# SSDs... heaven or hell?

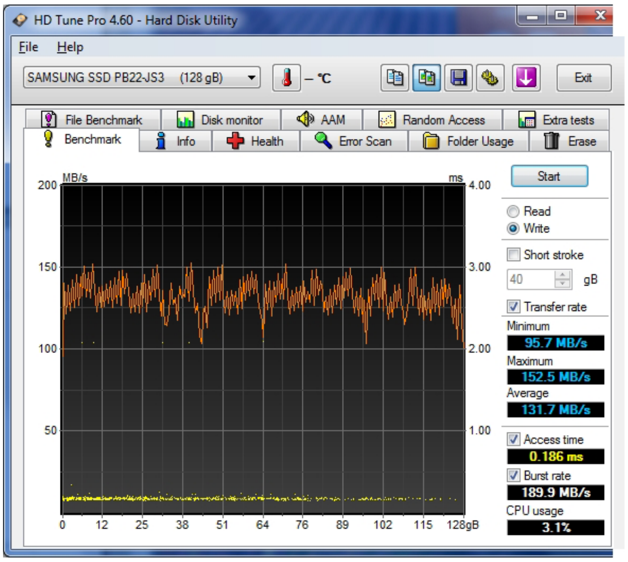
## The pros and cons of using NAND flash memory

**Ivar Poijes**
*Swedish Radio*

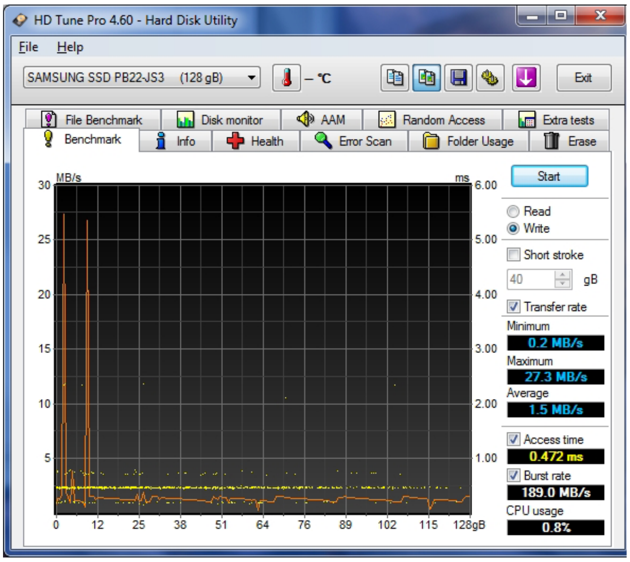
**This article covers the most important characteristics of flash-memory Solid State Drives and some of the more painful pitfalls you might discover when, in good faith, you start trusting the promises made by vendors and manufacturers.**

While the marketing of flash-memory Solid-State Drives (SSDs) offers you performance-heaven you might as easily end up in disappointment-hell. To learn how to hover safely between the heavenly but unreachable performance you read about in the ads and the hellish disasters people like the author have experienced – you have to learn about the pros and cons of using NAND-based flash-memory SSDs.

An example from an SSD implementation that started as a heavenly success and ended in the total opposite about a year later, is shown in *Figs 1 and 2*. The performance graphs show the write performance with two different workloads – sequential writes and non-sequential writes – on an SSD that has been running in a Windows XP audio workstation in an OB vehicle for just over a year. Every conventional disk test (until I tried non-sequential or random writes direct to the unpartitioned disk) told that the SSD worked fine – but it caused BIG spooky problems in the audio workstation.



SAMSUNG SSD PB22-JS3 (128 gB)
– °C
○ Read
● Write
☐ Short stroke
40 gB
☑ Transfer rate
Minimum 95.7 MB/s
Maximum 152.5 MB/s
Average 131.7 MB/s
☑ Access time 0.186 ms
☑ Burst rate 189.9 MB/s
CPU usage 3.1%



SAMSUNG SSD PB22-JS3 (128 gB)
– °C
○ Read
● Write
☐ Short stroke
40 gB
☑ Transfer rate
Minimum 0.2 MB/s
Maximum 27.3 MB/s
Average 1.5 MB/s
☑ Access time 0.472 ms
☑ Burst rate 189.0 MB/s
CPU usage 0.8%

**Figure 1**
**SSD write performance – sequential**
**Average transfer rate = 131.7 MByte/s**

**Figure 2**
**SSD write performance – non-sequential**
**Average transfer rate = 1.5 MByte/s**

The reasons why this SSD implementation failed will be explained at the end of this article – probably a good lesson for others to learn from too!

## HDD vs SSD

To start with, it's important to gain an understanding of the basic differences between a conventional Hard Disk Drive (HDD) and a Solid State Drive (SSD).

### *HDDs*

In an HDD the information is stored as magnetic patterns on spinning disks. The information is written and read by magnetic heads on moving swing arms. The position of the magnetic head and the rotation of the disk must be synchronized when data is written or read.

An HDD is a delicate electromechanical-electromagnetic construction, perfected during more than half a century. The first HDD was actually made in 1956 by IBM. It could store 5 MB on fifty 24-inch disks.

An HDD includes a lot of internal housekeeping functions to ensure reliability and data integrity. A higher level of HDD housekeeping is implemented in dedicated disk controllers in servers or storage systems.

Accessing the data on a modern HDD is usually done in tens of milliseconds, depending on the access pattern. Since the information is stored on rotating disks, the transfer rate decreases in a predictable way, the closer the data is located to the centre of the disk.

### *SSDs*

In an SSD the information is stored in memory chips. Depending on the technology used, the chips have different characteristics and are organized in different ways. An SSD with NAND flash memory chips, needs its own specific housekeeping to maintain performance, endurance and data retention. Just like the HDD, an SSD has some internal housekeeping processes and will benefit from the added functionality that a dedicated SSD controller can offer.

Accessing the data on a healthy SSD can be done a hundred times faster than on the average HDD.

The transfer rate, especially for writes, will decrease over time as the SSD gets overwritten with new data. From a fresh-out-of-the-box state, the transfer rate will finally stabilize at a much lower level – typically as low as 20 - 40 % depending on the I/O-pattern. This will be highlighted in more detail later in this article.

### *NAND-based flash SSDs*

The bright new world of NAND-based flash SSDs is ruled by concepts such as *program-erase cycles*, *write amplification*, *garbage collection*, *TRIM*, *wear levelling* and *over provisioning*. A better understanding of these concepts, and a few others, will bring a better understanding of when and why an SSD is the best – or the worst – choice for you.

## Two types – SLC and MLC

We have two main types of flash memories – Single-Level Cell (SLC) and Multi-Level Cell (MLC):

- SLC stores one bit in every memory cell;
- MLC stores two bits per memory cell – which would make “Dual-Level Cell” a better name for it, especially since the next step in MLC development will be Triple-Level Cells (TLC – coming soon) which can store three bits per cell.

The main differences between SLC and MLC are price, capacity and endurance. An SSD based on SLC technology is generally more expensive and has lower storage capacity, but also has about ten times longer endurance than an MLC drive. The concept of endurance in this case is represented by the statistical number of times a memory cell can be written and erased before bit errors (caused by memory cell wear-out) occur.

- An SLC memory cell can (statistically) last over at least 100’000 Program/Erase (P/E) cycles before bit errors become a risk (the P/E cycle and wear-out problems are described later).
- An MLC memory cell will last at least 10’000 P/E cycles.

## Two other types – Consumer Class and Enterprise Class

Two less distinctive types off SSDs are those made for ordinary consumer use and those made for professional enterprise use. For SSDs, as for everything else, there are important quality differences between low-cost consumer-class equipment and equipment designed for professional 24/7 enterprise use.

It can be very tempting to “save money” by using cheaper – and on the spec sheet, almost identical – consumer-class disks instead of much more expensive enterprise disks. For professional use this is rarely a choice that really saves money.

**Some typical characteristics of enterprise-class SSDs are:**

- Higher quality and longer life cycle,
- More advanced internal architecture and housekeeping functionality;
- Higher cost per GB – but still low cost per I/O compared to HDD;
- Are mostly SLC, but can also be MLC designed for high endurance;
- Are aimed for servers, SAN storage etc. with dedicated SSD controllers;
- Can handle very high I/O;
- Typically comes in a 2.5” HDD form factor, or a custom form factor.

**Some typical characteristics of consumer-class SSDs are:**

- Consumer quality – shorter life cycle (MLC 1/10 of SLC);
- Lower cost per GB, larger disks – compared to SLC;

### Abbreviations

| | | | |
|---|---|---|---|
| **CBE** | Corrected Bit Errors | **RAID** | Redundant Array of Independent Disks |
| **ECC** | Error-Correction Code | **RAM** | Random-Access Memory |
| **FOB** | Fresh Out of the Box | **RBE** | Raw Bit Errors |
| **HDD** | Hard Disk Drive | **SAN** | Storage Area Network |
| **I/O** | Input/Output | **SLC** | Single-Level Cell |
| **MLC** | Multi-Level Cell | **SNIA** | Storage Networking Industry Association |
| **NAND** | Logical “Not AND” | **SSD** | Solid-State Drive |
| **OB** | Outside Broadcast | **SSS** | Solid-State Storage |
| **OS** | Operating System | **TLC** | Triple-Level Cell |
| **P/E** | Program/Erase | **UBE** | Uncorrected Bit Errors |

- Are mostly MLC, but can also be SLC for higher endurance than MLC;
- Aimed for laptops, PCs, “surfpads”, cell phones etc.;
- Usually 1.8” and 2.5” HDD form factor, or a custom form factor.

## The Program/Erase Cycle

The NAND-based flash memory cell is a type of semiconductor that can store and keep charges in “the floating gate” – see *Fig. 3*. To (re)program a memory cell – i.e. write data to it – the existing data needs to be erased before the cell can be programmed. In an HDD, the data can be overwritten without any need for previous erase – which is a big difference between SSD and HDD.

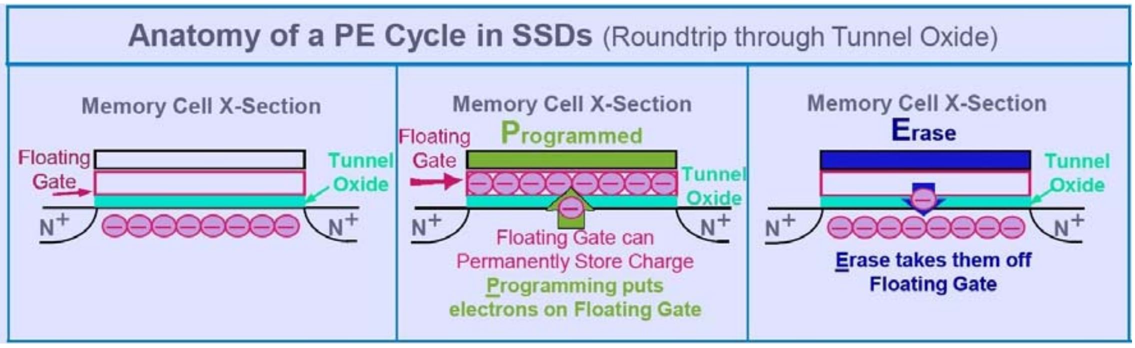


**Figure 3**
**PE Cycle in an SLC memory cell**

After over 100’000 Program/Erase cycles, the tunnel oxide in the SLC memory cell becomes gradually damaged, which sooner or later will cause single-bit errors. A single-bit error is when a digital one (1) appears as a zero (0) – or vice versa.

For a typical MLC memory cell, the tunnel oxide will last about 10’000 Program/Erase cycles before it starts to get damaged. To reduce the wear on single cells, a *wear levelling algorithm* in the SSD tries to spread the writes evenly among all the memory cells. To reduce the impact of bit errors, the SSDs also use *internal error correction* (ECC).

## Wear levelling

Wear levelling is a technique to compensate for P/E cycle wear – by distributing data evenly over the cells in the SSD. By avoiding “Hot Spots” (i.e. many repeated writes on the same few cells) on the SSD media, no cells will be worn out prematurely in the life cycle of the device.

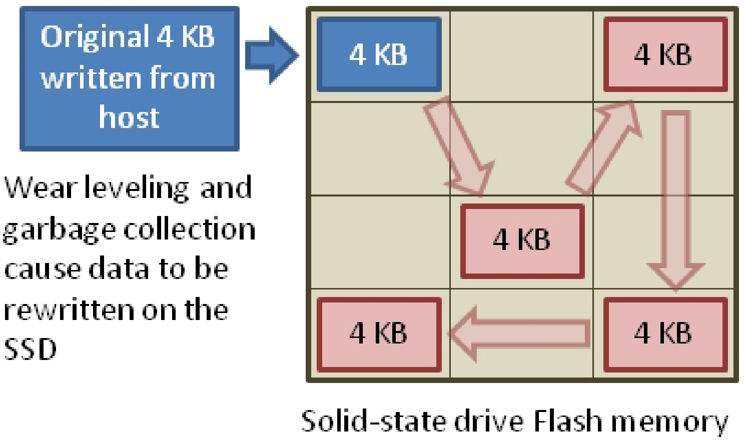


**Figure 4**
**Wear levelling rearranges data as shown by Wikipedia**

Wear levelling can also re-arrange already-written data between cells (pages) to further prolong the total life cycle of the SSD.

Depending on the type of workload (write pattern) and the amount of unused space on the drive, the wear levelling can be just an unnoticed transparent background process –

or actually decrease write performance significantly.

A way to ensure extra free space on the SSD – to minimize negative performance impact – is *over provisioning*, which means that the SSD contains hidden spare memory cells that can be used by the wear-levelling algorithm.

### *Write amplification*

In a flash SSD, data is organized in *pages* and *blocks*:

- A *page* is 4 kByte of data;
- 64 pages will form a 256 kByte *block*;
- Writing to an SSD is done by writing whole 4 kByte pages;
- To write or change just a bit or a byte, a whole 4 kByte page must be written;
- Erasing data in an SSD is done by erasing whole 256 kByte blocks.

As mentioned earlier, it is necessary to erase a used cell before it can be reprogrammed. The internal architecture of an SSD can't erase a single cell or a single page, so modifying/writing data will initiate the *Read-Erase-Modify-Write Process* in the SSD.

**To modify data in a specific 4 kByte page, we need to:**

- Read the whole 256 kByte block including the specific page (and the 63 other pages) to cache;
- Erase the 256 kByte block on the SSD;
- Modify the specific 4 kByte page in cache;
- Write the modified 256 kByte block back to the SSD.

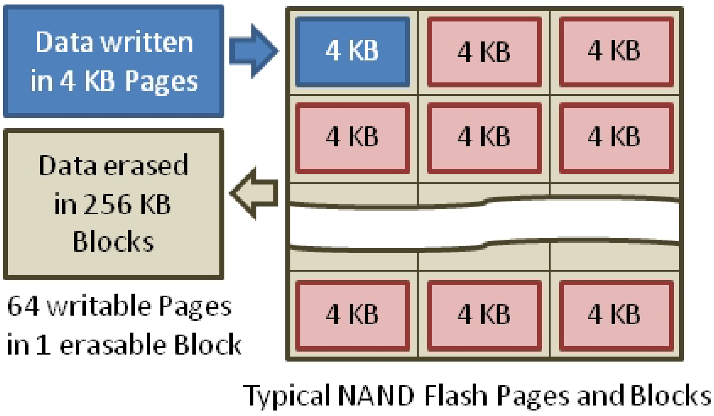


**Figure 5**
**SSD pages and blocks (from Wikipedia)**

The *Read-Erase-Modify-Write Process* in the SSD will of course interact with the wear-levelling algorithm(s), so step 4 above will most likely write data to different physical memory cells than where step 1 first read the data from.

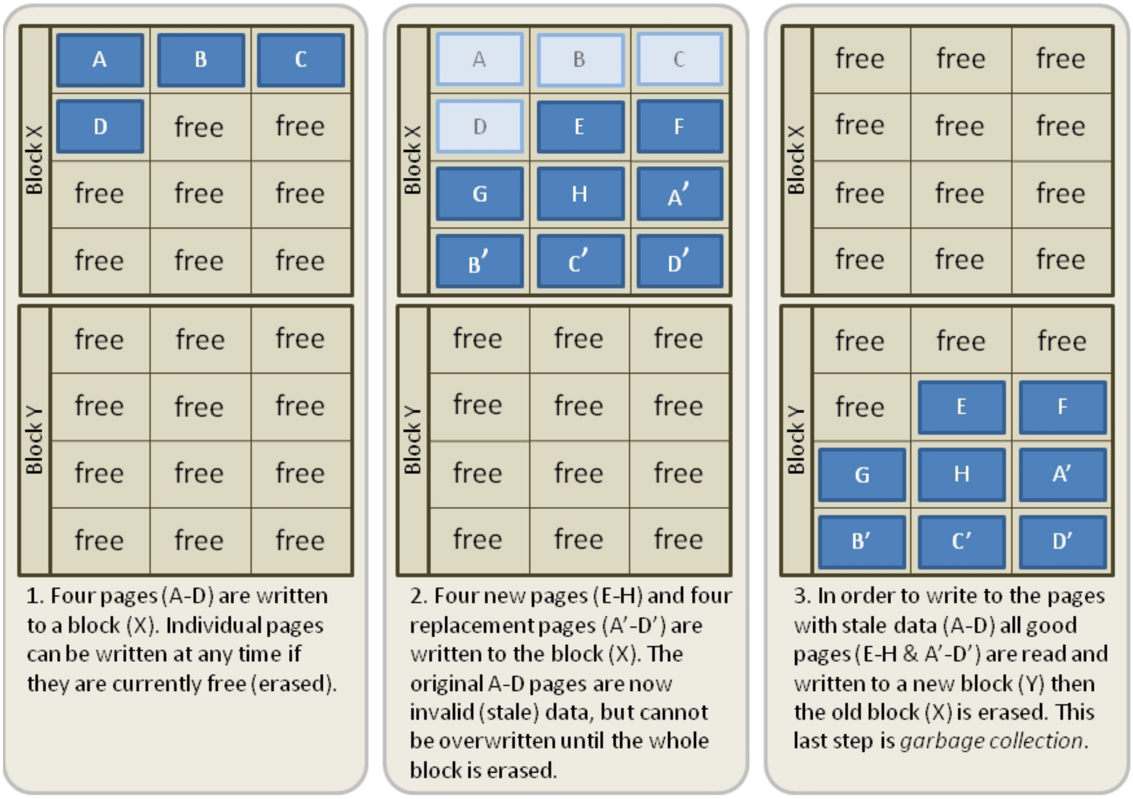
This is called *write amplification* and is the main reason why SSDs are slower on writes than on reads. In a brand new SSD, where all the cells are empty/ erased, the write performance is much higher but will drastically decrease after some period of use. The erasure of cells will also get slower when the number of P/E cycles increases.

To reduce the effect of write amplification, an SSD generally has a large internal cache. If connected to a dedicated SSD controller, the housekeeping process called *garbage collection* can erase and reclaim pages with stale data in advance, as a background process. TRIM is a similar functionality working at the file system level. Both garbage collection and TRIM will be covered in more detail next.

## Garbage collection and TRIM

The process called *garbage collection* is an important background process used to free up usable (i.e. writable) space in the SSD. As mentioned above, flash memory cells holding stale information can’t be reprogrammed (written) until they have been erased. Since erasing is done at block level (256 kByte) and writing is done at page level (4 kByte), a used SSD will end up with a lot of space that must induce the performance-killing *Read-Erase-Modify-Write Process* to be used.

A better solution would be if we could rearrange valid data between blocks to free up whole blocks that can be erased in advance, as a transparent background process.



1. Four pages (A-D) are written to a block (X). Individual pages can be written at any time if they are currently free (erased).

2. Four new pages (E-H) and four replacement pages (A’-D’) are written to the block (X). The original A-D pages are now invalid (stale) data, but cannot be overwritten until the whole block is erased.

3. In order to write to the pages with stale data (A-D) all good pages (E-H & A’-D’) are read and written to a new block (Y) then the old block (X) is erased. This last step is *garbage collection*.

**Figure 6**
**Garbage collection as described by Wikipedia**

The stale data in the middle of *Fig. 6* (light blue A, B, C, D) can be pages containing old data that have been modified by the file system (modified files) or pages rewritten by the wear-levelling process. An efficient garbage-collection process usually calls for an intelligent SSD controller, and will not take place if we just swap an HDD for an SSD in a common PC. Lack of efficient garbage collection will lead to unpredictable performance degradation over time. Depending on the SSD technology and workload, the write performance can suffer dramatically.

A way to further address the garbage-collection issue is TRIM, which became an official standard in 2009. TRIM is a way to perform garbage collection at the file system level. When a file is deleted by the file system, it is not erased from the disk media – it is just flagged as deleted, which means that the file system knows it can reuse the space. That’s why it is possible to recover accidentally-deleted files as long as they haven’t been overwritten.

The problem with the SSD is that the file system can’t reuse deleted file space until the corresponding blocks are erased. What TRIM does is tell the SSD that when a file is deleted, the corresponding pages and blocks are free to erase in advance, as a background process. The TRIM way of garbage collection usually starts when the SSD is filled with data above a certain threshold – for example, over 50% of used disk capacity.

As a metaphor you could think of TRIM as being the equivalent of an automatic HDD defragmentation process.

**A WARNING! Never use HDD defragmentation on an SSD – it will just make things worse and cause excessive wear to the SSD.**

For TRIM to work we need to have TRIM support not only in the *SSD firmware* but also in the *operating system* and the *file system*. Older SSDs without TRIM support can often get a firmware update to support TRIM. New SSDs should always have TRIM support, but if you decide to use SSD in a RAID-configuration, TRIM will not work – then you really need a dedicated SSD controller that takes care of garbage collection.

Not all operating systems and file systems support TRIM. On the Windows platform there is no support for TRIM prior to Windows 7 and Windows Server 2008 R2. For Linux, TRIM support was added at version 2.6.33 but will only work with the EXT4 and Btrfs file systems. On Apple Mac, a limited TRIM support was introduced with OS X v10.6.6 and a broader support is available with OS X v10.7.

A (so far) well-updated list of OS support for TRIM is found at **http://en.wikipedia.org/wiki/TRIM**

Both garbage collection and TRIM are very important for maintaining SSD performance and endurance.

## Endurance and Errors

Now you know that the tunnel oxide in the flash memory cell wears out when the P/E cycles are getting over specific numbers – the rule of thumb is 10’000 P/E cycles for MLC and 100’000 P/E cycles for SLC.

You also know that as the memory cell becomes gradually damaged, the risk of bit errors increases. (A single bit error is when a digital one (1) appears as a zero (0) – or vice versa.)

In all digital systems there is a risk of bit errors, which is why there are lots of specific techniques to detect and correct bit errors.

In the SSD, we have the concepts of:

- Raw Bit Errors – RBE (bit errors before correction);
- Corrected Bit Errors – CBE (corrected by ECC – Error Correction Code);
- Uncorrected Bit Errors – UBE (bit errors that cannot be corrected by ECC).

As a user, it’s impossible to check the different error rates, and as long as the RBE becomes CBE, we don’t care much. But from a manufacturer’s point of view, the error rates and the possibility to correct errors will indicate the quality and endurance of the SSD.

JEDEC Solid State Technology Association has specified an Endurance Verification Test where endurance is defined as the maximum TBytes written to SSD with less than 3% failure of the disk devices and a UBE rate less than 1 / $10^{16}$. There are disks produced with an UBE rate of 1 / $10^{17}$ or less which corresponds to about ten uncorrected errors per PByte. These look like really nice numbers – if only the endurance could be guaranteed in real life too.

Unfortunately the reality check tells us that SSDs in production have less optimal operating conditions than in the manufacturers’ labs.

**Actual SSD endurance, or life cycle, depends on a combination of several (interacting) factors such as:**

- Memory type – SLC or MLC;
- Production quality – enterprise class or consumer class;
- Internal architecture;
- Housekeeping processes – including TRIM, garbage collection and wear levelling;
- Temperature – SSDs are sensitive to high temperatures;

- Power supply – SSDs are more vulnerable to data loss/corruption or cell damage because of power loss or voltage spikes;
- Workload – the I/O pattern influences how efficient the housekeeping processes may be;
- Firmware updates – many manufacturers provide updated SSD firmware with enhancements and bug fixes that are important to apply on the SSD.

Sadly, we don't really know much about how long an SSD will last in our production environment. Of course we don't know the exact life cycle of an HDD either – but at least we have a long history and experience with HDDs that SSD technology is still lacking.

Another factor that might be important is *data retention time* which will be explained next.

## Flash memory retention

Let's have a look at the flash memory cell again. Programming the cell puts charge in the floating gate – and the charge is supposed to stay there forever, but...

Flash cells that are left without external voltage will slowly lose their charge over time – and one day we will get a raw bit error (RBE). Fortunately the ECC will correct the RBE to a CBE as long as the RBE rate is within the limits of what the ECC can handle (if this seemed like mumbo jumbo to you – please read the section about Endurance and Errors again).

However, since the tunnel oxide slowly degrades, the discharging will probably become faster and, if we have a high temperature in the SSD, this degradation will become even faster. If in addition we accidentally shut off our computer the wrong way or have other situations that might produce transient voltages to the SSD – we are vulnerable to further damage to the memory cells.

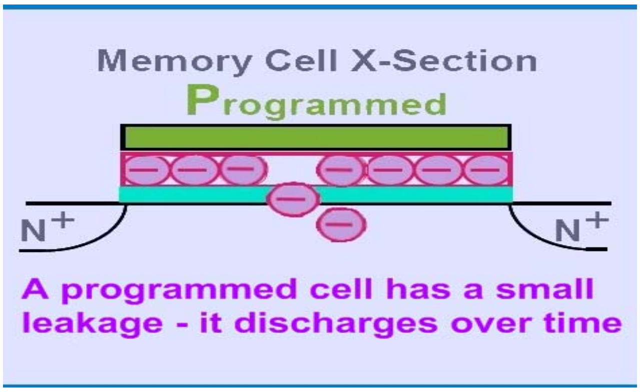


**Figure 7**
**A flash memory cell leaking charge**

Data retention can be an unpredictable issue – depending on circumstances – but even without external influences (like high temperatures, transient voltages or degraded tunnel oxide), there can be big quality differences between different SSDs.

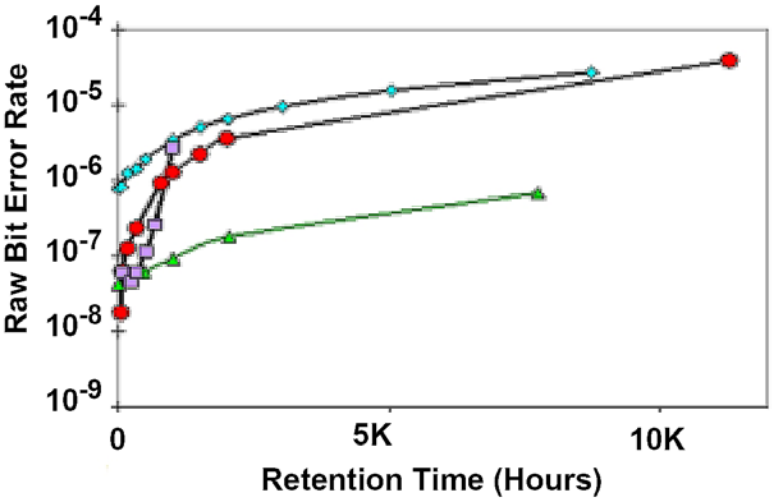


**Figure 8**
**Retention time for different flash memories**

Depending on the type of flash memory and operating conditions, the retention time can be from less than a year to more than ten years.

Some dedicated SSD controllers have an "always on" feature to keep the cells powered to ensure that they don't discharge over time.

Flash drives used for offline archiving should have a scheduled routine where they are powered on and checked for bit errors – or just have their data copied / rewritten at defined intervals. When we get a UBE in an existing file, we have an error that the file system is unaware of.  If we are unlucky we can have a file system with lots of corrupted data that nobody knows about – until the day we need it, and this is not purely an SSD-specific issue.  That is why file checksum checks (like hash) are important when it comes to long-term storage or archiving.

## Burn in!

Many of the more-or-less hidden weaknesses with flash SSDs have been covered here.  Some of them – like write amplification – are briefly mentioned by name in the manufacturers' spec. sheets.

However – as far as the author is aware – the two most obvious of all discouraging SSD oddities are never mentioned by the vendors:

1) The first stunning performance experience you get from your brand new SSD is like the very first kiss – it doesn't last as long as you wish and you will never ever reach the same sensation again (at least not together with the same object).
2) The second most obvious flaw from the manufacturers' side is the lack of standardized performance definitions for SSDs – which give them the possibility of making up the most fantastic stories about performances you will long for, but never get.

These are well-known facts that have added to give SSDs a bad reputation – especially for enterprise use – since so many high expectations have turned into disappointments after a period of use.

The Storage Networking Industry Association (SNIA) decided to define a specification to at least make comparative tests of Solid-Stage Storage (SSS) possible.  This way, the professional user can evaluate different SSD solutions and choose from true comparative performance results – since the manufacturers' specifications really don't tell what you need to know.

In the introduction to the SNIA's *Solid State Storage (SSS) Performance Test Specification (PTS) Enterprise, version 1.0*, from 26 April 2011, it is stated: "*Manufacturers need to set, and customers need to compare, the performance of Solid State Storage (SSS) devices.  This Specification defines a set of device level tests and methodologies intended to enable comparative testing of SSS devices in Enterprise systems.*"

This is a good start!  The spec. sheets will continue to tell fairy tales but at least we have a better chance to learn what to expect when the SSDs end up in the real world.

You can read more about this here:

**http://snia-europe.org/en/technology-topics/solid-state-storage/sss-publications.cfm/whitepaper0426**

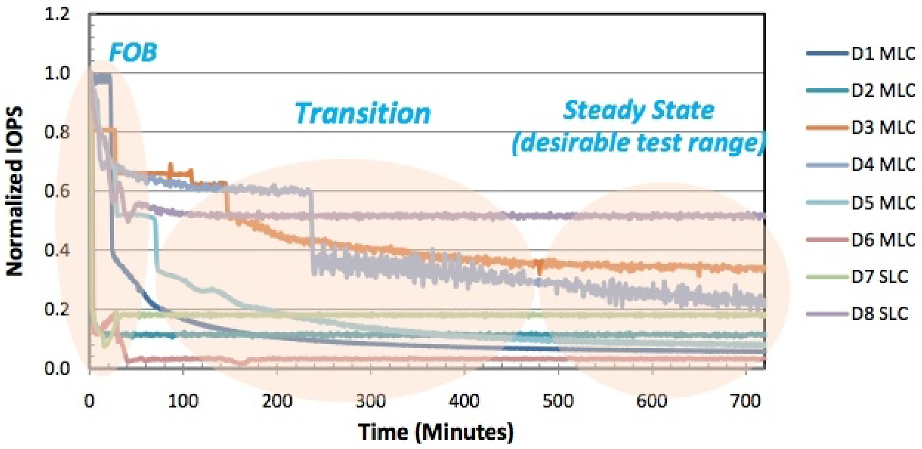


**Figure 9**
**Performance drop from Fresh Out off the Box (FOB) to Steady State**

*Fig. 9*, from SNIA SSS PTS Enterprise 1.0, shows the relative performance drop for eight different SSD devices when written with 4 kByte random writes. In the beginning, the disks are new – fresh out of the box (FOB) – that, during a transition phase, lose approximately 50% to 95% of their write performance, until they reach a *steady state performance*. The steady state performance is the performance the SSD will most likely maintain until it gets completely worn out.

It’s easy to see that an SSD implementation that expects steady FOB performance can end up in a big disappointment after a period of use.

I’d like to encourage the really serious manufacturers to ship their disks after a factory burn-in, so the customer knows what performance he or she is paying for.

## From heaven to hell

Let’s enter the real-world disaster I mentioned at the beginning of this article – i.e. what happened to the solid-state disk in our OB vehicle *(Figs 1 and 2)*?

If you managed to read this article from the beginning, I would say that in our implementation we nailed every SSD weakness you have now read about. But it started really good. Together with the PC manufacturer, we made a customized audio workstation with a very low acoustic noise level. It could be put in the radio studios without causing noise that the microphones would pick up.

We also put them in our OB vehicles, which lead to the first indications of spooky problems.

**So here it is – the selection of what obviously went most wrong for us:**

- We were early adopters:
  - Neither we nor the manufacturer understood the nature of SSDs when the audio workstation concept was created;
  - The concept was developed and tested on new fresh-out-of-the-box SSDs;
  - The need for burn-in was not known at the time.
- We used the common HDD controller for the SSD:
  - There was no working garbage collection;
  - The caching was optimized for a mechanical HDD.
- No support for TRIM:
  - The SSD was made in 2009 with firmware prior to TRIM;
  - Upgrading the firmware didn’t help since Windows XP doesn’t support TRIM in any case.
- Insufficient wear levelling on the SSD:
  - The SSD had an early version of a wear-levelling algorithm which had problems with large random or non-sequential writes.
- Insufficient internal cache on the SSD:
  - The write cache got saturated when the *Read-Erase-Modify-Write* process went too slow;
  - A better SSD cache might have helped during shorter write peaks.
- The workload:
  - All the conditions above created hot spots on the SSD where cells were prematurely worn out;
  - The error correction couldn’t handle remapping of faulty cells in an appropriate manner, maybe because of insufficient over provisioning.
- The operating conditions:
  - Mounted in OB vehicles on a sunny day, the SSD temperature could be quite high = decreased endurance;

  - Power on and power off could sometimes be quite brutal, probably with transient voltages = decreased endurance.

## *The pros*

However, let's also have a look at the pros of using SSD storage today and take a quick glance into the future.

Despite all the cons that you have learned about, there are a lot of situations when SSDs make good sense and give good value for money – as long as you know what to expect.

First I want to point out that SSDs are getting better and better – the technology and the production are constantly improving and prices are falling.

**However, before choosing an SSD solution, please consider the following:**

- Be sure that TRIM is supported by the SSD, OS & file system – especially if the SSD is used with a common HDD controller that lacks garbage collection functionality:
  - Check **http://en.wikipedia.org/wiki/TRIM** if you are unsure about TRIM support for your OS and file system;
  - There are some TRIM utilities trying to add TRIM functionality if the OS / file system doesn't have TRIM support – be sure to have this functionality verified before choosing this solution.
- SSD is usually a good choice when high I/O performance is needed – or low noise, shock proof operation or low power consumption are important factors:
  - Non-sequential *reads* will give best performance gain compared to HDD;
  - Be aware of the need for burn-in before knowing what performance you can get.
- SLC is a better choice than MLC for intensive I/O activity, or 24/7 operation:
  - Choosing SSD is great for speeding up web servers – but remember the need for garbage collection and/or TRIM;
  - RAID with SSD can give very good performance and endurance, but you must use a RAID-controller designed for SSD – otherwise you will end up in a great mess when neither TRIM nor garbage collection are working.
- SSDs with large over-provisioning give better endurance – i.e. a longer life cycle:
  - Choose a 100 GB SSD instead of a 128 GB device, 200 GB instead of a 256 GB device and so on. Then you know that 28 GB, or 56 GB etc. are probably reserved space for wear levelling, remapping of defective memory cells etc.
- For "Enterprise use" – choose enterprise-class drives, PCI Express (PCIe) SSDs or similar enterprise solutions:
  - PCIe cards with native dedicated SSD controllers and SSD storage can give very high I/O performance and endurance.

**Ivar Poijes** works as a development engineer at Swedish Radio (SR). In the late seventies he started his career at SR as a music-loving maintenance engineer. He gained a great experience in testing and troubleshooting both analogue and digital audio equipment. Then, when the first implementations of computer-based digital audio editing started in 1994, he became heavily engaged in designing, tuning and troubleshooting issues related to streaming media in network environments.

Over the years, Mr Poijes has followed and evaluated the evolution of SAN and storage from the perspective of the specific needs of a radio company. He has worked on several projects concerning performance tuning, optimization and architecture of high-availability solutions for streaming media storage. Currently he is also engaged in future long term strategies for SAN and storage at Swedish Radio.

- For other uses – always avoid the cheapest SSDs.
- Check that firmware updates are available for the SSD you choose.
- Be careful when shutting down systems with SSD devices – to avoid transient voltages.

Glancing in the crystal bowl, I can only hope that the SNIA initiative regarding performance tests on solid-state storage will lead to better and more predictable SSD products in the future. I'm looking forward to the day when all manufacturers are using the same standardized definitions for performance, endurance and retention of NAND-based flash storage. This might never happen since standards take time to establish and the development of other memory technologies are fast – which might require completely different test methodologies.

Some of the competing technologies for future SSDs are

- PCM: Phase Change Memory;
- MRAM: Magnetic RAM;
- FeRAM: Ferroelectric RAM;
- RRAM: Resistive RAM;
- Solid Electrolyte.

We've learned from history that it isn't necessarily the best technology that succeeds – so it will be very interesting to see how NAND-based flash memories will be used in about five years.

Until then we will get bigger, better and cheaper flash SSDs and better more-intelligent SSD controllers that give flash SSD solutions 24/7 reliability and life cycles of at least five years in production.

For the next 3-5 years we will find hybrid solutions with SSD+HDD as the dominating enterprise solution – letting SSD and HDD share the workload to get the best benefit from either technology.

Solid State Disks are here to stay. We will see fast and exciting development of SSD concepts in the near future. The old spinning disks will still continue to spin during the foreseeable future, furnishing what they do best – i.e. serving the forthcoming peta-, exa-, zetta- and yotta-byte needs of global online storage.

*This version: 30 August 2011*

Published by the European Broadcasting Union, Geneva, Switzerland

ISSN: 1609-1469

Editeur Responsable: Lieven Vermaele

Editor: Mike Meyer

E-mail: **tech@ebu.ch**

EBU·UER