SSD performance measurement: Best practices

Doug Rollins - August 23, 2013

This article takes you through the types of measurements and tools you need to measure enterprise SSD performance.

SSDs have pronounced write-history sensitivity, which means that they have unique requirements for accurately measuring their performance. This article documents how to measure enterprise SSD performance, with an emphasis on the following points:

• Main goals of performance measurement (accuracy, consistency, and repeatability)
• Key assumptions about enterprise SSD requirements
• Definitions of commonly used terms
• Tools used
• Metrics of primary interest
• Test mechanics, including:
  o Preconditioning
  o Using a common measurement tool to construct a write-saturation plot

This article cites empirical data demonstrating the main NAND-based SSD performance regions—FOB (fresh-out-of-box), transition, and steady state—and shows an example of the markedly different results that can be achieved (on the same drive) depending on the order in which the stimuli are applied. It also shows alignment to the Storage Network Industry Association (SNIA)’s Performance Test Specification protocol, provides explicit details on our precondition and measurement techniques (along with an example), and shows the relevance of the collected metrics.

Enterprise SSD performance measurement is concerned primarily with the following standards:
• Accuracy
• Consistency
• Repeatability

It is based on some key assumptions about enterprise SSDs and the enterprise market—specifically:
• Drive fill state
• Access and I/O traffic patterns
• Decision criteria (for purchase and deployment)

Accuracy, Consistency, Repeatability
The testing used is designed to minimize the impact of external elements. Specifically, minimizing the influence of the operating system, host configuration, and target variables to enable reliable, relative comparisons.
To do this, the tests discussed here use an invariant server-grade operating system and apply only predetermined service packs and hotfixes. Automatic patch installation is disabled, target cache options are set manually, and all host components have the same BIOS settings, firmware version, and drivers installed. This ensures consistent run-to-run, host-to-host, and drive-to-drive measurement consistency. For purposes of the tests discussed, drives are always put in the same state at the beginning of each measurement, preconditioned the same way for each measurement, and stimulated to the same performance state—the test process is deterministic.

Assumptions
When measuring performance for enterprise SSDs, it is assumed that the requirements are very different from those of client SSDs.

Enterprise SSD Performance Measurement Assumptions
• Drive fill state: The drive is always 100% full.
• Accesses: It is being accessed 100% of the time (that is, the drive gets no interface idle time).
• Decisions: The enterprise market chooses enterprise SSDs based on their performance in steady state, and that steady state, full, and worst case are not the same thing.
• Consequences of failure: Failure is catastrophic for multiple users.

Client SSD Performance Measurement Assumptions
• Drive fill state: The drive has less than 50% of its user space occupied.
• Accesses: It is accessed a maximum of 8 hours a day, 5 days a week (but typically is written much, much less frequently).
• Decisions: The client market chooses enterprise SSDs based on their performance in the FOB state.
• Consequences of failure: Failure is catastrophic for a single user.

Furthermore, it is assumed that the focus of enterprise SSD performance measurement should include the intended workload. For example, an enterprise SSD intended to accelerate a database should be measured at least using a typical database workload (typically 8K transfer size; random; two-thirds read, one-third write traffic; fully loaded queue); if the same drive is intended to be a cache for streaming files, it should be measured with that workload (larger blocks, read-intensive, fully loaded queue).

Performance States
Because SSD performance can change as it is written, enterprise performance measurement focuses on the steady-state performance region. The following definitions are used in measurements.

This paper assumes a complex definition of steady state, from the SNIA Solid State Storage Initiative’s Performance Test Specification:

• Max(y) - Min(y) within the measurement window is no more than 20% of the Ave(y) within the measurement window, AND
• [Max(y) as defined by the linear curve fit of the data with the measurement window] - [Max(y) as defined by the linear curve fit of the data with the measurement window] is within 10% of the average within the measurement window.

A full drive is one that has been overwritten some multiple (could be 1X) of the user-accessible LBA space by a fixed pattern that may vary from the test stimulus (i.e., 2X user LBA space written sequentially with 128K transfers).
Worst case performance is when the drive has been stimulated over some fixed time with a workload intentionally designed to demonstrate the drive’s worst possible performance. For example, this type of stimulus may include (but is not limited to) small transfers mixed with large transfers and intentionally misaligned writes.

**Test Sequence**
The mechanics of enterprise SSD performance measurement discussed in this paper include:

![Figure 1: Test-flow sequence](image)

**Purge:** Regardless of what has been done previously to the SSD, put the drive in a known, fixed state that emulates the state in which the drive would be received from the manufacturer. This is the fresh-out-of-box (FOB) state. This paper refers to using the secure erase command to place sample SATA SSDs in this state; other methods may be protocol- or vendor-specific.

**Precondition:** Following SNIA’s Performance Test Specification for workload-independent precondition; writes the drive with 128KB sequential transfers aligned to 4K boundaries.

**Test:** For any metric of interest, test only one metric at a time, and test into steady state.

**Collect & Report:** At the end of the test run, results are compiled, drive performance from FOB to steady state is analyzed, and then steady-state performance values are recorded.

**Setting Up the Test Platform**
This section gives an example of host configuration using Microsoft Windows Server 2008; other operating systems should be treated similarly.

To ensure run-to-run consistency, fix a service pack and patch level, and then disable the operating system from installing updates because automatic installation may cause an automatic system restart, without regard to a test that may be running, which can result in run-to-run variance. Note: This setting may not be in accordance with your personal or corporate security policy. Micron assumes no responsibility for any consequences of this setting or its adjustment. This information is provided for reference only. Because most SSD performance measurement is done on an isolated system, this may be less of an issue, but you should know and understand your corporate security policy.
Next, you need to ensure that the SSD is recognized. First, open the Disk Manager, locate the SSD to be tested, and then check that the drive does not contain a partition. Then, although the disk number assigned to the SSD may vary, ensure that the SSD is visible, as shown in the center of Figure 3. You may have to mark the SSD as online manually, and you may also have to initialize the SSD. (Note: Initializing the SSD is not the same as formatting it. You should see a prompt for initializing the disk. Click Yes, and then select the MBR option.)
Figure 3: Locate the SSD and check partitioning

If the SSD being tested does not support a complete power-backup solution (all volatile areas protected by an internal power-backup mechanism), then, for purposes of this discussion, the write cache option is disabled, as shown in Figure 4.
Figure 4: Disable write caching Mechanics: Test Setup and Execution - Benchmark Configuration

Mechanics: Test Setup and Execution - Benchmark Configuration
This section gives an example of host configuration using IOMeter® (an I/O subsystem measurement and characterization tool for single and clustered systems) as the benchmark. First, you need to launch IOMeter, which is typically done from the Start menu, but you may have to browse to the folder where it is stored and launch it using its icon.

Set Up the Precondition Test Specification in IOMeter
As noted before, this paper assumes use of the SNIA SSSI TWG PTS workload-independent precondition of 128K sequential traffic aligned on 4K boundaries to write twice the user capacity. The same precondition is applied to all devices tested.
Figure 5: Precondition test specification settings

1. Set the transfer size to 128K.
2. Set the percent random/sequential distribution to 100% sequential.
3. Set the percent read/write distribution to 100% write.
4. Align the I/O on the 4K page boundaries.

As a general rule, use a run time of five minutes per iteration, giving a time-axis granularity of five minutes in the performance vs. needed time plot. In addition, the queue depth is always set to the maximum supported by the SSD to ensure that it is written at the fastest rate possible. The Cycling Options section is set to fix the queue depth (no cycling) as shown in Figure 6. The 128K sequential write precondition access specification is now configured.
Determining the Precondition Time

IOMeter tests are based on time, not the number of GB written. Because SSD performance varies substantially, the process outlined in this paper does not fix a time to run the precondition. Instead, the following steps are taken:

1. Add the precondition access specification to the test run 25 (or more) times.

2. Note that the precondition access specification dwell time is set to five minutes, and the queue depth to the maximum that the device supports.

3. Execute the precondition test.

4. This test shows the SSD write performance from FOB to steady state for the precondition access specification. Examine the results to determine the steady state write rate for the precondition test.

5. Divide the drive user capacity by the steady state write rate to determine the time necessary to write the drive once (with 128K sequential traffic).

6. Double the time in step 5 to ensure that the drive is written twice.

7. Use the duration from step 6 for all future preconditioning for this drive.

Note that the time to precondition a drive may be unique and may vary from model to model, or from manufacturer to manufacturer. After the duration of the precondition has been determined for a specific drive make and model, it remains constant—but only for that make and model. The precondition duration must be determined for every drive to be tested and must be applied identically for all tests on that drive.

The number of times that the precondition access specification is added to the precondition run determines the number of values that can be shown on the time axis of a performance-vs.-time plot;
the duration of the precondition access specification determines the time between those time axis values.

To create the precondition run, start by adding ONLY the precondition access specification to the test run (general rule of thumb: add the access specification 25 times as a starting value) as shown in Figure 7.

Figure 7: Add precondition access specification
After the precondition access specification is added to the test run 25 (or more) times, execute the run, and then save the results.

Mechanics: Test Setup and Execution - Establishing Preconditioning
This section shows how the data collected in the benchmark configuration process is used to determine the preconditioning duration.

After parsing the resulting .csv file (via Microsoft Excel® or some other method) to remove the extraneous data, the results should look similar to Figure 8.
Because this phase of the test is concerned only with establishing the correct precondition duration, the only metric of interest in the data set in Figure 8 is the write MB/s. If this data is plotted, the result is shown in Figure 9.

From the above plot and from the definition of steady state, this example drive has a steady state...
write rate of about 300 MB/s.

Finally, the steady state write rate for 128K sequential, page-aligned traffic (with a queue depth of 32) sets the duration for all subsequent preconditioning.

The time to write this sample drive twice is shown in Table 1.

<table>
<thead>
<tr>
<th>Drive capacity</th>
<th>100GB = 102,400MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write rate</td>
<td>300 MB/s</td>
</tr>
<tr>
<td>Minutes to fill (1x)</td>
<td>6</td>
</tr>
<tr>
<td>Minutes to fill (2x)</td>
<td>12</td>
</tr>
<tr>
<td>Precondition time</td>
<td>12 minutes</td>
</tr>
</tbody>
</table>

Table 1: Precondition calculations

Note that this scenario is the time for the drive to go from a purged state into a preconditioned state. This can be different for different drives, different media, and different firmware revisions. It is consistent for each drive under test.

Mechanics: Test Setup and Execution - Set Up the Test Metric of Interest
With the precondition access specification and duration established, the access specification for each metric of interest needs to be set up. Note that the test flow is similar to previous steps, with only a few differences.

An example of an access specification to measure 4K random writes on an SSD with 4K pages and a maximum queue depth of 32 is shown in Figure 10.

Figure 10: Set access specifications for selected metric

1. The transfer size is set to 4K, and the percent random/sequential to 100% random.

2. Because the metric of interest is 100% write, the percent read/write distribution slider is set to 100% write.

3. Finally, all I/Os are aligned to 4K.

4. As before, the run time is set to five minutes, and the queue depth is set to the desired value for the test. (Note: in general, do not vary the queue depth per test, as it can make data compilation and analysis more difficult. Instead, repeat the test sequence for each queue depth of interest.)
5. Configure the remainder of the test-run settings as shown in Figure 12.
Table 2: Micron’s starting point for SLC and MLC iterations

<table>
<thead>
<tr>
<th>Type</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLC</td>
<td>25</td>
</tr>
<tr>
<td>MLC</td>
<td>75–100</td>
</tr>
</tbody>
</table>

Note that the total time to reach steady-state performance for small block random write traffic varies widely. Typically, SLC-based drives do so more quickly than MLC-based drives.

This example is for an SLC drive, so the 4K random write access specification is added 25 times, and the test run is executed as a starting baseline value. The test is executed, and the results are parsed and plotted, as shown in Figure 13.

Figure 13: SLC 4K random write saturation in steady state

In this example, steady state is achieved after the fourth test iteration. As each x-axis mark represents five minutes, this drive reaches 4K random write steady state in 20 minutes.
Mechanics: Test Setup and Execution - Workflow Overview

The overall workflow consists of:

1. Regardless of the metric being measured, each drive to be tested is put into the FOB state. This may be done via the secure erase command, a vendor-unique method, a serial port, or some other means.

2. The drive is then preconditioned with the SNIA PTS workload-independent preconditions.

3. A single, fixed stimulus is measured into the steady state region.

4. Results are then parsed and plotted; steady state performance is validated and reported.

5. If additional stimuli are of interest, the process is repeated for each additional stimuli; otherwise, the process is complete.

Example: Stimulus-Sequence Variant Performance

Throughout this paper, it is asserted that returning the SSD to the FOB state is key to precise, consistent performance measurement. In this section, an example is shown how not doing so and then simply changing the order in which the stimuli are applied can give vastly different results.

This example is an MLC intended for use in the client market, but which has been well-adopted in some enterprise applications.

<table>
<thead>
<tr>
<th>Element</th>
<th>Transfer Size</th>
<th>% Read</th>
<th>% Random</th>
<th>IOPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>35460.98405</td>
</tr>
<tr>
<td>2</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>17382.48189</td>
</tr>
<tr>
<td>3</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>20179.53067</td>
</tr>
<tr>
<td>4</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19460.9479</td>
</tr>
<tr>
<td>5</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19568.64219</td>
</tr>
<tr>
<td>6</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19657.92614</td>
</tr>
<tr>
<td>7</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19573.7974</td>
</tr>
<tr>
<td>8</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19561.21846</td>
</tr>
<tr>
<td>9</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19572.46237</td>
</tr>
<tr>
<td>10</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19597.42278</td>
</tr>
<tr>
<td>11</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19577.95913</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19595.8534</td>
</tr>
<tr>
<td>13</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19442.13415</td>
</tr>
<tr>
<td>14</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>18986.06345</td>
</tr>
<tr>
<td>15</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19065.45231</td>
</tr>
<tr>
<td>16</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>18993.24672</td>
</tr>
<tr>
<td>17</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19016.87946</td>
</tr>
<tr>
<td>18</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19011.96523</td>
</tr>
<tr>
<td>19</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19024.61239</td>
</tr>
<tr>
<td>20</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>19645.18573</td>
</tr>
<tr>
<td>21</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>18990.85739</td>
</tr>
<tr>
<td>22</td>
<td>4096</td>
<td>0</td>
<td>100</td>
<td>18997.49995</td>
</tr>
</tbody>
</table>

Table 3: Raw IOMeter data from Figure 13 plot
The plot in Figure 15 shows steady-state performance for the drive when the stimulus sequence applied is: 4K random write into steady state, then 128K sequential write into steady state, followed by 4K random write (again into steady state). The performance for each stimulus is plotted on the y-axis, with time plotted on the x-axis.

As expected, the IOPs for the 128K sequential write traffic is lower than the IOPs for the 4K random write traffic. Note that the drive’s performance for each 4K random write interval is consistent. Note also that the drive was NOT returned to the FOB state at any point during the test—each stimulus was applied immediately following the completion of the previous stimulus.

Next, the same drive was secure-erased to FOB state and the same stimulus applied, but their order was reversed. In this next plot, time is on the x-axis, performance (in IOPs) on the y-axis, but the test stimulus sequence was changed to: 128K sequential write into steady state, then 4K random write into steady state, followed by 128K sequential write (again into steady state). Based on the test data above, it would be reasonable to expect that the performance of the two 128K sequential write intervals would be the same; however, the results were quite different, as shown in Figure 16.
The performance seen in the second 128K sequential write interval, despite being written into steady state, is lower than what is seen in the first 128K write interval. In this example, the order in which the stimuli were applied was the only variable, but the measured performance was quite different. This drive clearly demonstrates the write-history sensitivity mentioned at the beginning of this brief. If the drive had been returned to FOB in-between these tests, this variance would not have been observed, and the accuracy of the test results would have increased.

**Conclusions**
The enterprise SSD performance measurement technique discussed in this paper ensures consistent, repeatable performance measurements relevant to the enterprise customer. Through a combination of preconditioning and single-stimulus testing, run-to-run and write-history-based performance fluctuation is eliminated. The method is in alignment with current industry test methods and provides a level playing field for all drives measured.

**About the Author**
Profile for Doug Rollins.

**Also See:**
Understanding SSD overprovisioning
The problem with memory test
How to test high-speed memory with non-intrusive embedded instruments

If you liked this feature, and would like to see a collection of related features delivered directly to your inbox, sign up for the Test & Measurement newsletter here.