SEF

SOFTWARE-ENABLED FLASH™

Software-Enabled Flash™ Technology:

Solutions for Powerful Workload Isolation

TECHNICAL BRIEF

Software-Enabled Flash technology is a new "Software-Defined" method for deploying and using the full potential of flash memory in data center and cloud-scale applications. It redefines the way that applications can access flash memory and exposes its inner parallelism without requiring developers to handle low-level flash operations. A key capability of Software-Enabled Flash technology allows developers to effectively isolate applications from each other, whether they are virtualized instances, containers or even bare metal programs. This brief examines the benefits of this powerful workload isolation capability.

Multi-Tenancy + Noisy Neighbors = Poor Application Performance

The cloud is an environment in which a single server can host numerous customer applications and data using virtual machines or containers to maximize the service provider's infrastructure investment. However, multi-tenant environments come with interference or "noisy neighbor" challenges[1]. If not managed carefully, a single tenant on a server could cause poor performance for other tenants running on that same server. For example, if a tenant starts a process that consumes massive amounts of Input/Output (I/O), such as a database table scan or a backup, the other tenants on that server could suffer delays in servicing their own I/O requests (Figure 1).

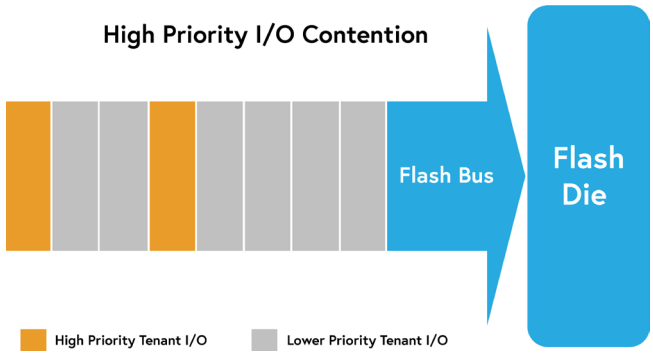

Figure 1: High priority tenant I/O contends with lower priority tenant I/O on the flash bus (Source: KIOXIA America, Inc.)

With Software-Enabled Flash technology, multiple workload isolation capabilities are provided using hardware and software techniques, and include:

- Avoiding cross-tenant interference with Virtual Devices
- Isolating and avoiding noisy neighbors with Quality of Service (QoS) Domains
- Minimizing head-of-queue blocking with Advanced Queueing and Die Time Scheduling

An open source software stack, including a low-level Application Programming Interface (API) (Figure 2) and high-level Software Development Kit (SDK) allows developers to make use of these capabilities in application and orchestration software. These software tools isolate virtual machines, containers or bare metal applications.

Avoiding Cross-Tenant Interference with Virtual Devices

A flash device is generally built with a flash controller, some Dynamic Random Access Memory (DRAM), and anywhere from one to thousands of flash memory dies. These chips are then connected to the flash controller via one or more channels at the individual device level – where many devices can be used. While the actual protocols used between the flash controller and the flash dies differ by vendor and technology, these protocols all generally transmit flash protocol requests significantly faster than the flash dies can execute them.

However, within an individual flash die, there is a different story. A flash die, in many cases, may only be capable of performing a single operation at a time, regardless of the size of memory it contains. For example, if a flash block is in the process of being erased, any other operation (such as a read, write or other erase) to that flash die, regardless of importance, must wait for the flash block to complete its erasure. This process can potentially

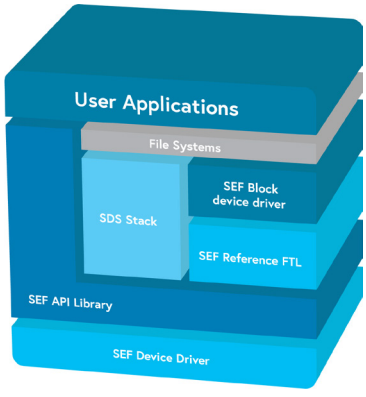

Figure 2: Software-Enabled Flash Technology API (Source: KIOXIA America, Inc.)

cause significant latency issues in a multi-tenant server, especially if those tenants share space on the flash die. A large write or read operation from one tenant could block other tenants from making progress with their respective operations, damaging their latency performance.

The Software-Enabled Flash Technology Solution:

Application developers can avoid the cross-tenant interference example above by using the Virtual Device feature of Software-Enabled Flash technology. The Virtual Device feature enables sets of flash dies to be assigned to specific workloads, effectively isolating them from other workloads and associated interferences (Figure 3). The Virtual Device flash die level isolation, configured by orchestration software when the device is deployed, provides the highest level of tenant and workload isolation.

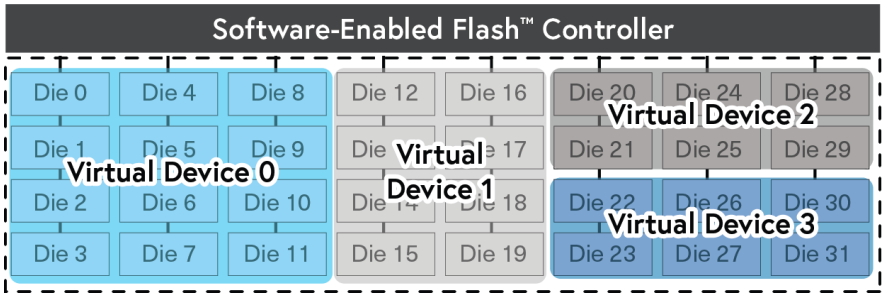

Figure 3: Software-Enabled Flash Virtual Devices block diagram
(Source: KIOXIA America, Inc.)

Isolating and Avoiding Noisy Neighbors with Quality of Service Domains

Virtual Devices are a powerful way of isolating workloads, but they are somewhat coarse grained. A finer grain of isolation control is available using Software-Enabled Flash Quality of Service Domains, which further subdivide Virtual Devices (Figure 4). These domains impose a secondary level of isolation between different workloads. While workloads in different Quality of Service Domains may utilize the same flash dies, their data is never intermingled within a flash super block - a group of flash blocks from flash dies written in parallel for performance reasons. Many Quality of Service Domains can be generated within a single Virtual Device simultaneously, allowing for many more simultaneous workloads than flash dies in the system.

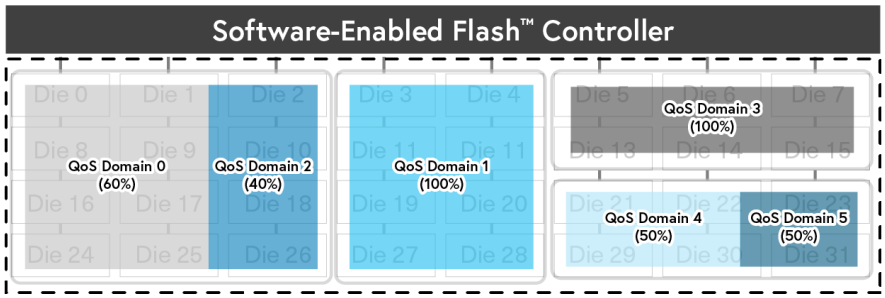

Figure 4: Software-Enabled Flash Quality of Service Domains block diagram
(Source: KIOXIA America, Inc.)

Orchestration software can create or destroy Quality of Service Domains on the fly. For example, when a virtual machine or container is initiated, the orchestration software can create a new Quality of Service Domain and allocate some portion of a Virtual Device to it. As multiple workloads perform multiple write operations, the data of each workload is separated from the data of other workloads at the superblock level. This capability delivers two key benefits - isolated garbage collection and an instant reclaim of the flash memory when a workload is terminated.

For flash drives without Software-Enabled Flash technology, a single super block can have data from many different workloads. This means "noisy" workloads could disrupt the data placement of "well-behaved" workloads, intermingling their data at the super block level. As a result, a well-behaved workload could suffer unexpected delays due to garbage collection caused by the noisy ones.

The Software-Enabled Flash Technology Solution:

Application developers can isolate and avoid the noisy workload impact by using the Quality of Service Domain feature of Software-Enabled Flash technology. The well-behaved workload data is never intermingled with noisy data in a flash super block. Only the noisy workload would suffer the impact of the garbage collection it required.

In a multi-tenant system, workloads come and go over time. When a workload completes, its data is normally completely invalidated. In typical flash drives, reclaiming this flash space would involve multiple read, write and erase operations because those invalid blocks would be spread to many different super blocks. However, with Software-Enabled Flash Quality of Service Domains, all super blocks can be immediately reclaimed without a single garbage collection read or write operation. Every super block containing data for that completed workload can be fully invalidated and erased instantly, and made immediately available for reuse.

Minimizing Head-of-Queue Blocking with Advanced Queueing and Die Time Scheduling

In addition to the physical flash-based isolation capabilities afforded by Virtual Devices and Quality of Service Domains, Software-Enabled Flash technology also includes an advanced queueing design that isolates workload requests as they are received by a flash drive. Some of this queueing is under software control via application-controllable I/O operation weights, and some is managed directly by the internal architecture of a Software-Enabled Flash device – also known as a SEF Unit.

Application or orchestration software can select among three different Software-Enabled Flash technology queueing models: (1) Priority Queueing; (2) Round Robin Queueing; or (3) Die Time Weighted Fair Queueing (Figure 5). Each is described below:

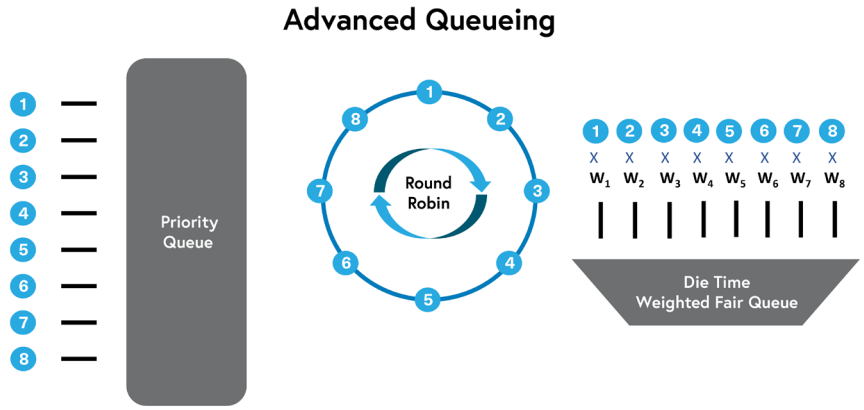

Figure 5: Software-Enabled Flash technology advanced queueing models (Source: KIOXIA America, Inc.)

Priority Queuing
Orders workloads strictly by importance. This enables higher priority workloads, such as a mission-critical database, to effectively monopolize the flash device. In this model, the device will only allow lower priority workloads access to the drive when it is not busy.

Round Robin Queueing
Allows all workloads to effectively time-slice access to the flash device. In this model, every workload gets access to the drive in turn.

Die Time Weighted Fair Queueing
Enables individual I/O weights to be finely-tuned, accounting for their on-die time. By adjusting these weights, different levels of latency QoS are possible, while avoiding I/O starvation of any particular workload.

Multiple parallel queues within the flash drive separate workload operations down to the flash die level. Once an operation is transmitted to the SEF Unit via standard NVMe® protocol submission queues, the read and write operations are split up and processed in separate, parallel paths. The software-defined weights and queueing models are applied independently, and in parallel. Finally, the separated read and write streams are assigned to per-flash-die queues.

The Software-Enabled Flash Technology Solution:

By separating read and write operations per flash die, Software-Enabled Flash technology enables SEF Units to minimize the impact of head-of-queue blocking – which occurs when a lower priority, long running operation, such as a flash write or erase, blocks other higher priority operations from being processed.

Workload Isolation Delivers QoS

Software-Enabled Flash technology was designed around the concept of isolating workloads to deliver predictable, programmable performance to multi-tenant servers. Flash die level isolation using Virtual Devices provides the highest level of isolation. Dynamic Quality of Service Domains provide a fine-grained way of isolating data and workloads from one another. Advanced queueing models enable tunable isolation and prioritization between I/O streams and workloads within the SEF Unit itself.

The Software-Enabled Flash Project

The Software-Enabled Flash Project is an open source, Linux Foundation® project dedicated to enabling collaboration on a new, purpose-built, media-centric flash hardware focused on the needs of the cloud and storage developers. For more information, visit https://softwareenabledflash.org.

Notes:
[1] The noisy neighbor effect occurs when an application or VM uses the majority of available resources that causes performance issues for others on the shared infrastructure.