Memory Systems and Memory-Centric Computing Systems

Part 1: Memory Importance and Trends

Prof. Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu
3 February 2020
Champbery Winter School
Brief Self Introduction

Onur Mutlu

- Full Professor @ ETH Zurich CS (EE), since September 2015
- Strecker Professor @ Carnegie Mellon University ECE/CS, 2009-2016, 2016-
- PhD from UT-Austin, worked at Google, VMware, Microsoft Research, Intel, AMD
- https://people.inf.ethz.ch/omutlu/
- omutlu@gmail.com (Best way to reach me)
- https://people.inf.ethz.ch/omutlu/projects.htm

Research and Teaching in:

- Computer architecture, computer systems, hardware security, bioinformatics
- Memory and storage systems
- Hardware security, safety, predictability
- Fault tolerance
- Hardware/software cooperation
- Architectures for bioinformatics, health, medicine
- ...
Current Research Focus Areas

Research Focus: Computer architecture, HW/SW, bioinformatics, security

- Memory and storage (DRAM, flash, emerging), interconnects
- Heterogeneous & parallel systems, GPUs, systems for data analytics
- System/architecture interaction, new execution models, new interfaces
- Hardware security, energy efficiency, fault tolerance, performance
- Genome sequence analysis & assembly algorithms and architectures
- Biologically inspired systems & system design for bio/medicine

Broad research spanning apps, systems, logic with architecture at the center
Four Key Directions

- Fundamentally **Secure/Reliable/Safe** Architectures

- Fundamentally **Energy-Efficient** Architectures
  - Memory-centric (Data-centric) Architectures

- Fundamentally **Low-Latency** Architectures

- Architectures for **Genomics, Medicine, Health**
A Motivating Detour: Genome Sequence Analysis
Our Dream (circa 2007)

- An embedded device that can perform comprehensive genome analysis in real time (within a minute)
  - Which of these DNAs does this DNA segment match with?
  - What is the likely genetic disposition of this patient to this drug?
  - . . .
What Is a Genome Made Of?

The chromosome is made up of genes

The genes consist of DNA

Cell

Nucleus

Chromosome - 23 pairs

SAFARI  The discovery of DNA's double-helical structure (Watson+, 1953)
DNA Under Electron Microscope

human chromosome #12 from HeLa’s cell
DNA Sequencing

- **Goal:**
  - Find the complete sequence of A, C, G, T’s in DNA.

- **Challenge:**
  - There is no machine that takes long DNA as an input, and gives the complete sequence as output.
  - All sequencing machines chop DNA into pieces and identify relatively small pieces (but not how they fit together).
Untangling Yarn Balls & DNA Sequencing
Genome Sequencers

Roche/454
AB SOLiD
Illumina HiSeq2000
Pacific Biosciences RS
Illumina MiSeq
Oxford Nanopore MinION
Illumina NovaSeq 6000
Oxford Nanopore GridION
Ion Torrent PGM
Ion Torrent Proton
Complete Genomics

... and more! All produce data with different properties.
The Genomic Era

- 1990-2003: The Human Genome Project (HGP) provides a complete and accurate sequence of all DNA base pairs that make up the human genome and finds 20,000 to 25,000 human genes.
The Genomic Era (continued)

development of high-throughput sequencing (HTS) technologies

Number of Genomes Sequenced

1 Sequencing

2 Read Mapping

3 Variant Calling

4 Scientific Discovery

Reference: TTTATCGCTTCCATGACGCAG
read1: ATCGCATCC
read2: TATCGCATC
read3: CATCCATGA
read4: CGCTTCCAT
read5: CCATGACGC
read6: TTCCATGAC

Billions of Short Reads

Short Read Alignment

Reference Genome
Example Question: If I give you a bunch of sequences, tell me where they are the same and where they are different.
Genome Sequence Alignment: Example

Source: By Aaron E. Darling, István Miklós, Mark A. Ragan - Figure 1 from Darling AE, Miklós I, Ragan MA (2008). "Dynamics of Genome Rearrangement in Bacterial Populations". PLOS Genetics. DOI:10.1371/journal.pgen.1000128., CC BY 2.5, https://commons.wikimedia.org/w/index.php?curid=30550950
### The Genetic Similarity Between Species

<table>
<thead>
<tr>
<th>Species Combination</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human ~ Chimpanzee</td>
<td>96%</td>
</tr>
<tr>
<td>Human ~ Cat</td>
<td>90%</td>
</tr>
<tr>
<td>Human ~ Cow</td>
<td>80%</td>
</tr>
<tr>
<td>Human ~ Banana</td>
<td>50-60%</td>
</tr>
<tr>
<td>Human ~ Human</td>
<td>99.9%</td>
</tr>
</tbody>
</table>

Safari
Metagenomics, genome assembly, de novo sequencing

Question 2: Given a bunch of short sequences, can you identify the approximate species cluster for genomically unknown organisms?

uncleaned de Bruijn graph

http://math.oregonstate.edu/~koslickd
### Sequencing

- **Illumina HiSeq4000**
  - **300 M bases/min**

### Read Mapping

- **Billions of Short Reads**
  - GAGTCAGAATTTGAC
  - GAGTCAGAATTTGAC
  - GAGTCAGAATTTGAC
  - GAGTCAGAATTTGAC
  - GAGTCAGAATTTGAC

**Bottlenecked in Mapping!!**

- **on average**
  - **2 M bases/min**
  - **(0.6%)**
Need to construct the entire genome from many reads
Read Mapping

- Map many short DNA fragments (reads) to a known reference genome with some differences allowed.

Mapping short reads to reference genome is challenging (billions of 50-300 base pair reads)
Read Alignment/Verification

- **Edit distance** is defined as the minimum number of edits (i.e. insertions, deletions, or substitutions) needed to make the read exactly match the reference segment.

NETHERLANDS x SWITZERLAND

<table>
<thead>
<tr>
<th>NE</th>
<th>T</th>
<th>H</th>
<th>E</th>
<th>R</th>
<th>L</th>
<th>A</th>
<th>N</th>
<th>D</th>
<th>S</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW</td>
<td>I</td>
<td>T</td>
<td>Z</td>
<td>E</td>
<td>R</td>
<td>L</td>
<td>A</td>
<td>N</td>
<td>D</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>match</th>
<th>deletion</th>
<th>insertion</th>
<th>mismatch</th>
</tr>
</thead>
</table>

SAFARI
Challenges in Read Mapping

- Need to find many mappings of each read
  - How can we find all mappings efficiently?

- Need to tolerate small variances/errors in each read
  - Each individual is different: Subject’s DNA may slightly differ from the reference (Mismatches, insertions, deletions)
  - How can we efficiently map each read with up to $e$ errors present?

- Need to map each read very fast (i.e., performance is important)
  - Human DNA is 3.2 billion base pairs long $\rightarrow$ Millions to billions of reads (State-of-the-art mappers take weeks to map a human’s DNA)
  - How can we design a much higher performance read mapper?
Our First Step: Comprehensive Mapping

- Guaranteed to find all mappings → sensitive
- Can tolerate up to $e$ errors

http://mrfast.sourceforge.net/

Personalized copy number and segmental duplication maps using next-generation sequencing

Can Alkan$^{1,2}$, Jeffrey M Kidd$^1$, Tomas Marques-Bonet$^{1,3}$, Gozde Aksay$^1$, Francesca Antonacci$^1$, Fereydoun Hormozdiari$^4$, Jacob O Kitzman$^1$, Carl Baker$^1$, Maika Malig$^1$, Onur Mutlu$^5$, S Cenk Sahinalp$^4$, Richard A Gibbs$^6$ & Evan E Eichler$^{1,2}$

Read Mapping Execution Time Breakdown

- Read Alignment (Edit-distance comp): 93%
- SAM printing: 3%
- Candidate alignment locations (CAL): 4%
The Read Mapping Bottleneck

Illumina HiSeq4000

300 Million bases/minute

2 Million bases/minute

150X slower
Idea

Filter fast before you align

Minimize costly

“approximate string comparisons”
Our First Filter: Pure Software Approach

- Download the source code and try for yourself
  - Download link to FastHASH
  - PDF article; Slides (pptx)

Xin et al. BMC Genomics 2013, 14(Suppl 1):S13
http://www.biomedcentral.com/1471-2164/14/S1/S13

PROCEEDINGS

Accelerating read mapping with FastHASH

Hongyi Xin¹, Donghyuk Lee¹, Farhad Hormozdiari², Samihan Yedkar¹, Onur Mutlu¹*, Can Alkan³*

From The Eleventh Asia Pacific Bioinformatics Conference (APBC 2013)
Vancouver, Canada. 21-24 January 2013
Shifted Hamming distance: a fast and accurate SIMD-friendly filter to accelerate alignment verification in read mapping

Hongyi Xin\textsuperscript{1,*}, John Greth\textsuperscript{2}, John Emmons\textsuperscript{2}, Gennady Pekhimenko\textsuperscript{1}, Carl Kingsford\textsuperscript{3}, Can Alkan\textsuperscript{4,*} and Onur Mutlu\textsuperscript{2,*}

Xin+, "\textit{Shifted Hamming Distance: A Fast and Accurate SIMD-friendly Filter to Accelerate Alignment Verification in Read Mapping}", Bioinformatics 2015.
GateKeeper: FPGA-Based Alignment Filtering

1st
FPGA-based Alignment Filter.

x10^{12}
mappings

High throughput DNA sequencing (HTS) technologies

Read Pre-Alignment Filtering
Fast & Low False Positive Rate

Read Alignment
Slow & Zero False Positives

x10^3 mappings

Low Speed & High Accuracy
Medium Speed, Medium Accuracy
High Speed, Low Accuracy

Billions of Short Reads
GateKeeper: FPGA-Based Alignment Filtering

Mohammed Alser, Hasan Hassan, Hongyi Xin, Oguz Ergin, Onur Mutlu, and Can Alkan

"GateKeeper: A New Hardware Architecture for Accelerating Pre-Alignment in DNA Short Read Mapping" Bioinformatics, [published online, May 31], 2017.
[Source Code]
[Online link at Bioinformatics Journal]

GateKeeper: a new hardware architecture for accelerating pre-alignment in DNA short read mapping

Mohammed Alser, Hasan Hassan, Hongyi Xin, Oğuz Ergin, Onur Mutlu, Can Alkan

Published: 31 May 2017 Article history ▼
MAGNET Accelerator [Alser+, TIR 2017]

[Source Code]
[Online link at Bioinformatics Journal]
DNA Read Mapping & Filtering

- **Problem:** Heavily bottlenecked by Data Movement

- GateKeeper FPGA performance limited by DRAM bandwidth [Alser+, Bioinformatics 2017]

- Ditto for SHD on SIMD [Xin+, Bioinformatics 2015]

- **Solution:** Processing-in-memory can alleviate the bottleneck

- However, we need to design mapping & filtering algorithms to fit processing-in-memory
In-Memory DNA Sequence Analysis


Proceedings of the *16th Asia Pacific Bioinformatics Conference (APBC)*, Yokohama, Japan, January 2018.
[arxiv.org Version (pdf)]

GRIM-Filter: Fast seed location filtering in DNA read mapping using processing-in-memory technologies

Jeremie S. Kim¹,6*, Damla Senol Cali¹, Hongyi Xin², Donghyuk Lee³, Saugata Ghose¹, Mohammed Alser⁴, Hasan Hassan⁶, Oguz Ergin⁵, Can Alkan⁴* and Onur Mutlu⁶,¹*

*From* The Sixteenth Asia Pacific Bioinformatics Conference 2018
Yokohama, Japan. 15-17 January 2018
Quick Note: Key Principles and Results

- **Two key principles:**
  - Exploit the structure of the genome to minimize computation
  - Morph and exploit the structure of the underlying hardware to maximize performance and efficiency

- **Algorithm-architecture co-design** for DNA read mapping
  - **Speeds up** read mapping by \( \sim 300X \) (sometimes more)
  - **Improves accuracy** of read mapping in the presence of errors

Kim et al., “Genome Read In-Memory (GRIM) Filter,” BMC Genomics 2018.
New Genome Sequencing Technologies

Nanopore sequencing technology and tools for genome assembly: computational analysis of the current state, bottlenecks and future directions

Damla Senol Cali, Jeremie S Kim, Saugata Ghose, Can Alkan, Onur Mutlu

Briefings in Bioinformatics, bby017, https://doi.org/10.1093/bib/bby017
Published: 02 April 2018  Article history ▼

Oxford Nanopore MinION

Nanopore Genome Assembly Pipeline

Raw signal data → Basecalling
  Tools: Metrichor, Nanonet, Scrappie, Nanocall, DeepNano
  DNA reads → Read-to-Read Overlap Finding
  Tools: GraphMap, Minimap
  Overlaps → Assembly
  Tools: Canu, Miniasm
  Draft assembly → Read Mapping
  Tools: BWA-MEM, Minimap, (GraphMap)
  Mappings of reads against draft assembly → Polishing
  Tools: Nanopolish, Racon
  Improved assembly

Figure 1. The analyzed genome assembly pipeline using nanopore sequence data, with its five steps and the associated tools for each step.

Recall Our Dream

- An embedded device that can perform comprehensive genome analysis in real time (within a minute)

- Still a long ways to go
  - Energy efficiency
  - Performance (latency)
  - Security
  - **Huge memory bottleneck**
More on Genome Analysis: Another Talk

Onur Mutlu,
"Accelerating Genome Analysis: A Primer on an Ongoing Journey"
Keynote talk at 2nd Workshop on Accelerator Architecture in Computational Biology and Bioinformatics (AACBB), Washington, DC, USA, February 2019.
[Slides (pptx)(pdf)]
[Video]

Accelerating Genome Analysis
A Primer on an Ongoing Journey

Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu
16 February 2019
AACBB Keynote Talk

SAFARI ETH Zürich Carnegie Mellon

https://www.youtube.com/watch?v=hPnSmfwu2-A
Four Key Directions

- Fundamentally Secure/Reliable/Safe Architectures
- Fundamentally Energy-Efficient Architectures
  - Memory-centric (Data-centric) Architectures
- Fundamentally Low-Latency Architectures
- Architectures for Genomics, Medicine, Health
Memory & Storage
Why Is Memory So Important? (Especially Today)
Importance of Main Memory

- The Performance Perspective

- The Energy Perspective

- The Scaling/Reliability/Security Perspective

- Trends/Challenges/Opportunities in Main Memory
Main memory is a critical component of all computing systems: server, mobile, embedded, desktop, sensor.

Main memory system must scale (in size, technology, efficiency, cost, and management algorithms) to maintain performance growth and technology scaling benefits.
Main memory is a critical component of all computing systems: server, mobile, embedded, desktop, sensor. Main memory system must scale (in size, technology, efficiency, cost, and management algorithms) to maintain performance growth and technology scaling benefits.
The Main Memory System

- Main memory is a critical component of all computing systems: server, mobile, embedded, desktop, sensor

- Main memory system must scale (in size, technology, efficiency, cost, and management algorithms) to maintain performance growth and technology scaling benefits
Memory System: A *Shared Resource* View

Most of the system is dedicated to storing and moving data.
State of the Main Memory System

- Recent technology, architecture, and application trends
  - lead to new requirements
  - exacerbate old requirements

- DRAM and memory controllers, as we know them today, are (will be) unlikely to satisfy all requirements

- Some emerging non-volatile memory technologies (e.g., PCM) enable new opportunities: memory+storage merging

- We need to rethink the main memory system
  - to fix DRAM issues and enable emerging technologies
  - to satisfy all requirements
Major Trends Affecting Main Memory (I)

- Need for main memory capacity, bandwidth, QoS increasing

- Main memory energy/power is a key system design concern

- DRAM technology scaling is ending
Major Trends Affecting Main Memory (II)

- Need for main memory capacity, bandwidth, QoS increasing
  - Multi-core: increasing number of cores/agents
  - Data-intensive applications: increasing demand/hunger for data
  - Consolidation: cloud computing, GPUs, mobile, heterogeneity

- Main memory energy/power is a key system design concern

- DRAM technology scaling is ending
Consequence: The Memory Capacity Gap

- Memory capacity per core expected to drop by 30% every two years
- Trends worse for memory bandwidth per core!
DRAM Capacity, Bandwidth & Latency

- **Capacity**
- **Bandwidth**
- **Latency**

- **128x** improvement in capacity
- **20x** improvement in bandwidth
- **1.3x** improvement in latency
DRAM Is Critical for Performance

In-memory Databases
[Maor+, EuroSys’12; Clapp+ (Intel), IISWC’15]

In-Memory Data Analytics
[Clapp+ (Intel), IISWC’15; Awan+, BDCloud’15]

Graph/Tree Processing
[Xu+, IISWC’12; Umuroglu+, FPL’15]

Datacenter Workloads
[Kanev+ (Google), ISCA’15]
DRAM Is Critical for Performance

In-memory Databases

Graph/Tree Processing

Memory → performance bottleneck

In-Memory Data Analytics
[Clapp+ (Intel), IISWC’15; Awan+, BDCloud’15]

Datacenter Workloads
[Kanev+ (Google), ISCA’15]
DRAM Is Critical for Performance

Chrome
Google’s web browser

TensorFlow Mobile
Google’s machine learning framework

VP9
YouTube
Video Playback
Google’s video codec

VP9
YouTube
Video Capture
Google’s video codec
DRAM Is Critical for Performance

Chrome

TensorFlow Mobile

Memory → performance bottleneck

VP9

Video Playback
Google’s video codec

Video Capture
Google’s video codec
Memory Bottleneck

- “It’s the Memory, Stupid!” (Richard Sites, MPR, 1996)

I expect that over the coming decade memory subsystem design will be the only important design issue for microprocessors.

Data from Runahead Execution [HPCA 2003]

The Memory Bottleneck


Runahead Execution: An Alternative to Very Large Instruction Windows for Out-of-order Processors

Onur Mutlu §    Jared Stark †    Chris Wilkerson ‡    Yale N. Patt §

§ECE Department
The University of Texas at Austin
{onur,patt}@ece.utexas.edu

†Microprocessor Research
Intel Labs
jared.w.stark@intel.com

‡Desktop Platforms Group
Intel Corporation
chris.wilkerson@intel.com
The Memory Bottleneck

- All of Google’s Data Center Workloads (2015):

The Memory Bottleneck

- All of Google’s Data Center Workloads (2015):

Figure 11: Half of cycles are spent stalled on caches.

Major Trends Affecting Main Memory (III)

- Need for main memory capacity, bandwidth, QoS increasing

- Main memory energy/power is a key system design concern
  - ~40-50% energy spent in off-chip memory hierarchy [Lefurgy, IEEE Computer’03] >40% power in DRAM [Ware, HPCA’10][Paul,ISCA’15]
  - DRAM consumes power even when not used (periodic refresh)

- DRAM technology scaling is ending
A memory access consumes \( \sim 1000X \) the energy of a complex addition.
62.7% of the total system energy is spent on data movement
Major Trends Affecting Main Memory (IV)

- Need for main memory capacity, bandwidth, QoS increasing

- Main memory energy/power is a key system design concern

- **DRAM technology scaling is ending**
  - ITRS projects **DRAM will not scale easily below X nm**
  - Scaling has provided many benefits:
    - higher capacity (density), lower cost, lower energy
The DRAM Scaling Problem

- DRAM stores charge in a capacitor (charge-based memory)
  - Capacitor must be large enough for reliable sensing
  - Access transistor should be large enough for low leakage and high retention time
  - Scaling beyond 40-35nm (2013) is challenging [ITRS, 2009]

- DRAM capacity, cost, and energy/power hard to scale
Limits of Charge Memory

- Difficult charge placement and control
  - Flash: floating gate charge
  - DRAM: capacitor charge, transistor leakage

- Reliable sensing becomes difficult as charge storage unit size reduces
As Memory Scales, It Becomes Unreliable

- Data from all of Facebook’s servers worldwide
- Meza+, “Revisiting Memory Errors in Large-Scale Production Data Centers,” DSN’15.

![Graph showing relative server failure rate vs. chip density (Gb)]

Intuition: quadratic increase in capacity
Large-Scale Failure Analysis of DRAM Chips

- Analysis and modeling of memory errors found in all of Facebook’s server fleet

- Justin Meza, Qiang Wu, Sanjeev Kumar, and Onur Mutlu, "Revisiting Memory Errors in Large-Scale Production Data Centers: Analysis and Modeling of New Trends from the Field" Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Rio de Janeiro, Brazil, June 2015.
  [Slides (pptx) (pdf)] [DRAM Error Model]

Revisiting Memory Errors in Large-Scale Production Data Centers: Analysis and Modeling of New Trends from the Field

JUSTIN MEZA  QIANG WU  SANJEEV KUMAR  ONUR MUTLU
Carnegie Mellon University  *Facebook, Inc.
Infrastructures to Understand Such Issues


- Flexible
- Easy to Use (C++ API)
- Open-source

github.com/CMU-SAFARI/SoftMC
SoftMC

- https://github.com/CMU-SAFARI/SoftMC

SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies

Hasan Hassan\textsuperscript{1,2,3} Nandita Vijaykumar\textsuperscript{3} Samira Khan\textsuperscript{4,3} Saugata Ghose\textsuperscript{3} Kevin Chang\textsuperscript{3} Gennady Pekhimenko\textsuperscript{5,3} Donghyuk Lee\textsuperscript{6,3} Oguz Ergin\textsuperscript{2} Onur Mutlu\textsuperscript{1,3}

\textsuperscript{1} ETH Zürich \quad \textsuperscript{2} TOBB University of Economics & Technology \quad \textsuperscript{3} Carnegie Mellon University
\textsuperscript{4} University of Virginia \quad \textsuperscript{5} Microsoft Research \quad \textsuperscript{6} NVIDIA Research
A Curious Discovery [Kim et al., ISCA 2014]

One can predictably induce errors in most DRAM memory chips
A simple hardware failure mechanism can create a widespread system security vulnerability.
Onur Mutlu,
"The RowHammer Problem and Other Issues We May Face as Memory Becomes Denser"
[Slides (pptx) (pdf)]
Onur Mutlu and Jeremie Kim,
"RowHammer: A Retrospective"
[Preliminary arXiv version]
The Technology Scaling Perspective

- Onur Mutlu,
  "Memory Scaling: A Systems Architecture Perspective"
  Proceedings of the 5th International Memory Workshop (IMW), Monterey, CA, May 2013. Slides (pptx) (pdf) EETimes Reprint

Memory Scaling: A Systems Architecture Perspective

Onur Mutlu
Carnegie Mellon University
onur@cmu.edu
http://users.ece.cmu.edu/~omutlu/

Major Trends Affecting Main Memory (V)

- **DRAM scaling has already become increasingly difficult**
  - Increasing cell leakage current, reduced cell reliability, increasing manufacturing difficulties [Kim+ ISCA 2014], [Liu+ ISCA 2013], [Mutlu IMW 2013], [Mutlu DATE 2017]
  - **Difficult to significantly improve capacity, energy**

- **Emerging memory technologies** are promising

<table>
<thead>
<tr>
<th>Technology</th>
<th>Benefits</th>
<th>Drawbacks</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D Stacked DRAM</td>
<td>Higher bandwidth</td>
<td>Smaller capacity</td>
</tr>
<tr>
<td>Reduced Latency DRAM</td>
<td>Lower latency</td>
<td>Higher cost</td>
</tr>
<tr>
<td>Low Power DRAM</td>
<td>Lower power</td>
<td>Higher latency, higher cost</td>
</tr>
<tr>
<td>Non-Volatile Memory</td>
<td>Larger capacity</td>
<td>Higher latency, higher dynamic power, lower endurance</td>
</tr>
<tr>
<td>3D XPoint</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Major Trends Affecting Main Memory (V)

- DRAM scaling has already become increasingly difficult
  - Increasing cell leakage current, reduced cell reliability, increasing manufacturing difficulties [Kim+ ISCA 2014], [Liu+ ISCA 2013], [Mutlu IMW 2013], [Mutlu DATE 2017]
  - Difficult to significantly improve capacity, energy

- **Emerging memory technologies** are promising

<table>
<thead>
<tr>
<th>3D-Stacked DRAM</th>
<th>higher bandwidth</th>
<th>smaller capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduced-Latency DRAM (e.g., RL/TL-DRAM, FLY-RAM)</td>
<td>lower latency</td>
<td>higher cost</td>
</tr>
<tr>
<td>Low-Power DRAM (e.g., LPDDR3, LPDDR4, Voltron)</td>
<td>lower power</td>
<td>higher latency higher cost</td>
</tr>
<tr>
<td>Non-Volatile Memory (NVM) (e.g., PCM, STTRAM, ReRAM, 3D Xpoint)</td>
<td>larger capacity</td>
<td>higher latency higher dynamic power lower endurance</td>
</tr>
</tbody>
</table>
Major Trend: Hybrid Main Memory

CPU

DRAM Ctrl

PCM Ctrl

DRAM

Fast, durable
Small, leaky, volatile, high-cost

Phase Change Memory (or Tech. X)

Large, non-volatile, low-cost
Slow, wears out, high active energy

Hardware/software manage data allocation and movement to achieve the best of multiple technologies

Yoon+, “Row Buffer Locality Aware Caching Policies for Hybrid Memories,” ICCD 2012 Best Paper Award.
One Foreshadowing

Main Memory Needs

Intelligent Controllers
Industry Is Writing Papers About It, Too

DRAM Process Scaling Challenges

- **Refresh**
  - Difficult to build high-aspect ratio cell capacitors decreasing cell capacitance
  - Leakage current of cell access transistors increasing

- **tWR**
  - Contact resistance between the cell capacitor and access transistor increasing
  - On-current of the cell access transistor decreasing
  - Bit-line resistance increasing

- **VRT**
  - Occurring more frequently with cell capacitance decreasing
Call for Intelligent Memory Controllers

DRAM Process Scaling Challenges

- **Refresh**
  - Difficult to build high-aspect ratio cell capacitors decreasing cell capacitance

THE MEMORY FORUM 2014

Co-Architecting Controllers and DRAM to Enhance DRAM Process Scaling

Uksong Kang, Hak-soo Yu, Churoo Park, *Hongzhong Zheng, **John Halbert, **Kuljit Bains, SeongJin Jang, and Joo Sun Choi

Samsung Electronics, Hwasung, Korea / *Samsung Electronics, San Jose / **Intel
An Orthogonal Issue: Memory Interference

Cores’ interfere with each other when accessing shared main memory. Uncontrolled interference leads to many problems (QoS, performance).
Goal: Predictable Performance in Complex Systems

- Heterogeneous agents: CPUs, GPUs, and HWAs
- Main memory interference between CPUs, GPUs, HWAs

How to allocate resources to heterogeneous agents to mitigate interference and provide predictable performance?
One Foreshadowing

Main Memory Needs
Intelligent Controllers
Solving the Memory Problem
How Do We Solve The Memory Problem?

- **Fix it**: Make memory and controllers more intelligent
  - New interfaces, functions, architectures: system-mem codesign

- **Eliminate or minimize it**: Replace or (more likely) augment DRAM with a different technology
  - New technologies and system-wide rethinking of memory & storage

- **Embrace it**: Design heterogeneous memories (none of which are perfect) and map data intelligently across them
  - New models for data management and maybe usage

- ...

88
How Do We Solve The Memory Problem?

- **Fix it**: Make memory and controllers more intelligent
  - New interfaces, functions, architectures: system-mem codesign

- **Eliminate or minimize it**: Replace or (more likely) augment DRAM with a different technology
  - New technologies and system-wide rethinking of memory & storage

- **Embrace it**: Design heterogeneous memories (none of which are perfect) and map data intelligently across them
  - New models for data management and maybe usage

**Solutions (to memory scaling) require software/hardware/device cooperation**
How Do We Solve The Memory Problem?

- **Fix it**: Make memory and controllers more intelligent
  - New interfaces, architectures: system-mem codesign

- **Eliminate or minimize it**: Replace or (more likely) augment DRAM with a different technology
  - New technologies and system-wide rethinking of memory & storage

- **Embrace it**: Design heterogeneous memories (none of which are perfect) and map data intelligently across them
  - New models for data management and maybe usage

Solutions (to memory scaling) require software/hardware/device cooperation
Solution 1: New Memory Architectures

- Overcome memory shortcomings with
  - Memory-centric system design
  - Novel memory architectures, interfaces, functions
  - Better waste management (efficient utilization)

- Key issues to tackle
  - Enable reliability at low cost → high capacity
  - Reduce energy
  - Reduce latency
  - Improve bandwidth
  - Reduce waste (capacity, bandwidth, latency)
  - Enable computation close to data
Solution 1: New Memory Architectures

- Seshadri+, "RowZone: Fast and Efficient In-DRAM Copy and Initialization of Bulk Data," MICRO 2013.
- Liu+, "Characterizing Application Memory Error Vulnerability to Optimize Data Center Cost," SIGMETRICS 2014.
- Seshadri+, "Gather-Scatter DRAM: In-DRAM Address Translation to Improve the Spatial Locality of Non-uniform Strided Accesses," MICRO 2015.
- Khan+, "A Case for Memory Content-Based Detection and Mitigation of Data-Dependent Failures in DRAM," IEEE CAL 2016.
- Mutlu, "The Rowhammer Problem and Other Issues We May Face as Memory Becomes Dense," DATE 2017.
- Khan+, "Detecting and Mitigating Data-Dependent DRAM Failures by Exploiting Current Memory Content," MICRO 2017.
- Avoid DRAM:
Solution 2: Emerging Memory Technologies

- Some emerging resistive memory technologies seem more scalable than DRAM (and they are non-volatile)

- Example: Phase Change Memory
  - Data stored by changing phase of material
  - Data read by detecting material’s resistance
  - Expected to scale to 9nm (2022 [ITRS 2009])
  - Prototyped at 20nm (Raoux+, IBM JRD 2008)
  - Expected to be denser than DRAM: can store multiple bits/cell

- But, emerging technologies have (many) shortcomings
  - Can they be enabled to replace/augment/surpass DRAM?
Solution 2: Emerging Memory Technologies

- Zhao+, “FIRM: Fair and High-Performance Memory Control for Persistent Memory Systems,” MICRO 2014.
PCM As Main Memory (2009)

- Benjamin C. Lee, Engin Ipek, Onur Mutlu, and Doug Burger, "Architecting Phase Change Memory as a Scalable DRAM Alternative". 

Architecting Phase Change Memory as a Scalable DRAM Alternative

Benjamin C. Lee† Engin Ipek† Onur Mutlu‡ Doug Burger†

†Computer Architecture Group
Microsoft Research
Redmond, WA
{blee, ipek, dburger}@microsoft.com

‡Computer Architecture Laboratory
Carnegie Mellon University
Pittsburgh, PA
onur@cmu.edu
More on PCM As Main Memory (2010)

- Benjamin C. Lee, Ping Zhou, Jun Yang, Youtao Zhang, Bo Zhao, Engin Ipek, Onur Mutlu, and Doug Burger,
  "Phase Change Technology and the Future of Main Memory"

Phase-Change Technology and the Future of Main Memory
Intel Optane Memory (Idea Realized in 2019)

- Non-volatile main memory
- Based on 3D-XPoint Technology

https://www.storagereview.com/intel_optane_dc_persistent_memory_module_pmm
Hybrid Memory Systems

Hardware/software manage data allocation and movement to achieve the best of multiple technologies

Yoon, Meza et al., “Row Buffer Locality Aware Caching Policies for Hybrid Memories,” ICCD 2012 Best Paper Award.
Exploiting Memory Error Tolerance with Hybrid Memory Systems

- **Vulnerable data**
- **Tolerant data**

- **Reliable memory**
- **Low-cost memory**

On Microsoft’s Web Search workload:
- Reduces server hardware cost by 4.7%
- Achieves single server availability target of 99.90%

**Heterogeneous-Reliability Memory** [DSN 2014]
Heterogeneous-Reliability Memory

Step 1: Characterize and classify application memory error tolerance

Step 2: Map application data to the HRM system enabled by SW/HW cooperative solutions
**Evaluation Results**

- **Typical Server**
- **Consumer PC**
- **HRM**
- **Less-Tested (L)**
- **HRM/L**

**Crashes/server/month**

**Server HW cost savings (%)**

**Memory cost savings (%)**

- Bigger area means better tradeoff

- Inner is worse

- Outer is better
More on Heterogeneous Reliability Memory

- Yixin Luo, Sriram Govindan, Bikash Sharma, Mark Santaniello, Justin Meza, Aman Kansal, Jie Liu, Badriddine Khessib, Kushagra Vaid, and Onur Mutlu, "Characterizing Application Memory Error Vulnerability to Optimize Data Center Cost via Heterogeneous-Reliability Memory" Proceedings of the 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Atlanta, GA, June 2014. [Summary] [Slides (pptx) (pdf)] [Coverage on ZDNet]
An Orthogonal Issue: Memory Interference

- **Problem:** Memory interference between cores is uncontrolled
  - unfairness, starvation, low performance
  - uncontrollable, unpredictable, vulnerable system

- **Solution:** QoS-Aware Memory Systems
  - Hardware designed to provide a configurable fairness substrate
    - Application-aware memory scheduling, partitioning, throttling
  - Software designed to configure the resources to satisfy different QoS goals

- QoS-aware memory systems can provide predictable performance and higher efficiency
Strong Memory Service Guarantees

- **Goal:** Satisfy performance/SLA requirements in the presence of shared main memory, heterogeneous agents, and hybrid memory/storage

- **Approach:**
  - Develop techniques/models to accurately estimate the performance loss of an application/agent in the presence of resource sharing
  - Develop mechanisms (hardware and software) to enable the resource partitioning/prioritization needed to achieve the required performance levels for all applications
  - All the while providing high system performance

DRAM Controllers
Method and apparatus to control memory accesses

Abstract

A method and apparatus for accessing memory comprising monitoring memory accesses from a hardware prefetcher and determining whether the memory accesses from the hardware prefetcher are used by an out-of-order core. A front side bus controller switches memory access modes from a minimize memory access latency mode to a maximize memory bus bandwidth mode if a percentage of the memory accesses generated by the hardware prefetcher are used by the out-of-order core.

Classifications

G06F12/0215  Addressing or allocation; Relocation with look ahead addressing means
Thomas Moscibroda and Onur Mutlu,
"Memory Performance Attacks: Denial of Memory Service in Multi-Core Systems"
Onur Mutlu and Thomas Moscibroda, "Stall-Time Fair Memory Access Scheduling for Chip Multiprocessors"
Proceedings of the 40th International Symposium on Microarchitecture (MICRO), pages 146-158, Chicago, IL, December 2007. [Summary] [Slides (ppt)]
Onur Mutlu and Thomas Moscibroda, "Parallelism-Aware Batch Scheduling: Enhancing both Performance and Fairness of Shared DRAM Systems". Proceedings of the 35th International Symposium on Computer Architecture (ISCA), pages 63-74, Beijing, China, June 2008. [Summary] [Slides (ppt)]
On PAR-BS

- Variants implemented in Samsung SoC memory controllers

Effective platform level approach and DRAM accesses are crucial to system performance. This paper touches this topics and suggest a superior approach to current known techniques.

Review from ISCA 2008
ATLAS: A Scalable and High-Performance Scheduling Algorithm for Multiple Memory Controllers

Yoongu Kim, Dongsu Han, Onur Mutlu, and Mor Harchol-Balter

"ATLAS: A Scalable and High-Performance Scheduling Algorithm for Multiple Memory Controllers"

Proceedings of the 16th International Symposium on High-Performance Computer Architecture (HPCA), Bangalore, India, January 2010. Slides (pptx)
Yoongu Kim, Michael Papamichael, Onur Mutlu, and Mor Harchol-Balter,
"Thread Cluster Memory Scheduling: Exploiting Differences in Memory Access Behavior"
Lavanya Subramanian, Donghyuk Lee, Vivek Seshadri, Harsha Rastogi, and Onur Mutlu,
"The Blacklisting Memory Scheduler: Achieving High Performance and Fairness at Low Cost"
Proceedings of the 32nd IEEE International Conference on Computer Design (ICCD), Seoul, South Korea, October 2014.
[Slides (pptx) (pdf)]
Staged Memory Scheduling: Achieving High Performance and Scalability in Heterogeneous Systems

Rachata Ausavarungnirun†, Kevin Kai-Wei Chang†, Lavanya Subramanian†, Gabriel H. Loh‡, and Onur Mutlu†
†Carnegie Mellon University
{rachata,kevincha,lsubrama,onur}@cmu.edu
‡Advanced Micro Devices, Inc.
gabe.loh@amd.com
Hiroyuki Usui, Lavanya Subramanian, Kevin Kai-Wei Chang, and Onur Mutlu,
"DASH: Deadline-Aware High-Performance Memory Scheduler for Heterogeneous Systems with Hardware Accelerators"
Presented at the 11th HiPEAC Conference, Prague, Czech Republic, January 2016.
[Slides (pptx) (pdf)]
[Source Code]
MISE: Predictable Performance [HPCA’13]

- Lavanya Subramanian, Vivek Seshadri, Yoongu Kim, Ben Jaiyen, and Onur Mutlu,

"MISE: Providing Performance Predictability and Improving Fairness in Shared Main Memory Systems"

Proceedings of the 19th International Symposium on High-Performance Computer Architecture (HPCA), Shenzhen, China, February 2013. Slides (pptx)
ASM: Predictable Performance [MICRO’15]

- Lavanya Subramanian, Vivek Seshadri, Arnab Ghosh, Samira Khan, and Onur Mutlu,
"The Application Slowdown Model: Quantifying and Controlling the Impact of Inter-Application Interference at Shared Caches and Main Memory"
Proceedings of the 48th International Symposium on Microarchitecture (MICRO), Waikiki, Hawaii, USA, December 2015.
[Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Poster (pptx) (pdf)]
[Source Code]

The Application Slowdown Model: Quantifying and Controlling the Impact of Inter-Application Interference at Shared Caches and Main Memory

Lavanya Subramanian*§  Vivek Seshadri*  Arnab Ghosh*†
Samira Khan*†  Onur Mutlu*

*Carnegie Mellon University  §Intel Labs  †IIT Kanpur  ‡University of Virginia

SAFARI
The Future

Memory Controllers are critical to research

They will become even more important
Memory Control is Getting More Complex

- Heterogeneous agents: CPUs, GPUs, and HWAs
- Main memory interference between CPUs, GPUs, HWAs

Many goals, many constraints, many metrics ...
The Future

Memory Controllers:
Many New Problems
Takeaway

Main Memory Needs
Intelligent Controllers
What We Will Cover

In These Lectures
Agenda for These Lectures

- Memory Importance, Trends, Solution Directions
- RowHammer: Memory Reliability and Security
- In-Memory Computation
- Low-Latency Memory
- Data-Driven and Data-Aware Architectures
- Guiding Principles & Conclusion
This Course

- Will cover many problems and potential solutions related to the design of memory systems in the many core era

- The design of the memory system poses many
  - Difficult research and engineering problems
  - Important fundamental problems
  - Industry-relevant problems
  - Problems whose solutions can revolutionize the world

- Many creative and insightful solutions are needed to solve these problems

- Goal: Acquire the basics to develop such solutions (by covering fundamentals and cutting edge research)
How To Make the Best Out of This Course

- Be alert during lectures – they will be fast paced
  - Do not try to read everything on slides

- Do the readings (and develop ideas)
  - I will provide many references

- Go back and reinforce fundamentals (as needed)
  - I will provide pointers to basic computer architecture materials (lecture videos, slides, readings, exams, ...)
  - https://www.youtube.com/onurmutlulectures

- Remember “Chance favors the prepared mind.” (Pasteur)
Unfortunately, No Time For:

- Memory Interference and QoS
- Predictable Performance
  - QoS-aware Memory Controllers
- Emerging Memory Technologies and Hybrid Memories
- Cache Management
- Interconnects

You can find many materials on these at my online lectures
- [https://people.inf.ethz.ch/omutlu/teaching.html](https://people.inf.ethz.ch/omutlu/teaching.html)
Course Information

- My Contact Information
  - Onur Mutlu
  - omutlu@gmail.com
  - https://people.inf.ethz.ch/omutlu
  - +41-79-572-1444 (my cell phone)
  - Find me during breaks and/or email any time.

- Website for Course Slides, Papers, Updates, Lecture Videos

- For the curious:
  - See the backup slides for reference works and papers
An “Early” Position Paper [IMW’13]

- Onur Mutlu,
  "Memory Scaling: A Systems Architecture Perspective"
  Proceedings of the 5th International Memory Workshop (IMW), Monterey, CA, May 2013. Slides (pptx) (pdf)
  EETimes Reprint

Memory Scaling: A Systems Architecture Perspective

Onur Mutlu
Carnegie Mellon University
onur@cmu.edu
http://users.ece.cmu.edu/~omutlu/

Challenges in DRAM Scaling

- Refresh
- Latency
- Bank conflicts/parallelism
- Reliability and vulnerabilities
- Energy & power
- Memory’s inability to do more than store data
A Recent Retrospective Paper [TCAD’19]

- Onur Mutlu and Jeremie Kim,
  "RowHammer: A Retrospective"


[Preliminary arXiv version]

---

RowHammer: A Retrospective

Onur Mutlu§‡  Jeremie S. Kim‡§

§ETH Zürich  ‡Carnegie Mellon University
Ramulator Motivation

- DRAM and Memory Controller landscape is changing
- Many new and upcoming standards
- Many new controller designs
- A fast and easy-to-extend simulator is very much needed

<table>
<thead>
<tr>
<th>Segment</th>
<th>DRAM Standards &amp; Architectures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commodity</td>
<td>DDR3 (2007) [14]; DDR4 (2012) [18]</td>
</tr>
<tr>
<td>Performance</td>
<td>eDRAM [28], [32]; RLDRAM3 (2011) [29]</td>
</tr>
</tbody>
</table>

Table 1. Landscape of DRAM-based memory
Ramulator

- Provides out-of-the-box support for many DRAM standards:
  - DDR3/4, LPDDR3/4, GDDR5, WIO1/2, HBM, plus new proposals (SALP, AL-DRAM, TLDRAM, RowClone, and SARP)
- ~2.5X faster than fastest open-source simulator
- Modular and extensible to different standards

<table>
<thead>
<tr>
<th>Simulator (clang -O3)</th>
<th>Cycles (10^6)</th>
<th>Runtime (sec.)</th>
<th>Req/sec (10^3)</th>
<th>Memory (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Random</td>
<td>Stream</td>
<td>Random</td>
<td>Stream</td>
</tr>
<tr>
<td>Ramulator</td>
<td>652</td>
<td>411</td>
<td>752</td>
<td>249</td>
</tr>
<tr>
<td>DRAMSim2</td>
<td>645</td>
<td>413</td>
<td>2,030</td>
<td>876</td>
</tr>
<tr>
<td>USIMM</td>
<td>661</td>
<td>409</td>
<td>1,880</td>
<td>750</td>
</tr>
<tr>
<td>DrSim</td>
<td>647</td>
<td>406</td>
<td>18,109</td>
<td>12,984</td>
</tr>
<tr>
<td>NVMain</td>
<td>666</td>
<td>413</td>
<td>6,881</td>
<td>5,023</td>
</tr>
</tbody>
</table>

Table 3. Comparison of five simulators using two traces
Case Study: Comparison of DRAM Standards

<table>
<thead>
<tr>
<th>Standard</th>
<th>Rate (MT/s)</th>
<th>Timing (CL-RCD-RP)</th>
<th>Data-Bus (Width×Chan.)</th>
<th>Rank-per-Chan</th>
<th>BW (GB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDR3</td>
<td>1,600</td>
<td>11-11-11</td>
<td>64-bit × 1</td>
<td>1</td>
<td>11.9</td>
</tr>
<tr>
<td>DDR4</td>
<td>2,400</td>
<td>16-16-16</td>
<td>64-bit × 1</td>
<td>1</td>
<td>17.9</td>
</tr>
<tr>
<td>SALP†</td>
<td>1,600</td>
<td>11-11-11</td>
<td>64-bit × 1</td>
<td>1</td>
<td>11.9</td>
</tr>
<tr>
<td>LPDDR3</td>
<td>1,600</td>
<td>12-15-15</td>
<td>64-bit × 1</td>
<td>1</td>
<td>11.9</td>
</tr>
<tr>
<td>LPDDR4</td>
<td>2,400</td>
<td>22-22-22</td>
<td>32-bit × 2*</td>
<td>1</td>
<td>17.9</td>
</tr>
<tr>
<td>GDDR5 [12]</td>
<td>6,000</td>
<td>18-18-18</td>
<td>64-bit × 1</td>
<td>1</td>
<td>44.7</td>
</tr>
<tr>
<td>HBM</td>
<td>1,000</td>
<td>7-7-7</td>
<td>128-bit × 8*</td>
<td>1</td>
<td>119.2</td>
</tr>
<tr>
<td>WIO</td>
<td>266</td>
<td>7-7-7</td>
<td>128-bit × 4*</td>
<td>1</td>
<td>15.9</td>
</tr>
<tr>
<td>WIO2</td>
<td>1,066</td>
<td>9-10-10</td>
<td>128-bit × 8*</td>
<td>1</td>
<td>127.2</td>
</tr>
</tbody>
</table>

Across 22 workloads, simple CPU model

Figure 2. Performance comparison of DRAM standards
Ramulator Paper and Source Code


- Source code is released under the liberal MIT License
  - https://github.com/CMU-SAFARI/ramulator

---

Ramulator: A Fast and Extensible DRAM Simulator

Yoongu Kim\(^1\)  Weikun Yang\(^1,2\)  Onur Mutlu\(^1\)
\(^1\)Carnegie Mellon University \(^2\)Peking University

SAFARI
Optional Assignment

- Review the Ramulator paper
  - Email me your review (omutlu@gmail.com)

- Download and run Ramulator
  - Compare DDR3, DDR4, SALP, HBM for the libquantum benchmark (provided in Ramulator repository)
  - Email me your report (omutlu@gmail.com)

- This **will** help you get into **memory systems research**
Memory Systems and Memory-Centric Computing Systems

Part 1: Memory Importance and Trends

Prof. Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu
3 February 2020
Champbery Winter School

SAFARI ETH Zürich Carnegie Mellon
Backup Slides:

Reference Materials
Readings, Videos, Reference Materials
Accelerated Memory Course (~6.5 hours)

- **ACACES 2018**
  - Memory Systems and Memory-Centric Computing Systems
  - Taught by Onur Mutlu July 9-13, 2018
  - ~6.5 hours of lectures

- **Website for the Course including Videos, Slides, Papers**
  - [https://safari.ethz.ch/memory_systems/ACACES2018/](https://safari.ethz.ch/memory_systems/ACACES2018/)
  - [https://www.youtube.com/playlist?list=PL5Q2soXY2Zi-HXxomthrpDpMJm05P6J9x](https://www.youtube.com/playlist?list=PL5Q2soXY2Zi-HXxomthrpDpMJm05P6J9x)

- **All Papers are at:**
  - [https://people.inf.ethz.ch/omutlu/projects.htm](https://people.inf.ethz.ch/omutlu/projects.htm)
  - Final lecture notes and readings (for all topics)
Longer Memory Course (~18 hours)

- **Tu Wien 2019**
  - Memory Systems and Memory-Centric Computing Systems
  - Taught by Onur Mutlu June 12-19, 2019
  - ~18 hours of lectures

- **Website for the Course including Videos, Slides, Papers**
  - [https://www.youtube.com/playlist?list=PL5Q2soXY2Zi_gntM55VoMIKlw7YrXOhbl](https://www.youtube.com/playlist?list=PL5Q2soXY2Zi_gntM55VoMIKlw7YrXOhbl)

- **All Papers are at:**
  - [https://people.inf.ethz.ch/omutlu/projects.htm](https://people.inf.ethz.ch/omutlu/projects.htm)
  - Final lecture notes and readings (for all topics)
Some Overview Talks

https://www.youtube.com/watch?v=kgiZlSOcGFM&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl

- **Future Computing Architectures**
  - https://www.youtube.com/watch?v=kgiZlSOcGFM&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=1

- **Enabling In-Memory Computation**
  - https://www.youtube.com/watch?v=oHqsNbxgdzM&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=7

- **Accelerating Genome Analysis**
  - https://www.youtube.com/watch?v=hPnSmfwu2-A&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=9

- **Rethinking Memory System Design**
  - https://www.youtube.com/watch?v=F7xZLNMIY1E&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=3
Processing Data Where It Makes Sense: Enabling In-Memory Computation

Onur Mutlu\textsuperscript{a,b}, Saugata Ghose\textsuperscript{b}, Juan Gómez-Luna\textsuperscript{a}, Rachata Ausavarungnirun\textsuperscript{b,c}

\textsuperscript{a}ETH Zürich
\textsuperscript{b}Carnegie Mellon University
\textsuperscript{c}King Mongkut’s University of Technology North Bangkok

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, and Rachata Ausavarungnirun, "Processing Data Where It Makes Sense: Enabling In-Memory Computation"
Invited paper in Microprocessors and Microsystems (MICPRO), June 2019.
[arXiv version]

Enabling the Adoption of Processing-in-Memory: Challenges, Mechanisms, Future Research Directions

SAUGATA GHOSE, KEVIN HSIEH, AMIRALI BOROUMAND, RACHATA AUSAVARUNGNIRUN
Carnegie Mellon University

ONUR MUTLU
ETH Zürich and Carnegie Mellon University


Reference Overview Paper III

- Onur Mutlu and Lavanya Subramanian, "Research Problems and Opportunities in Memory Systems". Invited Article in Supercomputing Frontiers and Innovations (SUPERFRI), 2014/2015.

Research Problems and Opportunities in Memory Systems

Onur Mutlu\textsuperscript{1}, Lavanya Subramanian\textsuperscript{1}

Onur Mutlu,
"The RowHammer Problem and Other Issues We May Face as Memory Becomes Denser"
[Slides (pptx) (pdf)]
Onur Mutlu,
"Memory Scaling: A Systems Architecture Perspective"
Technical talk at MemCon 2013 (MEMCON), Santa Clara, CA, August 2013. [Slides (pptx) (pdf)] [Video] [Coverage on StorageSearch]
Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives

This paper reviews the most recent advances in solid-state drive (SSD) error characterization, mitigation, and data recovery techniques to improve both SSD’s reliability and lifetime.

By Yu Cai, Saugata Ghose, Erich F. Haratsch, Yixin Luo, and Onur Mutlu

https://arxiv.org/pdf/1706.08642
Onur Mutlu and Jeremie Kim, "RowHammer: A Retrospective"
[Preliminary arXiv version]
Related Videos and Course Materials (I)

- Parallel Computer Architecture Course Materials (Lecture Videos)
Related Videos and Course Materials (II)


- Memory Systems Short Course Materials (Lecture Video on Main Memory and DRAM Basics)
Some Open Source Tools (I)

- **Rowhammer – Program to Induce RowHammer Errors**
  - [https://github.com/CMU-SAFARI/rowhammer](https://github.com/CMU-SAFARI/rowhammer)

- **Ramulator – Fast and Extensible DRAM Simulator**
  - [https://github.com/CMU-SAFARI/ramulator](https://github.com/CMU-SAFARI/ramulator)

- **MemSim – Simple Memory Simulator**
  - [https://github.com/CMU-SAFARI/memsim](https://github.com/CMU-SAFARI/memsim)

- **NOCulator – Flexible Network-on-Chip Simulator**
  - [https://github.com/CMU-SAFARI/NOCulator](https://github.com/CMU-SAFARI/NOCulator)

- **SoftMC – FPGA-Based DRAM Testing Infrastructure**
  - [https://github.com/CMU-SAFARI/SoftMC](https://github.com/CMU-SAFARI/SoftMC)

- **Other open-source software from my group**
  - [https://github.com/CMU-SAFARI/](https://github.com/CMU-SAFARI/)
  - [http://www.ece.cmu.edu/~safari/tools.html](http://www.ece.cmu.edu/~safari/tools.html)
Some Open Source Tools (II)

- MQSim – A Fast Modern SSD Simulator
  - https://github.com/CMU-SAFARI/MQSim
- Mosaic – GPU Simulator Supporting Concurrent Applications
  - https://github.com/CMU-SAFARI/Mosaic
- IMPICA – Processing in 3D-Stacked Memory Simulator
  - https://github.com/CMU-SAFARI/IMPICA
- SMLA – Detailed 3D-Stacked Memory Simulator
  - https://github.com/CMU-SAFARI/SMLA
- HWASim – Simulator for Heterogeneous CPU-HWA Systems
  - https://github.com/CMU-SAFARI/HWASim

- Other open-source software from my group
  - https://github.com/CMU-SAFARI/
  - http://www.ece.cmu.edu/~safari/tools.html
More Open Source Tools (III)

- A lot more open-source software from my group
  - https://github.com/CMU-SAFARI/
  - http://www.ece.cmu.edu/~safari/tools.html
Referenced Papers

- All are available at

https://people.inf.ethz.ch/omutlu/projects.htm

http://scholar.google.com/citations?user=7XyGUGkAAAAJ&hl=en

Ramuulator: A Fast and Extensible DRAM Simulator

[IEEE Comp Arch Letters’15]
Ramulator Motivation

- DRAM and Memory Controller landscape is changing
- Many new and upcoming standards
- Many new controller designs
- A fast and easy-to-extend simulator is very much needed

<table>
<thead>
<tr>
<th>Segment</th>
<th>DRAM Standards &amp; Architectures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low-Power</td>
<td>LPDDR3 (2012) [17]; LPDDR4 (2014) [20]</td>
</tr>
<tr>
<td>Graphics</td>
<td>GDDR5 (2009) [15]</td>
</tr>
<tr>
<td>Performance</td>
<td>eDRAM [28], [32]; RLDDRAM3 (2011) [29]</td>
</tr>
</tbody>
</table>

Table 1. Landscape of DRAM-based memory
Ramulator

- Provides out-of-the-box support for many DRAM standards:
  - DDR3/4, LPDDR3/4, GDDR5, WIO1/2, HBM, plus new proposals (SALP, AL-DRAM, TLDRAM, RowClone, and SARP)
- ~2.5X faster than fastest open-source simulator
- Modular and extensible to different standards

<table>
<thead>
<tr>
<th>Simulator (clang -O3)</th>
<th>Cycles (10^6)</th>
<th>Runtime (sec.)</th>
<th>Req/sec (10^3)</th>
<th>Memory (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Random</td>
<td>Stream</td>
<td>Random</td>
<td>Stream</td>
</tr>
<tr>
<td>Ramulator</td>
<td>652</td>
<td>411</td>
<td>752</td>
<td>249</td>
</tr>
<tr>
<td>DRAMSim2</td>
<td>645</td>
<td>413</td>
<td>2,030</td>
<td>876</td>
</tr>
<tr>
<td>USIMM</td>
<td>661</td>
<td>409</td>
<td>1,880</td>
<td>750</td>
</tr>
<tr>
<td>DrSim</td>
<td>647</td>
<td>406</td>
<td>18,109</td>
<td>12,984</td>
</tr>
<tr>
<td>NVMaint</td>
<td>666</td>
<td>413</td>
<td>6,881</td>
<td>5,023</td>
</tr>
</tbody>
</table>

Table 3. Comparison of five simulators using two traces
### Case Study: Comparison of DRAM Standards

<table>
<thead>
<tr>
<th>Standard</th>
<th>Rate (MT/s)</th>
<th>Timing (CL-RCD-RP)</th>
<th>Data-Bus (Width x Chan.)</th>
<th>Rank-per-Chan</th>
<th>BW (GB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDR3</td>
<td>1,600</td>
<td>11-11-11</td>
<td>64-bit x 1</td>
<td>1</td>
<td>11.9</td>
</tr>
<tr>
<td>DDR4</td>
<td>2,400</td>
<td>16-16-16</td>
<td>64-bit x 1</td>
<td>1</td>
<td>17.9</td>
</tr>
<tr>
<td>SALP†</td>
<td>1,600</td>
<td>11-11-11</td>
<td>64-bit x 1</td>
<td>1</td>
<td>11.9</td>
</tr>
<tr>
<td>LPDDR3</td>
<td>1,600</td>
<td>12-15-15</td>
<td>64-bit x 1</td>
<td>1</td>
<td>11.9</td>
</tr>
<tr>
<td>LPDDR4</td>
<td>2,400</td>
<td>22-22-22</td>
<td>32-bit x 2*</td>
<td>1</td>
<td>17.9</td>
</tr>
<tr>
<td>GDDR5 [12]</td>
<td>6,000</td>
<td>18-18-18</td>
<td>64-bit x 1</td>
<td>1</td>
<td>44.7</td>
</tr>
<tr>
<td>HBM</td>
<td>1,000</td>
<td>7-7-7</td>
<td>128-bit x 8*</td>
<td>1</td>
<td>119.2</td>
</tr>
<tr>
<td>WIO</td>
<td>266</td>
<td>7-7-7</td>
<td>128-bit x 4*</td>
<td>1</td>
<td>15.9</td>
</tr>
<tr>
<td>WIO2</td>
<td>1,066</td>
<td>9-10-10</td>
<td>128-bit x 8*</td>
<td>1</td>
<td>127.2</td>
</tr>
</tbody>
</table>

**Figure 2. Performance comparison of DRAM standards**

Across 22 workloads, simple CPU model
Ramulator Paper and Source Code


- Source code is released under the liberal MIT License
  - https://github.com/CMU-SAFAIR/ramulator

Ramulator: A Fast and Extensible DRAM Simulator

Yoongu Kim¹  Weikun Yang¹,²  Onur Mutlu¹

¹Carnegie Mellon University  ²Peking University
Optional Assignment

- **Review the Ramulator paper**
  - Email me your review (omutlu@gmail.com)

- **Download and run Ramulator**
  - Compare DDR3, DDR4, SALP, HBM for the libquantum benchmark (provided in Ramulator repository)
  - Email me your report (omutlu@gmail.com)

- **This will help you get into memory systems research**
Some More Suggested Readings
Some Key Readings on DRAM (I)

- DRAM Organization and Operation
Some Key Readings on DRAM (II)

- **DRAM Refresh**


Reading on Simulating Main Memory

- How to evaluate future main memory systems?
- An open-source simulator and its brief description

Some Key Readings on Memory Control 1

  https://people.inf.ethz.ch/omutlu/pub/parbs_isca08.pdf


Some Key Readings on Memory Control 2

  https://people.inf.ethz.ch/omutlu/pub/rlmc_isca08.pdf


More Readings

- To come as we cover the future topics

- Search for “DRAM” or “Memory” in:
  - [https://people.inf.ethz.ch/omutlu/projects.htm](https://people.inf.ethz.ch/omutlu/projects.htm)
Inside A DRAM Chip
DRAM Module and Chip
Goals

• Cost
• Latency
• Bandwidth
• Parallelism
• Power
• Energy
• Reliability
• ...

...
DRAM Chip
Sense Amplifier

Inverter

top

bottom

enable

Inverter
Sense Amplifier – Two Stable States

$V_{DD}$

Logical “1”

Logical “0”
Sense Amplifier Operation

\[
V_T > V_B
\]
DRAM Cell – Capacitor

Empty State
Logical “0”

Fully Charged State
Logical “1”

1. Small – Cannot drive circuits
2. Reading destroys the state
Capacitor to Sense Amplifier
DRAM Subarray – Building Block for DRAM Chip

- Cell Array
- Array of Sense Amplifiers (Row Buffer) 8Kb
- Row Decoder
DRAM Bank

Address

Row Decoder

Array of Sense Amplifiers (8Kb)

Cell Array

Row Decoder

Cell Array

Row Decoder

Array of Sense Amplifiers

Cell Array

Bank I/O (64b)

Address

Data
DRAM Chip

Shared internal bus

Memory channel - 8bits
DRAM Operation

1. ACTIVATE Row
2. READ/WRITE Column
3. PRECHARGE
End of Backup Slides