Project 3: Memory and Storage Performance Profiling

Rojan Karn and Almog Cohen

ECSE 6320 - Advanced Computer Systems

School of Engineering

Rensselaer Polytechnic Institute

Introduction

The purpose of our project is to gain a deeper understanding of the performance of modern memory and storage devices. Our goal is to measure the read/write latency and throughput of these devices under various data access workloads using publicly available software packages. Through these experiments, we aim to observe the trade-off between access latency and throughput, as explained by queueing theory. Ultimately, our findings will provide valuable insights into the behavior of memory and storage devices and how they interact with different workloads.

Background and Motivation

The performance of memory and storage devices is critical to overall system performance. With advancements in technology, these devices have become faster and more efficient. However, the trade-off between access latency and throughput has become increasingly significant. This project applies queueing theory to practical experiments, analyzing this trade-off.

Objectives

The primary objective of this project is to carry out comprehensive experiments to measure the read/write latency and throughput of memory and storage devices under various data access workloads. We will use publicly available software packages to perform these experiments, including Intel Memory Latency Checker (MLC) and Flexible IO Tester (FIO). The experiments will cover a range of settings, including different read/write intensity ratios, data access sizes, and throughput vs. latency measurements. Additionally, we will use queuing theory to explain the trade-off between access latency and throughput. Finally, we will compare our results with

the specifications of the Intel Data Center NVMe SSD D7-P5600 (1.6TB), which lists a random write-only 4KB IOPS of 130K.

Hardware Environment

Cache and memory testing was performed on the following environment:

Quad-Core Intel Core i5-1038NG7

Num. of Cores: 4 Cores

Num. of Threads: 8 Threads

CPU Speed: 2 GHz

Turbo Boost CPU Speed: 3.8 GHz

L1 Cache (per core): 80 KB

L2 Cache (per core): 512 KB

L3 Cache (Shared): 6 MB

Memory: 16 GB DDR4 3200 MHz

Storage testing was performed on a different machine, with the following environment:

Quad-Core Intel Core i5-4590

Num. of Cores: 4 Cores

Num. of Threads: 4 Threads

CPU Base Speed: 3.3 GHz

CPU Turbo Boost Speed: 3.7 GHz

Total Cache: 6M

SSD: Samsung 500 GB 940 EVO

Experiment Setup

Cache and memory

Cache and memory testing was done using the Intel Memory Latency Checker (MLC) software package. Using this software, both latency (in ns) and bandwidth (in MB/sec) were measured against various parameters such as read-write intensities and data access size. Each test performed with MLC provided 19 injection delay (in ns) data points showing throughput (bandwidth) vs. latency.

Tests were performed on a CLI with the following command format:

./mlc –loaded_latency -Wn -l256

The read-write intensity is controlled by the "-Wn" flag, where "n" is a number corresponding to a certain read-write ratio. For example, in this experiment "-W2" was used to specify a 2:1 read-write ratio and "-W6" was used to specify a 0:1 read-write ratio (write-only). The specific values of "n" and their meanings can be found in the MLC documentation. Omitting the "-Wn" flag will mean that the test will be performed in read-only mode.

The data access size is controlled by the "-l256" flag. Including "-l256" in the command will cause the test to have a 256B data access size. Omitting the "-l256" flag entirely will cause the test to have a 64B data access size.

The tests that were performed for cache and memory using MLC are a combination of the following parameters:

Data access size: 64B or 256B

Read-write intensity: Read-only, write-only, 1:1, 2:1, and 3:1

* It is important to note that the data was tested using sequential access only for cache/memory.

Storage

Long-term storage and SSD testing was conducted using the open-source Flexibile I/O (FIO) utility with a dedicated 100 GB empty partition running on the Ubuntu 20.04 operating system. The purpose of these tests is to measure the performance of a storage device under different workloads. The test included four different scenarios: read-only, write-only, 70%:30% read vs. write, and with data access size of 4KB/32KB/128KB.

The FIO tool was used to simulate the workloads and measure the performance. The following parameters were changed in each scenario:

- Read-only: set the write percentage to 0%
- Write-only: set the read percentage to 0%
- 70%:30% read vs. write: set the read percentage to 70% and write percentage to 30%
- Data access size: set the blocksize parameter to 4KB, 32KB, and 128KB

Below are the FIO command line arguments used:

Read-only Tests:

fio --name=read_test --ioengine=libaio --iodepth=64 --rw=read --bs=4k --direct=1 --size=1G --numjobs=1 --runtime=60 --time_based

--group_reporting --filename=/dev/sda5
--output=/home/acohen/Desktop/FIO_RESULTS/read_only_4k.txt

Write-only Tests:

fio --name=write_test --ioengine=libaio --iodepth=64 --rw=write --bs=4k
--direct=1 --size=1G --numjobs=1 --runtime=60 --time_based
--group_reporting --filename=/dev/sda5
--output=/home/acohen/Desktop/FIO_RESULTS/write_only_4k.txt

Mixed Read-Write Tests:

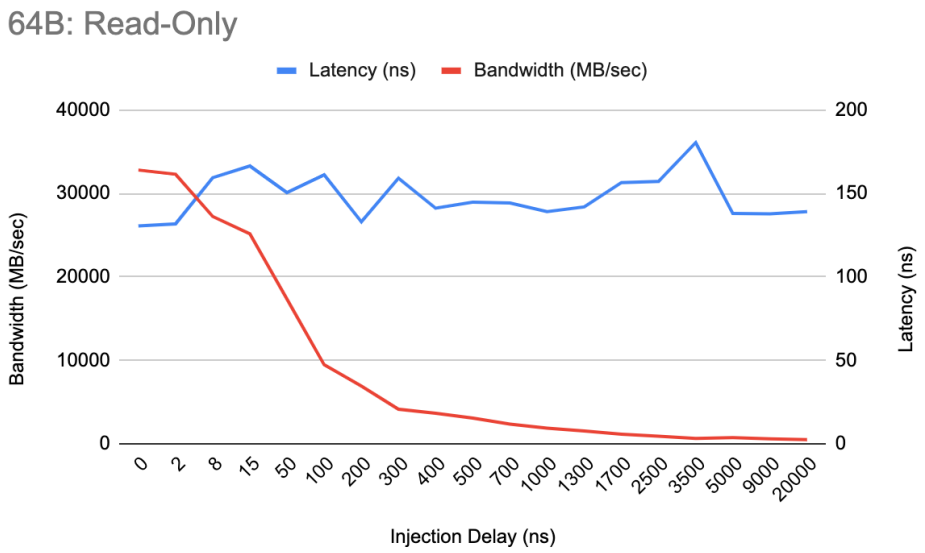
fio --name=randrw_test --ioengine=libaio --iodepth=64 --rw=randrw
--rwmixread=70 --bs=128k --direct=1 --size=1G --numjobs=1
--runtime=60 --time_based --group_reporting --filename=/dev/sda5
--output=/home/acohen/Desktop/FIO_RESULTS/read_write_70_30_128k.
txt

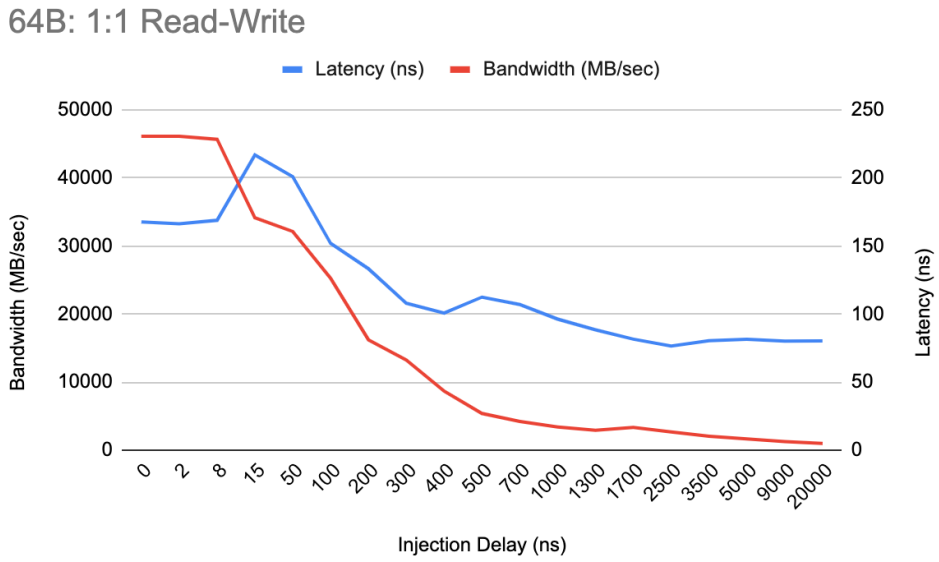
Experimental Results

Cache and memory

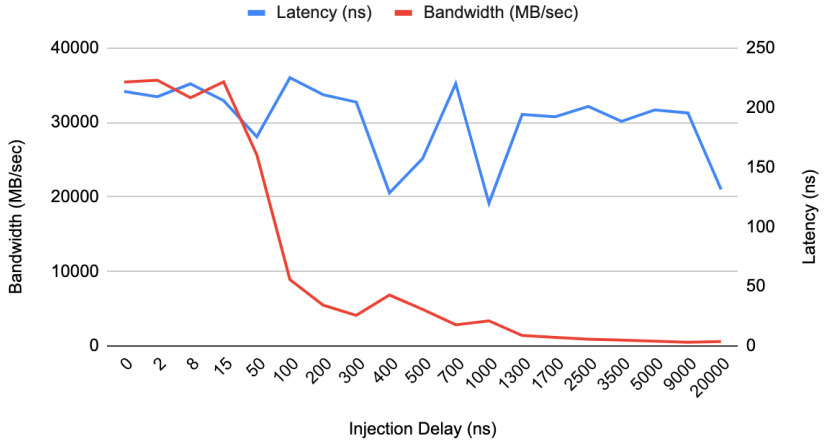
The following charts were generated using data from the MLC software with the specified parameters. In these charts, the injection delay is shown on the X-axis, the bandwidth (red line) is shown on the left Y-axis, and the latency (blue line) is shown on the right Y-axis.

64B Data Access Size:

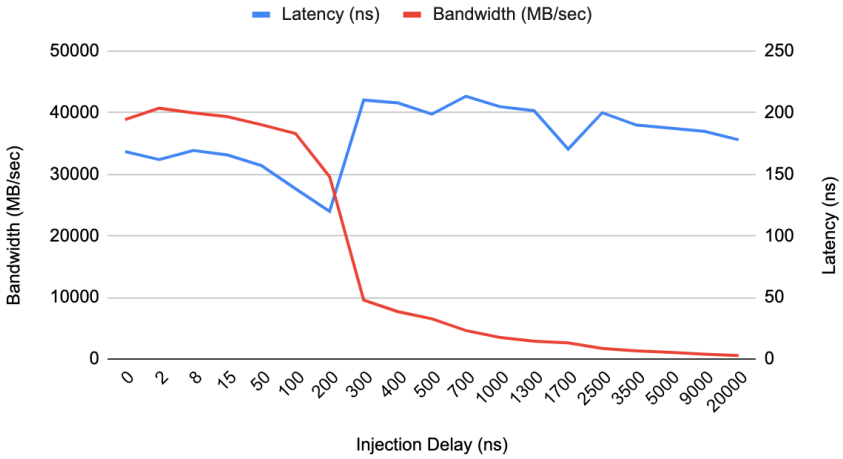




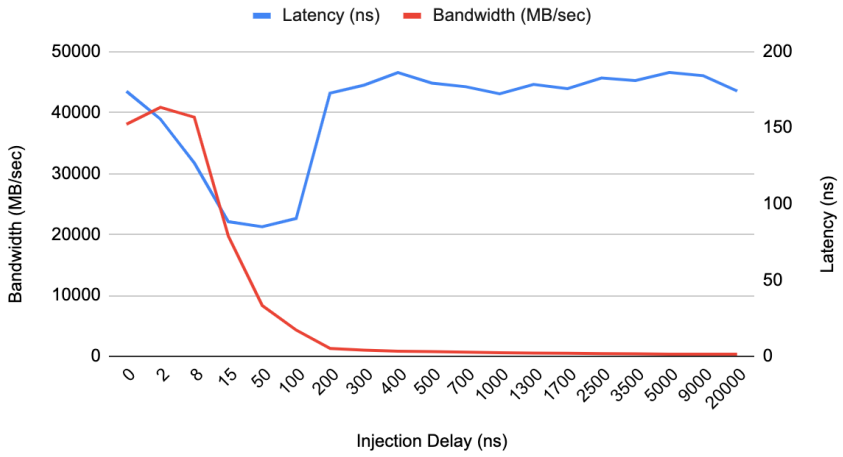
64B 2:1 Read-Write



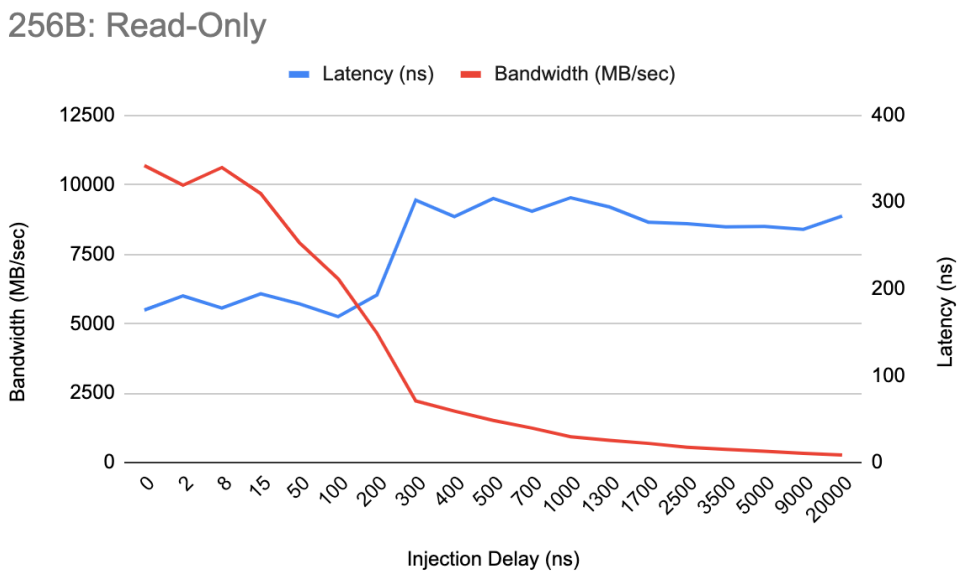
64B: 3:1 Read-Write

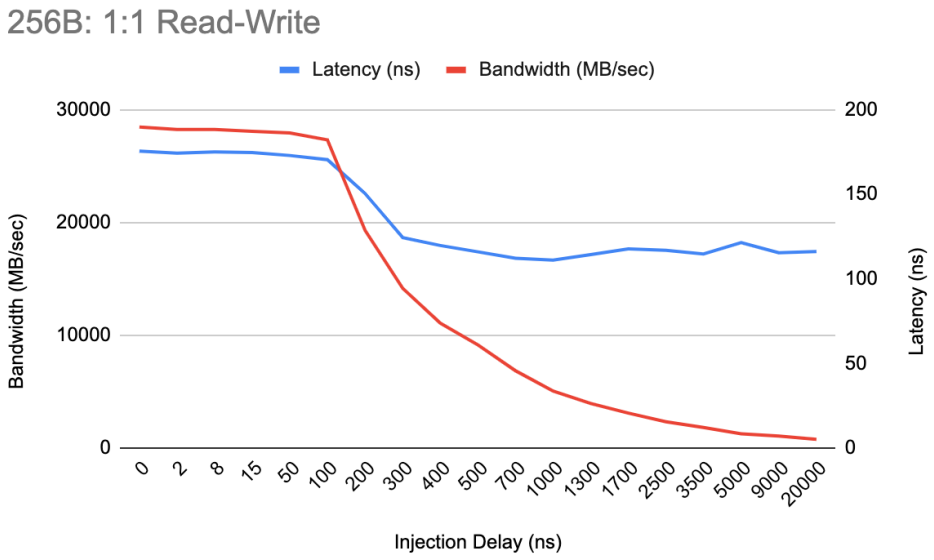


64B: Write-Only

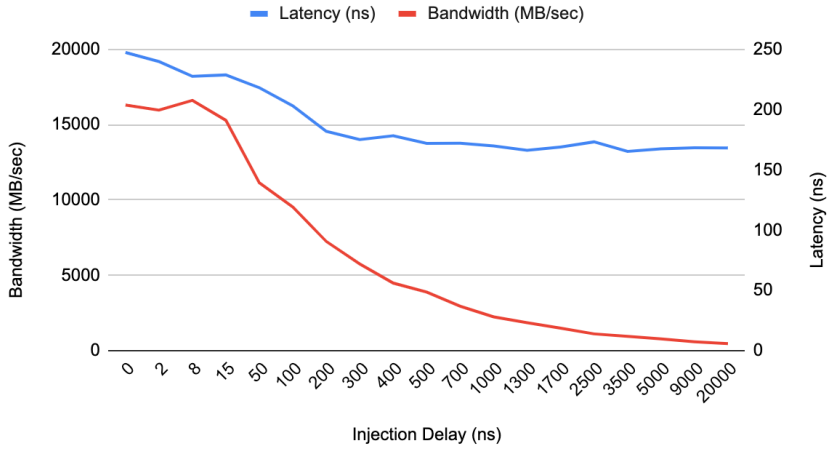


256B Data Access Size:

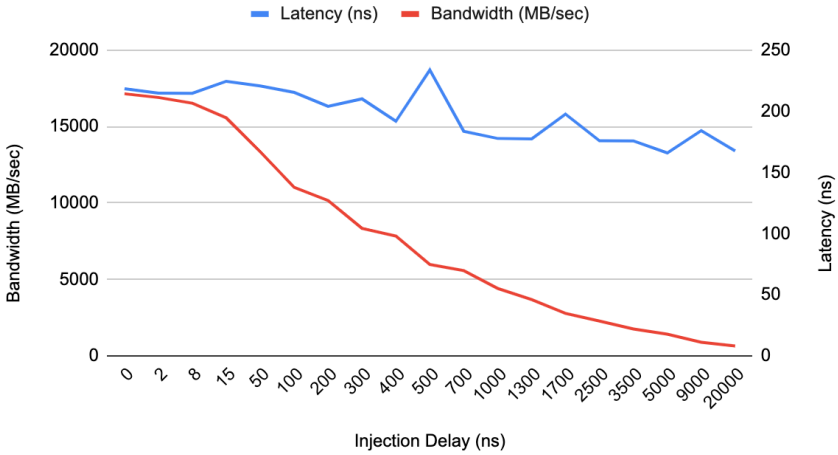




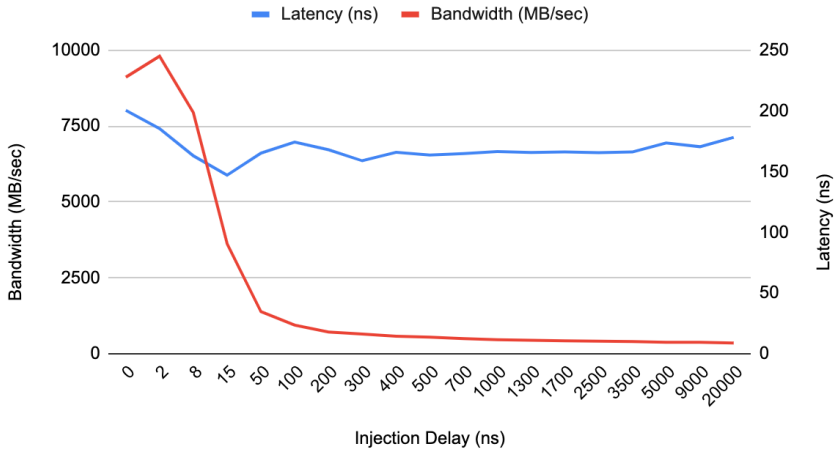
256B: 2:1 Read-Write



256B: 3:1 Read-Write



256B: Write-Only

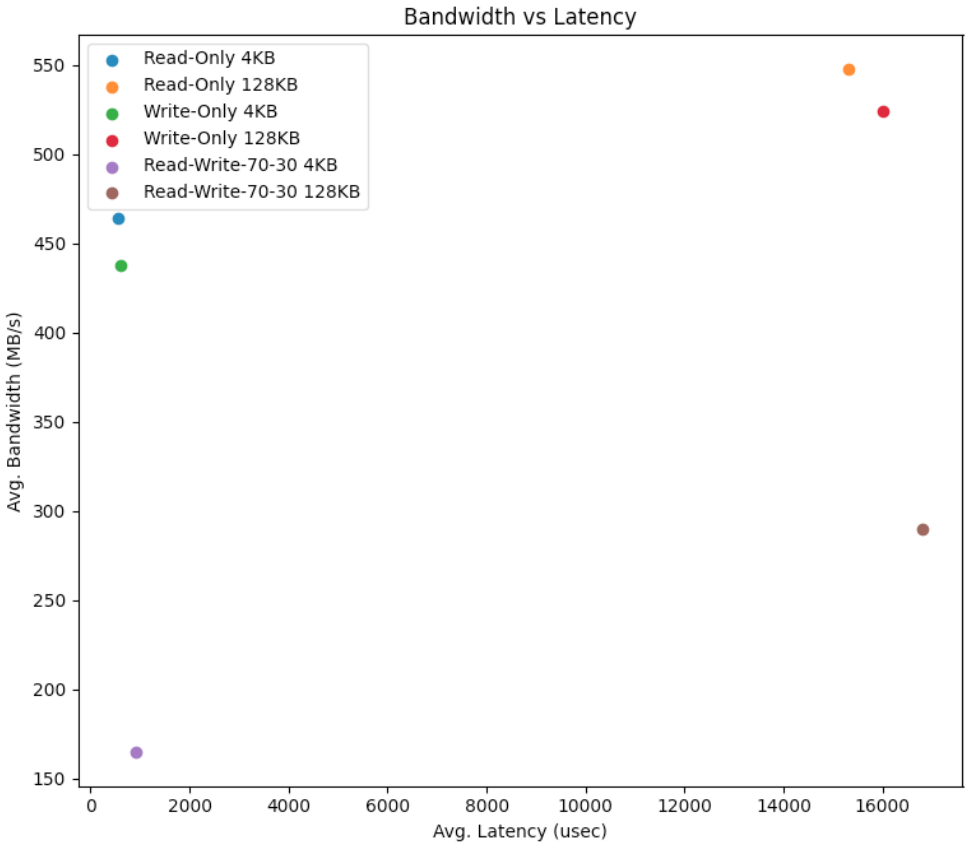


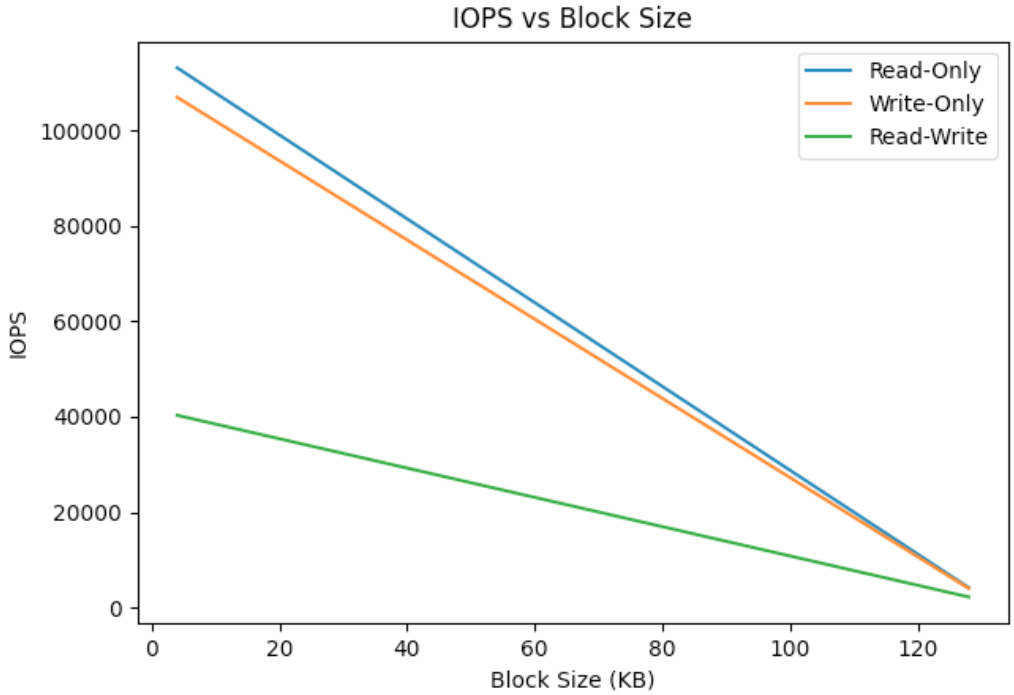
Storage

The following charts and data gathered were generated from the FIO commands above. The tests were run multiple times to ensure consistent results, and the average latency, bandwidth and IOPS values were recorded.

Test	Block Size (KB)	IOPS	Avg. Latency (usec)	Avg. Bandwidth (MB/s)
Read-Only 4KB	4	113197.87	564.91	464
Read-Only 128KB	128	4178.33	15312.61	548
Write-Only 4KB	4	107012.09	597.58	438
Write-Only 128KB	128	4000.46	15995.52	524
Read-Write-70-30 4KB	4	40311.407	918.934	164.92
Read-Write-70-30 128KB	128	2208.954	16803.851	290

Table 1: *Raw Average FIO Results*





Results Analysis

Cache and memory

The graphs displaying latency and bandwidth vs. injection delay show a clear tradeoff between access latency and throughput. In most cases, when the bandwidth is high, the latency is high as well. The results of these experiments can be analyzed through the framework of queueing theory, which is the study of queues used to predict queue lengths and waiting times.

In every graph, the bandwidth decreasing as injection delay increases is characteristic of queueing theory. This is because as injection delay increases, the queue length also increases, and the average time a request spends in the queue increases as well. As a result, the number of

requests that can be processed in a given time decreases, which reduces the overall throughput or bandwidth of the system.

In terms of data access size, it is shown that 256B causes a higher latency than 64B in most read-write intensities. This result was expected because larger data access sizes will result in higher latency due to the increased time it takes to transfer the larger amount of data. In the context of queueing theory, a larger data access size will increase the service time of each request, which can result in longer queueing delays and higher overall latency. Queueing theory also explains why 256B data access size resulted in lower bandwidth in most cases than 64B, as larger data access sizes increase queueing delays and decrease bandwidth due to the increased time required to process each data transfer.

In regards to read-write intensities, the data shows that the latency tends to be more consistent across bandwidths/injection delays when the application is read-only or write-only than when there is a mix of reads and writes. The dropoff in bandwidth seems to slow down as the application becomes more and more read-heavy according to the graphs. However, this also comes with increased latency as the read intensity increases. This is a clear depiction of the tradeoff between latency and throughput as an effect of the read-write intensities.

Overall, the tradeoff between throughput and latency suggests that there is an optimal point of throughput and latency that programmers should attempt to achieve to optimize the efficiency of their system. The graphs show trendlines of the two metrics, so programmers can use this data to choose their distribution of throughput vs. latency. For example, the intersection between the two lines may be an ideal point of acceptable bandwidth and latency for an application. Programmers

should also consider how read- or write-heavy their application is as this may have an effect on their system's efficiency.

Storage

The test measured the input/output operations per second (IOPS), average latency in microseconds, and average bandwidth in MB/s for different block sizes and read/write operations. From the data gathered, it can be seen that the lower block access sizes, directly result in lower IOPS, regardless of Read/Write/Read-Write operations. Specifically, For read-only operations, there is a significant difference in performance between block sizes. The 4KB block size showed a much higher IOPS (113,197.87) than the 128KB block size (4,178.33), but the 128KB block size had a higher average bandwidth (548 MB/s) compared to the 4KB block size (464 MB/s). Smaller block sizes result in more I/O operations per second, which puts more load on the CPU, but also require fewer I/O operations for the same amount of data. Therefore, larger block sizes have a higher average bandwidth, but smaller block sizes have higher IOPS.

For write-only operations, the trend is similar to read-only operations. The 4KB block size had a higher IOPS (107,012.09) than the 128KB block size (4,000.46), but the 128KB block size had a higher average bandwidth (524 MB/s) compared to the 4KB block size (438 MB/s).

For the read-write operations, the difference in performance between block sizes is even more significant. The 4KB block size had a much higher IOPS (40,311.407) than the 128KB block size (2,208.954). However, the 128KB block size had a higher average bandwidth (290 MB/s) compared to the 4KB block size (164.92 MB/s). In this case, the queuing delay is even more

pronounced since the SSD has to perform both read and write operations, which puts a higher load on the CPU.

In general, as the block size increases, the number of requests per transaction increases, resulting in increase to the average time per request, but a lower number of block requests in the queue. Since SSD has limited resources, when the number of requests increases, it can lead to a queuing delay backlog. Therefore, a key metric to look for is the SSD's utilization rate (μ).

When comparing industrial-grade SSDs to consumer-grade SSDs, their IOPS (Input/Output Operations Per Second) rating can be vastly. IOPS is a measure of the number of read and write operations that the SSD can perform in a given time period. Industrial-grade SSDs often have higher IOPS ratings than consumer-grade SSDs, which can make them better suited for use in applications that require a high degree of input/output activity, such as servers.

Summary of Findings

This project aims to analyzea the tradeoff between access latency and throughput in the context of queueing theory, specifically regarding cache and memory. It’s been shown that a larger data access size results in higher latency and decreased bandwidth due to increased queueing delays. Additionally, read-write intensities affect latency and throughput, with read-heavy applications resulting in slower dropoff in bandwidth but increased latency. This report also examines the impact of block sizes on input/output operations per second (IOPS), average latency, and average bandwidth for different read/write operations. It finds that smaller block sizes have higher IOPS but lower average bandwidth, while larger block sizes have higher average bandwidth but lower

IOPS. Overall, the findings of the report suggest that there are tradeoffs between access latency, throughput, and data access size in cache and memory systems, and between block size, IOPS, and bandwidth in storage systems.

References

"Queueing Theory." *Wikipedia*, Wikimedia Foundation, 18 Feb. 2023, https://en.wikipedia.org/wiki/Queueing_theory.