CHAPTER 4

Basics of Flash Memory

In this chapter, we will discuss different memory types and delve into the world of flash memory, exploring its different types and focusing on two primary types: NAND and NOR flash memory. We will discuss the architecture of NAND flash memory and its fundamental operations, including reading, writing, and erasing data. Understanding these basic operations is crucial to grasp how NAND flash memory functions and how it is utilized in solid-state drive (SSD) firmware. By the end of this chapter, you will have gained valuable insights into the basics of flash memory, enabling you to comprehend its architecture and the fundamental operations it supports.

Memory Types

Flash memory is a type of non-volatile memory that is used in a variety of electronic devices, including SSDs. Non-volatile memory can retain data even when the power is turned off, making it ideal for storing important information.

There are several different types of flash memory available, including NOR and NAND, as you can see in Figure 4-1.

G. Kuppan Thirumalai, *A Beginner's Guide to SSD Firmware*,
https://doi.org/10.1007/978-1-4842-9888-6_4

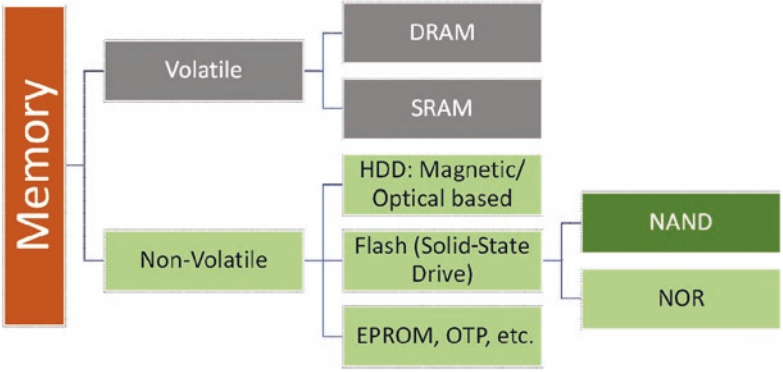


Figure 4-1. *Memory types*

NOR Flash Memory

NOR flash memory is capable of random access, meaning that data can be read or written to any location on the memory chip. It is commonly used in devices that require fast access to small amounts of data. It is possible to read/write one byte of data at a time. Erase operation is in sector wise. NOR flash memory is less dense, meaning it consumes more physical area and costs more than NAND flash memory.

Characteristics of NOR Flash Memory

The following are characteristics of NOR flash memory:

- Cost per bit is high.
- Code execution is easy.
- Capacity is low.
- Write speed is slower.
- Read speed is faster.
- Power consumption on standby is low.

NOR Memory Architecture

Take a look at the NOR memory architecture in Figure 4-2.

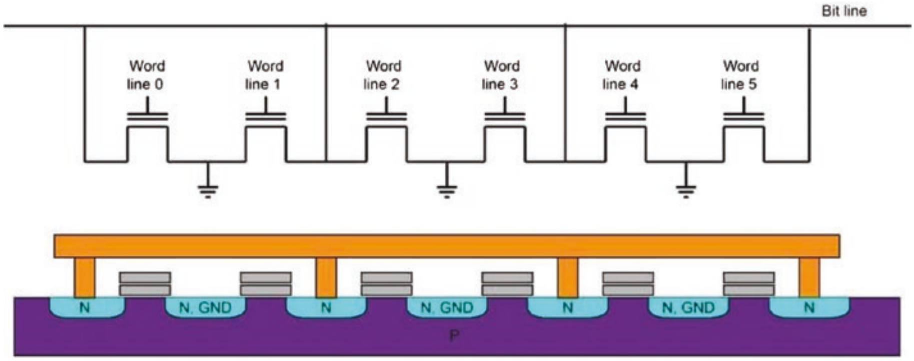


Figure 4-2. *NOR memory architecture*
Source: Wikipedia

NOR memory is a type of flash memory that uses NOR gates to store data. The gates are arranged in a grid, with each gate storing a single bit of data. The grid is divided into words, with each word containing a fixed number of bits.

To read data from NOR memory, the controller sends a `read` command to the memory. The memory then transfers the data from the selected word to the controller.

To write data to NOR memory, the controller sends a `write` command to the memory. The memory then writes the new data to the selected word.

NAND Flash Memory

NAND flash memory, however, is a type of flash memory that is optimized for high-capacity storage and fast data transfer. It is commonly used in SSDs and other storage devices, such as USB drives and memory cards. NAND memory is made up of tiny transistors that are arranged in a grid and can be used to store data in the form of bits (0s and 1s). It is fast and efficient, making it ideal for use in SSDs, and it is also relatively inexpensive and widely available.

NAND Memory Architecture

Take a look at the NAND memory architecture in Figure 4-3.

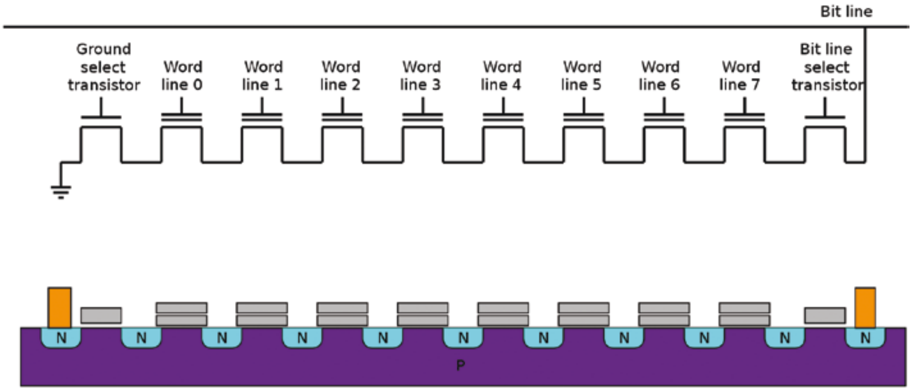


Figure 4-3. *NAND memory architecture*
Source: Wikipedia

NAND memory is a type of flash memory that uses floating-gate transistors to store data. The transistors are arranged in a grid, with each transistor storing a single bit of data. The grid is divided into pages, with each page containing a fixed number of bits.

To read data from NAND memory, the controller sends a `read` command to the memory. The memory then transfers the data from the selected page to the controller.

To write data to NAND memory, the controller sends a `write` command to the memory. The memory then erases the selected page and writes the new data to the page.

Similarities

NAND and NOR memory are both types of flash memory. They both use transistors to store data, and they both have a grid-like structure.

Differences

The main difference between NAND and NOR memory is the way that they store data. NAND memory uses floating-gate transistors, while NOR memory uses NOR gates. This difference in the way that they store data affects the performance and the features of the two types of memory.

NAND memory is generally faster than NOR memory, but it is also more expensive. NAND memory is also more durable than NOR memory.

NOR memory is slower than NAND memory, but it is also less expensive. NOR memory is also easier to program than NAND memory.

A Flash Memory Cell

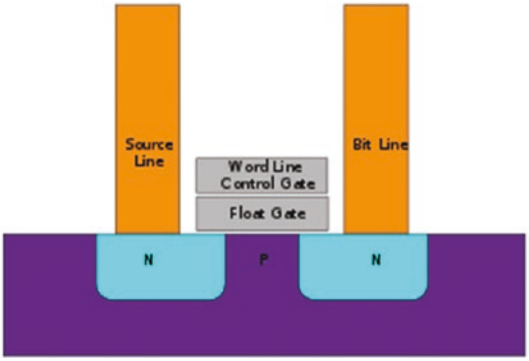


Figure 4-4. *A flash memory cell*

Flash memory, which is used in SSDs, combines the characteristics of ROM (read-only memory) and RAM (random access memory). It can retain information even when there is no power, like ROM, and it can be repeatedly erased and rewritten, like RAM. This is made possible through the use of a special type of transistor in flash memory.

Let’s break down how it works in a simplified manner, as follows:

1. Typical Transistors: In typical memory transistors, there are three connections: source, drain, and gate. The source is where electricity enters, the drain is where it exits, and the gate controls the flow. When the gate is closed, no current can flow, turning the transistor off and storing a zero (0). When the gate is open, power flows through, activating the transistor and storing a one (1).

2. Limitations of Typical Transistors: However, a typical transistor cannot remember its state when the power is switched off. When power is turned back on, it’s difficult to determine whether the transistor was on or off before the power was removed.

3. Flash Memory Transistors: Flash memory transistors have an additional connection called a floating gate. This floating gate is placed on top of the main gate. When the gate is open, electricity seeps through the first gate and remains trapped between the first and second gates, even when the power is off. Refer to Figure 4-5.

4. Retaining Information: The floating gate in flash memory allows it to remember its state even when the power is off. If you try to push current

through the transistor, the stored energy prevents it, representing a zero. Clearing the stored energy allows the current to flow, representing a one. This way, the flash transistor retains information regardless of whether the power is on or off.

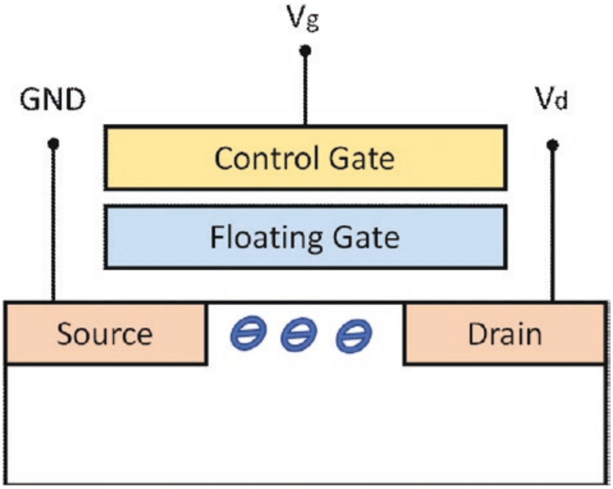


Figure 4-5. *Floating gate NMOS transistor*

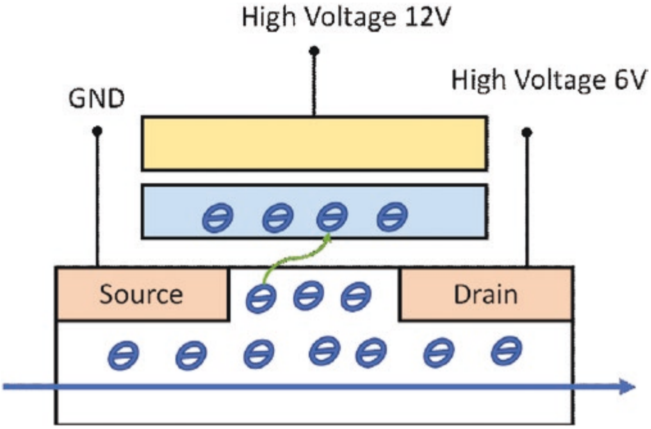


Figure 4-6. *Programming floating gate NMOS transistor*

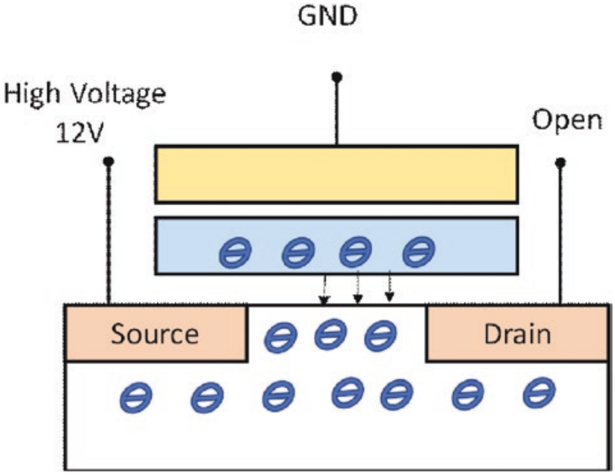


Figure 4-7. *Erasing floating gate NMOS transistor*

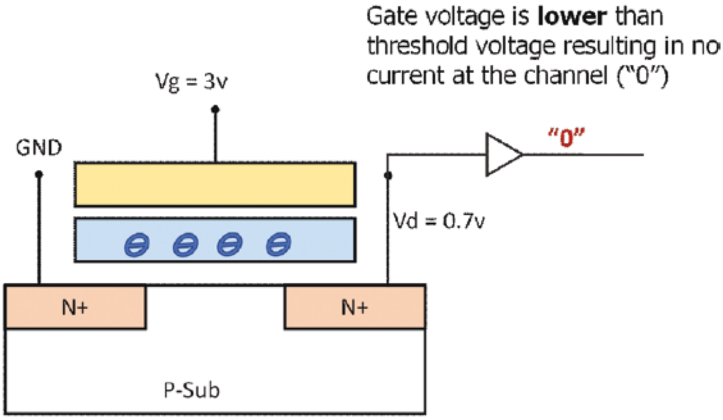

Gate voltage is **lower** than
threshold voltage resulting in no
current at the channel ("0")
Vg = 3v
GND
"0"
Vd = 0.7v
N+
N+
P-Sub

Figure 4-8. *Reading floating gate NMOS transistor*

Table 4-1. *Cell Node Voltages Required in Different Memory Operations*

Operation	Gate	Drain	Source	Bulk
Read	4.5	SA	0	0
Program	8.0	5.0	0	0
Erase	-8.0	Float	8.0	8.0

NAND Memory Organization

- The package is the memory chip, which contains one or more dies.
- The die is the smallest unit that can independently execute commands and report status.
- Each die contains one or more planes. Identical, concurrent operations can take place on each plane, although with some restrictions.
- Each plane contains a number of blocks, which are the smallest unit that can be erased. Remember that, as it's really important.
- Each block contains a number of pages, which are the smallest unit that can be programmed.

Addressing

It is NAND memory addressing. How physical nand can be addressed or accessed by Firmware.

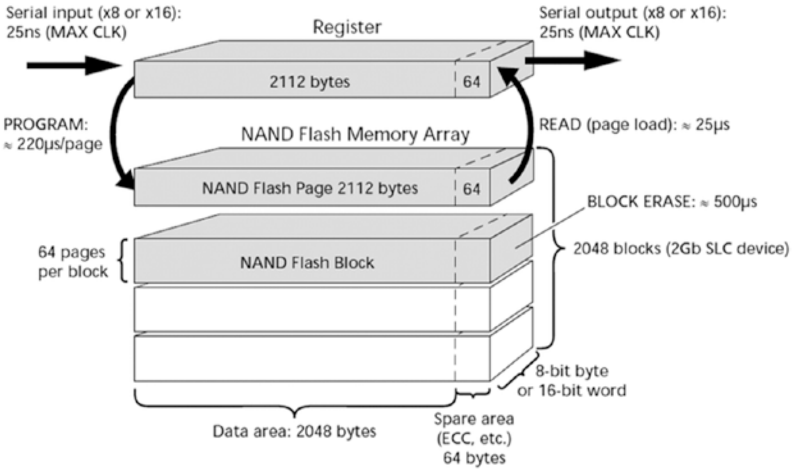


Figure 4-9. *The organizational structure of a NAND flash device*
Source: Micron Technology Inc.

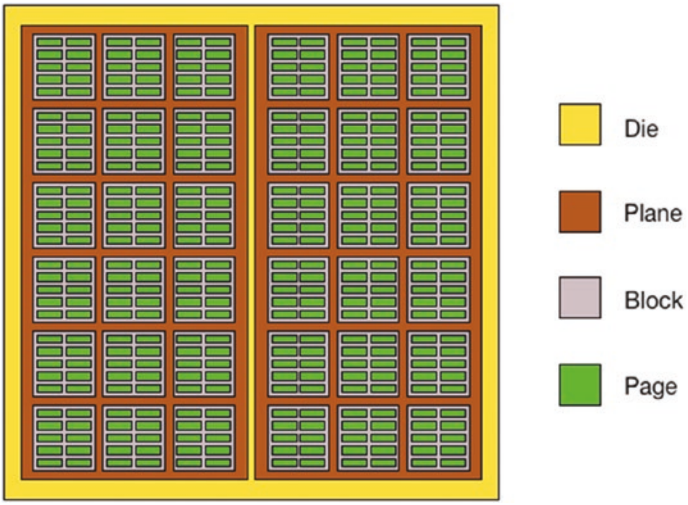


Figure 4-10. *NAND flash die layout*
Source: AnandTech

Erase

In a flash memory device, the `erase` operation is responsible for changing the state of a cell from "0" to "1" by removing electrons from the floating gate. It is important to note that a single cell cannot be directly changed from "1" to "0"; instead, the `erase` operation must be performed on a block-by-block basis. This means that before new data can be written to a block (through the programming process), the block must first be erased to ensure that it is empty. It is worth noting that the `erase` operation typically has a longer latency than the `read` and `program` operations, meaning it can take longer to complete. For example, the read, program, and erase latencies for a Micron 8 GB flash chip are 25 μs, 220 μs, and 1500 μs, respectively. As a result, the `erase` operation can be a performance bottleneck in NAND flash memories, and various firmware algorithms have been developed to minimize the impact of the long erase latency on overall performance.

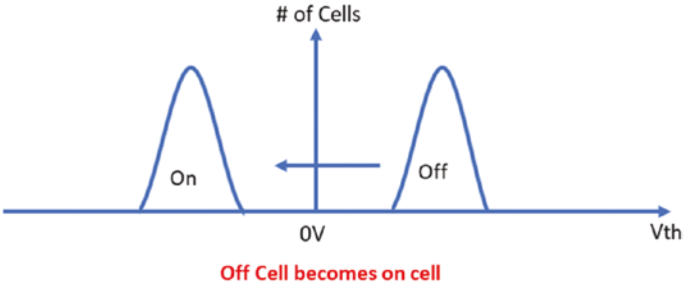


Figure 4-11. *Erase level Vth distribution*

Write

The `program` operation is performed on a page level. This means that the operation targets a specific page of memory on the drive. When the controller of the SSD requests a program operation on the NAND device, it specifies the chip select (CE) and provides the row address of the page to be targeted. The controller then transfers the data to be programmed to the NAND device and sends a final `program` command to complete the operation.

It is important to note that a page on an SSD cannot be written more than once without first performing an `erase` operation. This is because an `erase` operation is required to clear the page of any existing data before new data can be written to it. As a result, every time a `program` operation is performed on a page, it must be preceded by an `erase` operation. This ensures that the page is ready to accept new data and that the `program` operation is successful.

It is also important to say that pages need to be written in consecutive order within the block; page number 0 is to be written first followed by page 1. Writing out of sequence is not allowed, as violating this rule aggravates the bit error rate. A single block does not need to be written all at once. That is, a block can be written with pages from 0 to 11, and later on with pages from 12 to 32, for example. Generally, pages need to be written as a whole at once, though some memories support so-called *partial page programming*, which allows a subpage of 512 bytes + correlated spare area to be written.

The data to be written will be provided by the host or result from firmware internal data management. Firmware first transfers the data from cache to the NAND internal cache register. Once the data transfer is completed the programming should start; i.e., writing to actual NAND cells.

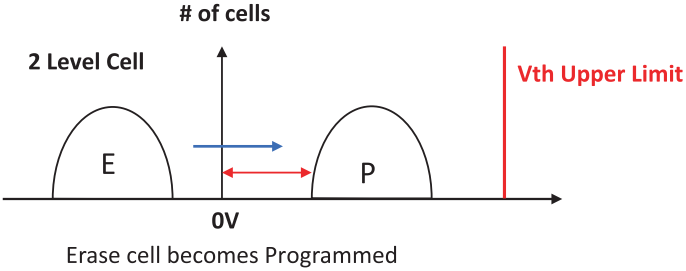


Figure 4-12. *Program level Vth distribution*

Read

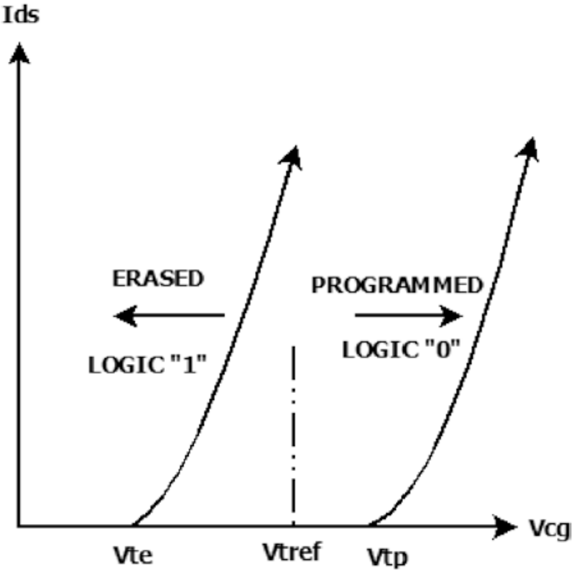
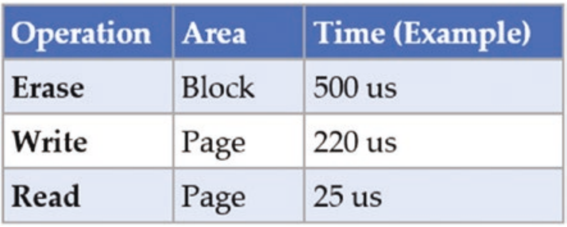


Figure 4-13. *Read sensing graph*

From the perspective of the NAND chips themselves, the `read` operation involves activating the appropriate word line to select the desired page of cells, and then reading the data stored in those cells by sensing the voltage levels on the bit lines. The NAND chips are organized into blocks, which are further divided into pages. Each page stores a fixed amount of data, typically 4 KB to 16 KB (or more), depending on the specific NAND device.

To read a specific page, the controller must first locate the block that contains the page and then activate the appropriate word line to select the page within that block. The `read` operation is typically performed by the SSD's controller, which uses firmware to manage the communication with the NAND chips and handle the necessary data transfer and error correction. The firmware is responsible for optimizing the read performance by minimizing the number of accesses required and maximizing the data transfer rate.

Table 4-2. *NAND Basic Operations Timings*



Operation	Area	Time (Example)
Erase	Block	500 us
Write	Page	220 us
Read	Page	25 us

Program/Erase Cycle (P/E Cycle)

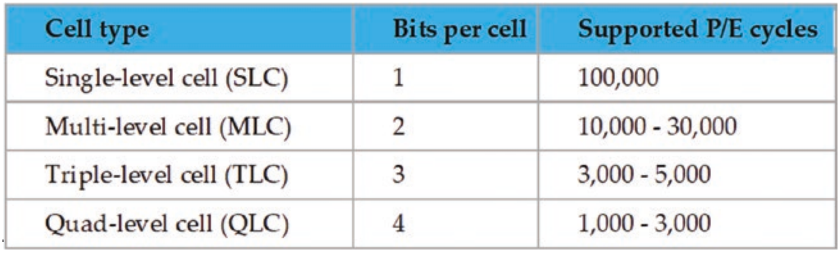
The program/erase (P/E) cycle is a fundamental aspect of NAND flash memory, which is commonly used in SSDs. NAND flash memory works by storing data in cells that are grouped into blocks. Each cell can store a single bit of data, and a group of cells is needed to store a larger amount of data. To write new data to a cell, the cell must first be erased, which is done by applying a high voltage to the cell. As we already explained, this process is known as the erase cycle.

Once the cell has been erased, new data can be written to it using a process called programming, which involves applying a lower voltage to the cell. The process of writing new data to a cell by first erasing it and then programming it with new data is known as the P/E cycle. The P/E cycle is a key factor in the endurance of NAND flash memory, as the erase cycle can cause wear on the cells over time. As a result, NAND flash memory has a limited number of P/E cycles that it can withstand before it begins to degrade. This is known as the endurance of the memory.

To extend the endurance of NAND flash memory, it is important to minimize the number of P/E cycles that the memory undergoes. One way to do this is to use the `TRIM` command, which allows the operating system to inform the SSD which data blocks are no longer in use and can be erased. This can reduce the number of P/E cycles by eliminating the need to move invalid data during the garbage-collection process, which is an internal SSD housekeeping operation that manages and maintains available storage space.

The number of bits that can be stored in each cell of a NAND flash memory drive can also affect the maximum number of program/erase (P/E) cycles that the drive can support. Table 4-3 provides an overview of the different types of NAND cells based on the number of bits they can store.

***Table 4-3.** PEC Cycle Based on NAND Cell Type*



Cell type	Bits per cell	Supported P/E cycles
Single-level cell (SLC)	1	100,000
Multi-level cell (MLC)	2	10,000 - 30,000
Triple-level cell (TLC)	3	3,000 - 5,000
Quad-level cell (QLC)	4	1,000 - 3,000

As the number of bits per cell increases, the number of supported P/E cycles tends to decrease. Single-level cell (SLC) NAND, which can store one bit per cell, generally has the highest endurance, while quad-level cell (QLC) NAND, which can store four bits per cell, has the lowest endurance. It is important to consider the endurance of an SSD when selecting a drive, as a drive with a lower endurance may not be suitable for use in cases that involve a high number of P/E cycles.

Summary

This chapter has discussed the basics of flash memory, including its different types, architecture, and fundamental operations. We have seen how NAND flash memory works and how it is used in SSDs. We have also seen the different types of operations that can be performed on NAND flash memory, such as `erase`, `program`, and `read`. We have also discussed the P/E cycle, which is a key factor in the endurance of NAND flash memory.