Overcoming Capacitance Constraints in SSD

Sanghyun Nam
Soongsil University

Seungmin Shin
Soongsil University

Sungjin Lee
DGIST

Bryan S. Kim
Syracuse University

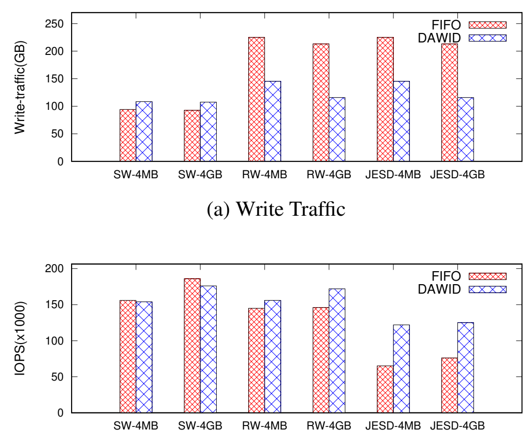
Eunji Lee
Soongsil University

Abstract

The growth in SSD capacity is reaching its limit due to the stunted growth of capacitors—electrical components that store charge to protect data for the volatile memory in case of power loss. While the SSD's capacity scaled from 256GB in 2011 [7] to 30TB in 2018 [1] (100×), the density of Aluminum and Tantalum electrolytic capacitors only by tenfold from 1960 to 2005 [3]. This slow scaling will eventually limit the amount of DRAM that can be used in an SSD, and this, in turn, will also limit the storage capacity as the size of DRAM and aggregate flash capacity proportionally scale [6, 8].

This paper presents `Dawid`, a novel SSD-internal DRAM management scheme that allows the SSD capacity to scale beyond the slow growth of capacitors. SSD-internal DRAM is used for (1) caching translation information (also known as mapping table) and (2) buffering user writes. In typical SSD designs, most of the capacitance is used for protecting the mapping table (to keep as many translation entries in DRAM) and the buffer for user writes is kept at a minimal (just enough to hide the flash program latency) [4]. However, in our design, we take a radically different approach. We buffer more user writes so that mapping entry eviction becomes more efficient by aggregating dirty updates. This substantially reduces the amount of mapping table-related write traffic, and in turn, improves the overall performance.

To realize this design, `Dawid` maintains two data structures: first, *a zero-cost list* that holds the write requests whose mapping entry is already in a dirty translation page, and second, *a max binary heap* that maintains the indexes to translation pages sorted by the number of buffered user write requests associated with that page. When there is sufficient bandwidth at underlying NAND flash subsystem for writes, `Dawid` first flushes user data from the zero-cost list, and then persists the dirty translation pages as ordered by the max binary heap. By doing so, each user write minimizes the number of eventual translation page write, and each translation page write maximizes the number of persisted mapping entries.



(a) Write Traffic

(b) IOPS

Figure 1: Fio benchmark results.

`Dawid` is built upon the current trend of increasing the queue depth of the storage interfaces. SATA and SAS support a single queue with 32 and 245 commands, but NVMe has up to 65,535 queues with as many as 65,536 commands per queue. This extension allows SSDs to further optimize the internal activities by taking advantage of the outstanding request information.

We implement `Dawid` in `FEMU`, an open-source SSD development framework [5]. We assume 1% of the mapping table is protected via capacitors in a 64GB SSD. We measured the performance of `Dawid` using fio benchmark [2], running 4KB sequential writes, 4KB random writes, and the skewed read-write mixed workload that follows JESD219 using 8 threads. A total of 90GB of data was written to the 30GB area.

Figure 1 compares IOPS and the write traffic of FIFO buffer management and `Dawid`. For sequential writes, there are no prominent differences between FIFO and `Dawid`. However, `Dawid` reduces the write traffic by 35% and 46% on average for random writes and JESD219 workloads when the buffer size is 4MB and 4GB, respectively. These workloads have a large footprint at a time window and thus buffering and scheduling them judiciously leads to a large reduction in mapping table persistence overhead. Consequently, IOPS of our design improves up to 18% and 88% for random write and JESD219, respectively.

References

[1] AnandTech. Samsung 30.72 TB SSDs: Mass production of PM1643 begins. https://www.anandtech.com/show/12448/samsung-begins-mass-production-of-pm1643-sas-ssds-with-3072-tb-capaci 2018.

[2] Jens Axboe. fio - flexible i/o tester. https://github.com/axboe/fio, 2021.

[3] Jens Both. The modern era of aluminum electrolytic capacitors. *IEEE Electrical Insulation Magazine*, 31(4):24–34, 2015.

[4] Woon-Hak Kang, Sang-Won Lee, Bongki Moon, Yang-Suk Kee, and Moonwook Oh. Durable write cache in flash memory SSD for relational and nosql databases. In *ACM International Conference on Management of Data (SIGMOD)*, pages 529–540, 2014. https://doi.org/10.1145/2588555.2595632.

[5] Huaicheng Li, Mingzhe Hao, Michael Hao Tong, Swaminathan Sundararaman, Matias Bjørling, and Haryadi S Gunawi. The CASE of FEMU: Cheap, accurate, scalable and extensible flash emulator. In *16th USENIX Conference on File and Storage Technologies (FAST)*, pages 83–90, 2018.

[6] Fan Ni, Chunyi Liu, Yang Wang, Chengzhong Xu, Xiao Zhang, and Song Jiang. A hash-based space-efficient page-level FTL for large-capacity SSDs. In *2017 International Conference on Networking, Architecture, and Storage (NAS)*, pages 1–6. IEEE, 2017.

[7] Samsung. Samsung SSD 830 series. https://www.samsung.com/us/support/owners/product/128gb-ssd-830-series, 2011.

[8] Samsung. Samsung V-NAND SSD 860 QVO. https://www.samsung.com/semiconductor/global.semi.static/Samsung%20SSD%20860%20QVO%20Data%20Sheet_Rev1.pdf, 2013.