All Tomorrow’s Memories

Bruce Jacob
Keystone Professor
University of Maryland
Stealth Revolution

All Tomorrow's Memories
Bruce Jacob
University of Maryland

SLIDE 2
Stealth Revolution

Application

Operating System

VM + FS

Page Mapping
Garbage Collection
Wear Leveling
Other
FTL Functions
Fusion-ish SSD
Stealth Revolution

Application

Operating System
VM + FS
Page Mapping
Garbage Collection
Wear Leveling

Fusion-ish SSD

Other FTL Functions
Stealth Revolution

All Tomorrow’s Memories
Bruce Jacob
University of Maryland

SLIDE 3
Stealth Revolution
Stealth Revolution

All Tomorrow's Memories
Bruce Jacob
University of Maryland

[FPGA]

[DATA]
Stealth Revolution

DATA

Device Protocol

[DATA]

FPGA
Stealth Revolution

Question: where is data?
Answer: <data owner>
To data owner: read X
To requester: mem[X]
Stealth Revolution

Question: where is data?
Answer: <data owner>
To data owner: read X
To requester: mem[X]

Major implications for OS and applications
Stealth Revolution

Question: where is data?
Answer: <data owner>
To data owner: read X
To requester: mem[X]

Major implications for OS and applications

(esp. considering Fusion-io like capabilities)
Background: Wish List

- Fine-Grained Access
- Bandwidth
- Capacity
- Low Power
- Nonvolatility

- DRAM - HBM/HMC*
- Flash, 3DXP, RRAM, PCM, etc - NVMM*
- HBNV*

*Things we did and/or are doing now (I’ll cover in talk)
Background: Wish List

- Fine-Grained Access
- Bandwidth
- Capacity
- Low Power
- Nonvolatility
- DRAM - HBM/HMC*
- Flash, 3DXP, RRAM, PCM, etc - NVMM*
- HBNV*

Major implications for OS* and applications

*Things we did and/or are doing now (I’ll cover in talk)
Background: Memory Latency

Cost of access is high; requires significant effort to amortize this over the (increasingly short) payoff.
Background: Memory Latency

Cost of access is high; requires significant effort to amortize this over the (increasingly short) payoff.
Hybrid Memory Cube

Off-chip: high speed SerDes and generic protocol

4 I/O Ports, up to 80 GB/s each

Next gen is 160 GB/s per (640 total)

Total conc’y = 16 x 8 x 2..8 (256–1024)
High Bandwidth Memory

Uses a simple ‘2.5D’ instead of full 3D stacking

- TSV Stack
  - Up to 4 or 8 DRAM dies

- 1024-bit x 2Gtps = 256 GB/sec

- HBM DRAMs
  - Interface
  - TSV Interposer

- 1024-bit 8-Channel Wide Interface

- GPU/CPU

All Tomorrow’s Memory Systems

Bruce Jacob

University of Maryland
High Bandwidth Memory

Each Link is 128 Bits Wide: \(1024\) Total
Performance Comparison

MEMSYS 2018

All Tomorrow's Memories
Bruce Jacob
University of Maryland
SLIDE 10
Non-Volatile Main Memory

<table>
<thead>
<tr>
<th>Memory Technology</th>
<th>Cost for 10 GB</th>
<th>Size of 10 GB</th>
<th>Power for 10 GB</th>
<th>Power per GB/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Off-Chip SRAM</td>
<td>$1,000</td>
<td>1 bucket</td>
<td>0.1–1 W</td>
<td>0.1 W</td>
</tr>
<tr>
<td>DDR4 SDRAM</td>
<td>$100</td>
<td>1 DIMM</td>
<td>1 W</td>
<td>0.1 W</td>
</tr>
<tr>
<td>NAND Flash</td>
<td>$10</td>
<td>&lt;1 chip</td>
<td>0</td>
<td>0.1 W (?)</td>
</tr>
<tr>
<td>3D XPoint</td>
<td>$40</td>
<td>&lt;1 chip</td>
<td>0</td>
<td>0.1 W (?)</td>
</tr>
</tbody>
</table>

**Note:** wear-out mitigated by using MANY devices (thousands). A single device would wear out in under two days; therefore, 1000 devices should last for at least a year.

Next, you can trade off longevity for access time and wearout: if the data need only last hours or minutes, wearout is reduced.

---

**Diagram**

- CPU
  - DDRx SDRAM Main Memory
  - NAND Flash Main Memory (… or *any* source of cheap bits)
  - DDRx SDRAM Last-Level Cache
A Tale of 3 Memory Systems

- **SSD**
  - $500 – 10W

- **NVMM**
  - $500 – 10s of W

- **Ideal**
  - $10,000 – 100W
A Tale of 3 Memory Systems

SSD

APP

read()

write()

OS

FS

FTL

Flash

NVMM

APP

LW

SW

32G DRAM

FTL

Flash

Ideal

APP

LW

SW

1TB DRAM
This is when we realized how good Linux is at prefetching out of SSDs.
Yeah, it’s a lot of engineering

DRAM Cache & Flash MM Controller (FTL)
Yeah, it’s a lot of engineering.

**HMC:**
- 320GB/s
- 16 channels

**HBM:**
- 256GB/s
- 8 channels

Crossbar ReRAM

Intel/Micron 3DXP

~100x faster than flash
High Bandwidth Non Volatiles

Borrow a page from the HMC playbook

Network Fabric

MC MC MC MC MC MC ...

NV ReRAM: up to 1000ns expected*
*trade-offs?
Crossbar 3D ReRAM

- Cells minimum area (no access transistor)
- 2-stack arrays @ 16nm, 20 x 20 mm die: 64GB of ReRAM
- 8-stack arrays => 256 GB of ReRAM
- Stacks arbitrarily high
No Access Transistor

1T1R Memory Array
Low Latency, Low Density

1TnR Memory Array
High Performance, High Density

(n = 1 .. 2048)

Crossbar RRAM Technology
No Access Transistor
No Access Transistor

(n = 1 .. 2048)

1TnR
(n=8)
For \( n = 2048 \) area is \(~75\%\) white space

Use for processor (cores, controllers, routers, NoC, etc.)
No Access Transistor

Monolithic
Not Die-Stacked

For \( n = 2048 \) area is \(~75\%\) white space

Use for processor (cores, controllers, routers, NoC, etc.)

\( (n = 1 \ldots 2048) \)
Recall: Real DRAM Latency Is Actually Quite Long (100ns)

This is for single core. Multicore can be much, much worse.
Example Monolithic Numbers

~64 cores, ~256GB ReRAM, ~4k banks

Assume 200ns latency for 8-byte payload:

\[
\text{Bandwidth} = \frac{4k \times 8 \text{ bytes}}{200\text{ns}}
\]

\[
= 4k \times 40 \text{ MB/s}
\]

\[
= 160 \text{ GB/s}
\]

e.g., 64 cores, each 4-way multithreaded, each with 512-bit (8-longword) SIMD, vectored & scatter-gather loads, 4-deep non-blocking => 8k
So what all does this enable?

**HBM/HMC**: hugely parallel systems (the duality of bandwidth and parallelism), streaming applications, 2x performance

**NVMM**: massive data sets, new OS paradigms such as merged VM+FS and journaled main memory (built-in checkpoint/restart)

**HBNV**: fine-grained operations on enormous sparse data sets
Datacenter & Cloud Issues

Distribution at storage-level interface simplifies application development

Potential for significant performance

RoCE appropriate for supercomputing?
How about RoXX?

At what round-trip latency does this rival MPI as programming model?

Expect a shake-up soon.
Nonvolatility Issues

Unified VM+FS Subsystems (OS redesign)

- By default, data in process address space temporary, garbage-collected at exit(); permanentify function to keep around

- Possible directions:

  - Persistent objects (e.g. Mneme, POMS) [failed only due to reliance on disk]
  - Named regions

- Journaled main memory w/ checkpointing
Capacity Issues

Rethink Protection & Translation

- TLB overhead is ~20%
  - So get rid of it already!
  - BUT: need protection, authentication

- Why not waste bits? Simplify both sharing and translation by eliminating much of VM

- OS/HW co-design needed: e.g., sharing via vaddr instead of paddr, language support?

Recall: Nonvolatile main memories ~TB per node
Bottom Line

It’s going to happen. :)

• Combined VM+FS subsystems
• Journaled main memory
• Persistent Object Store work from 80s
• OS: Simpler design, fewer potential bugs
• VM arguably a way better abstraction to distribute than the FS
• Monolithic = good for many applications
... and the storage guys are showing us the way! Monolithic = good for many applications. Combined VM+FS better abstraction than the FS. Persistent Object Store work from 80s. Journaled main memory. VM arguably a way better abstraction than the FS. It's going to happen. :)

- Combined VM+FS
- Journaled main memory
- Persistent Object Store work from 80s
- Monolithic = good for many applications
- VM arguably a way better abstraction than the FS
Shameless Plug

Washington DC Oct 2019

The IEEE International Symposium on Memory Systems 2018
Thank You!

Bruce Jacob
blj@umd.edu
www.ece.umd.edu/~blj