Solid State Drives (SSDs)

Jin-Soo Kim (jinsookim@skku.edu)
Computer Systems Laboratory
Sungkyunkwan University
http://csl.skku.edu
Memory Types

**FLASH**
- High-density
- Low-cost
- High-speed
- Low-power
- High reliability

**DRAM**
- High-density
- Low-cost
- High-speed
- High-power

**EPROM**
- Non-volatile
- High-density
- Ultraviolet light for erasure

**EEPROM**
- Non-volatile
- Lower reliability
- Higher cost
- Lowest density
- Electrically byte-erasable

**ROM**
- High-density
- Reliable
- Low-cost
- Suitable for high production with stable code

Source: Intel Corporation.
Flash Memory Characteristics

- **Erase-before-write**
  - Read
  - Write or Program: 1 → 0
  - Erase: 0 → 1

- **Bulk erase**
  - Program unit:
    - NOR: byte or word
    - NAND: page
  - Erase unit: block
Logical View of NAND Flash

- A collection of blocks
- Each block has a number of pages
- The size of a block or a page depends on the technology (but, it’s getting larger)
NAND Flash Types

- **SLC NAND**
  - Single Level Cell
  - 1 bit/cell

- **MLC NAND**
  - Multi Level Cell (misnomer)
  - 2 bits/cell

- **TLC NAND**
  - Triple Level Cell
  - 3 bits/cell

- **3D NAND**

Source: Micron Technology, Inc.
NAND Applications

- Universal Flash Drives (UFDs)
- Flash cards
  - CompactFlash, MMC, SD, Memory stick, …
- Smartphones
  - eMMC (Embedded MMC)
  - UFS (Universal Flash Storage)
- SSDs (Solid State Drives)
- Other embedded devices
  - MP3 players, Digital TVs, Set-top boxes, Car navigators, …
# Commercial SSDs

## Commercial SSDs Overview

### SSDs Compared

<table>
<thead>
<tr>
<th>SSD Type</th>
<th>Vendor</th>
<th>Model</th>
<th>Form Factor</th>
<th>intricacies</th>
<th>CNNP</th>
<th>Capacity</th>
<th>Retail Price</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z400s SSD</td>
<td>Samsung</td>
<td>Z400s</td>
<td>2.5&quot;</td>
<td>SATA 3</td>
<td>303ms</td>
<td>256GB</td>
<td>88,140</td>
<td>enuri.com/view/Listmp3.jsp?cate=071307</td>
</tr>
<tr>
<td>850 EVO</td>
<td>Samsung</td>
<td>850 EVO</td>
<td>2.5&quot;</td>
<td>SATA 3</td>
<td>400ms</td>
<td>250GB</td>
<td>812,960</td>
<td>enuri.com/view/Listmp3.jsp?cate=071307</td>
</tr>
<tr>
<td>850 PRO</td>
<td>Samsung</td>
<td>850 PRO</td>
<td>2.5&quot;</td>
<td>SATA 3</td>
<td>548ms</td>
<td>256GB</td>
<td>1,088,600</td>
<td>enuri.com/view/Listmp3.jsp?cate=071307</td>
</tr>
</tbody>
</table>

---

**Note:** The above prices are as of May 14, 2016. For the latest prices and specifications, please visit the respective vendor's website.
Anatomy of an SSD

- Samsung 850 Evo

# HDDs vs. SSDs

<table>
<thead>
<tr>
<th>Feature</th>
<th>SSD (Samsung)</th>
<th>HDD (Seagate)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model</strong></td>
<td>MZ-75E2T0B (850 Evo)</td>
<td>ST2000LM003 (SpinPoint M9T)</td>
</tr>
<tr>
<td><strong>Capacity</strong></td>
<td>2TB (128Gb 32-Layer 3D V-NAND TLC x 16 die/channel x 8 channels)</td>
<td>2TB (3 Discs, 6 Heads, 5400 RPM)</td>
</tr>
<tr>
<td><strong>Form factor</strong></td>
<td>2.5”, 66g</td>
<td>2.5”, 130g</td>
</tr>
<tr>
<td><strong>DRAM</strong></td>
<td>2 GB</td>
<td>32 MB</td>
</tr>
<tr>
<td><strong>Host interface</strong></td>
<td>SATA-3 (6.0 Gbps)</td>
<td>SATA-3 (6.0 Gbps)</td>
</tr>
<tr>
<td><strong>Power consumption</strong></td>
<td>3.7, 4.7 W / 0.5 W / 0.05 W</td>
<td>2.3 W / 0.7 W / 0.18 W</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
<td><strong>850 Evo</strong>¹: 128KB/QD2 Sequential: 128KB/QD2 Random: 4KB/QD32</td>
<td><strong>M9T</strong>²: 128KB/QD2 Sequential: 128KB/QD2 Random: 4KB/QD32</td>
</tr>
<tr>
<td></td>
<td>Sequential read: 544 MB/s</td>
<td>Sequential read: 124 MB/s</td>
</tr>
<tr>
<td></td>
<td>Sequential write: 520 MB/s</td>
<td>Sequential write: 124 MB/s</td>
</tr>
<tr>
<td></td>
<td>Random read: 97,687 IOPS</td>
<td>Random read: 56 IOPS</td>
</tr>
<tr>
<td></td>
<td>Random write: 89,049 IOPS</td>
<td>Random write: 98 IOPS</td>
</tr>
<tr>
<td></td>
<td>Random read: 11,335 IOPS (QD1)</td>
<td>Power-on to ready: 3.5 sec</td>
</tr>
<tr>
<td></td>
<td>Random write: 38,433 IOPS (QD1)</td>
<td>Average seek: 12/14 ms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average latency: 5.6 ms</td>
</tr>
<tr>
<td><strong>Price</strong>³</td>
<td>1,009,380 won (505won/GB)</td>
<td>117,060 won (59won/GB)</td>
</tr>
</tbody>
</table>

² [http://www.storagereview.com/samsung_spinpoint_m9t_hard_drive_review](http://www.storagereview.com/samsung_spinpoint_m9t_hard_drive_review)
State of the Art

- World’s first 2.5” SAS 16TB SSD @ FMS 2015
NAND Constraints

- No in-place update
  - Require sector remapping (or address translation)
- Bit errors
  - Require the use of error correction codes (ECCs)
- Bad blocks
  - Factory-marked and run-time bad blocks
  - Require bad block remapping
- Limited program/erase cycles
  - < 100K for SLCs, < 3K for MLCs, < 1K for TLCs
  - Require wear-leveling
Flash Translation Layer (FTL)

- A software layer to make NAND flash fully emulate traditional block devices (e.g. disks)

Source: Zeen Info. Tech.
SSD Internals

![SSD Internals Diagram]
Address Mapping

- Required since flash pages cannot be overwritten
Example: Page Mapping

- Flash configuration
  - Page size: 4KB
  - # of pages / block = 4

- Current state
  - Written to page 0, 1, 2, 8, 4, 5

- Reading page 5

Logical page #5 \(0000000101\)
Example: Page Mapping

- **Flash configuration**
  - Page size: 4KB
  - # of pages / block = 4

- **Current state**
  - Written to page 0, 1, 2, 8, 4, 5

- **New requests (in order)**
  - Write to page 9
  - Write to page 3
  - Write to page 5
Example: Page Mapping

- **Flash configuration**
  - Page size: 4KB
  - # of pages / block = 4

- **Current state**
  - Written to page 0, 1, 2, 8, 4, 5

- **New requests (in order)**
  - Write to page 9
  - Write to page 3
  - Write to page 5
Example: Page Mapping

- Flash configuration
  - Page size: 4KB
  - # of pages / block = 4

- Current state
  - Written to page 0, 1, 2, 8, 4, 5

- New requests (in order)
  - Write to page 9
  - Write to page 3
  - Write to page 5
Example: Page Mapping

- **Flash configuration**
  - Page size: 4KB
  - # of pages / block = 4

- **Current state**
  - Written to page 0, 1, 2, 8, 4, 5

- **New requests (in order)**
  - Write to page 9
  - Write to page 3
  - Write to page 5
Garbage Collection

- **Garbage collection (GC)**
  - Eventually, FTL will run out of blocks to write to
  - GC must be performed to reclaim free space
  - Actual GC procedure depends on the mapping scheme

- **GC in page-mapping FTL**
  - Select victim block(s)
  - Copy all valid pages of victim block(s) to free block
  - Erase victim block(s)
  - Note: At least one free block should be reserved for GC
Example: GC in Page Mapping

- **Current state**
  - Written to page 0, 1, 2, 8, 4, 5
  - Written to page 9, 3, 5

- **New requests (in order)**
  - Write to page 8
  - Write to page 9
  - Write to page 3
  - Write to page 1
  - Write to page 4
Example: GC in Page Mapping

- **Current state**
  - Written to page 0, 1, 2, 8, 4, 5
  - Written to page 9, 3, 5

- **New requests (in order)**
  - Write to page 8
  - Write to page 9
  - Write to page 3
  - Write to page 1
  - Write to page 4
Example: GC in Page Mapping

- **Current state**
  - Written to page 0, 1, 2, 8, 4, 5
  - Written to page 9, 3, 5

- **New requests (in order)**
  - Write to page 8
  - Write to page 9
  - Write to page 3
  - Write to page 1
  - Write to page 4
**Example: GC in Page Mapping**

- **Current state**
  - Written to page 0, 1, 2, 8, 4, 5
  - Written to page 9, 3, 5

- **New requests (in order)**
  - Write to page 8
  - Write to page 9
  - Write to page 3
  - Write to page 1
  - Write to page 4
Example: GC in Page Mapping

- **Current state**
  - Written to page 0, 1, 2, 8, 4, 5
  - Written to page 9, 3, 5

- **New requests (in order)**
  - Write to page 8
  - Write to page 9
  - Write to page 3
  - Write to page 1
  - Write to page 4
Example: GC in Page Mapping

- **Current state**
  - Written to page 0, 1, 2, 8, 4, 5
  - Written to page 9, 3, 5

- **New requests (in order)**
  - Write to page 8
  - Write to page 9
  - Write to page 3
  - Write to page 1
  - Write to page 4
OS Implications

- NAND flash has different characteristics compared to disks
  - No seek time
  - Asymmetric read/write access times
  - No in-place-update
  - Good sequential read/write and random read performance, but bad random write performance
  - Wear-leveling
  - ...
  - Traditional operating systems have been optimized for disks. What should be changed?
SSD Support in OS

- Turn off “defragmentation” for SSDs
- New “TRIM” command
  - Remove-on-delete
- Simpler I/O scheduler
- Align file system partition with SSD layout
- Flash-aware file systems (e.g. F2FS in Linux)
- Larger block size (4KB)
Beauty and the Beast

- NAND Flash memory is a beauty
  - Small, light-weight, robust, low-cost, low-power non-volatile device
- NAND Flash memory is a beast
  - Much slower program/erase operations
  - No in-place-update
  - Erase unit > write unit
  - Limited lifetime
  - Bit errors, bad blocks, …
- Software support is essential for performance and reliability!
Beyond Flash

- Resistance-based memory technologies

<table>
<thead>
<tr>
<th>DRAM</th>
<th>NAND</th>
<th>MRAM</th>
<th>PRAM</th>
<th>ReRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cell size</td>
<td>~8F^2</td>
<td>~5F^2 (SLC)</td>
<td>~8F^2</td>
<td>~4F^2 (transistor)</td>
</tr>
<tr>
<td>Density</td>
<td>xGigabit</td>
<td>xxGigabit</td>
<td>xxMegabit</td>
<td>xGigabit</td>
</tr>
<tr>
<td>Latency</td>
<td>~50 ns</td>
<td>20 to ~200 µs</td>
<td>~50 ns</td>
<td>~200 ns</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>~1 GBps</td>
<td>~100 MBps</td>
<td>~1 GBps</td>
<td>~100 MB/s</td>
</tr>
<tr>
<td>Volatility</td>
<td>Volatile</td>
<td>Nonvolatile</td>
<td>Nonvolatile</td>
<td>Nonvolatile</td>
</tr>
<tr>
<td>Endurance</td>
<td>&gt;10^{15}</td>
<td>&gt;10^5</td>
<td>&gt;10^{15}</td>
<td>&gt;10^6 to 10^{-12}</td>
</tr>
<tr>
<td>Retention</td>
<td>&gt;64 ms</td>
<td>&gt;10 years</td>
<td>&gt;10 years</td>
<td>&gt;10 years</td>
</tr>
<tr>
<td>Application</td>
<td>Working memory</td>
<td>Data storage memory</td>
<td>Working nonvolatile RAM</td>
<td>Code memory and buffer memory</td>
</tr>
</tbody>
</table>