Flash Drives (Us Crazy)

Chris Hinrichs Joshua Yanchar
{hinrichs@cs.wisc.edu, yanchar@cs.wisc.edu}
University of Wisconsin - Madison

Abstract

The next important secondary storage technology, Flash Memory, has finally reached a level of maturity where Flash SSDs are actively being considered to replace HDDs in many applications, including consumer-grade PCs. As flash memory performance and density continue to improve, it will only become more important to understand the performance ramifications of selecting a flash SSD over a more traditional HDD. Keeping the design and organization of both SSDs and HDDs in mind, we designed a test suite that should emphasize the differences between these technologies. While much of the data we've collected has made sense after some analysis, some results have continued to confound despite extended analysis.

1 Introduction

Calling Flash Memory 'the next important secondary storage technology' sounds like a rather bold claim, but it certainly appears justified. While Flash is not the only technology attempting to replace the humble HDD, it is by far the most mature. Due to Flash's solid state nature, there are fundamental advantages over current, mechanical, drives; including far more resistance to physical jolts and stresses. Additionally, replacing all of the moving parts with electronics strongly reduces power usage and heat generation, and eliminates noise and mechanical latencies. Any solid state technology could share Flash's advantages over traditional mechanical devices, but no competing nonvolatile memory technology appears to be anywhere near matching the density or production-cost of Flash.

Unfortunately, Flash isn't better in every way; HDDs still have advantages in capacity and price. (Flash drives are now available in relatively small multi-GB sizes, but Flash should continue to grow exponentially, like other transistor-based technologies, as the manufacturing techniques improve.)

As there are both advantages and disadvantages to Flash, it is worth examining the performance profile of Flash to determine if there are any additional workload factors that should be taken into consideration when evaluating the appropriate technology for any project.

We hope to clearly illustrate the general performance patterns of current Flash SSDs, any workloads that are difficult for flash to perform, and how to avoid those difficult workloads if possible. In order to provide some context for our conclusions, we will walk through the following topics: Section 2 will give an overview of flash technology, emphasizing implementation details where they might reasonably be expected to impact performance. Section 3 will describe our testing hardware, the tests we performed, and the motivation for those tests. Section 4 will review selected graphs of data recorded during testing, and our analysis. We will conclude in Section 5, noting any unexplained anomalies we found in the data.

2 Flash Memory

Flash Memory was invented by Dr. Fujio Masuoka for Toshiba in 1984. The first incarnation, NOR Flash, was an EEPROM replacement, allowing in-circuit programming. As an EEPROM replacement, byte-accessibility was an important feature, and read speed was emphasized over write speed.

The implementation of Flash requires a two-step write process; involving an erase step after which the memory is ready to be written to. Read speeds were fast; write speeds were slow, and erase speeds were slower still. To reduce the costs involved with erasing, Flash chips were divided into large erase blocks. The erase step would be applied to an entire erase block at a time, amortizing the cost of the operation. To be useful, erase blocks must be significantly larger than individual read blocks.

Unfortunately, in addition to erasing being much slower than other operations, it also stressed the substrate with significantly higher voltages than either reading or writing. This stress will, over time, cause irreparable damage to the erase block. To counteract the possibility of certain blocks wearing out before others, a layer of indirection known as the Flash Translation Layer (FTL) has been added to Flash chips. This proprietary protection layer is implemented differently by each manufacturer, sometimes differently from one revision to the next within the same company. The purpose of the FTL is to distribute writes evenly across erase blocks; and is most often accomplished through the use of additional 'invisible' capacity that the user never sees or uses directly.

Five years after the introduction of NOR Flash memory, NAND Flash, targeted as a secondary storage replacement, was invented. NAND Flash significantly improved storage density, write/erase times, write/erase durability, and cost-per-byte. It also replaced the previous byte-accessible organization with a block-accessible organization, like other secondary storage devices. This new organization requires the addition of a small read cache, which, we believe, has a visible performance impact in certain situations.

3 Test Methodologies

Our experiments were conducted on a P4 machine running Ubuntu Linux. It was disconnected from the internet, and was made quiescent during all tests. Three drives were present: A Seagate 80GB SATA disk; an IBM 15.8GB SSD drive, and a Transcend 32GB drive. The boot volume was mounted on the Seagate drive. All tests were run from a separate USB jump drive so that loading and running the test program would not cause extra work to be done on any drive under test.

In order to prevent the Linux kernel from buffering or caching any of this data, which would influence our tests, we used the O_DIRECT flag when opening the devices. This hint tells the 2.6 kernel not to cache any data read from the device.

Apart from preliminary max-sustained read and write tests, all tests consisted of a workload in which a certain number of blocks were read or written, and each was separated by a certain stride. This was repeated 32 times (to look for effects of timing) for each test, and each test was repeated 10 times, (to average out noisy results) for a total of 320 repetitions of each set of parameters.

Two categories of tests were performed. In the first, a series of sectors at a randomly chosen location (chosen at a different disk location for each iteration so as to defeat drive caching,) were read in order (separated by a stride,) and then re-read in a random order. The motivation of this test is to highlight differences between the Seagate disk, which has mechanical delays, and the two SSD disks.

The second test consisted of a write followed by a read to a similarly randomly chosen location on the disk. The purpose of this test is to exercise the drive's write capabilities, and try to force a commit by reading from it. Admittedly, simply re-reading the sector may not have much of an effect on the write cache, but we believe it was better than nothing.

Once the tests were run to completion, the results were graphed in a number of ways. The graphs we have chosen to include are CDF plots, relating what percentage of iterations would have completed within the time indicated on the x-axis.

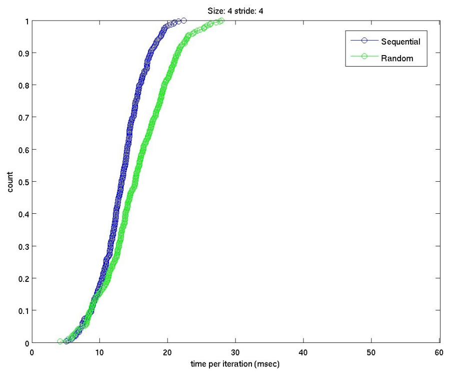
4 Test Results

4.1 Read Test Results

4.1.1 HDD Results

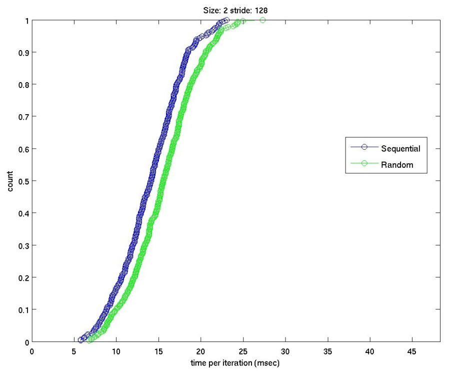
When testing the HDD, our results largely matched our expectations. In general, sequential orderings performed better, with random orderings matching sequential in the best cases, and doing notably worse in the average case. In general, increasing the stride increased both the average time to completion of a test iteration and the variance in test completion times. Lastly, increasing the length of the test increased the time to complete and slightly decreases the random test's performance relative to the sequential test.

HDD: (Typical Results, match expectation)

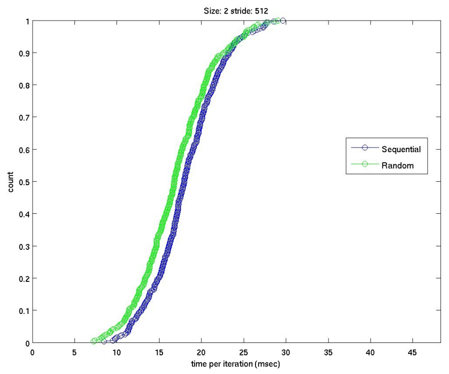


There was however, an anomaly. After stride hit 512, there were times when random sequences completed faster than sequential orderings. This only occurred when the test length was low but more than 1. (Test length 1 means there is no difference between random or sequential.) As the test length increased, the effect diminished, disappearing entirely after length 32. We believe we stumbled upon the track size of the HDD. If the stride is on the order of the size of a track, and the blocks you want to read are placed such that you incur less rotational delay when reading them in an order other than sequentially, random has the opportunity to have slight performance gains relative to sequential. With a test length of two, random would be expected to perform better than sequential 50% of the time, and exactly the same the rest of the time.

HDD: Immediately before anomaly



HDD: Start of track-length anomaly

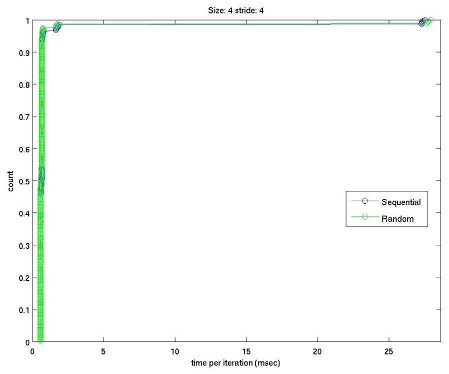


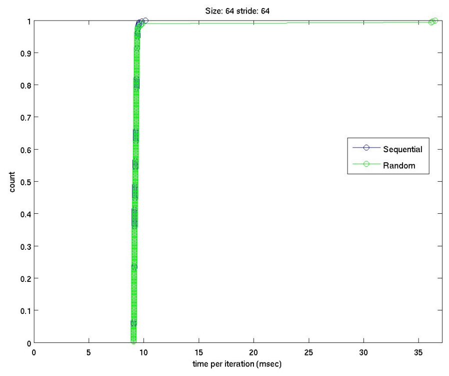
As Flash devices are SSDs, we expect none of these effects to be visible in the results for either the IBM or the Transcend. In general, this expectation was found to be correct.

4.1.2 IBM Results

The IBM results were remarkably stable. We saw almost no variance from iteration to iteration, and different sets of parameters produced shockingly similar graphs. Of interest, on workloads no larger than 64 blocks, about 2% of runs resulted in "outlier" results, which take about 27.5 or 55 msec longer than usual. We hypothesize that these correspond to extra FTL-related metadata recording/cleanup. These outliers do not appear on workloads larger than 64 blocks. In fact, an effect can be seen where the worst case performance improves drastically when the size of the workload increases through 64 blocks and the stride increases through 128 blocks.

IBM: Very stable, includes small number of outliers.





Stride was found to have no direct effect on the time taken, however for the combination of large strides and large test sizes, the completion time variance increases slightly.

No appreciable difference between random or sequential reads was found.

4.1.3 Transcend Results

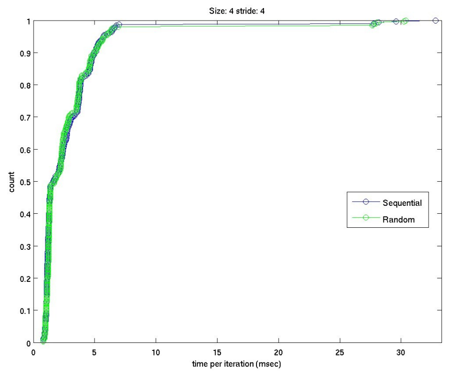
Transcend results were somewhat similar to IBM results, however there was far more noise and variation for all test results, in addition to effects not seen in the recorded IBM results. For example, while IBM had %2 of the results as outliers, the Transcend has that *and* 35% of the single block reads have an extra delay between 0 to about 2.5 msec. On two block reads, the delay increases to about 50% of the time.

Read time increases linearly with **size**.

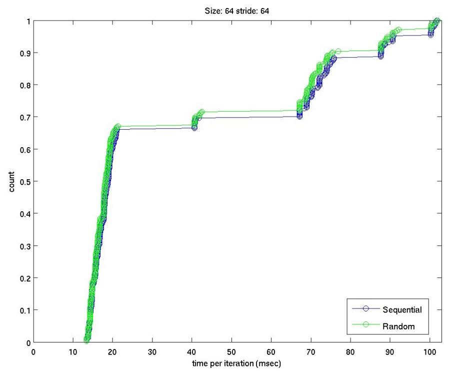
For **strides** larger than 512 the variance of the workload increases noticeably.

For workloads **sized** more than 8 blocks, there is a 'stair step' behavior, displaying many small modes of operation. Stride does not appear to be a factor in this behavior, until the test length hits 1024, and the stride reaches 2048. At this point, the lines for sequential and random separate significantly, and delays become even more common than before.

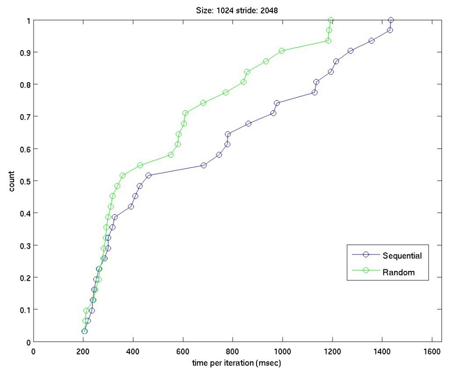
Transcend: Note larger variance than IBM



Transcend: Note multimodal performance arc. (Random vs Sequential difference was noted multiple times, though we were unable to determine any pattern, as one would do better than the other seemingly at random.)



Transcend: New anomaly at Size 1024, Stride 2048

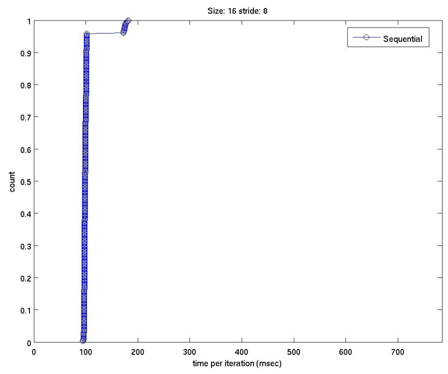


4.2 Write/Read Test Results

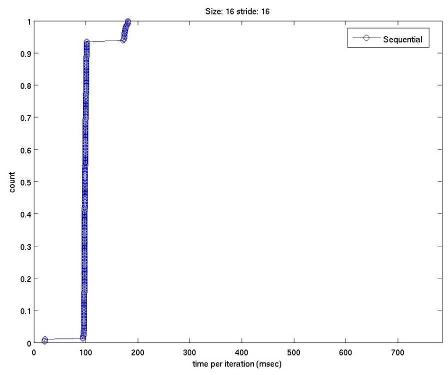
4.1.1 IBM Results

Again, the IBM results proved to be exceedingly stable, showing little to no variance in completion times when parameters remain unchanged. There are clear modes of operation, with each mode 75msec of delay away from the previous mode. Furthermore, the occurrence of each operational mode is strongly tied to both the length of the test, and the stride, appearing to be a linear function of their product. Please note that the time taken within each operating mode does not significantly change with altering the test parameters; merely the likelihood of any test resulting in a particular performance mode is changed. We believe this is a result of the erase block being erased in preparation for writes. We also believe that the different modes represent different numbers of erase blocks being cleared during the test. Increasing the stride makes it more likely that erase block boundaries are being crossed during the test, justifying a larger chance of experiencing the slower 'mode'. Similarly, a longer test also increases the likelihood of crossing more erase block boundaries.

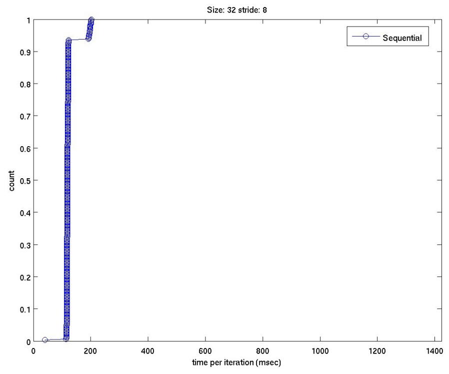
IBM: Clear Bi-modal distribution



IBM: Doubling the stride increases the likelihood of triggering the second mode.

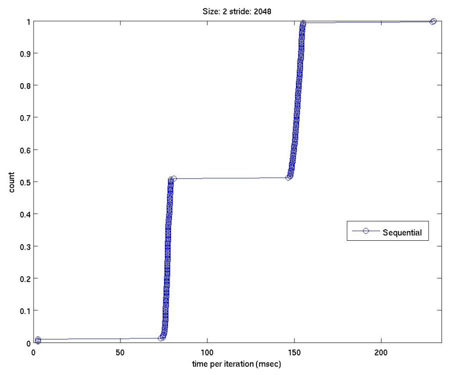


IBM: Doubling the length of the test also increases the likelihood of triggering the second mode.



One graph in particular provides evidence that the erase blocks are 4096 sectors long. This graph shows that when you have 2 blocks, separated by 2048 blocks, you have a ~50% chance of them being in the same erase block and a ~50% chance of crossing an erase block boundary and taking twice as long, exactly as one would expect with an erase block size of 4096.

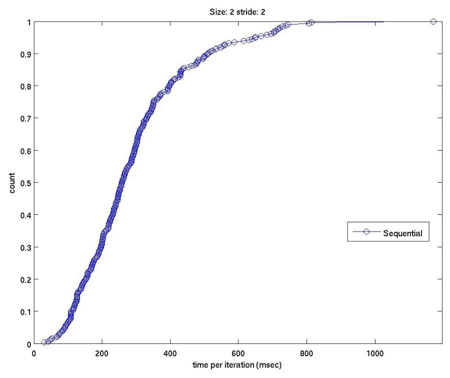
IBM:



4.1.2 Transcend Results

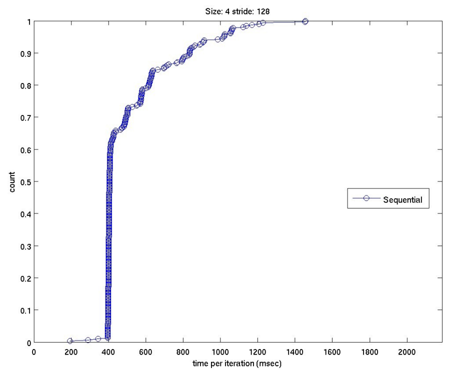
The Transcend continued to provide surprisingly noisy results during the write/read test.

Transcend:



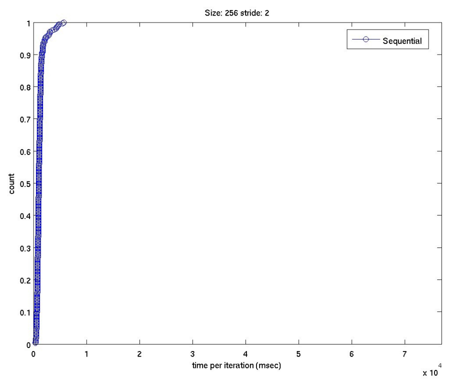
There do appear to be the outliers, as noticed earlier, but the high variance in write times appears to mask the erase block contribution to test completion time. Lack of defined erase block contributions could indicate significantly smaller erase blocks.

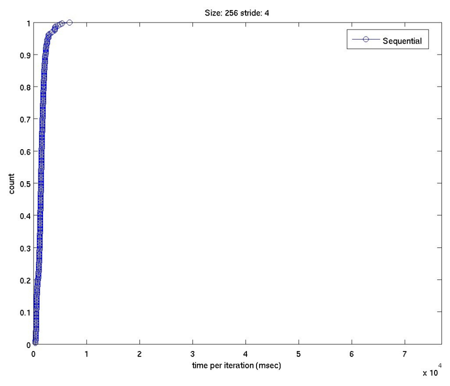
Transcend: The graph with the clearest modal-behavior, length 4 and stride 128, seems to imply that the erase blocks are a multiple of 128. Possibly 384 blocks long, as additional erase block latency appears to occur 1/3rd of the time.



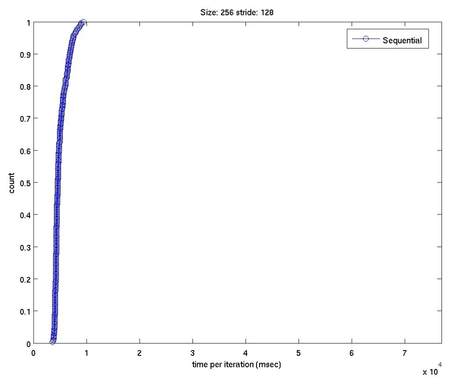
For large enough workloads, the completion time is linear with the stride. Otherwise, completion time is linear with the size. This may be a result of the read buffer becoming saturated after a certain data-range is exceeded within one test.

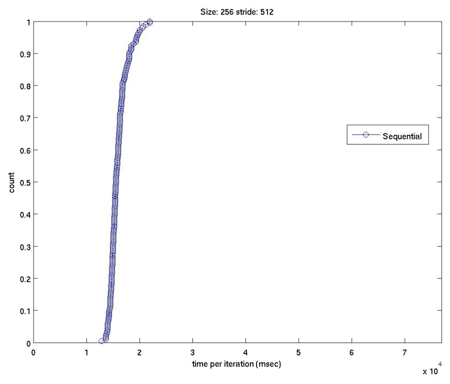
Transcend: Note that at low strides, increasing the stride does not have an appreciable effect on the time to complete a test iteration

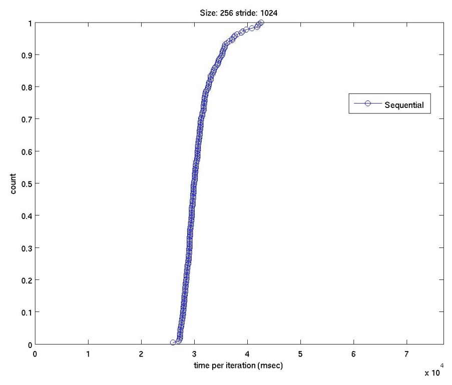




Transcend: However, later changes in stride have a much more noticeable impact on performance.







5 Conclusion

We believe it is clear that while no flash drive will behave much like an HDD, there is significant variation from one drive to the next in terms of performance.

We clearly showed that high-quality Flash SSDs have remarkably consistent behavior under similar workloads, but that occasionally we would see odd performance artifacts we were unable to motivate. Unfortunately, with a technology like Flash, where quite a bit of money will be made, and proprietary technology plays a large role in product performance, companies will be unwilling to provide explanations for the various performance artifacts. Protecting trade secrets will trump consumer understanding, most likely at least until Flash is replaced by whatever will come next.

It is worth noting that the IBM cost almost double what the Transcend cost, despite the Transcend having twice the usable capacity. We believe that, for the foreseeable future at least, the impact of the FTL implementation is strong enough that you will get what you pay for when you buy Flash.

Unfortunately, while Flash SSDs do not suffer the same effects that plague HDDs, they just don't blow HDDs away yet. SDD data transfer rates are slow enough such that there do remain workloads where HDDs can perform better, despite their flaws. We believe that at the moment (though this will probably change in a Flash technology generation or two) Flash will be more attractive when its secondary advantages (physical durability, low power use, etc.) are particularly relevant to the application. When performance per price, or even simply performance alone, is the primary consideration, HDDs still have an edge.

The full set of graphs can be found at:
http://pages.cs.wisc.edu/~hinrichs/736/