Design of PRAM-based Persistent NVDIMM Controllers to Prepare the Data Age

Myoungsoo Jung

Computer Architecture and MEMory systems Lab.
School of Integrated Technology
Yonsei University

Flash Memory Summit 2018
Santa Clara, CA
Data Explosion

- **1900** London library (500,000 books)
- **1992** Internet explosion
- **2002** Less than 0.1% is stored on paper
- **2005** 12 stacks of books from Earth to Sun
- **2015** 2 stacks of books from Earth to Pluto

**2020** 44 ZB

---

It comes as no surprise to any Facebook user that the social network gathers a considerable amount of information based on their actions and interests. But according to a report from ProPublica, the world’s largest social network knows far more about its users than just what they do online.

Data becomes the new currency

Flash Memory Summit

Facebook Privacy: Social Network Buys Data From Third-Party Brokers To Fill In User Profiles

BY AJ DELLINGER ON 12/28/16 AT 4:57 PM

2025
163 ZB

(IDC-2017)
Data Explosion
Past 20 years

2005  0.1 ZB
2010  1.2 ZB
2012  2.8 ZB
2015  8.5 ZB
2020  44 ZB
2025  163 ZB

Storage capacity shipped across all media types
(HDD, Flash, SSD, Tape, Optical, and DRAM)

GAP

2025  19 ZB

(IDC-2017)
What Can System/Architecture Help?

Create
163 ZB

Store
19 ZB

(IDC-2017)
What Can System/Architecture Help?

Create
163 ZB
(IDC-2017)

Store
19 ZB

DRAM
New Memory
PRAM
Memory Expansion
DRAM
ULL
NVMe SSD
Flash
SATA SSD
Flash
DISK

Flash Memory Summit 2018
Santa Clara, CA
What Can System/Architecture Help?

- Make media more reliable (ECC/LDPC)
- Addressing penalty
- Incarnating heterogeneous memory
- Hiding complexity
- Aggregating storage media
- Higher capacity
- Better performance
- Interface/Protocol

Store

Create 163 ZB (IDC - 2017)

New Memory

- DRAM
- PRAM

Memory Expansion

- DRAM

NVMe SSD

- Flash

SATA SSD

- Flash

Flash Memory Summit 2018
Santa Clara, CA
Cost scaling

**Cost Reduction Rate**

- **DRAM**
- **NAND**
- New Memory (PRAM, STT-MRAM, ReRAM)

**Scaling limitation**

- Managed DRAM
- 3D NAND

**Time & Technology node**

Source: SK Hynix
Memory Replacement

New Memory: PRAM

- DDR
- DDR
- DDR/PCIe
- PCIe
- SATA

Memory Expansion:
- DRAM
- ULL

Other Storage Options:
- NVMe SSD
- Flash
- SATA SSD
- Flash

- PCH
- DISK
General Assumptions of PRAM Latency

Many previous works assume PRAM’s write latency as similar to or slightly worse than DRAM (1.5x)
Our performance measurement on real 3x nm PRAM exhibits expensive write latency than DRAM (190x).
New Memory Placement

- PRAM offers promising read performance, but terrible write latency, compared to DRAM
What Are The Considerations To Design Hybrid Memory (DRAM+PRAM) Controller?

To get insights of controller design, let’s understand the details of DRAM and PRAM
**DRAM’s Multi-bank Architecture**

- **Multiple banks** to serve multiple memory requests in parallel
- **Single row buffer** within a bank

### Diagram Details

- **Row decoder**
- **Column decoder**
- **Row buffer**

---

Flash Memory Summit 2018
Santa Clara, CA
Does PRAM have the Same Internal Architecture with DRAM?

**Challenge 1**: PRAM’s write latency is long
- PRAM employs multiple row buffers

**Challenge 2**: PRAM’s asymmetric latency incurs lots of bank conflicts
- PRAM uses multi-partition architecture
PRAM’s Multi-partition Architecture

- Multiple partitions **within the bank** for partition-level parallelism
- Multiple row buffers to mitigate long write latency
Can a conventional DRAM controller be aware of multi-partition? *(Inside of a Bank)*

(Revisited) Conventional DRAM scheduler just utilizes **bank-level parallelism**. Cannot see inside of bank!

∴ **Partition-level parallelism** should be supported
How Memory Requests Can Be Scheduled By Exploiting Multi-Partition Architecture?

**Limitation of PRAM**: In PRAM design, WRITE request blocks whole PRAM bank.
How Memory Requests Can Be Scheduled By Exploiting Multi-Partition Architecture?

**Key insight:** Although WRITE cannot be serviced, READs can be serviced if partition number is different.
Non-Blocking Read Service (NBRS)

**Solution**: Add a *register* to store ‘partition number of WRITE’ and compare it with partition number of incoming READ request.

*Same partition READ can’t be serviced!*
Design1: **Scheduling Support Module** for PRAM-aware New Scheduling Scheme
Now Requests Are Scheduled. Then, How Then Can It Serve to Hybrid Memory?

Firstly, as is generally known, LPDDR is JEDEC standard low-power memory interface (used for DRAM)
1) Activation: activate target row & write that data to row buffer
2) Read/Write: accessing row buffer with column address
3) Precharge: charge half-voltage of bit-line
Does PRAM have the Same Memory Interface (LPDDR2) with DRAM?

(Revisited) PRAM has a different architecture with DRAM such as **Multiple row buffers** and **More larger capacity**

∴ Different interface is required
PRAM’s Timing

- PRAM requires different timing model from DRAM
  - NVM memory space is much larger than a DRAM
- 3-Phase addressing (LPDDR-NVM by JEDEC)

1. Pre-active
   - Row Address Buffer (RAB) is selected by the memory controller

2. Activate
   - Lower row address
   - PRAM array
   - Row Data Buffer (RDB) is also selected by the memory controller

3. READ
   - Column address
   - Data out
Design2: Heterogeneity Support Module for both LPDDR & LPDDR-NVM

Our own new physical layer (400MHz) is implemented
Don’t Forget DRAM is For Cache. Then, How Caching Can Be Supported?

**Solution:** Keep which data exist in DRAM (caching info) in lookup table.

Moreover, like conventional cache, controller should have algorithms such as DRAM dataline update, eviction, and find empty dataline.
Hardware Support for Lookup Table

Lookup table do not include data value, includes address information

Only 512KB BRAM is used for lookup table

With multiple comparators, 4 ways can be parallelized
Design3: **Caching Support Module** for Use DRAM As Inclusive Cache of PRAM
BTW, How Non-Volatility of PRAM Can Be Maintained Although DRAM Is Integrated? (Hybrid)

Challenge of hybrid memory: Data in DRAM will disappear when there is a power failure
**Solution**: Provide *Flush* operation which moves DRAM data to PRAM. Memory controller generates ‘PRAM write’ request corresponding to the target DRAM row.

NOTE) ‘PRAM write’ will be stored in command queue which exists in memory controller. And DRAM dataline is invalidated.
Okay, Data Delivery Is Guaranteed. Is It Good Enough?

**Challenge of flush**: User believes data has the latest value. But, the memory controller can reorder the order of memory request.
Solution: Provide *Fence* operation to enforce data delivery order of memory requests. The memory controller can simply add ‘*fence flag*’ to check fenced or not.
Design4: **Persistent Support Module** to Guarantee Data Delivery & Delivery Order
Performance comparison (memory access with synthetic, read-write inter-mixed trace)
Demo – Slow Version

- DRAM
- PRAM
- P-NV
- DIMM

Host sent 1M memory requests
Memory services incoming requests
Still PRAM servicing memory requests
Demo – Normal Version

- **DRAM**
- **PRAM**
- **P-NV DIMM**

Performance Times:
- 0.105 sec
- 1.27 sec
- 0.468 sec
Further Enhancement

New Memory Controller

- FPGA CTL (CH1)
- Hybrid Cache Logic
- Customized PRAM Phy

New Memory Device

- PRAM chip
- PRAM chip
- PRAM chip
- PRAM chip

PRAM-based NVDIMM

On-chip MC1

On-chip MC2

L2 cache
Further Enhancement

New Memory Controller

FPGA CTL (CH0)
- Transl. Mode Reg.
- Addr Rreg.
- Mem CTL Logic
- Initializer
- CMD generator
- Datapath

FPGA CTL (CH1)

New Memory Device

PRAM PHY
- cs_n
- CMD
- Addr.
- CLK
- dq[15:0]

PRAM chip

PRAM-based NVDIMM

On-chip MC1
- L2 cache
- RD/WR Req.
- MCU

On-chip MC2
- [127:0]
- Rwdata[127:0]

MCU

Server

[Image of PRAM-based NVDIMM]

http://camellab.org

CAMEL & MEMRAY
Prototype Results (APEX MAP for HPC)
Real-Time Data

Number of interactions/capita/day

<table>
<thead>
<tr>
<th>Year</th>
<th>Interactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010</td>
<td>85</td>
</tr>
<tr>
<td>2015</td>
<td>218</td>
</tr>
<tr>
<td>2020</td>
<td>601</td>
</tr>
<tr>
<td>2025</td>
<td>4,785</td>
</tr>
</tbody>
</table>

Internet of Things

Zettabytes

Flash Memory Summit 2018, Santa Clara, CA
Amazon won't say if it hands your Echo data to the government

Amazon has a transparency issue with its flagship Echo, an "always listening" speaker, which collects vast amounts of customer data that's openly up for grabs by the government.

But Amazon's bi-annual transparency figures don't want you to know that.

In fact, Amazon has been downright deceptive in how it presents the data.
Endpoint

• Data should be tagged and classified by criticality
  • Security is matter – the private data should be managed by local or endpoint part

• Can we make each device as a standalone accelerator?
  • IoT leverages very power-limited devices
  • Even OS or file system can be a burden for IoT!
Storage-based Accelerator!

Integrate NVM chips into embedded coprocessor.

All cores are directly connected to storage.

Process data within an accelerator.
Overall Architecture

Many-core Host

Core  Core  Core

Core  Core  Core

North Bridge

Memory

IO Controller

SSD

Accelerator

Processor

Network

Storage

Heterogeneous Platform
NearZero architecture

Host

Interface

Crossbar Network

Controller

Processor

PRAM Storage
NearZero (Results)

Read-intensive (processing power/energy)

Write-intensive (processing power/energy)

Cycle Reduction!

Complete (NearZero)

Complete (withDRAM)

Complete (SLC-flash)

Complete (Old-PRAM)
One More Direction w/ PRAM

- Full FPGA Automation for SCM-based Storage
- Performance

Latency is sustainable for all random and sequential access patterns

<table>
<thead>
<tr>
<th>Type</th>
<th>NVMe SSD (ASIC)</th>
<th>New Memory SSD (ASIC)</th>
<th>Our Prototype (FPGA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bandwidth (read/write)</td>
<td>1.1~2.3GB/sec</td>
<td>1.5~3.5GB/sec</td>
<td>2.5 ~ 5.2 GB/sec</td>
</tr>
<tr>
<td>Latency</td>
<td>15 ~ 150 us</td>
<td>8 us ~ 100 us</td>
<td>11~13 us</td>
</tr>
</tbody>
</table>