Understanding system integrity and how testing can help in preventing field failures

Sami Kassimäki (Author)
Product Manager
sami@tuxera.com

Tuxera
Espoo, Finland
www.tuxera.com

INTRODUCTION

When a device fails in the field, consequences can be significant. In our conversations with embedded original equipment manufacturers (OEMs), one thing is certain: addressing a problem in the field is costly. With the complexity of modern embedded designs, root cause analysis can be difficult and painstaking. This process pulls resources for field diagnostics and post-mortem analysis – most typically from new product development – and it impacts time-to-market.

Embedded devices operate in conditions where sudden power interruptions and other hazards can occur at any time. As data storage needs of these devices has increased dramatically over the years, unreliable data storage can be a significant contributor to field failures. Failsafe and long-lasting data storage is not a simple matter and requires careful planning. This paper discusses the different methods to long-term reliability, and how thoughtful and diligent storage testing can help ensure a rugged data storage system that contributes to preventing field failures and increasing the lifetime of an embedded device.

DEFINING RELIABILITY

"I don't care about reliability" said no one, ever. In reality, the definition of reliability and its relative importance can vary. Asking yourself what level of reliability you, or perhaps more importantly your customers, require is a key first step in determining your storage needs.

When the system crashes or power is lost, data not yet committed to the media is usually lost as well.

Consider the importance of these three key aspects of storage system reliability:

1. **Does it power on?** This is hardware reliability – mean time between failures (MTBF) that exceeds expected lifetime under typical use conditions.

2. **Does it boot?** This is system integrity – devices which feature high system integrity are designed to start up after an unexpected power loss or system crash. File data consists of user data as well as metadata. Metadata refers to data that provides information about the user data, as well as details like the location of the data on the media. A file system with high system integrity maintains the metadata in an always functional state, while a file system check will clean up any fragments of data. Some of these checks run in the background – for example fsck() on ext4.

3. **Is the data I expect to be there intact?** This is data integrity – what was written by an application is what is read back from the media.

Understanding what is or isn't committed is therefore crucial to an application's ability to ensure data integrity. This challenge can exist for both the file system and the media – both can have uncommitted data in cache or buffers. Controlling how much and what data is "at risk" of being lost is the goal in order to achieve the expectation of data integrity.

Control over data-at-risk is managed with mount settings, flush and fsync commands, and the design of the software

and hardware. In many designs, the techniques required to achieve this control can directly work against the raw performance of the solution.

HOW FILE SYSTEMS CAN IMPROVE SYSTEM INTEGRITY

Not all file systems are created equal. One of the most used file system formats in embedded systems is FAT (File Allocation Table), which originated in the desktop environment and has been around since the 1970s. It's widely adopted largely due to its simplicity and interoperability. While commonly in use, it's a less than perfect solution for embedded devices. This is largely because FAT was originally developed for the desktop environment, where hazards such as power disruptions are a rare occurrence. This can lead to problems with embedded devices, where these power disruptions are common. For example, interrupting a write in the middle could cause a catastrophic corruption – in the worst case causing the whole partition to be lost and irretrievable, and potentially preventing the system from booting back up.

However, interoperability is one key consideration in modern, connected embedded devices. When interoperability is required from the system, using industry standard file systems – such as FAT, ExFAT or NTFS – is needed. While the original implementation of the file system is prone to system reliability issues under power loss, these issues can be mitigated with an optimized file system implementation. GravityCS by Tuxera offers a comprehensive portfolio of industry-standard file systems – including APFS, ExFAT, FAT, HFS+ and NTFS – with additional methods for protecting system integrity. One of the methods GravityCS uses is patented additional memory structures and prioritized write operations. While this keeps the on-media structure the same, the order of operations can improve the chances of recovery and reduce the impact to the meta-data, thus protecting system integrity.

Journaling

Beyond industry-standards file systems, other types of file systems use different methods for system integrity – the most common being journaling. File systems that use this technique track metadata changes in an additional reserved location called the journal. When recovering from an unexpected interruption, a journaling file system walks the structures on the media to decide which files on the media are valid and which are not. Then the fsck() tool is able to recover lost space and correct any other errors. Many Linux file systems use journaling, including ext4, Btrfs and F2FS. The downside of journaling is the additional overhead that's caused by journal replay during mount time, which can result in inconsistent mount times when booting up after power is lost. Journaling can also be used to improve data integrity. However, using journaling to provide better data integrity has significant overhead, as all the changes in data are tracked in the journal and this usually leads to significantly reduced overall performance.

Transactional file systems

Transactional file systems are developed with embedded systems in mind. Transactional file systems, such as Tuxera's Reliance family of file systems, can mitigate performance issues caused by journaling and provide both system integrity and data integrity. One of the ways this is achieved is by not overwriting data on the media, thus preserving a "known good state". After unexpected power loss or system failure, this type of file system mounts quickly – it merely must determine the proper media state by reading the metaroots. No file system check or other cleanup is required.

On the file system level, data-at-risk can be managed with mount settings, flush and fsync commands, and the design of the software and hardware. In many designs, this control can be directly opposed to the raw performance of the solution. For this reason, the Reliance family of Tuxera file systems provide runtime access to this control through an API and provided system library.

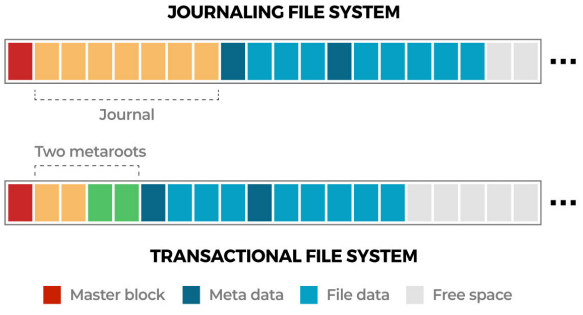

Fig. 1: On-disk layout in journaling and transactional file systems

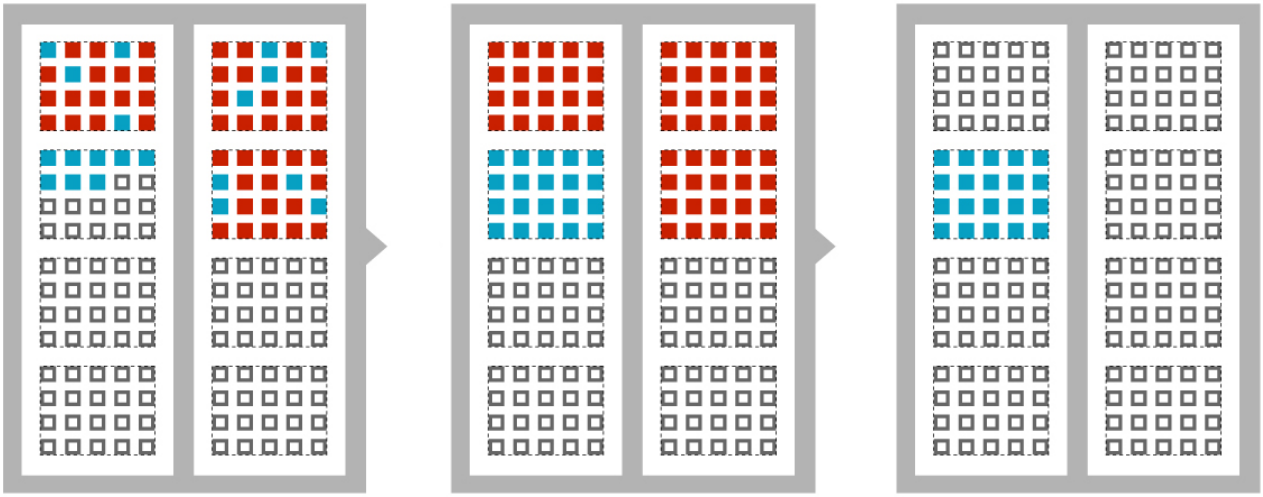

Fig. 2: Illustration of write amplification caused by garbage collection flash memory.

SOFTWARE – ONLY AS RELIABLE AS THE HARDWARE IT'S RUNNING ON

Software can go a long way toward providing system integrity, but software is only as reliable as the hardware it operates on. Flash memory provides some unique challenges with regard to the customary expectations of Moore's Law – the hardware is not always improving in every respect. Raw flash performance has been reduced as stronger error detection and correction (EDC) has become necessary, and flash endurance has steadily decreased as lithographies have been shrinking. What's more, write amplification has compounded the challenge of preserving flash lifetime.

Let's look at some of these hardware-related factors in more detail.

Power disruption issues

Flash devices are optimized for maximum data storage, and by design a certain low level of bit errors are expected. All NAND flash chips suffer from vulnerabilities when erase or program operations are interrupted, the most frequent cause of which is power disruptions. Interrupting the programming of a NAND page by power loss can result in problems that are hard to detect or work around. A partially programmed NAND cell may not consistently read back the same value, or may have poor data retention. In the unfortunate case of corruption occurring where file system metadata is being stored, the system might even become inoperable.

Memory wear out issues

As more bits are stored in one cell, the lifetime of the flash device diminishes. With embedded devices generating more data, this can lead to issues. All flash memory must be erased before it is written – modern flash can never be overwritten. While it can be written in relatively small pages, typically from 512 bytes to 16 KB in size, it must be erased in larger chunks, referred to as erase blocks. Most eMMC and UFS media use a massively parallel array of NAND chips, and have an effective erase block size of 2MB to 16MB.

Flash lifespan is measured in write/erase cycles. However, it's important to note that it's really the erase portion which counts against flash life. For example, you could write absolutely nothing to the media, and erase the same block repeatedly until it was officially worn out and would no longer process an erase request. Regardless of whether there was a large or small amount of data written to that block, it was the number of erases that actually mattered.

Write amplification

Exacerbating the already tenuous situation of the ever-decreasing media lifespan is the issue of write amplification, whereby writing a relatively small amount of data to the solid-state media may use up a disproportionate amount of the media's endurance.

Write amplification can be caused by several layers, and it is use case specific. File system write amplification

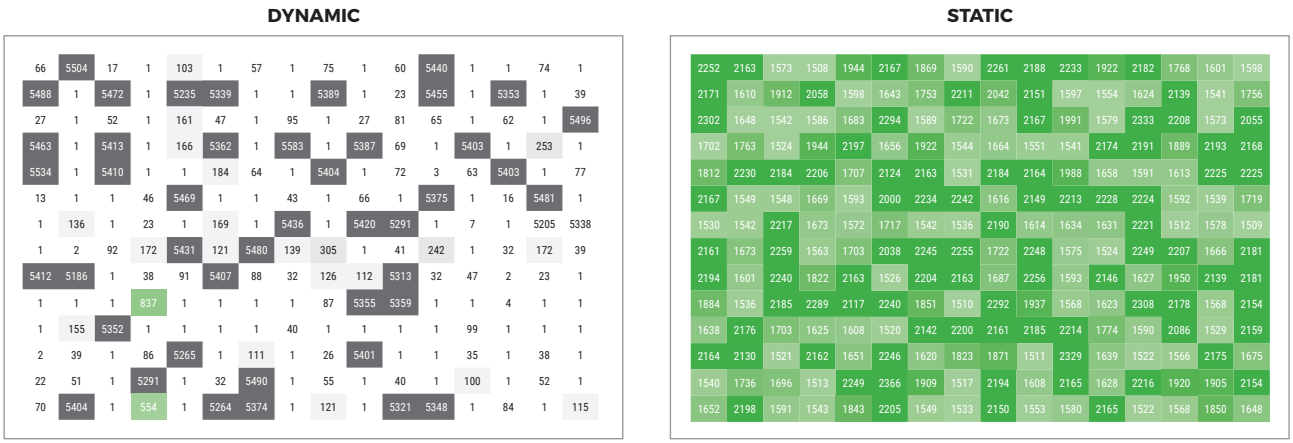

Fig. 3: Difference between dynamic and static wear-leveling.

is the ratio between what the application requested the file system to write and the total amount the file system requests the underlying block device to write. When flash wear-out is a significant concern for overall reliability, a file system can help keep the file system write amplification low with small metadata overhead and low fragmentation.

Device write amplification is measured as the ratio between what the file system explicitly requests the block device stack to write and the actual amount which is written on to the physical media. Device write amplification is caused by garbage collection and wear-leveling.

Wear-leveling

One of the factors causing device write amplification is wear-leveling. This is due to the fact that in order to get the maximum lifetime from the device, the device must be used equally. Wear-leveling is a process to ensure that an entire flash memory device – or an array of devices – is used in a uniform fashion in order to extend the overall lifetime of the flash. However, this can also cause write amplification as static data is moved around the device multiple times.

As the file system and flash device together contribute to the total write amplification of the device, it's important to test how these two components work together.

Data retention

Data retention is the time a device can hold data without refreshing. Embedded devices are used in environments where maintenance and replacement can be difficult, such as in space, or soldered inside a car system. This means that devices need to have a long lifetime, and that they have to work reliably. In order to do that, the data stored on the device must be kept safe. While the device ages, its capability to retain that data decreases drastically. With fresh flash, a device can usually hold the data for many years. However, when these devices are nearing end of life, the likelihood of data corruption increases significantly.

TESTING FOR SYSTEM INTEGRITY

To prevent field failures and to ensure system integrity, extensive testing is needed. As we've discussed above, the file system and flash device are both key components for system integrity. What's more, testing those components and ensuring that they work well together is crucial for the reliability of the system.

As flash technology is moving forward with incredible speed, it brings performance and reliability challenges to device manufacturers. These professionals struggle to find the time, resources, and expertise to adequately validate vendor claims – and to make wise product choices that will truly satisfy their requirements. This shortcoming in the product selection process can result in choosing a more expensive part than needed "just in case", or ending up with something that seemed ok with limited testing but ends up failing in the field – leading to product recall expenses or other bad outcomes.

As we have worked with suppliers and device OEMs in industries such as automotive and aerospace that have demanding data storage needs, we have seen

the consequences of sub-optimal hardware selection firsthand. With our expertise we have helped our customers in bringing next-generation products quickly to market, assisting them through every step of the way.

Tuxera Flash Testing Service provides comprehensive and comparative testing, analysis, and reporting of the impact of complex use cases on the lifetime and performance of a customer specified selection of flash devices. Our Flash Testing Service provides customers with independent guidance in the selection of storage technologies that will meet the requirements for cost-effective and strong performing products. This guidance helps OEMs calculate real-world usage of flash storage, potentially saving in bill of materials costs while lowering testing overhead for the customer. Complex system level test cases provide insights beyond standardized testing on the behavior of flash devices in customer use cases, and identify possible failure points early in the development phase.

CONCLUSIONS

Reliability remains one of the most important factors that distinguish leading embedded products from their also-ran competitors. Anticipating and preventing field failures enable market leaders to invest in innovation rather than costly resource-draining diagnoses, repair, and redesign. Today, data integrity is a strict requirement in some industries but not in others. However, as embedded devices are entrusted with more data all the time, the need for failproof data storage is emerging as a more relevant need across the board. The correct testing can enable you to deeply understand the reliability of your devices – with precision unique to your use case. Don't entrust the success of your next innovative product to a storage subsystem that is inadequately designed and tested.

Let's talk about optimizing your flash memory reliability and lifetime.
Get in touch with us at sales@tuxera.com.

ABOUT TUXERA

Tuxera is the leading provider of quality-assured embedded storage management software and networking technologies. Helping people and businesses store and do more with their data, our software is at the core of phones, tablets, cars, TV sets, cameras, drones, external storage, routers, spacecraft, IoT devices, and more. We help you store your data reliably, while making file transfers fast and content easily accessible. Tuxera is also an active member of multiple industry organizations, including JEDEC, AGL, SD Association, The Linux Foundation, and many others. Founded in 2008, Tuxera's headquarters are located in Finland, with regional offices in China, Germany, South Korea, Japan, Taiwan, and the US.

Tuxera Inc. Global HQ
Westendintie 1
02160 Espoo
Finland
www.tuxera.com