International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 9 Issue 8 ǁ 2021 ǁ PP. 37-39

Challenges and Solutions for Improving SSD Performance

Gomanth D Reddy[1] ,Mounika Thatikonda [2] ,Dinesh Motati [3] ,Bhaskar Gurram [4]

[*1]*Department of Computer Science Engineering, SRM Institute of Science and Technology, Chennai, India*
[2]*Department of Informational Technology,* Mallareddy College of Engineering and Technology, *Hyderabad, India*
[3]*Department of Computer Science Engineering, SRM Institute of Science and Technology, Chennai, India*
[4]*Department of Computer Science Engineering, SRM Institute of Science and Technology, Chennai, India*

Abstract

There are many things to consider when we choose the right SSD, like price, performance, capacity, power efficiency, data integrity, durability, reliability etc. But, the criteria of choosing this will vary based on the usage of the SSD. There are many debates on which criteria to choose among all of them, this paper will focus on performance of SSD (and factors those affect it). While selecting an SSD, the amount of performance it has directly affects the user experience and has tremendous impact on overall computing value. Naturally, performance relates to how fast the drive can access, retrieve and save the data. Also, on modern SSDs, the scheduling of I / O s is a challenge for success in the first order. Nonetheless, the best way to optimize I / O patterns is unknown, as a complex layer of proprietary firmware masks several key aspects of efficiency as well as lifetime SSD and losing this important information leads to incorrect conclusions about prototype systems, as well as real-world systems realizing sub-optimal performance and lifetime. Here, we illustrate the performance metrics and the limitations of current SSD modelling tools and disk statistics and also observe an opportunity to resolve this problem by reverse engineering SSDs, leveraging recent trends towards component standardization within SSD.

Keywords: *SSD, HDD, NAND, ONFI.*

Date of Submission: 01-08-2021 Date of acceptance: 15-08-2021

I. INTRODUCTION

The performance of SSD and HDD generally differ and SSDs are fast because of the NAND flash based SSD architecture and they also have distinct performance states. The performance of SSD changes due to the following reasons:

1) Changes over time
2) Depends on write history of SSD
3) Depends on the type of stimulus being applied to the drive.

System designers need performance models for the underlying persistent storage devices and there are optimizations to minimize seeks on tape and on mechanical disks. The performance model here is largely influenced by the mechanical components of the device.

While HDDs have some on-board firmware that can optimize requests, research has repeatedly shown that the OS can further boost performance by reordering requests before sending them to the underlying system in order to better match the device's performance model. There are a number of papers which demonstrate that the performance improvements are through better I /O scheduling on newer and flash- based SSDs, despite the fact that the SSDs are faster and have narrow gap between sequential and random I / O performance. The SSDs present a logical block of address (LBA) interface comparable to HDD, which hide the performance aspects of the SSD.

Thus, system designers struggle to increase the performance of SSD without the performance models. Although there are some well-understood performance issues at the hardware level — such as larger I / O s performing better than smaller ones and parallelism at the banking level —a complex flash translation layer (FTL) in system firmware can also produce substantial performance and lifetime artifacts that are difficult to optimize. Specifically, this firmware will produce objects that affect the writes variance. Such variation can be especially annoying in a program where a write can go to another computer or location to avoid an FTL-internal delay.

HDD and SSD performance are defined most often in terms of three basic metrics: Input Output Operations per Second (IOPS), Throughput (usually expressed in Megabytes per second or MB / s) and

Response Time (or Latency, typically expressed in milliseconds or microseconds). A metric is the output calculation against a given parameter – such as the speed expressed in miles per hour or height in inches.

IOPS refers to the device's IO service transfer rate or the amount of transactions that can take place within a given unit of time (in this case seconds). In IOPS the transaction rate of the IO is calculated.

Throughput – abbreviated as "TP" and often expressed as bandwidth – refers to the data transfer rate or, in this case, the amount of data transmitted to or from the SSD or HDD. Throughput is measured in MB/sec.

Response Time (or Latency)-abbreviated as "LAT" or shortened from Latency-refers to the time it takes for a host-generated command to go to and return to the storage device, i.e. the round-trip time for an I / O request. The response time is calculated in milliseconds or microseconds and is often stated as either an average response time (AVE) or maximum response time (MAX).

Researchers and storage practitioners often resort to different observational heuristics and assumptions rather than make educated decisions. There is little reason to reveal FTL information to researchers outside of a strict Non-Disclosure Agreement (NDA), because FTL is a key value-add over rivals and usually a trade secret. As a result, a number of papers recorded output artifacts they can't explain without internal devices.

There are new possibilities for reverse engineering SSD behavior, but the first opportunity lies in industry-wide flash package interfacing standardization. We demonstrate how we can indirectly infer the policies and mechanisms employed by the firmware of the system by monitoring the electrical signal contact between the SSD's micro-controller and flash packets

II. MOTIVATION

Modern SSDs are a complex embedded hardware platform, including a multi-core microcontroller that manages I / O requests from the host and drives several physical flash packages. SSD FTLs label physical flash pages with logical addresses. Such logical-to-physical mapping is required to simulate overwriting a logical block by writing the logical block to a new physical location, since physical flash pages can only be written once without erasing a larger block. Illustration of the value of recognizing FTL internals relates to intra-SSD compression, an established technique used to minimize physical writing in commercial SSDs. Examined the efficacy of various intra-SSD compression schemes within OLTP workloads. This internal, implementation-specific, FTL feature can affect device lifetime and performance.

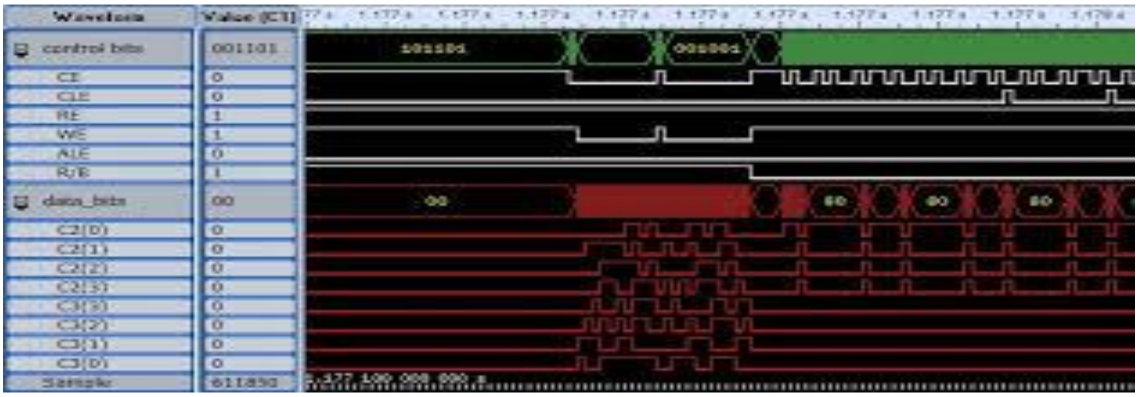
SSDs have low fidelity because they are difficult to model accurately for some reasons. First, some of the mechanisms are used only during periods of inactivity making them virtually unpredictable background operations. Second, some modeling tools rely on speculation or proprietary expertise not inherently applicable to various SSD models. Finally, and crucially, SSDs 'increasingly proprietary and complex nature makes it difficult to understand their internal mechanisms correctly. It is also difficult to model SSDs correctly simply by observing external performance metrics such as latency and throughput, or even by presenting advanced profiling knowledge.

According to the reference 1, they ran three different random work-load on SSD in it’s priming stage using fio benchmark tool. That workload managed its own separate portion of the logical address space to minimize interference. The first workload issued 4 KB requests spread uniformly and randomly throughout the LBA space; the second issued 4 KB requests using an 80-20 distribution (80 per cent of writings submitted to 20 per cent of the LBA space); and the third issued 16 KB requests uniformly random. They ran these separately as 5 minute runs and ran all loads concurrently in fourth run and they wanted the result in WAF. After the runs, the expected a WAF of 0.56 but they got a factor of 0.9 in the mixed run. So, it is unclear how to relate this increase in WAF to increasing individual workload even in this controlled, fairly simple environment. Without a better performance model, it is difficult to know how representative a given set of SSD measurements is.

III. REVERSE ENGINEERING

In some computers, such as D-Link and Schneider Electric Quantum Ethernet Unit, back door has been discovered by reverse engineering of the firmware. If the data stored on the devices is encrypted, reverse engineering can be applied to get the encryption keys and even the encryption key which will recover the clear data.

In 2006, all the major flash vendors formed a “Open NAND Flash Interface” (ONFI) which creates chip level interface for NAND flash memory to host the systems and this normalization can be helpful in reverse engineering of SSDs. Using workloads, they monitored the command sequences of flash packages and they were able to infer firmware policies and mechanisms from high-level to low level operations. Choosing a single flash package of OCZ vertex II 55GB SSD, they were able to attach all probes to pinouts and extend them to logic-analyzer. Then using that logic analyzer, they collected some data as represented in the below figure. The implementation of the proposed methodology poses many technological constraints. Flash packages and SSD's keep decreasing in size. Many modern flash module packages use ball grid arrays (BGA) with pinouts at the bottom of the box. Electrical delays introduced by the tracing system will cause the unit to fail. With special equipment and experience, these technological challenges may all be solved, but can hinder the widespread use of this system.



Waveform	Value (C1)
control bits	001101
CI	0
CLE	0
RE	1
WE	1
ALE	0
R/B	1
data_bits	00
C2(0)	0
C2(1)	0
C2(2)	0
C2(3)	0
C3(3)	0
C3(2)	0
C3(1)	0
C3(0)	0
Sample	611850

There is a JTAG-compliant hardware debugging protocol for many embedded devices, including SSD hardware and firmware. JTAG is the industry standard for Circuit Boards inspection. SSD developers can inspect and alter on-board memory content using JTAG, and break and phase through code execution.

IV. CONCLUSION

As with the proprietary mechanisms employed in industrial SSDs, researchers and device designers are frequently left in the dark. The findings from the tests above show that many existing performance modeling approaches can be very inaccurate; it is difficult to say whether a sample of measurements is representative without more insight into firmware behavior. Other than reverse engineering, our other promising option is to exploit the JTAG interface.

REFERENCES

[1]. Zuck, A., Philipp Gühring, Tao Zhang, Donald E. Porter and Dan Tsafrir. “Why and How to Increase SSD Performance Transparency.” *Proceedings of the Workshop on Hot Topics in Operating Systems* (2019): n. pag.

[2]. Zhang, Li & Hao, Shen-gang & Zheng, Jun & yu-an, Tan & Zhang, Quan-xin & Li, Yuan-zhang. (2015). Descrambling data on solid-state disks by reverse-engineering the firmware. Digital Investigation. 12. 10.1016/j.diin.2014.12.003.