Adesto

RENESAS

Application Note AN500A

NOR Flash Memory Erase Operation

Revision History

Version	Date	Description
Rev A	November 2019	Initial release.

Table of Contents

List of Figures

1. Introduction

In today's technology-driven world, gadgets, mobile devices and other electronic equipment rely on NOR Flash memory to store

- code for execution,
- important system parameters,
- calibration data,
- data logs, and
- other applications requiring fast non-volatile memory.

The purpose of the document is to provide information about the internal processes and algorithms occurring during an Erase operation. The description is at a level that allows the user/systems designer to design a robust and functional system. It also discusses the procedures to follow to ensure an orderly execution and completion of these operations, while avoid situations that might affect data integrity.

A fundamental principle of the NOR Flash memory is that it must be erased before it can be programmed. Another important characteristic is that the erase operation must happen over an entire block of memory simultaneously (in bulk), rather than sequentially in a byte-by-byte fashion.

This means the Erase operation consists of three distinct phases: Pre-Program, Erase, and Recovery.

To enable robust and reliable operation, it is important for the system designer to understand these underlying processes, as well as the potential undesirable side-effects if any of the three phases are asynchronously interrupted intentionally or accidentally (for example: a programmatic erase-suspend operation or a power supply brown-out or glitch).

The descriptions in this document are general; they do not specifically target a particular Adesto Flash device.

The diagrams and pictures are conceptual in nature.

For details about the commands, timing and limitations. please refer to the device datasheet.

2. Storing Information in a Flash Memory

The basic storage element (cell) of a memory is comprised of a MOSFET with a “floating gate”. The amount of electrical charge trapped inside the floating gate of the memory cell defines whether the cell is **Programmed** or **Erased**.

Figure 1 depicts the physical composition and the symbol of a floating gate MOSFET.

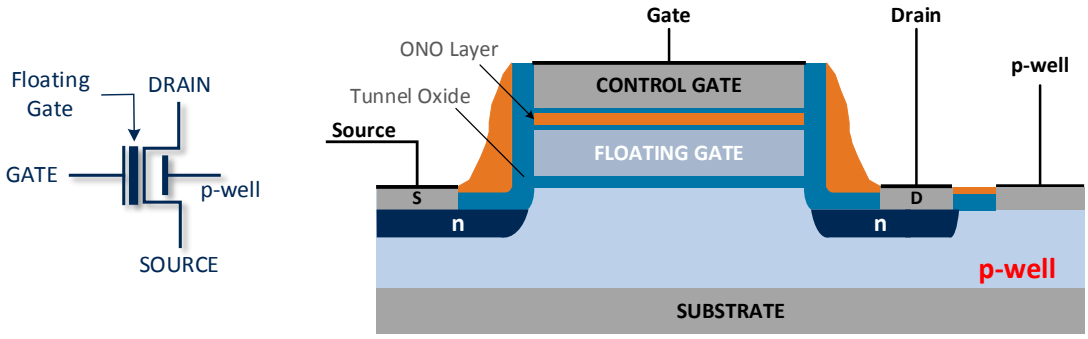

Figure 1 Symbol and structure of a floating gate MOSFET

To store large amounts of information, the Flash memory is organized into arrays of memory cells. The memory array is a matrix of N rows by M columns. At each row-column intersection there is exactly one memory cell (a MOSFET with a floating gate). This MOSFET stores one bit of information.

The capacity of the memory array (in bits) is calculated as N [rows] x M [columns] (see Figure 2)

By convention, the rows are called WORD-LINES (WL) and the columns BIT-LINES (BL).

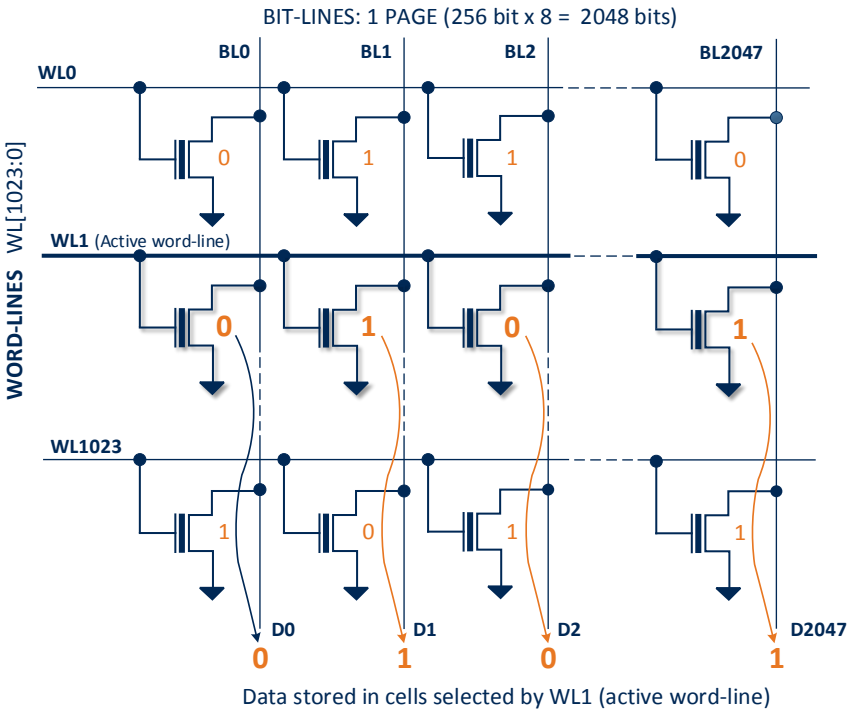

Figure 2 Organization of the Memory Array

A PAGE is a collection of bit-lines attached on a single word-line. One word-line can contain one or more pages. One PAGE typically consists of 256 or 512 bytes of memory (2048 or 4096 bits).

Silicon fabrication requires that all memory cells be built upon a common foundation, called a **substrate**.

To make memory operations more manageable, a range of p-wells are implanted into the substrate. These p-wells are electrically insulated from each other and are called **common p-wells**, as depicted in Figure 3. Arrays of memory cells are embedded into the common p-wells. Each p-well has its own electrode, allowing it to be biased with specific voltages under the control of the memory controller.

The collection of memory cells embedded into a single common p-well, from this point on in the document, is referred to a **Physical Block** or **Partition** (these two terms are used interchangeably in the rest of this document).

The typical size of a Physical Block is 1MB. It can vary from device to device, depending on device family, memory organization, and capacity. Typically, this information is not spelled out explicitly in the datasheet.

An important point is that in each Physical Block the p-well and the bit-lines are shared. Under some circumstances, this has a profound effect on device operation, as described later in this document.

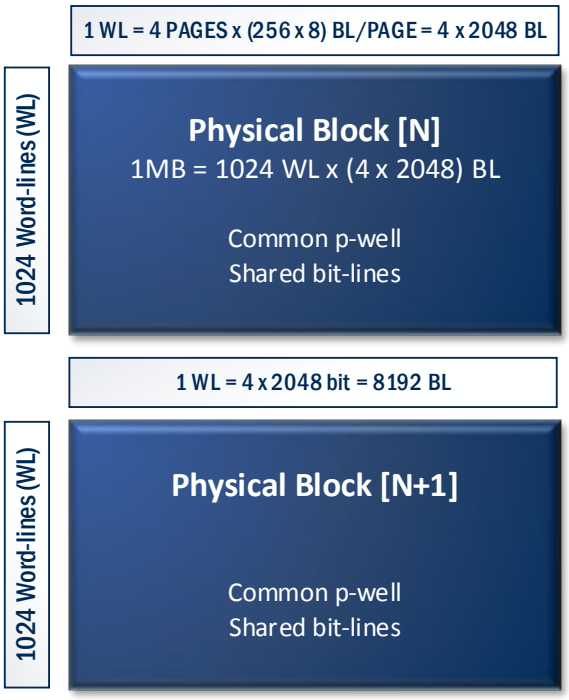

Figure 3 Organization of a Physical Block / Partition (with common p-well and shared bit-lines)

For convenience, the Physical Blocks are further divided into smaller **Logical Blocks**, allowing the user to work with smaller, more manageable memory sizes (Figure 4).

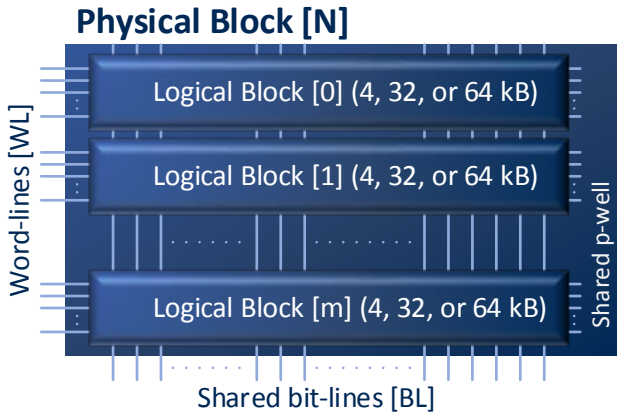

Figure 4 Detailed view of a Physical Block / Partition, containing a range of Logical Blocks. Note the shared p-well and bit-lines

The typical block sizes are 4 kB, 32 kB, and 64 kB. Additional benefits of using smaller memory blocks is that certain internal operations can be pipelined and parallelized, resulting in improved erase performance (shorter erase times).

This type of partitioning is logical in nature. These blocks are still part of the larger Physical Blocks.

Figure 5 shows a picture of an actual 16 MB Adesto Flash memory device with 16 Physical Blocks [15:0], one MB each.

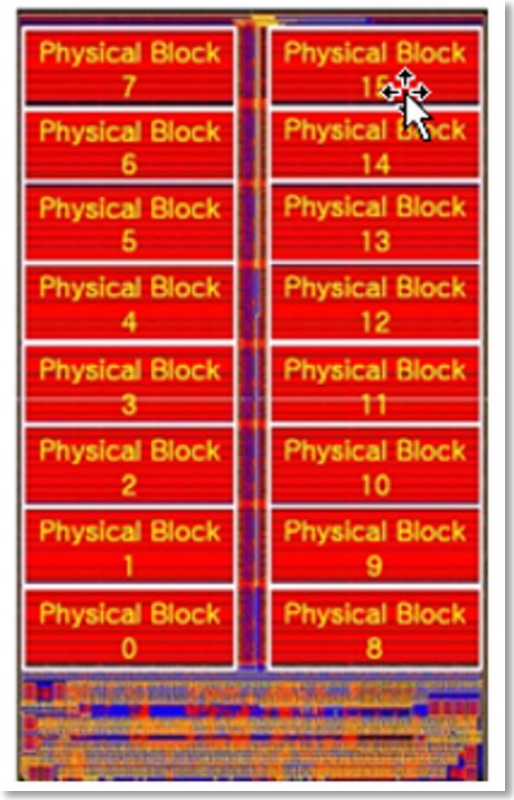

Figure 5 Organization of a 16 MB device with 16 one-MB Physical Blocks

3. Flash Memory Operations

There are three main operation that can be performed on any flash memory: **Read**, **Program** and **Erase**.

- A key concept inherent to NOR Flash technology is that before a previously programmed memory block can be (re)programmed, it first must be erased.
- Another important characteristic is that the Erase operation involves pre-programming of already erased cells.

While an operation is being executed, after the completion of each step, internal checks are performed to ensure the operation in progress is executing properly and on-track to accomplishing the objectives.

To successfully erase a NOR Flash, all three operations permissible on a NOR Flash (erase, program and read) are internally invoked as part of the Erase operation. Thus, to fully understand Erase, we must also get familiar with Program and Read operations.

3.1 Read

To determine the state of a cell (whether it was programmed or erased), the content of the cell must be read. This is done by applying a reference voltage of approximately 5.5 V connected to the GATE of the MOSFET. This is called the "reference threshold voltage" and is denoted by V_{TREF}. If the MOSFET conducts current at that level, it is closed; if does not, it is open. (See Figure 6; here we have two MOSFETS, Q1 and Q2).

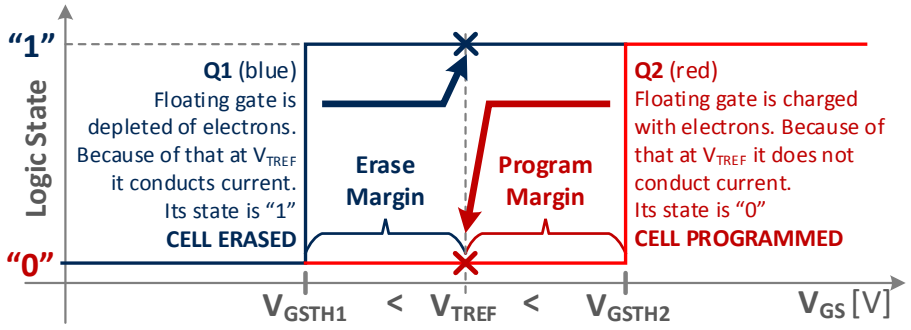

Figure 6 The process of determining the state of a cell by using a reference threshold voltage

Q1 is ERASED (blue curve). Its floating gate is depleted of trapped electrons. Because of that its GATE-SOURCE threshold voltage is smaller than V_{TREF} ($V_{GSTH1} < V_{TREF}$). When determining the state of this cell, a reference voltage V_T = 5.5 V is applied to its GATE. Under such conditions, the MOSFET conducts current (blue cross in Figure 6); thus, it is in a logical "1" state: **CONDUCTS CURRENT → '1'**.

Q2 is PROGRAMMED (red curve). Its floating gate has high concentrations of trapped electrons; as a result, its GATE-SOURCE threshold voltage is larger than V_{TREF} ($V_{GSTH2} > V_{TREF}$). When determining the state of this cell, a reference voltage $V_T = V_{TREF}$ = 5.5V is applied to its GATE. If it does NOT conduct current (red cross in Figure 6), it is in a logical "0" state: **DOES NOT CONDUCT CURRENT → '0'**.

Often, the read operation is performed as part of internal algorithms as an intermediate step to verify whether a cell was sufficiently programmed or erased. In these cases V_T is set to any value between 0 and the maximum allowable for that device, suitable for the objective of that operation.

As discussed in the section introduing the concepts of Pre-program and Erase, during Pre-program V_T is set to a voltage level corresponding to “Pre-program Verify”; during an Erase operation to “Erase Verify”.

Based on the results of these intermediate checks, the algorithm keeps repeating the on-going operation (Erase or Program), until the required reference voltages are reached, or until a predetermined number of attempts had been reached. These concepts are discussed in more detail later in the document.

3.2 Program

The Program operation is based on the physical process called “Hot Electron Injection”. The GATE of the MOSFET is connected to a large positive voltage (+9 V). Simultaneously the DRAIN is positively biased to about +4 V, causing electron flow from the SOURCE to DRAIN. The electrons, on their way from SOURCE to DRAIN, pass underneath the positively biased floating gate, as depicted on Figure 7.

Due to the strong positive electrical field applied to the GATE, some of the electrons are “pulled into” the floating gate. Once inside, these electrons no longer have the required energy to escape the confines of the floating gate, and they remain trapped: the cell is programmed.

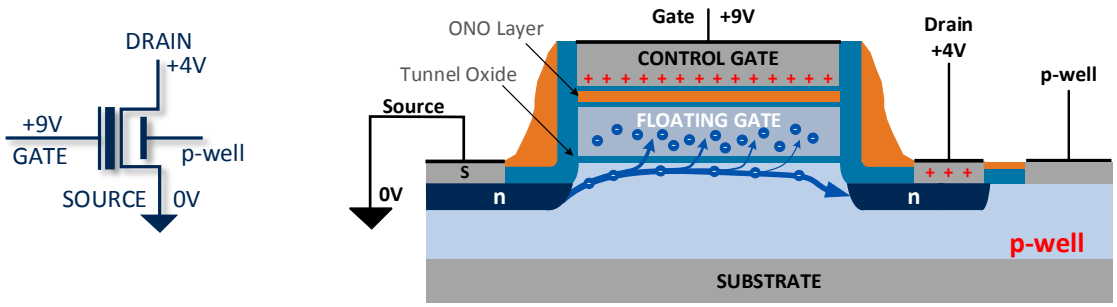

Figure 7 Programming a memory cell

The GATE threshold voltage, required for the MOSFET to close (to start conducting current) is largely determined by the number of electrons trapped inside the floating gate, in other words, how “strongly” the MOSFET was programmed. The larger the concentration of trapped electrons inside the floating gate, the stronger the cell was programmed, and the higher its GATE-SOURCE threshold voltage (V_{GSTH}).

For a cell to be placed into a valid programmed state, its threshold voltage must be larger than ~5.5 V. Ideally, there is some margin, so the desired voltage is at least 6.5V

3.3 Erase

The Erase operation happens through the process known as “Fowler-Nordheim Tunneling”.

A high, positive voltage (approximately +8 V) is applied to the common p-well, attracting trapped charges inside the floating gate of the MOSFET. Simultaneously, a high negative voltage (-10 V) is applied to the GATE of the MOSFET, which repels the trapped negative charges. The SOURCE and DRAIN of the MOSFET are left unconnected (see Figure 8).

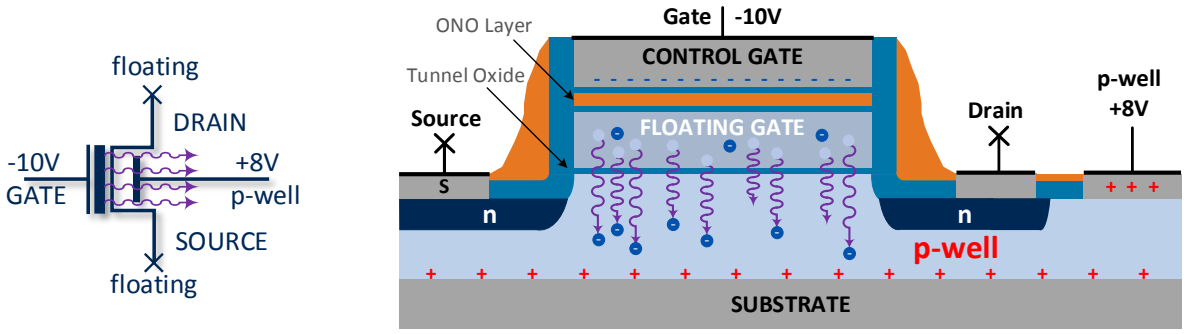

Figure 8 Erasing a cell via Fowler-Nordheim Tunneling

These two voltages superimpose and create a large electrical field that enable electrons to “tunnel” through the oxide barrier. Through this process the floating gates are depleted of electrons, lowering their threshold voltage. The currents are minute if looking at the individual cell level (typically in the nA range). Because of the very small currents, the electrical fields must be maintained for a long time to extract sufficient numbers of electrons, making the Erase operation comparatively slow.

3.4 Program vs. Erase

Turning a “1” into a “0” is called a **Program** operation. It is possible to individually program a cell.

Turning a “0” into a “1” is called an **Erase** operation.

The way the NOR Flash memory is structured, it is impossible to individually erase memory cells. An erase must happen in bulk, by erasing entire logical blocks simultaneously.

When comparing the Erase and Program operations, we can observe a few important facts:

- Program is a fast operation; it requires more energy than an erase, but it is more concentrated in time: one byte or word at a time, which makes it a more controlled operation.
- Programming is typically performed iteratively, by applying programming pulses. Each programming pulse injects a certain number of electrons into the floating gate. At the end of each pulse, the threshold voltage is verified. If the desired level has not yet been reached, another programming pulse is applied. This process is repeated until the desired margins are achieved, or until a specified number of attempts has been reached and the controller “gives up”. For the devices that support this feature, an ERASE/PROGRAM ERROR (EPE) flag is asserted.

The erase operation is slow compared to program and read operations because the “extraction” of trapped electrons (tunneling) from the floating gate is a slow process; also because it happens

concurrently over an entire logical block of memory, where each cell starts from a slightly different level of trapped charges.

For a typical Flash device, it can take 60 ms, 200 ms, and 350ms to erase a 4 kB, 32 kB, or 64 kB block, respectively. Just like the program, the erase operation is also iterative.

While the Program operation is faster than the Erase operation, (typically <5 µs for a byte), it also happens at a finer resolution (at individual byte or Page size, which is typically 128 or 256 bytes).

IN SUMMARY:

In a NOR Flash memory, **programmed** bits are stored as logical “0s” and **erased** bits as logical “1s”.

Operation	State Transition	Individual or Bulk Operation	Speed
Programming	”1” → “0”	Byte or Page level (1..256 Bytes)	Fast operation
Erasing	“0” → “1”	Bulk (typ. 4, 32, 64 kB or chip)	Relatively slow

4. V_T Distribution

To better explain what is happening during a Block Erase operation, we must introduce the concept of the V_T distribution curve.

The V_T distribution curve is a histogram that plots the number of cells as function of V_T. It is similar to Figure 6, except it is plotted over the entire population of the memory array.

V_T is a threshold voltage that is required for a MOSFET in the cell to conduct electrical current at a specific level (typically 10 µA). It correlates with the amount of trapped charges in the floating gates. Cells with a lot of trapped electrons (programmed cells), have a high V_T, typically 6.5 V or higher.

Cells depleted of electrons (erased cells) have a low V_T, typically 4 V or lower.

During an erase, multiple bits are erased in parallel; thus, the distribution of V_T can be quite large. It is possible for a memory cell to be put into an **Over-Erased** state. An over-erased cell conducts a small amount of current even when its GATE voltage is close to 0V; this is called leakage current. Such a state is problematic for a NOR Flash architecture because of shared bit-lines; it must be avoided.

Figure 9 shows the V_T distribution of a 32 Mb (4 MB) NOR Flash device that was programmed with a checkerboard pattern, where half of the cells (16 Mb) are programmed, and the other half (16 Mb) erased.

The diagram consists of two distinct humps: one corresponds to Erased cells, peaking at around V_T = 3 V; the other to Programmed cells. This one peaks at approximately V_T = 8 V.

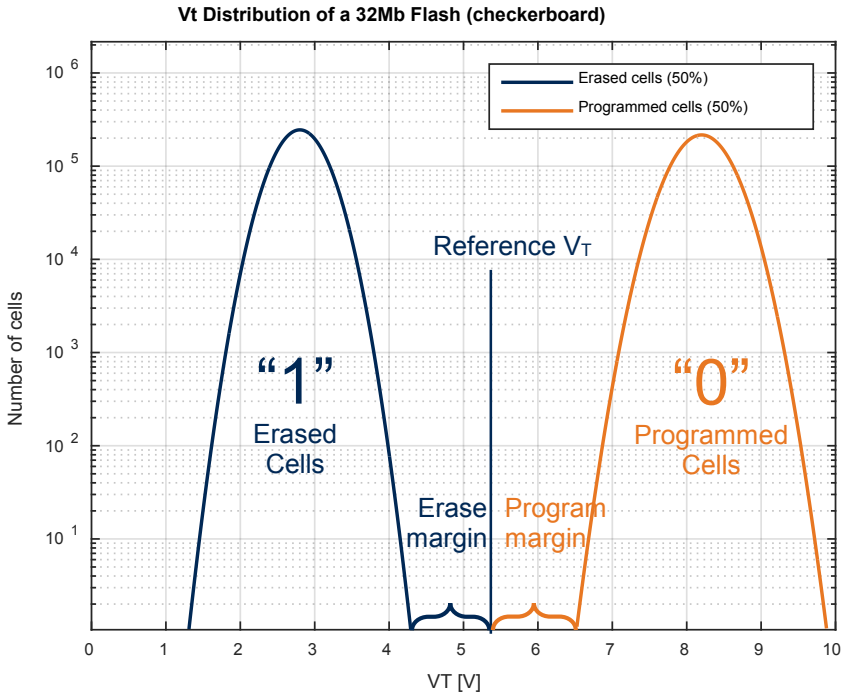

Figure 9 VT distribution curve for a checkerboard pattern

For a device that is fully erased or programmed, the distribution curve contains only a single hump.

Note:

- the scale on the y-axis is logarithmic
- the humps approximate a Gaussian Normal Distribution

The middle of the V_T range between the erase and program typically is used as a reference threshold voltage (V_{TREF}) to determine whether a particular group of cells are programmed or erased.

This voltage is applied to a selected word-line. In Figure 9, V_{TREF} is set to 5.5 V. This sets the gate voltage to V_{TREF} for the entire page (Figure 2). Depending on the state of the floating gate of each cell and which bit-lines are decoded/selected, an analog signal is output to the shared bit-lines. Sense Amplifiers and comparators are used to determine whether the output of each cell is a logical 1 or 0.

Cells that have a logical 1 as their output are the erased cells; those whose output is logical 0 are the programmed ones.

Technically, this is how the normal Read operation is implemented in the chip.

The V_T Distribution Curve is obtained by sweeping the word-line voltage (WL) from 0 V to 10V (typically), in 100 mV steps, while simultaneously applying 800 mV to the selected bit-lines. As WL voltage sequentially increases from lower to higher voltages, the number of cells that switch state from Erased to Programmed are recorded on the y-axis.

As the device cells are erased and programmed, the state of the cells (their V_T) gets distributed. If we take a snapshot at any particular moment in time, we find that each device develops its own unique distribution. The distribution dynamically changes during the lifetime of the device after numerous programs and erases.

Toward the end of the lifetime of the device, the cells gradually lose their ability to be programmed and erased and ultimately get “stuck” in a narrow V_T range.

For reliable operation, it is important to have a margin between programmed and erased cells, to ensure that over a prolonged usage the charges do not flip from “0” to “1”, or vice versa, as shown in Figure 9.

The integrated memory controller executes iterative algorithms that ensure sufficient margins are reached for each program and erase cycle. Between iterative erase/program operations, verification read operations are performed at specific targeted V_T voltages (“Pre-Program Verify”, “Erase Verify” and “Recovery” lines on the V_T diagram).

5. Block Erase

During the life-time of a NOR Flash, the device is erased and programmed many times. In this process, the V_T distribution changes because the state of individual cells is changing from PROGRAMMED to ERASED, and vice versa.

To maintain a nice V_T distribution with good margins, the Erase operation is done in three distinct phases:

1. Pre-Program Phase
2. Erase Phase
3. Recovery Phase

As discussed earlier, Erase implies a state transition from ‘1’ to ‘0’. Performing a Block Erase over a large number of programmed cells on Figure 9 means shifting the PROGRAMMED cells (orange curve) to the left, to the position where the ERASED cells (blue curve) currently reside.

Considering that the Erase operation affects all cells simultaneously, both humps (orange and blue curves) shift to the left.

The ERASED cells (blue curve) theoretically ends up in the negative V_T territory. That is not possible physically, so, these cells become over-erased with V_T close to 0 V, leaking current even when their GATE voltage is set to 0 V. This, of course, is bad.

To prevent this from happening, the already erased cells are first pre-programmed. The objective is to create a distribution that is easier to manipulate as a single, compact group. Pre-program is done at the individual byte level; pre-programming is done only on those cells requiring it, the rest are skipped.

In the next few sections, each of the three phases of the Erase operation is discussed in more detail.

5.1 Pre-Program Phase

The purpose of Pre-program is to create an ideal condition for the Erase operation. If a block of memory is erased without being pre-programmed, the cells already erased becomes over-erased.

To avoid creating over-erased cells, all erased cells must be Pre-programmed first. On the V_T diagram (Figure 10) this is represented by shifting the distribution of the erased cells (dotted blue line) to the right, into the programmed region (dashed blue line).

After Pre-program, all cells inside the Flash device are programmed.

After Pre-program, the originally programmed cells (orange dashed line), as well as the pre-programmed cells (blue dashed line), produce a combined distribution represented by the solid red curve.

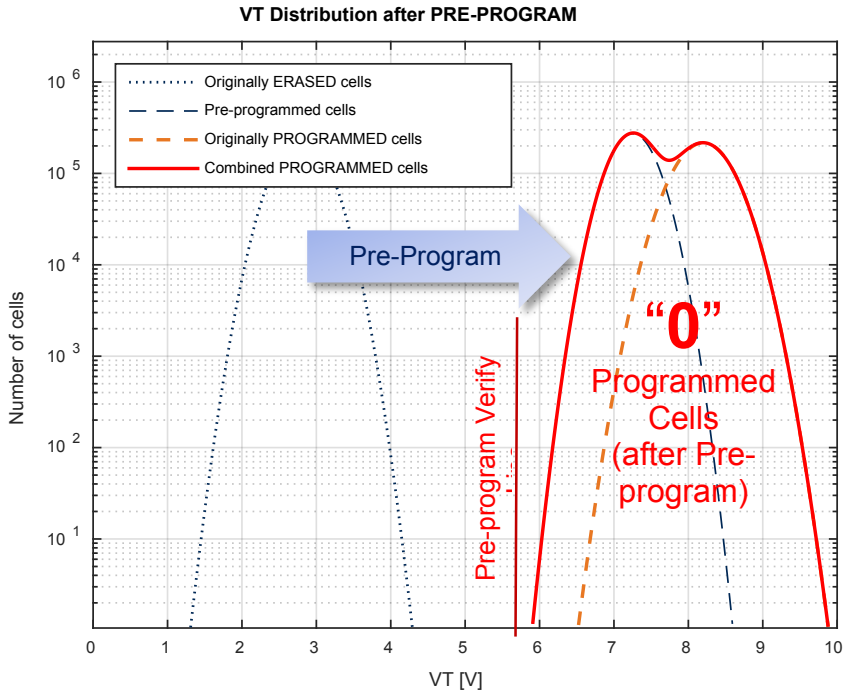

Figure 10 Pre-programming erased cells

5.2 Erase Phase

The **Erase** of an individual cell was already explained in section “3.3 Erase”. Conceptually, the Erase Phase of a Block Erase is very similar, except in this case an entire logical block is operated on simultaneously (instead of a single cell).

A large electrical field – created by a positive electrical voltage (+8 V) applied to the common p-well of the Physical Block, and a negative voltage (-10 V) applied to all word-lines within the Logical Block being erased – enable the electrons to “tunnel” through the potential barrier. The floating gates get more and more depleted, lowering their V_T voltage, as shown in Figure 8.

This process is repeated with multiple erase and verify pulses, until the V_T of all cells moves below (to the left of) the Erase Verify Line. Graphically this is represented by shifting the composite pre-programmed cell distribution to the left, as depicted in Figure 11.

During this process, some cells can move close to the over-erased territory. As discussed above, such cells may conduct current, even when WL is inactive and its V_T set to near 0 V.

Because it is done in bulk, and Erase must continue until the slowest cell verifies, the Erase operation can result in a large distribution.

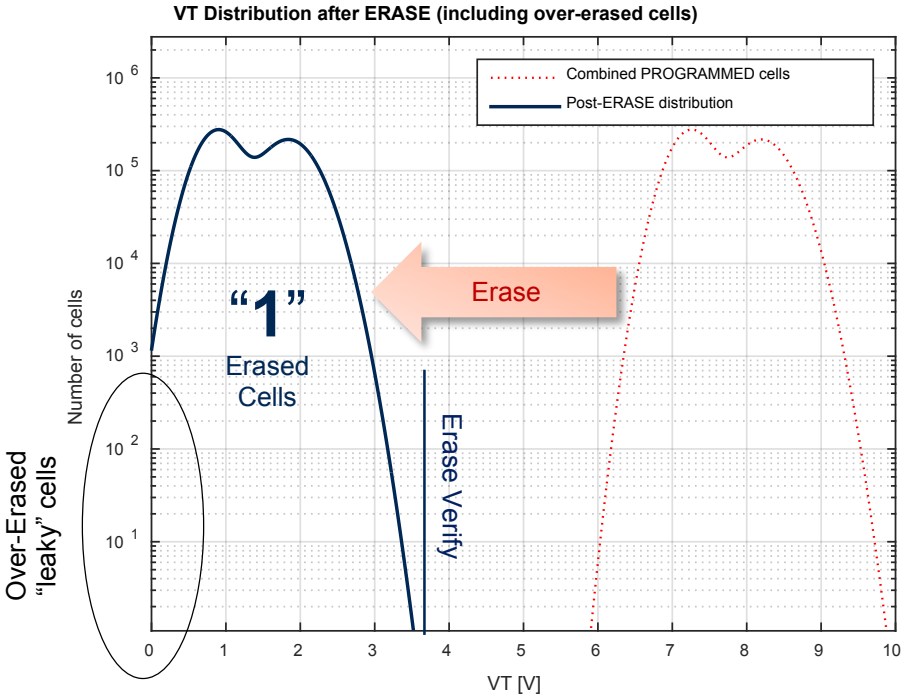

Figure 11 V_T distribution after the Erase phase

5.3 Recovery Phase

Over-erased cells are systematically soft programmed (on a byte-by-byte basis) until their V_T shifts sufficiently to the right, so that it crosses the Recovery Line from the left, as depicted in Figure 12.

Recovery adjusts the threshold of any bits over-erased during the Erase phase.

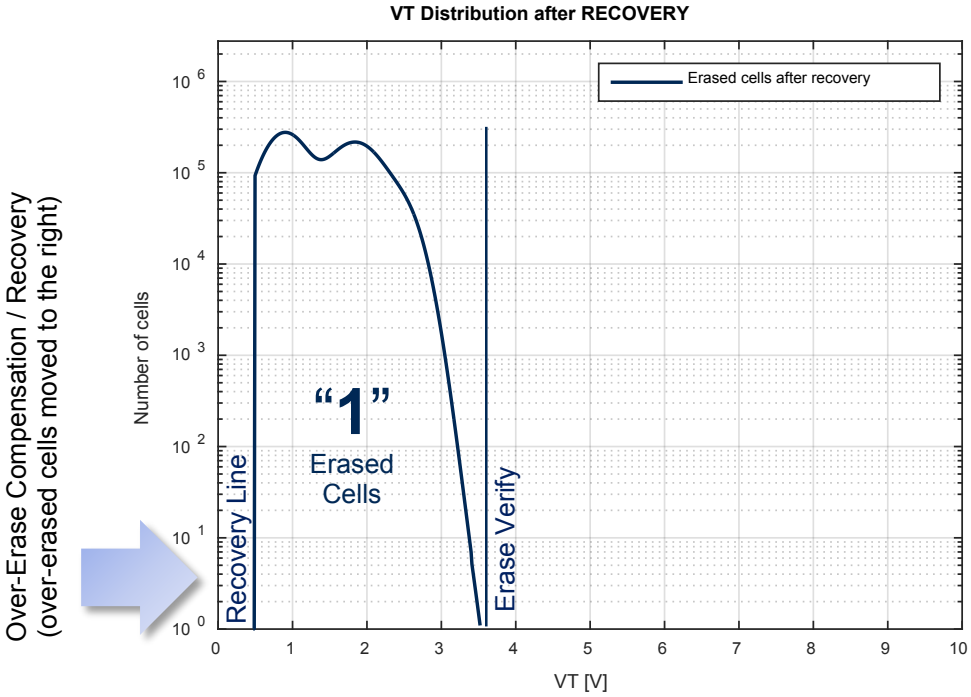

Figure 12 Over-erase Compensation / Recovery phase

While these internal algorithms are in progress, the Flash memory's integrated controller keeps a user-readable status-bit asserted (BUSY/READY bit in the status register). This bit indicates whether an on-going internal operation has been completed or not.

The user MUST NOT reset the device, remove the power from the chip, and/or interrupt the Erase operation in any manner while the BUSY bit is set.

6. Erase Interruption

If the Erase operation is interrupted by resetting the device or by turning off and on its power supply, the three phases of the erase operation do not have a chance to complete, and this may leave the memory in an undefined state. Thus, the bits in the target block can be in any of the three states: programmed, erased, or over-erased. In this case, it is safe to assume that the read data cannot be trusted.

There are several distinct mechanisms under which the memory device can return false data if an Erase process was interrupted:

1. If an interrupt occurred due to a hard or soft reset, power-supply brownout, or glitch, the memory controller is reset and completely “forgets” that an erase operation even started. It cannot continue where it left off, leaving the memory in an undefined state.

   Depending which phase the Erase operation was in when the interruption occurred, multiple conditions are possible:

   a. Interrupted the Pre-program phase -- some cells programmed, some erased.

   b. Interrupted the Erase phase – some cells erased, some programmed, some over-erased.

   c. Interrupted the Recovery phase – some cells erased, some over-erased.

   For a., the corruption is obvious, since the targeted block has nonsensical data.

   For b. and c., the logical block being erased can appear fully erased, but, in fact, some of the bits may not have a sufficient erase margin or be over-erased. An insufficient margin can lead to early life failures of these bits.

   Over-erased bits, as discussed earlier, result in leakage on the shared bit line, which leads to reduced program margin for all other bits sharing the same physical bit line. In this way, an interrupted erase can create a negative side effect for the entire shared Physical Block.

Figure 13 illustrates how a “leaky cell” on an unselected word-line can affect the reading of a “0 cell” on a selected Word-line if there is a leaky cell on a shared bit-line.

In this example there is an over-erased (leaky) cell on intersection of WL[65] and BL[1082]. The currently selected Word-line is WL[241]. Although not likely, it may be possible for the amount of leakage (originating from the leaky cell on a shared bit-line) to be high enough so that the stored “0” value in the selected cell (intersection of the Active Wordline WL[65] and shared bit line BL[1082]) to be incorrectly read or detected as “1”.

If the number of leaky cells increases on a shared bit line, due to the cumulative effect of the leakage currents, the probability of error increases (a “0” on a selected Word-line to be erroneously read as “1”).

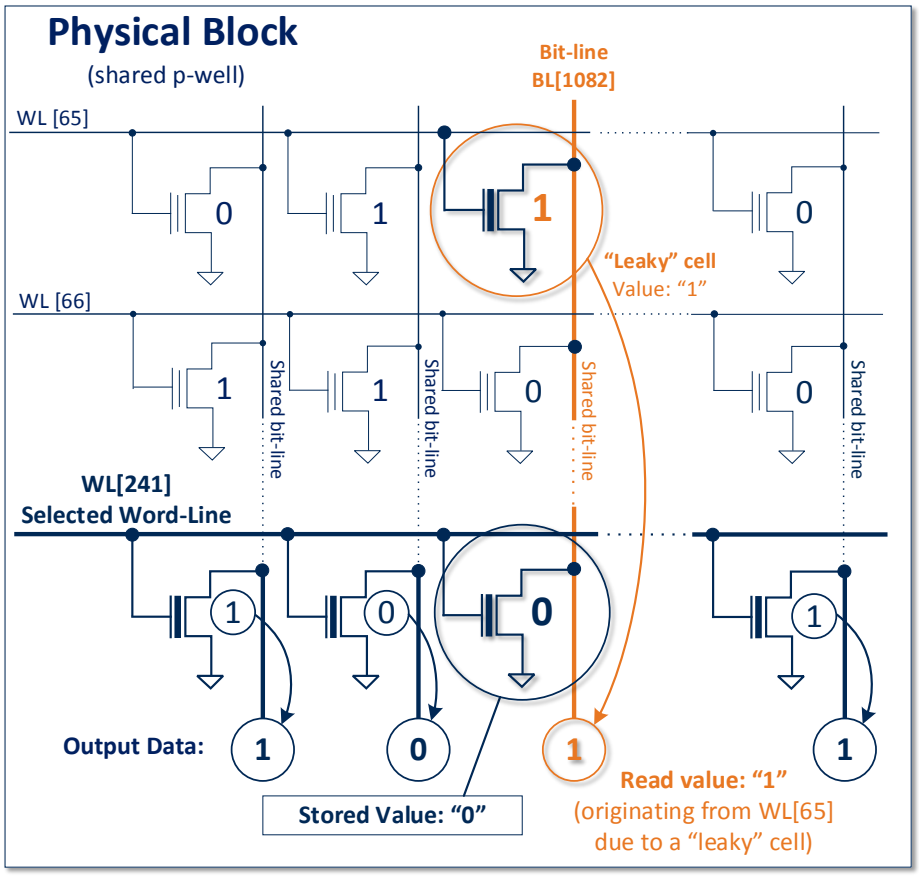

Figure 13 Data corruption as consequence of over-erased (“leaky”) cells

2. An Erase Suspend command also interrupts the erase operation. So, the same effects as above can occur. The difference is that in this case the device has a SUS bit (indicating that a SUSPEND operation is in progress), which the user/designer can track. By taking into consideration the state of the status bit, a successful completion of the operation can be ensured, if proper pre-cautions are taken.

While it is unlikely that an over-erase can corrupt data within the same Physical Block, it is good practice not to map data to be programmed or read during a suspend in same Physical Block.

6.1 Recommendations

The following recommendations help avoid the above potential failure scenarios.

1. Always check the BUSY/READY status bit for any on-going operations before issuing a soft or hard RESET and/or recycle the power supply. Failing to do so can leave the memory in an undefined state, resulting in data corruption.
2. If it is necessary to interrupt an Erase operation, such as to perform a high-priority read or program (if supported by the device), use the built-in Erase-Suspend (75h) and Erase-Resume (7Ah) commands. Note that as a result of an on-going (unfinished) Erase operation, there may be over-erased (leaky) cells in the entire Physical Block, as described previously.

   The best approach is to keep important system-level code (that needs to be read while an Erase-Suspend operation is in progress) in a separate Physical Block, away from memory used for data logging or other scratch-pad type of memory that is frequently erased and programmed during normal device operation.

   The important system-level code or data in a separate Physical Block is unaffected by any potential undesirable intermediate states in the Physical Block being Erase-Suspended.
3. Another way of dealing with systems where asynchronous erase interrupts can occur is to program a unique byte value (“signature”) to the beginning of the logical block, as the last step of each successful Erase operation.

   If there was a system crash, due to an unscheduled power-supply interrupt or glitch while an Erase operation was in progress, as part of the crash recovery, the signatures of the logical blocks are read and compared against a known good value. Discrepancy in a particular block indicates that the Erase operation was not properly completed, and a repeated Erase is required, to restore its correct distribution.

If you have further questions, Adesto’s Applications Engineering is available to provide answers and clarifications.