STORAGE INDUSTRY SUMMIT

The Future of Computing:
The Convergence of Memory and Storage through Non-Volatile Memory (NVM)

JANUARY 28, 2014, SAN JOSE, CA

James Pinkerton
Microsoft
Lead Partner Architect

OS Vendors: What NVM Means to Them

SNIA™
Solid State Storage Initiative

Why should NVM be Interesting to OS Vendors?

- New levels of performance for applications & OS
  - Example: File system metadata to substantially increase small file creations/sec
  - Example: New API for applications to unleash RAM-class latencies to storage (Byte Mode)
- Lower storage costs - Extreme IOPs of NVM can substantially reduce $/IOP
  - Example: Storage tiering - when combined with the capacity of HDD gives excellent $/IOP while maintaining great $/GB
  - Example: Power reduction
- New compelling distributed applications
  - Example: Distributed “in-memory” databases that are now non-volatile
  - Example: Remote storage with SMB3 RDMA has the same performance as local
    - **Windows Server 2012 R2 SMB3: 16.8 Gbytes/s, 560K 8 KB IOPs from a single client to SSD storage**
    - But as NVM technology is deployed, the bar is raised.
- Portability and new form factors for devices

Some Characteristics of Existing Non-Volatile Storage (from an App & Ops Perspective) 1/2

SNIA™ Solid State Storage Initiative

- Device Life Cycle
  - Failures
  - Replacement
  - Disposal
- Data Life Cycle
  - Backup
  - Replication
  - Security (privacy, etc..)
  - Data Integrity

- Application Semantics – which mode?
  - **Block mode** enables 100% app compat, unleashing compelling new capabilities
  - **Byte mode** (memory mapped files) has the potential for much higher IOPs, but what is the semantic of the interface?

What to support for NVM?
What to not support?

Some Characteristics of Existing Non-Volatile Storage (from an App & Ops perspective) – 2/2

- Errant program does not destroy all data
  - Most do not memory map a file
- There is “infinite” memory
  - Do we map paging file semantics? How does the OS tie into the infrastructure?

What to support for NVM?
What to not support?

Some Characteristics of Existing Non-Volatile Storage (from an OS perspective)

SNIA™ Solid State Storage Initiative

- What API to present to applications?
  - Block mode – everything "just works", but not optimal perf
  - Byte mode – fundamentally new API for applications
  - Either approach requires
    - Write error atomicity
    - Transactional Semantics
- What features to provide (or break) in the OS?
  - Defragmentation for optimized access
  - Virtualized Resources
  - Storage stack extensibility (filter drivers on Windows)
  - Data Reduction
  - Security
- Data Availability

Moving NVM to Cloud Scale

- Existing single system models for non-volatile data storage are insufficient
  - Data must be reliable **and available**
- Single system reliability is not a Cloud Scale design point
  - Availability implies replication of data so a single system outage does not prevent access to data
- Replication implies either
  - Operating System handles replication
  - Application handles replication (implies a platform API)
- Either way, replication must occur…

Thus storage and networking in one place actually makes sense :-)

Optimizing CPU/Latency in the real world

- Classic approach to lowest latency is for app/OS to poll
  - This approach can lead to ugly CPU utilization
- Classic approach to networking is interrupt moderation
  - For light load - Going to an interrupt based model can *increase* CPU utilization (context switch to handle interrupt)
  - For heavy load – nice reduction in CPU utilization, but typically higher latency for light and medium load.

Some relative numbers

- The platform
  - Westmere platform (so a little older hardware), tests read/write to memory
  - QDR InfiniBand (32 gbps) network, but PCI-E gen 2 ;-)
  - SMB3 RDMA, latency measured from user-mode application
  - You can guess which operating system :-)
- The Tests
  - Average latency doesn't matter to the cloud – **90th Percentile**
  - 8 threads, running different numbers of IO
- SMB3 used to replicate data
  - Server never allows client to RDMA directly into it's buffers – DOS
  - File Read is an RDMA Write (1+1 round trips) – early version of RDMA interface required deregister of RDMA buffers before Ack of IO
  - File Write is an RDMA Read (2+1 round trips)
    - Arguably for NVM both peers are trusted, thus File Read model is better to examine

Looking at 0 usec moderation Vary IO size

Single IO Latency (us)	1 KiB 0 usec moderation		8 KiB 0 usec moderation	
	90th Read	90th Write	90th Read	90th Write
Westmere	30	32	45	62

Old platform – Sandy Bridge/Ivy Bridge is faster

Numbers above are just illustrative – do not map to any shipping operating system

Read vs Write

Constant **50 usec delay**, vary queue depth

Total QD	8KiB Read (RDMA Write) 50us moderation IO/s	90th Rd	c/B	8KiB Write 50us moderation IO/s	90th Wr	c/B
8	36,300	**236**	13.32	31,600	**271**	14.9
16	94,300	176	8.91	75,200	231	10.03
24	141,300	179	7.99	99,400	263	9.14
32	168,700	201	7.78	112,400	318	8.82
40	180,600	233	7.64	119,300	384	8.51
48	186,900	269	7.58	124,600	440	8.56
56	188,900	311	7.58	128,000	503	8.46
64	192,500	355	7.56	130,200	561	8.54

Higher latency for writes due to **not** allowing client to RDMA into server (DOS concern)

Arguably not an issue for peer-to-peer communication.

Numbers above are just illustrative – do not map to any shipping operating system

Looking at 0 usec vs 50 usec moderation

	8KiB Read					
	0us moderation			50us moderation		
Total QD	IO/s	90th Rd	c/B	IO/s	90th Rd	c/B
8	87,300	80	9.98	36,300	236	13.32
16	151,200	98	9.43	94,300	176	8.91
24	175,800	120	9.09	141,300	179	7.99
32	186,400	154	8.92	168,700	201	7.78
40	186,300	190	8.97	180,600	233	7.64
48	185,800	232	8.93	186,900	269	7.58
56	186,300	290	8.89	188,900	311	7.58
64	186,300	333	8.9	192,500	355	7.56

Issue: Trade-off in latency vs. "clumping"

~3x latency with moderate level of interrupt moderation

takes QD=48 to get to parity on IOPs, better CPU

under reasonable load, better IOPs, better CPU

Numbers above are just illustrative – do not map to any shipping operating system

In Closing…

- Cloud scale storage systems require solving the availability problem
- Availability requires coupling NVM with networking
- Real world deployment requires thinking through
  - Security models (RDMA buffer DOS concerns)
  - Optimal network hardware interfaces
    - For buffer management
    - Interrupt moderation techniques, including scaling from light load to heavy load

"storage and networking" – who woulda thunk?