Transitioning from MLC to TLC flash memory on Raspberry Pi Compute Modules

Colophon

Release	2
Build date	10/07/2026
Build version	9b91b9d9cdcb

Legal disclaimer notice

TECHNICAL AND RELIABILITY DATA FOR RASPBERRY PI PRODUCTS (INCLUDING DATASHEETS) AS MODIFIED FROM TIME TO TIME ("RESOURCES") ARE PROVIDED BY RASPBERRY PI LTD ("RPL") "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW IN NO EVENT SHALL RPL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THE RESOURCES, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

RPL reserves the right to make any enhancements, improvements, corrections or any other modifications to the RESOURCES or any products described in them at any time and without further notice.

The RESOURCES are intended for skilled users with suitable levels of design knowledge. Users are solely responsible for their selection and use of the RESOURCES and any application of the products described in them. User agrees to indemnify and hold RPL harmless against all liabilities, costs, damages or other losses arising out of their use of the RESOURCES.

RPL grants users permission to use the RESOURCES solely in conjunction with the Raspberry Pi products. All other use of the RESOURCES is prohibited. No licence is granted to any other RPL or other third party intellectual property right.

HIGH RISK ACTIVITIES. Raspberry Pi products are not designed, manufactured or intended for use in hazardous environments requiring fail safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, weapons systems or safety-critical applications (including life support systems and other medical devices), in which the failure of the products could lead directly to death, personal injury or severe physical or environmental damage ("High Risk Activities"). RPL specifically disclaims any express or implied warranty of fitness for High Risk Activities and accepts no liability for use or inclusions of Raspberry Pi products in High Risk Activities.

Raspberry Pi products are provided subject to RPL's Standard Terms. RPL's provision of the RESOURCES does not expand or otherwise modify RPL's Standard Terms including but not limited to the disclaimers and warranties expressed in them.

Document version history

Release	Date	Description
1	08 Jun 2026	Initial release
2	6 Jul 2026	Major rework

Scope of document

This document applies to the following Raspberry Pi products:

Compute Modules

CM0	CM1	CM3	CM3+	CM4	CM4S	CM5
			✓	✓	✓	✓

Introduction

Flash memory (or eMMC) forms the backbone of modern data storage. On Raspberry Pi Compute Modules (not including Lite models), eMMC flash memory is used for program and data storage, as opposed to the SD cards used by our single-board computer range (up to and including Raspberry Pi 5).

At its core, flash memory stores data inside microscopic components called memory cells. The structural differences in how these cells manage electrical charges dictate the device's sustained throughput under load, I/O performance, reliability, and manufacturing cost.

The two most prominent types of flash memory currently used in computing are **multi-level cell (MLC)** and **triple-level cell (TLC)**. While their names sound similar, their internal architectures and operational characteristics differ significantly.

Important

The flash memory industry is transitioning away from MLC and towards TLC as the standard technology. This is a broad market shift driven by the advantages of manufacturing TLC flash memory. Consequently, MLC flash is quickly reaching obsolescence across the majority of manufacturers, including those used by Raspberry Pi. In response, Raspberry Pi has been transitioning to TLC flash memory and has recently qualified new TLC parts for use across the Raspberry Pi Compute Module range.

Comparing MLC and TLC flash memory

The fundamental difference between MLC and TLC lies entirely in data density. Data density refers to the number of bits stored inside a single physical memory cell.

Like all NAND flash memory, both MLC and TLC use error-correction hardware to maintain data integrity throughout their operational lifespan.

The flash translation layer

The flash translation layer (FTL) is a vital firmware component that runs on the internal controller of any managed flash device, such as eMMC. It acts as a translator between the operating system and the storage device's physical NAND flash memory chips, emulating a traditional hard disk drive so that operating systems can read and write data using linear Logical Block Addresses (LBAs), even though flash memory has a fundamentally different physical organisation. Throughout this document, wherever flash firmware is mentioned, it is the firmware running in the FTL.

Important

FTL firmware is built into the flash device itself — the end user has no access to it and cannot affect its actions in any way. It is a 'black box' from the end user's perspective.

Multi-level cell

- **Bits per cell:** Stores exactly **two bits** of data per memory cell
- **Voltage states:** Requires **four distinct voltage levels** (2^2) to represent the binary combinations (00, 01, 10, 11)
- **Performance:** Offers faster write speeds, as the memory controller only needs to target and verify four voltage zones
- **Durability:** Typically endures **3000–10,000 program/erase (P/E) cycles**[1]

Triple-level cell

- **Bits per cell:** Stores **three bits** of data per memory cell
- **Voltage states:** Requires **eight distinct voltage levels** (2^3) to represent combinations from 000 to 111
- **Performance:** Write speeds are managed in phases to achieve precise voltage placement across eight states; SLC caching (described below) largely compensates for this in practice
- **Durability:** Typically endures **1000–3000 P/E cycles**[1]

The narrower decoding thresholds for TLC cause the raw bit error rate (BER) to be higher than that of MLC. BER is also affected by temperature, and this effect increases as the flash ages.

[1] All raw P/E figures quoted are vendor specific.

Cost versus longevity

TLC provides 50% more storage capacity than MLC using the same number of physical transistors. This higher density makes TLC significantly cheaper to manufacture per gigabyte. However, TLC's higher storage density comes with narrower voltage margins between states, which is why modern TLC devices pair this architecture with advanced firmware — including LDPC error correction and wear levelling — to achieve reliability comparable to MLC during real-world use.

NAND cell degradation (in both TLC and MLC) is a function of the number of erase operations on a block. The effects of degradation include a shift in the programmed threshold voltages and an increase in floating gate leakage. This is why the firmware mitigations described below are integral to every modern TLC device.

Temperature

All TLC flash devices used by Raspberry Pi have equivalent or better temperature ratings than the previous MLC parts.

Erase cycle degradation

Erase cycle degradation refers to the physical and electrical wear that flash memory cells experience during program/erase cycles. It primarily stems from the high voltages required to move electrons across a thin insulating oxide layer, which physically damages the material over time.

Both MLC and TLC memory suffer from this degradation. Over time, this wear increases the raw bit error rate and P/E times — effects that modern flash controllers are specifically designed to manage. To mitigate these physical limits, modern devices rely on flash controllers equipped with sophisticated algorithms to ensure reliable performance across the rated endurance of the storage media.

The following sections describe these mitigations in more detail, focusing on their use for TLC memory.

Software and firmware adaptations

To manage wear levelling and device longevity, the firmware implements vendor-proprietary algorithms that run directly on the FTL inside the flash device.

> **Important**
>
> These adaptations are implemented entirely by the flash device. There is no impact on the compute module boot firmware or the host operating system.

Advanced error correction codes

With eight voltage states, TLC requires more sophisticated error correction codes (ECCs) than the Bose–Chaudhuri–Hocquenghem (BCH) codes used by MLC. Modern TLC devices address this by using Low-Density Parity-Check (LDPC) codes, which provide robust error correction well beyond what the raw cell characteristics would suggest.

In summary, LDPC codes provide superior robustness in error correction but need enhanced hardware, while BCH codes offer lower complexity for short-to-medium block lengths. In practice, the additional processing overhead of LDPC is negligible for typical workloads, making read performance between TLC and MLC broadly comparable — this is confirmed by our test results.

Dynamic single-level cell caching

TLC drives use dynamic single-level cell (SLC) caching to deliver fast burst write speeds regardless of native cell write speed.

The controller treats a portion of the TLC space as single-level cell memory, writing only one bit per cell instead of three. This enables ultra-fast data transfer during short bursts. When the drive is idle, the controller processes this data in the background, moving it from the fast SLC cache into the denser TLC blocks.

This mechanism is transparent to the host, but it is not without risk, and the effects are more pronounced under the kind of sustained, write-heavy loads typical of industrial applications:

- **Cache exhaustion:** The SLC cache is a limited-size pool, with its size determined by the vendor and their controller architecture. The cache fills up under sustained write loads that exceed the controller's ability to flush the cache in the background. Once full, further writes must go directly to TLC at its native (slower) speed, so throughput can drop sharply and without warning for as long as the write load continues.
- **Migration integrity:** If power is lost while data is being moved from the SLC cache into TLC, there is a small risk to the integrity of the data. This risk is further mitigated on systems using a journalling file system, which guards against exactly this kind of interrupted write.

Wear levelling

Without load distribution, repeated erases on the same physical blocks would exhaust those cells prematurely. In all flash memory types, highly sophisticated **dynamic and static wear-levelling algorithms** are integrated into the FTL firmware.

The firmware tracks the program/erase cycle count of each block on the drive. Dynamic wear levelling rotates newly written data among the empty blocks, while static wear levelling moves rarely modified ('cold') data to ensure all of the blocks wear equally. A combination of the two ensures that the entire device wears down at a uniform rate.

The wear-levelling algorithms in the FTL of all flash devices are highly complex and can remap blocks invisibly to the end user, who has no influence over this process. For example, an algorithm may detect that one area of the device is less healthy than another and that a particular file has been written only a small number of times; it may then move the less healthy blocks so that they serve as the backing store for the less frequently written file.

Increased over-provisioning management

As cells reach the end of their rated endurance, the drive uses spare capacity to seamlessly replace them. To achieve this, the flash firmware reserves a larger percentage of the drive's total raw capacity as hidden space, a process known as **over-provisioning**.

While an MLC drive might only reserve 7% of its flash blocks for maintenance, a TLC drive often requires 10–15% or more. The firmware uses this space for garbage collection, temporary caching, and replacing worn-out blocks, which prevents drive failure. Most flash vendors have standardised on 91% user-area availability. Any increase in the spare blocks required by TLC is at the expense of free sectors.

Pseudo-SLC (pSLC)

The dynamic SLC caching described above is a temporary, controller-managed buffer: data passes through on its way to TLC, and the capacity it occupies is not fixed or visible to the host. This is distinct from **pseudo-SLC**, commonly referred to as **pSLC**, in which a region of the device is permanently reconfigured.

Pseudo-SLC is typically implemented as an **enhanced user area** (also referred to as enhanced storage). The JEDEC eMMC standard defines this feature, its attributes, and how it is configured by a host, but leaves the physical implementation up to the vendor.

In practice, pSLC has become the commonly used term for the enhanced user area, and the two terms are often used interchangeably. However, exactly how a given part implements an enhanced user area is specific to each vendor, and it is not always straightforward to determine which implementation a particular part uses from its datasheet alone.

Reliability is not guaranteed

Configuring a region as pSLC does not, by itself, guarantee improved reliability. Some vendors write pSLC cells using true single-level voltage states, which meaningfully increases endurance and data retention in that region. Other vendors may implement enhanced user areas in a way that provides little or no endurance benefit over standard TLC operation.

Warning

The behaviour of a flash part configured for pSLC is defined by the vendor, not the flash standard. Raspberry Pi Ltd is responsible for helping customers assess whether enabling pSLC on their Raspberry Pi Compute Module is appropriate for their application, and will endeavour to advise on this where possible. However, Raspberry Pi Ltd may be bound by non-disclosure agreements with flash vendors that limit what can be published about a specific part's implementation.

The capacity trade-off

Because each cell in a pSLC region stores one bit instead of three, enabling pSLC reduces the usable capacity of the area it is applied to, typically to a third or less of its original size. This is a direct trade-off between usable capacity and endurance: a customer with a write-heavy workload may choose to sacrifice capacity for a device that is more likely to tolerate that workload over its service life, while a customer with a lighter, more read-heavy workload may prefer to retain the full TLC capacity. There is no single correct choice; it depends entirely on the target application and its write profile.

An irreversible one-time operation

Regardless of the vendor's implementation, enabling pSLC (or enhanced user areas more generally) is treated by `mmc-utils` as a **one-time operation**.

Warning

Enabling pSLC erases all of the data in the affected area and cannot be undone through `mmc-utils`. Some vendors may provide their own tools for reversing the configuration, but this cannot be assumed.

Enabling pSLC should be treated as an irreversible decision made during provisioning, not something to be changed during the operational lifetime of a device. As such, pSLC should only be enabled before an operating system is installed. See rpi-sb-provisioner for more details about the provisioning flow of Raspberry Pi devices.

Summary

Whether to run a compute module's flash memory with or without pSLC enabled is a workload-dependent decision:

- **With:** Reduced usable capacity in the configured area, with endurance and retention that depend on the specific flash vendor's implementation, which may or may not be a meaningful improvement over TLC
- **Without (i.e. TLC):** Full rated capacity, with endurance and retention as described elsewhere in this document

Note

In many cases, implementing an oversized TLC device may be a better option than using the same device in pSLC mode.

Warning

Customers with reliability-critical or write-heavy applications who wish to use pSLC are encouraged to contact the Raspberry Pi applications team via applications@raspberrypi.com to discuss pSLC support, including its guarantees for the specific flash part(s) on their module, before committing to it at scale.

Architectural comparison

Table 1.
Architectural comparison

Feature	MLC flash	TLC flash
Bits per cell	2	3
Voltage states	4	8
Manufacturing cost	High	Low
Data density	Moderate	High
Native lifespan	3000–10,000 P/E cycles	1000–3000 P/E cycles
Primary ECC used	BCH or basic LDPC	Advanced LDPC (soft-decision)

Consequences for Raspberry Pi Compute Module users

Raspberry Pi does not expect any noticeable impact for the majority of Compute Module users.

Important

The move to TLC requires **no** changes to software, firmware, or programming tools. While there are differences in endurance and speed, these are almost entirely mitigated by the techniques described above.

Performance

The read performance of TLC and MLC flash memory is broadly the same. Write speeds are impacted somewhat, but this is often mitigated by the operating system's use of caching, as well as the firmware on the flash devices themselves. For large writes that exceed cache capacity, write performance will be impacted.

If extreme reliability is required, pSLC mode can be enabled on all flash variants.

Reliability

Although TLC cells have a lower raw P/E cycle count than MLC, the combination of over-provisioning, wear levelling, and SLC caching means that the effective device lifetime is comparable for the vast majority of compute module workloads.

Tip

For write-intensive applications, choosing a higher-capacity variant provides proportionally more endurance, as this presents more wear-levelling opportunity. For example, doubling the size of the flash memory on a Raspberry Pi Compute Module will double its flash lifetime, assuming the write workload is constant.

User-level mitigations

There are a number of mitigations that can be made to improve flash performance and longevity, which are relevant to all flash types:

- Reduce writing to flash
  - Reduce log writes
  - Avoid lots of small individual writes
- Use RAM file systems for temporary data
- Choose appropriate file systems (e.g. OverlayFS)
- Use off-device file systems (e.g. local network, cloud storage)

More information can be found in the *Making a More Resilient File System)* white paper.

Conclusion

This white paper has described the differences between MLC and TLC flash memory. While TLC has lower cell-level raw endurance than MLC, its increased capacity and dedicated firmware ensure it is equally reliable when considered at the operating system level.

MLC flash availability is dwindling worldwide, with all major manufacturers cutting back on MLC production. As a consequence, Raspberry Pi is having to transition to TLC flash faster than previously expected, and has recently qualified multiple TLC parts for use on Raspberry Pi Compute Modules. We are confident that these provide comparable performance and endurance.

Appendix A. JEDEC standards

JESD218 and JESD219A are industry standards from the JEDEC Solid State Technology Association that define rules (JESD218) and workloads (JESD219A) for testing the endurance rating and longevity of solid-state drives (SSDs) across different application classes. A copy of these standards can be found on the JEDEC website. Registration is required.

We are currently talking to all of our flash suppliers to gather data relevant to these standards. This document will be updated with endurance data once Raspberry Pi Ltd has received it and been given permission to publish by the suppliers.

Appendix B. References

There are many articles comparing MLC to TLC and pSLC available online. Some useful ones can be found below:

- *4 reasons to use industrial 3D TLC NAND flash as pseudo-SLC* by Swissbit
- *Ultimate guide to enterprise SSDs* by Micron
- *MLC vs. TLC: Which is the Better SSD?* by Everpure
- *Bye Bye MLC: QLC and TLC to be dominant SSD technologies amidst AI's ravenous appetite for NAND — and PLC won't probably happen till the era of Petabyte SSDs* by Techradar Pro

Appendix C. Testing regime

Raspberry Pi has test procedures that all new parts must pass in order to qualify for use on Raspberry Pi devices.

C.1. Reliability test

C.1.1. Extended storage stress

This is an automated test run by scripts on the device under test (DUT). It executes the following sequence in a loop:

- Precondition the DUT by padding the root file system to a maximum of 90% capacity or a minimum of 1GB free space using random data
- Announce boot to a central results-gathering server
- Run a storage benchmark in a loop that includes sequential and random read and write tests
- Upload results from completed runs to the server
- Randomly cut power input after 2–5 minutes of runtime

Raspberry Pi OS has been modified to:

- Disable watchdog reboot
- Run `fstrim` daily (via systemd unit)

The test cycle identifies issues related to:

- Long-term performance
- Wear-out behaviour (bad block replacement or lifetime exhaustion)
- Resilience to power loss events

The intensive nature of the test is designed to compress years of typical usage into a week or less. To pass, the DUT must successfully complete at least 1000 power cycles per device — and 10,000 cycles per set of devices — with zero boot failures.

If a device is unable to boot after a power failure, it fails the test. This criterion encapsulates the vast majority of faults caused by failures at the FTL level.

The testing platform captures estimated wear-out/lifetime data for the DUT. Program/erase cycles vary depending on device capacity and test length, but at least 30% of the expected P/E lifetime will be used. Health warning flags are also recorded during tests.

Contact us for more information

Please contact applications@raspberrypi.com if you have any queries about this whitepaper.

Web: www.raspberrypi.com

Raspberry Pi

Raspberry Pi Ltd