SSD blkdev: A kernel level SSD Emulator for host-based FTL

Raghavendra Rao Ananta, Jian Zhou, Jun Wang
University of Central Florida
Email: raghav3276@knights.ucf.edu, {jzhou, jwang}@eecs.ucf.edu

Modern days' Solid State Drives (SSDs) have become open-ended in their design choices. The introduction of the Flash Translation Layer (FTL), to comply with the existing mechanical Hard Disk Drives (HDDs), has been taking a major leap. Modern days FTL uses complex algorithms such as data mining in order to efficiently handle the mappings. Also, research has concluded that decision making of an FTL without the behavioral understanding of the application might lead to performance degradation of the system. Due to these reasons, the processing demands have also increased, and as a result, the placement of FTLs has moved from the SSD to the host side.

Unfortunately, research leap taken toward host-based FTLs cannot be evaluated by conventional simulators or emulators. Hence, in order to make such developments, fast, easy and cheaper, we propose an SSD emulator platform which could be best suited for evaluating host-based or FTLs.

The architecture of the emulator is shown in the Figure 1. At the higher level, the architecture is divided into two parts: a kernel-space entity and a user-space entity. The kernel-space gives the advantage that we can deal with the Logical Block Addresses (LBAs) directly and also allows large chunks of contiguous memory to emulate the flash area. However, the kernel-space doesn't enjoy the luxury of complex libraries, such as complex data mining libraries. Hence, the part of the FTL, which requires the support of these libraries are moved to the user-space. As and when the kernel generates an I/O request, the LBA for that request is sent to the user-space for translation and the I/O is performed (emulated) on this translated address (Physical Page Number).

The following contributions are made by the paper:

- Framework oriented design, such that a new developer can plug the FTL under test at any layer of the system stack.
- As a case study, various performance parameters such as throughput, IOPS and total execution time has been compared for two different filesystems, by varying two different FTL logics.

The validation of the emulator was performed against a commercial SSD. In order to perform the validation, the emulator was configured as a 32 GB SSD, with a 4-way channel-level parallelism. The FTL was chosen as a basic page-level FTL (PFTL). The results showed that the performance of the emulator follows almost the similar pattern as that of the commercial SSD.

As a case study (Figure 2), two FTLs, DFTL [1] and PFTL, were applied onto two filesystems, ext4 and btrfs. The results

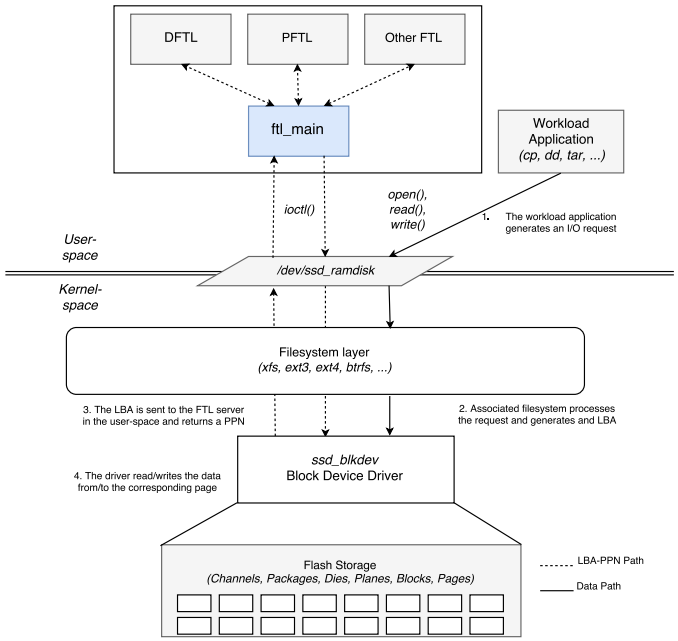

Figure 1: Architecture of SSD emulator

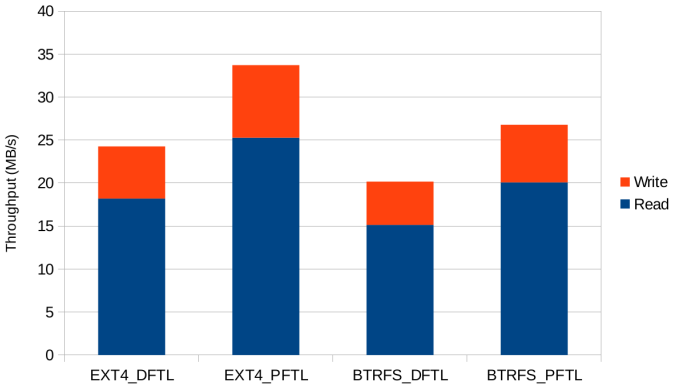

Figure 2: Emulator application

shows that, by varying different FTLs, for the same filesystem, results could vary drastically.

References

[1] Y. Kim A. Gupta and B. Urgaonkar. Dftl: a flash translation layer employing demand-based selective caching of page-level address mappings. In *Proceedings of the 14th international conference on Architectural support for programming languages and operating systems*, 2009.