SOFTWARE-ENABLED FLASH™

Software-Enabled Flash™ Technology:

New Ways to Minimize Write Amplification

TECHNICAL BRIEF

High performance storage has evolved from hard drives to NVMe™ SSDs, delivering significant benefits in terms of speed, reliability and ease of use. With this evolution come new challenges to obtain the best possible lifetime and performance from flash-based SSDs. One of the biggest challenges to impact SSD lifetime and performance is the write amplification factor (WAF). This tech brief describes WAF, why developers need to minimize it, and ways that new Software-Enabled Flash (SEF) technology provides developers with the tools to achieve important WAF reductions.

What is WAF?

The WAF of a storage subsystem is the ratio of the total amount of data written to flash by the drive, divided by the total amount of data written by an application (see Figure 1).

$$\text{WAF} = \frac{\text{Application Data + Additional Drive Written Data}}{\text{Application Data}}$$

Figure 1. WAF formula

The important and non-obvious part of this equation is the 'Additional Drive Written Data' component. Since flash memory has unique structure and erase requirements, SSDs often have to rewrite application data multiple times over its lifetime. These rewrites represent required overhead for on-drive data management. The best SSD outcome is when the WAF ratio is 1.0. In practice, WAF can vary from nearly 1.0 up to 4.0 or higher. For example, when WAF is 4.0, every gigabyte[1] of the application data writes require four gigabytes of flash writes.

Root Causes of WAF

There are multiple causes of a high WAF. This could include an application repeatedly rewriting a logical block address (LBA) or the drive itself performing flash memory patrol operations. However, in many cases, garbage collection (GC) is often the most significant. The more work that the GC process has to perform, the more write overhead that is required, resulting in a higher WAF and less time that the SSD can dedicate to application Input/Output (I/O).

In SSDs, drive mapping transitions from physical flash blocks to LBAs via a flash translation layer (FTL). Instead of the old data being overwritten in an LBA, a new physical block is written to flash memory. The FTL updates to point to this new physical block, and the old block is marked invalid and ready for reclaim. The same process happens when a block is 'trimmed' (erased) by the application or operating system except that there is new data to write, only an update to the FTL to mark a block invalid. Most importantly, there is no erasure of individual LBAs. Over time, the total physical flash blocks fill up with many invalid blocks alongside valid data. When this happens, the GC process starts to reclaim the invalid space to free it for reuse with new data.

Garbage collection reads the valid data out of multiple erase blocks, skips over invalid LBAs, and writes the valid data to a new erase block without including any invalid data. The SSD then updates its FTL mapping so it can point to this new copy of valid data during the GC process. With valid data collected from partially valid erase blocks in this manner, the erase blocks can then be cleared and made available for new writes (Figure 2). This reading and rewriting of old on-drive data increases the system WAF.

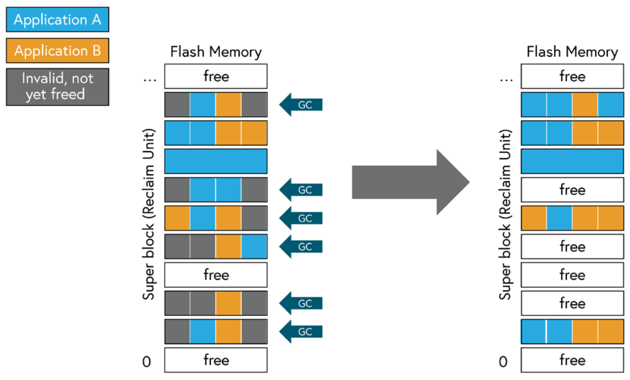

Figure 2. The GC process with data layout before and after completion
(Used with permission from KIOXIA America, Inc.)

Benefits Associated with Minimizing WAF

There are several critical benefits in the data center with minimizing WAF, and includes:

- *More usable flash write lifetime since fewer of the underlying flash memory's limited write cycles will be used for GC*
- *Lower I/O latency unpredictability because background GC will operate less frequently and for shorter periods*
- *Higher average throughput as less flash bandwidth will need to be dedicated to overhead operations*
- *Lower effective power consumption because of the overhead reductions in read and write operations*

Early Industry Attempts to Minimize WAF

WAF can have significant impact on flash memory at data center scale so the industry has made many attempts to minimize it through application changes and modifying SSDs.

Applications can help minimize WAF by writing data in a single sequential stream and in large chunks. This maps well to use cases such as log-structured databases and other logs, but in general, an efficient mapping is difficult to achieve. Writing sequential sets of LBAs can allow some drives to place data in a way that requires almost no GC operations.

Unfortunately, even when an application writes to an SSD in this manner, the drive often is unable to take advantage of it because of the I/O blender effect[2] (Figure 3). The application's sequential I/O accesses will aggregate with other applications' random I/O accesses, effectively obscuring the sequential map pattern from the drive.

To overcome the I/O blender effect, new SSD command sets are available, but with limited uptake so far. One of the first of these was Zoned Namespaces (ZNS), which exposes the underlying flash structure to developers, forcing a complete rewrite of their applications to eliminate all random writes. Due to this incompatibility and restriction of random writes, the ZNS interface has been limited to a few specific use cases and applications.

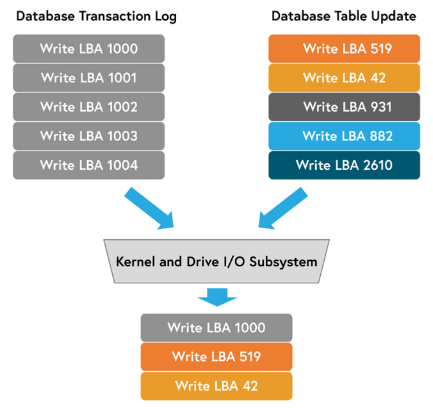

Figure 3. The I/O blender effect mixing sequential and random writes at the drive level

(Used with permission from KIOXIA America, Inc.)

Another attempt to ameliorate the I/O blender effect and reduce WAF was SmartFTL, which in turn led to a new NVM Express™ proposed SSD standard called Flexible Direct Placement (FDP). While there are many more components of SmartFTL and FDP, in relation to WAF their biggest contribution is the ability to control data placement on an individual I/O level. In the case above with a well-behaved logging application, these technologies could allow sequential application writes to remain separated from other random accesses.

Features for Minimizing WAF Using SEF

SEF technology is a new open source storage paradigm that delivers a more comprehensive solution for minimizing WAF and maximizing the value of flash memory. It removes the legacy block interface and provides developers with the power to access the full capabilities of flash memory. The technology utilizes a customized hardware platform called the SEF Unit. A host-based storage Application Programming Interface (API) and Software Development Kit (SDK) enables the SEF Unit to become a software-defined and controlled element of the application (Figure 4). SEF provides developers with powerful data placement, workload isolation and latency control capabilities, and is managed by the Software-Enabled Flash Project under the auspices of the Linux Foundation®.

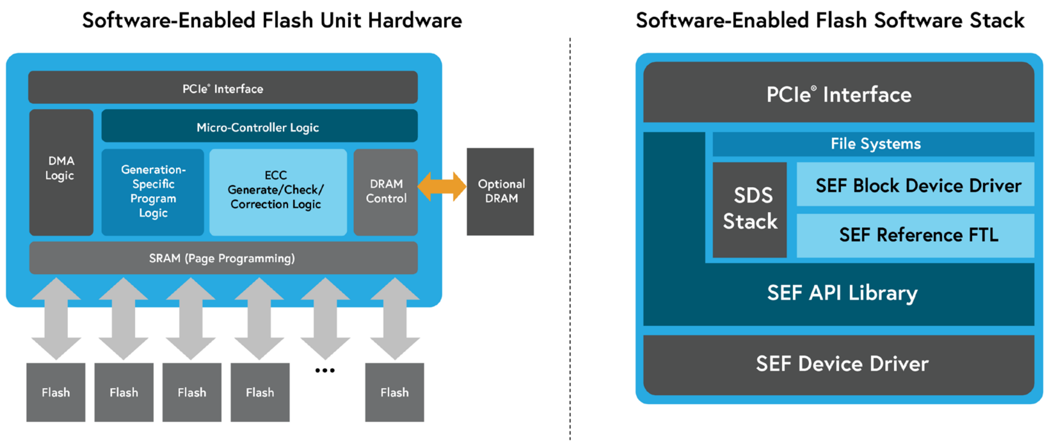

Figure 4. SEF Unit hardware and SEF software stack
(Used with permission from KIOXIA America, Inc.)

The new SEF storage model includes hardware-based isolation via Virtual Devices. Hardware-based isolation enables workloads to tie to a specific set of flash dies. This level of coupling can ensure that applications in different Virtual Devices are guaranteed to physically not interfere with each other.

Software-based isolation builds upon the hardware Virtual Devices and segments a Virtual Device for multiple workloads with a much finer grained control. While applications in the same Virtual Device but different QoS Domains can share the same flash die, their data will never comingle in a single flash super block. Placement IDs provide even finer grained control used to separate, at run time, different application data streams from each other in the same QoS Domain (Figure 5).

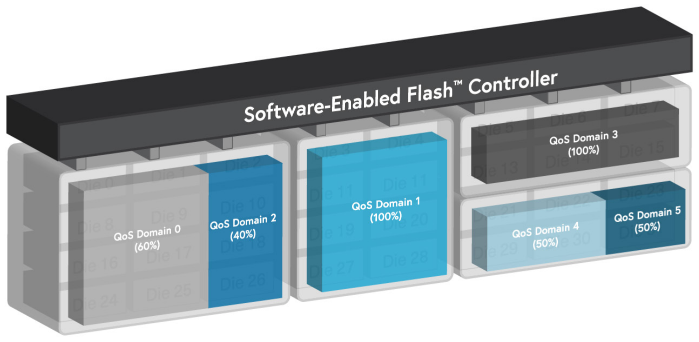

Figure 5: Example data layout using SEF Virtual Devices and QoS Domains
(Used with permission from KIOXIA America, Inc.)

This coarse hardware-based and fine-grained software-based isolation allows developers to control data placement and effectively minimize the WAF of their applications, while simultaneously insulating those applications from other noisy tenants on the system.

Minimizing WAF in a Multi-Tenant Server

Many servers in the data center today run multiple applications in parallel using virtualization or containerization technology. This allows for increased server utilization, but at a higher WAF:

1. The I/O blender effect from multiple tenants can cause super blocks to contain unrelated data.
2. A significant GC effort is required to free the flash memory used by completed tenants.

Both of these scenarios can result in a heavy GC workload, which in turn increases system WAF and wastes SSD performance on background management operations. SEF QoS Domains provide a solution to both of these problems by ensuring that workload data is never intermingled in a single super block.

Using SEF QoS Domains to isolate workloads eliminates the I/O blender effect. Once the management layers assign tenants to separate QoS Domains, their write operations separate at the flash super block level (Figure 6). From this separation, the WAF of a poorly behaved random tenant workload has no impact on a better-behaved sequential one. Additionally, since QoS Domains also support thin provisioning[3], workload isolation does not incur increased complexity for the virtualization or containerization layer.

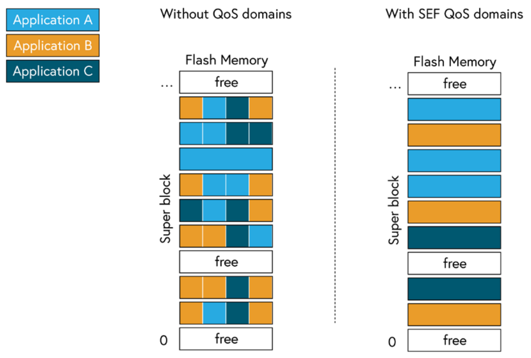

Figure 6. The impact that QoS Domains have on data isolation

(Used with permission from KIOXIA America, Inc.)

When workloads complete their lifecycle and terminate, 'instant reclaiming' of the space they used in flash memory is also an easy task with SEF and involves no garbage collection. Since all application data is in a separate set of flash super blocks, the SEF Unit simply needs to mark those super blocks as invalid (and ready for erasure). Because there is no rewriting of data, unlike in a legacy drive where super blocks may contain multiple tenants' data, there is no increase to WAF. Terminated tenants' flash memory reclaim is instantaneous without GC overhead (Figure 7).

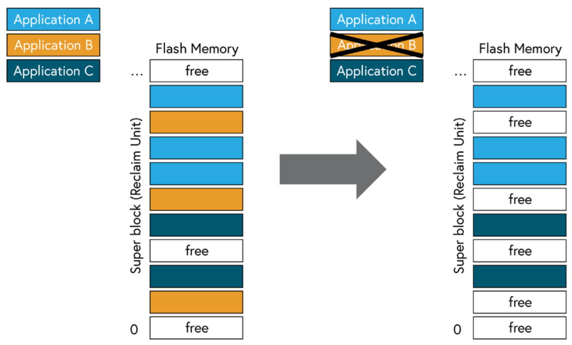

Figure 7. Example of instant reclaim of a terminated workload's flash memory usage

(Used with permission from KIOXIA America, Inc.)

Minimizing WAF with Mixed Sequential and Random Workloads

The I/O blender effect can occur even within a single application on an isolated system. A workload that has multiple access patterns can still result in a drive receiving a mixed, randomized workload. For example, a relational database could have a transaction log with sequential writes combined with random updates to table files as records are changed. A legacy drive will often end up mixing these two types of writes in single super blocks, resulting in a high GC workload, high WAF and lower overall application performance.

SEF provides a solution for these kinds of workloads using individual placement IDs per write type, all within a single QoS Domain and application. A placement ID is a hint to the drive that data with the same placement ID will have the same typical lifespan and access pattern, and should be placed together. The technology also enables applications to define the total number of placement IDs required in their QoS Domain, and then tag any I/O operation according to application logic. The drive will then isolate data to separate sets of super blocks, eliminating the I/O blender effect within a single application instance. Using database workloads as an example, there could be one placement ID for transaction logs and others for data table space updates.

SEF Can Minimize WAF for Modern Applications

Minimizing write amplification is a significant challenge for any storage system. Using the capabilities of SEF Virtual Devices and QoS Domains, developers have the power to control data placement in flash memory and minimize WAF. This can lead to significant benefits including longer effective device lifetime, lower drive overhead and better storage Quality of Service. SEF also features additional capabilities such as die time weighted fair queueing and on-drive copy offload extending the power that developers have over their flash memory.

For more information on the complete SEF technology stack and its benefits, visit https://softwareenabledflash.org.

FOOTNOTES:

[1] Definition of capacity - KIOXIA Corporation defines a kilobyte (KB) as 1,000 bytes, a megabyte (MB) as 1,000,000 bytes, a gigabyte (GB) as 1,000,000,000 bytes, a terabyte (TB) as 1,000,000,000,000 bytes, and a petabyte (PB) as 1,000,000,000,000,000 bytes. A computer operating system, however, reports storage capacity using powers of 2 for the definition of 1Gbit = 2^{30} bits = 1,073,741,824 bits, 1GB = 2^{30} bytes = 1,073,741,824 bytes, 1TB = 2^{40} bytes = 1,099,511,627,776 bytes, and 1PB = 2^{50} bytes = 1,125,899,906,842,624 bytes and therefore shows less storage capacity. Available storage capacity (including examples of various media files) will vary based on file size, formatting, settings, software and operating system, and/or pre-installed software applications, or media content. Actual formatted capacity may vary.

[2] The I/O blender effect occurs when multiple virtual machines (VMs) simultaneously send I/O to a storage subsystem causing the sequential I/O to become highly randomized, which in turn can increase latency in the storage system.

[3] Relating to storage virtualization systems, thin provisioning is a technique to better utilize the available physical storage space. It allows easy allocation of space to servers, on a just-enough and just-in-time basis.