AN221050

Flash Analysis Techniques and Best Practices

Author: Thomas Nirschl

Associated Part Family: FL-L, FL-S, FS-S, GL-T, GL-S, KL-S, KS-S, KL-1, KS-1

Related Application Notes: AN200991, AN219790, AN200381, AN98549, AN98547, AN69061

This application note revisits the basic operation of Flash memory devices and recommends several analysis techniques.

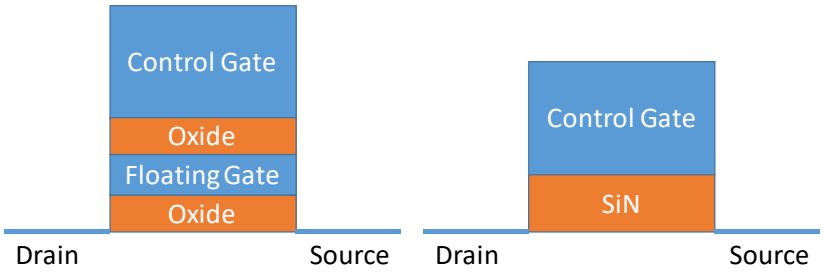
1 Introduction

Nonvolatile memory devices (like Flash) are often used in embedded applications that rely on the storage of system configuration, dynamic data, or both. This data needs to be retained during a power interrupt or power cycle. In case of a system malfunction, the reproducibility of the failure is influenced and depends on the state of the nonvolatile memory. This application note briefly explains the basic operation of Flash memory devices and recommends several analysis techniques.

2 Flash Device Operation

Figure 1 shows the simplified cross section of two of the most common types of Flash memory process technology, a floating gate memory cell (for example, FG) and a charge trapping (for example, MirrorBit) memory cell.

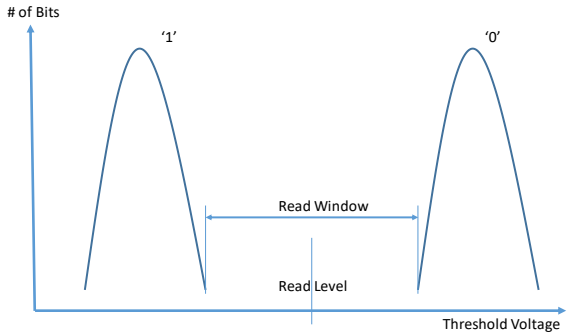
Figure 1. Floating Gate (left) and Charge Trapping (right) Memory Cells



Both technologies store information by trapping electrons on a storage medium. Logic ‘0’ and ‘1’ are defined as the presence or absence of electrons. Predominantly, logic ‘1’ is defined as the erased state, i.e., the absence of electrons, and logic ‘0’ is defined as the programmed state, i.e., the presence of electrons. The number of electrons (charge) present on the storage medium will determine the threshold voltage of the bit cell during read operation. The presence of more electrons means lesser current drawn by the memory cell which in turn is converted to a logic ‘0’ by the memory’s read amplifier. After an erase operation, no carriers will be left on the memory cell, resulting in a larger cell current, which is interpreted as a logic ‘1’ by the read amplifier.

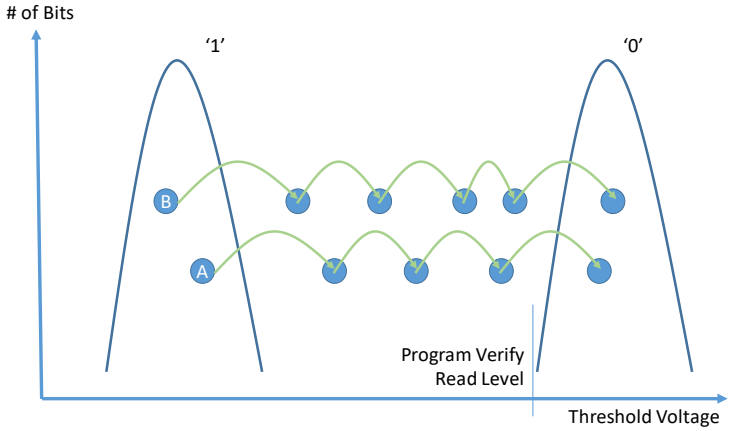
Due to statistical variability, logic ‘0’ and ‘1’ form two separate distributions as shown in Figure 2. Geometrical variations in the size of the memory cells, as well as fluctuations at interfaces and the number of electrons stored on the bit cell will influence the threshold distribution.

Figure 2. Example of Threshold Voltage Distribution of Memory Array



In modern nonvolatile memories, complex algorithms are used to shape the threshold voltage distribution. Instead of a single erase or program operation, multiple small pulses are applied to a group of memory cells to transition from one state to the other. Figure 3 shows an example of a program verify operation for two different bits.

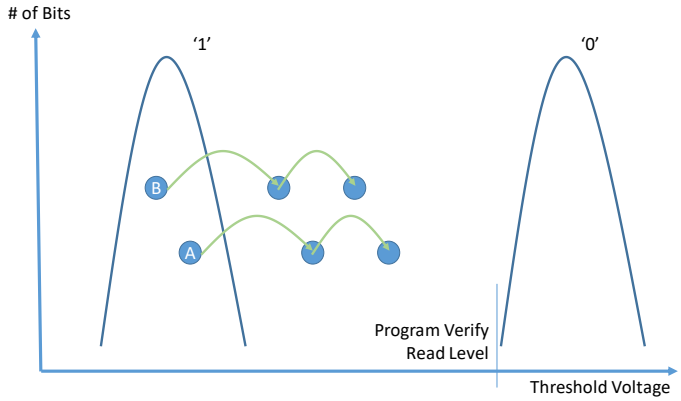
Figure 3. Example of Two Bits Programmed as a Group



This example shows Bit A requiring four programming pulses to transition from an erased level to the program verify level, whereas Bit B requires five pulses.

In the event of a power loss, a hardware reset, or a programming operation interruption, the algorithm will be abruptly terminated, resulting in so-called flyer bits in the memory device. These flyer bits are now in an intermediate (corrupted) state between erased and programmed (see Figure 4).

Figure 4. Threshold Voltage Distribution after Interrupted Program Operation



Under such circumstances, the Flash device will not be able to resume the interrupted operation on its own. Interrupted operations as shown in Figure 4 may prevent the application from booting up. Thus, it is critical that the application software ensures proper power handling and detection of interrupted states (See application notes AN200991 and AN219790) to avoid getting into a corrupted state. If not done correctly, the Flash memory may end up in a corrupted state which may prevent the application from booting up.

Additionally, it is also possible to alter the devices' configuration data such as sector protection or speed options of some types of Flash devices. If those settings are stored as nonvolatile information, special care has to be taken to prevent similar corrupted memory state due to power interrupt or hardware reset (see application note AN200381). Once the nonvolatile part of the configuration registers is corrupted, the application might not be able to boot anymore.

When returning devices to Cypress for failure analysis, it is important to leave the device in the failed state, so that possible flyer bits can be detected. Altering the state of the device (such as erasing or reprogramming the Flash device) can destroy important evidence that can allow accurate failure analysis.

3 Analysis Techniques

3.1 Configuration Registers

When updating nonvolatile configuration registers of a Flash device in the field or during an update of the application software, follow these steps:

- Verify and confirm all configuration register bits are in their correct state ('as expected').
- Verify and confirm all configuration register bits to be in a stable state, i.e., check if multiple read out across temperature result in the same information.

It is important that the application boots into debug mode without depending on external components, for example. the Flash device. This enables complete failure analysis of the application and permits isolation of the contributing device without an unnecessary swap test.

It is tempting to reset the Flash nonvolatile configuration register bits 'to check what happens', but avoid resetting as this will destroy critical evidence needed for root cause analysis at a device level. It is important to note that on customer application, it is not possible to read out the analog level of the memory cells storing the configuration register bits. However, Cypress has special test programs and test modes to determine the threshold voltage distribution of the internal sectors. In addition, Cypress can verify the consistency of the internal registers and settings of the Flash device.

Table 1 shows the configuration register of the S25FL-S as an example.

Table 1. Example: S25FL-S Configuration Register CR1

Bits	Field Name	Function	Type	Default State	Description
7	LC1	Latency Code	Nonvolatile	0	Selects number of initial read latency cycles See Latency Code Tables
6	LC0			0	

Bits	Field Name	Function	Type	Default State	Description
5	TBPROT	Configures Start of Block Protection	OTP	0	1 = BP starts at bottom (Low address) 0 = BP starts at top (High address)
4	RFU	RFU	RFU	0	Reserved for future use
3	BPNV	Configures BP2-0 in Status Register	OTP	0	1 = Volatile 0 = Nonvolatile
2	RFU	RFU	RFU	0	Reserved for future use
1	QUAD	Puts the device into Quad I/O operation	Nonvolatile	0	1 = Quad 0 = Dual or Serial
0	FREEZE	Locks current state of BP2-0 bits in Status Register, TBPROT in Configuration Register, and OTP regions	Volatile	0	1 = Block Protection and OTP locked 0 = Block Protection and OTP unlocked

As an example, let's assume the application software sets or resets the nonvolatile QUAD bit at the same time power interrupts or hardware resets occur. If this happens, there is a risk that the internal sector state gets corrupted along with its configuration register. Similar to the QUAD bit, the Latency Code registers could be unstable and this will result in read failures due to wrong timing settings.

To analyze effects such read failures, proper system application-level debug modes are required to detect the corrupted register. If the memory device gets into a state with corrupted nonvolatile configuration registers, the software application execution will follow the incorrect device settings resulting in the wrong conclusion.

If system application-level debug modes are not available, Cypress recommends reading out the memory device configuration registers and verifying it to be in a stable state using a standard programming system or an evaluation kit before the unit is soldered back to a known good or the failing board.

3.2 Read-only Applications

After the verification of the configuration registers, the read only address range should be checked for consistency:

- Does the unit contain data that matches the golden production data?
- Was there a software update in the field?
- Is the data pattern stable when comparing multiple read outs at different temperatures and voltages?

If you observe unstable data or checksum error, do not re-program the device. Record the address range, expected, and actual data. Where possible, dump multiple data from the Flash memory device to determine address range(s) or Flash sector(s) which exhibit data inconsistencies (vs. consistent data). Provide this information to Cypress for Flash device root cause analysis. Make sure to provide the address offset which might be used in application software, so that Cypress is able to determine the absolute Flash device address. It is important to provide the read settings used by the application software, i.e., the read mode and timing settings. Multiple read options are available for Flash devices offering a serial interface (SPI). Hence, a detailed problem description is essential.

Only if the device is left in the original failed state, the measurement of the analog level of the memory array, which is performed on automated test equipment (ATE) at Cypress will lead to the right conclusions of the application-level failure.

3.3 Dynamic Non-Volatile Applications

The Flash device might contain read-only portions mixed with dynamic areas, for example, Flash File System (FFS), EEPROM emulation, or application error memory.

The consistency of FFS and EEPROM emulation need to be verified for consistency:

- Does the dynamic area contain unstable data?
  - Single logical sector (i.e. erase unit) or multiple sectors

  - I/O dependence or multiple I/Os
  - Is the application power-fail saved (for example, AN219790)?

- Are FFS or EEPROM emulation header structures unique and consistent?
- Does the application software boot-up with corrupted dynamic area?
- Is the application software able to recover a corrupted logical sector?
- Are there indications for end of life cycling (See application note AN98549)?

In case of an application failure, the error memory might be updated frequently due to the system hanging up if it were to enter a loop of reset and program or erase cycles. If this occurs, two scenarios arise:

- Permanent damage to the error memory by rapid cycling (See application note AN98549)
- Over-erase of error memory

The first scenario will result in physical damage to the part and cause data corruption with no data retention. After a few rapid cycles, the damage can be observed. Heat applied during unit extraction will further age the unit causing a corrupted state in the error memory. If the failing application shows repeated resets (with a timing less than 90 seconds), stop further failure analysis to avoid the Flash device from being damaged permanently. Then, you can carry out analysis using debugger environment to prevent rapid reset cycles.

Secondly, the state of over-erasure can be caused by repeated interrupted erase operation. As described above, state-of-the-art nonvolatile memory devices use complex algorithms to perform program and erase operation. In case the embedded algorithm is interrupted repeatedly and pre-maturely with a deterministic timing caused by an application-level hang-up, the sector containing the application error memory could end up in over-erase state.

Let's consider an example where the application watchdog is not served in parallel with update attempts of the application error memory. This could cause repeated interrupted embedded operation of the Flash device and further result in a corrupted state of the application error memory. During next boot up sequence, the application will either hang-up due to corrupted error memory or result in over-erase state.

3.4 Unit Extraction

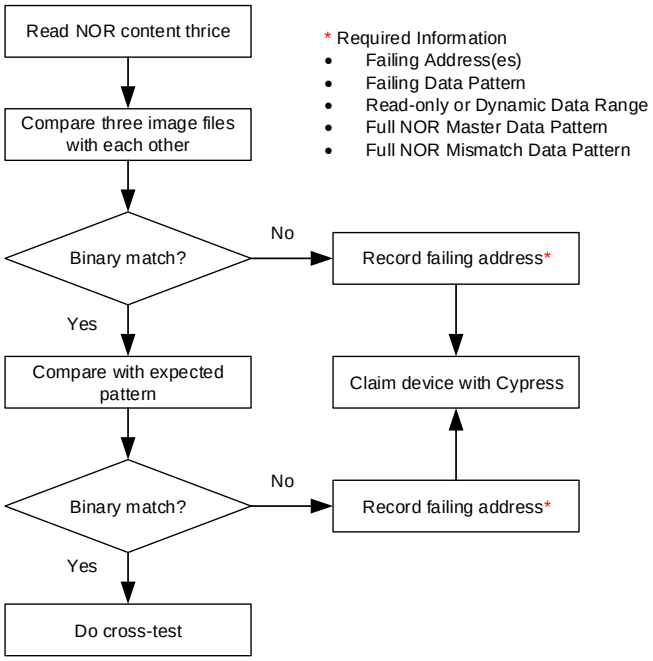
Here are some recommendations for performing unit extraction:

- Bake the application before unit extraction (IPC/JEDEC J-Std-033B)
- Minimize X-Ray inspection during production and failure analysis (See application note AN98547)
- Unit extraction:
  - Do not use hot air gun because of uncontrolled temperature applied to unit
  - Do not try manual extraction using tweezers (For example, see application note AN69061 for CSLP package)

3.5 Recommended Customer Failure Analysis Flow

Figure 5 shows the recommended Failure Analysis Flow which can be used as a guideline to collect the necessary information before claiming the device with Cypress. Document all steps and provide the detailed problem description to Cypress.

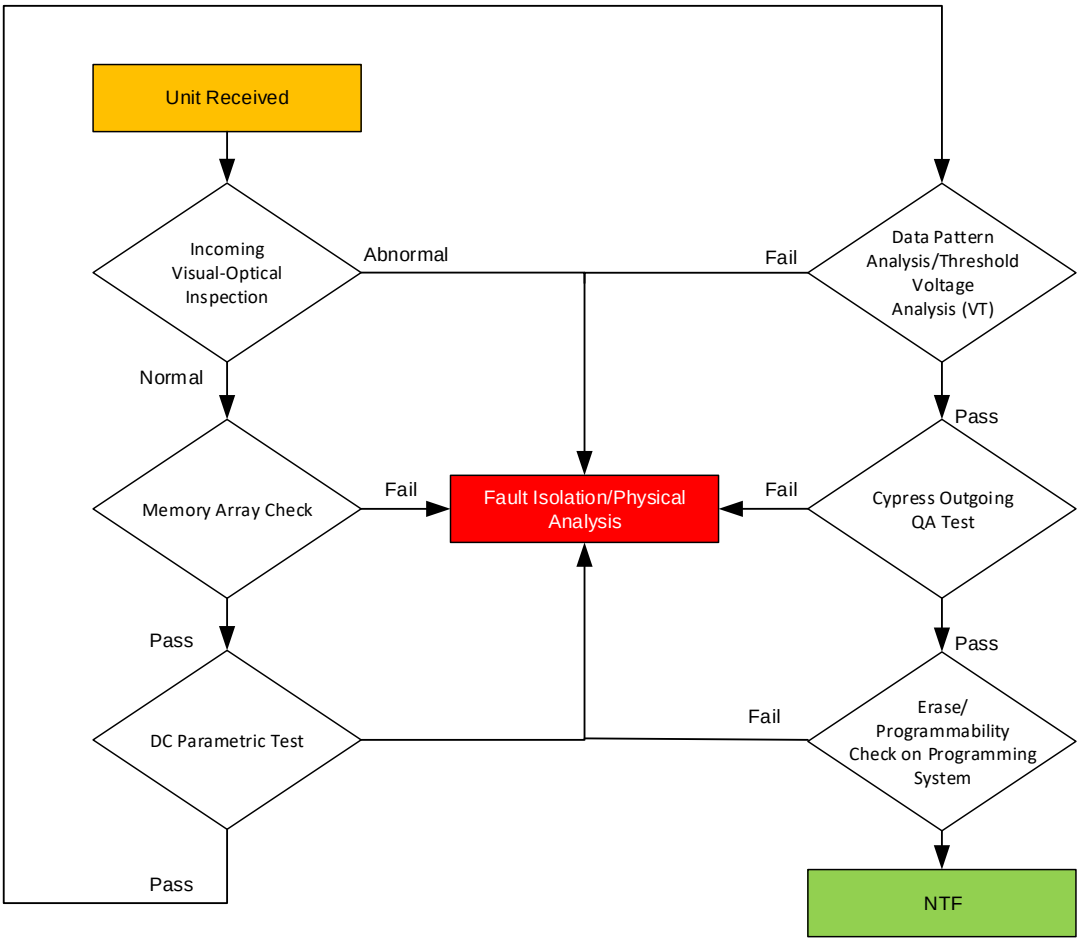
Figure 5. Customer Failure Analysis Flow



3.6 Cypress Analysis Flow

Upon receipt of the complaint unit, the failure analysis begins as shown in Figure 6 . Most items are standardized so that a complete read-only data set is available, i.e., the state of the Flash device will not be altered. Since electrical and physical fault isolation are unit specific, this box is not further described here. As a rule of thumb, the read-only pre-analysis will be finished within five days after the unit received. Electrical and physical fault isolation can take up to 30 days, depending on the complexity of the failure.

Figure 6. Cypress Failure Analysis Flow



Cypress offers to re-program customer pattern to a NTF unit before returning the device back for re-verification.

4 Summary

During the analysis of nonvolatile memories, exercise caution so that the evidence contained within the device is not destroyed. The first step should always be to perform read-only analysis. Record findings carefully to draw the right conclusions.

Record corrupted address range carefully and compare it with the master pattern (if available). Check dynamic data ranges for valid header information.

Before claiming the device, contact your Cypress FAE or Sales person to discuss the observation and possible next steps.

About the Author

Name:	Thomas Nirschl
Title:	Senior Manager, Product Engineering / Failure Analysis
Background:	Thomas Nirschl has a Doctoral Degree in EE from Technical University Munich, Germany

Document History

Document Title: AN221050 - Flash Analysis Techniques and Best Practices

Document Number: 002-21050

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5898600	THNI	09/22/2017	New Application Note
*A	6447926	THNI	01/17/2019	Sunset review

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Arm® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6 MCU

Cypress Developer Community

Community | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709