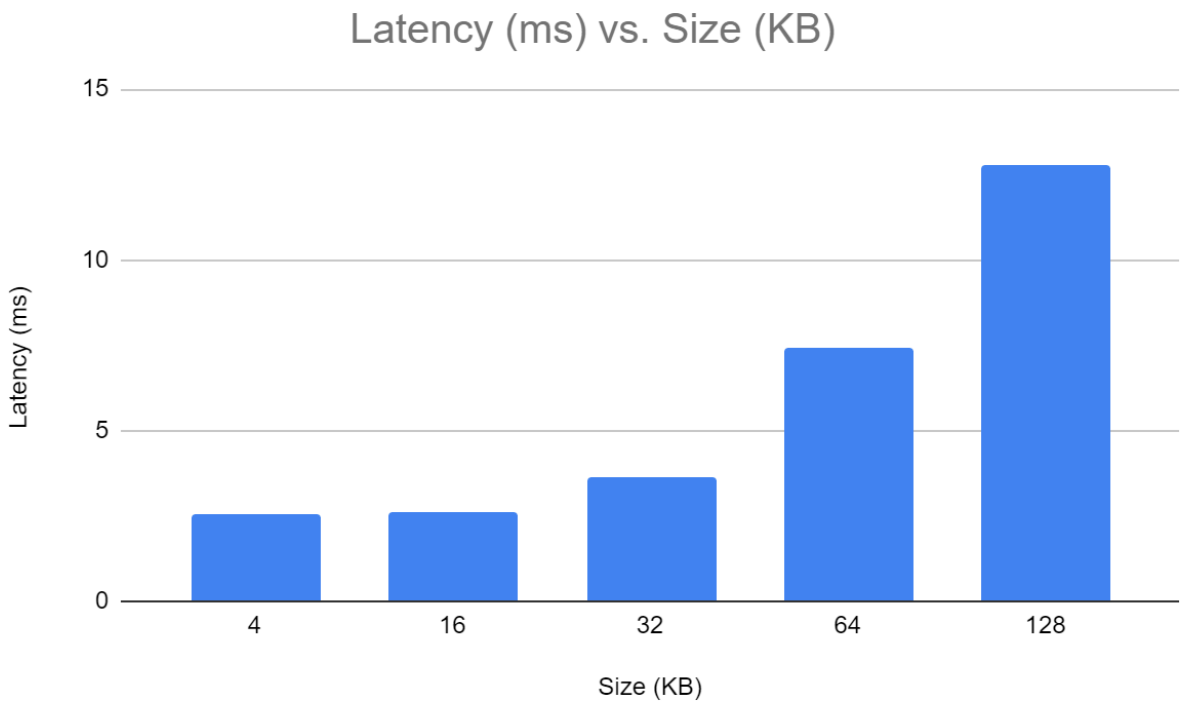
Raymond Zhu

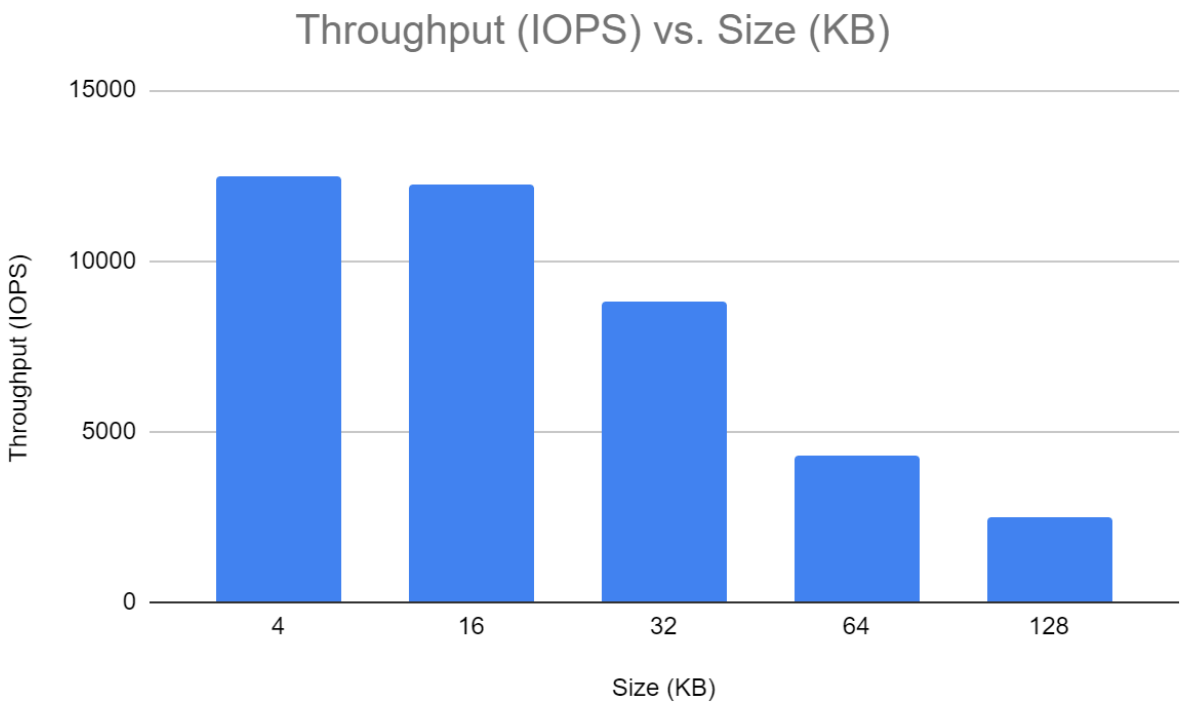
Project 3

Data Access Size:

The first step of this project started by testing different data access sizes and recording the latency and throughput results. To perform this section of tests, the parameter of --bs or block size was changed from 4kb to 128kb. The other parameters of name=ssd_test, filename=/mnt/d/testfile, size=1G, rw=randread, rwmixread=50, direct=1, iodepth=32, numjobs=1, runtime=60, time_based, and ioengine=libaio were all kept the same for each size test. As shown below, as the size increased, the latency increased, but the throughput decreased.

Size (KB)	Latency (ms)	Throughput (IOPS)	Throughput >=128KB (MB/s)
4	2.55863	12500.28	
16	2.6122	12242.73	
32	3.632	8806.33	
64	7.44021	4299.33	
128	12.78096	2502.95	328.0667





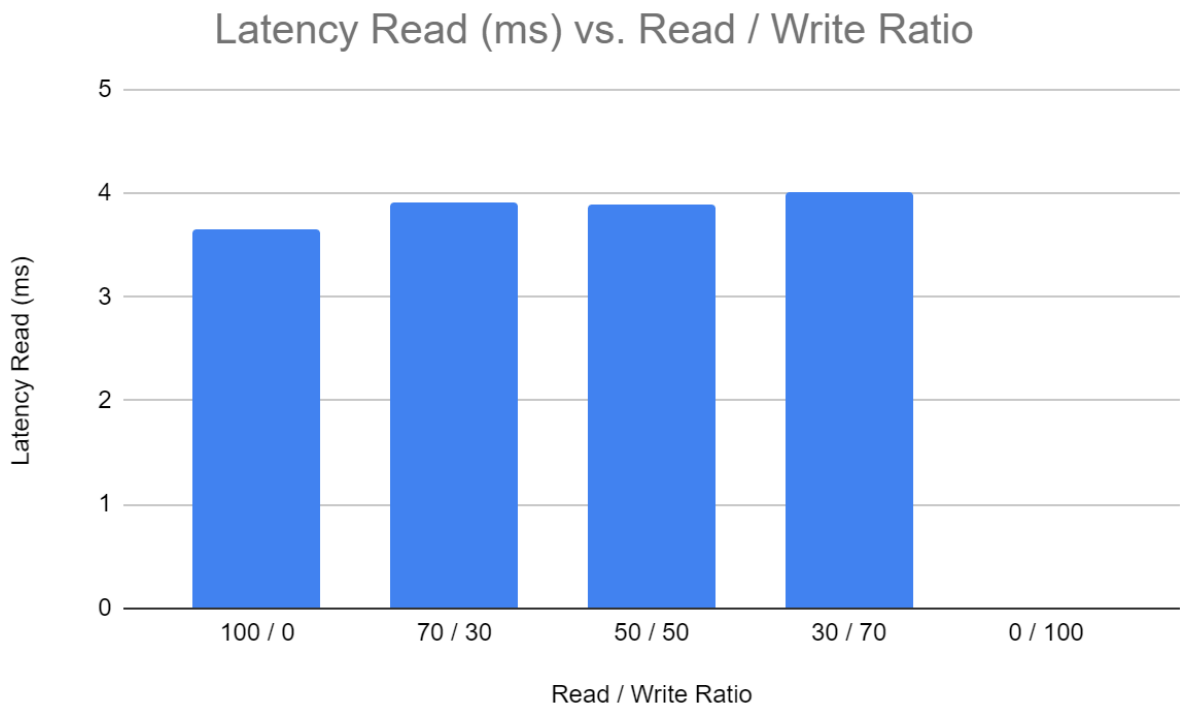
From these results and graphs, it follows that the smaller data access sizes allow for more efficient use of the SSD's resources, leading to lower latency and higher throughput. As the

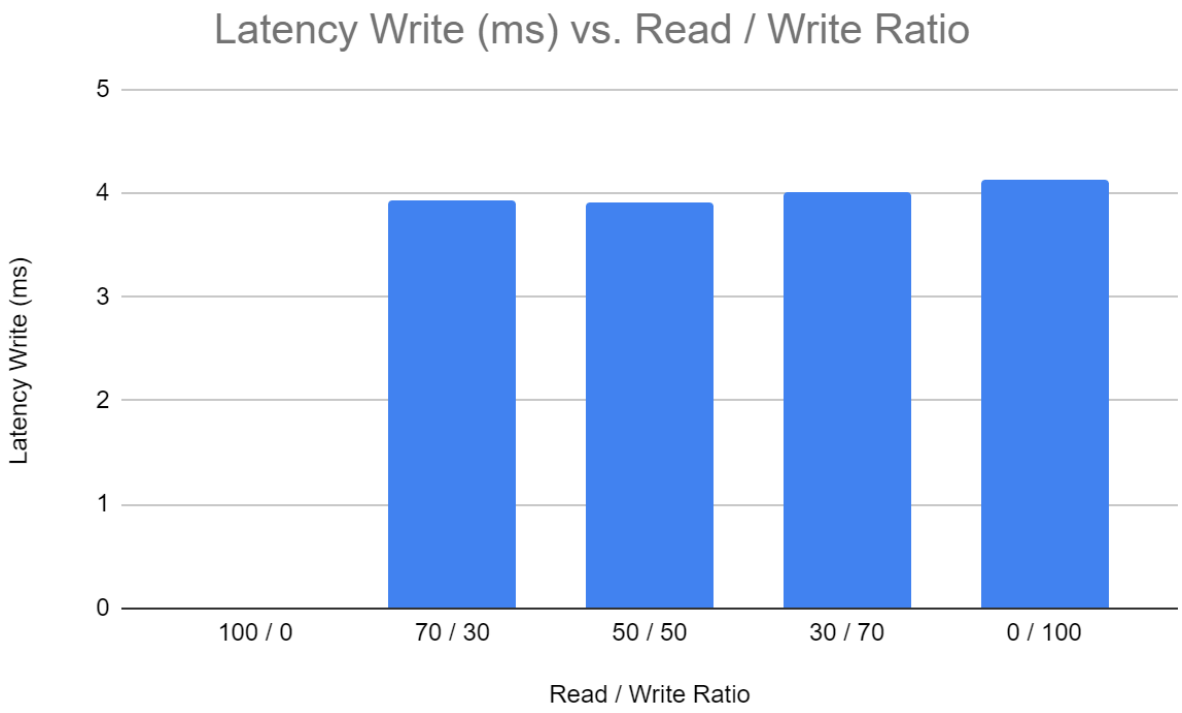
access size increases, the system struggles to keep up, resulting in higher latency and lower throughput.

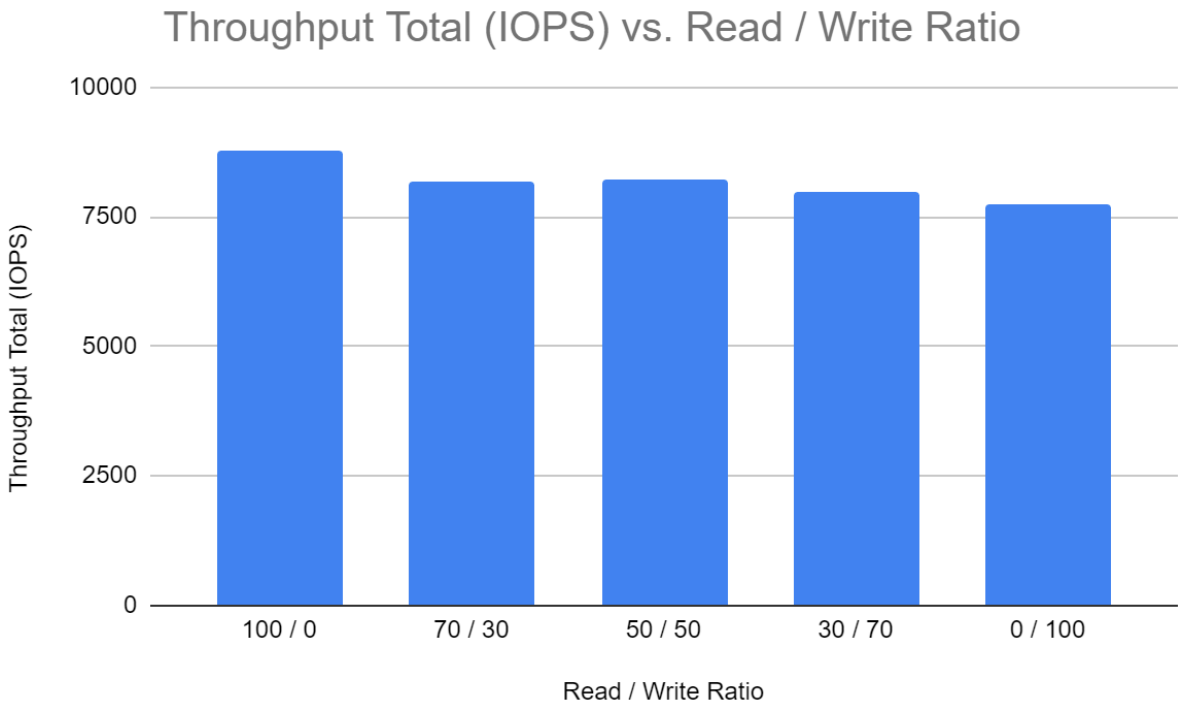
Read/Write Intensity Ratios:

The next experiment involved changing the read/write ratio and recording the latency and throughput. This required changing the parameter of rw=randread from the previous experiment to rw=randrw and changing rwmixread to the percentage of read needed for that test. The other parameters of name=ssd_test, filename=/mnt/d/testfile, size=1G, --bs=32k, direct=1, iodepth=32, numjobs=1, runtime=60, time_based, and ioengine=libaio were all kept the same for each of the read/write ratio tests. The data and resulting graphs can be seen below.

Read / Write Ratio	Latency Read (ms)	Latency Write (ms)	Throughput Read (IOPS)	Throughput Write (IOPS)	Throughput Total (IOPS)
100 / 0	3.64295	0	8780.45	0	8780.45
70 / 30	3.91417	3.9325	5706.53	2453.13	8159.66
50 / 50	3.89121	3.90745	4099.73	4102.78	8202.51
30 / 70	4.00566	4.01864	2395.85	5571.43	7967.28
0 / 100	0	4.13084	0	7742.97	7742.97







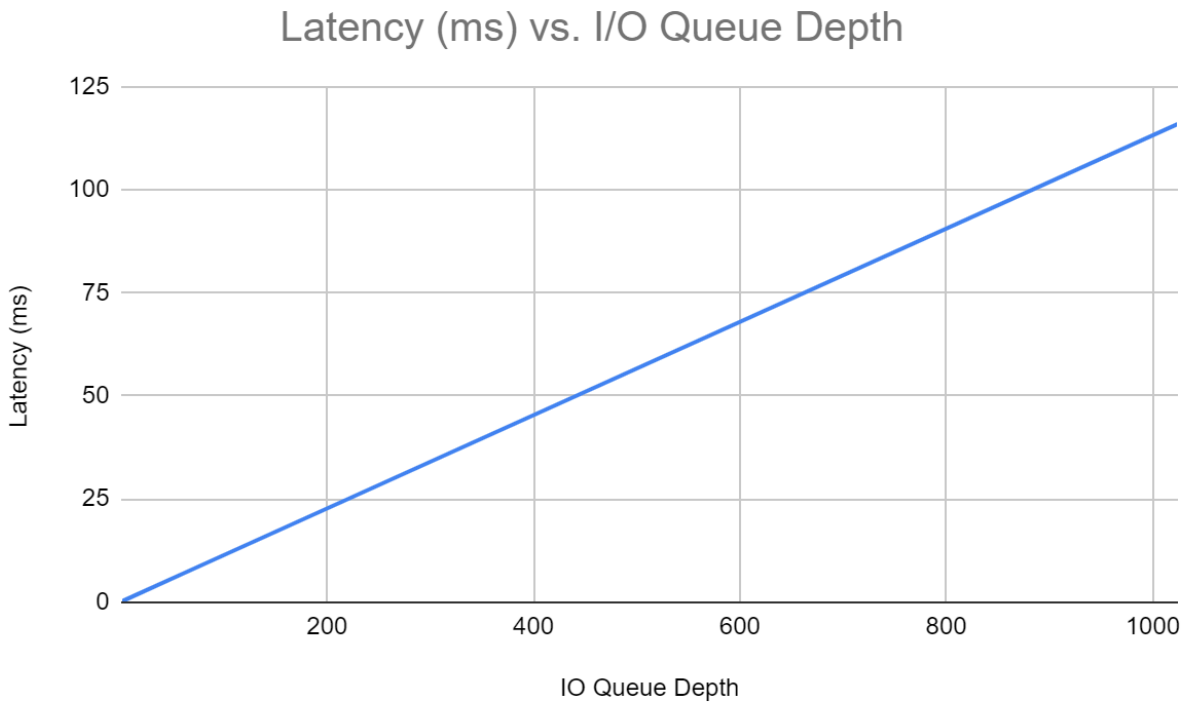
From these graphs, we can see small increases in latency for both read and write. The highest throughput occurred when the SSD was primarily performing read operations, as reads

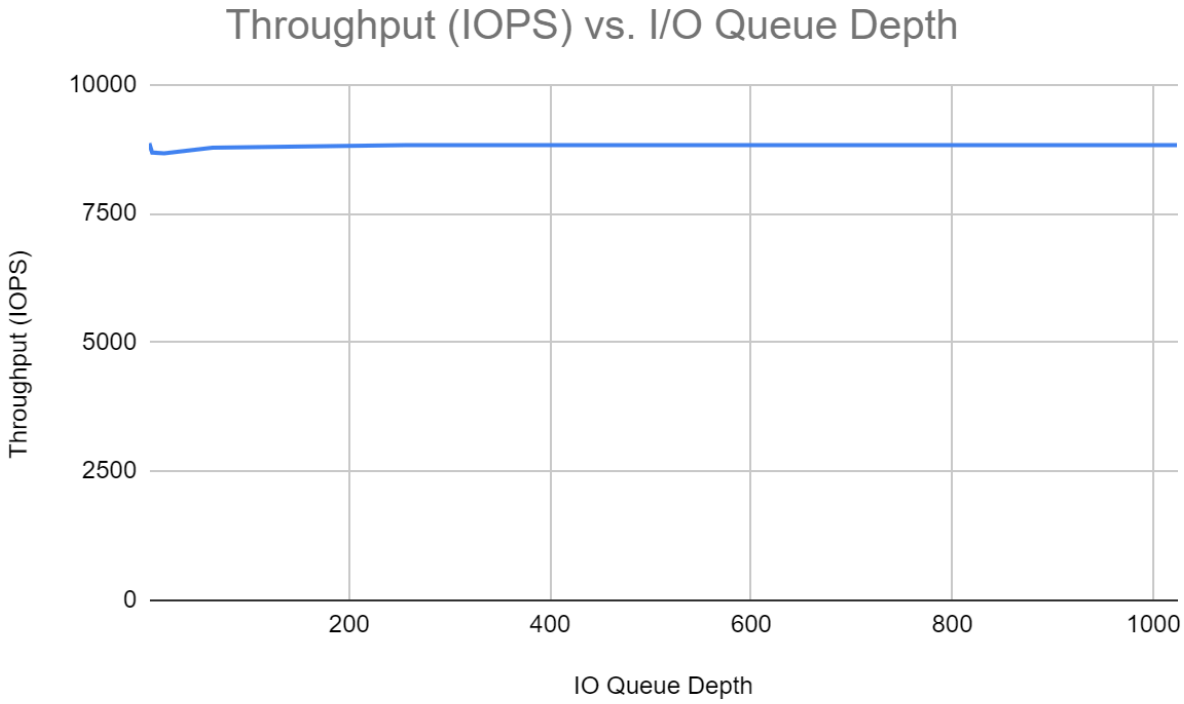
generally require less processing time than writes. As the write intensity increases, the overall throughput decreases due to the additional time required to handle writes.

I/O Queue Depth:

The last test was conducted by changing the number of I/O units. To do this, the parameter of --iodepth=32 was changed from 1 to 1024. The other parameters of name=ssd_test, filename=/mnt/d/testfile, size=1G, rw=randread, rwmixread=50, --bs=32k, direct=1, numjobs=1, runtime=60, time_based, and ioengine=libaio were all kept the same for each of the I/O tests. Data and graphs are shown below.

I/O Queue Depth	Latency (ms)	Throughput (IOPS)
1	0.11172	8861.83
4	0.45974	8678.08
16	1.84556	8663.78
64	7.29303	8773.3
256	29.00775	8822.45
1024	115.88171	8827.07





The graphs show that latency has a steady increase as the I/O queue depth increases. This is probably because each request must wait longer in the queue before it can be processed. The

overall throughput did not seem to change by a significant amount. It seems that once the SSD reached a saturation point in terms of queue depth, increasing the depth further didn't increase the throughput because the internal controller is already operating at its maximum efficiency. Adding more does not add more work for the SSD to do, and therefore the throughput stays constant.

Overall, the results show a clear trade-off between access latency and throughput. Smaller data access sizes result in lower latency and higher throughput, while larger data access sizes increase latency and reduce throughput. Similarly, higher read intensities lead to better performance than write-intensive workloads. The SSD also has the ability to maintain consistent throughput across different queue depths but it comes at the cost of increased latency as the queue depth grows.

I went back and performed these experiments again but with bs=4KB. The throughputs observed in these experiments was significantly lower than the 130K IOPS expected from the Intel Data Center NVMe SSD D7-P5600 for random 4KB writes (my observed throughputs ranged from 125K to 100K). This discrepancy could be due to differences in hardware architecture or the type of workloads the SSD is optimized for. It is possible that the client-grade SSD tested in these experiments may not have the same level of optimization for write-heavy workloads as the enterprise-grade Intel SSD. I also performed this project on the 4-5 year old laptop I received when I first arrived here at RPI. There may be some issues that came from wear and tear of not taking that well of a care for it.