Creating Storage Class Persistent Memory With NVDIMM

PAUL SWEERE
Vice President, Engineering
paul.sweere@vikingtechnology.com
MEMORY/STORAGE HIERARCHY

- Data-Intensive Applications Need Fast Access To Storage
- Large Performance Gap Between Main Memory And HDD
- SSDs Have Narrowed The Gap, But A Gap Still Exists
- Opportunity For Innovation!

Performance Gap
NEW MEMORY LANDSCAPE

Arguably, Several Years Before SCM Is Ready For Broad Commercial Adoption

Storage Class Memory Technology Vision

- Non-volatile
- Speed of DRAM
- Infinite Write Endurance
- Dense
- Scalable, Low Cost
- Low Power

But, The Promise Of SCM Can Be Realized Today....
By Combining DRAM and NAND To Create A New Device -> NVDIMM...
NVDIMM

• “Hybrid” Memory Module Combining DRAM and NAND
  – Plugs Into JEDEC Standard DIMM Socket
• Leverages Beneficial Characteristics Of Each Memory Technology
  – Speed, endurance, and random byte addressability of DRAM
  – Non-volatility of NAND Flash
• Enables Main Memory Persistence
  – Data written to DRAM is preserved through system power loss
  – Main memory becomes non-volatile but operates at speed of DRAM
NVDIMM BLOCK DIAGRAM

NVDIMM

NAND Flash

NVDIMM Controller

JEDEC Standard DRAM And Logic

Bus Isolation

Supercapacitor

Power Management

DDR Host Power

Add/Cmd, Data

JEDEC Standard DIMM Edge Connector
NORMAL OPERATION: Operates from host power, data transferred b/w host and DRAM
RESTORE OPERATION: Host power restored, transfers data from NAND To DRAM
SAVE OPERATION: NVDIMM disconnects from host, transfers data from DRAM to NAND
NORMAL OPERATION: NVDIMM reconnects to host.
NVDIMM SYSTEM INTEGRATION

SAVE Trigger
- Which system events should trigger a SAVE?
- Incorporate appropriate H/W signaling

System BIOS
- Mapping NVDIMM into system memory
- Support for RESTORE operation

System Mgmt.
- Configure and monitor state of NVDIMM
- Monitor health of supercapacitors

Power / Mech.
- System power supply holdup requirements
- Mechanical mounting of supercap pack

Application I/F
- Accessing NVDIMM from application
SAVE OPERATION

AC → System Power Supply → System DC Voltage → CPLD → SAVE_Trigger → Processor / Chipset → SAVE_n → NVDIMM

A. System Power Supply
B. System DC Voltage
C. Powergood
D. Normal Execution
E. Flush Data
F. SR
G. Dead Loop

DC Holdup Time (1-12 msec)

Normal Mode: NORMAL
SAVE Mode: SR → SAVE

Processor

NVDIMM

SAVE_n
SYSTEM BIOS

- BIOS Modifications To Support RESTORE Operation
  - Distinguish NVDIMMs from standard DIMMs, map into E820 table, EFI memory map
  - Allow NVDIMMs in SAVE to complete before proceeding
  - Execute RESTORE on all NVDIMMs with valid SAVE data
NVDIMM OPERATION – SYSTEM PERSPECTIVE

- Initial OS Boot
- Load NVDriver, Map NV space
- Verify NVDIMM Ready
- Arm NVDIMM
- Periodic HM Checks
- NORMAL MODE / Not Armed
- NORMAL MODE / Armed
- Recover Data
- Boot OS, Load NVdriver, Map NV space
- Initiate RESTORE Operation
- OS
- BIOS
- Contains Restored Data
- Contains Valid SAVE Data
- SAVE Trigger Event
SUPERCAPACITOR HEALTH MONITORING

- Supercapacitor Aging
  - Capacitance declines, ESR increases
  - Overprovisioned for target application

- Health Monitoring Interface
  - Host can initiate health check via i2c interface or NVdriver API
  - Early warning and EOL status bits

Graph:
- % of Nominal
- Capacitance declines over time
- Beginning of Life (BOL)
- End of Life Warning 1
- End of Life Warning 2
- End of Life (EOL)
- Minimum Capacitance Req’d For SAVE
- Operational Life
- EOL
**ELECTROMECHANICAL**

- **Power Supply Holdup Time**
  - Non-cached NV space requires ~500 usec holdup
  - Cached NV implementation may require up to 10 msec

- **Supercapacitor Mounting Options**
APPLICATION INTERFACE

- **System Memory Map**
- **Mapping NVM Into Application**
  - Today: “ArxMalloc()”
    - Application is responsible for managing memory
  - Future: “nvmalloc()”
    - Library manages memory (e.g. shared access)

- **Data Access Method**
  - Byte: Direct access to NVM, no driver layer required
  - Block/File: port ramdisk or file system to NVM

- **Other Considerations**
  - Processor cache policy for NV space
  - Processor memory consistency model
  - Impact of reduced data access time
DATA PROTECTION

- **End-To-End ECC Protection**
  - ECC generated by memory controller and passed through to SSD on SAVE
  - Same ECC bits passed back to DRAM on RESTORE

- **Data Transfer Protection**
  - DDR paths: ECC, address parity
SYSTEM LEVEL RELIABILITY ENHANCEMENT

» Leverage Platform Reliability Features For Standard DIMMs
  - DIMM mirroring (e.g. Intel Ivybridge)
  - Intel machine check architecture
  - Memory scrubbing

» Software Techniques
  - Software RAID
  - Checkpointing / Journaling
Benchmark: VDBENCH, Platform: Intel Sandybridge, Linux, Two DDR3-1333 NVDIMMs as interleaved pair (channel interleaving), PRAMFS vs. SATA SSD as Linux block device
NVDIMM PERFORMANCE

Benchmark: VDBENCH, Platform: Intel Sandybridge, Linux, Two DDR3-1333 NVDIMMs as interleaved pair (channel interleaving), PRAMFS vs. SATA SSD as Linux block device
NVDIMM APPLICATIONS

» Storage Array Controller I/O Cache
  ▪ Description: I/O cache mapped to NVDIMM
  ▪ Benefit: Performance. Enables “early” acknowledgment on writes

» VDI
  ▪ Description: Cache VDI I/O in NVDIMM
  ▪ Benefit: Performance

» IMDB
  ▪ Description: Place database in NVDIMM
  ▪ Benefit: Performance increase, lower power by reducing disk I/O

» SSD Replacement
  ▪ Description: Accelerate access to metadata
  ▪ Benefit: Performance
INDUSTRY STANDARDIZATION

» NVDIMM Gaining Support From Key Ecosystem Players
  ▪ Intel announced NVDIMM support for Grantley at Spring 2013 IRUM
  ▪ Supermicro / Viking joint announcement in May, 2013

» JEDEC Hybrid Memory Task Group
  ▪ 12V and SAVE_n pins added to DDR4 DIMM socket
  ▪ 12V in DDR4 socket will simplify and improve efficiency of NVDIMM power circuitry, simplify cable routing

» SNIA NVM Programming TWG
NVDIMM ROADMAP

» Increased Standardization For DDR4
» Higher Capacities (16GB, 32GB)
» Expanded OS Compatibility
   » Windows, VMWARE
» Support For NVM Integrated Directly Into OS / Languages
   » Persistent variable support
   » NVM-optimized file systems
   » NVMalloc()
Questions?
Thank You