Using Storage Class Memory for Extreme Performance In-Memory Computing

Presented at the Percona Performance Conference 2009

Earle F. Philhower, III
Technical Marketing Manager
Virident Systems
earle@virident.com
http://www.virident.com
Purpose of this talk

• Virident backgrounder

• Explain Storage Class Memory and In-Memory Computing
  — What it is and why you should care

• Share some of the lessons we have learned

• Describe Virident’s optimization of MySQL, Memcached, other applications
  — In-Memory Computing accelerates different application classes

• Get you excited about the possibilities of 1TB and beyond!
  — There are dozens of new areas to leverage Storage Class Memory to improve application performance
Virident overview

• Vision: SmartScaling the Internet Data Center
  — Deliver orders of magnitude higher performance, cost, and energy efficiency with tight integration of Silicon, Systems and Software

• Founded in 2006 by notable Silicon Valley veterans
  — Multi-disciplinary team with expertise in Servers, Software and Silicon from Intel, Sun, Google, Apple, SGI
  — Well-funded by venture and corporate capital

• Virident products:
  — GreenCloud Servers for Web Databases (MySQL)
  — GreenCloud Servers for Web Caching (memcached)
  — GreenCloud servers for Search, Analytics, Grids
Virident GreenCloud Servers

<table>
<thead>
<tr>
<th>Memcached</th>
<th>MySQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>• M100</td>
<td>• Q100</td>
</tr>
<tr>
<td>• M300</td>
<td>• Q300</td>
</tr>
<tr>
<td>• M500</td>
<td>• Q500</td>
</tr>
</tbody>
</table>

1U / 2U Rack Mount Chassis
- 1 – 4 Quad-Core CPUs
- 32 – 64 GB SDRAM
- 64 – 512GB Storage Class Memory
- Dual/Quad GigE Network Ports
- Up to 6 15K RPM SAS 73/146GB Disks
- RAID 5 Controller
- Linux Operating System
What is Storage Class Memory (SCM)?

• SCM is **L - A - R - G - E**
  — Memory is the new disk! 512GB...1TB+ no sweat!

• SCM is **Persistent**
  — New memory technologies are inherently persistent

• SCM is **Randomly Accessible**
  — Word addressable directly by the processor, no OS interaction or DRAM buffering required

• SCM can be **Asymmetric**
  — Read and write speeds may not be equal, requiring application optimization
How Virident utilizes Storage Class Memory for In-Memory Computing acceleration

• We developed Storage Class Memory
  — But we’re not a memory vendor...

• We developed optimized In-Memory Computing applications for SCM
  — But we’re not an applications vendor...

• We combine the two to provide systems value
  — Co-optimize hardware and software to provide highest levels of performance while reducing power requirements in the datacenter
What about SATA/SAS/PCIE SSDs?

- SCM is not SSD - not even if you put swapfile or MMAPed files on it

- SSDs for swap is a great idea for some applications, but not all
  - Response time is not critical
  - Lower performance requirements

- MMAP/Swap uses DRAM for buffering, OS for mediation
  - Some accesses proceed at DRAM speed (100ns)
  - Unlucky accesses proceed at disk/SSD speed (50...15000 us)
  - Unpredictable latency means unpredictable application response time
What are the available technologies?

<table>
<thead>
<tr>
<th>Attribute</th>
<th>DRAM</th>
<th>NOR Flash</th>
<th>DRAM + NAND Flash</th>
<th>PCM (Phase change)</th>
<th>NAND Flash SSD</th>
<th>Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacity</td>
<td>10's of GB</td>
<td>~350 ns, 2-5 GB/s</td>
<td>100's of GB – 0.5 TB</td>
<td>~250 ns, 2-5 GB/s</td>
<td>1-2 TB</td>
<td>1-10 TB</td>
</tr>
<tr>
<td>Read performance</td>
<td>~100 ns, 10+ GB/s</td>
<td>~250 – 500 ns, 2-5 GB/s</td>
<td>~50 us, 2 GB/s</td>
<td>~5 us, 2-5 GB/s</td>
<td>10 ms, ~200 MB/s (seq)</td>
<td>~10 MB/s (random)</td>
</tr>
<tr>
<td>Write performance</td>
<td>~100 us, 200 – 400 MB/s</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Persistence</td>
<td>No</td>
<td>Yes</td>
<td>PCM</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

©2009 Virident Systems Inc.
Why is Storage Class Memory important?

• Multicore processors are here to stay
  – Quad-core today, 6- and 8-core in near future

• BIG problem: What to do with all those cores?
  – Gulf between processor speed and storage speed shrunk 100x with SSDs...
  – ...but as multicores and CPU speeds increase the gulf is still growing

• SCM **narrows** the distance from CPU to large data sets by 100x over SSDs

• SCM **increases** the amount of data available per CPU by 10x over DRAM

• SCM **decreases** the number of servers required to store a fixed set of data
Why you should care

You’re going to have to worry about SCM sooner or later...

- Memory per system is not going to go down
- Performance requirements are not going to decrease
- Datacenter power and cooling are not getting cheaper

...Why not get a jump on the problem?
Table stakes

• 64-bits, of course
  – You’d be surprised how many custom applications out there still have issues.

• Multithreaded and Scalable
  – One does not necessarily imply the other.
  – “NO NEW GLOBAL MUTEXES”
  – IO bottleneck removal often brings up other, more subtle bottlenecks.

• Amenability to potentially asymmetric R/W nature of SCM
  – Pure write-dominated workloads may not be optimal with all SCM technologies
  – Algorithms often modifiable into a “read-mostly” portion and a “write-mostly” subset
Best practices

• Put all your data into SCM
  – Even the fastest SSD read is >100x slower than a processor memory access

• Read only what you need, when you need it
  – SCM is not a disk, not a block device
  – If you only need 1 byte of data, just read 1 byte

• Don’t let the OS get in your way
  – Use direct memory accesses whenever possible
  – An OS call is a signal to the kernel to put you to sleep!

• When present, use the asymmetry to your advantage
  – Most workloads are great matches (read-mostly)
  – For others (write-mostly) stage updated data in DRAM
Optimizing InnoDB for In-Memory Computing

- **Problem:** Even after threading optimizations, MySQL is IO bound
  - More threads doesn’t help if you’re stuck waiting for OS to give you data

- InnoDB IO is based on standard sized buffers
  - Entire page must be accessible before work can continue
  - But most queries only touch a small portion of data in that buffer

- **Solution:** *Read only what you need, when you need it*
  - Minimal changes to access DB stored in SCM
  - No file format changes necessary for speedup using SCM
  - When InnoDB requests a buffer, return a pointer to pre-mapped buffer in SCM
  - No copying of data into DRAM buffer pool
  - InnoDB core algorithms can remain unmodified and read only required data
Up to 100x less data transferred per query

SELECT * from UID
where USERID=151;
100x faster access to data in database

Conventional page buffer requests
- MySQL/InnoDB
- File System
- Device Driver (libata/FTL/etc.)
- Block Storage
  
  User/Kernel Transition
  Variable Delay
  60,000 - 10,000,000 ns

Storage Class Memory page buffer requests
- MySQL/InnoDB
- File System
- SCM Device Driver
- Storage Class Memory
  
  User/Kernel Transition
  Direct Processor Access
  100-300ns
Results of InnoDB optimization

- 50-70x+ performance versus Industry Standard Server with hybrid disk/DRAM configuration on third party benchmarks

- 7x+ versus fastest PCIe-based SSD systems
  - In-memory algorithm means more work can be done with less data transfer, lower latency

- Available on our website
  - Third-party benchmark information available at www.virident.com

GreenCloud MySQL/InnoDB Performance:
Workload Modeled on Yahoo del.icio.us

©2009 Virident Systems Inc.
Optimizing Memcached for SCM

- **Problem:** Memcached can use tons of memory but it likes to do frequent, small updates to regions (hash tables, pointer chains, etc.)
  - Hmm ... not optimal for SCM with asymmetric read/write profile?

- **Solution:** *Exploit the asymmetry of SCM to application advantage*
  Use SCM to store longer-lived, lesser-modified structures
  - DRAM is great for short-lived data, metadata, etc.
  - SCM has the capacity to handle the long-tail storage requirements
  - Aging algorithm already implemented (LRU) by memcached

- **Bottlenecks behind bottlenecks ...**
  - Global mutexes held while task switching (e.g. interrupt processing)
  - Network interrupt handling also needs care
Optimizing search applications for In-Memory Computing

- Problem: Search indexes spread across many servers (to keep data in DRAM)
  - SOLR and other make it easier, but still wasteful in terms of space, power, $$$

- Solution: Put all your data in SCM, and don’t let the OS get in the way
  Combine indices, use memory mapped FSDirectories
  - No warm-up time, warm-up queries needed!

- Further optimizations
  - Uncompress data layouts (VARINT lists, field data, etc.)
  - Index locking issues

- Beyond optimizing...enriching search relevance
  - Now the scorer can inexpensively access more metadata, and compute relative rankings by actually examining the documents side-by-side.
Additional uses for Storage Class Memory

• Traditional data-intensive applications...
  – Caching
  – Business Intelligence
  – Bio-informatics (Genomics, ...)
  – Clustering

• And beyond using the same old application algorithms...
  – Complete user history available instantly to a web application
  – Perform searches in more depth and with more meaning by examining more data
  – Pre-compute and store every likely user interaction for instant response times
  – And more...
Call to action

When (not if!) your servers have 1 terabyte or more of memory, how will it change the way you write applications?

In-Memory Computing powered by Storage Class Memory is here today.

www.virident.com
Thank You!