Wear Leveling

About this document

Scope and purpose

AN98521 discusses the wear leveling approach, its advantages, and methods of implementation to extend the flash and product life expectancy.

Table of contents

1 Abstract

Today's embedded systems utilize flash devices for both code and data storage applications. Some of these applications use intensive programming and erasure operations which raises concerns about flash product life expectancy. Flash memory devices are limited to a finite number of Program Erase (PE) cycles. An embedded system's product life expectancy is tied directly to the flash usage model and the number of PE cycles guaranteed by the flash device. Utilization of appropriate systems tools like wear leveling can extend the flash and the product life expectancy. It is good practice to consider the flash usage model in a system to determine if wear leveling could extend flash life to meet the system requirements.

This document will highlight three areas:

- What is wear leveling?
- Advantages of using wear leveling.
- Wear Leveling methods.

2 Wear Leveling

The NOR flash endurance specification defines the conditions and the number of erase operations that can be successfully performed on a given flash erase unit (sector). Infineon offers two types of non-volatile NOR flash memory: Single-bit-per-cell floating-gate flash and Infineon proprietary two-bit-per-cell MirrorBit™ flash. These two technologies are designed to typically withstand 1,000,000 (floating gate) and 100,000 (MirrorBit) Program followed by Erase (PE) cycles.

There are cases where the flash endurance specification may not meet the projected cumulative erasures, for a given sector, in a particular system application (usage model). Flash File Systems (FFS) that employ wear-leveling algorithms provide a means to extend flash life expectancy.

Wear-leveling algorithms arrange or store data in a manner that sector erasures are distributed more evenly across the flash memory array or portions of the array. A host system using a flash file system performs its reads and writes to logical-sector addresses. A wear-leveling algorithm is typically part of a flash file system which remaps logical-sector addresses to different physical-sector addresses in the flash array. The wear-leveling algorithm tracks the least used sectors in the flash to identify the next area to write data. This process reduces or eliminates single sector failures resulting from a high concentration of erase cycles in a limited number of sectors during the expected system life time.

There is also an inverse relationship between the number of PE cycles performed on a sector and the duration of time the sector will reliably retain the data programmed into the sector. Typical data retention time is specified in the flash industry assuming no erase operations on a sector. As more PE cycles are applied to a sector the time that data will be reliably retained is reduced. Generally, this does not cause a problem for system operation because a sector that is frequently erased is likely to be erased again long before there is any danger of data loss. However, wear leveling reduces the average number of PE cycles on frequently cycled logical sectors, and thereby extends the data retention time on these sectors longer than it would be if all PE cycles were applied to the same physical sector.

Practical Guide to Endurance and Data Retention application note provides additional useful information to better understand Flash Endurance and Data Retention.

2.1 How Wear Leveling Affects Flash Life Expectancy

Below are example cases that highlight how flash (and the system) life expectancy is affected by wear leveling. These examples use a Infineon MirrorBit S29GL01GP flash device that has 1024 (128 Kbyte) sectors with a typical endurance rating of 100K PE cycles.

2.1.1 Example Application With No Wear Leveling

The first example assesses the flash device life expectancy in a system that does not utilize wear leveling. This system uses fixed addressing that maps each logical sector to the same physical sector, which results in all updates occurring at the same physical address. The sector associated with the subject physical addresses must be erased before new data can be programmed. In this example a 128 KByte data file will be updated 50 times per hour. Since there is no wear leveling the system reuses the same sector for each update.

The total flash life expectancy is:

$$= \frac{100{,}000\ cycles\ \times\ 1\ sector\ used\ in\ the\ Flash}{1 sector\ /\ file\ \times\ 50\ updates\ per\ hour\ \times\ 24\ hours\ per\ day}$$

$$< 84\ days \ll 1\ year$$

In this configuration where wear leveling is not utilized, the flash has a life expectancy of much less than one year.

2.1.2 Extending Flash Life Expectancy Via Wear Leveling

In this second example, wear leveling is utilized. This means the data file or logical sector(s) are no longer tied to a single physical sector, allowing the file programming and erasure to be distributed across a number of sectors. After a limited number of PE cycles on a physical sector, a new physical sector with less PE cycles on it is selected as the new location of data in the frequently updated logical sector. The logical to physical address map is updated to point to the new physical location for the logical sector.

Even though the logical sector may receive many PE cycles, those PE cycles are performed only a few times on each of the physical sectors that hold the logical sector during a small portion of system operational life. In this way the PE cycle wear is spread across multiple physical sectors. Sometimes, the lowest cycled sector needed for the new location of a frequently cycled logical sector already contains data. In this case the data in the low cycled sector is first copied to another unused sector before erasing the low cycled sector.

This example will use 96 of the total 1024 available sectors in the wear-leveling process. Note the 96 sectors will be programmed and erased equally; the 128 Kbyte file is programmed at a rate of 50 file updates/ hour. Distributing the Erasure cycles across the 96 sectors significantly extends the useful flash life.

The total flash life expectancy using wear leveling is:

$$= \frac{100{,}000\ cycles\ \times\ 96\ sectors}{1 sector\ /\ file\ \times\ 50\ updates\ per\ hour\ \times\ 24\ hours\ per\ day}$$

$\sim$8000 days $\sim$ 21.9 years

This example demonstrates that wear leveling increases the flash life expectancy to ~22 years.

Most file updates do not require changing large amounts of data such as the entire contents of a 128 KB sector. In the last simple example, even a small update of 512 bytes within the 128 KB sector did require copying the entire sector to RAM, updating the desired 512 bytes, erasing the physical sector, then writing back all 128 KB from RAM. This requires a great deal of copying and programming overhead for each small update and cycles the entire sector for every small update.

A more efficient FFS will provide logical to physical location mapping for much smaller blocks of data such as 512 bytes or 4 KB blocks. With this finer granularity of mapping it is only necessary to program an updated version of the modified block to an erased portion of a sector and mark the old copy of the block in the map as deleted. In this way each small update uses a much smaller portion of a sector and only when all of a sector has been used for these small updates is it necessary to copy the remaining valid blocks in the sector to a new sector and erase the sector to make room for more updates. This approach greatly increases the number of times data may be updated without requiring a PE cycle on a sector and thus further extending the update capacity or operational life of the flash.

This third example will assume the same 96 sectors of available space but updates will be performed in 512 byte blocks. There are 256 blocks of 512 bytes in each 128 KB sector.

The total flash life expectancy using 512 byte blocks with wear leveling is:

$$= \frac{100{,}000\ cycles\ \times\ 96\ sectors\ \times\ 256\ blocks}{50\ block\ updates\ per\ hour\ \times\ 24\ hours\ per\ day}$$

$$\cong\ 2048000\ days\ \cong\ 5610\ years$$

Or the block updates could be increased to 28000 updates per hour:

$$= \frac{100{,}000\ cycles\ \times\ 96\ sectors\ \times\ 256\ blocks}{28000\ block\ updates\ per\ hour\ \times\ 24\ hours\ per\ day}$$

Wear Leveling

$\cong 3657\ days \cong 10\ years$

Clearly, a modern FFS with wear leveling can dramatically increase the useful life time of the flash device to meet the needs of nearly any application.

2.2 Wear Leveling Methods

Wear leveling is a system level tool to address flash endurance limitations by arranging data so that required erasures and re-writes are distributed more evenly across the flash storage medium. In this way, no single flash sector fails due to a high concentration of erase cycles during the desired system life time. A designer needs to understand the flash usage model for a given design and its life expectancy. This information can be used to determine the appropriate wear leveling method to be implemented. Below are three simple ways wear leveling could be utilized:

2.2.1 A System with No Wear Leveling

Some applications do not need frequent updates and can meet the required system life time requirement without wear leveling. But, the system designer must recognize that the system life expectancy can be limited by frequently erasing and exhausting a single or a few sectors in a flash device. A system not utilizing wear leveling may see flash sector failures due to a high concentration of erase cycles in a limited number of sectors. Note in many cases, there may still be areas in the flash array that are either unused or under used. Effectively, the system life ends much sooner than most of the flash array because of the dependence on frequent cycling of only a few physical sectors.

2.2.2 Dynamic Wear Leveling

Dynamic wear leveling spreads the PE cycles across sectors in a portion of the total memory space where updates are done on a regular basis. Systems using dynamic wear leveling do not touch static data. In a flash usage model where 75% of the array is used for code storage or static data, only 25% is available for wear leveling. This model offers increased flash endurance compared to no wear leveling, but all updates are concentrated within 25% of the available space. The advantage is some wear leveling without requiring a means to remap all accesses to memory. The static areas can be read directly without software or hardware overhead to locate the physical sectors where code or static data is placed.

2.2.3 Static Wear Leveling

A static wear-leveling algorithm tracks the least used physical sectors across the entire flash address space and uses all sectors to more evenly distribute PE cycles over the entire flash array. If a low cycle sector is unused, update programming is simply done in the erased sector. If a low cycle sector contains static data, the data is copied to a more heavily used location before erasing and programming the low cycle sector with an update. The static wear leveling further optimizes the flash endurance compared to no wear leveling or dynamic wear leveling.

However, it requires that all read accesses to the flash go through some type of logical to physical mapping that can slow down the reading process for static as well as dynamic logical sectors. Often this mapping for code and static data is handled by a hardware Memory Management Unit (MMU) in the system microprocessor. When a static area of the flash is first accessed an interrupt handler is called to access the FFS logical to physical map and move the mapping into the MMU. The MMU then performs the logical to physical mapping at high speed (needing no additional access time) as code or static data is needed by the microprocessor.

3 Conclusion

Today's embedded systems have progressed beyond using NOR flash exclusively for code storage to more feature rich data storage applications requiring periodic data updates. The examples highlight how the flash's useful life expectancy can be dramatically improved when wear leveling is employed. It is recommended to understand the flash usage model of the intended application and utilize system tools like a FFS with wear leveling as required to meet product life expectations.

Infineon makes available flash driver and FFS software. The flash file system source code provided by Infineon delivers a production-grade block driver and stand-alone file system that are specifically designed for flash memory. This file system significantly reduces the amount of software development time, enabling faster time to market by simplifying flash memory integration.

References

[1] **AN99121 - Practical Guide to Endurance and Data Retention**

[2] **Software and Drivers for Flash Memory**

Revision history

Document version	Date of release	Description of changes
**	2008-04-30	Initial version
*A	2010-06-15	Global updates
*B	2015-10-12	Updated to template
*C	2017-07-18	Updated logo and Copyright
*D	2018-05-02	Sunset review
*E	2021-03-19	Updated to Infineon template

Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

Edition 2021-03-19

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Go to www.cypress.com/support

Document reference

001-98521 Rev. *E