A Case for Flash Memory SSD in Enterprise Database Applications

Sang-Won Lee  Bongki Moon
Sungkyunkwan University  University of Arizona

Chanik Park  Jae-Myung Kim  Sang-Woo Kim
Samsung Electronics Co., Ltd.  Altibase Corp.  Sungkyunkwan University
Magnetic Disk vs Flash SSD

Champion for 50 years

Seagate ST340016A
40GB, 7200rpm

New challengers!

M-Tron Flash SSD
32GB 2.5 inch

Samsung FlashSSD
32GB 1.8 inch
Trend in Market Today

• **In mobile storage market**
  - NAND flash memory wins over hard disk in mobile storage market
    - PDA, MP3, mobile phone, digital camera, ...
  - Due to advantages in size, weight, shock resistance, power consumption, noise ...

• **In personal computer market**
  - Compete with hard disk in personal computer market
    - 32GB Flash SSD: M-Tron, Samsung, SanDisk
  - Vendors launched new lines of personal computers with NAND flash SSD replacing hard disk
    - Apple, Samsung, and others
Market Trend in Prospect

- **Price drops quickly**
  - NAND flash is a lot cheaper than DRAM;
  - Still much more expensive than magnetic disk.
  - Annual drop in ASP/MB was about 60% in 2006.
  - Projected annual drop in ASP/MB is about 30-40% in next 5 years. [Eli Harari@SanDisk, August 2007]

- **Emerging Enterprise Market**
  - NAND ASP was $10/GB in 2007. With 40% annual drop, it could be $800/TB in 2012.
  - Not inconceivable to run a full database server on a computing platform with TB-scale Flash SSD as secondary storage.
Technology Trend in Prospect

- **NAND flash density increases faster than Moore’s law**
  - Predicted *twofold annual increase* of NAND flash density until 2012 [Hwang, ProcIEEE’03]
  - Toshiba hopes for 512GB SSD by the end of 2009
    - 30 nm chip-making process, Multi-level-cell (MLC)

- **Bandwidth catches up**
  - Samsung MCAQE32G8APP-0XA [2006]
    - Sustained read 56 MB/sec, sustained write 32 MB/sec
  - Samsung, Mtron [Feb. 2008]
    - Sustained read 100~120 MB/sec, sustained write 80~90 MB/sec
  - Intel-Micron’s 4-plane architecture + higher clock speed [Feb. 2008]
    - Sustained read 200 MB/sec, sustained write 100 MB/sec
  - Samsung MLC-based 256GB SSD with SATA-II [May 2008]
    - Sustained read 200 MB/sec, sustained write 160 MB/sec
Past Trend of Disk

• From 1983 to 2003 [Patterson, CACM 47(10) 2004]
  ▪ Capacity increased about 2500 times (0.03 GB → 73.4 GB)
  ▪ Bandwidth improved 143.3 times (0.6 MB/s → 86 MB/s)
  ▪ Latency improved 8.5 times (48.3 ms → 5.7 ms)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>CDC 94145-36</td>
<td>Seagate ST41600</td>
<td>Seagate ST15150</td>
<td>Seagate ST39102</td>
<td>Seagate ST373453</td>
</tr>
<tr>
<td>Capacity</td>
<td>0.03 GB</td>
<td>1.4 GB</td>
<td>4.3 GB</td>
<td>9.1 GB</td>
<td>73.4 GB</td>
</tr>
<tr>
<td>RPM</td>
<td>3600</td>
<td>5400</td>
<td>7200</td>
<td>10000</td>
<td>15000</td>
</tr>
<tr>
<td>Bandwidth (MB/sec)</td>
<td>0.6</td>
<td>4</td>
<td>9</td>
<td>24</td>
<td>86</td>
</tr>
<tr>
<td>Media diameter</td>
<td>5.25</td>
<td>5.25</td>
<td>3.5</td>
<td>3.0</td>
<td>2.5</td>
</tr>
<tr>
<td>Latency (msec)</td>
<td>48.3</td>
<td>17.1</td>
<td>12.7</td>
<td>8.8</td>
<td>5.7</td>
</tr>
</tbody>
</table>
Latency of Disk Lags

- Trend
  - In the time that bandwidth doubles, latency improves by no more than a factor of 1.2 to 1.4.
    - Latency improves by no more than square root of the improvement in bandwidth.
  - The bandwidth-latency imbalance may be even more evident in the future.

- The trouble is
  - Latency remains important for
    - Interactive applications, database logging (or whenever I/O must be done synchronously)

- What can NAND Flash Memory do for this?
### Magnetic Disk vs NAND Flash

- **Below is what the data sheets show**

<table>
<thead>
<tr>
<th></th>
<th>Sustained Transfer Rate</th>
<th>Average Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magnetic Disk</td>
<td>110 MB/sec</td>
<td>8.33 msec</td>
</tr>
<tr>
<td>NAND Flash SSD</td>
<td>56 MB/sec (read)</td>
<td>0.2 msec (read)</td>
</tr>
<tr>
<td></td>
<td>32 MB/sec (write)</td>
<td>0.4 msec (write)</td>
</tr>
</tbody>
</table>

- Magnetic Disk : Seagate Barracuda 7200.10 ST3250310AS
- NAND Flash SSD : Samsung MCAQE32G8APP-0XA drive with K9WAG08U1A 16 Gbits SLC NAND chips

  - Newer SSD products report much higher bandwidth for read and write
Characteristics of NAND Flash

- **No mechanical latency**
  - Flash memory is an electronic device without moving parts
  - Provides *uniform* random access speed without seek/rotational latency
    - Very low latency, independently of physical location of data
- **Asymmetric read & write speed**
  - Read speed is typically at least twice faster than write speed
    - (E.g.) Samsung 16 Gbits SLC NAND chips: 80 μsec vs 200 μsec (2 KB)
- **No in-place update**
  - No data item or page can be updated in place before erasing it first.
    - An erase unit (typically 128 KB) is much larger than a page (2 KB).
    - (E.g.) Samsung 16 Gbits SLC NAND chips: 1.5 msec (128 KB)
  - *Write (and erase) optimization* is critical
Flash SSD for Databases?

- **Immediate benefit for some DB operations**
  - Reduce commit-time delay by fast logging
  - Reduce read time for multi-versioned data

- **Still, many concerns to be addressed**
  - Random scattered I/O is very common in OLTP
    - Slow random writes by flash SSD can handle this?
      - *Flash-aware design of DBMS?*
      - *Flash-friendly algorithms?*
      - *Flash-friendly implementation?*
Transactional Log

SQL Queries

System Buffer Cache

Database
Table space

Transaction
(Redo) Log

Temporary
Table Space

Rollback
Segments
Commit-time Delay by Logging

- Write Ahead Log (WAL)
  - A committing transaction force-writes its log records
  - Makes it hard to hide latency
  - With a separate disk for logging
    - No seek delay, but …
    - Half a revolution of spindle on average
    - 4.2 msec (7200 RPM), 2.0 msec (15k RPM)
  - With a Flash SSD: about 0.4 msec

- Commit-time delay remains to be a significant overhead
  - Group-commit helps but the delay doesn’t go away altogether.

- How much commit-time delay?
  - On average, 8.1 msec (HDD) vs 1.3 msec (SDD) : 6-fold reduction
    - TPC-B benchmark with 20 concurrent users.
HDD vs SSD for Logging

- With SSD for log
  - CPU better utilized
    - By shortening commit-time, and serving more active transactions.
  - Leads to higher TPS
- Exaggerated by caching entire DB in memory
- TPC-B to stress-test logging
  - Transaction commit rate higher than TPC-C
Temporary Table Space

SQL Queries

System Buffer Cache

Database
Table space

Transaction
(Redo) Log

Temporary
Table Space

Rollback
Segments
Temp Data and Query Time

- Query processing often generates temp data
  - Sorts, joins, index creation, etc.
  - Typically bulky, performed in foreground; Direct impact on query processing time
- Typically stored in separate storage devices

- Ask the same question
  - What happens if SSD replaces HDD for temporary table spaces?
External Sort: I/O Pattern

- External Sort algorithm runs in two phases
  - Sorted run generation
    - Partitioned to chunks, sorted separately and, saved in sorted runs
    - Read sequentially from table space, written sequentially into temp space
  - Merging sorted runs
    - Read randomly from temp space, written sequentially into table space

- Dominant I/O patterns are sequential write followed by random read
  - No-in-place-update limitation is avoided.
  - These are flash-friendly I/O patterns!!
External Sort: Performance

- HDD vs SSD as a medium for a temp table space
  - Sort a table of 2 M tuples (200 MB), with 2 MB buffer cache
- SSD is good at *sequential write + random read*
  - Almost an order of magnitude reduction in merge times
One Less Tuning Knob?

- Cluster sizes for Sorting?
- With a larger cluster
  - Disk bandwidth improves (*by hiding latency*)
  - The amount of I/O may also increase due to *reduced fan-in* for merging sorted runs
- Flash SSD is
  - *With low latency*, not as sensitive to the cluster size
  - 2KB page was the best with the max fan-in
Hash-Sort Duality a Myth?

• The I/O pattern of hashing is said to be
  - random write (for writing hash buckets) + sequential read (for probing hash buckets)
  - As opposed to sort (sequential write + random read)

• If it’s the case, hashing is not flash-friendly.
  - Re-implement hashing to make it flash-friendly?
  - It appears already done by some vendors.
    - The observed I/O pattern was quite similar to that of sort (sequential write + random read)
Hash Join: Performance

- HDD vs SSD as a medium for a temp table space
  - Hash-join two tables of 2 M tuples (200 MB) each, with 2 MB buffer cache
  - About 3-fold reduction in join time
Rollback Segments

SQL Queries

System Buffer Cache

Database
Table space

Transaction
(Redo) Log

Temporary
Table Space

Rollback
Segments
MVCC Rollback Segments

• Multi-version Concurrency Control (MVCC)
  ▪ Alternative to traditional Lock-based CC
  ▪ Support read consistency and snapshot isolation
  ▪ Oracle, PostgreSQL, Sybase, SQL Server 2005, MySQL

• Rollback Segments
  ▪ When updating an object, its current value is recorded in the rollback segment
  ▪ To fetch the correct version of an object, check whether it has been updated by other transactions
  ▪ Each transaction is assigned to a rollback segment; old images of data are written to the rollback segment sequentially (in *append-only* fashion).
MVCC Write Pattern

- Write requests from TPC-C workload
  - Concurrent transactions generate multiple streams of append-only traffic in parallel (apart by approximately 1 MB)
  - HDD moves disk arm very frequently
  - SSD has no negative effect from no in-place update limitation
To support MV read consistency, I/O activities will increase
- A long chain of old versions may have to be traversed for each access to a frequently updated object

- Read requests are scattered randomly
  - Old versions of an object may be stored in several rollback segments
  - With SSD, 10-fold read time reduction was not surprising
Database Table Space

SQL Queries

System Buffer Cache

Database
Table space

Transaction
(Redo) Log

Temporary
Table Space

Rollback
Segments
Workload in Table Space

- **TPC-C workload**
  - Exhibit little locality and sequentiality
    - Mix of small/medium/large read-write, read-only (join)
  - Highly skewed
    - ~80% of accesses to 20% of tuples
- **Write caching not as effective as read caching**
  - Physical read/write ratio is much lower than logical read/write ratio
- **All bad news for flash memory SSD**
  - Due to the *No-in-place-update* limitation
  - *In-Page Logging (IPL)* approach [SIGMOD’07]
Concluding Remarks

- Clear and present evidences that Flash memory SSD can co-exist or even replace Magnetic Disk
  - Even now for logging, rollback segments and temp table spaces
  - Write optimization needed for database table spaces
- Flash-Aware DBMS Design is a must!
  - Flash-friendly algorithms, flash-friendly implementations
  - Need fresh new look at almost everything: Buffer management, B-trees, Sorting and Hashing, Self-Tuning, File Systems, etc.