SAMSUNG

Performance and Endurance Enhancements with Multi-stream SSDs on Apache Cassandra

A Detailed Analysis

Hingkwan Huen
Changho Choi
Vijay Balakrishnan

Memory Solutions Lab
Samsung Semiconductor, Inc.
January 25, 2017

Executive Summary

SSD popularity continues to soar. Both consumer products and business solutions are embracing SSDs to reap the vast performance, reliability, power consumption and security benefits that the devices provide compared to conventional hard disk drives (HDDs). However, SSDs do have a few limitations. For one, SSDs cannot avoid garbage collection that deterministically causes write amplification. This not only decreases device performance, but also decreases the lifetime of the SSD. Though SSD core flash memory technology improves over generations, garbage collection wear-out remains a significant consideration. Moreover, as NAND flash cells have been shrinking to accommodate SSD capacity increases within the same form factor, SSD endurance and lifetime have been decreasing too.

However, Samsung has introduced multi-stream technology to address this SSD limitation. The multi-stream technology is already standardized in SCSI standard group, T10, and the specification has been publicly released. Samsung's multistream-enabled 12G SAS SSD is one of the first commercially available devices to implement this technology. It is capable of grouping write operations into common data streams, based on application "hints." As a result, it dramatically reduces write amplification, which has long been an undesirable side effect.

This white paper presents a highly efficient use case of Samsung multistream-enabled 12G SAS SSDs functioning within Apache Cassandra, and delineates comparative performance results for different workloads.

When using Samsung multistream-enabled 12G SAS SSD as backend storage for Cassandra, the primary benchmark findings are:

- Cassandra write performance improves up to 300%.
- Cassandra average write latency decreases up to 67%
- SSD WA factor decreases up to 66%, enabling the SSD to last three times longer

Understanding SSD Garbage Collection and Write Amplification

To better understand our findings, it would help to have a very clear understanding of garbage collection (GC) and write amplification (WA). First, data is written to flash memory in units called *pages*. However, flash memory can only be erased in larger units called *blocks* (composed of multiple pages). When data in some block pages are no longer needed (often called stale pages), only pages within the block with valid data are read. Then the valid data is rewritten into a new, previously-erased empty block, making the first block available for erasure and subsequent reuse. This is the process of garbage collection.

Because of these GC data movements, the actual amount of data written into flash memory can be more than the amount that the host system requested – the *WA* phenomenon. WA is typically measured by the ratio of the amount of data committed to the flash memory over the amount of data arriving from the host system. This ratio is commonly known as the *write amplification factor* (*WAF*).

From a host system's perspective, WAF is an excellent indicator of an SSD's write performance. If an SSD has a high WAF, the SSD controller will be required to write extra data to the flash memory, in addition to the amount the host had to transfer for a write request, thereby decreasing performance.

WAF is also a good indicator of SSD endurance. The increased flash memory writes caused by a high WAF wears out the SSD quicker and decreases SSD lifetime.

Multi-stream Overview

SSD flash memory has unique characteristics. Therefore, directly replacing a conventional HDD with an SSD does not exploit the SSD to its full potential. One of the major reasons is SSD's unavoidable GC process. Conventional operating systems (OSs) and applications do not distinguish between hot/cold data or store them differently. In practice, mixing data with different lifespans increases the GC activity necessary to manage and reclaim memory. This affects overall performance and the period of time that an SSD will operate efficiently.

SSD vendors and storage technical committees have already defined the new multistream-enabled SSDs, pioneered by Samsung, to overcome the problems associated with GC. Released in the National Committee on Information Technology Standards' (NCITS) T10 SBC4, revision 9 specification, multi-stream technology provides the OS and applications interfaces needed to write the storage data with data lifespan hints. Multistream-enabled SSDs use these hints to group data internally, effectively reducing GC overhead. The Samsung multi-stream-enabled 12G SAS SSD is a good example of a multistream-enabled SSD that is already commercially available.

For the benchmark presented here, we chose Cassandra as the database since it is widely recognized as being one of the most popular NoSQL databases in the industry. To enable Cassandra to exploit the Samsung multistream-enabled 12G SAS SSD, we needed to modify its source code to allow the Linux kernel's posix_fadvise() system to call in its write path in order to pass hints (i.e., a stream ID) of the associated files. We also patched the Linux kernel to accept the stream ID from posix_fadvise() and issue corresponding stream write commands to the multistream-enabled SSD. This gave Cassandra fine data placement control for all associated files inside the multistream-enabled SSDs based on

expected data lifespan, which results in a more efficient GC process with fewer unnecessary data writes, and a decreased WAF.

Cassandra Overview

Cassandra is well known within the industry as one of the best NoSQL solutions to accommodate the demanding requirements of modern business applications. Its popularity is derived from its outstanding technical features. It is durable, seamlessly scalable, and provides consistency as well as SQL query support. This pervasiveness makes it a good candidate for multi-stream benchmarking.

Leveraging Cassandra's Write Path for Multi-streaming

For storage, the Cassandra write path consists of three different types of write operations:

- Logging data to the commit log

  When a write occurs, Cassandra appends the write to a storage device commit log. The commit log receives every write made to a Cassandra node and these *durable writes* are permanent, even surviving power failures. After the corresponding memtable data flushes to a sorted-string table (i.e., sstable, an in-storage data structure), the commit log data is purged.

- Flushing data from memtable to sstable

  Besides the commit log, Cassandra stores the data in a memory structure, called the memtable. The memtable is a write-back cache of data partitions that Cassandra references using a key. The memtable stores data until reaching a configurable threshold limit, and is then flushed to sstable in a storage device (e.g., SSD).

- Performing background compaction

  As inserts/updates occur, instead of overwriting data, Cassandra writes a new time-stamped version of the inserted or updated data in another sstable. Cassandra manages the accumulation of sstables in storage using compaction. Also, it does not perform in-place deletes because the sstable is immutable. Instead, Cassandra marks data to be deleted, using a tombstone. Tombstones exist for a configured interval defined by the table's gc (grace_seconds) value. In the data compaction process, each sstable is merged by selecting the latest data for storage based on an associated time stamp. Cassandra (Version 3.5.0) supports three different compaction strategies: SizeTieredCompactionStrategy, DateTieredCompactionStrategy, and LeveledCompactionStrategy. For our benchmark, we chose LeveledCompactionStrategy for its higher IO rate.

For log write commits, Cassandra only issues sequential, uniform-size writes. Here, a single stream ID hint for all commit log file writes suffices.

Either the memtable flush or the compaction will generate sstable writes. Sstable writes involve multiple metadata file writes (e.g., bloom filter, sstable indices, etc.) associated with the sstable. These data lifespans differ significantly from DB data. So separation of this data from the DB data will increase SSD performance and endurance. Table 11 identifies a stream ID assignment that leverages the write pattern, size and frequency of each file type within the target workload. It is the same stream ID assignment used in all benchmark configurations. Moreover, we found it optimal for the Samsung multistream-enabled 12G SAS SSD.

Table 1. Stream ID Hint Assignment

Cassandra File Type	Stream ID (Hint Assignment)
Commit log	1
Cache/Cache CRC	2
Compressed meta data	2
Digest checksum	2
Bloom filter	3
sstable statistics	4
sstable indices	5
sstable summaries	6
sstable data	L0:7, L1:8, L2-Ln:9

Benchmark Environment

The test server was a Dell Precision T7810 with dual, 2.40 GHz Intel Xeon E5-2630 v3 processors and 64GB of memory. In total, the server had 16 physical cores. With hyper-threading enabled, the logical CPU count was 32.

For storage, we used a Samsung multistream-enabled 12G SAS SSD that was designed for data center and enterprise applications. The SSD connected to the test server via an LSI Logic SAS3008-based SAS host bus adapter (HBA). Table 2 shows the benchmark system configuration.

Table 2. Server Hardware and OS Configuration

Processor/Memory Details	Operating System	HW Details
Processor Dual Socket: Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz. **Total Logical CPU:** 32 **Total Memory:** 64 GB	**Distro:** Ubuntu 16.04 LTS **Kernel:** 4.4.0-24-generic, patched for multi-stream support **Arch:** x86_64	**SSD:** Samsung multistream-enabled 12G SAS SSD **SAS HBA:** LSI Logic SAS3008 Fusion-MPT SAS-3

The benchmarks compare a Samsung multistream-enabled 12G SAS SSD with the multi-stream feature disabled (denoted as the "legacy" case) to one with the multi-stream feature enabled (denoted as the "multi-stream" case).

Table 3. Software Details

Software	Functionality	Version/Remarks
Cassandra	Persistent Key Value Store	3.5.0 (Modified to add multi-stream support)
cassandra-stress	Cassandra Built-in Benchmark Tool	3.5.0
sg3_utils	Linux SCSI Disk Utilities	1.4.1 (Added tools to manage streams of the multistream-enabled 12G SAS SSD)

The Cassandra server was configured as a single node to simplify setup. Since Cassandra read and write throughput increases linearly with machine scaling, benchmarking a single node allowed for easy estimating of multi-node configuration performance.

The benchmark client (cassandra-stress) was configured to run on the same server. The dataset was initially created by performing one million, 16K key-value record inserts. The same dataset was used for all benchmark runs. To get the Samsung multistream-enabled 12G SAS SSD and database into a steady state, we performed preconditioning with a 100 percent write for four hours. After every pre-conditioning run, the Cassandra server was idled to allow sufficient compaction to complete all remaining tasks. Benchmark workloads were run after this pre-conditioning.

Benchmark workloads consisted of 100 percent write, 50 percent/50 percent read/write mix, and 70 percent/30 percent read/write mix. For performance metric comparisons, we measured throughput, latency and WAF.

Table 4. Benchmark Configuration Details

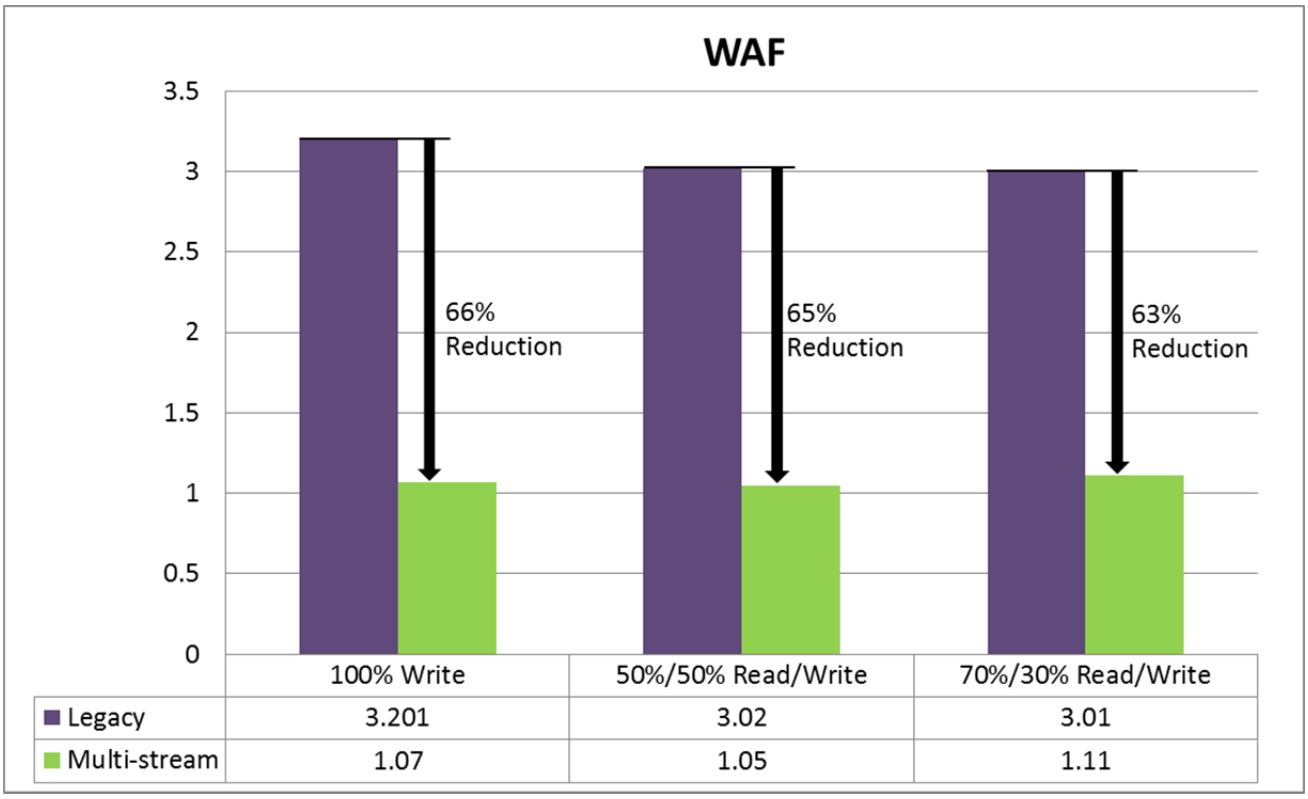
Parameter	Value
Number of Operations	1 Million
Schema	1024 bytes x 16 columns LeveledCompactionStrategy
Number of Client Threads	200
concurrent_reads	32
concurrent_writes	128
memtable_flush_writer	8
concurrent_compactors	8
compaction_throughput_mb_per_sec	64

Benchmark Results – WAF

Multi-stream directly benefits the WAF. Because data with similar lifespans have been grouped into common write streams, this allows the SSD to efficiently place data that belong together into common erase blocks. This way, the chance of potential data mix has been minimized. This mix of data with different lifespans within the same erase block is the main culprit for heavy GC work inside the SSD. We found that the less work the GC needs to do, the more WAF can be reduced.

As illustrated in Figure 1, there is significant WAF reduction across all three benchmark cases. As was expected, the more extensive the write portion, the more WAF is reduced.

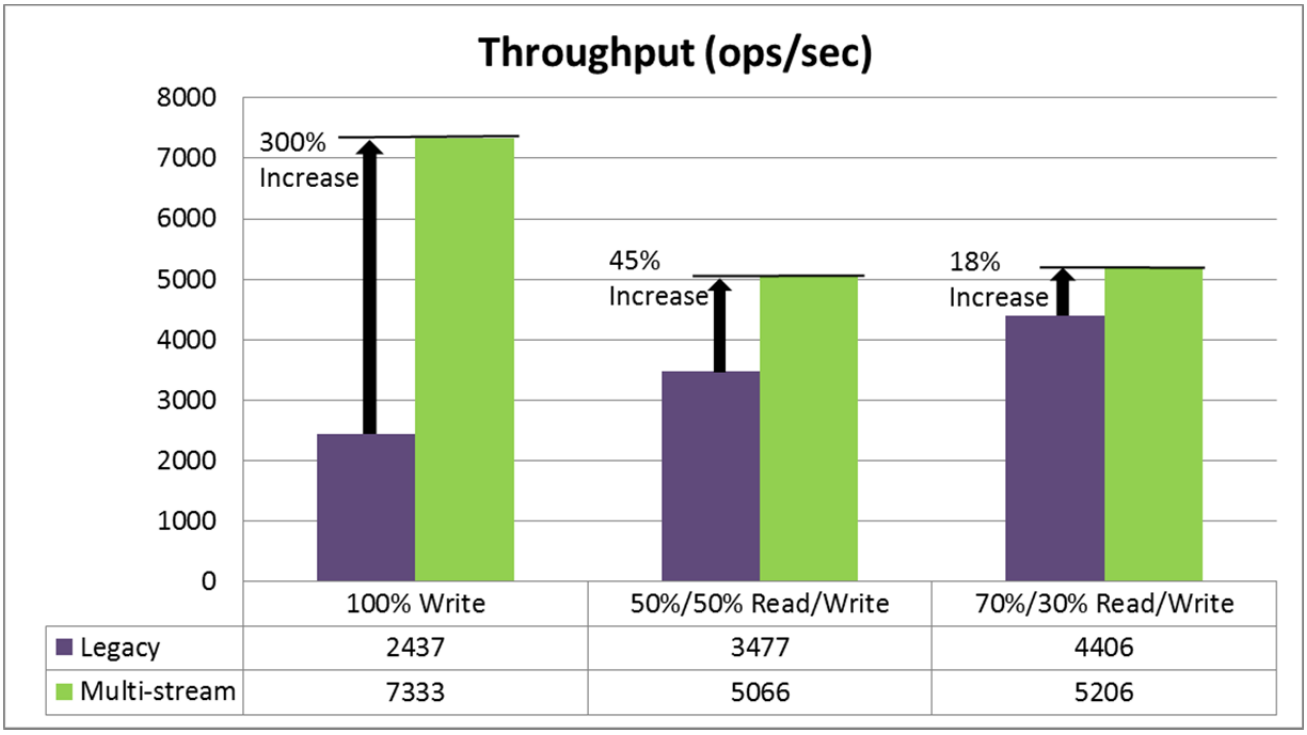
Figure 1. WAF Result



Benchmark Results – Throughput

Because multi-stream minimizes the need for extra data copy during GC operations, the SSD can service much higher throughput. Figure 2 shows the improvement in throughput across all three cases. Similar to WAF improvements, the more extensive the write portion, the higher the throughput.

Figure 2. Throughput Result



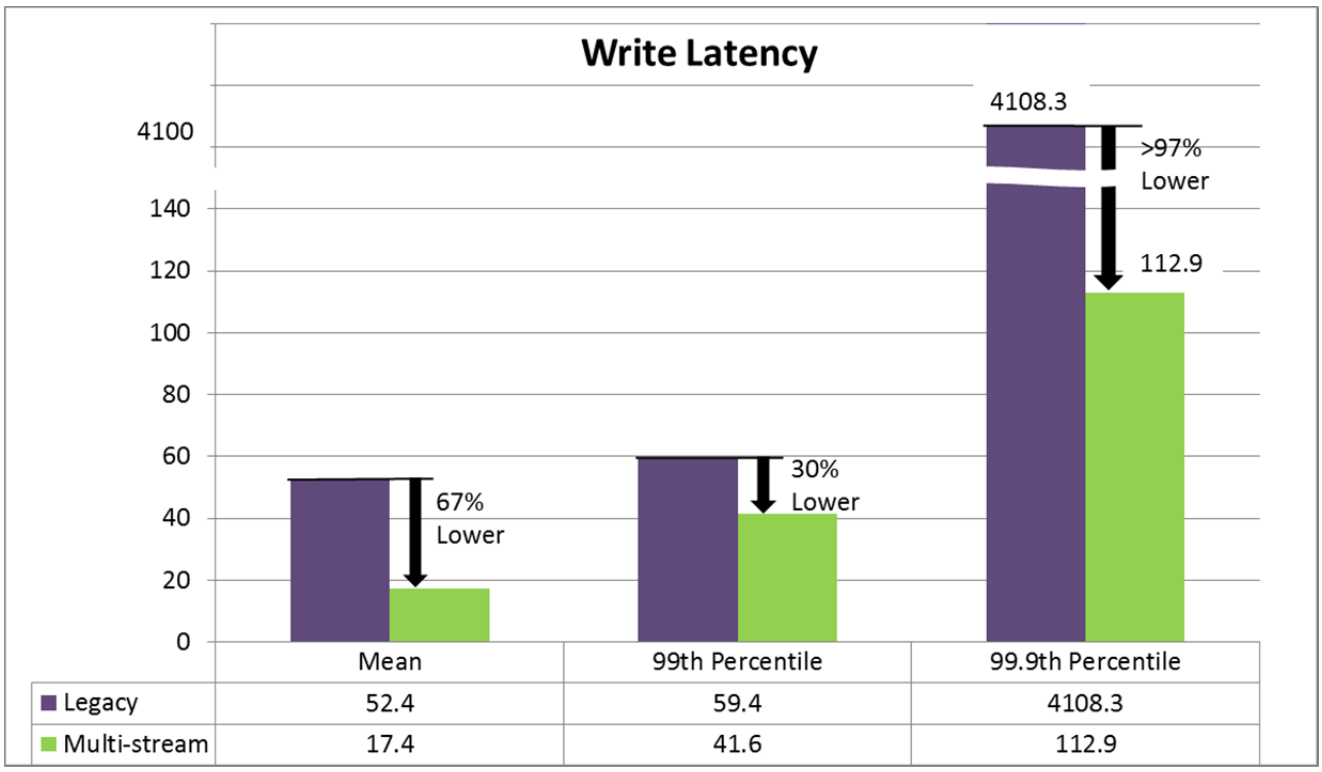
	100% Write	50%/50% Read/Write	70%/30% Read/Write
Legacy	2437	3477	4406
Multi-stream	7333	5066	5206

Benchmark Results – Write Latency

At this time, multi-stream is mainly applied to write requests. Therefore, we measured write latency in a 100 percent write workload case.

Figure 3 summarizes the latency measurement result for the Cassandra writes. Multistream-enabled SSD average latency is only one third of the legacy SSD average latency. Tail latency is a major interest of Cassandra users. As shown in Figure 3, the 99.9th percentile latency of the multistream-enabled SSD is less than 3 percent of the legacy alternative. The multistream-enabled SSD's tail latency improvement can mostly be attributed to minimization of GC overhead.

Figure 3. Write Latency Summary



Conclusion

Write amplification is an inevitable SSD limitation. It decreases an SSD's raw performance and lifetime. Multi-stream technology, which is already incorporated in the T10 SBC4 revision 9 standard specification, addresses this limitation by enabling SSDs to exploit data lifespan hints from the host. This allows SSDs to provide smart and efficient internal data placement, dramatically reducing write amplification.

As this paper conveys, when Samsung multistream-enabled 12G SAS SSDs are used as Cassandra backend storage, the multi-stream technology has the potential to provide significantly greater performance with:

- Up to 300% higher write throughput
- Up to 67% lower average write latency
- Up to 66% reduction in write amplification, thereby increasing the SSD lifetime by 300%

###