A Flash Storage Technical and Economic Primer

By Mark May
Storage Consultant,

By James Green, vExpert
Virtualization Consultant

and

Scott D. Lowe, vExpert
Co-Founder, ActualTech Media

March, 2015
# Table of Contents

## TABLE OF CONTENTS

- HISTORY ................................................................................................................. 3
- TECHNICAL CHARACTERISTICS .............................................................................. 3
  - Flash Cells ........................................................................................................... 3
    - Single Level Cell Media ..................................................................................... 4
    - Multi-Level Cell Media ..................................................................................... 5
    - Triple Level Cell Media ..................................................................................... 6
    - Real World Media Type Impact ........................................................................ 6
  - Under the Hood: Physical and Logical Organization ........................................... 7
    - Drive Architecture ............................................................................................ 8
      - Drive Enclosure ............................................................................................... 8
      - Controller ....................................................................................................... 8
  - Flash Media Economics ......................................................................................... 11
    - Storage Cost Metrics ....................................................................................... 11
      - Performance: Cost per IOPS .......................................................................... 11
      - Capacity: Cost per GB ................................................................................... 11
      - The Break-Even for Flash .............................................................................. 12
    - Data Reduction ................................................................................................. 12
      - Data Reduction for Capacity .......................................................................... 13
      - Data Reduction for Performance .................................................................... 14
  - Conclusion ............................................................................................................ 16
  - Summary .............................................................................................................. 16

---

**ActualTech Media © 2015. All rights reserved.**

Under no circumstances should this document be sold, copied, or reproduced in any way except with written permission.

The information contained with the document is given in good faith and is believed to be accurate, appropriate and reliable at the time it is given, but is provided without any warranty of accuracy, appropriateness or reliability.

The author does not accept any liability or responsibility for any loss suffered from the reader’s use of the advice, recommendation, information, assistance or service, to the extent available by law.
History

Flash memory is a type of non-volatile memory storage, which can be electrically erased and programmed. What was the event that precipitated the introduction of this new storage medium? Well, it started in the mid-1980s, when Toshiba was working on a project to create a replacement for the EEPROM, a low-cost type of non-volatile memory, which could be erased and reprogrammed. The problem with the EEPROM was its cumbersome erasure process; it needed to be exposed to an ultraviolet light source to perform a complete erasure. To overcome this challenge, the E2PROM was created. The E2PROM type of memory cell was block erasable, but it was eight times the cost of the EEPROM. The high cost of the E2PROM led to rejection from consumers who wanted the low cost of EEPROM coupled with the block erasable qualities of the E2PROM.

This market desire led to the creation of what became known as the NOR architecture at Toshiba. While Toshiba was the first to create a NOR-based flash memory cell, the first commercially successfully design did not arrive until the late 1980s and was known as the ETOX Cell. The ETOX cell was slow to write and slow to erase, but was quick to read. This low-capacity NOR architecture became the industry standard replacement for read-only memory.

From there came new advances in storage technology, which has had a radical impact on the storage market as well as on data center economics. This paper discusses both the technical aspects of flash storage as well as the economic impact it has had.

Technical Characteristics

In the late 1980s, Toshiba introduced a new kind of architecture — the NAND architecture — which had a significantly lower cost-per-bit, a much larger capacity, and performance improvements in every area. The larger capacity made it suitable for storing data. Today’s flash-based storage systems are direct descendants on this architecture. In the following sections, discover the technology underpinnings that make these speedy systems work in the modern data center.

Flash Cells

The key component of a flash storage device is the cell. Information is stored in groups of memory cells comprised of floating-gate transistors (Figure 1). Information is comprised of 1s and 0s which are stored as electrons inside the floating-gate transistor. Think of this transistor as a light switch that can be either on or off. The light will stay on until turned off just as data will reside in flash media until it is erased. This statefulness (the ability to retain information) of the data is made possible by the floating gate.
The floating-gate is isolated, insulated all around by an oxide layer, with no electrical contacts. This means that any electrons placed on the floating-gate are literally trapped until removed. This persistence is the foundation for using flash memory as storage.

For electrons to be placed on the floating gate, they must penetrate the insulating material used for isolation. In order to penetrate the insulation, the electrons must be exposed to a high voltage in a process called tunneling. Once the charged electron passes through the insulation, it lands on the floating-gate where it is then stored. The process of tunneling is the Achilles’ heel of a flash cell because it causes physical degradation to the insulation material. Each time a data store on the floating-gate needs to be reused, it has to be completely erased and then rewritten. The more this cycle of programming and erasing (called a P/E Cycle) is invoked, the more the material degrades. Thus each flash cell has a limited write endurance before the cell is completely degraded. This phenomenon is the reason that flash storage has a finite life span.

**Single Level Cell Media**

The traditional approach, which is the light switch example from earlier, is called a single-level cell (SLC). As the name suggests, only one bit of data is stored per cell (Figure 2). The SLC benefits from the highest levels of performance, lowest power consumption, and also enjoys high write endurance (more P/E cycles). These benefits result in the cost being higher on SLC than with other approaches. This fact has led to the near extinction of SLC in the storage market. Few NAND flash providers in today’s market build SLC-based systems.

<table>
<thead>
<tr>
<th>Value</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Erased</td>
</tr>
<tr>
<td>0</td>
<td>Programmed</td>
</tr>
</tbody>
</table>

With SLC, the state of the cell is determined by the voltage range of the cell (Figure 3). As a charge is placed on the floating-gate, and the voltage is raised to 4.0V, the cell will be marked as “programmed.” Anything less and the cell will be considered erased.
Multi-Level Cell Media

The second approach to storing data is to use what’s called the multi-level cell (MLC). As the name implies, MLC media allows a cell to have multiple storage states. Traditional MLCs allow up to four states which allows for double the number of bits to be stored on the same number of transistors as an SLC device. The primary benefit of MLC flash memory, compared to SLC, is lower costs due to higher density. However, this higher density comes as the cost of less write endurance; MLC media supports far fewer program/erase cycles than SLC.

The spaces between each of the voltage ranges in Figure 3 and Figure 4 are known as reference points. The reduced size of the reference point between each state in an MLC (versus an SLC) means a more rigid control is needed to ensure precise charge detection. Without precise detection, it would not be possible to determine the voltage value and, hence, there would be no way to determine a data value.

A good way to think about MLC is as a glass of water. It can be either full, ⅔ full, ⅓ full, or empty. These four states yield two bits of information (Figure 5).
Triple Level Cell Media

The desire for even higher density has led to the creation of another type of cell. The triple-level cell (TLC) adds a third bit stored on the same number of transistors. This density is achieved by further reducing the size of the reference point between each state (Figure 6). By reducing the size of the reference point, a TLC has 8 distinct states. The more bits that are stored on a single floating gate, the more changes it will be written to. The stresses placed on the oxide layer caused by these added writes will speed up its rate of deterioration. This means that TLC has far less write endurance then MLC, which in turn, has less write endurance than SLC.

<table>
<thead>
<tr>
<th>Value</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Fully Programmed</td>
</tr>
<tr>
<td>01</td>
<td>Partially Programmed</td>
</tr>
<tr>
<td>10</td>
<td>Partially Erased</td>
</tr>
<tr>
<td>11</td>
<td>Fully Erased</td>
</tr>
</tbody>
</table>

Real World Media Type Impact

Each flash memory cell type has different advantages and disadvantages. This leads to different use cases which factor in density, cost, performance, and write endurance. As seen in Figure 7, the higher density cell types have increasingly slower performance. This is due to the added amount of work required to accurately discern the increasingly tighter voltage ranges. Visualize the glass of water analogy; it is difficult to distinguish between a glass of water that is ⅝ full and one that is ¾ full (Figure 7). Such measurements require the controller for denser MLC flash media to be far more refined than those necessary to detect voltage levels in SLC media.

Figure 5 – Four MLC states yield two bits of information

Figure 6 - TLC Voltage References

Voltage

111 110 101 100 011 010 001 000
As more bits are stacked on a single cell the fewer that are needed to reach the desired capacity. This higher capacity leads to a sacrificing of write endurance, but carries with it a much lower cost. Moreover, MLC and TLC media types take longer to read data and erase cells, introducing latency into the storage equation (Figure 8). However, flash storage vendors have developed array-based software workarounds that largely mitigate this issue.

<table>
<thead>
<tr>
<th>Type</th>
<th>Bits/cell</th>
<th>States</th>
<th>Endurance</th>
<th>Read Time</th>
<th>Erase Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLC</td>
<td>1</td>
<td>2</td>
<td>100k</td>
<td>25us</td>
<td>1.5ms</td>
</tr>
<tr>
<td>MLC</td>
<td>2</td>
<td>4</td>
<td>10-30k</td>
<td>50us</td>
<td>3ms</td>
</tr>
<tr>
<td>TLC</td>
<td>3</td>
<td>8</td>
<td>1k</td>
<td>75us</td>
<td>4.5ms</td>
</tr>
</tbody>
</table>

Under the Hood: Physical and Logical Organization

The capacity of a single NAND cell is rather small and is not much use on its own, but by combining multiple NAND cells, manufacturers can create a sizeable pool of capacity. Memory cells are combined to form a NAND string. Groups of NAND strings are then combined to form a NAND array.

The NAND array is shown in Figure 9. To form the NAND string, 32 or 64 floating-gates are joined together in a serial fashion. This string becomes the minimum unit for a read operation and holds 4 bytes of data. Horizontal rows of adjacent control gates are connected to form the logical page. The logical page is the minimum unit for writing. The NAND strings that share this horizontal connection form the block and become the minimum unit for erasing. This information is combined to form a reference to a particular area of data call the address.
Finally, groups of NAND strings are combined on a printed circuit board (PCB). This leads to a limitation of capacity as a PCB is generally not especially large. In order to overcome this physical limitation, an added layer of abstraction is needed — the controller. The controller combines groups of memory arrays and acts as the bridge to the host computer. The controller, together with the flash memory cell arrays, form the basis of the flash-based drive.

**Drive Architecture**

With the underlying storage constructs in place, attention needs to be turned to ways to make these devices usable in computing systems. The following sections describe the various aspects of the drive architecture and how they make flash storage work in a practical way.

**Drive Enclosure**

Groups of flash memory arrays are configured together to make a single usable pool of capacity that can be housed in many different form factors. One of the most common form factors is that of the standard spinning hard drive. This has the benefit of being able to leverage existing infrastructure in desktop computers and servers. In environments where space is at a premium, such as laptop computers, small form factors such as mSATA are used. Another option, often used in large enterprises, is to make use of the PCIe interface to provide flash storage to the host server. Because flash storage is not bound by the size constraints of traditional hard drives, the form factor options are nearly limitless.

**Controller**

The controller is one of the key components of a flash drive. Its goal is to provide the interface and protocol to the host and the flash memory cells. The controller is also responsible for software...
which helps mitigate the limitations of flash cell memory. This is done through a series of proactive monitoring and management tasks in the firmware.

**Host Interface**
The Host Interface, as the name implies, is responsible for the implementation of the connection to the host. In many consumer devices, such as a portable drive, this is often implemented as USB. Another popular option is the PCIe interface which connects the flash storage directly to the motherboard. The two most common protocols are Serial Attached SCSI (SAS) and Serial ATA (SATA), both of which are industry standards for data storage devices.

**Flash File System**
The flash file system is a mechanism to actually provide data access to information on the memory cells and is generally implemented as firmware inside the controller. It has three major jobs: wear leveling, bad block management, and garbage collection.

**Wear Leveling**
Wear leveling is a means of extending the life of a flash storage cell. Not all information stored within the same memory cell array is accessed and updated at the same time. Some data is updated frequently while other is much more static. This means that flash cells will not wear evenly across the board, but will wear based on data access patterns. Wear leveling is the controller’s attempt to mitigate uneven wear. A logical block address (LBA) from the host is translated to the physical flash memory location. When a host updates existing data, it is actually written to a new physical location and the LBA map is updated. The previously used cell is marked as invalid. This is called *dynamic wear leveling*. Another similar approach is called *static wear leveling* where all blocks, not just the updated ones, are moved to ensure even cell wear. Dynamic wear leveling is less complex, but static wear leveling will enable a longer lifetime use of the flash memory (Figure 10).

<table>
<thead>
<tr>
<th>Method</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static</td>
<td>Maximize flash lifespan</td>
<td>Complex</td>
</tr>
<tr>
<td></td>
<td>Robust and efficient</td>
<td>Slower writes</td>
</tr>
<tr>
<td>Dynamic</td>
<td>Easier to implement</td>
<td>Does not maximize lifespan</td>
</tr>
<tr>
<td></td>
<td>less impact on writes</td>
<td></td>
</tr>
</tbody>
</table>

*Figure 10 – Static vs. Dynamic wear leveling*

**Garbage Collection**
A cell is marked invalid due to data being updated by the host or because of wear leveling moving the data in order to balance the system. When this cell is marked as invalid it cannot be reused until it is erased. In flash storage, data erasing happens to entire blocks, not individual pages. Due to the nature of writing to flash storage, data on a block is often a mix of valid and invalid pages. In order to reclaim the stale invalid pages, the valid data must be moved to a new location so the entire block and be erased and reused. This process is called *garbage collection* and is a major reason why flash storage has extra writes. This problem is known as *write amplification*. Write
amplification is a major problem in flash storage because a cell has a limited number of writes before it deteriorates.

**Figure 11 – Garbage collection process**

**Bad Block Management**

No amount of wear leveling will prevent a cell from eventually going bad. After enough P/E cycles, any cell will become unreliable and need to be removed from service. Bad block management maintains a table of cells which are no longer working and replaces them with cells which have not yet reached the maximum P/E cycles. The new block generally comes from spare cells set aside for this very reason. If at any time a spare cell does not exist to replace the bad ones, the flash storage will experience data loss.

**Handling Read Disturb**

Reading data from a flash memory cell can have the eventual unintended consequence of causing nearby cells to change. This phenomenon is known as a read disturb. This means that when a particular cell is read over and over again, a nearby cell – and not the read target – could fail. The controller monitors the number of times each cell is read and moves data from cells about to be disturbed to new blocks. The disturbed cell is removed from service, erased, and then returned to service.

**Error Correcting Codes (ECC)**

The limited lifespan of a flash cell creates a higher potential for data loss. In order to combat this problem, the flash controller employs a mechanism called error-correcting codes (ECC). When data is written, an ECC is generated and stored with the data. During a data read operation, the controller calculates another ECC and compares it against the stored code. If the two codes do not match, the controller knows an error exists in the data bits. By using the discrepancy in the code, the controller can, in many cases, adjust and fix the errors.
Flash Media Economics

In today’s world, IT organizations want everything to be better, faster, and cheaper. As changes come to the industry, it’s important to understand how to measure improvement. Specific to flash storage, it’s important to understand how choices about flash versus disk impact the bottom line. When it comes to making this determination, how can you be sure you’re getting the most out of every dollar?

Storage Cost Metrics

There are two primary ways to measure the cost of storage: it can be calculated in terms of performance or in terms of capacity.

Performance: Cost per IOPS

One way to measure the cost of storage is to calculate based on total IOPS performance in relation to size; this is called Cost per IOPS. As an example, if a PCIe SSD which costs $6,750 can deliver 200,000 IOPS, the $/IOPS would be about $0.03 per IOPS. By comparison, even a high-performing HDD like a 15K SAS disk of a comparable size would cost $180 and might deliver about 180 IOPS. This gives it a $/IOPS of $1/IOPS.

It is clear from the example that flash storage stands head and shoulders above spinning disks when it comes to economy in terms of performance. However, storage performance is but one factor in a storage decision. While flash storage is inexpensive in terms of performance, it still comes with a substantial cost in terms of capacity.

Capacity: Cost per GB

The other way storage cost is commonly measured is in terms of capacity in relation to cost. This number is called Cost per Gigabyte (GB). The same PCIe SSD used in the previous calculations costs $6,750 and has a raw capacity of 700 GB. This gives it a $/GB of $9.64/GB. The HDD costs $180 and has a capacity of 600 GB; so it comes in at $0.30/GB. Cost per GB generally drops substantially as the size of a spinning disk increases. For example, a 4 TB SATA disk costing $220 has a $/GB figure of just $0.05/GB.

Because organizations care about both performance and cost, and each medium is outstanding in only one metric, there must be a way to determine which is the right choice.
The Break-Even for Flash

By using a simple formula, you can begin to determine whether flash is more or less economical for any given purpose. This formula can be complicated by data reduction methods. Data reduction methods will be covered in-depth below. But at a raw capacity level, the economy of flash can be calculated this way:

\[
\frac{\text{IOPS required}}{\text{GB required}} < \frac{\text{cost per GB (SSD)}}{\text{cost per IOPS (HDD)}}
\]

Spinning disk as a medium is still substantially less expensive in terms of capacity than flash, but flash is much higher performing. To find the most economical choice, you must find the break-even point between the two factors. This is the purpose of the formula above.

When it costs more to produce enough performance with spinning disks than it costs to provide enough capacity with flash, flash is the most economical choice. A practical example using SharePoint as a workload could be calculated as follows: Microsoft documentation states\(^1\) that for SharePoint 2010, 2 IOPS/GB is preferred for performance. This calculation uses a SATA SSD rather than the PCIe SSD used in the previous calculation:

\[
\frac{2 \text{ IOPS}}{\text{GB}} > \frac{\$1.96(\text{GB/flash})}{\$1(\text{IOPS/HDD})}
\]

Answer: \(2 > 1.96\)

This expression evaluates to "True," which means that for this particular application, it would actually be slightly cheaper to use flash, and this expression has not yet accounted for any capacity optimizations (data reduction) of the flash storage. If more data can be squeezed into flash, it brings the break-even point even lower. So how might that be accomplished?

Data Reduction

Modern storage arrays have various ways of handling data such that more data can be stored on less physical media. By compressing and deduplicating data, flash storage can be made a more viable capacity-based solution. Today, driving economy in flash storage is all about how much data can fit on the drive. Data reduction can also further increase performance, leading to a very high performance storage platform at a very reasonable cost.

Data Reduction for Capacity

Two main methods are used to reduce the size of data written to physical storage—compression and deduplication. Compression and deduplication are cousins. They take advantage of the fact that individual blocks are commonly repeated to make up a file. These data reduction techniques remove this redundancy in such a way that it can be restored upon access. This allows storage of significantly more information on the physical medium.

Compression

Compression in the enterprise storage sense means using a lossless compression algorithm like LZ or one of its derivatives. These algorithms are the basis for common archive file formats in the end-user space, like ZIP, GZ, and CAB. They look at a single file, and using CPU processing power, remove redundancies within the file. Depending on the size of the files, this CPU utilization can be quite demanding. The key to successfully implementing a compression scheme is to strike a balance between CPU overhead and reduction in size on disk. Achieving high compression ratios and doing it with less CPU overhead both work toward making flash storage more economically feasible.

Post-Process Deduplication

Deduplication is similar to compression in that it removes redundancies. The difference is scope. While compression works at a file level, deduplication works at a file-system level. This technology compares the data that makes up all files being stored and removes redundancies. Deduplication technologies may need substantial memory to compare data during the reduction process. During a task known as “post-process deduplication,” the storage platform removes redundancies on a schedule (nightly, for instance), as shown in Figure 7. The advantage to this method is that no additional latency or processing power is needed at the time of the write. All data is written initially and then dealt with at a later time. The disadvantage is that extra storage capacity is required to store the hydrated data until the deduplication process is run.

In-Line Deduplication
Contrary to the post-processing method, a storage platform can use in-line deduplication (Figure 8) to identify duplicate writes before they are ever actually written to disk. This alleviates the need for excess storage capacity to store hydrated data. The risk, however, is that in-line deduplication requires processing overhead in real-time and can potentially introduce latency. As write IOs come in, the data is fingerprinted and checked against existing, unique data. If it turns out that this data already exists, a pointer is written to the unique copy of the data, rather than writing the data a second time. If it does not exist, the data is written as usual. One can see how the overhead can quickly become great when every block to be written must first be fingerprinted and checked against the existing data.

Between compression and deduplication, very high reduction ratios can be achieved. This further drives down the cost of flash. Of course, the calculations can be quite complex, because extra CPU power must also be allocated to do the reduction tasks.

Data Reduction for Performance
There is yet another way to increase the economy of storage. A storage platform can achieve greater value in a given capacity by asking the disk to store less information. In the same way, it can achieve greater performance by asking the disk to read or write less data in the first place.

Read Caching
Regardless of the storage medium, reading from it always costs some sort of latency. In order to optimize read times, storage platforms commonly include a small amount of DRAM with which to cache the hottest of hot blocks. (“Hot” meaning it is read quite frequently.) The more read IOs exist in DRAM, the better the overall read performance of the array. Unfortunately, DRAM is quite expensive and just adding more is often not an option. It would be helpful, then, to be able to fit more hot blocks in the finite amount of DRAM.

Write Penalties
All data written to any sort of failure tolerant storage will incur write penalties. For example, RAID 1, 5, 6, and 10 all have an associated write penalty. This is because in order to protect the data, it must be written to more than one place. As a simple example, RAID 1 writes the same block twice, meaning it has a penalty of 2.
A real scenario where this is relevant would be mirrored cache on the storage platform. In order to acknowledge writes very quickly, a storage platform may use write-back cache. Write-back cache acknowledges writes before actually writing the data down to disk, and thus must be tolerant to failure. In order to meet this resilience requirement, perhaps the cache write is mirrored to cache on a second controller. So for every write IO that comes in, it must be written twice before it can be acknowledged.

As fast as this may be on a modern array, it would be much faster if it never had to be written at all. It is commonly said about in-line deduplication that, “The least expensive write operation is the one you don’t have to do.”

**Impact on Read Performance**
To solve the problem of having very limited space available in DRAM, applying data reduction techniques can allow for substantially more hot blocks to be served by only caching one copy (Figure 8). As mentioned previously, the more blocks that are served from ultra fast DRAM, the higher the overall read performance of the array.

**Impact on Write Performance**
Due to the time it takes to acknowledge a write, and especially in light of potential write penalties, being able to deduplicate a block before it is written can provide dramatic increases in write performance. In the example shown in Figure 9, rather than writing 12 blocks (before write penalties), only the four unique blocks are written. Data reduction before writing has essentially increased write performance in the example by 300%.
Conclusion
With all the options present in the storage market today, it can be overwhelming to know which is
the right choice. One thing is for certain: flash storage is an economical storage medium for certain
workloads. When the cost for required performance on spinning disk outweighs the cost of the
capacity on flash, flash is the right choice. And the break-even point is lowered proportional to the
effectiveness of the data reduction methods used prior to access.

Data reduction techniques in combination with flash storage can also serve to dramatically
increase the overall performance of the array by allowing more read IOs to be served from cache
and less write IOs to be written.

Summary
Although today’s NAND flash storage has its roots in 30-year-old technology, innovation has
negated almost all of the challenges that are inherent in the media. Moreover, modern storage
companies are taking even more software-based steps to further overcome such challenges.
Between these advances, it’s clear that flash media use will continue to grow in the data center.
About the Authors

Mark May has worked in the technology industry since 1995, starting his career by co-founding a local internet service provider where he handled network issues, Unix administration, and development efforts. In 1998 Mark accepted a Unix administration role at a large health insurance, where he led the effort to implement centralized enterprise storage. Since this first exposure to storage he has held a number of specialist roles helping enterprises meet the demanding challenge of an ever-changing storage environment.

James Green is an independent blogger at www.virtadmin.com, and is a two-time vExpert, serial Tech Field Day delegate, and works as a virtualization consultant in the Midwest.

Scott Lowe is co-founder of ActualTech Media and the Senior Editor of EnterpriseStorageGuide.com. Scott has been in the IT field for over twenty years and spent ten of those years in filling the CIO role for various organizations. Scott has written thousands of articles and blog postings for such sites as TechRepublic, Wikibon, and virtualizationadmin.com and now works with ActualTech Media to help technology vendors educate their customers.

About Tegile

Tegile Systems is pioneering a new generation of intelligent flash arrays that balance performance, capacity, features and price for virtual desktop and database applications. With Tegile’s line of all-flash and hybrid storage arrays, the company is redefining the traditional approach to storage by providing a family of arrays that accelerate business critical enterprise applications and allow customers to significantly consolidate mixed workloads in virtualized environments.