes) to (# of data bytes + # of spare bytes -1). For 8-bit data access NAND, the manufacturer shall set the first byte in the first or last page of the defect block to a value of 00h. For 16-bit data access NAND, the manufacturer shall set the first word in the first or last page of the defect block to a value of 0000h.
N/A:	Not applicable. Fields marked as "na" are not used.
O/M:	Optional/Mandatory requirement. When the entry is set to "M", the item is mandatory. When the entry is set to "O", the item is optional.
Reserved:	A keyword indicating reserved bits, bytes, words, fields, and opcode values that are set aside for future standardization. Their use and interpretation may be specified by future extensions to this or other specifications. A reserved bit, byte, word, or field may be cleared to zero or in accordance with a future extension to this publication. A host should not read/use reserved information.

The relationship between bytes, words, and Dwords in NAND flash device interface standards is illustrated below (Figure 2.*24*).

Byte

Word

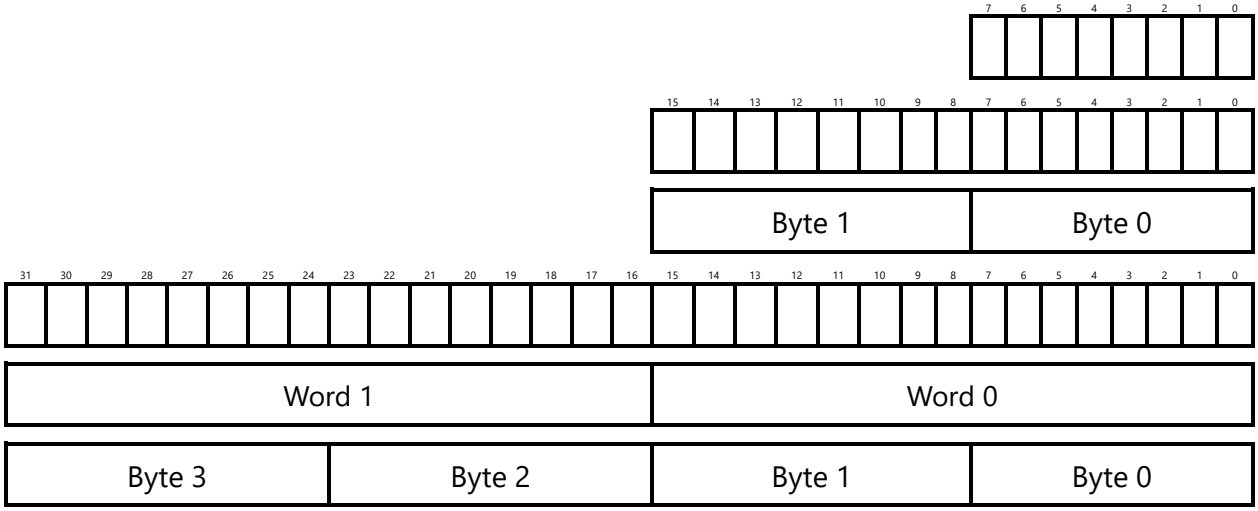


Figure 2.24 Byte, Word and Dword Relationships.

Conventional protocol signal description of the NAND flash device interface standards is summarized below (Table 2.*2*).

Table 2.2 Common Protocol Signal Description of NAND Flash Interfaces.

Name	Input/Output	Description
IO0 ~ IO7 (IO15) DQ0 ~ DQ7	I/O	Data Inputs/Outputs These signals are bidirectional ports used to input commands, address, and data, and to output data during read operations. The signals float to a high-Z (neither HIGH nor LOW) when the chip is deselected or when the outputs are disabled. IO0~IO15 are used in a 16-bit wide target configuration. With multi-channel support, IO0_0~IO7_0 and IO0_1~IO7_1 are used for the I/Os of channel 0 and channel 1, respectively. These signals are also known as DQ0~DQ7 for the Toggle DDR and Synchronous DDR.
R/B_n $(R/\bar{B})$	O	Ready/Busy Output The R/B_n output indicates the status of the target operation. When low, it indicates that one or more operations (program, erase, and read operations) are in progress and returns to a high state upon completion. It is an open-drain output and does not float to a high-Z condition when the chip is deselected or when outputs are disabled.
CE_n $(\overline{CE})$	I	Chip Enable The CE_n input is the target selection control. When CE_n is high and the target is in the ready state, the target goes into a low-power standby state. When CE_n is low, the target is selected.

		If the target is in the busy state, CE_n high is ignored. The target is in the standby mode during the program or erase operations.
CLE	I	Command Latch Enable The CLE signal is one of the signals used by the host to indicate the type of bus cycle (command, address, data). For the Toggle DDR: The CLE input controls the activating path for commands sent to the command register. When active high, commands are latched into the command register through the DQ ports on the rising edge of the WE_n signal.
ALE	I	Address Latch Enable The ALE signal is one of the signals used by the host to indicate the type of bus cycle (command, address, data). For the Toggle DDR: The ALE input controls the activating path for addresses sent to the address registers. When active high, addresses are latched on the rising edge of the WE_n signal.
WE_n ($\overline{WE}$)	I	Write Enable The WE_n input controls write to the I/O port. For the Asynchronous SDR, input data, commands, and addresses are latched on the rising edge of the WE_n pulse. For the Toggle DDR, commands and addresses are latched on the rising edge of the WE_n pulse.
RE_n ($\overline{RE}$)	I	Read Enable The RE_n input is the serial data-out control. For the Asynchronous SDR, data is valid after the falling edge of RE_n. For the Toggle DDR, data is valid after both the falling edge and rising edge of RE_n, which also increments the internal column address counter by one. The RE_n is paired with differential signal RE_c to provide the differential pair signaling to the system during reads.
RE_c	I	Complement of Read Enable This is the complementary signal to the Read Enable signal. Specifically, the Read Enable complement has the opposite value of the Read Enable signal when CE_n is low.
W/R	I	Write/Read Direction The Write/Read Direction signal indicates the owner of the DQ bus and DQS signal in the Synchronous DDR data interface. This signal shares the same pin as RE_n in the asynchronous data interface.

DQS	I/O	Data Strobe The data strobe signal indicates the data valid window for the Toggle DDR and Synchronous DDR data interfaces. Output with read data (driven by the DDR memory), input with write data (driven by the memory controller). Edge-aligned with read data, centered in write data. The data strobe DQS is paired with differential signal DQS_c to provide the differential pair signaling to the system during reads and writes.
DQS_c	I/O	Complement of Data Strobe This is the complementary signal to the Data Strobe signal. Specifically, the Data Strobe complement has the opposite value of Data Strobe signal when CE_n is low,
CK	I	Clock The Clock signal is used as the clock in the Synchronous DDR data interface. This signal shares the same pin as WE_n in the asynchronous data interface.
WP_n ($\overline{WP}$)	I	Write Protect The WP_n disables the flash array program and erase operations, which provides inadvertent program/erase protection during power transitions.
Vcc	I	Power Vcc is the power supply for the device.
VccQ	I	I/O Power The VccQ is the power supply for input and/or output signals.
Vss	I	Ground The Vss signal is the power supply ground.
VssQ	I	I/O Ground The VssQ signal is the ground for input and/or output signals.
VSP	n/a	Vendor Specific The function of these signals is defined and specified by the NAND vendor. Any VSP signal not used by the NAND vendor shall not be connected internally to the device.
Vref	n/a	Reference Voltage This is used as an external voltage reference.
ENi / ENo	I/O	Enumeration Pins These pins may be used for ONFI NAND.
R	n/a	Reserved The host shall not connect these pins.
RFU	n/a	Reserved For Future Use

		These pins may be assigned for specific functions in the future.
NU	n/a	Not Usable A pin that is not to be used in normal applications and that may or may not have an internal connection.
NC	n/a	No (internal) Connection A pin that has no internal connection and that can be used as a support for external wiring without disturbing the function of the device, provided that the voltage applied to this terminal (by means of wiring) does not exceed the highest supply voltage rating of the circuit.
ZQ	n/a	Reference pin for ZQ calibration This is used on ZQ calibration.

a) Asynchronous SDR

The first NAND flash device interface, known as Legacy, used the asynchronous transfer scheme with timing criteria for data transfers. The asynchronous scheme allows triggering of transfer tasks to be wholly dependent on the "Write-Enable (WE)" and "Read-Enable (RE)" control signals, without a clock. In other words, data is latched with the "Write-Enable (WE)" signal for the write operation and the "Read-Enable (RE)" signal for the read operation. In addition, Legacy supports data transfer at Single Data Rate (SDR), processing it on only one edge of the control signals. Specifically, the interface enforces signal alignment only on the rising edges of write signals and the falling edges of read signals, as data is "moved" in and out of devices.

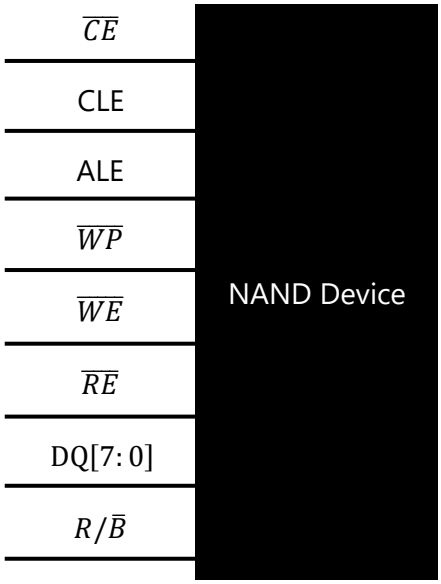


Figure 2.25 Legacy SDR Interface Schematic.

b) Synchronous DDR

c) Toggle DDR

The Toggle DDR interface allowed bidirectional data transfer using DQS strobe signals, with each rising or falling edge corresponding to a data transfer. The Toggle data transfer operates without a clock signal, making it asynchronous and requiring power only during read and write operations.

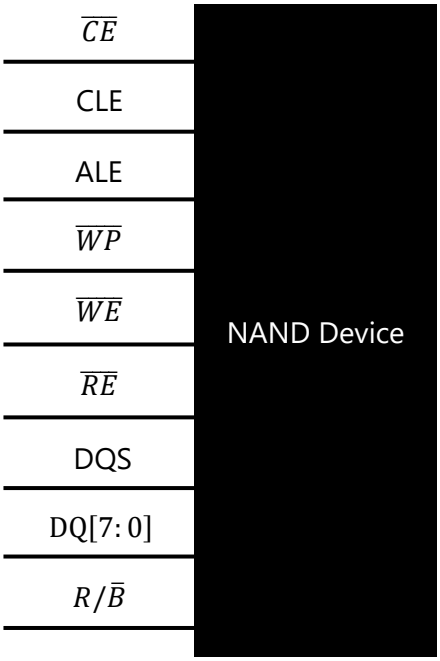


Figure 2.26 Toggle DDR Interface Schematic.

Basic Operation

Below, the basic operations of toggle DDR are explained.

Page Read Operation

The Page Read function reads a page of data identified by row address for the selected LUN. The data page is made available for reading from the page register, starting at the specified column address. Figure 2.*27* defines the Page Read behavior and timings.

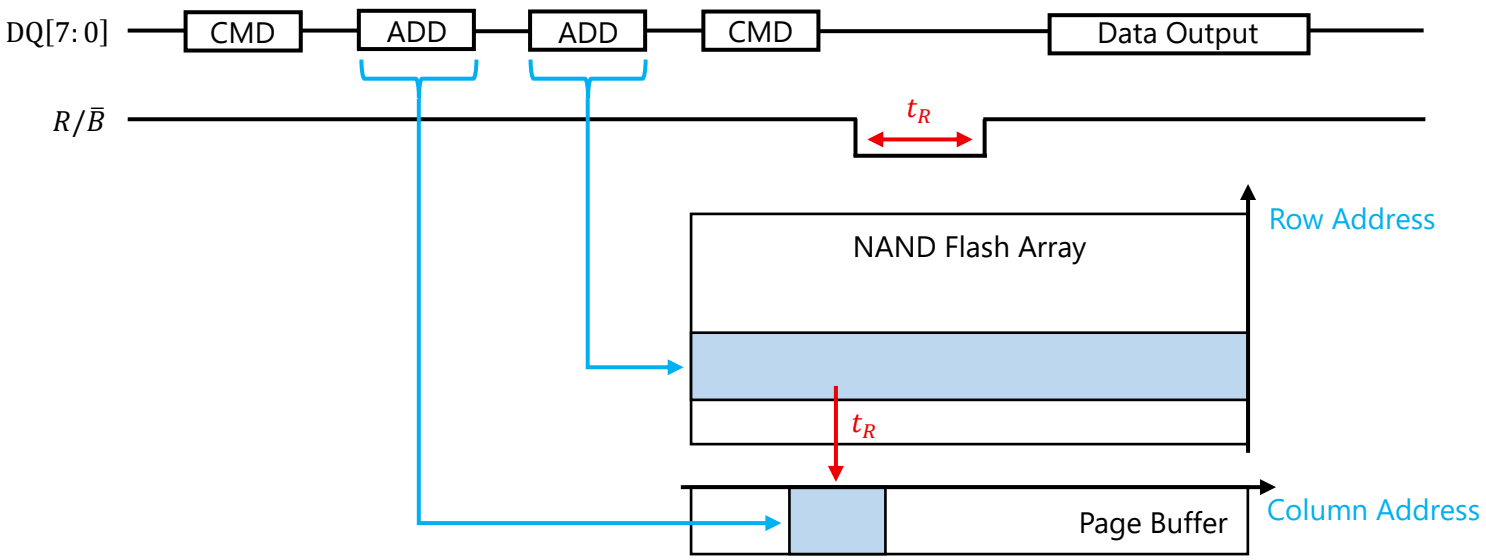


Figure 2.27 Page Read Operation.

Full Sequence Program (FSP) Operation

For the device that is programmed on a three-page basis (e.g., LSB, CSB, MSB pages for the TLC device), each page shall be programmed only once before being erased. The WL address of programming shall be ascending order within a block. The contents of the page register are programmed into the flash array specified by row address.

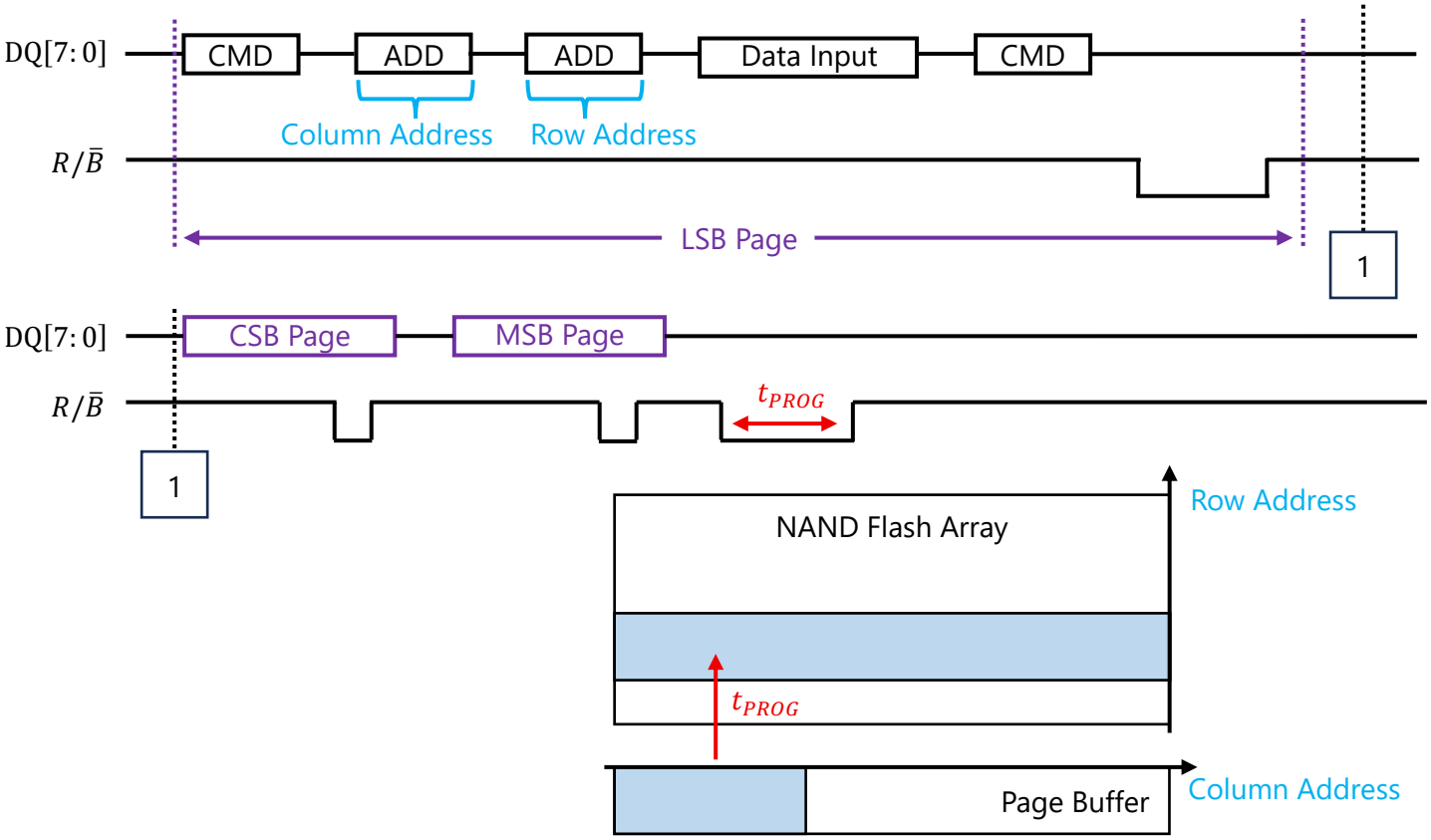


Figure 2.28 Full Sequence Program Operation.

Block Erase Operation

The Block Erase operation is done on a block basis. Only cycles of row addresses are required for the Block Erase operation, and a WL address within the cycles is ignored while the plane and block addresses are valid. After the Block Erase operation passes, all bits in the block shall be set to one.

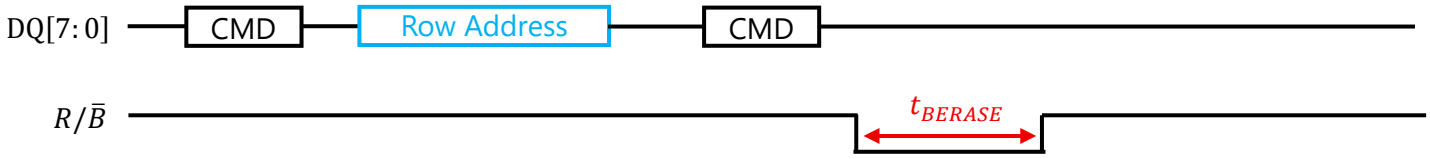


Figure 2.29 Block Erase Operation.

Set Feature Operation

Users may set particular features using 'Set Feature' operation. Once the Set Feature operation begins, the operation shall be completed without any disturbance or interruption.

Get Feature Operation

Users find how the target is set through the 'Get Feature' operation. The function shall return the current setting information.

Read ID Operation

The ID of a target is read using 'Read ID' operation. Once the Read ID operation begins, the operation shall be completed without any disturbance or interruption.

Read Status Operation

Users retrieve status value of the status register through the 'Read Status' operation.

General Timing

Table 2.*3* describes the bus state for the Toggle DDR. Command and address are all written through DQs by bringing $\overline{WE}$ to low while $\overline{CE}$ is low. Those are latched on the rising edge of $\overline{WE}$. Command Latch Enable (CLE) and Address Latch Enable (ALE) are used to multiplex the command and address respectively, via the DQ pins. The host reads or writes data to the device using the DQS signal. Data is latched on both falling and rising edges of DQS on data input.

Table 2.3 Common Mode Selection of Toggle DDR.

$\overline{CE}$	CLE	ALE	$\overline{WP}$	$\overline{RE}$	DQS	$\overline{WE}$	Mode	
Low	High	Low	Low	High	X	Rising Edge	Read Mode	Command Input
Low	Low	High	Low	High	X	Rising Edge		Address Input
Low	High	Low	High	High	X	Rising Edge	Write Mode	Command Input
Low	Low	High	High	High	X	Rising Edge		Address Input
Low	Low	Low	High	High	Falling Edge & Rising Edge	High	Data Input	
Low	Low	Low	Low	Falling Edge & Rising Edge	Falling Edge & Rising Edge	High	Data Output	
X	X	X	X	High	X	X	During Read (Busy)	
X	X	X	High	X	X	X	During Program (Busy)	
X	X	X	High	X	X	X	During Erase (Busy)	
X	X	X	Low	X	X	X	Write Protect	
High	X	X	X	X	X	X	Stand-by	
Low	Low	Low	High	High	X	High	Bus Idle	

Note:

1) X means "don't care", which can be High or Low.

2) Data is valid after both the falling edge and rising edge of RE_n

Command Latch Cycle

Command information is latched by $\overline{WE}$ going 'High', when $\overline{CE}$ is 'Low', CLE is 'High', and ALE is 'Low'. $\overline{WP}$ is low in read mode, and high in write mode.

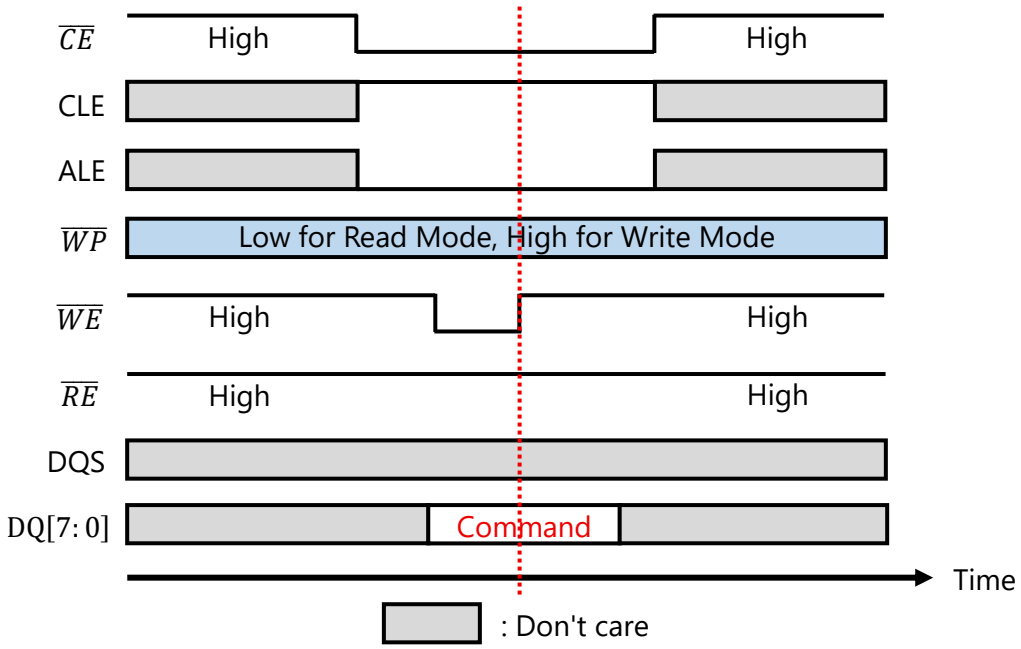


Figure 2.30 Command Latch Cycle Timing.

Address Latch Cycle

Address information is latched by $\overline{WE}$ going 'High', when $\overline{CE}$ is 'Low', CLE is 'Low', and ALE is 'High'. $\overline{WP}$ is low in read mode, and high in write mode.

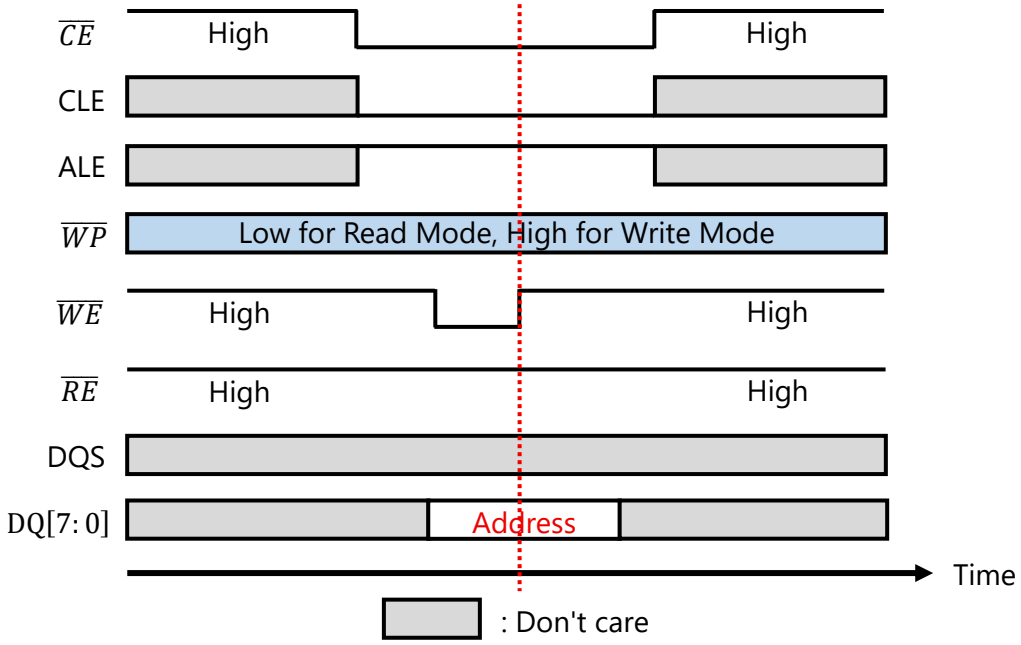


Figure 2.31 Address Latch Cycle Timing.

Basic Data Input Timing

DQS and data input buffers are turned on when $\overline{CE}$ and DQS go 'Low' and data inputs begin with DQS toggling simultaneously (centered with write data).

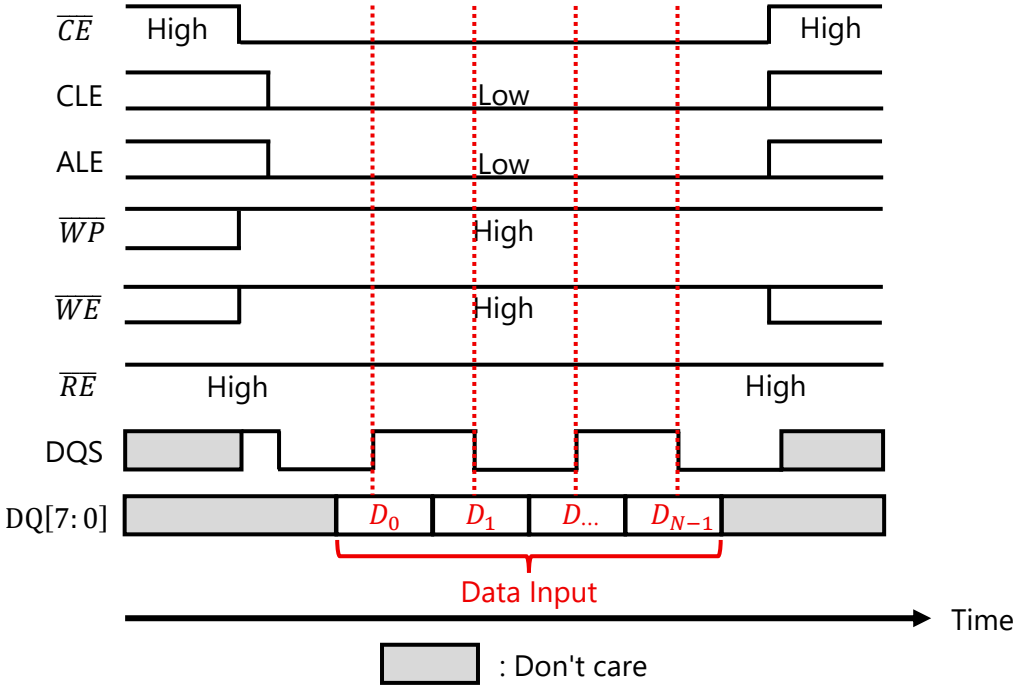


Figure 2.32 Basic Data Input Timing.

Basic Data Output Timing

DQS and DQ drivers are turned on when $\overline{CE}$ and $\overline{RE}$ go 'Low' for data-out operation, and data outputs begin with DQS toggling simultaneously (edge-aligned with write data). Note that the DQS and DQ drivers switch from valid to high-Z when either CLE or $\overline{CE}$ goes high.

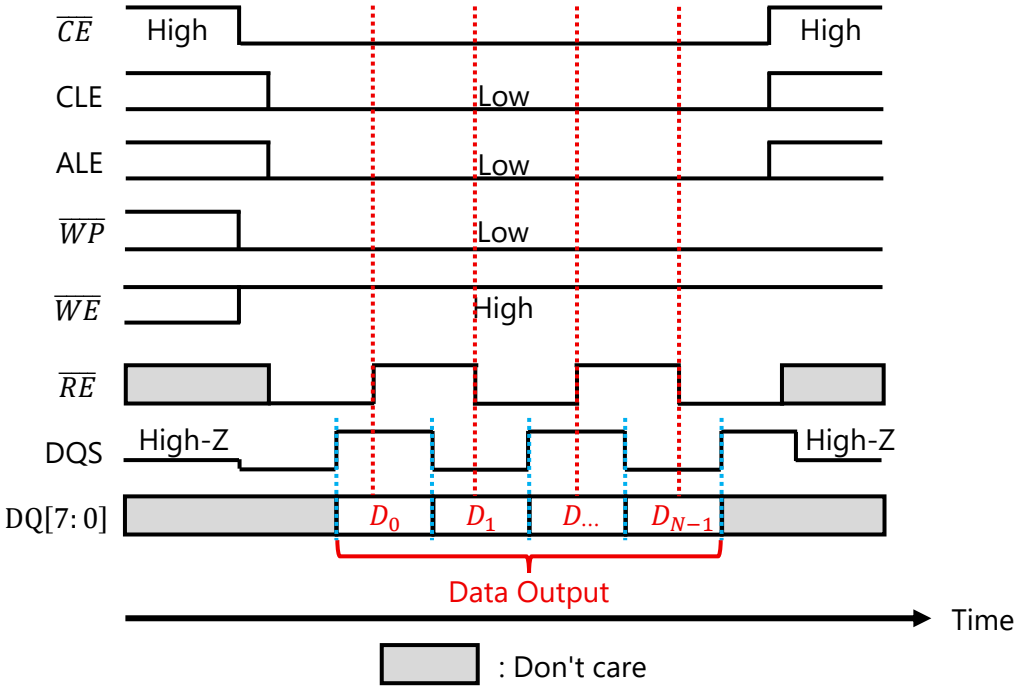


Figure 2.33 Basic Data Output Timing.

Page Write Operation Timing

DQS shall be set to High before data-input.

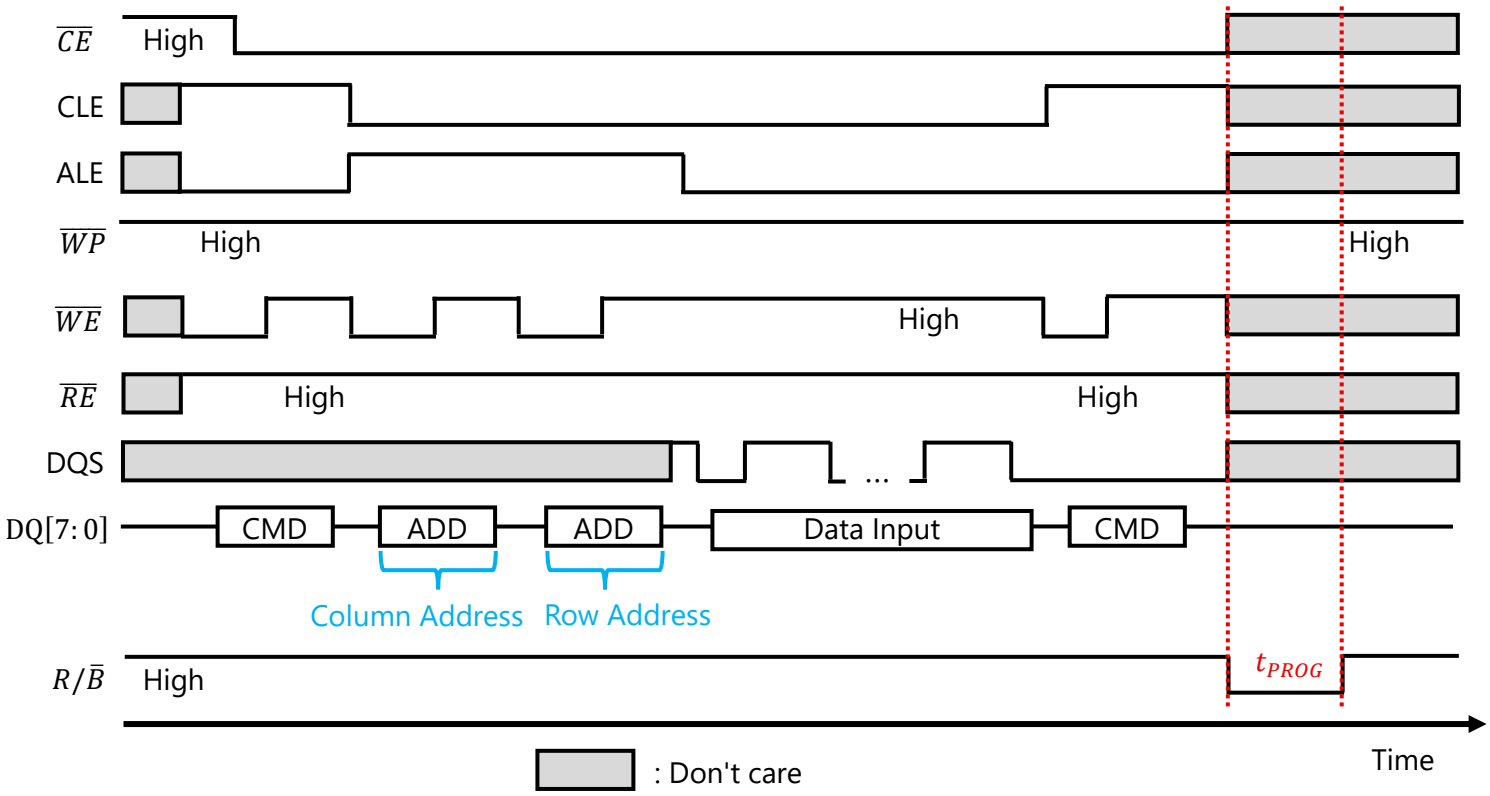


Figure 2.34 Page Write Operation Timing.

Page Read Operation Timing

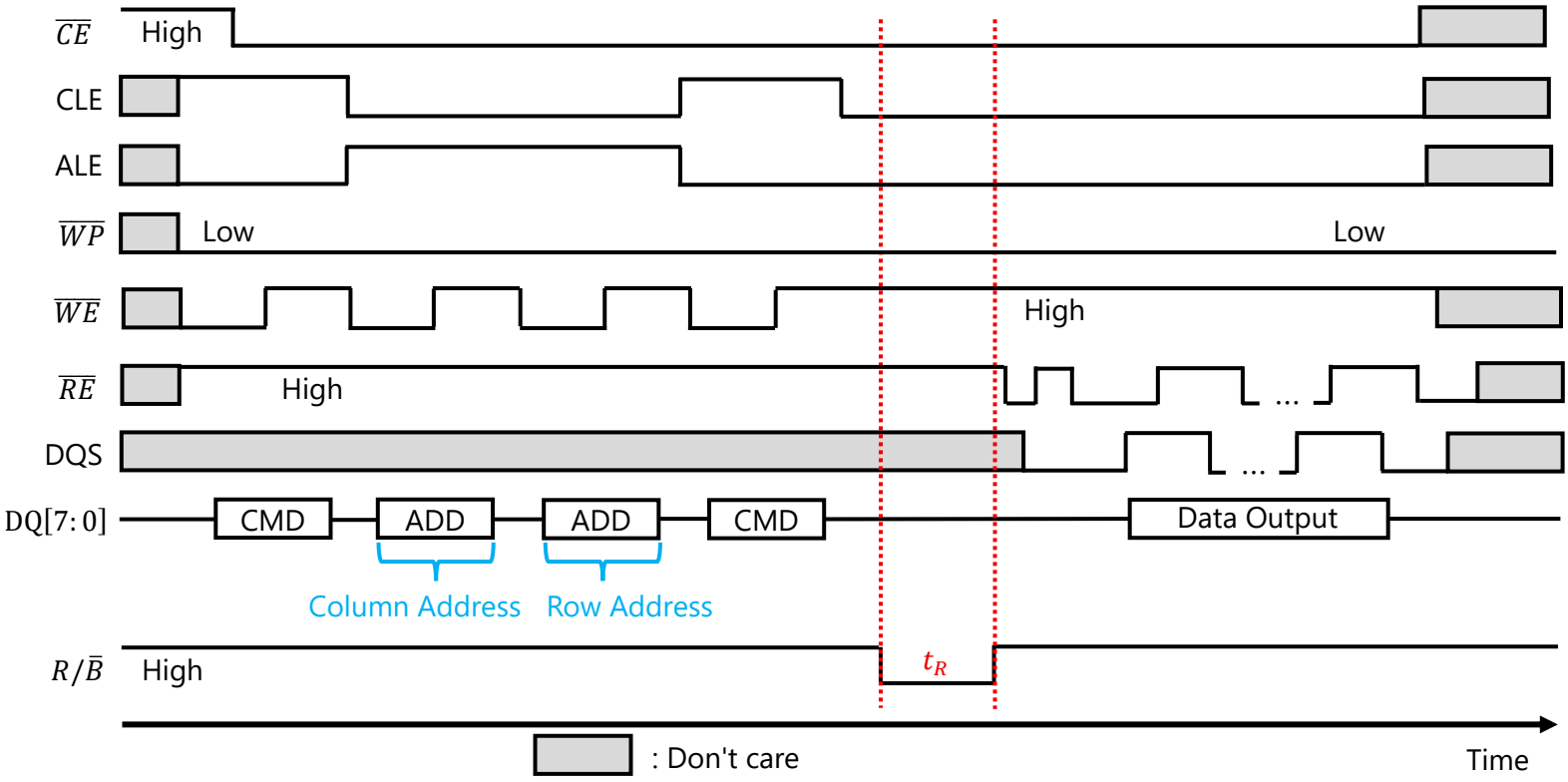


Figure 2.35 Page Read Operation Timing.

Timing Parameters Description

Table 2.4 Toggle DDR Timing Parameters Description.

Parameter	Description
t_R	Data transfer from flash array to register
t_{PROG}	Program time
t_{BERASE}	Erase time

Basic Command Sets

Toggle DDR Flash Memory has addresses multiplexed into I/Os. Command and address are all written through DQ[7:0] by bringing $\overline{WE}$ to low while $\overline{CE}$ is low. Those are latched on the rising edge of $\overline{WE}$. Command Latch Enable (CLE) and Address Latch Enable (ALE) are used to multiplex the command and address, respectively, via the DQ[7:0] pins.

Table 2.5 Basic Command Sets.

Function	Description
Page Read	Read a page of data.
Full Sequence Program	Program a full sequence program unit within a plane.

	A full sequence program (FSP) unit of the TLC device consists of three pages: LSB, CSB, and MSB. LSB, CSB, and MSB of each FSP unit shall be configured by the same row address.
Block Erase	Erase a block of data.
Set Feature	Set particular features.
Get Feature	Return the current setting information.
Read ID	Read the ID of a target.
Read Status	Retrieve status value of the status register.

Extended Command Sets

Table 2.6 defines the Extended Command Sets.

Table 2.6 Extended Command Sets.

Function	Description
Multi Plane Read	Read a page of data.
Multi Plane Full Sequence Program	Program a full sequence program unit for multiple planes.
Multi Block Erase	Erase multiple blocks, one from each plane.

Multi-Plane and Multi-Block operations have specific restrictions on the address input. For read and program operation, multiple WL addresses may be set over multiple planes. When setting the WL address for each plane, the WL addresses shall be identical, even if the block addresses differ. For the erase operation, multiple block addresses may be set over multiple planes.

Multi Plane Read Operation

The Multi-Plane Read operation extends the Page Read operation. The device supporting Multi Plane Read operation also allows multiple random data outputs from each page (i.e., Multi Plane Random Data Output) once multiple planes are loaded to the page registers.

Multi Plane Full Sequence Program Operation

The Multi Plane Full Sequence Program function extends an effective programmable size using multiple planes.

Multi Block Erase Operation

Multi-Block Erase allows users to simultaneously erase multiple blocks, one from each plane.

Interleaving Operation

When multiple LUNs of a target share a common CE_n, the device provides interleaving across those LUNs. At first, the host issues an operation command to one of the LUNs in the target, say LUN #M. The target goes into the busy state. During this time, the other LUN(s), say LUN #N, on

the same target are in the ready state. So it can execute the operation command issued by the host.

Parallelism

A NAND flash device contains one or more chips, and a chip is organized into one or more logical units (LUNs), as shown in Figure 2.*36*. There are three primary mechanisms for achieving parallelism within this architecture. One is that there can be multiple commands outstanding to different LUNs simultaneously. To get further parallelism within a LUN, plane addressing can be used to execute additional dependent operations in parallel. Additionally, parallel page operations within a plane can be used if the device supports their functionality.

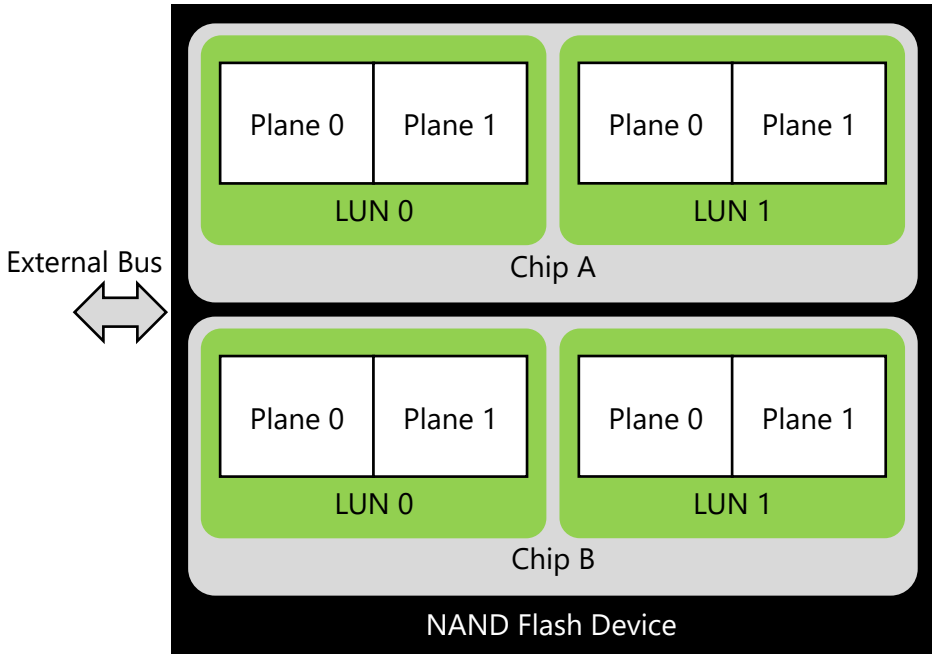


Figure 2.36 Parallelism in NAND Flash Device.

Addressing

There are two types of addresses used: the *column address* and the *row address*. The column address is used to access bytes within a page; specifically, it is the byte offset into the page. The least significant bit of the column address shall always be zero for a DDR interface, i.e., an even number of bytes is always transferred. The row address is used to address WLs, blocks, and LUNs.

The row address structure is shown in Figure 2.*37* with the least significant row address bit to the right and the most significant row address bit to the left. The WL address is set by the least significant row address bits, and the LUN address is set by the most significant row address bit(s). The block address is between a WL address and a LUN address.

Figure 2.37 Row Address Layout.

The plane address comprises the lowest order bits of the block address, as shown in Figure 2.*38*. The plane address is used when performing a multi-plane operation on a particular LUN.

Figure 2.38 Position of Plane Address.

A typical addressing of an 8-bit wide NAND flash device is shown in Table 2.*7*.

Table 2.7 Common Protocol Signal Description of NAND Flash Interfaces.

	IO7	IO6	IO5	IO4	IO3	IO2	IO1	IO0
First Cycle (Column Address 1)	C1-7	C1-6	C1-5	C1-4	C1-3	C1-2	C1-1	C1-0
Second Cycle (Column Address 2)	C2-7	C2-6	C2-5	C2-4	C2-3	C2-2	C2-1	C2-0
Third Cycle (Row Address 1)	R1-7	R1-6	R1-5	R1-4	R1-3	R1-2	R1-1	R1-0
Fourth Cycle (Row Address 2)	R2-7	R2-6	R2-5	R2-4	R2-3	R2-2	R2-1	R2-0
Fifth Cycle (Row Address 3)	R3-7	R3-6	R3-5	R3-4	R3-3	R3-2	R3-1	R3-0

Note:
1) R1-0 to R1-7 is the WL address.
2) R2-0 to R3-3 is the block address.
3) R3-4 to R3-7 is the LUN address.

Factory Defect Mapping

The NAND flash array is not presumed to be pristine, and a number of defective cells that make the blocks unusable may be present. Therefore, invalid blocks (defective) shall be sorted out from normal blocks by software.

d) Device Requirements

The manufacturer erases all pages in non-defective blocks, as the majority of bits are read as a specific value (assumed to be 0xFF). For a defective block (initial defect), the manufacturer shall mark the block as defective by setting the defective block marking, as shown in Figure 2.*39*, to a specific value (assume to be 0x00). The defective block marking is located on the first byte of the main data area or the first byte of the spare data area in the first LSB page or the last MSB page within a block.

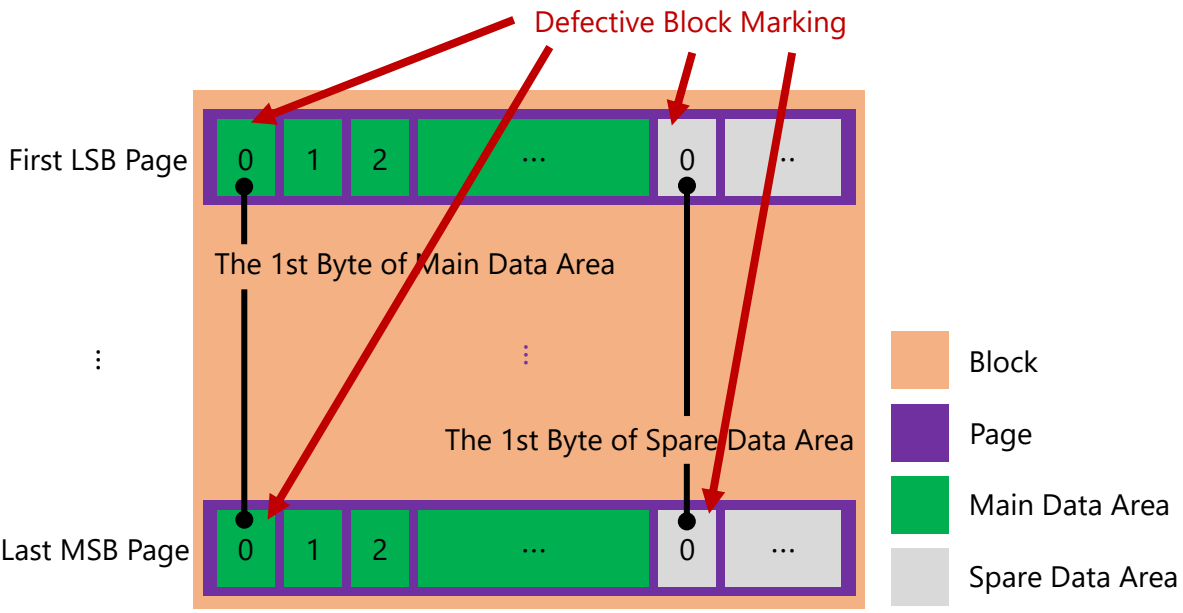


Figure 2.39 Area Marked in the First LSB Page or the Last MSB Page of Block Indicating Defect.

e) Host Requirements

The host (i.e., users of NAND flash memory) shall not erase or program blocks marked as defective by the manufacturer, and any attempt to do so yields indeterminate results.

To avoid accessing initial defective blocks, the host creates the initial bad block table prior to performing any erase or program operations on the target block. A defective block is indicated by the majority of bits being read as 0x00 in the defective block marking. The host shall verify the defective block marking to confirm the block is valid before performing any erase or program operations on that block. When the host encounters data neither 0xFFh nor 0x00, the host should select either 0xFF or 0x00 that has a closer Hamming distance to the data. Figure 2.*40* outlines the flow chart to create an initial invalid block table. Note that over the lifetime use of a flash memory, the defective block marking of defective blocks may encounter read disturbs that cause bit changes. The initial defect marks by the manufacturer may change in value over the device's lifetime and are expected to be read by the host.

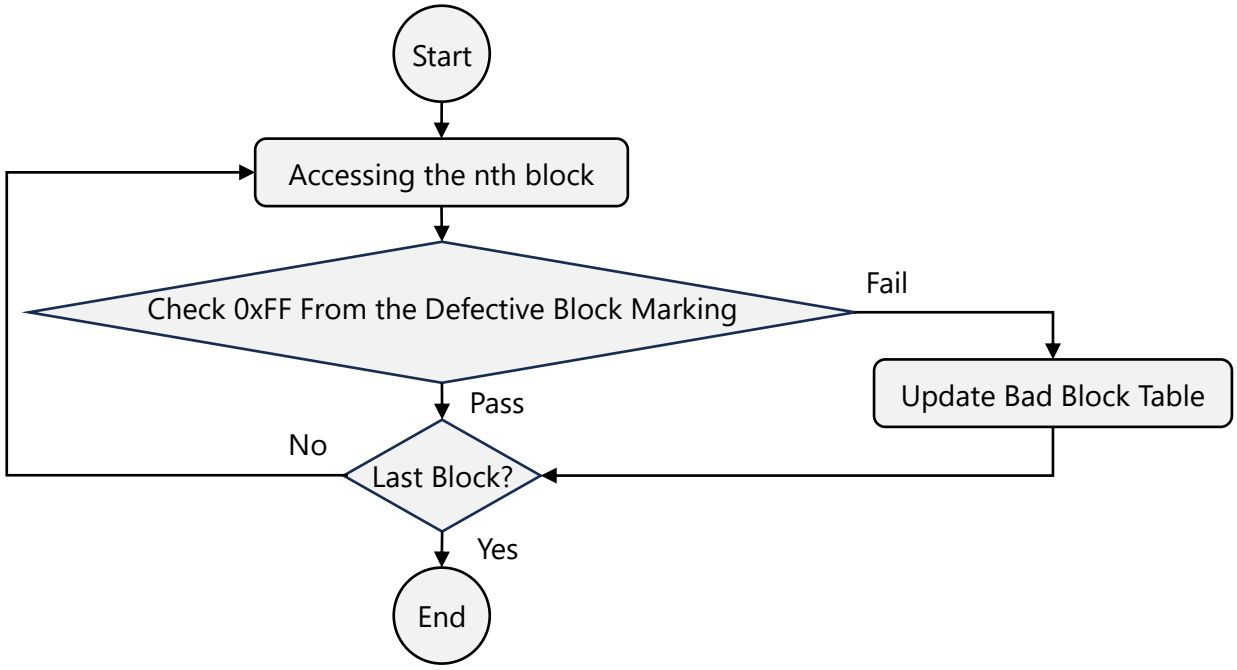


Figure 2.40 Flow Chart to Create Initial Invalid Block Table.

(todo)

Characteristics of Typical NAND Flash Memory

Table *2.8* summarizes the parameters of five different NAND flash memories from four manufacturers. Generally, reading an entire page of data in parallel from the plane into the internal buffer takes 10-25 us. It indicates a non-uniform access latency for reading. It also shows that a sequential read in flash memory is three orders of magnitude faster than a random read. Writing a page involves moving the data from the package pins to the chip. Once the data is in an internal buffer, writing typically takes between 200 us (SLC) and 800 us (MLC). Erasure typically takes 1.5 - 2 ms to erase the entire block.

Table 2.8 Characteristics of Typical NAND Flash Memories.

Manufacturer	Samsung	Intel	AMD	FUJITSU
Type	K9NBG08U5A	JS29F16G08FANB1	Am30LV0064D	MBM30LV0128

Capacity	4G x 8 bit	2G x 8 bit	8M x 8 bit	16M x 8 bit
Page size (byte)	(2K + 64)	(2K + 64)	(512 + 16)	(512 + 16)
Block size (byte)	(128K + 4K)	(128K + 4K)	(8K + 256)	(16K + 512)
Random read	25 us (max)	25 us (max)	N/A	10 us (max)
Serial read	50 ns (min)	25 ns (min)	<50 ns	35 ns (min)
Program time	200 us (typ.)	220 us (typ.)	200 us	200 us (typ.)
Erase time	1.5 ms (typ.)	1.5 ms (typ.)	2 ms	2 ms
Endurance	100K	100K	10K	1 Million
Voltage	2.7 - 3.6 V	2.7 - 3.6 V	2.7 - 3.6 V	2.7 - 3.6 V
Power (active)	75 mW	75 mW	30 mW	72 mW
Power (standby)	60 mW	3 mW	0.03 mW	3.6 mW

(todo)

NAND flash memory errors

NAND flash memory errors can be induced by a variety of sources, including flash cell wearout, disturb effects (errors introduced during programming, interference from operations performed on adjacent cells), and data retention issues due to charge leakage. (three major sources)

Disturb effects alter the memory transistors threshold voltage unintentionally during memory access operations under the influence of specific disturb conditions. Since it is essential for an efficient area consumption to arrange the storage transistors in a contact saving way, the sharing of voltage nodes can not be avoided. During the read and program operations, positive voltages are applied to the gate nodes of the memory transistors, wherein the channel is at a lower potential or even grounded. Therefore this condition is called gate disturb (also word line disturb) and it is the most common disturb mechanism in NAND Flash memory arrays.

(Program disturb and read disturb)

Read disturbs are the most frequent source of disturbs in NAND architectures. This kind of disturb may occur when reading many times the same cell without any erase operation. All the cells belonging to the same string of the cell to be read must be driven in a ON state, independently of their stored charge. The relatively high V_{pass} bias applied on the control gate and the sequence of V_{pass} pulses applied during successive read operation may trigger the Stress Induced Leakage Current (SILC) effects in some cells that, therefore, may gain charge. Note read disturbs do not provoke permanent oxide damages: if erased and then reprogrammed, the correct charge content will be present within the floating gate.

Pass disturb is similar to the read disturbs and affects cells belonging to the same string of a cell to be programmed.

The Program disturbs, on the contrary, affect cells that are not to be programmed (inhibit) and belong to the same word line of those that are to be programmed. In that case the program disturb is strongly related to the voltages and pulse sequences used for the self-boosting techniques. Although the program inhibit boosts the channel potential, soft programming can not be avoided especially when a high number of program pulses are applied.

The criticality of an effective program operation limiting program disturbs and/or possible successive errors is attested by the fact that in NAND memories the program operation should follow a precise and well defined "hierarchy": it is necessary to start from the cell nearest to the source selector and proceed along the string up to the cell nearest to the drain selector. This procedure is important, because the threshold voltage of a cell depends on the state of the cells placed between the considered cell and the source contact (the background pattern dependency phenomenon); the series resistance of the cells is different if they are programmed or erased.

(further) When manufacturing process scales down to a smaller technology node (i.e., the size of each flash memory cell), the amount of charge that can be trapped within the floating gate also decreases, which exacerbates reliability issues.

NAND Flash Memory Organization:
The flash memory is spread across multiple flash chips (typical values: 4, 16 chips), where each chip contains one or more flash *dies*, which are individual pieces of silicon wafer that are connected together to the pins of the chip. Each chip is connected to one or more physical memory channels, and these memory channels are not shared across chips. A flash die operates independently of other flash dies, and contains between one and four *planes*. Each plane contains hundreds to thousands of flash *blocks*. Each block is a 2D array that contains hundreds of rows of flash cells (typically 256-1024 rows) where the rows store contiguous pieces of data.

In NAND Flash memories, a logical page is the smallest addressable unit for reading and writing; a logical block is the smallest erasable unit.

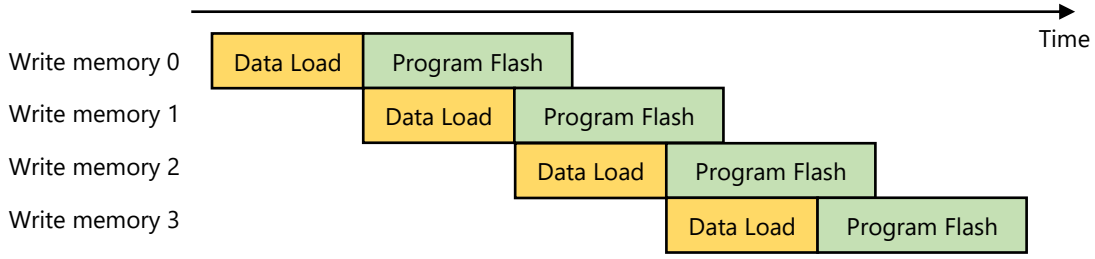
If we consider the SLC case with interleaved architecture, even cells belong to the even page (BLe), while odd pages belong to the odd page (BLo). For example, a SLC device with 4 KiB page has a WL of 32,768 + 32,768 = 65,536 cells. In the MLC case we have MSB and LSB pages on even BL, and MSB and LSB pages on odd BL.

Each page is made up by main area (data) and spare area as shown in Fig. 1.5. Main area can be 4, 8 or 16 KiB. Spare area can be used for ECC (Error Correction Code) and is in the order of hundred of Bytes every 4 KiB of main area.

Each page has a fixed-size main data area and a spare data area. The data area is for the storage of data, and the spare area stores the corresponding LBA, ECC, and other information.

The **planes** can execute flash operations in parallel, but the planes within a die share a single set of data and control buses. Hence, an operation can be started in a different plane in the same die in a pipelined manner, every cycle.

Channel is the data bus (typical width: 8-bit) for connect different memories to the SSD controller (or NAND controller inside). Operations on a channel can be interleaved, which means that a second chip can be addressed while the first one is still busy. For instance, a sequence of multiple write operations can be directed to a channel, addressing different NANDs, as shown in Fig. 1.13: in this way, the channel utilization is maximized by pipelining the data load phase; in fact, while the program operation takes place within a memory chip, the corresponding Flash channel is free.



Data in a block is written at the unit of a *page*, which is typically between 8 and 16 KiB in size in NAND flash memory. All read and write operations are performed at the granularity of a page. Each block typically contains hundreds of pages.

Flash cards, USB keys and Solid State Drives are definitely the most known examples of electronic systems based on non-volatile memories.

Chapter 3 Solid State Drive Design

A solid-state drive is a complete, small system where every component is soldered on a PCB and is independently packaged. The basic structure of a solid-state drive is shown in Figure 3.1. Essentially, it is composed of a controller and NAND flash memory. In addition, there are usually other components. For instance, an external DC-DC converter can be added to derive the internal power supply, or a quartz crystal can be used for improved clock precision. Of course, reasonable filter capacitors are inserted to stabilize the power supply. It is also very common to have a temperature sensor for power management reasons. For data caching, a fast DDR (Double Data Rate) memory is often added to the board. During a write access, the cache is used to store data before transferring it to the flash memory. The benefit is that data updating, e.g., in mapping tables, is faster and does not wear out the flash memory.

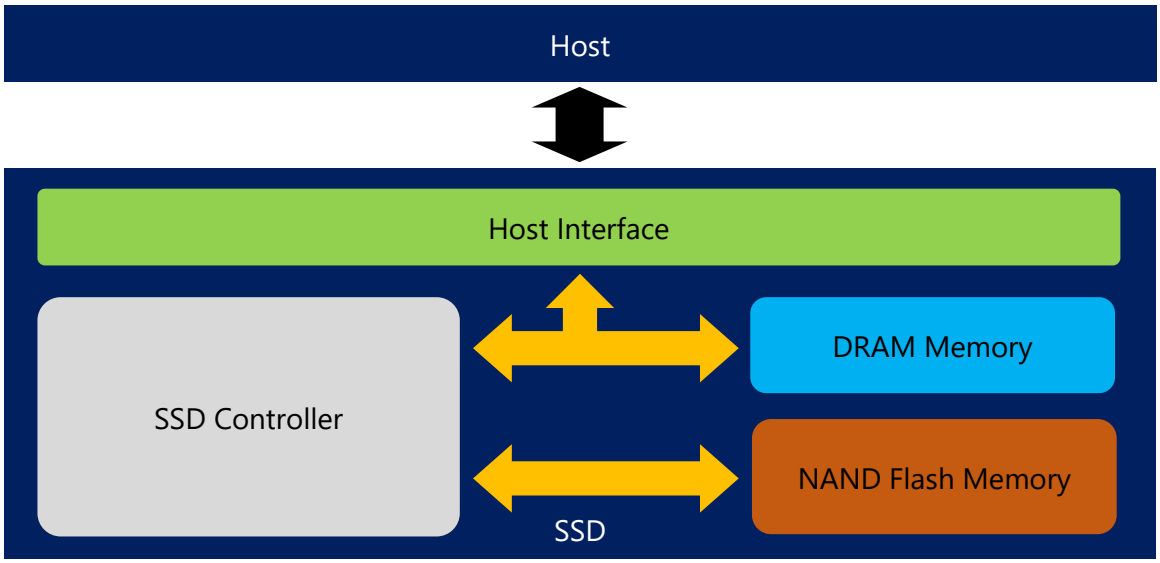


Figure 3.1 Block Diagram of a SSD system

For many applications, the host interface to SSDs remains a bottleneck to performance. PCI Express (PCIe)-based SSDs together with flash-optimized host control interface standards address this interface bottleneck. SSDs with legacy storage interfaces are proving useful, and PCIe SSDs will further increase performance and improve responsiveness by connecting directly to the host processor.

NANDs are usually available both in TSOP (Thin small outline package) and BGA (Ball grid array) packages. In order to improve performances, NANDs are organized in different Flash Channels

Supplement materials

Flash memory is usually accessed by embedded systems as a raw medium or indirectly through a block-oriented device. In other words, the management of flash memory is carried out by either software on a host system (as a raw medium) or hardware/firmware inside its device. Therefore, there are two kinds of approaches that utilize flash memory. One is designing a new flash file system, such as log-structured file systems, the journaling flash file system (JFFS), yet another flash

file system (YAFFS), the Trimble file system, the Microsoft flash file system, the Norris flash file system, and some other commercial embedded file systems. The other approach uses traditional file systems (e.g., FAT and Ext2) and wraps the flash memory to mimic a block-level storage device. Consequently, the algorithms and functionalities used to handle the characteristics of flash memory (e.g. erasing before writing and the limited life span) can be integrated into either a file system or the firmware inside a flash memory device.

However, due to the compatibility with traditional file systems, the second solution is more popular and widely used in products.

3.1. Host Interface

Two main interfaces are used by SSDs to exchange data with a host device (such as a computer): SATA and PCIe. (For SATA). PCIe (PCI Express) is the same interface used for graphic cards. It offers much faster speeds than SATA.

PCI express (PCIe)-based SSDs use NVMe (Non-Volatile Memory Express) which is a system for transferring data over PCIe. It not only speeds up transfers but also reduces latency for SSDs, meaning transfers start more quickly.

The SATA protocol interfacing 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. As a result, 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 joined to the availability of high density NAND flash memories that can store 2, 3 or even 4 bits in a single cell, 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.

The most common form factors for an SSD are a 2.5-inch model, which usually uses SATA, and a longer, narrower add-in card, which usually uses PCIe.

A third design, M.2, is smaller than the other two but is compatible with both SATA and PCIe. That makes it particularly useful for portable computers and devices.

3.2. ECC scheme

The bottom-up design first designs the ECC scheme, which defines the fundamental performance and reliability.

Figure 7.2a 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. The structure of the decoder is represented in Fig. 7.2b, 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

Multi-level NAND flash memories require the availability of an ECC (Error Correction Code) scheme that can correct errors detected when reading the memory. The choice of the ECC code and the design of the correction engine represent key points in the design of present SSDs, as they must be carefully calibrated with respect to the figures of merit of the selected nonvolatile memories. 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 also on the system power consumption. Based on the selected ECC code and of the designed **ECC engine**, an optimal error reduction algorithm for the memory read operation could be identified.

3.3. SSD Controller

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 controller, to avoid that the design efforts devoted to optimize the ECC scheme vanish, must guarantee the bandwidth provided by the ECC block. 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 so that SAS and PCI-Express are adopted for enterprise environments.

3.4. Data Path

(todo)

The SSD controller is responsible for scheduling the distributed accesses at the memory channels. And it uses dedicated engines (i.e., NAND controller) for the low-level communication protocol with the Flash.

SSD Controller: The SSD controller is responsible for (1) handling I/O requests received from the host, (2) ensuring data integrity and efficient storage, and (3) managing the underlying NAND flash memory.

Charge pumps are used to generate all the needed voltages within the chip

In multilevel storage, cell's gate biasing voltages need to be very accurate and voltage regulators become a must.

The Row Decoder is the block in charge of addressing and biasing each single word line and it is located between the planes. Bit lines are connected to a sensing circuit. The purpose of sense amplifiers is to read the analog information stored in the memory cell.
The Row Decoder, also called Word line Decoder or Word line Driver.

Especially, SSDs call for a higher read and write throughputs; in other words, SSDs need to manage more NAND dies in parallel. Basically, there are a couple of options:
- The first one is to increase the number of dies per channel;
- The second option is to increase the number of channels.

Flash chip controllers (FCCs): A Flash chip controller is assigned to a flash memory channel for data and control connection.

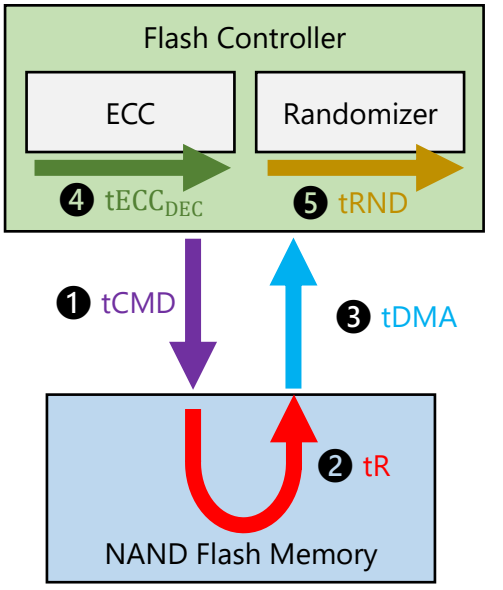
DRAM: The on-board DRAM memory stores various controller metadata (e.g., how host memory addresses map to physical SSD addresses) and to cache relevant (e.g., frequently accessed) SSD pages.

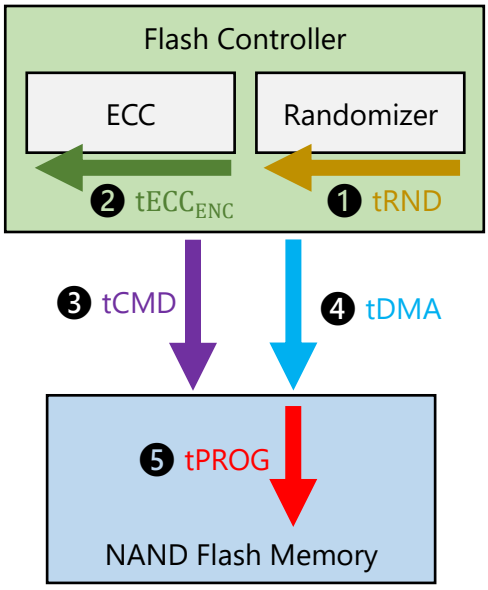
SSD Performance:
In the evaluation of SSD performance, there are three metrics and they are: (1) latency (or response time), (2) bandwidth, (3) throughput.

First two metrics are similar as mentioned before, latency is the time delay until the request is returned and bandwidth is the amount of data that can be accessed per unit time. Typical values are,
Latency:
Average read latency (4 KiB): 67 us
Average write latency (4 KiB): 47 us

Bandwidth:
Sequential read bandwidth: up to 3,500 MB/s
Sequential write bandwidth: up to 3,000 MB/s.

Throughput is the number of requests that can be serviced per unit time. SSDs define the measurement of the throughput as the IOPS: Input/output Operations Per Second. Typical values are,
Random read throughput: up to 500K IOPS
Random write throughput: up to 480K IOPS





3.5. Reliability of SSD

The drive's reliability is specified with two metrics: the Mean Time Between Failures (MTBF) and the Uncorrectable Bit Error Rate (UBER).

(todo)

Chapter 4 Firmware

To translate I/O requests from the host system into NAND memory operations, the SSDs utilize the firmware, also called flash translation layer (FTL). FTL is a special software that maps the logical page address from the file system to the physical page address in flash memory devices. It is

responsible for a plug-and-play connection between the host system where the application is running and the SSD.

Wear Leveling (endurance):
(insight) Usually, not all the information stored within the same memory location change with the same frequency: some data are often updated while others remain always the same for a very long time in the extreme case, for the whole life of the device.

To extend the durability by using wear-leveling technique.

(goal) In order to mitigate disturbs, it is important to keep the aging of each page/block as minimum and as uniform as possible: that is, the number of both read and program cycles applied to each page must be monitored.

(endurance definition) the maximum number of allowed program/erase cycles for a block (i.e. its endurance)

The controller firmware groups blocks with the same ID number across multiple chips and planes together into a *superblock*. Within each superblock, the pages with the same page number are considered a *superpage*. The controller opens one superblock (i.e., an empty superblock is selected for write operations) at a time, and typically writes data to the NAND flash memory one superpage at a time to improve sequential read/write performance and make error correction efficient, since some parity information is kept at superpage granularity. Having the ability to write to all of the pages in a superpage simultaneously, the SSD can fully exploit the internal parallelism offered by multiple planes/chips, which in turn maximizes write throughput.

4.1. Error Mitigation Techniques
SSDs implement a set of error mitigation techniques at several levels (from NAND flash physics and integrated circuit architecture to SSD firmware) to achieve a high reliability.

Proper reliability management solutions like the read retry and the soft decoding error correction codes (ECCs) are introduced.

Issues at die level like yield defects or extrinsic failures are exposed. Their mitigation is addressed by techniques like the RAID (Redundant Array of Independent Disks) to improve the overall SSD reliability.

Chapter 5 Debuggability

(Telemetry)
Telemetry is a technology that allows the remote measurement and reporting of information of interest to the system designer or operator.

Telemetry typically refers to wireless data transfer mechanisms (e.g., using radio, ultrasonic, or infrared systems). However, it can also refer to data transfer over other media, such as a telephone or computer network, an optical link, or other wired communications, like power line carriers.

(Tracing)
Tracing in software engineering refers to the process of capturing and recording information about the execution of a software program. Programmers typically use this information for debugging purposes. Additionally, depending on the type and detail of information contained in a trace log, experienced system administrators or technical-support personnel, as well as software monitoring tools, can use it to diagnose common software problems.