OpenMPDK and uNVMe User Space Device Driver for Server and Data Center

Open source for maximally utilizing Samsung's state-of-art Storage Solution in shorter development time

White Paper

SAMSUNG

Samsung's uNVMe Device Driver

Target Audience

Data center Storage engineers, system architect, interface specialists, systems analysts, Storage software specialists and manager, product and technology planners, CTO

Abstract

OpenMPDK is Samsung's Open Memory Platform Development Kit which is open source to share the industry in freely. This OpenMPDK will allow key features of Samsung's state-of-the-art memory solutions to be better utilized at a system level. The OpenMPDK includes Samsung's user space NVMe device driver (uNVMe), which is designed specifically for enterprise and data center servers. This white paper will explain the technical details of the uNVMe, which is a part of OpenMPDK, including how they work together to achieve shorter latency and better performance at the system level for enterprise and IT professionals.

The Growing Demand for Optimized Total Storage Solutions

As one of the world's largest suppliers of leading memory and storage products, Samsung has researched, developed and released best quality and highest performance memory and storage products to IT industries for helping their development of various IT products and services like data center, server, PC, mobile phone, database system and etc.

However, after the release of new state-of-the-art Samsung memory products, OEM host vendors and customers typically need to invest significant time and training period to learn how to maximize the latest, advanced features and how to best use each product. In addition, OEM host vendors and customers had to dedicate time and resources to implement software solutions on top of Samsung hardware in order to maximize the potential of each memory product. Unfortunately, this learning curve often caused a delay to fully utilize those Samsung's state-of-the-art features in their host products.

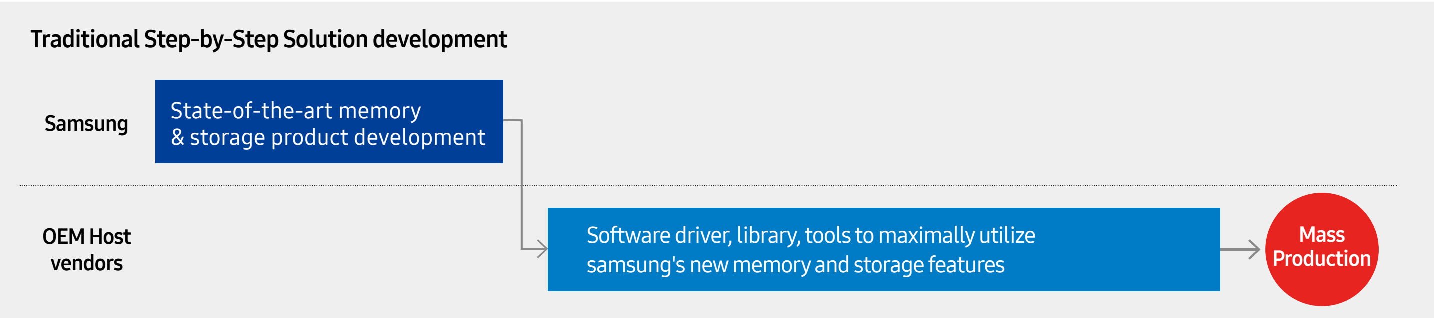

Figure 1: Traditional Solution Development Procedure

Nowadays technology and related product improvement is faster and faster. Rather than waiting the OEM host vendor to study and implement software to fully utilize latest Samsung's state-of-the-art memory and storage product, Samsung decided to help building of software ecosystem and started to develop and provide reference software library, device driver and software tools for saving OEM host vendor's effort. This could be helpful for OEM host vendor to have better competitive power through faster whole system development, integration and solution release to the market.

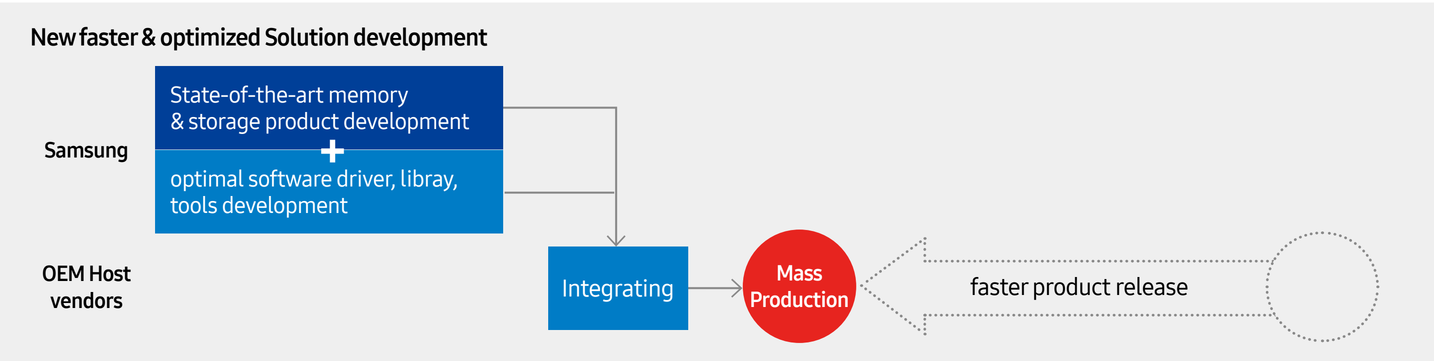

Figure 2: Accelerated Solution Development Procedure by utilizing Samsung's MPDK

Samsung's uNVMe Device Driver

Samsung's OpenMPDK: A Software Platform for Memory and Storage Solutions

To apply the new feature included in latest memory and storage product, software which is supporting those new features should be developed. To better serve OEM host vendors' needs, Samsung provides software packages that include user-level drivers, kernel, drivers, APIs, application-specific user level file systems, sample applications, test suits, profiler and other tools. These software packages are collectively called as Open Memory Platform Development Kits (OpenMPDK).

These OpenMPDKs allow OEM host vendors to more easily integrate with Samsung's memory and storage products and in a fraction of the time. Another critical benefit of the OpenMPDK is that it provides maximally optimized software implementation and better performance for the newest Samsung memory and storage products. Furthermore, these OpenMPDKs are continuously updated and evolving because they are offered as "open source", meaning new ideas and enhancements can be easily added, resulting in faster improvements and better stability from a wide variety of contributors.

As a result, by utilizing the OpenMPDK with the Samsung memory and storage product, the OEM host vendor can have high performance and optimized system level solution in shortened the whole system development and integration time, and their solution could be continuously evolved.

Producing Faster and More Optimized Drivers for Server and Data Center Applications

As one part of the OpenMPDK, the user space uNVMe device driver, which provides an optimal storage solution for enterprise and data center servers, will be the focus of the remainder of this white paper. The device driver is a key and essential component in the software ecosystem for any memory and storage product. Processing data IO requests from the application and controlling hardware of the storage device are performed through the device driver software. Traditionally, device drivers run in kernel mode and are placed under the kernel subsystem as show in Figure 3.

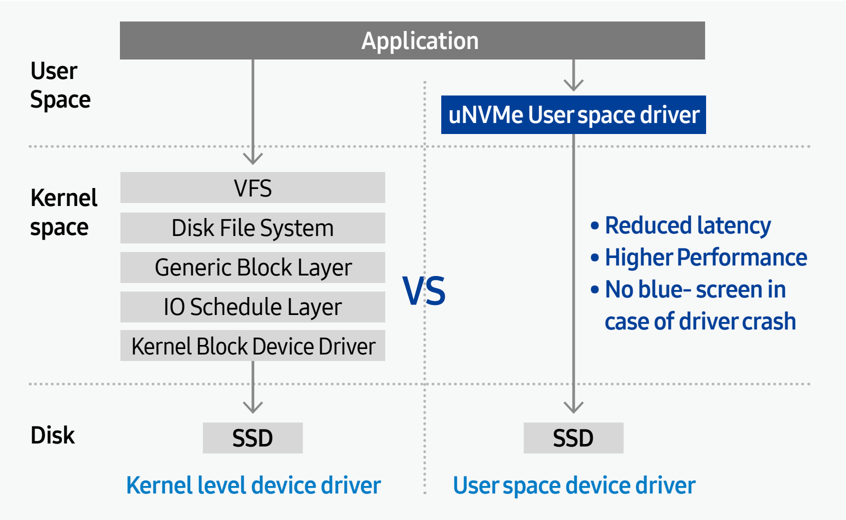

Figure 3: Traditional Kernel Driver vs. Samsung uNVMe User Space Device Driver

In the case of traditional kernel driver methods, host application requests data IO and control this request using system call interface, which is provided by the operating system. The system call then invokes the kernel level device driver via a Virtual File System (VFS) API, target file system for the device, block device driver and IO scheduler. This process imposes massive overhead by passing through many layers before finally executing the real service performing software, called kernel space device driver.

In addition to this overhead, if write system call is executed, the data is copied from the user space memory to the kernel space memory since actual data processing is performed in kernel mode. In the case of read system call, the read data from the device is copied from kernel memory to user space buffer memory. Unfortunately, this redundant data copy also increases the latency and decreases the performance of the storage device.

To make matters worse these issues, when the data IO is requested, the application is sent to the wait queue until the data service is completed in device side and all other previous IO requests from other applications are finished. This requires minimum two context switches —one when it issues the request for data IO and one when it wakes up for the completion of the requested data IO. And, depending on how many other IO processes are in queue for the target device, the latency will likely increase to complete the request. Even with current operating systems supporting polling mode IO processing models, transitions between user-mode and kernel-mode still impose considerably long latency and performance degradation overhead.

Because of these various intrinsic overheads in the traditional IO model, it has been difficult to meet the low-latency and high-throughput requirements of enterprise server and data center server applications—even with the emergence of higher performance storage devices like NVMe SSDs.

Samsung's uNVMe Device Driver

To address these issues, the user-space IO (UIO) system was designed in Linux, which has created a shift toward running storage applications in the user-space context. The Linux kernel provides a user-space application with a means to directly map the memory available to kernel to a user-space range. In the context of device drivers, this allows user-space applications to have direct access on memory and IO configuration registers for storage devices. As a result, all access by the application to the assigned address range ends up directly accessing storage device.
In addition to those good points of the legacy user space device driver, Samsung's user space uNVME driver software incorporate advanced IO architecture like separating IO submissions and IO completions to different CPUs. Furthermore, CPU for IO completion can be configured with a user defined polling period to avoid wasting performance by too long checking whether requested IO is completed or not. As a result, hosts CPU utilization is efficiently improved as well while using user space driver scheme. Therefore scalability, i.e increasing number of attached SSDs, is improved as well.

Sharing uNVMe Source Code Github

Samsung's uNVMe device driver is a user space device driver software that is implemented as a library where sample applications can be linked together. Users may download Samsung's uNVME driver in https://github.com/OpenMPDK/uNVMe.
This particular release includes sample applications and tools for the uNVMe device driver. The sample application is combined with the uNVMe device driver into one process and takes ownership and control of a device upon execution. An NVMe device can be accessed by one application at any given time, while the application can access multiple devices simultaneously. By using the Samsung SDK, the sample application initializes, submits and processes IO workloads directly to attached NVMe devices. Because the server's file system is fixed, there is no need of common file system interfaces such as VFS. These good features of uNVMe device driver reduce overhead, which occurs when the kernel context switching and IO handling happens on interrupt mode.
As a result, Samsung user space uNVMe device driver provide especially more optimized performance in case of lower latency SSDs, like NVMe SSD, as illustrated in Figure 4.

```
uNVMe
├── app
│   ├── external
│   │   └── fio
│   └── fio_plugin
├── driver
│   ├── build
│   │   ├── debug
│   │   └── release
│   ├── core
│   ├── external
│   │   ├── common
│   │   ├── dpdk
│   │   └── spdk
│   ├── include
│   ├── mk
│   └── test
│       ├── command_ut
│       ├── hash_perf
│       ├── iterate
│       ├── iterate_async
│       ├── udd_perf
│       └── udd_perf_async
├── include
├── io
│   ├── deps
│   ├── doxygen
│   ├── kv_perf_scripts
│   ├── src
│   │   ├── common
│   │   └── slab
│   └── tests
└── script
```

Table 1: Source Tree Structure of the Samsung uNVMe User Space Device Driver

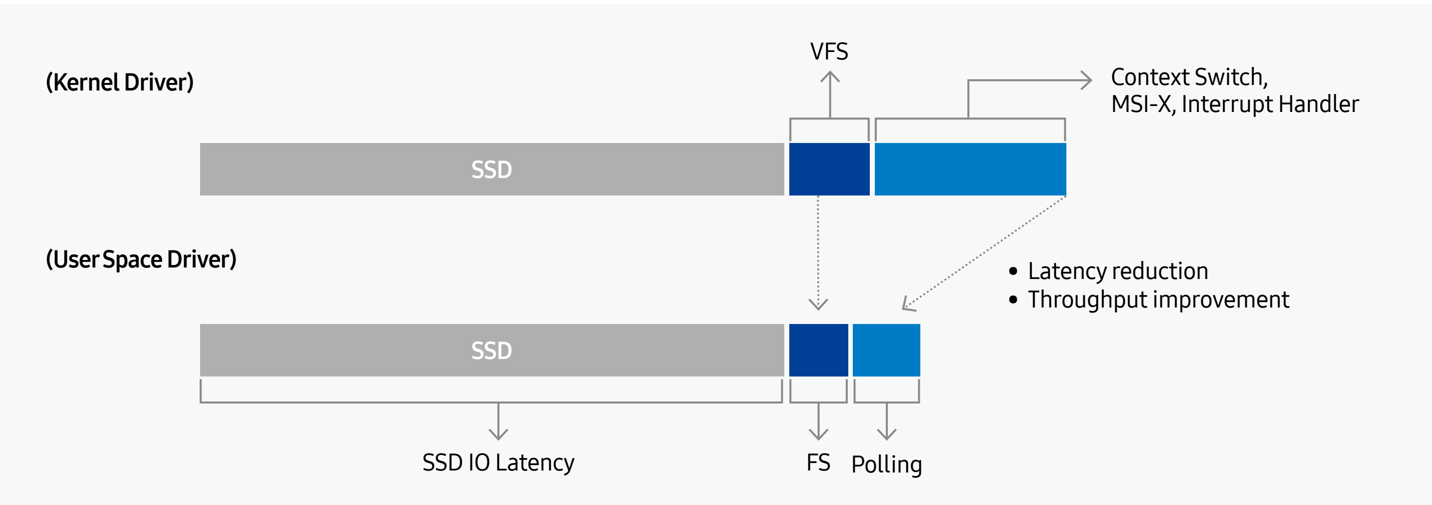

Figure 4: Latency Reduction and Throughput Improvement using Samsung uNVMe User Space Device Driver

Performance Improvement

Performance of Samsung's PM983 NVMe SSD storage was measured using both a kernel device driver and an uNVME device driver in two types of server systems, including an Intel CPU and AMD CPU, as laid out in Table 2.

	System 1 (Intel CPU)	System 2 (AMD CPU)
System	Dell R740xd	SMC 2023US-TR4
CPU	Intel Xeon SP Gold 6142 @ 2.6GHz (per Socket:16core, 32thread)	AMD EPYC 7451 @ 2.0GHz (per Socket:24core, 48thread)
Memory	64GB	
Storage	Samsung PM983 1.92TB x 4ea	
OS	CentOS 7.5 (Linux Kernel 3.10)	
NVMe Device Driver	Kernel Driver: Kernel 3.10 / User Driver: uNVMe Driver v2.0 + FIO Plug-in (https://github.com/OpenMPDK/uNVMe)	
Test tool	FIO 3.3	
IO engine & Workload	libaio for kernel IO / uNVMe2_fio_plugin for user-level IO / Workload: 4KB Random Read	

Table 2: Test system information and measurement condition

By using an uNVMe device driver, the performance improvement is shown biggest in case of random read workload which is most important to data center and enterprise server system. In the case of write performance, where the required write time to the NAND is the bottleneck normally, the improvement is limited but could show improved performance in the future, depending on the SSD writing performance improving.
The improvement in random read performance is shown up to 350% in real server environment testing. When considering the plethora of requests that are received by a server, so even sequential data requests are transformed to random IO requests on the device side, such superior random read performance improvement shows usefulness of Samsung's user space uNVMe device driver for enterprise and data center server applications. Figure 5 below shows the random read measured performance comparison between the uNVMe device driver and the traditional kernel mode driver for the Intel CPU-based test system. Specifically, in Queue Depth 32, a 170% random read performance increase is observed, and, in Queue Depth 128, a 290% random read performance is shown.

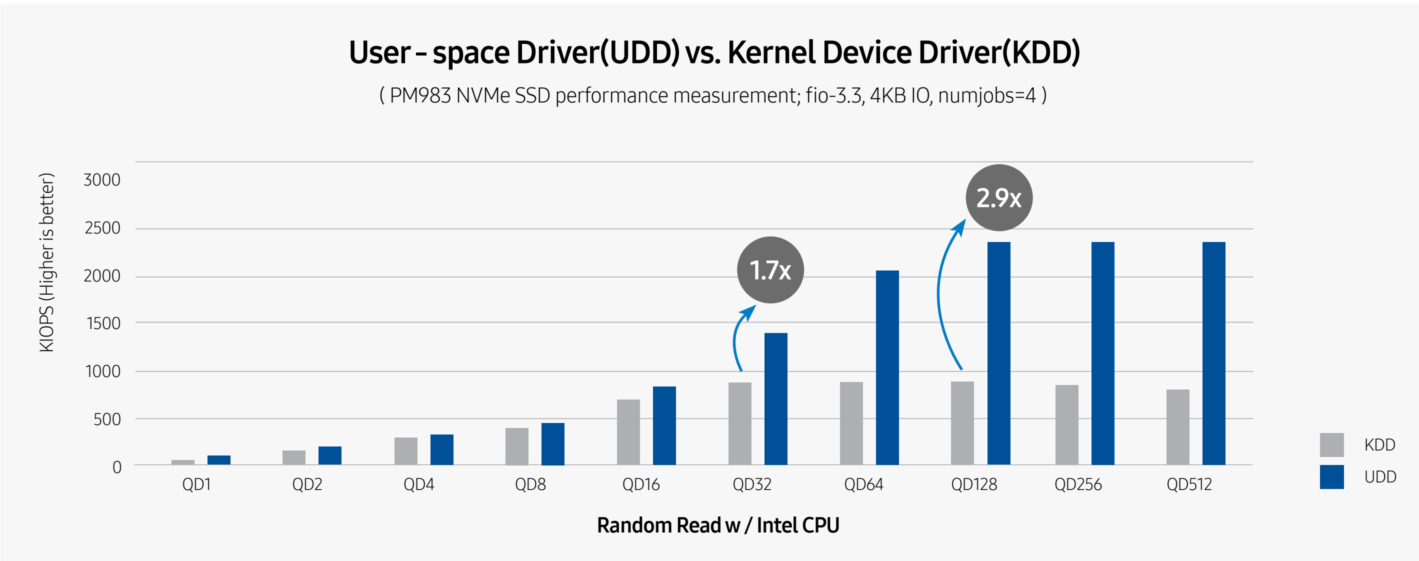

Figure 5: Performance in Intel's CPU: User-Space Device Driver (UDD) vs. Kernel Device Driver (KDD)

Samsung's uNVMe Device Driver

Figure 6 below illustrates the random read performance comparison between uNVMe device driver and traditional kernel mode driver in an AMD CPU-based test system. In Queue Depth 32, a 230% random read performance increase is observed, and, in Queue 128, a 350% random read performance increase is shown.

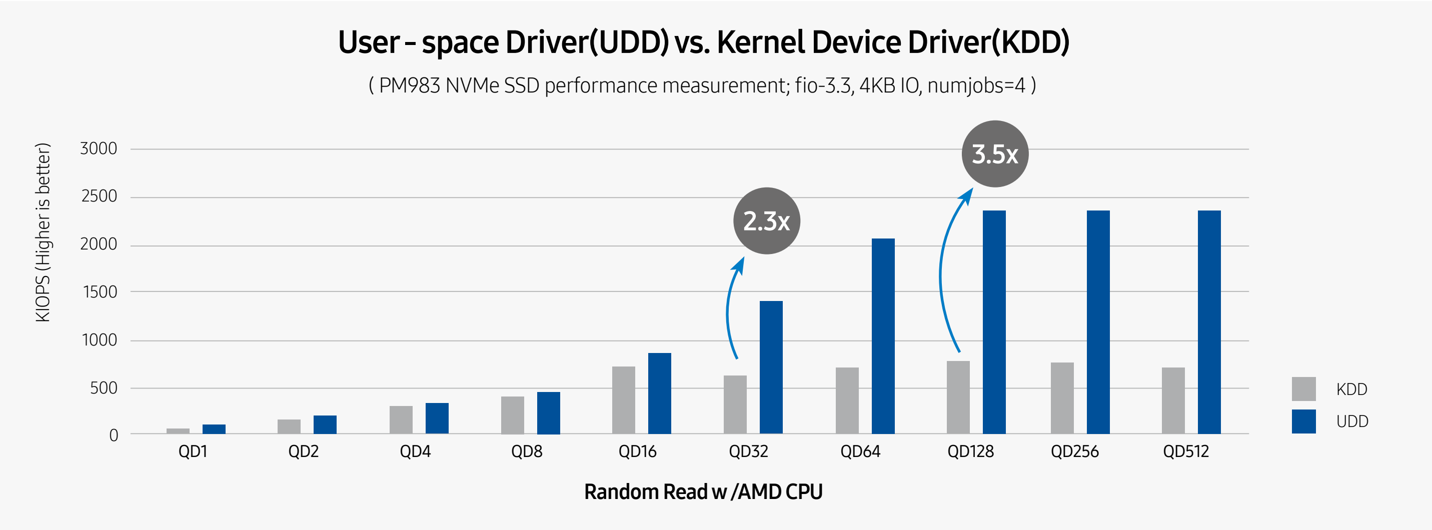

Figure 6: Performance in AMD's CPU: User-Space Device Driver (UDD) vs. Kernel Device Driver (KDD)

Download Samsung's OpenMPDK to Reap the Benefits of uNVMe Device Driver

Samsung's user space uNVMe device driver is an open source reference device driver, which is allowing for simplified development as well as a reduction in the risk of serious bugs within a kernel module when a device driver is developed in a traditional manner (i.e. at the kernel level). Bugs in a driver will not crash the kernel. Updating your driver can be done without recompiling the kernel. At the same time, while kernel-space applications need to conform to General Public License guidelines, user-space driver software are not bound by such restrictions and only a very small kernel module is needed. The most part of the device driver will run in user mode. By utilizing Samsung's user space uNVMe device driver, the context switches, locks and serialization, IO scheduling and many redundancy layers like VFS can be skipped. As a result, the uNVMe device driver delivers much higher performance than using a traditional NVMe kernel level device driver.

This uNVMe device driver is one part of the Samsung OpenMPDK software package, which is distributed free of charge to host OEM vendors in order to reduce time and costs associated with the integration of Samsung memory and storage products. Samsung's OpenMPDK will receive ongoing updates in order to deliver a software platform for effectively and efficiently building host systems that utilize Samsung's state-of-the-art memory and storage features. And, by sharing it as an open source project via Github.com, Samsung is providing software developers an opportunity to improve and further develop the OpenMPDK for various memory and storage solutions.

To integrate a cutting-edge memory or storage solution into your current system with better performance and shorter system integration time, please visit OpenMPDK open source web site[1] and download those reference software, integrate it as guided, test it, and release your whole system product.

1) OpenMPDK open source website is http://github.com/OpenMPDK.

For more information

For more information about the Samsung OpenMPDK and uNVMe User Space Device Driver , visit **www.samsung.com/semiconductor**.

Samsung Electronics Co., Ltd.
129 Samsung-ro, Yeongtong-gu, Suwon-si, Gyeonggi-do 16677, Korea **www.samsung.com** 2018-07

SAMSUNG