Memory-Centric Computing

Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu
22 September 2022
Simon Fraser University Distinguished Lecture
The Problem

Computing is Bottlenecked by Data
Data is Key for AI, ML, Genomics, …

- Important workloads are all data intensive

- They require rapid and efficient processing of large amounts of data

- Data is increasing
  - We can generate more than we can process
Huge Demand for Performance & Efficiency

Exponential Growth of Neural Networks

1800x more compute
In just 2 years

Tomorrow, multi-trillion parameter models

Source: https://youtu.be/Bh13Idwcb0Q?t=283
Data is Key for Future Workloads

In-memory Databases

-In-memory Data Analytics

[Clapp+ (Intel), IISWC’15]

Graph/Tree Processing

[Xu+, IISWC’12; Umuroglu+, FPL’15]

In-Memory Data Analytics

[Clapp+ (Intel), IISWC’15; Awan+, BDCloud’15]

Datacenter Workloads

[Kanev+ (Google), ISCA’15]
Data Overwhelms Modern Machines

In-memory Databases

Graph/Tree Processing

Data → performance & energy bottleneck

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

Datacenter Workloads
[Kanev+ (Google), ISCA’15]
Data is Key for Future Workloads

Chrome
Google’s web browser

TensorFlow Mobile
Google’s machine learning framework

VP9
Video Playback
Google’s video codec

VP9
Video Capture
Google’s video codec
Data Overwhelms Modern Machines

Chrome

TensorFlow Mobile

Data → performance & energy bottleneck

VP9

Video Playback

Google’s video codec

VP9

Video Capture

Google’s video codec
Data is Key for Future Workloads

Development of high-throughput sequencing (HTS) technologies

Number of Genomes Sequenced

1 Sequencing

Genome Analysis

Data → performance & energy bottleneck

2 Read Mapping

3 Variant Calling

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

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

Data → performance & energy bottleneck
One Problem with (Genome) Analysis Today

Special-Purpose Machine for Data Generation

General-Purpose Machine for Data Analysis

FAST  SLOW

Slow and inefficient processing capability

SAFARI  This picture is similar for many “data generators & analyzers” today

[Slides (pptx)(pdf)]
[Talk Video (1 hour 2 minutes)]
FPGA-based Near-Memory Analytics


FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications

Gagandeep Singh◊  Mohammed Alser◊  Damla Senol Cali✉
Dionysios Diamantopoulos▼  Juan Gómez-Luna◊
Henk Corporaal*  Onur Mutlu✉☎

◊ETH Zürich  ◆Carnegie Mellon University
*Eindhoven University of Technology  ▼IBM Research Europe
Near-Memory Acceleration using FPGAs

IBM POWER9 CPU

HBM-based FPGA board

Near-HBM FPGA-based accelerator

Two communication technologies: CAPI2 and OCAPI
Two memory technologies: DDR4 and HBM
Two workloads: Weather Modeling and Genome Analysis
Performance & Energy Greatly Improve

5-27× performance vs. a 16-core (64-thread) IBM POWER9 CPU

12-133× energy efficiency vs. a 16-core (64-thread) IBM POWER9 CPU

HBM alleviates memory bandwidth contention vs. DDR4
GenASM Framework [MICRO 2020]


- [Lighting Talk Video](#) (1.5 minutes)
- [Lightning Talk Slides (pptx) (pdf)](#)
- [Talk Video](#) (18 minutes)
- [Slides (pptx) (pdf)](#)

---

GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis

Damla Senol Cali†‡\[^{M}\] Gurpreet S. Kalsi‡\[^{M}\] Züllal Bingöl\[^{V}\] Can Firtina\[^{O}\] Lavanya Subramanian‡\[^{†}\] Jeremie S. Kim\[^{†}\]
Rachata Ausavarungnirun\[^{C}\] Mohammed Alser\[^{O}\] Juan Gomez-Luna\[^{O}\] Amirali Boroumand‡\[^{†}\] Anant Nori\[^{M}\]
Allison Scibisz‡\[^{†}\] Sreenivas Subramoney\[^{M}\] Can Alkan\[^{V}\] Saugata Ghose\[^{†}\] Onur Mutlu\[^{†}\]^\[^{V}\]

†Carnegie Mellon University \[^{M}\] Processor Architecture Research Lab, Intel Labs \[^{O}\] Bilkent University \[^{O}\] ETH Zürich
‡Facebook \[^{C}\] King Mongkut’s University of Technology North Bangkok \[^{M}\] University of Illinois at Urbana–Champaign

SAFARI
In-Storage Genome Filtering [ASPLOS 2022]

- Nika Mansouri Ghiasi, Jisung Park, Harun Mustafa, Jeremie Kim, Ataberk Olgun, Arvid Gollwitzer, Damla Senol Cali, Can Firtina, Haiyu Mao, Nour Almadhoun Alserr, Rachata Ausavarungnirun, Nandita Vijaykumar, Mohammed Alser, and Onur Mutlu,

"GenStore: A High-Performance and Energy-Efficient In-Storage Computing System for Genome Sequence Analysis"


[Talk Slides (pptx) (pdf)]
[Lightning Talk Slides (pptx) (pdf)]
[Lightning Talk Video (90 seconds)]
[Talk Video (17 minutes)]

GenStore: A High-Performance In-Storage Processing System for Genome Sequence Analysis

Nika Mansouri Ghiasi\(^1\)  Jisung Park\(^1\)  Harun Mustafa\(^1\)  Jeremie Kim\(^1\)  Ataberk Olgun\(^1\)  Arvid Gollwitzer\(^1\)  Damla Senol Cali\(^2\)  Can Firtina\(^1\)  Haiyu Mao\(^1\)  Nour Almadhoun Alserr\(^1\)  Rachata Ausavarungnirun\(^3\)  Nandita Vijaykumar\(^4\)  Mohammed Alser\(^1\)  Onur Mutlu\(^1\)

\(^1\)ETH Zürich \(^2\)Bionano Genomics \(^3\)KMUTNB \(^4\)University of Toronto
Accelerating Sequence-to-Graph Mapping

- Damla Senol Cali, Konstantinos Kanellopoulos, Joel Lindegger, Zulal Bingol, Gurpreet S. Kalsi, Ziyi Zuo, Can Firtina, Meryem Banu Cavlak, Jeremie Kim, Nika MansouriGhiasi, Gagandeep Singh, Juan Gomez-Luna, Nour Almadhoun Alserr, Mohammed Alser, Sreenivas Subramoney, Can Alkan, Saugata Ghose, and Onur Mutlu,

*SeGraM: A Universal Hardware Accelerator for Genomic Sequence-to-Graph and Sequence-to-Sequence Mapping*


[arXiv version]

SeGraM: A Universal Hardware Accelerator for Genomic Sequence-to-Graph and Sequence-to-Sequence Mapping

- Damla Senol Cali\(^1\)  Konstantinos Kanellopoulos\(^2\)  Joël Lindegger\(^2\)  Züla\l\ Bingöl\(^3\)
- Gurpreet S. Kalsi\(^4\)  Ziyi Zuo\(^5\)  Can Firtina\(^2\)  Meryem Banu Cavlak\(^2\)  Jeremie Kim\(^2\)
- Nika Mansouri Ghiasi\(^2\)  Gagandeep Singh\(^2\)  Juan Gómez-Luna\(^2\)  Nour Almadhoun Alserr\(^2\)
- Mohammed Alser\(^2\)  Sreenivas Subramoney\(^4\)  Can Alkan\(^3\)  Saugata Ghose\(^6\)  Onur Mutlu\(^2\)

\(^1\)Bionano Genomics  \(^2\)ETH Zürich  \(^3\)Bilkent University  \(^4\)Intel Labs
\(^5\)Carnegie Mellon University  \(^6\)University of Illinois Urbana-Champaign

Future of Genome Sequencing & Analysis

Mohammed Alser, Zülal Bingöl, Damla Senol Cali, Jeremie Kim, Saugata Ghose, Can Alkan, Onur Mutlu


Accelerating Genome Analysis: A Primer on an Ongoing Journey
DOI Bookmark: 10.1109/MM.2020.3013728

FPGA-Based Near-Memory Acceleration of Modern Data-Intensive Applications
DOI Bookmark: 10.1109/MM.2021.3088396

MinION from ONT

SmidgION from ONT
More on Fast & Efficient Genome Analysis …

- Onur Mutlu,
  "Accelerating Genome Analysis: A Primer on an Ongoing Journey"
  Invited Lecture at Technion, Virtual, 26 January 2021.
  [Slides (pptx) (pdf)]
  [Talk Video (1 hour 37 minutes, including Q&A)]
  [Related Invited Paper (at IEEE Micro, 2020)]
Detailed Lectures on Genome Analysis

- **Computer Architecture, Fall 2020, Lecture 3a**
  - **Introduction to Genome Sequence Analysis** (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=CrRb32v7SJc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=5](https://www.youtube.com/watch?v=CrRb32v7SJc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=5)

- **Computer Architecture, Fall 2020, Lecture 8**
  - **Intelligent Genome Analysis** (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=ygmQpdDTL7o&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=14](https://www.youtube.com/watch?v=ygmQpdDTL7o&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=14)

- **Computer Architecture, Fall 2020, Lecture 9a**
  - **GenASM: Approx. String Matching Accelerator** (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=XoLpzmNPas&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=15](https://www.youtube.com/watch?v=XoLpzmNPas&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=15)

- **Accelerating Genomics Project Course, Fall 2020, Lecture 1**
  - **Accelerating Genomics** (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=rgjl8ZyLsAg&list=PL5Q2soXY2Zi9E2bBVAgCqLgwiDRQDTyId](https://www.youtube.com/watch?v=rgjl8ZyLsAg&list=PL5Q2soXY2Zi9E2bBVAgCqLgwiDRQDTyId)

[https://www.youtube.com/onurmutlulectures](https://www.youtube.com/onurmutlulectures)
Increasingly Demanding Applications

Dream...

More will come...

As applications push boundaries, computing platforms will become increasingly strained.
Data Overwhelms Modern Machines …

- Storage/memory capability
- Communication capability
- Computation capability
- *Greatly impacts robustness, energy, performance, cost*
A Computing System

- Three key components
- Computation
- Communication
- Storage/memory

Burks, Goldstein, von Neumann, “Preliminary discussion of the logical design of an electronic computing instrument,” 1946.
Perils of Processor-Centric Design

Most of the system is dedicated to storing and moving data.

Yet, system is still bottlenecked by memory.
Deeper and Larger Memory Hierarchies

Core Count: 8 cores/16 threads
L1 Caches: 32 KB per core
L2 Caches: 512 KB per core
L3 Cache: 32 MB shared

AMD Ryzen 5000, 2020

AMD’s 3D Last Level Cache (2021)

AMD increases the L3 size of their 8-core Zen 3 processors from 32 MB to 96 MB

Additional 64 MB L3 cache die stacked on top of the processor die
- Connected using Through Silicon Vias (TSVs)
- Total of 96 MB L3 cache

https://youtu.be/gqAYMx34euU
https://www.tech-critter.com/amd-keynote-computex-2021/
Deeper and Larger Memory Hierarchies

IBM POWER10, 2020

Cores:
15-16 cores, 8 threads/core

L2 Caches:
2 MB per core

L3 Cache:
120 MB shared
Deeper and Larger Memory Hierarchies

Apple M1 Ultra System (2022)
Data Overwhelms Modern Machines

Chrome

Google’s web browser

TensorFlow Mobile

Google’s machine learning framework

Data → performance & energy bottleneck

VP9

Video Playback

Google’s video codec

VP9

Video Capture

Google’s video codec
62.7% of the total system energy is spent on data movement
An Intelligent Architecture
Handles Data Well
How to Handle Data Well

- Ensure data does not overwhelm the components
  - via intelligent algorithms
  - via intelligent architectures
  - via whole system designs: algorithm-architecture-devices

- Take advantage of vast amounts of data and metadata
  - to improve architectural & system-level decisions

- Understand and exploit properties of (different) data
  - to improve algorithms & architectures in various metrics
Corollaries: Computing Systems Today …

- Are processor-centric vs. data-centric

- Make designer-dictated decisions vs. data-driven

- Make component-based myopic decisions vs. data-aware
Fundamentally Better Architectures

Data-centric

Data-driven

Data-aware
We Need to Revisit the Entire Stack

We can get there step by step
A Blueprint for Fundamentally Better Architectures

Onur Mutlu,
"Intelligent Architectures for Intelligent Computing Systems"
[Slides (pptx) (pdf)]
[IEDM Tutorial Slides (pptx) (pdf)]
[Short DATE Talk Video (11 minutes)]
[Longer IEDM Tutorial Video (1 hr 51 minutes)]

Intelligent Architectures for Intelligent Computing Systems

Onur Mutlu
ETH Zurich
omutlu@gmail.com
Data-Centric (Memory-Centric) Architectures
Data-Centric Architectures: Properties

- **Process data where it resides** *(where it makes sense)*
  - Processing in and near memory structures

- **Low-latency and low-energy data access**
  - Low latency memory
  - Low energy memory

- **Low-cost data storage and processing**
  - High capacity memory at low cost: hybrid memory, compression

- **Intelligent data management**
  - Intelligent controllers handling robustness, security, cost, perf.
Processing Data
Where It Makes Sense
Process Data Where It Makes Sense

Apple M1 Ultra System (2022)

https://www.gsmarena.com/apple_announces_m1_ultra_with_20core_cpu_and_64core_gpu-news-53481.php
Processing in/near Memory: An Old Idea


IEEE TRANSACTIONS ON COMPUTERS, VOL. C-18, NO. 8, AUGUST 1969

Cellular Logic-in-Memory Arrays

WILLIAM H. KAUTZ, MEMBER, IEEE

Abstract—As a direct consequence of large-scale integration, many advantages in the design, fabrication, testing, and use of digital circuitry can be achieved if the circuits can be arranged in a two-dimensional iterative, or cellular, array of identical elementary networks, or cells. When a small amount of storage is included in each cell, the same array may be regarded either as a logically enhanced memory array, or as a logic array whose elementary gates and connections can be “programmed” to realize a desired logical behavior.

In this paper the specific engineering features of such cellular logic-in-memory (CLIM) arrays are discussed, and one such special-purpose array, a cellular sorting array, is described in detail to illustrate how these features may be achieved in a particular design. It is shown how the cellular sorting array can be employed as a single-address, multiword memory that keeps in order all words stored within it. It can also be used as a content-addressed memory, a pushdown memory, a buffer memory, and (with a lower logical efficiency) a programmable array for the realization of arbitrary switching functions. A second version of a sorting array, operating on a different sorting principle, is also described.

Index Terms—Cellular logic, large-scale integration, logic arrays logic in memory, push-down memory, sorting, switching functions.

Fig. 1. Cellular sorting array I.

A Logic-in-Memory Computer

HAROLD S. STONE

Abstract—If, as presently projected, the cost of microelectronic arrays in the future will tend to reflect the number of pins on the array rather than the number of gates, the logic-in-memory array is an extremely attractive computer component. Such an array is essentially a microelectronic memory with some combinational logic associated with each storage element.
Why In-Memory Computation Today?

- **Push from Technology**
  - Memory technology scaling is not going well (e.g., RowHammer)
  - Many scaling issues demand intelligence in memory

- **Pull from Applications & Systems**
  - Data access is the major bottleneck
  - Systems are energy & power limited
  - Data movement much more energy-hungry than computation
  - Need all at the same time: performance, energy, sustainability
  - We can improve all metrics by minimizing data movement

- **Designs are squeezed in the middle**
Why In-Memory Computation Today?

- **Push from Technology**
  - DRAM Scaling at jeopardy
    - Controllers close to DRAM
    - Industry open to new memory architectures
Why In-Memory Computation Today?

And, many other experimental chips and startups
Memory Scaling Issues Were Real

- 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/

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.

**Intuition:** quadratic increase in capacity
A Curious Phenomenon [Kim et al., ISCA 2014]

One can predictably induce errors in most DRAM memory chips

Rowhammer
Repeatedly reading a row enough times (before memory gets refreshed) induces disturbance errors in adjacent rows in most real DRAM chips you can buy today.
Most DRAM Modules Are Vulnerable

A company

86%
(37/43)

Up to
1.0\times 10^7
errors

B company

83%
(45/54)

Up to
2.7\times 10^6
errors

C company

88%
(28/32)

Up to
3.3\times 10^5
errors

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors, (Kim et al., ISCA 2014)
The RowHammer Vulnerability

A simple hardware failure mechanism can create a widespread system security vulnerability
Onur Mutlu,
"Security Aspects of DRAM: The Story of RowHammer"
[Slides (pptx)(pdf)]
[Tutorial Video (57 minutes)]
Onur Mutlu, "The Story of RowHammer"
Invited Talk at the Workshop on Robust and Safe Software 2.0 (RSS2), held with the 27th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), Virtual, 28 February 2022.

[Slides (pptx) (pdf)]

https://www.youtube.com/watch?v=ctKTRyi96Bk
Memory Scaling Issues Are Real

- Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu,

"Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors"
[Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Source Code and Data] [Lecture Video (1 hr 49 mins), 25 September 2020]

One of the 7 papers of 2012-2017 selected as Top Picks in Hardware and Embedded Security for IEEE TCAD (link).

Flipping Bits in Memory Without Accessing Them:
An Experimental Study of DRAM Disturbance Errors

Yoongu Kim¹  Ross Daly*  Jeremie Kim¹  Chris Fallin*  Ji Hye Lee¹
Donghyuk Lee¹  Chris Wilkerson²  Konrad Lai  Onur Mutlu¹
¹Carnegie Mellon University  ²Intel Labs
Onur Mutlu,
"The RowHammer Problem and Other Issues We May Face as Memory Becomes Denser"
[Slides (pptx) (pdf)]
Memory Scaling Issues Are Real

- Onur Mutlu and Jeremie Kim, "RowHammer: A Retrospective"
  [Preliminary arXiv version]
  [Slides from COSADE 2019 (pptx)]
  [Slides from VLSI-SOC 2020 (pptx) (pdf)]
  [Talk Video (1 hr 15 minutes, with Q&A)]
Hybrid Memory Enables Better Scaling

Yoon+, “Row Buffer Locality Aware Caching Policies for Hybrid Memories,” ICCD 2012 Best Paper Award.
The Push from Circuits and Devices

Main Memory Needs
Intelligent Controllers
An Example Intelligent Controller

- A. Giray Yaglıkci, Minesh Patel, Jeremie S. Kim, Roknoddin Azizi, Ataberk Olgun, Lois Orosa, Hasan Hassan, Jisung Park, Konstantinos Kanellopoulos, Taha Shahroodi, Saugata Ghose, and Onur Mutlu,

"BlockHammer: Preventing RowHammer at Low Cost by Blacklisting Rapidly-Accessed DRAM Rows"


[Slides (pptx) (pdf)]
[Short Talk Slides (pptx) (pdf)]
[Intel Hardware Security Academic Awards Short Talk Slides (pptx) (pdf)]
[Talk Video (22 minutes)]
[Short Talk Video (7 minutes)]
[Intel Hardware Security Academic Awards Short Talk Video (2 minutes)]
[BlockHammer Source Code]

Intel Hardware Security Academic Award Finalist (one of 4 finalists out of 34 nominations)

BlockHammer: Preventing RowHammer at Low Cost by Blacklisting Rapidly-Accessed DRAM Rows

A. Giray Yağlıkçı¹, Minesh Patel¹, Jeremie S. Kim¹, Roknoddin Azizi¹, Ataberk Olgun¹, Lois Orosa¹, Hasan Hassan¹, Jisung Park¹, Konstantinos Kanellopoulos¹, Taha Shahroodi¹, Saugata Ghose², Onur Mutlu¹

¹ETH Zürich ²University of Illinois at Urbana–Champaign
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
Another Example Intelligent Controller


[Slides (pptx) (pdf)]
[Short Talk Slides (pptx) (pdf)]
[Lightning Talk Slides (pptx) (pdf)]
[Talk Video (20 minutes)]
[Lightning Talk Video (1.5 minutes)]
[HARP Source Code (Officially Artifact Evaluated with All Badges)]

HARP: Practically and Effectively Identifying Uncorrectable Errors in Memory Chips That Use On-Die Error-Correcting Codes

Minesh Patel
ETH Zürich

Geraldo F. Oliveira
ETH Zürich

Onur Mutlu
ETH Zürich
Aside: Intelligent Controller for NAND Flash

HAPS-52 Mother Board

USB Daughter Board

Virtex-V FPGA (NAND Controller)

Virtex-Ⅳ Pro (USB controller)

1x-nm NAND Flash

NAND Daughter Board

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
Emerging Memories Also Need Intelligent Controllers


One of the 13 computer architecture papers of 2009 selected as Top Picks by IEEE Micro. Selected as a CACM Research Highlight. 2022 Persistent Impact Prize.

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
Intelligent Memory Controllers Can Avoid Many Failures & Enable Better Scaling
The Push from Circuits and Devices

Main Memory Needs

Intelligent Controllers
Why In-Memory Computation Today?

- **Push from Technology**
  - DRAM Scaling at jeopardy
    → Controllers close to DRAM
    → Industry open to new memory architectures

- **Pull from Systems and Applications**
  - Data access is the major system and application bottleneck
  - Systems are energy & power limited
  - Data movement much more energy-hungry than computation
Three Key Systems & Application Trends

1. Data access is the major bottleneck
   - Applications are increasingly data hungry

2. Energy consumption is a key limiter

3. Data movement energy dominates compute
   - Especially true for off-chip to on-chip movement
Do We Want This?

Source: V. Milutinovic
Or This?

Source: V. Milutinovic
Challenge and Opportunity for Future

High Performance, Energy Efficient, Sustainable (All at the Same Time)
The Problem

Data access is the major performance and energy bottleneck

Our current design principles cause great energy waste
(and great performance loss)
The Problem

Processing of data is performed far away from the data
A Computing System

- Three key components
  - Computation
  - Communication
  - Storage/memory

Burks, Goldstein, von Neumann, “Preliminary discussion of the logical design of an electronic computing instrument,” 1946.
A Computing System

- Three key components
- Computation
- Communication
- Storage/memory

Burks, Goldstein, von Neumann, “Preliminary discussion of the logical design of an electronic computing instrument,” 1946.

Today’s Computing Systems

- Are overwhelmingly processor centric
- All data processed in the processor → at great system cost
- Processor is heavily optimized and is considered the master
- Data storage units are dumb and are largely unoptimized

![Diagram of computing system with computing unit, communication unit, memory/storage unit, memory system, and storage system]
Yet …

- “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.

The Performance Perspective

- Onur Mutlu, Jared Stark, Chris Wilkerson, and Yale N. Patt, "Runahead Execution: An Alternative to Very Large Instruction Windows for Out-of-order Processors"
  One of the 15 computer arch. papers of 2003 selected as Top Picks by IEEE Micro. HPCA Test of Time Award (awarded in 2021).

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 Performance Perspective (Today)

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

The Performance Perspective (Today)

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

![Graph showing cache-bound cycles for different workloads](image)

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

Perils of Processor-Centric Design

- Grossly-imbalanced systems
  - Processing done only in **one place**
  - All else just stores and moves data: **data moves a lot**
    - Energy inefficient
    - Low performance
    - Complex

- Overly complex and bloated processor (and accelerators)
  - To tolerate data access from memory
  - Complex hierarchies and mechanisms
    - Energy inefficient
    - Low performance
    - Complex
Most of the system is dedicated to storing and moving data.

Yet, system is still bottlenecked by memory.

Perils of Processor-Centric Design
The Energy Perspective

Communication Dominates Arithmetic

Dally, HiPEAC 2015

- 64-bit DP: 20pJ
- 256-bit buses
- 256-bit access: 8 kB SRAM
- 26 pJ
- 256 pJ
- 16 nJ
- 500 pJ
- 1 nJ

Efficient off-chip link

DRAM Rd/Wr
A memory access consumes \( \sim 100-1000X \) the energy of a complex addition.
Data Movement vs. Computation Energy

Energy for a 32-bit Operation (log scale)

- ADD (int)
- ADD (float)
- Register File
- MULT (int)
- MULT (float)
- SRAM Cache
- DRAM

Energy (pJ) vs. ADD (int) Relative Cost

Data Movement vs. Computation Energy

A memory access consumes 6400X the energy of a simple integer addition.
Data Movement vs. Computation Energy

- **Data movement is a major system energy bottleneck**
  - Comprises 41% of mobile system energy during web browsing [2]
  - Costs ~115 times as much energy as an ADD operation [1, 2]

[1]: Reducing data Movement Energy via Online Data Clustering and Encoding (MICRO’16)
[2]: Quantifying the energy cost of data movement for emerging smart phone workloads on mobile platforms (IISWC’14)
Energy Waste in Mobile Devices

- Amirali Boroumand, Saugata Ghose, Youngsok Kim, Rachata Ausavarungnirun, Eric Shiu, Rahul Thakur, Daehyun Kim, Aki Kuusela, Allan Knies, Parthasarathy Ranganathan, and Onur Mutlu, "Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks"

62.7% of the total system energy is spent on data movement

Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks

Amirali Boroumand¹ Rachata Ausavarungnirun¹
Saugata Ghose¹ Youngsok Kim²
Eric Shiu³ Rahul Thakur³
Daehyun Kim⁴,³ Onur Mutlu⁵,¹
Aki Kuusela³ Allan Knies³
Parthasarathy Ranganathan³

SAFARI
We Do Not Want to Move Data!

Communication Dominates Arithmetic

Dally, HiPEAC 2015

A memory access consumes ~100-1000X the energy of a complex addition
We Need A **Paradigm Shift** To …

- Enable computation with *minimal data movement*

- **Compute where it makes sense** *(where data resides)*

- Make computing architectures more *data-centric*
Goal: Processing Inside Memory

Many questions ... How do we design the:

- compute-capable memory & controllers?
- processors & communication units?
- software & hardware interfaces?
- system software, compilers, languages?
- algorithms & theoretical foundations?

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, and Rachata Ausavarungnirun.

A Modern Primer on Processing in Memory

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

SAFARI Research Group

\textsuperscript{a}ETH Zürich
\textsuperscript{b}Carnegie Mellon University
\textsuperscript{c}University of Illinois at Urbana-Champaign
\textsuperscript{d}King Mongkut's University of Technology North Bangkok

Abstract

Modern computing systems are overwhelmingly designed to move data to computation. This design choice goes directly against at least three key trends in computing that cause performance, scalability and energy bottlenecks: (1) data access is a key bottleneck as many important applications are increasingly data-intensive, and memory bandwidth and energy do not scale well, (2) energy consumption is a key limiter in almost all computing platforms, especially server and mobile systems, (3) data movement, especially off-chip to on-chip, is very expensive in terms of bandwidth, energy and latency, much more so than computation. These trends are especially severely-felt in the data-intensive server and energy-constrained mobile systems of today.

At the same time, conventional memory technology is facing many technology scaling challenges in terms of reliability, energy, and performance. As a result, memory system architects are open to organizing memory in different ways and making it more intelligent, at the expense of higher cost. The emergence of 3D-stacked memory plus logic, the adoption of error correcting codes inside the latest DRAM chips, proliferation of different main memory standards and chips, specialized for different purposes (e.g., graphics, low-power, high bandwidth, low latency), and the necessity of designing new solutions to serious reliability and security issues, such as the RowHammer phenomenon, are an evidence of this trend.

This chapter discusses recent research that aims to practically enable computation close to data, an approach we call processing-in-memory (PIM). PIM places computation mechanisms in or near where the data is stored (i.e., inside the memory chips, in the logic layer of 3D-stacked memory, or in the memory controllers), so that data movement between the computation units and memory is reduced or eliminated. While the general idea of PIM is not new, we discuss motivating trends in applications as well as memory circuits/technology that greatly exacerbate the need for enabling it in modern computing systems. We examine at least two promising new approaches to designing PIM systems to accelerate important data-intensive applications: (1) processing using memory by exploiting analog operational properties of DRAM chips to perform massively-parallel operations in memory, with low-cost changes, (2) processing near memory by exploiting 3D-stacked memory technology design to provide high memory bandwidth and low memory latency to in-memory logic. In both approaches, we describe and tackle relevant cross-layer research, design, and adoption challenges in devices, architecture, systems, and programming models. Our focus is on the development of in-memory processing designs that can be adopted in real computing platforms at low cost. We conclude by discussing work on solving key challenges to the practical adoption of PIM.

Keywords: memory systems, data movement, main memory, processing-in-memory, near-data processing, computation-in-memory, processing using memory, processing near memory, 3D-stacked memory, non-volatile memory, energy efficiency, high-performance computing, computer architecture, computing paradigm, emerging technologies, memory scaling, technology scaling, dependable systems, robust systems, hardware security, system security, latency, low-latency computing
1. Introduction

Main memory, built using the Dynamic Random Access Memory (DRAM) technology, is a major component in nearly all computing systems, including servers, cloud platforms, mobile/embedded devices, and sensor systems. Across all of these systems, the data working set sizes of modern applications are rapidly growing, while the need for fast analysis of such data is increasing. Thus, main memory is becoming an increasingly significant bottleneck across a wide variety of computing systems and applications [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]. Alleviating the main memory bottleneck requires the memory capacity, energy, cost, and performance to all scale in an efficient manner across technology generations. Unfortunately, it has become increasingly difficult in recent years, especially the past decade, to scale all of these dimensions [1, 2, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54].

A major reason for the main memory bottleneck is the high energy and latency cost associated with data movement. In modern computers, to perform any operation on data that resides in main memory, the processor must retrieve the data from main memory. This requires the memory controller to issue commands to a DRAM module across a relatively slow and power-hungry off-chip bus (known as the memory channel). The DRAM module sends the requested data across the memory channel, after which the data is placed in the caches and registers. The CPU can perform computation on the data once the data is in its registers. Data movement from the DRAM to the CPU incurs long latency and consumes a significant amount of energy [7, 50, 51, 52, 53, 54]. These costs are often exacerbated by the fact that much of the data brought into the caches is not reused by the CPU [52, 53, 54, 55], providing little benefit in return for the high latency and energy cost.

The cost of data movement is a fundamental issue with the processor-centric nature of contemporary computer systems. The CPU is considered to be the master in the system, and computation is performed only in the processor (and accelerators). In contrast, data storage and communication units, including the main memory, are treated as unintelligent workers that are incapable of computation. As a result of this processor-centric design paradigm, data moves a lot in the system between the computation units and communication/storage units so that computation can be done on it. With the increasingly data-centric nature of contemporary and emerging appli-
We Need to Think Differently from the Past Approaches
Processing in Memory: Two Approaches

1. Processing using Memory
2. Processing near Memory
Approach 1: Processing Using Memory

- Take advantage of operational principles of memory to perform bulk data movement and computation in memory
  - Can exploit internal connectivity to move data
  - Can exploit analog computation capability
  - ...

Examples: RowClone, In-DRAM AND/OR, Gather/Scatter DRAM

- RowClone: Fast and Efficient In-DRAM Copy and Initialization of Bulk Data (Seshadri et al., MICRO 2013)
- Fast Bulk Bitwise AND and OR in DRAM (Seshadri et al., IEEE CAL 2015)
- Gather-Scatter DRAM: In-DRAM Address Translation to Improve the Spatial Locality of Non-unit Strided Accesses (Seshadri et al., MICRO 2015)
- "Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology” (Seshadri et al., MICRO 2017)
Starting Simple: Data Copy and Initialization

*memmove & memcpy: 5% cycles in Google’s datacenter* [Kanev+ ISCA’15]

- Forking
- Zero initialization (e.g., security)
- Checkpointing
- VM Cloning
- Deduplication
- Page Migration
- Many more

5% cycles in Google’s datacenter [Kanev+ ISCA’15]
Future Systems: In-Memory Copy

1) Low latency
2) Low bandwidth utilization
3) No cache pollution
4) No unwanted data movement

1046ns, 3.6µJ  ➔  90ns, 0.04µJ
RowClone: In-DRAM Row Copy

Idea: Two consecutive ACTivates

Negligible HW cost

Step 1: Activate row A
Step 2: Activate row B

DRAM subarray

4 Kbytes

Row Buffer (4 Kbytes)

8 bits

Data Bus
RowClone: Latency and Energy Savings

More on RowClone

Vivek Seshadri, Yoongu Kim, Chris Fallin, Donghyuk Lee, Rachata Ausavarungnirun, Gennady Pekhimenko, Yixin Luo, Onur Mutlu, Michael A. Kozuch, Phillip B. Gibbons, and Todd C. Mowry,

"RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization"

Proceedings of the 46th International Symposium on Microarchitecture (MICRO), Davis, CA, December 2013. [Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Poster (pptx) (pdf)]
RowClone in Off-the-Shelf DRAM Chips

- Idea: Violate DRAM timing parameters to mimic RowClone

ComputeDRAM: In-Memory Compute Using Off-the-Shelf DRAMs

Fei Gao  
feig@princeton.edu  
Department of Electrical Engineering  
Princeton University

Georgios Tziantzioulis  
georgios.tziantzioulis@princeton.edu  
Department of Electrical Engineering  
Princeton University

David Wentzlafl  
wentzlaf@princeton.edu  
Department of Electrical Engineering  
Princeton University

Real Processing Using Memory Prototype

- End-to-end RowClone & TRNG using off-the-shelf DRAM chips
- Idea: Violate DRAM timing parameters to mimic RowClone

PiDRAM: A Holistic End-to-end FPGA-based Framework for Processing-in-DRAM

Ataberk Olgun$^\dagger$  Juan Gómez Luna$^\S$  Hasan Hassan$^\S$  Konstantinos Kanellopoulos$^\S$  Behzad Salami$^\S^*$

$^\S$ETH Zürich  $^\dagger$TOBB ETÜ  $^*$BSC

https://github.com/cmu-safari/pidram
https://www.youtube.com/watch?v=qeukNs5XI3g&t=4192s
Real Processing-using-Memory Prototype

https://github.com/cmu-safari/pidram
https://www.youtube.com/watch?v=qeukNs5XI3g&t=4192s
Real Processing - using Memory Prototype

Building a PiDRAM Prototype

To build PiDRAM's prototype on Xilinx ZC706 boards, developers need to use the two sub-projects in this directory. `fpga-zynq` is a repository branched off of UCB-BAR's `fpga-zynq` repository. We use `fpga-zynq` to generate rocket chip designs that support end-to-end DRAM PuM execution. `controller-hardware` is where we keep the main Vivado project and Verilog sources for PiDRAM's memory controller and the top level system design.

Rebuilding Steps

1. Navigate into `fpga-zynq` and read the README file to understand the overall workflow of the repository
   - Follow the readme in `fpga-zynq/rocket-chip/riscv-tools` to install dependencies
2. Create the Verilog source of the rocket chip design using the `ZynqCopyFPGACfg`
   - Navigate into zc706, then run `make rocket CONFIG=ZynqCopyFPGACfg -j number_of_cores`
3. Copy the generated Verilog file (should be under zc706/src) and overwrite the same file in `controller-hardware/source/hdl/impl/rocket-chip`
4. Open the Vivado project in `controller-hardware/Vivado_Project` using Vivado 2016.2
5. Generate a bitstream
6. Copy the bitstream (`system_top.bit`) to `fpga-zynq/zc706`
7. Use the `./build_script.sh` to generate the new `boot.bin` under `fpga-images-zc706`, you can use this file to program the FPGA using the SD-Card.
   - For details, follow the relevant instructions in `fpga-zynq/README.md`

Generating DDR3 Controller IP sources

You can run programs compiled with the RISC-V Toolchain supplied within the `fpga-zynq` repository. To install the toolchain, follow the instructions under `fpga-zynq/rocket-chip/riscv-tools`.

https://github.com/cmu-safari/pidram
https://www.youtube.com/watch?v=qeukNs5XI3g&t=4192s
In-DRAM Copy and Initialization improve throughput by 119x and 89x
Lecture on RowClone & Processing using DRAM

Mindset: Memory as an Accelerator

Memory similar to a “conventional” accelerator

https://www.youtube.com/watch?v=n6Pwg1qax_E&list=PL5Q2soXY2Zi_7UBNmC9B8Yr5JSwTG9yH4&index=4
Mindset: Memory as an Accelerator

Memory similar to a “conventional” accelerator
(Truly) In-Memory Computation

- We can support in-DRAM AND, OR, NOT, MAJ
- At low cost
- Using analog computation capability of DRAM
  - Idea: activating multiple rows performs computation
- 30-60X performance and energy improvement

- New memory technologies enable even more opportunities
  - Memristors, resistive RAM, phase change mem, STT-MRAM, ...
  - Can operate on data with minimal movement
In-DRAM AND/OR: Triple Row Activation

Final State
\[ AB + BC + AC \]

\[ C(A + B) + \sim C(AB) \]
Bulk Bitwise Operations in Workloads

- Bitmap indices (database indexing)
- Set operations
- Encryption algorithms
- BitWeaving (database queries)
- BitFunnel (web search)
- DNA sequence mapping

[1] Li and Patel, BitWeaving, SIGMOD 2013
In-DRAM Acceleration of Database Queries

'Select count(*) from T where c1 <= val <= c2'

Row count \( r \) = 1m, 2m, 4m, 8m

Speedup offered by Ambit

>4-12X Performance Improvement

Number of Bits per Column \( b \)

Figure 11: Speedup offered by Ambit over baseline CPU with SIMD for BitWeaving

More on Ambit

- Vivek Seshadri, Donghyuk Lee, Thomas Mullins, Hasan Hassan, Amirali Boroumand, Jeremie Kim, Michael A. Kozuch, Onur Mutlu, Phillip B. Gibbons, and Todd C. Mowry,

"Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology"
Proceedings of the 50th International Symposium on Microarchitecture (MICRO), Boston, MA, USA, October 2017.
[Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Poster (pptx) (pdf)]
In-DRAM Bulk Bitwise Execution

Vivek Seshadri and Onur Mutlu,
"In-DRAM Bulk Bitwise Execution Engine"
[Preliminary arXiv version]

In-DRAM Bulk Bitwise Execution Engine

Vivek Seshadri
Microsoft Research India
visesha@microsoft.com

Onur Mutlu
ETH Zürich
onur.mutlu@inf.ethz.ch
SIMDRAM: A Framework for Bit-Serial SIMD Processing using DRAM

*Nastaran Hajinazar\textsuperscript{1,2}  
Nika Mansour\textsuperscript{1} Ghiasi  
\\  
*Geraldo F. Oliveira\textsuperscript{1}  
Minesh Patel\textsuperscript{1}  
Juan Gómez-Luna\textsuperscript{1}  
Sven Gregorio\textsuperscript{1}  
Mohammed Alser\textsuperscript{1}  
Onur Mutlu\textsuperscript{1}  
João Dinis Ferreira\textsuperscript{1}  
Saugata Ghose\textsuperscript{3}  

\textsuperscript{1}ETH Zürich  
\textsuperscript{2}Simon Fraser University  
\textsuperscript{3}University of Illinois at Urbana–Champaign
SIMDRAM Framework: Overview

**User Input**

Desired operation

AND/OR/NOT logic

 Desired operation

**Step 1: Generate MAJ logic**

MAJ

MAJ/NOT logic

**Step 2: Generate sequence of DRAM commands**

- ACT/PRE
- ACT/PRE
- ACT/PRE
- ACT/ACT/PRE
done

**SIMDRAM Output**

New SIMDRAM \( \mu \)Program

\( \mu \)Program

Main memory

ISA

**User Input**

SIMDRAM-enabled application

```c
foo () {
    bbop_new
}
```

**Step 3: Execution according to \( \mu \)Program**

Control Unit

\( \mu \)Program

**SIMDRAM Output**

Instruction result in memory

- ACT/PRE
- ACT/PRE
- ACT/ACT/PRE
- ACT/PRE/PRE
done

Memory Controller
SIMDRAM Key Results

Evaluated on:
- 16 complex in-DRAM operations
- 7 commonly-used real-world applications

SIMDRAM provides:

• 88× and 5.8× the throughput of a CPU and a high-end GPU, respectively, over 16 operations

• 257× and 31× the energy efficiency of a CPU and a high-end GPU, respectively, over 16 operations

• 21× and 2.1× the performance of a CPU an a high-end GPU, over seven real-world applications

SAFARI
More on SIMDREAM


SIMDRAM: A Framework for Bit-Serial SIMD Processing using DRAM

*Nastaran Hajinazar\textsuperscript{1,2} Nika Mansouri Ghiasi\textsuperscript{1} Geraldo F. Oliveira\textsuperscript{1} Minesh Patel\textsuperscript{1} Sven Gregorio\textsuperscript{1} João Dinis Ferreira\textsuperscript{1}

*Geraldo F. Oliveira\textsuperscript{1} Minesh Patel\textsuperscript{1} Sven Gregorio\textsuperscript{1} Mohammed Alser\textsuperscript{1} Onur Mutlu\textsuperscript{1}

Juan Gómez-Luna\textsuperscript{1} Saugata Ghose\textsuperscript{3}

\textsuperscript{1}ETH Zürich \hspace{1cm} \textsuperscript{2}Simon Fraser University \hspace{1cm} \textsuperscript{3}University of Illinois at Urbana–Champaign
In-DRAM Physical Unclonable Functions

Jeremie S. Kim, Minesh Patel, Hasan Hassan, and Onur Mutlu,
"The DRAM Latency PUF: Quickly Evaluating Physical Unclonable Functions by Exploiting the Latency-Reliability Tradeoff in Modern DRAM Devices"
[Lightning Talk Video]
[Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)]
[Full Talk Lecture Video (28 minutes)]

The DRAM Latency PUF: Quickly Evaluating Physical Unclonable Functions by Exploiting the Latency-Reliability Tradeoff in Modern Commodity DRAM Devices

Jeremie S. Kim$^\dagger\$ Minesh Patel$\$ Hasan Hassan$\$ Onur Mutlu$\dagger$
$\dagger$Carnegie Mellon University $\$ETH Zürich

SAFARI
In-DRAM True Random Number Generation

- Jeremie S. Kim, Minesh Patel, Hasan Hassan, Lois Orosa, and Onur Mutlu, "D-RaNGe: Using Commodity DRAM Devices to Generate True Random Numbers with Low Latency and High Throughput"
  
  
  **Slides (pptx) (pdf)**
  **[Full Talk Video](#) (21 minutes)**
  **[Full Talk Lecture Video](#) (27 minutes)**

*Top Picks Honorable Mention by IEEE Micro.*

---

D-RaNGe: Using Commodity DRAM Devices to Generate True Random Numbers with Low Latency and High Throughput

Jeremie S. Kim‡§ Minesh Patel§ Hasan Hassan§ Lois Orosa§ Onur Mutlu‡§

‡Carnegie Mellon University §ETH Zürich
In-DRAM True Random Number Generation

- Ataberk Olgun, Minesh Patel, A. Giray Yaglikci, Haocong Luo, Jeremie S. Kim, F. Nisa Bostanci, Nandita Vijaykumar, Oguz Ergin, and Onur Mutlu,

"QUAC-TRNG: High-Throughput True Random Number Generation Using Quadruple Row Activation in Commodity DRAM Chips"


[Slides (pptx) (pdf)]
[Short Talk Slides (pptx) (pdf)]
[Talk Video (25 minutes)]
[SAFARI Live Seminar Video (1 hr 26 mins)]

QUAC-TRNG: High-Throughput True Random Number Generation Using Quadruple Row Activation in Commodity DRAM Chips

Ataberk Olgun§† Minesh Patel§ A. Giray Yağlıkçı§ Haocong Luo§
Jeremie S. Kim§ F. Nisa Bostancı§† Nandita Vijaykumar§○ Oguz Ergin† Onur Mutlu§

§ETH Zürich †TOBB University of Economics and Technology ○University of Toronto
In-DRAM True Random Number Generation

- F. Nisa Bostanci, Ataberk Olgun, Lois Orosa, A. Giray Yağlıkçı, Jeremie S. Kim, Hasan Hassan, Oğuz Ergin, and Onur Mutlu,
  "DR-STRaNGe: End-to-End System Design for DRAM-based True Random Number Generators"
  Proceedings of the 28th International Symposium on High-Performance Computer Architecture (HPCA), Virtual, April 2022.
  [Slides (pptx) (pdf)]
  [Short Talk Slides (pptx) (pdf)]

DR-STRaNGe: End-to-End System Design for DRAM-based True Random Number Generators

F. Nisa Bostanci†§  Ataberk Olgun†§  Lois Orosa§  A. Giray Yağlıkçı§
Jeremie S. Kim§  Hasan Hassan§  Oğuz Ergin†  Onur Mutlu§

†TOBB University of Economics and Technology  §ETH Zürich
In-Flash Bulk Bitwise Execution

To appear at MICRO 2022

Flash-Cosmos: In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory

Jisung Park§ ▽ Roknoddin Azizi§ Geraldo F. Oliveira§ Mohammad Sadrosadati§
Rakesh Nadig§ David Novo† Juan Gómez-Luna§ Myungsuk Kim‡ Onur Mutlu§

§ETH Zürich ▽ POSTECH †LIRMM, Univ. Montpellier, CNRS ‡Kyungpook National University

In-DRAM Lookup-Table Based Execution

- To appear at MICRO 2022

**pLUTo: Enabling Massively Parallel Computation in DRAM via Lookup Tables**

João Dinis Ferreira$  Gabriel Falcao†  Juan Gómez-Luna$  Mohammed Alser$
Lois Orosa$  Mohammad Sadrosadati$  Jeremie S. Kim$  Geraldo F. Oliveira$
Taha Shahroodi‡  Anant Nori*  Onur Mutlu$

$ETH Zürich  †IT, University of Coimbra  ‡Galicia Supercomputing Center  †TU Delft  *Intel

Processing in Memory: Two Approaches

1. Processing using Memory
2. Processing near Memory
Mindset: Memory as an Accelerator

Memory similar to a “conventional” accelerator
Specialization: In-Memory Graph Processing

- Large graphs are everywhere (circa 2015)
  - 36 Million Wikipedia Pages
  - 1.4 Billion Facebook Users
  - 300 Million Twitter Users
  - 30 Billion Instagram Photos

- Scalable large-scale graph processing is challenging

![Graph Speedup Chart](chart.png)
Key Bottlenecks in Graph Processing

1. Frequent random memory accesses

2. Little amount of computation
Opportunity: 3D-Stacked Logic+Memory

Other “True 3D” technologies under development
Tesseract System for Graph Processing

Interconnected set of 3D-stacked memory+logic chips with simple cores

Host Processor

Memory-Mapped Accelerator Interface
(Noncacheable, Physically Addressed)

In-Order Core

Message Queue

PF Buffer

MTP

LP

DRAM Controller

Logic

Memory

SAFARI Ahn+, “A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing” ISCA 2015.
Tesseract System for Graph Processing

Host Processor
Memory-Mapped Accelerator Interface
(Noncacheable, Physically Addressed)

Memory

Logic

Crossbar Network

In-Order Core

Communications via Remote Function Calls

Message Queue

DRAM Controller

NI
Tesseract System for Graph Processing

Host Processor
Memory-Mapped Accelerator Interface
(Noncacheable, Physically Addressed)

Crossbar Network

Logic

Memory

Prefetching

LP
PF Buffer
MTP

Message Queue

DRAM Controller

NI

SAFARI
## Evaluated Systems

<table>
<thead>
<tr>
<th>System</th>
<th>Architecture</th>
<th>Clock Frequency</th>
<th>Bandwidth</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDR3-OoO</td>
<td>8 0oO 4GHz</td>
<td></td>
<td>102.4GB/s</td>
</tr>
<tr>
<td>HMC-OoO</td>
<td>8 0oO 4GHz</td>
<td></td>
<td>640GB/s</td>
</tr>
<tr>
<td>HMC-MC</td>
<td>128 In-Order 2GHz</td>
<td></td>
<td>640GB/s</td>
</tr>
<tr>
<td>Tesseract</td>
<td>32 Tesseract Cores</td>
<td></td>
<td>8TB/s</td>
</tr>
</tbody>
</table>

**SAFARI** Ahn+, “A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing” ISCA 2015.
Tesseract Graph Processing Performance

>13X Performance Improvement

On five graph processing algorithms

<table>
<thead>
<tr>
<th>Speedup</th>
<th>DDR3-OoO</th>
<th>HMC-OoO</th>
<th>HMC-MC</th>
<th>Tesseract</th>
<th>Tesseract-LP</th>
<th>Tesseract-LP-MTP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+56%</td>
<td>+25%</td>
<td></td>
<td>9.0x</td>
<td>11.6x</td>
<td>13.8x</td>
</tr>
</tbody>
</table>

SAFARI Ahn+, “A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing” ISCA 2015.
Tesseract Graph Processing System Energy

![Bar chart showing energy reduction comparison between HMC-OoO and Tesseract with Prefetching]

> 8X Energy Reduction

HMC-OoO

Tesseract with Prefetching

SAFARI Ahn+, “A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing” ISCA 2015.
More on Tesseract

- Junwhan Ahn, Sungpack Hong, Sungjoo Yoo, Onur Mutlu, and Kiyoung Choi,

"A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing"


[Slides (pdf)] [Lightning Session Slides (pdf)]
In-Storage Genomic Data Filtering [ASPLOS 2022]

- Nika Mansouri Ghiasi, Jisung Park, Harun Mustafa, Jeremie Kim, Ataberk Olgun, Arvid Gollwitzer, Damla Senol Cali, Can Firtina, Haiyu Mao, Nour Almadhoun Alserr, Rachata Ausavarungnirun, Nandita Vijaykumar, Mohammed Alser, and Onur Mutlu,

"GenStore: A High-Performance and Energy-Efficient In-Storage Computing System for Genome Sequence Analysis"
[Lightning Talk Slides (pptx) (pdf)]
[Lightning Talk Video (90 seconds)]

GenStore: A High-Performance In-Storage Processing System for Genome Sequence Analysis

Nika Mansouri Ghiasi\textsuperscript{1} Jisung Park\textsuperscript{1} Harun Mustafa\textsuperscript{1} Jeremie Kim\textsuperscript{1} Ataberk Olgun\textsuperscript{1} Arvid Gollwitzer\textsuperscript{1} Damla Senol Cali\textsuperscript{2} Can Firtina\textsuperscript{1} Haiyu Mao\textsuperscript{1} Nour Almadhoun Alserr\textsuperscript{1} Rachata Ausavarungnirun\textsuperscript{3} Nandita Vijaykumar\textsuperscript{4} Mohammed Alser\textsuperscript{1} Onur Mutlu\textsuperscript{1}

\textsuperscript{1}ETH Zürich \textsuperscript{2}Bionano Genomics \textsuperscript{3}KMUTNB \textsuperscript{4}University of Toronto

SAFARI
Genome Sequence Analysis

Data Movement from Storage

Storage System
Main Memory
Cache
Computation Unit (CPU or Accelerator)

Alignment

Computation overhead
Data movement overhead
Compute-Centric Accelerators

- Heuristics
- Accelerators
- Filters

Storage System

Main Memory

Cache

Computation Unit (CPU or Accelerator)

✓ Computation overhead

✗ Data movement overhead

SAFARI
Key Idea: In-Storage Filtering

Filter reads that do not require alignment inside the storage system

Exactly-matching reads
Do not need expensive approximate string matching during alignment

Non-matching reads
Do not have potential matching locations and can skip alignment
GenStore

Filter reads that do not require alignment inside the storage system

GenStore-Enabled Storage System

Main Memory

Cache

Computation Unit (CPU or Accelerator)

✓ Computation overhead

✓ Data movement overhead

GenStore provides significant speedup (1.4x - 33.6x) and energy reduction (3.9x – 29.2x) at low cost
In-Storage Genomic Data Filtering [ASPLOS 2022]

- Nika Mansouri Ghiasi, Jisung Park, Harun Mustafa, Jeremie Kim, Ataberk Olgun, Arvid Gollwitzer, Damla Senol Cali, Can Firtina, Haiyu Mao, Nour Almadhoun Alserr, Rachata Ausavarungrirun, Nandita Vijaykumar, Mohammed Alser, and Onur Mutlu,

"GenStore: A High-Performance and Energy-Efficient In-Storage Computing System for Genome Sequence Analysis"


[Lightning Talk Slides (pptx) (pdf)]
[Lightning Talk Video (90 seconds)]

GenStore: A High-Performance In-Storage Processing System for Genome Sequence Analysis

Nika Mansouri Ghiasi\textsuperscript{1} Jisung Park\textsuperscript{1} Harun Mustafa\textsuperscript{1} Jeremie Kim\textsuperscript{1} Ataberk Olgun\textsuperscript{1} Arvid Gollwitzer\textsuperscript{1} Damla Senol Cali\textsuperscript{2} Can Firtina\textsuperscript{1} Haiyu Mao\textsuperscript{1} Nour Almadhoun Alserr\textsuperscript{1} Rachata Ausavarungrirun\textsuperscript{3} Nandita Vijaykumar\textsuperscript{4} Mohammed Alser\textsuperscript{1} Onur Mutlu\textsuperscript{1}

\textsuperscript{1}ETH Zürich \textsuperscript{2}Bionano Genomics \textsuperscript{3}KMUTNB \textsuperscript{4}University of Toronto
Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks

Amirali Boroumand
Saugata Ghose, Youngsok Kim, Rachata Ausavarungnirun, Eric Shiu, Rahul Thakur, Daehyun Kim, Aki Kuusela, Allan Knies, Parthasarathy Ranganathan, Onur Mutlu

SAFARI

Carnegie Mellon

Google

SAMSUNG

ETH Zürich
Consumer Devices

Consumer devices are everywhere!

Energy consumption is a first-class concern in consumer devices
Popular Consumer Workloads

Chrome
Google’s web browser

TensorFlow Mobile
Google’s machine learning framework

VP9
Video Playback
Google’s video codec

VP9
Video Capture
Google’s video codec
Energy Cost of Data Movement

1st key observation: 62.7% of the total system energy is spent on data movement

Potential solution: move computation close to data

Challenge: limited area and energy budget
Using PIM to Reduce Data Movement

2nd key observation: a significant fraction of the data movement often comes from simple functions

We can design lightweight logic to implement these simple functions in memory

Offloading to PIM logic reduces energy and improves performance, on average, by 2.3X and 2.2X
Workload Analysis

Chrome
Google’s web browser

TensorFlow Mobile
Google’s machine learning framework

VP9
Video Playback
Google’s video codec

VP9
Video Capture
Google’s video codec
57.3% of the inference energy is spent on data movement

54.4% of the data movement energy comes from packing/unpacking and quantization
More on PIM for Mobile Devices

Amirali Boroumand, Saugata Ghose, Youngsok Kim, RachataAusavarungnirun, Eric Shiu, Rahul Thakur, Daehyun Kim, Aki Kuusela, Allan Knies, Parthasarathy Ranganathan, and Onur Mutlu,
"Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks"
[Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Poster (pptx) (pdf)] [Lightning Talk Video (2 minutes)] [Full Talk Video (21 minutes)]

Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks

Amirali Boroumand\textsuperscript{1} Rachata Ausavarungnirun\textsuperscript{1} Aki Kuusela\textsuperscript{3} Allan Knies\textsuperscript{3}
Saugata Ghose\textsuperscript{1} Eric Shiu\textsuperscript{3} Rahul Thakur\textsuperscript{3} Parthasarathy Ranganathan\textsuperscript{3}
Youngsok Kim\textsuperscript{2} Daehyun Kim\textsuperscript{4,3} Onur Mutlu\textsuperscript{5,1}

SAFARI
Truly Distributed GPU Processing with PIM

#pragma once

void applyScaleFactorsKernel( uint8_T * const out,
    uint8_T const * const in, const double *factor,
    size_t const numRows, size_t const numCols )
{
    // Work out which pixel we are working on.
    const int rowIdx = blockIdx.x * blockDim.x + threadIdx.x;
    const int colIdx = blockIdx.y;
    const int sliceIdx = threadIdx.z;

    // Check this thread isn't off the image
    if ( rowIdx >= numRows ) return;

    // Compute the index of my element
    size_t linearIdx = rowIdx + colIdx*numRows +
                        sliceIdx*numRows*numCols;

    // Compute the new position
    uint8_T new_value = in[linearIdx] * factor[0];
Accelerating GPU Execution with PIM (I)


[Slides (pptx) (pdf)]
[Lightning Session Slides (pptx) (pdf)]

---

Transparent Offloading and Mapping (TOM): Enabling Programmer-Transparent Near-Data Processing in GPU Systems

Kevin Hsieh†  Eiman Ebrahimi†  Gwangsun Kim*  Niladrish Chatterjee†  Mike O'Connor†
Nandita Vijaykumar‡  Onur Mutlu§†  Stephen W. Keckler†
‡Carnegie Mellon University  †NVIDIA  *KAIST  §ETH Zürich
Scheduling Techniques for GPU Architectures with Processing-In-Memory Capabilities

Ashutosh Pattnaik¹  Xulong Tang¹  Adwait Jog²  Onur Kayıran³
Asit K. Mishra⁴  Mahmut T. Kandemir¹  Onur Mutlu⁵,⁶  Chita R. Das¹

¹Pennsylvania State University  ²College of William and Mary
³Advanced Micro Devices, Inc.  ⁴Intel Labs  ⁵ETH Zürich  ⁶Carnegie Mellon University
Accelerating Linked Data Structures

Kevin Hsieh, Samira Khan, Nandita Vijaykumar, Kevin K. Chang, Amirali Boroumand, Saugata Ghose, and Onur Mutlu,
"Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation"
Proceedings of the 34th IEEE International Conference on Computer Design (ICCD), Phoenix, AZ, USA, October 2016.

Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation

Kevin Hsieh†  Samira Khan‡  Nandita Vijaykumar†
Kevin K. Chang†  Amirali Boroumand†  Saugata Ghose†  Onur Mutlu§†
†Carnegie Mellon University  ‡University of Virginia  §ETH Zürich
Accelerating Dependent Cache Misses

- Milad Hashemi, Khubaib, Eiman Ebrahimi, Onur Mutlu, and Yale N. Patt, "Accelerating Dependent Cache Misses with an Enhanced Memory Controller"

[Slides (pptx) (pdf)]
[Lightning Session Slides (pptx) (pdf)]

Accelerating Dependent Cache Misses with an Enhanced Memory Controller

Milad Hashemi*, Khubaib†, Eiman Ebrahimi‡, Onur Mutlu§, Yale N. Patt*

*The University of Texas at Austin †Apple ‡NVIDIA §ETH Zürich & Carnegie Mellon University
Accelerating Runahead Execution

- Milad Hashemi, Onur Mutlu, and Yale N. Patt, "Continuous Runahead: Transparent Hardware Acceleration for Memory Intensive Workloads" Proceedings of the 49th International Symposium on Microarchitecture (MICRO), Taipei, Taiwan, October 2016. [Slides (pptx) (pdf)] [Lightning Session Slides (pdf)] [Poster (pptx) (pdf)]

Continuous Runahead: Transparent Hardware Acceleration for Memory Intensive Workloads

Milad Hashemi*, Onur Mutlu§, Yale N. Patt*

*The University of Texas at Austin  §ETH Zürich
Accelerating Climate Modeling

Gagandeep Singh, Dionysios Diamantopoulos, Christoph Hagleitner, Juan Gómez-Luna, Sander Stuijk, Onur Mutlu, and Henk Corporaal,

"NERO: A Near High-Bandwidth Memory Stencil Accelerator for Weather Prediction Modeling"

*Proceedings of the 30th International Conference on Field-Programmable Logic and Applications (FPL), Gothenburg, Sweden, September 2020.*

[Slides (pptx) (pdf)]

[Lightning Talk Slides (pptx) (pdf)]

[Talk Video (23 minutes)]

*Nominated for the Stamatis Vassiliadis Memorial Award.*
Accelerating Approximate String Matching

- Damla Senol Cali, Gurpreet S. Kalsi, Zulal Bingol, Can Firtina, Lavanya Subramanian, Jeremie S. Kim, Rachata Ausavarungnirun, Mohammed Alser, Juan Gomez-Luna, Amirali Boroumand, Anant Nori, Allison Scibisz, Sreenivas Subramoney, Can Alkan, Saugata Ghose, and Onur Mutlu,

"GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis"


[Lighting Talk Video (1.5 minutes)]
[Lightning Talk Slides (pptx) (pdf)]
[Talk Video (18 minutes)]
[Slides (pptx) (pdf)]

GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis

Damla Senol Cali†, Gurpreet S. Kalsi, Zülal Bingöl, Can Firtina, Lavanya Subramanian†, Jeremie S. Kim†, Rachata Ausavarungnirun, Mohammed Alser, Juan Gomez-Luna, Amirali Boroumand, Anant Nori, Allison Scibisz, Sreenivas Subramoney, Can Alkan, Saugata Ghose†, Onur Mutlu†

†Carnegie Mellon University, ‡Processor Architecture Research Lab, Intel Labs, ††Bilkent University, ‡‡ETH Zürich, †Facebook, †King Mongkut’s University of Technology North Bangkok, ⋆University of Illinois at Urbana-Champaign

SAFARI
Accelerating Sequence-to-Graph Mapping

- Damla Senol Cali, Konstantinos Kanellopoulos, Joel Lindegger, Zulal Bingol, Gurpreet S. Kalsi, Ziyi Zuo, Can Firtina, Meryem Banu Cavlak, Jeremie Kim, Nika MansouriGhiasi, Gagandeep Singh, Juan Gomez-Luna, Nour Almadhoun Alserr, Mohammed Alser, Sreenivas Subramoney, Can Alkan, Saugata Ghose, and Onur Mutlu,

"SeGraM: A Universal Hardware Accelerator for Genomic Sequence-to-Graph and Sequence-to-Sequence Mapping"


[arXiv version]

**SeGraM: A Universal Hardware Accelerator for Genomic Sequence-to-Graph and Sequence-to-Sequence Mapping**

Damla Senol Cali\(^1\) Konstantinos Kanellopoulos\(^2\) Joël Lindegger\(^2\) Zülal Bingöl\(^3\) Gurpreet S. Kalsi\(^4\) Ziyi Zuo\(^5\) Can Firtina\(^2\) Meryem Banu Cavlak\(^2\) Jeremie Kim\(^2\) Nika Mansouri Ghiasi\(^2\) Gagandeep Singh\(^2\) Juan Gómez-Luna\(^2\) Nour Almadhoun Alserr\(^2\) Mohammed Alser\(^2\) Sreenivas Subramoney\(^4\) Can Alkan\(^3\) Saugata Ghose\(^6\) Onur Mutlu\(^2\)

\(^1\)Bionano Genomics \(^2\)ETH Zürich \(^3\)Bilkent University \(^4\)Intel Labs
\(^5\)Carnegie Mellon University \(^6\)University of Illinois Urbana-Champaign

Accelerating Basecalling + Read Mapping

- To appear at MICRO 2022

GenPIP: In-Memory Acceleration of Genome Analysis via Tight Integration of Basecalling and Read Mapping

Haiyu Mao¹ Mohammed Alser¹ Mohammad Sadrosadati¹ Can Firtina¹ Akanksha Baranwal¹ Damla Senol Cali² Aditya Manglik¹ Nour Almadhoun Alserī¹ Onur Mutlu¹

¹ETH Zürich ²Bionano Genomics
Accelerating Time Series Analysis

- Ivan Fernandez, Ricardo Quislant, Christina Giannoula, Mohammed Alser, Juan Gómez-Luna, Eladio Gutiérrez, Oscar Plata, and Onur Mutlu,

"NATSA: A Near-Data Processing Accelerator for Time Series Analysis"


[Slides (pptx) (pdf)]
[Talk Video (10 minutes)]
[Source Code]

NATSA: A Near-Data Processing Accelerator for Time Series Analysis

Ivan Fernandez§ Ricardo Quislant§ Christina Giannoula† Mohammed Alser‡
Juan Gómez-Luna‡ Eladio Gutiérrez§ Oscar Plata§ Onur Mutlu‡

§University of Malaga †National Technical University of Athens ‡ETH Zürich
Accelerating Graph Pattern Mining

- Maciej Besta, Raghavendra Kanakagiri, Grzegorz Kwasniewski, Rachata Ausavarunngirun, Jakub Beránek, Konstantinos Kanellopoulos, Kacper Janda, Zur Vonarburg-Shmaria, Lukas Gianinazzi, Ioana Stefan, Juan Gómez-Luna, Marcin Copik, Lukas Kapp-Schwoerer, Salvatore Di Girolamo, Nils Blach, Marek Konieczny, Onur Mutlu, and Torsten Hoefler,

"SISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems"

Proceedings of the 54th International Symposium on Microarchitecture (MICRO), Virtual, October 2021.
[Slides (pdf)]
[Talk Video (22 minutes)]
[Lightning Talk Video (1.5 minutes)]
[Full arXiv version]

SISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems

Maciej Besta\textsuperscript{1}, Raghavendra Kanakagiri\textsuperscript{2}, Grzegorz Kwasniewski\textsuperscript{1}, Rachata Ausavarunngirun\textsuperscript{3}, Jakub Beránek\textsuperscript{4}, Konstantinos Kanellopoulos\textsuperscript{1}, Kacper Janda\textsuperscript{5}, Zur Vonarburg-Shmaria\textsuperscript{1}, Lukas Gianinazzi\textsuperscript{1}, Ioana Stefan\textsuperscript{1}, Juan Gómez-Luna\textsuperscript{1}, Marcin Copik\textsuperscript{1}, Lukas Kapp-Schwoerer\textsuperscript{1}, Salvatore Di Girolamo\textsuperscript{1}, Nils Blach\textsuperscript{1}, Marek Konieczny\textsuperscript{5}, Onur Mutlu\textsuperscript{1}, Torsten Hoefler\textsuperscript{1}

\textsuperscript{1}ETH Zurich, Switzerland \hspace{1cm} \textsuperscript{2}IIT Tirupati, India \hspace{1cm} \textsuperscript{3}King Mongkut’s University of Technology North Bangkok, Thailand \hspace{1cm} \textsuperscript{4}Technical University of Ostrava, Czech Republic \hspace{1cm} \textsuperscript{5}AGH-UST, Poland
Accelerating HTAP Database Systems

- Amirali Boroumand, Saugata Ghose, Geraldo F. Oliveira, and Onur Mutlu,
  "Polynesia: Enabling High-Performance and Energy-Efficient Hybrid
  Transactional/Analytical Databases with Hardware/Software Co-Design"
  Proceedings of the 38th International Conference on Data Engineering (ICDE),
  Virtual, May 2022.
  [arXiv version]
  [Slides (pptx) (pdf)]
  [Short Talk Slides (pptx) (pdf)]

Polynesia: Enabling High-Performance and Energy-Efficient Hybrid Transactional/Analytical Databases with Hardware/Software Co-Design

Amirali Boroumand†
†Google

Saugata Ghose○
○Univ. of Illinois Urbana-Champaign

Geraldo F. Oliveira‡

Onur Mutlu‡
‡ETH Zürich

Accelerating Neural Network Inference

- Amirali Boroumand, Saugata Ghose, Berkin Akin, Ravi Narayanaswami, Geraldo F. Oliveira, Xiaoyu Ma, Eric Shiu, and Onur Mutlu,

"Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks"

Proceedings of the 30th International Conference on Parallel Architectures and Compilation Techniques (PACT), Virtual, September 2021.

[Slides (pptx) (pdf)]
[Talk Video (14 minutes)]

Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks

Amirali Boroumand\textsuperscript{†○}
Saugata Ghose\textsuperscript{‡}
Berkin Akin\textsuperscript{§}
Ravi Narayanaswami\textsuperscript{§}

Geraldo F. Oliveira\textsuperscript{*}
Xiaoyu Ma\textsuperscript{§}
Eric Shiu\textsuperscript{§}
Onur Mutlu\textsuperscript{*†}

\textsuperscript{†}Carnegie Mellon Univ. \quad \textsuperscript{○}Stanford Univ. \quad \textsuperscript{‡}Univ. of Illinois Urbana-Champaign \quad \textsuperscript{§}Google \quad \textsuperscript{*}ETH Zürich
Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks

Amirali Boroumand  Saugata Ghose  Berkin Akin
Ravi Narayanaswami  Geraldo F. Oliveira  Xiaoyu Ma
Eric Shiu  Onur Mutlu

PACT 2021

SAFARI

Carnegie Mellon  University of Illinois Urbana-Champaign  Google  ETH Zürich
Executive Summary

Context: We extensively analyze a state-of-the-art edge ML accelerator (Google Edge TPU) using 24 Google edge models
- Wide range of models (CNNs, LSTMs, Transducers, RCNNs)

Problem: The Edge TPU accelerator suffers from three challenges:
- It operates significantly below its peak throughput
- It operates significantly below its theoretical energy efficiency
- It inefficiently handles memory accesses

Key Insight: These shortcomings arise from the monolithic design of the Edge TPU accelerator
- The Edge TPU accelerator design does not account for layer heterogeneity

Key Mechanism: A new framework called Mensa
- Mensa consists of heterogeneous accelerators whose dataflow and hardware are specialized for specific families of layers

Key Results: We design a version of Mensa for Google edge ML models
- Mensa improves performance and energy by 3.0X and 3.1X
- Mensa reduces cost and improves area efficiency
We analyze inference execution using 24 edge NN models.
Diversity Across the Models

**Insight 1:** There is significant variation in terms of layer characteristics across the models.

---

![Graph showing diversity across models](image-url)

The graph illustrates the diversity in terms of FLOP/Byte and Parameter Footprint (MB) across various models. The models are categorized into those derived from CNNs and RCNNs and those from LSTMs and Transducers.
Diversity Within the Models

Insight 2: even within each model, layers exhibit significant variation in terms of layer characteristics.

For example, our analysis of edge CNN models shows:

Variation in MAC intensity: up to 200x across layers

Variation in FLOP/Byte: up to 244x across layers
Mensa High-Level Overview

**Edge TPU Accelerator**

Model A  Model B  Model C

![Diagram showing the Edge TPU Accelerator with models A, B, and C, each represented by a neural network structure.]

**Mensa**

Model A  Model B  Model C

![Diagram showing the Mensa architecture with models A, B, and C, each connected to a runtime.]

**Runtime**

Family 1  Family 2  Family 3

![Diagram showing the Mensa architecture with families 1, 2, and 3, each connecting to a CPU and 3D-stacked DRAM.]

**Heterogeneous Accelerators**

Acc. 1  Acc. 2  Acc. 3

![Diagram showing the heterogeneous accelerators acc. 1, 2, and 3, each with a NoC and Buffer.]
Identifying Layer Families

Key observation: the majority of layers group into a small number of layer families

- Families 1 & 2: low parameter footprint, high data reuse and MAC intensity → compute-centric layers
- Families 3, 4 & 5: high parameter footprint, low data reuse and MAC intensity → data-centric layers
Mensa: Energy Reduction

Mensa-G reduces energy consumption by 3.0X compared to the baseline Edge TPU

SAFARI
Mensa-G improves inference throughput by 3.1X compared to the baseline Edge TPU
Mensa: Highly-Efficient ML Inference

- Amirali Boroumand, Saugata Ghose, Berkin Akin, Ravi Narayanaswami, Geraldo F. Oliveira, Xiaoyu Ma, Eric Shiu, and Onur Mutlu,

"Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks"

Proceedings of the 30th International Conference on Parallel Architectures and Compilation Techniques (PACT), Virtual, September 2021.

[Slides (pptx) (pdf)]
[Talk Video (14 minutes)]

Google Neural Network Models for Edge Devices:
Analyzing and Mitigating Machine Learning Inference Bottlenecks

Amirali Boroumand†‡
Geraldo F. Oliveira*
Saugata Ghose‡
Xiaoyu Ma§
Berkin Akin§
Eric Shiu§
Ravi Narayanaswami§
Onur Mutlu*†

†Carnegie Mellon Univ.  °Stanford Univ.  ‡Univ. of Illinois Urbana-Champaign  §Google  *ETH Zürich
FPGA-based Processing Near Memory


FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications

Gagandeep Singh◊  Mohammed Alser◊  Damla Senol Cali¶
Dionysios Diamantopoulos∇  Juan Gómez-Luna◊
Henk Corporaal*  Onur Mutlu◊¶

◊ETH Zürich  ¶Carnegie Mellon University
*Eindhoven University of Technology  ∇IBM Research Europe
Near-Memory Acceleration Using FPGAs

IBM POWER9 CPU

HBM-based FPGA board

Near-HBM FPGA-based accelerator

Two communication technologies: CAPI2 and OCAPI
Two memory technologies: DDR4 and HBM
Two workloads: Weather Modeling and Genome Analysis
Performance & Energy Greatly Improve

5-27× performance vs. a 16-core (64-thread) IBM POWER9 CPU

12-133× energy efficiency vs. a 16-core (64-thread) IBM POWER9 CPU

HBM alleviates memory bandwidth contention vs. DDR4
We Need to Revisit the Entire Stack

We can get there step by step
A Modern Primer on Processing in Memory

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

SAFARI Research Group

\textsuperscript{a}ETH Zürich
\textsuperscript{b}Carnegie Mellon University
\textsuperscript{c}University of Illinois at Urbana-Champaign
\textsuperscript{d}King Mongkut’s University of Technology North Bangkok

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, and Rachata Ausavarungnirun, "A Modern Primer on Processing in Memory"

\textbf{SAFARI}

A Workload and Programming Ease Driven Perspective of Processing-in-Memory

Saugata Ghose†  Amirali Boroumand†  Jeremie S. Kim†§  Juan Gómez-Luna§  Onur Mutlu§†

†Carnegie Mellon University  §ETH Zürich

Saugata Ghose, Amirali Boroumand, Jeremie S. Kim, Juan Gomez-Luna, and Onur Mutlu, "Processing-in-Memory: A Workload-Driven Perspective"
[Preliminary arXiv version]

Processing in Memory: Adoption Challenges

1. Processing using Memory
2. Processing near Memory
Eliminating the Adoption Barriers

How to Enable Adoption of Processing in Memory
Potential Barriers to Adoption of PIM

1. **Applications & software** for PIM

2. Ease of **programming** (interfaces and compiler/HW support)

3. **System** and **security** support: coherence, synchronization, virtual memory, isolation, communication interfaces, ...

4. **Runtime** and **compilation** systems for adaptive scheduling, data mapping, access/sharing control, ...

5. **Infrastructures** to assess benefits and feasibility

All can be solved with change of mindset
We Need to Revisit the Entire Stack

We can get there step by step
Adoption: Accelerating Key Applications (I)

- Mohammed Alser, Zulal Bingol, Damla Senol Cali, Jeremie Kim, Saugata Ghose, Can Alkan, and Onur Mutlu,

"Accelerating Genome Analysis: A Primer on an Ongoing Journey"

[Slides (pptx)(pdf)]
[Talk Video (1 hour 2 minutes)]

Accelerating Genome Analysis: A Primer on an Ongoing Journey

- **Mohammed Alser**
  ETH Zürich

- **Zülal Bingöl**
  Bilkent University

- **Damla Senol Cali**
  Carnegie Mellon University

- **Jeremie Kim**
  ETH Zurich and Carnegie Mellon University

- **Saugata Ghose**
  University of Illinois at Urbana–Champaign and Carnegie Mellon University

- **Can Alkan**
  Bilkent University

- **Onur Mutlu**
  ETH Zurich, Carnegie Mellon University, and Bilkent University
Adoption: Accelerating Key Applications (II)

- Nika Mansouri Ghiasi, Jisung Park, Harun Mustafa, Jeremie Kim, Ataberk Olgun, Arvid Gollwitzer, Damla Senol Cali, Can Firtina, Haiyu Mao, Nour Almadhoun Alserr, Rachata Ausavarungnirun, Nandita Vijaykumar, Mohammed Alser, and Onur Mutlu,

"GenStore: A High-Performance and Energy-Efficient In-Storage Computing System for Genome Sequence Analysis"


[Lightning Talk Slides (pptx) (pdf)]
[Lightning Talk Video (90 seconds)]

GenStore: A High-Performance In-Storage Processing System for Genome Sequence Analysis

Nika Mansouri Ghiasi¹  Jisung Park¹  Harun Mustafa¹  Jeremie Kim¹  Ataberk Olgun¹  Arvid Gollwitzer¹  Damla Senol Cali²  Can Firtina¹  Haiyu Mao¹  Nour Almadhoun Alserr¹  Rachata Ausavarungnirun³  Nandita Vijaykumar⁴  Mohammed Alser¹  Onur Mutlu¹

¹ETH Zürich  ²Bionano Genomics  ³KMUTNB  ⁴University of Toronto
A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing

Junwhan Ahn  Sungpack Hong§  Sungjoo Yoo  Onur Mutlu†  Kiyoungh Choi
junwhan@snu.ac.kr, sungpack.hong@oracle.com, sungjoo.yoo@gmail.com, onur@cmu.edu, kchoi@snu.ac.kr
Seoul National University  §Oracle Labs  †Carnegie Mellon University
Adoption: Accelerating Key Applications (IV)

Maciej Besta, Raghavendra Kanakagiri, Grzegorz Kwasniewski, Rachata Ausavarungnirun, Jakub Beránek, Konstantinos Kanellopoulos, Kacper Janda, Zur Vonarburg-Shmaria, Lukas Gianinazzi, Ioana Stefan, Juan Gómez-Luna, Marcin Copik, Lukas Kapp-Schwoerer, Salvatore Di Girolamo, Nils Blach, Marek Konieczny, Onur Mutlu, and Torsten Hoefler,
"SISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems"
Proceedings of the 54th International Symposium on Microarchitecture (MICRO), Virtual, October 2021.
[Slides (pdf)]
[Talk Video (22 minutes)]
[Lightning Talk Video (1.5 minutes)]
[Full arXiv version]

SISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems

Maciej Besta¹, Raghavendra Kanakagiri², Grzegorz Kwasniewski¹, Rachata Ausavarungnirun³, Jakub Beránek⁴, Konstantinos Kanellopoulos¹, Kacper Janda⁵, Zur Vonarburg-Shmaria¹, Lukas Gianinazzi¹, Ioana Stefan¹, Juan Gómez-Luna¹, Marcin Copik¹, Lukas Kapp-Schwoerer¹, Salvatore Di Girolamo¹, Nils Blach¹, Marek Konieczny⁵, Onur Mutlu¹, Torsten Hoefler¹

¹ETH Zurich, Switzerland  ²IIT Tirupati, India  ³King Mongkut’s University of Technology North Bangkok, Thailand  ⁴Technical University of Ostrava, Czech Republic  ⁵AGH-UST, Poland
Adoption: Accelerating Key Applications (V)

- Amirali Boroumand, Saugata Ghose, Berkin Akin, Ravi Narayanaswami, Geraldo F. Oliveira, Xiaoyu Ma, Eric Shiu, and Onur Mutlu,

"Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks"

Proceedings of the 30th International Conference on Parallel Architectures and Compilation Techniques (PACT), Virtual, September 2021.

[Slides (pptx) (pdf)]
[Talk Video (14 minutes)]

Google Neural Network Models for Edge Devices: Analyzing and Mitigating Machine Learning Inference Bottlenecks

Amirali Boroumand†◊, Saugata Ghose‡, Berkin Akin§, Ravi Narayanaswami§, Geraldo F. Oliveira*, Xiaoyu Ma§, Eric Shiu§, Onur Mutlu*†

†Carnegie Mellon Univ. ◊Stanford Univ. ‡Univ. of Illinois Urbana-Champaign §Google *ETH Zürich
Gagandeep Singh, Dionysios Diamantopoulos, Christoph Hagleitner, Juan Gómez-Luna, Sander Stuijk, Onur Mutlu, and Henk Corporaal,
"NERO: A Near High-Bandwidth Memory Stencil Accelerator for Weather Prediction Modeling"
Proceedings of the 30th International Conference on Field-Programmable Logic and Applications (FPL), Gothenburg, Sweden, September 2020.
[Slides (pptx) (pdf)]
[Lightning Talk Slides (pptx) (pdf)]
[Talk Video (23 minutes)]
Nominated for the Stamatis Vassiliadis Memorial Award.

NERO: A Near High-Bandwidth Memory Stencil Accelerator for Weather Prediction Modeling

Gagandeep Singh$^{a,b,c}$ Dionysios Diamantopoulos$^c$ Christoph Hagleitner$^c$ Juan Gómez-Luna$^b$
Sander Stuijk$^a$ Onur Mutlu$^b$ Henk Corporaal$^a$
$^a$Eindhoven University of Technology  $^b$ETH Zürich  $^c$IBM Research Europe, Zurich

[Slides (pptx) (pdf)]
[Talk Video (10 minutes)]
[Source Code]

NATSA: A Near-Data Processing Accelerator for Time Series Analysis

Ivan Fernandez§ Ricardo Quislant§ Christina Giannoula† Mohammed Alser‡
Juan Gómez-Luna‡ Eladio Gutiérrez§ Oscar Plata§ Onur Mutlu‡

§University of Malaga †National Technical University of Athens ‡ETH Zürich
Adoption: Accelerating Key Applications (VIII)

- Damla Senol Cali, Gurpreet S. Kalsi, Zulal Bingol, Can Firtina, Lavanya Subramanian, Jeremie S. Kim, Rachata Ausavarungnirun, Mohammed Alser, Juan Gomez-Luna, Amirali Boroumand, Anant Nori, Allison Scibisz, Sreenivas Subramoney, Can Alkan, Saugata Ghose, and Onur Mutlu,

"GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis"


[Lighting Talk Video (1.5 minutes)]
[Lightning Talk Slides (pptx) (pdf)]
[Talk Video (18 minutes)]
[Slides (pptx) (pdf)]

GenASM: A High-Performance, Low-Power Approximate String Matching Acceleration Framework for Genome Sequence Analysis

Damla Senol Cali†‡, Gurpreet S. Kalsi‡, Zülal Bingölˇ, Can Firtina‡, Lavanya Subramanian†, Jeremie S. Kim‡†, Rachata Ausavarungnirun©, Mohammed Alser‡, Juan Gomez-Luna‡, Amirali Boroumand†, Anant Nori‡, Allison Scibisz†, Sreenivas Subramoney‡, Can Alkan‡, Saugata Ghose††, Onur Mutlu‡††

† Carnegie Mellon University ‡ Processor Architecture Research Lab, Intel Labs ◊ Bilkent University ◎ ETH Zürich
© Facebook ◊ King Mongkut’s University of Technology North Bangkok * University of Illinois at Urbana–Champaign

SAFARI
SeGraM: A Universal Hardware Accelerator for Genomic Sequence-to-Graph and Sequence-to-Sequence Mapping

Damla Senol Cali\(^1\) Konstantinos Kanellopoulos\(^2\) Joël Lindegger\(^2\) Zülal Bingöl\(^3\) Gurpreet S. Kalsi\(^4\) Ziyi Zuo\(^5\) Can Firtina\(^2\) Meryem Banu Cavlak\(^2\) Jeremie Kim\(^2\) Nika Mansouri Ghiasi\(^2\) Gagandeep Singh\(^2\) Juan Gómez-Luna\(^2\) Nour Almadhoun Alserr\(^2\) Mohammed Alser\(^2\) Sreenivas Subramoney\(^4\) Can Alkan\(^3\) Saugata Ghose\(^6\) Onur Mutlu\(^2\)

\(^1\)Bionano Genomics \(^2\)ETH Zürich \(^3\)Bilkent University \(^4\)Intel Labs \\(^5\)Carnegie Mellon University \(^6\)University of Illinois Urbana-Champaign

Adoption: Accelerating Key Applications (X)

  [arXiv version]
  [Slides (pptx) (pdf)]
  [Short Talk Slides (pptx) (pdf)]

Polynesia: Enabling High-Performance and Energy-Efficient Hybrid Transactional/Analytical Databases with Hardware/Software Co-Design

Amirali Boroumand†
†Google
Saugata Ghose◊
◊Univ. of Illinois Urbana-Champaign
Geraldo F. Oliveira‡
‡ETH Zürich
Onur Mutlu‡
Adoption: Accelerating Key Applications (XI)

- To appear at MICRO 2022

GenPIP: In-Memory Acceleration of Genome Analysis via Tight Integration of Basecalling and Read Mapping

Haiyu Mao¹ Mohammed Alser¹ Mohammad Sadrosadati¹ Can Firtina¹ Akanksha Baranwal¹ Damla Senol Cali² Aditya Manglik¹ Nour Almadhoun Alser⁰ Onur Mutlu¹

¹ETH Zürich ²Bionano Genomics

Adoption: How to Keep It Simple?


PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture

Junwhan Ahn  Sungjoo Yoo  Onur Mutlu†  Kiyoungh Choi
junwhan@snu.ac.kr, sungjoo.yoo@gmail.com, onur@cmu.edu, kchoi@snu.ac.kr

Seoul National University  †Carnegie Mellon University
Challenge: Coherence for Hybrid CPU-PIM Apps

![Graph showing speedup for different applications with traditional coherence, no coherence, and different PIM architectures. The graph compares CPU-only, FG, CG, NC, LazyPIM, and Ideal-PIM configurations.](image)

- Traditional coherence
- No coherence overhead

<table>
<thead>
<tr>
<th>Application</th>
<th>Components</th>
<th>Radii</th>
<th>PageRank</th>
<th>Components</th>
<th>Radii</th>
<th>PageRank</th>
<th>Radii</th>
<th>PageRank</th>
<th>HTAP-256</th>
<th>HTAP-128</th>
<th>GMean</th>
</tr>
</thead>
<tbody>
<tr>
<td>arXiv</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gnutella</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enron</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMDB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SAFARI
Adoption: How to Maintain Coherence? (II)

- Amirali Boroumand, Saugata Ghose, Minesh Patel, Hasan Hassan, Brandon Lucia, Kevin Hsieh, Krishna T. Malladi, Hongzhong Zheng, and Onur Mutlu,
"CoNDA: Efficient Cache Coherence Support for Near-Data Accelerators"

CoNDA: Efficient Cache Coherence Support for Near-Data Accelerators

Amirali Boroumand†
Brandon Lucia†
Saugata Ghose†
Rachata Ausavarungnirun‡‡
Hasan Hassan*
Nastaran Hajinazar†‡
Rachana Mansingh‡
Krishna T. Malladi§
Kevin Hsieh†
Hongzhong Zheng§
Onur Mutlu*†

†Carnegie Mellon University
‡‡Korea University
*ETH Zürich
§Samsung Semiconductor, Inc.
Adoption: How to Support Synchronization?

  [Slides (pptx) (pdf)]
  [Short Talk Slides (pptx) (pdf)]
  [Talk Video (21 minutes)]
  [Short Talk Video (7 minutes)]

**SynCron: Efficient Synchronization Support for Near-Data-Processing Architectures**

Christina Giannoula†‡ Nandita Vijaykumar*‡ Nikela Papadopoulou† Vasileios Karakostas† Ivan Fernandez§‡
Juan Gómez-Luna‡ Lois Orosa‡ Nectarios Koziris† Georgios Goumas† Onur Mutlu‡
†National Technical University of Athens  ‡ETH Zürich  *University of Toronto  §University of Malaga
Adoption: How to Support Virtual Memory?

Kevin Hsieh, Samira Khan, Nandita Vijaykumar, Kevin K. Chang, Amirali Boroumand, Saugata Ghose, and Onur Mutlu,
"Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation"
Proceedings of the 34th IEEE International Conference on Computer Design (ICCD), Phoenix, AZ, USA, October 2016.

Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation

Kevin Hsieh† Samira Khan‡ Nandita Vijaykumar†
Kevin K. Chang† Amirali Boroumand† Saugata Ghose† Onur Mutlu§†
†Carnegie Mellon University ‡University of Virginia §ETH Zürich
Eliminating the Adoption Barriers

Processing-in-Memory in the Real World
Processing-in-Memory Landscape Today

This does not include many experimental chips and startups
UPMEM Processing-in-DRAM Engine (2019)

- Processing in DRAM Engine
- Includes **standard DIMM modules**, with a **large number of DPU processors** combined with DRAM chips.

- Replaces **standard** DIMMs
  - DDR4 R-DIMM modules
    - 8GB+128 DPUs (16 PIM chips)
    - Standard 2x-nm DRAM process
  - **Large amounts of** compute & memory bandwidth

[Upmem webpage](https://www.anandtech.com/show/14750/hot-chips-31-analysis-inmemory-processing-by-upmem)
UPMEM Memory Modules

• E19: 8 chips DIMM (1 rank). DPUs @ 267 MHz
• P21: 16 chips DIMM (2 ranks). DPUs @ 350 MHz
2,560-DPU Processing-in-Memory System

Benchmarking a New Paradigm: An Experimental Analysis of a Real Processing-in-Memory Architecture

JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
IZZAT EL HAJJ, American University of Beirut, Lebanon
IVAN FERNANDEZ, ETH Zürich, Switzerland and University of Malaga, Spain
CHRISTINA GIANNOPULA, ETH Zürich, Switzerland and NTUA, Greece
GERALDO F. OLIVEIRA, ETH Zürich, Switzerland
ONUR MUTLU, ETH Zürich, Switzerland

Many modern workloads, such as neural networks, databases, and graph processing, are fundamentally memory-bound. For such workloads, the data movement between main memory and CPU cores imposes a significant overhead in terms of both latency and energy. A major reason is that this communication happens through a narrow bus with high latency and limited bandwidth, and the low data reuse in memory-bound workloads is insufficient to amortize the cost of main memory access. Fundamentally unbinding this data movement bottleneck requires a paradigm whereby the memory system assumes an active role in computing by integrating processing capabilities. This paradigm is known as processing-in-memory (PIM).

Recent research explores different forms of PIM architecture, motivated by the emergence of new 3D-stacked memory technologies that integrate memory with a logic layer where processing elements can be easily placed. Past works evaluate these architectures in simulation or at best, with simplified hardware prototypes. In contrast, the UPMEM company has designed and manufactured the first publicly available real-world PIM architecture. The UPMEM PIM architecture combines traditional DRAM memory arrays with general-purpose in-order cores, called SIM Instructions (SIMPs), integrated in the same chip.

This paper provides the first comprehensive analysis of the first publicly available real-world PIM architecture. We make two key contributions. First, we conduct an experimental characterization of the UPMEM-based PIM system using microbenchmarks to assess various architecture limits such as compute throughput and memory bandwidth, yielding new insights. Second, we present PIM benchmarking (PIMbench), a benchmark suite of 16 workloads from different application domains (e.g., dense/sparse linear algebra, databases, data analytics, graph processing, neural networks, bioinformatics, image processing), which we identify as memory-bound. We evaluate the performance and scaling characteristics of PIM benchmarks on the UPMEM PIM architecture, and compare their performance and energy consumption to their state-of-the-art CPU and GPU counterparts. Our extensive evaluation conducted on two real UPMEM-based PIM systems with 140 and 2,560 DPUs provides new insights about suitability of different workloads to the PIM system, programming recommendations for software designers, and suggestions and hints for hardware and architecture designers of future PIM systems.

More on the UPMEM PIM System

DRAM Processing Unit (II)

PIM Chip

Control/Status Interface ↔ DDR4 Interface

24-KB Instruction Memory

64-KB WRAM

64-MB DRAM Bank

DMA Engine

ETH ZÜRICH HAUSTEIL
Computer Architecture - Lecture 12d: Real Processing-in-DRAM with UPMEM (ETH Zürich, Fall 2020)
1,120 views • Oct 31, 2020

Onur Mutlu Lectures
16.7K subscribers

https://www.youtube.com/watch?v=Sscy1Wrr22A&list=PL5Q2soXY2ZI9xidy1gBxUz7xRPS-wisBN&index=26
Experimental Analysis of the UPMEM PIM Engine

Benchmarking a New Paradigm: An Experimental Analysis of a Real Processing-in-Memory Architecture

JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
IZZAT EL HAJJ, American University of Beirut, Lebanon
IVAN FERNANDEZ, ETH Zürich, Switzerland and University of Malaga, Spain
CHRISTINA GIANNOULA, ETH Zürich, Switzerland and NTUA, Greece
GERALDO F. OLIVEIRA, ETH Zürich, Switzerland
ONUR MUTLU, ETH Zürich, Switzerland

Many modern workloads, such as neural networks, databases, and graph processing, are fundamentally memory-bound. For such workloads, the data movement between main memory and CPU cores imposes a significant overhead in terms of both latency and energy. A major reason is that this communication happens through a narrow bus with high latency and limited bandwidth, and the low data reuse in memory-bound workloads is insufficient to amortize the cost of main memory access. Fundamentally addressing this data movement bottleneck requires a paradigm where the memory system assumes an active role in computing by integrating processing capabilities. This paradigm is known as processing-in-memory (PIM).

Recent research explores different forms of PIM architectures, motivated by the emergence of new 3D-stacked memory technologies that integrate memory with a logic layer where processing elements can be easily placed. Past works evaluate these architectures in simulation or, at best, with simplified hardware prototypes. In contrast, the UPMEM company has designed and manufactured the first publicly-available real-world PIM architecture. The UPMEM PIM architecture combines traditional DRAM memory arrays with general-purpose in-order cores, called DRAM Processing Units (DPUs), integrated in the same chip.

This paper provides the first comprehensive analysis of the first publicly-available real-world PIM architecture. We make two key contributions. First, we conduct an experimental characterization of the UPMEM-based PIM system using microbenchmarks to assess various architecture limits such as compute throughput and memory bandwidth, yielding new insights. Second, we present PrIM (Processing-In-Memory benchmarks), a benchmark suite of 16 workloads from different application domains (e.g., dense/sparse linear algebra, databases, data analytics, graph processing, neural networks, bioinformatics, image processing), which we identify as memory-bound. We evaluate the performance and scaling characteristics of PrIM benchmarks on the UPMEM PIM architecture, and compare their performance and energy consumption to their state-of-the-art CPU and GPU counterparts. Our extensive evaluation conducted on two real UPMEM-based PIM systems with 640 and 2,556 DPUs provides new insights about suitability of different workloads to the PIM system, programming recommendations for software designers, and suggestions and hints for hardware and architecture designers of future PIM systems.

Understanding a Modern Processing-in-Memory Architecture: Benchmarking and Experimental Characterization

Juan Gómez Luna, Izzat El Hajj, Ivan Fernandez, Christina Giannoula, Geraldo F. Oliveira, Onur Mutlu

https://github.com/CMU-SAFARI/prim-benchmarks
Recent SRC TECHCON Presentation

- **Dr. Juan Gomez-Luna**
  - Benchmarking Memory-Centric Computing Systems: Analysis of Real Processing-in-Memory Hardware
  - Based on two major works

---

**Benchmarking Memory-Centric Computing Systems: Analysis of Real Processing-In-Memory Hardware**

Year: 2021, Pages: 1-7

DOI Bookmark: 10.1109/IGSC54211.2021.9651614

**Authors**

Juan Gómez-Luna, ETH Zürich
Izzat El Hajj, American University of Beirut
Ivan Fernandez, University of Malaga
Christina Giannoula, National Technical University of Athens
Geraldo F. Oliveira, ETH Zürich
Onur Mutlu, ETH Zürich

[https://www.youtube.com/watch?v=nphV36SrysA](https://www.youtube.com/watch?v=nphV36SrysA)
**Key Takeaway 1**

The UPMEM PIM architecture is fundamentally compute bound. As a result, the most suitable workloads are memory-bound.

The throughput saturation point is as low as \( \frac{1}{4} \) OP/B, i.e., 1 integer addition per every 32-bit element fetched.
**Key Takeaway 2**

The most well-suited workloads for the UPMEM PIM architecture use no arithmetic operations or use only simple operations (e.g., bitwise operations and integer addition/subtraction).
**KEY TAKEAWAY 3**

The most well-suited workloads for the UPMEM PIM architecture require little or no communication across DPUs (inter-DPU communication).
Understanding a Modern Processing-in-Memory Architecture: Benchmarking and Experimental Characterization

Juan Gómez Luna, Izzat El Hajj, Ivan Fernandez, Christina Giannoula, Geraldo F. Oliveira, Onur Mutlu

el1goluj@gmail.com

https://github.com/CMU-SAFARI/prim-benchmarks
Juan Gomez-Luna, Izzat El Hajj, Ivan Fernandez, Christina Giannoula, Geraldo F. Oliveira, and Onur Mutlu,
"Benchmarking Memory-Centric Computing Systems: Analysis of Real Processing-in-Memory Hardware"
[arXiv version]
[PrIM Benchmarks Source Code]
[Slides (pptx) (pdf)]
[Talk Video (37 minutes)]
[Lightning Talk Video (3 minutes)]
# PrIM Benchmarks: Application Domains

<table>
<thead>
<tr>
<th>Domain</th>
<th>Benchmark</th>
<th>Short name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dense linear algebra</td>
<td>Vector Addition</td>
<td>VA</td>
</tr>
<tr>
<td></td>
<td>Matrix-Vector Multiply</td>
<td>GEMV</td>
</tr>
<tr>
<td>Sparse linear algebra</td>
<td>Sparse Matrix-Vector Multiply</td>
<td>SpMV</td>
</tr>
<tr>
<td>Databases</td>
<td>Select</td>
<td>SEL</td>
</tr>
<tr>
<td></td>
<td>Unique</td>
<td>UNI</td>
</tr>
<tr>
<td>Data analytics</td>
<td>Binary Search</td>
<td>BS</td>
</tr>
<tr>
<td></td>
<td>Time Series Analysis</td>
<td>TS</td>
</tr>
<tr>
<td>Graph processing</td>
<td>Breadth-First Search</td>
<td>BFS</td>
</tr>
<tr>
<td>Neural networks</td>
<td>Multilayer Perceptron</td>
<td>MLP</td>
</tr>
<tr>
<td>Bioinformatics</td>
<td>Needleman-Wunsch</td>
<td>NW</td>
</tr>
<tr>
<td>Image processing</td>
<td>Image histogram (short)</td>
<td>HST-S</td>
</tr>
<tr>
<td></td>
<td>Image histogram (large)</td>
<td>HST-L</td>
</tr>
<tr>
<td>Parallel primitives</td>
<td>Reduction</td>
<td>RED</td>
</tr>
<tr>
<td></td>
<td>Prefix sum (scan(scan-add))</td>
<td>SCAN-SSA</td>
</tr>
<tr>
<td></td>
<td>Prefix sum (reduce(scan(scan)))</td>
<td>SCAN-RSS</td>
</tr>
<tr>
<td></td>
<td>Matrix transposition</td>
<td>TRNS</td>
</tr>
</tbody>
</table>
**PrIM Benchmarks are Open Source**

- All microbenchmarks, benchmarks, and scripts
- [https://github.com/CMU-SAFARI/prim-benchmarks](https://github.com/CMU-SAFARI/prim-benchmarks)

---

**PrIM (Processing-In-Memory Benchmarks)**

PrIM is the first benchmark suite for a real-world processing-in-memory (PIM) architecture. PrIM is developed to evaluate, analyze, and characterize the first publicly-available real-world processing-in-memory (PIM) architecture, the UPMEM PIM architecture. The UPMEM PIM architecture combines traditional DRAM memory arrays with general-purpose in-order cores, called DRAM Processing Units (DPUs), integrated in the same chip.

PrIM provides a common set of workloads to evaluate the UPMEM PIM architecture with and can be useful for programming, architecture and system researchers all alike to improve multiple aspects of future PIM hardware and software. The workloads have different characteristics, exhibiting heterogeneity in their memory access patterns, operations and data types, and communication patterns. This repository also contains baseline CPU and GPU implementations of PrIM benchmarks for comparison purposes.

PrIM also includes a set of microbenchmarks can be used to assess various architecture limits such as compute throughput and memory bandwidth.
Understanding a Modern PIM Architecture

Benchmarking a New Paradigm: Experimental Analysis and Characterization of a Real Processing-in-Memory System

JUAN GÓMEZ-LUNA¹, IZZAT EL HAJJ², IVAN FERNANDEZ¹,³, CHRISTINA GIANNOLLA¹,⁴, GERALDO F. OLIVEIRA¹, AND ONUR MUTLU¹

¹ETH Zürich
²American University of Beirut
³University of Malaga
⁴National Technical University of Athens

Corresponding author: Juan Gómez-Luna (e-mail: juang@ethz.ch).

https://github.com/CMU-SAFARI/prim-benchmarks
Understanding a Modern PIM Architecture

Juan Gómez Luna, Izzat El Hajj, Ivan Fernandez, Christina Giannoula, Geraldo F. Oliveira, Onur Mutlu

https://github.com/CMU-SAFARI/prim-benchmarks

SAFARI Live Seminar: Understanding a Modern Processing-in-Memory Architecture

2,579 views • Streamed live on Jul 12, 2021

https://www.youtube.com/watch?v=D8Hjiy2iU9I4&list=PL5Q2soXY2Zi_tOTAYm--dYByNPL7JhwR9
More on Analysis of the UPMEM PIM Engine

Inter-DPU Communication

- There is no direct communication channel between DPUs

Inter-DPU communication takes places via the host CPU using CPU-DPU and DPU-CPU transfers

Example communication patterns:
- Merging of partial results to obtain the final result
- Only DPU-CPU transfers
- Redistribution of intermediate results for further computation
- DPU-CPU transfers and CPU-DPU transfers

SAFARI Live Seminar: Understanding a Modern Processing-in-Memory Architecture

1,868 views • Streamed live on Jul 12, 2021

Onur Mutlu Lectures
17.6K subscribers

Talk Title: Understanding a Modern Processing-in-Memory Architecture: Benchmarking and Experimental Characterization
Dr. Juan Gómez-Luna, SAFARI Research Group, D-ITET, ETH Zurich

https://www.youtube.com/watch?v=D8Hjy2iU9j4&list=PL5Q2soXY2Zi_tOTAYm--dYByNPL7JhwR9
More on Analysis of the UPMEM PIM Engine

Data Movement in Computing Systems

- Data movement dominates performance and is a major system energy bottleneck
- Total system energy: data movement accounts for
  - 62% in consumer applications
  - 40% in scientific applications
  - 35% in mobile applications

Understanding a Modern Processing-in-Memory Arch: Benchmarking & Experimental Characterization; 21m

3,482 views · Premiered Jul 25, 2021

https://www.youtube.com/watch?v=Pp9jSU2b9oM&list=PL5Q2soXY2Zi8_VVChACnON4sfh2bJ5IrD&index=159
ML Training on a Real PIM System

Machine Learning Training on a Real Processing-in-Memory System

Juan Gómez-Luna¹ Yuxin Guo¹ Sylvan Brocard² Julien Legriel² Remy Cimadomo² Geraldo F. Oliveira¹ Gagandeep Singh¹ Onur Mutlu¹

¹ETH Zürich ²UPMEM

An Experimental Evaluation of Machine Learning Training on a Real Processing-in-Memory System

Juan Gómez-Luna¹ Yuxin Guo¹ Sylvan Brocard² Julien Legriel² Remy Cimadomo² Geraldo F. Oliveira¹ Gagandeep Singh¹ Onur Mutlu¹

¹ETH Zürich ²UPMEM

https://www.youtube.com/watch?v=qeukNs5Xl3g&t=11226s
ML Training on a Real PIM System

• Need to optimize data representation
  (1) fixed-point
  (2) quantization
  (3) hybrid precision

• Use lookup tables (LUTs) to implement complex functions (e.g., sigmoid)

• Optimize data placement & layout for streaming

• Large speedups: 2.8X/27X vs. CPU, 1.3x/3.2x vs. GPU
ML Training on Real PIM Talk Video

Machine Learning Training
on a Real Processing-in-Memory System

Juan Gómez Luna, Yuxin Guo, Sylvan Brocard,
Julien Legriel, Remy Cimadomo, Geraldo F. Oliveira,
Gagandeep Singh, Onur Mutlu

juang@ethz.ch

ETH Zürich "SAFARI"

ISVLSI 2022 Special Session on Processing-in-Memory

1,345 views • Premiered Aug 9, 2022

https://www.youtube.com/watch?v=qeukNs5XI3g&t=11226s
SpMV Multiplication on Real PIM Systems

- Appears in SIGMETRICS 2022

**SparseP**: Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Systems

CHRISTINA GIANNELLA, ETH Zürich, Switzerland and National Technical University of Athens, Greece

IVAN FERNANDEZ, ETH Zürich, Switzerland and University of Malaga, Spain

JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland

NECTARIES KOZIRIS, National Technical University of Athens, Greece

GEORGIOS GOUUMAS, National Technical University of Athens, Greece

ONUR MUTLU, ETH Zürich, Switzerland


[https://github.com/CMU-SAFARI/SparseP](https://github.com/CMU-SAFARI/SparseP)

[https://www.youtube.com/watch?v=5kaOsJKlGrE](https://www.youtube.com/watch?v=5kaOsJKlGrE)
SparseP
Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Architectures

Christina Giannoula
Ivan Fernandez, Juan Gomez-Luna,
Nectarios Koziris, Georgios Goumas, Onur Mutlu
SparseP Summary

Efficient Algorithmic Designs

The first open-source Sparse Matrix Vector Multiplication (SpMV) software package, SparseP, for real Processing-In-Memory (PIM) systems

SparseP is Open-Source

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

Extensive Characterization

The first comprehensive analysis of SpMV on the first real commercial PIM architecture

Recommendations for Architects and Programmers

SparseP: SpMV Library for Real PIMs

Our Contributions:

1. Design efficient SpMV kernels for current and future PIM systems
   - 25 SpMV kernels
     - 4 compressed matrix formats (CSR, COO, BCSR, BCOO)
     - 6 data types
     - 4 data partitioning techniques
     - Various load balancing schemes among PIM cores/threads
     - 3 synchronization approaches

2. Provide a comprehensive analysis of SpMV on the first commercially-available real PIM system
   - 26 sparse matrices
   - Comparisons to state-of-the-art CPU and GPU systems
   - Recommendations for software, system and hardware designers
SparseP Talk Video

Towards Efficient Sparse Matrix Vector Multiplication on Real Processing-In-Memory Architectures

Christina Giannoula
Ivan Fernandez, Juan Gomez-Luna, Nectarios Koziris, Georgios Goumas, Onur Mutlu

https://www.youtube.com/watch?v=5kaOsJKlGrE
Samsung Develops Industry’s First High Bandwidth Memory with AI Processing Power

Korea on February 17, 2021

The new architecture will deliver over twice the system performance and reduce energy consumption by more than 70%

Samsung Electronics, the world leader in advanced memory technology, today announced that it has developed the industry’s first High Bandwidth Memory (HBM) integrated with artificial intelligence (AI) processing power — the HBM-PIM. The new processing-in-memory (PIM) architecture brings powerful AI computing capabilities inside high-performance memory, to accelerate large-scale processing in data centers, high performance computing (HPC) systems and AI-enabled mobile applications.

Kwangil Park, senior vice president of Memory Product Planning at Samsung Electronics stated, “Our groundbreaking HBM-PIM is the industry’s first programmable PIM solution tailored for diverse AI-driven workloads such as HPC, training and inference. We plan to build upon this breakthrough by further collaborating with AI solution providers for even more advanced PIM-powered applications.”

Samsung Function-in-Memory DRAM (2021)

- FIMDRAM based on HBM2

Chip Specification

- 128DQ / 8CH / 16 banks / BL4
- 32 PCU blocks (1 FIM block/2 banks)

FP16 ADD / Multiply (MUL) / Multiply-Accumulate (MAC) / Multiply-and-Add (MAD)

[3D Chip Structure of HBM with FIMDRAM]
Programmable Computing Unit

- Configuration of PCU block
  - Interface unit to control data flow
  - Execution unit to perform operations
  - Register group
    - 32 entries of CRF for instruction memory
    - 16 GRF for weight and accumulation
    - 16 SRF to store constants for MAC operations

[Block diagram of PCU in FIMDGRAM]
## Available instruction list for FIM operation

<table>
<thead>
<tr>
<th>Type</th>
<th>CMD</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Floating Point</td>
<td>ADD</td>
<td>FP16 addition</td>
</tr>
<tr>
<td></td>
<td>MUL</td>
<td>FP16 multiplication</td>
</tr>
<tr>
<td></td>
<td>MAC</td>
<td>FP16 multiply-accumulate</td>
</tr>
<tr>
<td></td>
<td>MAD</td>
<td>FP16 multiply and add</td>
</tr>
<tr>
<td>Data Path</td>
<td>MOVE</td>
<td>Load or store data</td>
</tr>
<tr>
<td></td>
<td>FILL</td>
<td>Copy data from bank to GRFs</td>
</tr>
<tr>
<td>Control Path</td>
<td>NOP</td>
<td>Do nothing</td>
</tr>
<tr>
<td></td>
<td>JUMP</td>
<td>Jump instruction</td>
</tr>
<tr>
<td></td>
<td>EXIT</td>
<td>Exit instruction</td>
</tr>
</tbody>
</table>
Samsung Function-in-Memory DRAM (2021)

Chip Implementation

- Mixed design methodology to implement FIMDRAM
  - Full-custom + Digital RTL

[Digital RTL design for PCU block]
Samsung AxDIMM (2021)

- DDRx-PIM
  - DLRM recommendation system

SK hynix Develops PIM, Next-Generation AI Accelerator

February 16, 2022

Seoul, February 16, 2022

SK hynix (or "the Company", [www.skhynix.com](http://www.skhynix.com)) announced on February 16 that it has developed PIM*, a next-generation memory chip with computing capabilities.

*PIM (Processing in Memory): A next-generation technology that provides a solution for data congestion issues for AI and big data by adding computational functions to semiconductor memory

It has been generally accepted that memory chips store data and CPU or GPU, like human brain, process data. SK hynix, following its challenge to such notion and efforts to pursue innovation in the next-generation smart memory, has found a breakthrough solution with the development of the latest technology.

SK hynix plans to showcase its PIM development at the world's most prestigious semiconductor conference, 2022 ISSCC*, in San Francisco at the end of this month. The company expects continued efforts for innovation of this technology to bring the memory-centric computing, in which semiconductor memory plays a central role, a step closer to the reality in devices such as smartphones.

*ISSCC: The International Solid-State Circuits Conference will be held virtually from Feb. 20 to Feb. 24 this year with a theme of "Intelligent Silicon for a Sustainable World"

For the first product that adopts the PIM technology, SK hynix has developed a sample of GDDR6-AIM (Accelerator in memory). The GDDR6-AIM adds computational functions to GDDR6™ memory chips, which process data at 16Gbps. A combination of GDDR6-AIM with CPU or GPU instead of a typical DRAM makes certain computation speed 16 times faster. GDDR6-AIM is widely expected to be adopted for machine learning, high-performance computing, and big data computation and storage.

29.1 184QPS/W 64Mb/mm² 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System

Dimin Niu¹, Shuangchen Li¹, Yuhao Wang¹, Wei Han¹, Zhe Zhang², Yijin Guan², Tianchan Guan³, Fei Sun¹, Fei Xue¹, Lide Duan¹, Yuanwei Fang¹, Hongzhong Zheng¹, Xiping Jiang⁴, Song Wang⁴, Fengguo Zuo⁴, Yubing Wang⁴, Bing Yu⁴, Qiwei Ren⁴, Yuan Xie¹

Figure 29.1.2: Illustration of 3D-stacked chip, cross-illustration of package, DRAM array layout and design blocks on logic die.

Figure 29.1.7: Die micrographs of DRAM die, NE and ME. Detailed specifications of DRAM die and logic die.
Eliminating the Adoption Barriers

Processing-in-Memory in the Real World
DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks

GERALDO F. OLIVEIRA, ETH Zürich, Switzerland
JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
LOIS OROSA, ETH Zürich, Switzerland
SAUGATA GHOSE, University of Illinois at Urbana–Champaign, USA
NANDITA VIJAYKUMAR, University of Toronto, Canada
IVAN FERNANDEZ, University of Malaga, Spain & ETH Zürich, Switzerland
MOHAMMAD SADROSADATI, Institute for Research in Fundamental Sciences (IPM), Iran & ETH Zürich, Switzerland
ONUR MUTLU, ETH Zürich, Switzerland

Data movement between the CPU and main memory is a first-order obstacle against improving performance, scalability, and energy efficiency in modern systems. Computer systems employ a range of techniques to reduce overheads tied to data movement, spanning from traditional mechanisms (e.g., deep multi-level cache hierarchies, aggressive hardware prefetchers) to emerging techniques such as Near-Data Processing (NDP), where some computation is moved close to memory. Prior NDP works investigate the root causes of data movement bottlenecks using different profiling methodologies and tools. However, there is still a lack of understanding about the key metrics that can identify different data movement bottlenecks and their relation to traditional and emerging data movement mitigation mechanisms. Our goal is to methodically identify potential sources of data movement overhead over a broad set of applications and to comprehensively compare traditional compute-centric data movement mitigation techniques (e.g., caching and prefetching) to more memory-centric techniques (e.g., NDP), thereby developing a rigorous understanding of the best techniques to mitigate each source of data movement.

With this goal in mind, we perform the first large-scale characterization of a wide variety of applications, across a wide range of application domains, to identify fundamental program properties that lead to data movement to/from main memory. We develop the first systematic methodology to classify applications based on the sources contributing to data movement bottlenecks. From our large-scale characterization of 77K functions across 345 applications, we select 144 functions to form the first open-source benchmark suite (DAMOV) for main memory data movement studies. We select a diverse range of functions that (1) represent different types of data movement bottlenecks, and (2) come from a wide range of application domains. Using NDP as a case study, we identify new insights about the different data movement bottlenecks and use these insights to determine the most suitable data movement mitigation mechanism for a particular application. We open-source DAMOV and the complete source code for our new characterization methodology at https://github.com/CMU-SAFARI/DAMOV.

When to Employ Near-Data Processing?

Mobile consumer workloads
(GoogleWL²)

Graph processing
(Tesseract¹)

Databases
(Polynesia⁵)

Time series analysis
(NATSA⁶)

Near-Data Processing

Neural networks
(GoogleWL²)

DNA sequence mapping
(GenASM³; GRIM-Filter⁴)

...
Step 1: Application Profiling

- We analyze 345 applications from distinct domains:
  - Graph Processing
  - Deep Neural Networks
  - Physics
  - High-Performance Computing
  - Genomics
  - Machine Learning
  - Databases
  - Data Reorganization
  - Image Processing
  - Map-Reduce
  - Benchmarking
  - Linear Algebra
  ...

SAFARI
Step 3: Memory Bottleneck Analysis

Six classes of data movement bottlenecks:

- each class ↔ data movement mitigation mechanism

Memory Bottleneck Class

1a: DRAM Bandwidth
1b: DRAM Latency
1c: L1/L2 Cache Capacity
2a: L3 Cache Contention
2b: L1 Cache Capacity
2c: Compute-Bound
DAMOV is Open Source

• We open-source our benchmark suite and our toolchain
DAMOV is Open Source

• We open-source our benchmark suite and our toolchain

Get DAMOV at:
https://github.com/CMU-SAFARI/DAMOV

DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks

DAMOV is a benchmark suite and a methodical framework targeting the study of data movement bottlenecks in modern applications. It is intended to study new architectures, such as near-data processing.

The DAMOV benchmark suite is the first open-source benchmark suite for main memory data movement-related studies, based on our systematic characterization methodology. This suite consists of 144 functions representing different sources of data movement bottlenecks and can be used as a baseline benchmark set for future data-movement mitigation research. The applications in the DAMOV benchmark suite belong to popular benchmark suites, including BWA, Chai, Darknet, GASE, Hardware Effects, Hashjoin, HPCC, HPCG, Ligra, PARSEC, Parboil, PolyBench, Phoenix, Rodinia, SPLASH-2, STREAM.
More on DAMOV Analysis Methodology & Workloads

**Step 3: Memory Bottleneck Classification (2/2)**

- **Goal:** identify the specific sources of data movement bottlenecks

**Scalability Analysis:**
- 1, 4, 16, 64, and 256 out-of-order/in-order host and NDP CPU cores
- 3D-stacked memory as main memory

SAFARI Live Seminar: DAMOV: A New Methodology & Benchmark Suite for Data Movement Bottlenecks

https://www.youtube.com/watch?v=GWideVyo0nM&list=PL5Q2soXY2Zi_TOTAYm--dYByNPL7JhwR9&index=3
More on DAMOV Methods & Benchmarks

- Geraldo F. Oliveira, Juan Gomez-Luna, Lois Orosa, Saugata Ghose, Nandita Vijaykumar, Ivan Fernandez, Mohammad Sadrosadati, and Onur Mutlu,
  "DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks"
  [arXiv preprint]
  [IEEE Access version]
  [DAMOV Suite and Simulator Source Code]
  [SAFARI Live Seminar Video (2 hrs 40 mins)]
  [Short Talk Video (21 minutes)]

DAMOV: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks

GERALDO F. OLIVEIRA, ETH Zürich, Switzerland
JUAN GÓMEZ-LUNA, ETH Zürich, Switzerland
LOIS OROSA, ETH Zürich, Switzerland
SAUGATA GHOSE, University of Illinois at Urbana–Champaign, USA
NANDITA VIJAYKUMAR, University of Toronto, Canada
IVAN FERNANDEZ, University of Malaga, Spain & ETH Zürich, Switzerland
MOHAMMAD SADROSADATI, ETH Zürich, Switzerland
ONUR MUTLU, ETH Zürich, Switzerland
Challenge and Opportunity for Future

Fundamentally Energy-Efficient (Data-Centric) Computing Architectures
Challenge and Opportunity for Future

Fundamentally High-Performance (Data-Centric) Computing Architectures
Challenge and Opportunity for Future Computing Architectures with Minimal Data Movement
More Info in This Longer Tutorial…

- Onur Mutlu,
  "Memory-Centric Computing"
  Education Class at *Embedded Systems Week (ESWEEK)*, Virtual, 9 October 2021.
  [Slides (pptx) (pdf)]
  [Abstract (pdf)]
  [Talk Video (2 hours, including Q&A)]
  [Invited Paper at DATE 2021]
  ["A Modern Primer on Processing in Memory" paper]

https://www.youtube.com/watch?v=N1Ac1ov1JOM

https://www.youtube.com/onurmutlulectures
Concluding Remarks
Concluding Remarks

- We must design systems to be balanced, high-performance, energy-efficient (all at the same time) → intelligent systems
  - Data-centric, data-driven, data-aware

- Enable computation capability inside and close to memory

- This can
  - Lead to orders-of-magnitude improvements
  - Enable new applications & computing platforms
  - Enable better understanding of nature
  - ...

- Future of truly memory-centric computing is bright
  - We need to do research & design across the computing stack
Fundamentally Better Architectures

Data-centric

Data-driven

Data-aware
We Need to Revisit the Entire Stack

We can get there step by step
We Need to Exploit Good Principles

- Data-centric system design
- All components intelligent
- Better (cross-layer) communication, better interfaces
- Better-than-worst-case design
- Heterogeneity
- Flexibility, adaptability

Open minds
Onur Mutlu,
"Intelligent Architectures for Intelligent Computing Systems"
[Slides (pptx) (pdf)]
[IEDM Tutorial Slides (pptx) (pdf)]
[Short DATE Talk Video (11 minutes)]
[Longer IEDM Tutorial Video (1 hr 51 minutes)]

Intelligent Architectures for Intelligent Computing Systems

Onur Mutlu
ETH Zurich
omutlu@gmail.com
Funding Acknowledgments

- Alibaba, AMD, ASML, Google, Facebook, Hi-Silicon, HP
- Labs, Huawei, IBM, Intel, Microsoft, Nvidia, Oracle,
- Qualcomm, Rambus, Samsung, Seagate, VMware, Xilinx
- NSF
- NIH
- GSRC
- SRC
- CyLab
- EFCL
Acknowledgments

think BIG, Aim HIGH!

https://safari.ethz.ch
Onur Mutlu’s SAFARI Research Group

Computer architecture, HW/SW, systems, bioinformatics, security, memory

https://safari.ethz.ch/safari-newsletter-january-2021/

Think BIG, Aim HIGH!

SAFARI
https://safari.ethz.ch
SAFARI Newsletter December 2021 Edition

https://safari.ethz.ch/safari-newsletter-december-2021/

Think Big, Aim High

View in your browser
December 2021
Referenced Papers, Talks, Artifacts

- All are available at

  [https://people.inf.ethz.ch/omutlu/projects.htm](https://people.inf.ethz.ch/omutlu/projects.htm)

  [https://www.youtube.com/onurmutlulectures](https://www.youtube.com/onurmutlulectures)

  [https://github.com/CMU-SAFARI/](https://github.com/CMU-SAFARI/)
Open Source Tools: SAFARI GitHub

SAFARI Research Group at ETH Zurich and Carnegie Mellon University

Site for source code and tools distribution from SAFARI Research Group at ETH Zurich and Carnegie Mellon University.

ETH Zurich and Carnegie Mellon U...  https://safari.ethz.ch/  omutlu@gmail.com

Overview  Repositories  71  Projects  Packages  Teams  1  People  44  Settings

Pinned

ramulator  Public
A Fast and Extensible DRAM Simulator, with built-in support for modeling many different DRAM technologies including DDRx, LPDDRx, GDDRx, WIOx, HBMx, and various academic proposals. Described in the...

C++  311  161

prim-benchmarks  Public
PrIM (Processing-In-Memory benchmarks) is the first benchmark suite for a real-world processing-in-memory (PIM) architecture. PrIM is developed to evaluate, analyze, and characterize the first publ...

C  53  21

DAMOV  Public
DAMOV is a benchmark suite and a methodical framework targeting the study of data movement bottlenecks in modern applications. It is intended to study new architectures, such as near-data processin...

C++  26  4

SneakySnake  Public
SneakySnake is the first and the only pre-alignment filtering algorithm that works efficiently and fast on modern CPU, FPGA, and GPU architectures. It greatly (by more than two orders of magnitude...

VHDL  41  8

MQSim  Public
MQSim is a fast and accurate simulator modeling the performance of modern multi-queue (MQ) SSDs as well as traditional SATA based SSDs. MQSim faithfully models new high-bandwidth protocol implement...

C++  146  93

rowhammer  Public

C  189  41

https://github.com/CMU-SAFARI/
Special Research Sessions & Courses

- Special Session at ISVLSI 2022: 9 cutting-edge talks

[Video Link](https://www.youtube.com/watch?v=qeukNs5XI3g)
Comp Arch (Fall’21)

- **Fall 2021 Edition:**
  - [https://safari.ethz.ch/architecture/fall2021/doku.php?id=schedule](https://safari.ethz.ch/architecture/fall2021/doku.php?id=schedule)

- **Fall 2020 Edition:**

- **Youtube Livestream (2021):**
  - [https://www.youtube.com/watch?v=4yfkM_5EFg0&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAGILkTOF](https://www.youtube.com/watch?v=4yfkM_5EFg0&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAGILkTOF)

- **Youtube Livestream (2020):**
  - [https://www.youtube.com/watch?v=c3mPdZA-Fmc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN](https://www.youtube.com/watch?v=c3mPdZA-Fmc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN)

- Master’s level course
  - Taken by Bachelor’s/Masters/PhD students
  - Cutting-edge research topics + fundamentals in Computer Architecture
  - 5 Simulator-based Lab Assignments
  - Potential research exploration
  - Many research readings

[https://www.youtube.com/onurmutlulectures](https://www.youtube.com/onurmutlulectures)
PIM Course (Spring 2022)

- **Spring 2022 Edition:**
  - https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=processing_in_memory

- **Youtube Livestream:**
  - https://www.youtube.com/watch?v=9e4Chnwdovo&list=PL5Q2soXY2Zi-841fUYYUK9EsXKhQKRPyX

- **Project course**
  - Taken by Bachelor’s/Master’s students
  - Processing-in-Memory lectures
  - Hands-on research exploration
  - Many research readings
Genomics (Spring 2022)

- **Spring 2022 Edition:**
  - [https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=bioinformatics](https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=bioinformatics)

- **Youtube Livestream:**
  - [https://www.youtube.com/watch?v=DEL5A_Y3TI&list=PL5Q2soXY2Zi8NrPDgOR1yRU_Cxxjw-u18](https://www.youtube.com/watch?v=DEL5A_Y3TI&list=PL5Q2soXY2Zi8NrPDgOR1yRU_Cxxjw-u18)

- **Project course**
  - Taken by Bachelor’s/Master’s students
  - Genomics lectures
  - Hands-on research exploration
  - Many research readings
Hetero. Systems (Spring’22)

- **Spring 2022 Edition:**
  - [https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=heterogeneous_systems](https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=heterogeneous_systems)

- **Youtube Livestream:**
  - [https://www.youtube.com/watch?v=oFO5fTrgFIY&list=PL5Q2soXY2Zi9XrgXR38IM_FTjmY6h7Gzm](https://www.youtube.com/watch?v=oFO5fTrgFIY&list=PL5Q2soXY2Zi9XrgXR38IM_FTjmY6h7Gzm)

- **Project course**
  - Taken by Bachelor’s/Master’s students
  - GPU and Parallelism lectures
  - Hands-on research exploration
  - Many research readings

---

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Livestream</th>
<th>Meeting</th>
<th>Learning Materials</th>
<th>Assignments</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>15.03 Tue.</td>
<td>Premiere</td>
<td>M1: P&amp;S Course Presentation (PDF) (PPT)</td>
<td>Required Materials</td>
<td>HW 0 Out</td>
</tr>
<tr>
<td>W2</td>
<td>22.03 Tue.</td>
<td>Premiere</td>
<td>M2: SIMD Processing and GPUs (PDF) (PPT)</td>
<td>Recommended Materials</td>
<td></td>
</tr>
<tr>
<td>W3</td>
<td>29.03 Tue.</td>
<td>Premiere</td>
<td>M3: GPU Software Hierarchy (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W4</td>
<td>05.04 Tue.</td>
<td>Premiere</td>
<td>M4: GPU Memory Hierarchy (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W5</td>
<td>12.04 Tue.</td>
<td>Premiere</td>
<td>M5: GPU Performance Considerations (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W6</td>
<td>19.04 Tue.</td>
<td>Premiere</td>
<td>M6: Parallel Patterns: Reduction (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W7</td>
<td>26.04 Tue.</td>
<td>Premiere</td>
<td>M7: Parallel Patterns: Histogram (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W8</td>
<td>03.05 Tue.</td>
<td>Premiere</td>
<td>M8: Parallel Patterns: Convolution (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W9</td>
<td>10.05 Tue.</td>
<td>Premiere</td>
<td>M9: Parallel Patterns: Prefix Sum (Scan) (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W10</td>
<td>17.05 Tue.</td>
<td>Premiere</td>
<td>M10: Parallel Patterns: Sparse Matrices (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W11</td>
<td>24.05 Tue.</td>
<td>Premiere</td>
<td>M11: Parallel Patterns: Graph Search (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W12</td>
<td>01.06 Wed.</td>
<td>Premiere</td>
<td>M12: Parallel Patterns: Merge Sort (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W13</td>
<td>07.06 Tue.</td>
<td>Premiere</td>
<td>M13: Dynamic Parallelism (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W14</td>
<td>15.06 Wed.</td>
<td>Premiere</td>
<td>M14: Collaborative Computing (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W15</td>
<td>24.06 Fri.</td>
<td>Premiere</td>
<td>M15: GPU Acceleration of Genome Sequence Alignment (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W16</td>
<td>14.07 Thu.</td>
<td>Premiere</td>
<td>M16: Accelerating Agent-based Simulations (PDF) (OP)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
HW/SW Co-Design (Spring 2022)

- **Spring 2022 Edition:**
  - [https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=hw_sw_co_design](https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=hw_sw_co_design)

- **Youtube Livestream:**
  - [https://youtube.com/playlist?list=PL5Q2soXY2Zi8nH7un3ghD2nutKWWDk-NK](https://youtube.com/playlist?list=PL5Q2soXY2Zi8nH7un3ghD2nutKWWDk-NK)

- Project course
  - Taken by Bachelor’s/Master’s students
  - HW/SW co-design lectures
  - Hands-on research exploration
  - Many research readings

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Livestream</th>
<th>Meeting</th>
<th>Materials</th>
<th>Assignments</th>
</tr>
</thead>
<tbody>
<tr>
<td>W0</td>
<td>16.03</td>
<td>Live</td>
<td>Intro to HW/SW Co-Design (PPTX) (PDF)</td>
<td>Required</td>
<td>HW 0 Out</td>
</tr>
<tr>
<td>W1</td>
<td>23.03</td>
<td>Live</td>
<td>Project selection</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>W2</td>
<td>30.03</td>
<td>Live</td>
<td>Virtual Memory (I)</td>
<td>Required</td>
<td></td>
</tr>
<tr>
<td>W3</td>
<td>13.04</td>
<td>Live</td>
<td>Virtual Memory (II)</td>
<td>Required</td>
<td></td>
</tr>
</tbody>
</table>
SSD Course (Spring 2022)

- **Spring 2022 Edition:**

- **Youtube Livestream:**
  - [https://www.youtube.com/watch?v=_q4rm71DsY4&list=PL5Q2soXY2Zi8vabcse1kL22DEcgMI2RAq](https://www.youtube.com/watch?v=_q4rm71DsY4&list=PL5Q2soXY2Zi8vabcse1kL22DEcgMI2RAq)

- **Project course**
  - Taken by Bachelor’s/Master’s students
  - SSD Basics and Advanced Topics
  - Hands-on research exploration
  - Many research readings
Memory-Centric Computing

Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu
22 September 2022
Simon Fraser University Distinguished Lecture
Backup Slides
Aside: Intelligent Controller for NAND Flash


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
System Desirables

- Self-managing, independent components
- All components intelligent & equal partners
- Easy collaboration & partitioning across all components
- Fine-grained communication of data & tasks
- Seamless caching & translation & protection anywhere
- Execution anywhere without rewriting code
- Flexibility, adaptability, self-optimization
SAFARI Research Group
Dear SAFARI friends,

2019 and the first three months of 2020 have been very positive and eventful times for SAFARI.
Dear SAFARI friends,

Happy New Year! We are excited to share our group highlights with you in this second edition of the SAFARI newsletter (You can find the first edition from April 2020 [here](https://safari.ethz.ch/safari-newsletter-january-2021/)). 2020 has

A Talk on Impactful Research & Teaching

Applying to Grad School & Doing Impactful Research

Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu
13 June 2020
Undergraduate Architecture Mentoring Workshop @ ISCA 2021

SAFAIR ETH Zürich Carnegie Mellon

Arch. Mentoring Workshop @ISCA'21 - Applying to Grad School & Doing Impactful Research - Onur Mutlu
1,563 views • Premiered Jun 16, 2021

Panel talk at Undergraduate Architecture Mentoring Workshop at ISCA 2021
(https://sites.google.com/wisc.edu/uar...)

https://www.youtube.com/watch?v=83tlorht7Mc&list=PL5Q2soXY2zi8D_5MGV6EnXEJHnV2YFBJl&index=54
Onur Mutlu,
"Memory-Centric Computing"

Education Class at *Embedded Systems Week (ESWEEK)*, Virtual, 9 October 2021.

[Slides (pptx) (pdf)]
[Abstract (pdf)]
[Talk Video (2 hours, including Q&A)]
[Invited Paper at DATE 2021]
["A Modern Primer on Processing in Memory" paper]

https://www.youtube.com/watch?v=N1Ac1ov1JOM

https://www.youtube.com/onurmutlulectures
Latest Longer & Detailed Tutorial on PIM


[Slides (pptx)](https://people.inf.ethz.ch/omutlu)
[Abstract (pdf)](https://www.youtube.com/onurmutlulectures)
[Talk Video (2 hours, including Q&A)](https://www.youtube.com/watch?v=N1Ac1ov1JOM)
[Invited Paper at DATE 2021](https://www.youtube.com/watch?v=N1Ac1ov1JOM)

https://www.youtube.com/watch?v=N1Ac1ov1JOM

https://www.youtube.com/onurmutlulectures
Detailed Lectures on PIM (I)

- Computer Architecture, Fall 2020, Lecture 6
  - Computation in Memory (ETH Zürich, Fall 2020)
  - https://www.youtube.com/watch?v=oGcZAGwfEUE&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=12

- Computer Architecture, Fall 2020, Lecture 7
  - Near-Data Processing (ETH Zürich, Fall 2020)
  - https://www.youtube.com/watch?v=j2GIigqn1Qw&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=13

- Computer Architecture, Fall 2020, Lecture 11a
  - Memory Controllers (ETH Zürich, Fall 2020)
  - https://www.youtube.com/watch?v=TeG773OgiMQ&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=20

- Computer Architecture, Fall 2020, Lecture 12d
  - Real Processing-in-DRAM with UPMEM (ETH Zürich, Fall 2020)
  - https://www.youtube.com/watch?v=Sscy1Wrr22A&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=25

SAFARI

https://www.youtube.com/onurmutlulectures
Detailed Lectures on PIM (II)

- **Computer Architecture, Fall 2020, Lecture 15**
  - Emerging Memory Technologies (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=AlE1rD9G_YU&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=28](https://www.youtube.com/watch?v=AlE1rD9G_YU&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=28)

- **Computer Architecture, Fall 2020, Lecture 16a**
  - Opportunities & Challenges of Emerging Memory Technologies (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=pmLszWGmMGQ&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=29](https://www.youtube.com/watch?v=pmLszWGmMGQ&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=29)

- **Computer Architecture, Fall 2020, Guest Lecture**
  - In-Memory Computing: Memory Devices & Applications (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=wNmqQHiEZNk&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=41](https://www.youtube.com/watch?v=wNmqQHiEZNk&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=41)

[SAFARI](https://www.youtube.com/onurmutlulectures)
Comp Arch (Current)

- **Fall 2021 Edition:**
  - [https://safari.ethz.ch/architecture/fall2021/doku.php?id=schedule](https://safari.ethz.ch/architecture/fall2021/doku.php?id=schedule)

- **Youtube Livestream:**
  - [https://www.youtube.com/watch?v=4yfKM_5EFgo&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAG1kTOF](https://www.youtube.com/watch?v=4yfKM_5EFgo&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAG1kTOF)

- Master’s level course
  - Taken by Bachelor’s/Masters/PhD students
  - Cutting-edge research topics + fundamentals in Computer Architecture
  - 5 Simulator-based Lab Assignments
  - Potential research exploration
  - Many research readings
PIM Course (Current)

- **Fall 2021 Edition:**

- **Youtube Livestream:**
  - [https://www.youtube.com/watch?v=9e4Chnwdovo&list=PL5Q2soXY2Zi-841fUYYUK9EsXKhQKRPyX](https://www.youtube.com/watch?v=9e4Chnwdovo&list=PL5Q2soXY2Zi-841fUYYUK9EsXKhQKRPyX)

- **Project course**
  - Taken by Bachelor’s/Master’s students
  - Processing-in-Memory lectures
  - Hands-on research exploration
  - Many research readings

---

**Fall 2021 Meetings/Schedule**

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Livestream</th>
<th>Meeting</th>
<th>Learning Materials</th>
<th>Assignments</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>05.10 Tue.</td>
<td>Live</td>
<td>M1: P&amp;S PIM Course Presentation (PDF) (PPT)</td>
<td>Required Materials</td>
<td>HW 0 Out</td>
</tr>
<tr>
<td>W2</td>
<td>12.10 Tue.</td>
<td>Live</td>
<td>M2: Real-World PIM Architectures (PDF) (PPT)</td>
<td>Recommended Materials</td>
<td></td>
</tr>
<tr>
<td>W3</td>
<td>19.10 Tue.</td>
<td>Live</td>
<td>M3: Real-World PIM Architectures II (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W4</td>
<td>26.10 Tue.</td>
<td>Live</td>
<td>M4: Real-World PIM Architectures III (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W5</td>
<td>02.11 Tue.</td>
<td>Live</td>
<td>M5: Real-World PIM Architectures IV (PDF) (PPT)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W6</td>
<td>09.11 Tue.</td>
<td>Live</td>
<td>M6: End-to-End Framework for Processing using Memory (PDF) (PPT)</td>
<td>Required Materials</td>
<td>HW 0 Out</td>
</tr>
<tr>
<td>W7</td>
<td>16.11 Tue.</td>
<td>Live</td>
<td>M7: How to Evaluate Data Movement Bottlenecks (PDF) (PPT)</td>
<td>Required Materials</td>
<td>HW 0 Out</td>
</tr>
<tr>
<td>W8</td>
<td>23.11 Tue.</td>
<td>Live</td>
<td>M8: Programming PIM Architectures (PDF) (PPT)</td>
<td>Required Materials</td>
<td>HW 0 Out</td>
</tr>
<tr>
<td>W9</td>
<td>30.11 Tue.</td>
<td>Live</td>
<td>M9: Benchmarking and Workload Suitability on PIM (PDF) (PPT)</td>
<td>Required Materials</td>
<td>HW 0 Out</td>
</tr>
<tr>
<td>W10</td>
<td>07.12 Tue.</td>
<td>Live</td>
<td>M10: Bit-Serial SIMD Processing using DRAM (PDF) (PPT)</td>
<td>Required Materials</td>
<td>HW 0 Out</td>
</tr>
</tbody>
</table>
Data-Driven Architectures
Corollaries: Architectures Today …

- Architectures are terrible at dealing with data
  - Designed to mainly store and move data vs. to compute
  - They are processor-centric as opposed to data-centric

- Architectures are terrible at taking advantage of vast amounts of data (and metadata) available to them
  - Designed to make simple decisions, ignoring lots of data
  - They make human-driven decisions vs. data-driven decisions

- Architectures are terrible at knowing and exploiting different properties of application data
  - Designed to treat all data as the same
  - They make component-aware decisions vs. data-aware
Exploiting Data to Design Intelligent Architectures
System Architecture Design Today

- Human-driven
  - Humans design the policies (how to do things)

- Many (too) simple, short-sighted policies all over the system

- No automatic data-driven policy learning

- (Almost) no learning: cannot take lessons from past actions

Can we design fundamentally intelligent architectures?
An Intelligent Architecture

- Data-driven
  - Machine learns the “best” policies (how to do things)

- Sophisticated, workload-driven, changing, far-sighted policies

- Automatic data-driven policy learning

- All controllers are intelligent data-driven agents

How do we start?
Self-Optimizing Memory Controllers
Memory Controller

Resolves memory contention by scheduling requests

How to schedule requests to maximize system performance?
Why are Memory Controllers Difficult to Design?

- Need to obey **DRAM timing constraints** for correctness
  - There are many (50+) timing constraints in DRAM
  - $t_{WTR}$: Minimum number of cycles to wait before issuing a read command after a write command is issued
  - $t_{RC}$: Minimum number of cycles between the issuing of two consecutive activate commands to the same bank
  - ...

- Need to **keep track of many resources** to prevent conflicts
  - Channels, banks, ranks, data bus, address bus, row buffers, ...

- Need to handle **DRAM refresh**

- Need to **manage power** consumption

- Need to **optimize performance & QoS** (in the presence of constraints)
  - Reordering is not simple
  - Fairness and QoS needs complicates the scheduling problem
  - ...

...
Many Memory Timing Constraints

Many Memory Timing Constraints


![Diagram of Three Phases of DRAM Access](image)

Table 2. Timing Constraints (DDR3-1066) [43]

<table>
<thead>
<tr>
<th>Phase</th>
<th>Commands</th>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ACT $\rightarrow$ READ</td>
<td>tRC</td>
<td>15ns</td>
</tr>
<tr>
<td></td>
<td>ACT $\rightarrow$ WRITE</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACT $\rightarrow$ PRE</td>
<td>tRAS</td>
<td>37.5ns</td>
</tr>
<tr>
<td>2</td>
<td>READ $\rightarrow$ data</td>
<td>tCL</td>
<td>15ns</td>
</tr>
<tr>
<td></td>
<td>WRITE $\rightarrow$ data</td>
<td>tCWL</td>
<td>11.25ns</td>
</tr>
<tr>
<td></td>
<td>data burst</td>
<td>tBL</td>
<td>7.5ns</td>
</tr>
<tr>
<td>3</td>
<td>PRE $\rightarrow$ ACT</td>
<td>tRP</td>
<td>15ns</td>
</tr>
<tr>
<td>1 &amp; 3</td>
<td>ACT $\rightarrow$ ACT</td>
<td>tRC</td>
<td>52.5ns</td>
</tr>
<tr>
<td></td>
<td>(tRAS+tRP)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Memory Controller Design Is Becoming More Difficult

- Heterogeneous agents: CPUs, GPUs, and HWAs
- Main memory interference between CPUs, GPUs, HWAs
- Many timing constraints for various memory types
- Many goals at the same time: performance, fairness, QoS, energy efficiency, ...
Reality and Dream

- **Reality**: It difficult to design a policy that maximizes performance, QoS, energy-efficiency, ...
  - Too many things to think about
  - Continuously changing workload and system behavior

- **Dream**: Wouldn’t it be nice if the DRAM controller automatically found a good scheduling policy on its own?
Self-Optimizing DRAM Controllers

- Problem: DRAM controllers are difficult to design
  - It is difficult for human designers to design a policy that can adapt itself very well to different workloads and different system conditions

- Idea: A memory controller that adapts its scheduling policy to workload behavior and system conditions using machine learning.

- Observation: Reinforcement learning maps nicely to memory control.

- Design: Memory controller is a reinforcement learning agent
  - It dynamically and continuously learns and employs the best scheduling policy to maximize long-term performance.

Goal: Learn to choose actions to maximize $r_0 + \gamma r_1 + \gamma^2 r_2 + \ldots$ ($0 \leq \gamma < 1$)
Self-Optimizing DRAM Controllers

- Dynamically adapt the memory scheduling policy via interaction with the system at runtime
  - Associate system states and actions (commands) with long term reward values: each action at a given state leads to a learned reward
  - Schedule command with highest estimated long-term reward value in each state
  - Continuously update reward values for <state, action> pairs based on feedback from system
Self-Optimizing DRAM Controllers


Figure 4: High-level overview of an RL-based scheduler.
<table>
<thead>
<tr>
<th>Reward function</th>
<th>State attributes</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1 for scheduling Read and Write commands</td>
<td>Number of reads, writes, and load misses in transaction queue</td>
<td>Activate</td>
</tr>
<tr>
<td>0 at all other times</td>
<td>Number of pending writes and ROB heads waiting for referenced row</td>
<td>Write</td>
</tr>
<tr>
<td>Goal is to maximize long-term data bus utilization</td>
<td>Request’s relative ROB order</td>
<td>Read - load miss</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Read - store miss</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Precharge - pending</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Precharge - preemptive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOP</td>
</tr>
</tbody>
</table>
Performance Results

Figure 7: Performance comparison of in-order, FR-FCFS, RL-based, and optimistic memory controllers

Large, robust performance improvements over many human-designed policies

Figure 15: Performance comparison of FR-FCFS and RL-based memory controllers on systems with 6.4GB/s and 12.8GB/s peak DRAM bandwidth
Self Optimizing DRAM Controllers

+ **Continuous learning** in the presence of changing environment

+ **Reduced designer burden** in finding a good scheduling policy. Designer specifies:
  1) What system variables might be useful
  2) What target to optimize, but not how to optimize it

-- How to specify different objectives? (e.g., fairness, QoS, ...)

-- Hardware complexity?

-- Design **mindset** and flow
More on Self-Optimizing DRAM Controllers


Self-Optimizing Memory Controllers: A Reinforcement Learning Approach

Engin İpek\textsuperscript{1,2} Onur Mutlu\textsuperscript{2} José F. Martínez\textsuperscript{1} Rich Caruana\textsuperscript{1}

\textsuperscript{1}Cornell University, Ithaca, NY 14850 USA
\textsuperscript{2}Microsoft Research, Redmond, WA 98052 USA
Self-Optimizing Memory Prefetchers

- Rahul Bera, Konstantinos Kanellopoulos, Anant Nori, Taha Shahroodi, Sreenivas Subramoney, and Onur Mutlu,

"Pythia: A Customizable Hardware Prefetching Framework Using Online Reinforcement Learning"

Proceedings of the 54th International Symposium on Microarchitecture (MICRO), Virtual, October 2021.
[Slides (pptx) (pdf)]
[Short Talk Slides (pptx) (pdf)]
[Lightning Talk Slides (pptx) (pdf)]
[Talk Video (20 minutes)]
[Lightning Talk Video (1.5 minutes)]
[Pythia Source Code (Officially Artifact Evaluated with All Badges)]
[arXiv version]

Pythia: A Customizable Hardware Prefetching Framework Using Online Reinforcement Learning

Rahul Bera$^1$ Konstantinos Kanellopoulos$^1$ Anant V. Nori$^2$ Taha Shahroodi$^{3,1}$
Sreenivas Subramoney$^2$ Onur Mutlu$^1$

$^1$ETH Zürich $^2$Processor Architecture Research Labs, Intel Labs $^3$TU Delft

Pythia
A Customizable Hardware Prefetching Framework Using Online Reinforcement Learning

Rahul Bera, Konstantinos Kanellopoulos, Anant V. Nori, Taha Shahroodi, Sreenivas Subramoney, Onur Mutlu

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

SAFARI
ETH Zürich
Intel
TU Delft
Executive Summary

- **Background**: Prefetchers predict addresses of future memory requests by associating memory access patterns with program context (called **feature**).

- **Problem**: Three key shortcomings of prior prefetchers:
  - Predict mainly using a single program feature
  - Lack inherent system awareness (e.g., memory bandwidth usage)
  - Lack in-silicon customizability

- **Goal**: Design a prefetching framework that:
  - Learns from multiple features and inherent system-level feedback
  - Can be customized in silicon to use different features and/or prefetching objectives

- **Contribution**: Pythia, which formulates prefetching as reinforcement learning problem
  - Takes adaptive prefetch decisions using multiple features and system-level feedback
  - Can be customized in silicon for target workloads via simple configuration registers
  - Proposes a realistic and practical implementation of RL algorithm in hardware

- **Key Results**:
  - Evaluated using a wide range of workloads from SPEC CPU, PARSEC, Ligra, Cloudsuite
  - Outperforms best prefetcher (in 1-core config.) by **3.4%**, **7.7%** and **17%** in 1/4/bw-constrained cores
  - Up to **7.8% more performance** over basic Pythia across Ligra workloads via simple customization

[github.com/CMU-SAFARI/Pythia](https://github.com/CMU-SAFARI/Pythia)
Key Shortcomings in Prior Prefetchers

• We observe **three key shortcomings** that significantly limit performance benefits of prior prefetchers

1. **Predict mainly using a single program feature**

2. **Lack inherent system awareness**

3. **Lack in-silicon customizability**
Our Goal

A **prefetching framework** that can:

1. Learn to prefetch using **multiple features** and **inherent system-level feedback** information

2. Be **easily customized in silicon** to use different features and/or change prefetcher’s objectives
Our Proposal

Pythia

Formulates prefetching as a reinforcement learning problem

Pythia is named after the oracle of Delphi, who is known for her accurate prophecies.
https://en.wikipedia.org/wiki/Pythia
Basics of Reinforcement Learning (RL)

• Algorithmic approach to learn to take an **action** in a given **situation** to maximize a numerical **reward**

![Diagram showing the interaction between an Agent and an Environment]

• Agent stores **Q-values** for **every** state-action pair
  - **Expected return** for taking an action in a state
  - Given a state, selects action that provides **highest** Q-value
Formulating Prefetching as RL

Prefetcher
Processor & Memory Subsystem

Reward

Prefetch from address A+o
ff set (O)

Features of memory request to address A (e.g., PC)
Pythia Overview

- **Q-Value Store**: Records Q-values for all state-action pairs
- **Evaluation Queue**: A FIFO queue of recently-taken actions

![Pythia Diagram]

1. **Assign reward to corresponding EQ entry**
2. **State Vector**
3. **Look up QVStore**
4. **Find the Action with max Q-Value**
5. **Generate prefetch**
6. **Evict EQ entry and update QVStore**
7. **Set filled bit**

**Q-Value Store (QVStore)**

**Evaluation Queue (EQ)**

**Memory Hierarchy**

**Prefetch Fill**

SAFARI
Simulation Methodology

- **Champsim** [3] trace-driven simulator

- **150** single-core memory-intensive workload traces
  - SPEC CPU2006 and CPU2017
  - PARSEC 2.1
  - Ligra
  - Cloudsuite

- Homogeneous and heterogeneous multi-core mixes

- **Five** state-of-the-art prefetchers
  - SPP [Kim+, MICRO’16]
  - Bingo [Bakhshali, HPCA’19]
  - MLOP [Shakerinava+, 3rd Prefetching Championship, 2019]
  - SPP+DSPatch [Bera+, MICRO’19]
  - SPP+PPF [Bhatia+, ISCA’20]

Basic Pythia Configuration

• Derived from automatic design-space exploration

• **State:** 2 features
  - PC+Delta
  - Sequence of last-4 deltas

• **Actions:** 16 prefetch offsets
  - Ranging between -6 to +32. Including 0.

• **Rewards:**
  - $R_{AT} = +20; R_{AL} = +12; R_{NP-H} = -2; R_{NP-L} = -4;$
  - $R_{IN-H} = -14; R_{IN-L} = -8; R_{CL} = -12$
Performance with Varying Core Count

Geomean speedup over no prefetching

Number of cores

Bingo
MLOP
SPP
Pythia

3.4%
7.7%
1. Pythia consistently provides the highest performance in all core configurations.

2. Pythia’s gain increases with core count.
Performance with Varying DRAM Bandwidth

Geomean speedup over no prefetching

DRAM MTPS (in log scale)

- Pythia
- Bingo
- MLOP
- SPP

~Intel Xeon 6258R
~AMD EPYC Rome 7702P
~AMD Threadripper 3990x
Pythia outperforms prior best prefetchers for a wide range of DRAM bandwidth configurations.
Pythia’s Overhead

• **25.5 KB** of total metadata storage per core
  - Only simple tables

• We also model functionally-accurate Pythia with full complexity in **Chisel** [4] HDL

  1. 1.03% area overhead
  2. 0.4% power overhead
  3. Satisfies prediction latency

of a desktop-class 4-core Skylake processor (Xeon D2132IT, 60W)

More in the Paper

- Performance comparison with unseen traces
  - Pythia provides equally high performance benefits

- Comparison against multi-level prefetchers
  - Pythia outperforms prior best multi-level prefetchers

Pythia: A Customizable Hardware Prefetching Framework Using Online Reinforcement Learning

Rahul Bera\textsuperscript{1} Konstantinos Kanellopoulos\textsuperscript{1} Anant V. Nori\textsuperscript{2} Taha Shahroodi\textsuperscript{3,1}
Sreenivas Subramoney\textsuperscript{2} Onur Mutlu\textsuperscript{1}
\textsuperscript{1}ETH Zürich \textsuperscript{2}Processor Architecture Research Labs, Intel Labs \textsuperscript{3}TU Delft


- Performance sensitivity towards different features and hyperparameter values

- Detailed single-core and four-core performance
Pythia is Open Source

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

• MICRO’21 artifact evaluated

• Champsim source code + Chisel modeling code

• All traces used for evaluation
Pythia
A Customizable Hardware Prefetching Framework Using Online Reinforcement Learning

Rahul Bera, Konstantinos Kanellopoulos, Anant V. Nori, Taha Shahroodi, Sreenivas Subramoney, Onur Mutlu

https://github.com/CMU-SAFARI/Pythia
Self-Optimizing Memory Prefetchers

- Rahul Bera, Konstantinos Kanellopoulos, Anant Nori, Taha Shahroodi, Sreenivas Subramoney, and Onur Mutlu,

"Pythia: A Customizable Hardware Prefetching Framework Using Online Reinforcement Learning"
Proceedings of the 54th International Symposium on Microarchitecture (MICRO), Virtual, October 2021.
[Slides (pptx) (pdf)]
[Short Talk Slides (pptx) (pdf)]
[Lightning Talk Slides (pptx) (pdf)]
[Talk Video (20 minutes)]
[Lightning Talk Video (1.5 minutes)]
[Pythia Source Code (Officially Artifact Evaluated with All Badges)]
[arXiv version]

Pythia: A Customizable Hardware Prefetching Framework Using Online Reinforcement Learning

Rahul Bera\textsuperscript{1} Konstantinos Kanellopoulos\textsuperscript{1} Anant V. Nori\textsuperscript{2} Taha Shahroodi\textsuperscript{3,1}
Sreenivas Subramoney\textsuperscript{2} Onur Mutlu\textsuperscript{1}
\textsuperscript{1}ETH Zürich \hspace{1cm} \textsuperscript{2}Processor Architecture Research Labs, Intel Labs \hspace{1cm} \textsuperscript{3}TU Delft

\url{https://arxiv.org/pdf/2109.12021.pdf}
Self-Optimizing Hybrid Storage Systems

- To appear in ISCA 2022

Sibyl: Adaptive and Extensible Data Placement in Hybrid Storage Systems Using Online Reinforcement Learning

Gagandeep Singh\textsuperscript{1} Rakesh Nadig\textsuperscript{1} Jisung Park\textsuperscript{1} Rahul Bera\textsuperscript{1} Nastaran Hajinazar\textsuperscript{1}
David Novo\textsuperscript{3} Juan Gómez-Luna\textsuperscript{1} Sander Stuijk\textsuperscript{2} Henk Corporaal\textsuperscript{2} Onur Mutlu\textsuperscript{1}
\textsuperscript{1}ETH Zürich \textsuperscript{2}Eindhoven University of Technology \textsuperscript{3}LIRMM, Univ. Montpellier, CNRS

Sibyl: Adaptive and Extensible Data Placement in Hybrid Storage Systems Using Online Reinforcement Learning

Gagandeep Singh, Rakesh Nadig, Jisung Park, Rahul Bera, Nastaran Hajinazar, David Novo, Juan Gómez-Luna, Onur Mutlu
**Executive Summary**

**Background:** Hybrid storage systems (HSSs) complement different storage technologies to extend the overall capacity and reduce the system cost with minimal effect on the application performance.

**Problem:** Accurately identify the performance-critical data of an application and placing it in the “best-fit” storage device. Three key shortcomings of prior data placement policies (heuristic-based and supervised learning-based) of hybrid storage systems:
- Lack of **adaptability**
- Lack of **device awareness** (e.g., read/write latencies of each device)
- Lack of **extensibility**

**Goal:** Develop a new, efficient, and high performance data-placement mechanism for hybrid storage systems that can:
- Dynamically derive an adaptive data-placement strategy by **continuously learning and adapting** to the application and underlying device characteristics
- **Easily extensible** to incorporate a wide range of hybrid storage configurations.

**Key Idea:** Sibyl, an online reinforcement learning-based self-optimizing mechanism for data placement that:
- **Dynamically learns** from past experiences and **continuously adapts** its policy to improve long-term performance by interacting with the hybrid storage system
- **Learns the asymmetry in the read/write latencies** present in modern hybrid storage devices while **taking into account** the inherent characteristics of an application

**Key Results:** Sibyl is evaluated on a real system with multiple device configurations
- Evaluated using a **wide range of workloads** from MSR Cambridge and Filebench
- In a performance (cost) optimized hybrid storage configuration, Sibyl provides up to **21.6% (19.9%)** performance improvement compared to prior data placement policies
- On a tri-hybrid storage system, Sibyl outperforms a heuristics-based policy by **23.9% -48.2%**
- Sibyl achieves **80%** performance of an oracle policy with storage overhead of **124.4 KiB**
Hybrid Storage Systems
Key Shortcomings of Prior Data Placement Techniques

We observe three key shortcomings that significantly limit performance benefits of data-placement techniques:

- Lack of adaptability
- Lack of device awareness
- Lack of extensibility
Lack of Adaptability (1/2)

- Prior heuristic-based techniques consider only a few characteristics (e.g., access frequency) to perform data placement.

- **Statically tuned characteristics** (based on fixed thresholds) are ineffective when used on a wide range of applications and system configurations.

- Supervised learning techniques need labeled data and frequent retraining to adapt to varying workloads and system conditions.

Prior techniques offer 41.1% lower performance compared to an Oracle policy.
Lack of Adaptability (2/2)

CDE shows an average performance gap of 41.1% (32.6%) to Oracle for H&M (H&L)

HPS shows an average performance gap of 37.2% (55.5%) to Oracle for H&M (H&L)
Lack of Device Awareness

Prior data placement techniques:

• do not adapt well to changes in underlying device characteristics (e.g., storage read latency)

• do not consider the data migration cost between storage devices while making a data placement decision

• are highly inefficient in hybrid storage systems that have devices with significantly different read/write latencies
Lack of Extensibility

- Prior data placement techniques are typically designed for a hybrid storage system with only two storage devices.

- Significant effort is required to extend the data placement policies for more than two devices.

Compared to a RL-based solution, a heuristic-based policy provides 48.2% lower performance when extended from two to three devices.
Our Goal

A **data-placement mechanism** that can

- dynamically derive an adaptive data-placement strategy by **continuously learning** and **adapting** to the application and underlying device characteristics

- be **easily extended** to incorporate a wide range of hybrid storage configurations
Basics of Reinforcement Learning

- RL is a framework for decision making
  - An **autonomous agent observes** the current **state** of the environment
  - It **interacts** with the environment by taking **actions**
  - Agent is **rewarded** or **penalized** based on the consequences of its actions
  - Agent tries to maximize the cumulative reward
Applying RL to Data Placement

Key factors in applying RL for data placement in a hybrid storage system

- RL agent needs to be aware of:
  - asymmetry in read/write latencies of a storage device
  - differences in latencies across hybrid storage devices
  - application access patterns

- Data placement module should decide which actions to reward and penalize (credit assignment)

- Low implementation overhead
RL State

- Feature selection is performed to select only the most correlated features that affect data placement

- Divide the states into a small number of bins to reduce the state space

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th># of bins</th>
<th>Encoding (bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$size_t$</td>
<td>Size of the requested page (in pages)</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>$type_t$</td>
<td>Type of the current request (read/write)</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>$intr_t$</td>
<td>Access interval of the requested page</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>$cnt_t$</td>
<td>Access count of the requested page</td>
<td>64</td>
<td>8</td>
</tr>
<tr>
<td>$cap_t$</td>
<td>Remaining capacity in the fast storage device</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>$curr_t$</td>
<td>Current placement of the requested page (fast/slow)</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>
Reward

- For every action at time-step $t$, Sibyl gets a reward from the environment at time-step $t + 1$
- **Reward** acts as a feedback to the agent's past action
- **Request latency** faithfully captures the status of the hybrid storage system
- **Penalty** value is chosen to prevent the agent from aggressively servicing all the requests from the faster device

$$R = \begin{cases} 
\frac{1}{L_t} & \text{if no eviction} \\
\max(0, \frac{1}{L_t} - R_p) & \text{if an eviction happens}
\end{cases}$$

$L_t =$ latency of the request
$R_p =$ eviction penalty
Overview of Sibyl

The two threads run asynchronously to prevent training delay from affecting the inference time.
Hyper-parameter Tuning

- Different hyper-parameter configurations were chosen using the design of experiments (DoE) technique

<table>
<thead>
<tr>
<th>Hyper-parameter</th>
<th>Design Space</th>
<th>Chosen Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discount factor ( (\gamma) )</td>
<td>0-1</td>
<td>0.9</td>
</tr>
<tr>
<td>Learning rate ( (\alpha) )</td>
<td>1e^{-5} – 1e^{0}</td>
<td>1e^{-4}</td>
</tr>
<tr>
<td>Exploration rate ( (\epsilon) )</td>
<td>0-1</td>
<td>0.001</td>
</tr>
<tr>
<td>Batch size</td>
<td>64-256</td>
<td>128</td>
</tr>
<tr>
<td>Experience buffer size ( (e_{EB}) )</td>
<td>10-10000</td>
<td>1000</td>
</tr>
</tbody>
</table>
Evaluation Methodology

- Evaluated on a **real system** with different hybrid storage configurations
- Hybrid storage system constitutes one contiguous logical block address space
- A custom block driver was implemented to manage the I/O requests to the storage devices
- We evaluate **three** different hybrid storage configurations
  - Performance-optimized (H&M)
  - Cost-optimized (H&L)
  - Tri-hybrid storage system
## Evaluation Methodology

<table>
<thead>
<tr>
<th>Host System</th>
<th>AMD Ryzen 7 2700G [146], 8-cores@3.5 GHz, 8×64/32 KiB L1-I/D, 4 MiB L2, 8 MiB L3, 16 GiB RDIMM DDR4 2666 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Storage Devices</strong></td>
<td><strong>Characteristics</strong></td>
</tr>
<tr>
<td>H: Intel Optane SSD P4800X [94]</td>
<td>375 GB, PCIe 3.0 NVMe, SLC, R/W: 2.4/2 GB/s, random R/W: 550000/500000 IOPS</td>
</tr>
</tbody>
</table>
| L: Seagate HDD ST1000DM010 [98] | 1 TB, SATA 6Gb/s 7200 RPM  
Max. Sustained Transfer Rate: 210 MB/s                                                                                 |
| $L_{SSD}$: ADATA SU630 SSD [99] | 960 GB, SATA 6 Gb/s, TLC,  
Max R/W: 520/450 MB/s                                                                                           |
| **HSS Configurations**      | **Fast Device**                                              | **Slow Device**                                       |
| H&M (Performance-oriented)  | high-end (H)                                                | middle-end (M)                                        |
| H&L (Cost-oriented)         | high-end (H)                                                | low-end (L)                                           |
Evaluation Methodology

• 18 different workloads from MSR Cambridge and FileBench suites
• Sibyl is compared against four baselines
  - Heuristic-based policies
    • Cold data eviction (CDE) [Matsui et. al., “Design of Hybrid SSDs With Storage Class Memory and NAND Flash Memory,” IEEE 2017]
  - Supervised learning-based policies
    • Recurrent neural network (RNN)-based technique adapted from Kleio [Doudali et.al., “Kleio: A Hybrid Memory Page Scheduler with Machine Intelligence,” HPDC, 2019]
Latency Improvement

(a) Performance-optimized (H&M)

(b) Cost-optimized (H&L)

<table>
<thead>
<tr>
<th>Configuration</th>
<th>CDE</th>
<th>HPS</th>
<th>Archivist</th>
<th>RNN-HSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>H&amp;M</td>
<td>28.1%</td>
<td>23.2%</td>
<td>36.1%</td>
<td>21.6%</td>
</tr>
<tr>
<td>H&amp;L</td>
<td>19.9%</td>
<td>45.9%</td>
<td>68.8%</td>
<td>34.1%</td>
</tr>
</tbody>
</table>
Throughput Improvement

(a) Performance-optimized (H&M)

(b) Cost-optimized (H&L)

<table>
<thead>
<tr>
<th>Configuration</th>
<th>CDE</th>
<th>HPS</th>
<th>Archivist</th>
<th>RNN-HSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>H&amp;M</td>
<td>32.6%</td>
<td>21.9%</td>
<td>54.2%</td>
<td>22.7%</td>
</tr>
<tr>
<td>H&amp;L</td>
<td>22.8%</td>
<td>49.1%</td>
<td>86.9%</td>
<td>41.9%</td>
</tr>
</tbody>
</table>
Latency in Tri-Hybrid System

Sibyl outperforms the heuristic-based data placement policy for tri-hybrid system by 48.2% on average across all workloads.
In H&M (H&L) configurations, Sibyl outperforms RNN-HSS and Archivist by 46.1% (54.6%) and 8.5% (44.1%) respectively.
Sensitivity to Fast Storage Capacity

(a) H&M
(b) H&L
Sibyl consistently provides highest performance by dynamically adapting its data-placement policy.
Overhead Analysis

• **Performance Overhead**
  - *10ns* for every inference on the evaluated system; this is several orders of magnitude less than I/O latency of high-end SSD

• **Implementation Overhead**
  - *124.4 KiB* of implementation overhead

• **Metadata overhead**
  - 0.1% of the total storage capacity when using a 4 KiB data placement granularity
  - *40-bit* metadata overhead per data placement unit
For More on Sybil

- To appear in ISCA 2022

**Sibyl: Adaptive and Extensible Data Placement in Hybrid Storage Systems Using Online Reinforcement Learning**

Gagandeep Singh¹  Rakesh Nadig¹  Jisung Park¹  Rahul Bera¹  Nastaran Hajinazar¹
David Novo³  Juan Gómez-Luna¹  Sander Stuijk²  Henk Corporaal²  Onur Mutlu¹

¹ETH Zürich  ²Eindhoven University of Technology  ³LIRMM, Univ. Montpellier, CNRS

An Intelligent Architecture

- Data-driven
  - Machine learns the “best” policies (how to do things)

- Sophisticated, workload-driven, changing, far-sighted policies

- Automatic data-driven policy learning

- All controllers are intelligent data-driven agents

**We need to rethink design (of all controllers)**
Challenge and Opportunity for Future

Data-Driven
(Self-Optimizing)
Computing Architectures
Data-Aware Architectures
Corollaries: Architectures Today ...

- Architectures are terrible at dealing with data
  - Designed to mainly store and move data vs. to compute
  - They are processor-centric as opposed to data-centric

- Architectures are terrible at taking advantage of vast amounts of data (and metadata) available to them
  - Designed to make simple decisions, ignoring lots of data
  - They make human-driven decisions vs. data-driven decisions

- Architectures are terrible at knowing and exploiting different properties of application data
  - Designed to treat all data as the same
  - They make component-aware decisions vs. data-aware
Data-Aware Architectures

- A data-aware architecture understands what it can do with and to each piece of data.

- It makes use of different properties of data to improve performance, efficiency and other metrics:
  - Compressibility
  - Approximability
  - Locality
  - Sparsity
  - Criticality for Computation X
  - Access Semantics
  - ...
One Problem: Limited Expressiveness

Higher-level information is not visible to HW

Data Structures

Code Optimizations

Access Patterns

Software

Hardware

100011111...
101010011...

Instructions
Memory Addresses

Integer
Float
Data Type
Char
A Solution: More Expressive Interfaces

- **Performance**
  - Software
  - ISA
  - Virtual Memory
- **Functionality**
  - Expressive Memory “XMem”
- **Higher-level Program Semantics**
A Case for Richer Cross-layer Abstractions: Bridging the Semantic Gap with Expressive Memory

Nandita Vijaykumar†§, Abhilasha Jain†, Diptesh Majumdar†, Kevin Hsieh†, Gennady Pekhimenko‡, Eiman Ebrahimi§, Nastaran Hajinazar†, Phillip B. Gibbons†, Onur Mutlu§†

†Carnegie Mellon University  ‡University of Toronto  §ETH Zürich
SW provides key program information to HW

Data Structures

Access Patterns

Data Type/Layout

Integer

Float

Char

Data Placement

Prefetcher

Data Compression
Broader goal: Enable many cross-layer optimizations

Express:
Data structures  
Access semantics  
Data types  
Working set  
Reuse  
Access frequency  
...

Optimizations:
Cache Management  
Data Placement in DRAM  
Data Compression  
Approximation  
DRAM Cache Management  
NVM Management  
NUCA/NUMA Optimizations  
...

Benefits:
More efficient HW:
✓Performance

Reduced SW burden:
✓Programmability  
✓Portability
Our approach: Rich cross-layer abstractions

1. Generality: Enable a wide range of cross-layer approaches
2. Minimize programmer effort
3. Overhead

Approach: Flexibly associate specific semantic information with any data & code
Example: XMem

- **Goal**: convey data semantics to the hardware enables more intelligent management of resources.
- **XMem**: introduces a new HW/SW abstraction, called *Atom*, for conveying data semantics.

## XMem Aids/Enables Many Optimizations

<table>
<thead>
<tr>
<th>Memory optimization</th>
<th>Example semantics provided by XMem (described in §3.3)</th>
<th>Example Benefits of XMem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache management</td>
<td>(i) Distinguishing between data structures or pools of similar data; (ii) Working set size; (iii) Data reuse</td>
<td>Enables: (i) applying different caching policies to different data structures or pools of data; (ii) avoiding cache thrashing by knowing the active working set size; (iii) bypassing/prioritizing data that has no/high reuse. (§5)</td>
</tr>
<tr>
<td>Page placement in DRAM e.g., [23, 24]</td>
<td>(i) Distinguishing between data structures; (ii) Access pattern; (iii) Access intensity</td>
<td>Enables page placement at the data structure granularity to (i) isolate data structures that have high row buffer locality and (ii) spread out concurrently-accessed irregular data structures across banks and channels to improve parallelism. (§6)</td>
</tr>
<tr>
<td>Cache/mem compression e.g., [25–32]</td>
<td>(i) Data type: integer, float, char; (ii) Data properties: sparse, pointer, data index</td>
<td>Enables using a <em>different compression algorithm</em> for each data structure based on data type and data properties, e.g., sparse data encodings, FP-specific compression, delta-based compression for pointers [27].</td>
</tr>
<tr>
<td>Data prefetching e.g., [33–36]</td>
<td>(i) Access pattern: strided, irregular, irregular but repeated (e.g., graphs), access stride; (ii) Data type: index, pointer</td>
<td>Enables (i) highly accurate software-driven prefetching while leveraging the benefits of hardware prefetching (e.g., by being memory bandwidth-aware, avoiding cache thrashing); (ii) using different prefetcher types for different data structures: e.g., stride [33], tile-based [20], pattern-based [34–37], data-based for indices/pointers [38,39], etc.</td>
</tr>
<tr>
<td>DRAM cache management e.g., [40–46]</td>
<td>(i) Access intensity; (ii) Data reuse; (iii) Working set size</td>
<td>(i) Helps avoid cache thrashing by knowing working set size [44]; (ii) Better DRAM cache management via reuse behavior and access intensity information.</td>
</tr>
<tr>
<td>Approximation in memory e.g., [47–53]</td>
<td>(i) Distinguishing between pools of similar data; (ii) Data properties: tolerance towards approximation</td>
<td>Enables (i) each memory component to track how approximable data is (at a fine granularity) to inform approximation techniques; (ii) data placement in heterogeneous reliability memories [54].</td>
</tr>
<tr>
<td>Data placement: NUMA systems e.g., [55, 56]</td>
<td>(i) Data partitioning across threads (i.e., relating data to threads that access it); (ii) Read-Write properties</td>
<td>Reduces the need for profiling or data migration (i) to co-locate data with threads that access it and (ii) to identify Read-Only data, thereby enabling techniques such as replication.</td>
</tr>
<tr>
<td>Data placement: hybrid memories e.g., [16, 57, 58]</td>
<td>(i) Read-Write properties (Read-Only/Read-Write); (ii) Access intensity; (iii) Data structure size; (iv) Access pattern</td>
<td>Avoids the need for profiling/migration of data in hybrid memories to (i) effectively manage the asymmetric read-write properties in NVM (e.g., placing Read-Only data in the NVM) [16, 57]; (ii) make tradeoffs between data structure &quot;hotness&quot; and size to allocate fast/high bandwidth memory [14]; and (iii) leverage row-buffer locality in placement based on access pattern [45].</td>
</tr>
<tr>
<td>Managing NUCA systems e.g., [15, 59]</td>
<td>(i) Distinguishing pools of similar data; (ii) Access intensity; (iii) Read-Write or Private-Shared properties</td>
<td>(i) Enables using different cache policies for different data pools (similar to [15]); (ii) Reduces the need for reactive mechanisms that detect sharing and read-write characteristics to inform cache policies.</td>
</tr>
</tbody>
</table>
A Case for Richer Cross-layer Abstractions: Bridging the Semantic Gap with Expressive Memory

Nandita Vijaykumar†§  Abhilasha Jain†  Diptesh Majumdar†  Kevin Hsieh†  Gennady Pekhimenko‡  
Eiman Ebrahimi₹  Nastaran Hajinazar†  Phillip B. Gibbons†  Onur Mutlu§†

†Carnegie Mellon University  ‡University of Toronto  ₹NVIDIA
+Simon Fraser University  §ETH Zürich
Expressive (Memory) Interfaces for GPUs

- Nandita Vijaykumar, Eiman Ebrahimi, Kevin Hsieh, Phillip B. Gibbons and Onur Mutlu, "The Locality Descriptor: A Holistic Cross-Layer Abstraction to Express Data Locality in GPUs"
  *Proceedings of the 45th International Symposium on Computer Architecture (ISCA), Los Angeles, CA, USA, June 2018.*
  [Slides (pptx) (pdf)] [Lightning Talk Slides (pptx) (pdf)] [Lightning Talk Video]

---

The Locality Descriptor:
A Holistic Cross-Layer Abstraction to Express Data Locality in GPUs

Nandita Vijaykumar†§ Eiman Ebrahimi‡§ Kevin Hsieh†
Phillip B. Gibbons† Onur Mutlu§†

†Carnegie Mellon University ‡NVIDIA §ETH Zürich
Exploiting data locality in GPUs is a challenging task

**Locality Descriptor: Executive Summary**

Flexible, architecture-agnostic interface

**Performance Benefits:**
- 26.6% (up to 46.6%) from cache locality
- 53.7% (up to 2.8x) from NUMA locality
An Example: Hybrid Memory Management

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.
An Example: 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]
Exploiting Memory Error 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

- Reliable memory
- Parity memory + software recovery (Par+R)
- Low-cost memory
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]
Data-Aware Cross-Layer Hybrid System Management

- Heterogeneous agents: CPUs, GPUs, and HWAs
- Main memory interference between CPUs, GPUs, HWAs
- Many timing constraints for various memory types
- Many goals at the same time: performance, fairness, QoS, energy efficiency, ...
Another Example: EDEN for DNNs

- Deep Neural Network evaluation is very DRAM-intensive (especially for large networks)

1. Some data and layers in DNNs are very tolerant to errors

2. Reduce DRAM latency and voltage on such data and layers

3. While still achieving a user-specified DNN accuracy target by making training DRAM-error-aware

Data-aware management of DRAM latency and voltage for Deep Neural Network Inference
Example DNN Data Type to DRAM Mapping

Mapping example of ResNet-50:

Map more error-tolerant DNN layers to DRAM partitions with lower voltage/latency

4 DRAM partitions with different error rates
Key idea: Enable **accurate, efficient** DNN inference using approximate DRAM

**EDEN** is an **iterative** process that has **3 key steps**
CPU: DRAM Energy Evaluation

Average 21% DRAM energy reduction maintaining accuracy within 1% of original
CPU: Performance Evaluation

Average 8% system speedup
Some workloads achieve 17% speedup

EDEN achieves close to the ideal speedup possible via tRCD scaling
GPU, Eyeriss, and TPU: Energy Evaluation

- **GPU**: average 37% energy reduction
- **Eyeriss**: average 31% energy reduction
- **TPU**: average 32% energy reduction
EDEN: Data-Aware Efficient DNN Inference


Proceedings of the 52nd International Symposium on Microarchitecture (MICRO), Columbus, OH, USA, October 2019.

[Lightning Talk Slides (pptx) (pdf)]
[Lightning Talk Video (90 seconds)]

---

EDEN: Enabling Energy-Efficient, High-Performance Deep Neural Network Inference Using Approximate DRAM

Skanda Koppula  Lois Orosa  A. Giray Yaglikci
Roknoddin Azizi  Taha Shahroodi  Konstantinos Kanellopoulos  Onur Mutlu

ETH Zürich
SMASH: SW/HW Indexing Acceleration

Konstantinos Kanellopoulos, Nandita Vijaykumar, Christina Giannoula, Roknoddin Azizi, Skanda Koppula, Nika Mansouri Ghiasi, Taha Shahroodi, Juan Gomez-Luna, and Onur Mutlu,
"SMASH: Co-designing Software Compression and Hardware-Accelerated Indexing for Efficient Sparse Matrix Operations"
Proceedings of the 52nd International Symposium on Microarchitecture (MICRO), Columbus, OH, USA, October 2019.
[Slides (pptx) (pdf)]
[Lightning Talk Slides (pptx) (pdf)]
[Poster (pptx) (pdf)]
[Lightning Talk Video (90 seconds)]
[Full Talk Lecture (30 minutes)]
Data-Aware Virtual Memory Framework

Nastaran Hajinazar, Pratyush Patel, Minesh Patel, Konstantinos Kanellopoulos, Saugata Ghose, Rachata Ausavarungnirun, Geraldo Francisco de Oliveira Jr., Jonathan Appavoo, Vivek Seshadri, and Onur Mutlu,
"The Virtual Block Interface: A Flexible Alternative to the Conventional Virtual Memory Framework"


[Slides (pptx) (pdf)]
[Lightning Talk Slides (pptx) (pdf)]
[ARM Research Summit Poster (pptx) (pdf)]
[Talk Video (26 minutes)]
[Lightning Talk Video (3 minutes)]
[Lecture Video (43 minutes)]

Nominated for the Stamatis Vassiliadis Memorial Award.
NATSA: A Near-Data Processing Accelerator for Time Series Analysis

Ivan Fernandez§, Ricardo Quislant§, Christina Giannoula†, Mohammed Alser†, Juan Gómez-Luna‡, Eladio Gutiérrez§, Oscar Plata§, and Onur Mutlu‡,

"NATSA: A Near-Data Processing Accelerator for Time Series Analysis"

[Slides (pptx) (pdf)]
[Talk Video (10 minutes)]
[Source Code]
FPGA-based Processing Near Memory


FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications

Gagandeep Singh♦ Mohammed Alser♦ Damla Senol Cali♦
Dionysios Diamantopoulos▽ Juan Gómez-Luna♦
Henk Corporaal* Onur Mutlu♦▽

♦ETH Zürich ▽Carnegie Mellon University
*Eindhoven University of Technology ▼IBM Research Europe
Accelerating Linked Data Structures

Kevin Hsieh, Samira Khan, Nandita Vijaykumar, Kevin K. Chang, Amirali Boroumand, Saugata Ghose, and Onur Mutlu,
"Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation"
Proceedings of the 34th IEEE International Conference on Computer Design (ICCD), Phoenix, AZ, USA, October 2016.

Accelerating Pointer Chasing in 3D-Stacked Memory: Challenges, Mechanisms, Evaluation

Kevin Hsieh† Samira Khan‡ Nandita Vijaykumar†
Kevin K. Chang† Amirali Boroumand† Saugata Ghose† Onur Mutlu§†
†Carnegie Mellon University ‡University of Virginia §ETH Zürich
Accelerating Approximate String Matching


[Lighting Talk Video (1.5 minutes)]
[Lightning Talk Slides (pptx) (pdf)]
[Talk Video (18 minutes)]
[Slides (pptx) (pdf)]
Mohammed Alser, Zulal Bingol, Damla Senol Cali, Jeremie Kim, Saugata Ghose, Can Alkan, and Onur Mutlu,
"Accelerating Genome Analysis: A Primer on an Ongoing Journey"
[Slides (pptx)(pdf)]
[Talk Video (1 hour 2 minutes)]
Challenge and Opportunity for Future

Data-Aware (Expressive) Computing Architectures
We Need to **Rethink** the Entire Stack

<table>
<thead>
<tr>
<th>Problem</th>
<th>Algorithm</th>
<th>Program/Language</th>
<th>System Software</th>
<th>SW/HW Interface</th>
<th>Micro-architecture</th>
<th>Logic</th>
<th>Devices</th>
<th>Electrons</th>
</tr>
</thead>
</table>

**We can get there case by case**
Principled Architectures &
What They Can Enable
A Quote from A Famous Architect

“architecture [...] based upon principle, and not upon precedent”
Precedent-Based Design?

- “architecture [...] based upon principle, and not upon precedent”
Principled Design

“architecture [...] based upon principle, and not upon precedent”
The Overarching Principle

Organic architecture

From Wikipedia, the free encyclopedia

Organic architecture is a philosophy of architecture which promotes harmony between human habitation and the natural world through design approaches so sympathetic and well integrated with its site, that buildings, furnishings, and surroundings become part of a unified, interrelated composition.

A well-known example of organic architecture is Fallingwater, the residence Frank Lloyd Wright designed for the Kaufmann family in rural Pennsylvania. Wright had many choices to locate a home on this large site, but chose to place the home directly over the waterfall and creek creating a close, yet noisy dialog with the rushing water and the steep site. The horizontal striations of stone masonry with daring cantilevers of colored beige concrete blend with native rock outcroppings and the wooded environment.
Another Example: Precedent-Based Design

Source: http://cookiemagik.deviantart.com/art/Train-station-207266944
Principled Design

Source: By Toni_V, CC BY-SA 2.0, https://commons.wikimedia.org/w/index.php?curid=4087256
Another Principled Design
Another Principled Design
Principle Applied to Another Structure


Source: By 准建築人手札網站 Forgemind ArchiMedia - Flickr: IMG_2489 JPG, CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=31493356
The Overarching Principle

Zoomorphic architecture

From Wikipedia, the free encyclopedia

Zoomorphic architecture is the practice of using animal forms as the inspirational basis and blueprint for architectural design. "While animal forms have always played a role adding some of the deepest layers of meaning in architecture, it is now becoming evident that a new strand of biomorphism is emerging where the meaning derives not from any specific representation but from a more general allusion to biological processes."[1]

Some well-known examples of Zoomorphic architecture can be found in the TWA Flight Center building in New York City, by Eero Saarinen, or the Milwaukee Art Museum by Santiago Calatrava, both inspired by the form of a bird’s wings.[3]
Overarching Principles for Computing?

Source: http://spectrum.ieee.org/image/MjYzMzAyMg.jpeg
Readings, Videos, Reference Materials
More on My Research & Teaching
Brief Self Introduction

Onur Mutlu

- Full Professor @ ETH Zurich ITET (INFK), 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 Mission

Computer architecture, HW/SW, systems, bioinformatics, security

Build fundamentally better architectures
Four Key Current Directions

- Fundamentally Secure/Reliable/Safe Architectures
- Fundamentally Energy-Efficient Architectures
  - Memory-centric (Data-centric) Architectures
- Fundamentally Low-Latency and Predictable Architectures
- Architectures for AI/ML, Genomics, Medicine, Health
The Transformation Hierarchy

Computer Architecture (expanded view)

- Problem
- Algorithm
- Program/Language
- System Software
- SW/HW Interface
- Micro-architecture
- Logic
- Devices
- Electrons

Computer Architecture (narrow view)
Axiom

To achieve the highest energy efficiency and performance:

we must take the expanded view of computer architecture

Co-design across the hierarchy: Algorithms to devices

Specialize as much as possible within the design goals
Current Research Mission & Major Topics

Build fundamentally better architectures

- Data-centric arch. for low energy & high perf.
  - Proc. in Mem/DRAM, NVM, unified mem/storage
- Low-latency & predictable architectures
  - Low-latency, low-energy yet low-cost memory
  - QoS-aware and predictable memory systems
- Fundamentally secure/reliable/safe arch.
  - Tolerating all bit flips; patchable HW; secure mem
- Architectures for ML/AI/Genomics/Health/Med
  - Algorithm/arch./logic co-design; full heterogeneity
- Data-driven and data-aware architectures
  - ML/AI-driven architectural controllers and design
  - Expressive memory and expressive systems

Broad research spanning apps, systems, logic with architecture at the center
Onur Mutlu’s SAFARI Research Group

Computer architecture, HW/SW, systems, bioinformatics, security, memory

https://safari.ethz.ch/safari-newsletter-april-2020/

Think BIG, Aim HIGH!

SAFARI

https://safari.ethz.ch
Dear SAFARI friends,

2019 and the first three months of 2020 have been very positive eventful times for SAFARI.
Dear SAFARI friends,

Happy New Year! We are excited to share our group highlights with you in this second edition of the SAFARI newsletter (You can find the first edition from April 2020 here). 2020 has...
SAFARI Newsletter December 2021 Edition

https://safari.ethz.ch/safari-newsletter-december-2021/
Papers, Talks, Artifacts

- All are available at

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

  https://www.youtube.com/onurmutlulectures

  https://github.com/CMU-SAFARI/
Open Source Tools: SAFARI GitHub

SAFARI Research Group at ETH Zurich and Carnegie Mellon University
Site for source code and tools distribution from SAFARI Research Group at ETH Zurich and Carnegie Mellon University.

- ETH Zurich and Carnegie Mellon ...
- https://safari.ethz.ch/
- omuvtlu@gmail.com

Pinned

- **ramulator**
  A Fast and Extensible DRAM Simulator, with built-in support for modeling many different DRAM technologies including DDRx, LPDDRx, GDDRx, WIOx, HBMx, and various academic proposals. Described in the...
  - C++
  - 250 stars
  - 130 forks

- **prim-benchmarks**
  PrIM (Processing-In-Memory benchmarks) is the first benchmark suite for a real-world processing-in-memory (PIM) architecture. PrIM is developed to evaluate, analyze, and characterize the first publ...
  - C
  - 18 stars
  - 8 forks

- **DAMOV**
  DAMOV is a benchmark suite and a methodical framework targeting the study of data movement bottlenecks in modern applications. It is intended to study new architectures, such as near-data processin...
  - C++
  - 12 stars
  - 1 fork

Repositories

- **Pythia**
  A Customizable Hardware Prefetching Framework Using Online Reinforcement Learning.
  - C++
  - 0 stars
  - 1 forks
  - 0 watchers
  - Updated yesterday

- **BurstLink**
  - 0 stars
  - 0 forks
  - 0 watchers
  - Updated 21 days ago

[https://github.com/CMU-SAFARI/](https://github.com/CMU-SAFARI/)
SAFARI PhD and Post-Doc Alumni

- [https://safari.ethz.ch/safari-alumni/](https://safari.ethz.ch/safari-alumni/)

- Minesh Patel (ETH Zurich), MICRO 2020 and DSN 2020 Best Paper Awards; ISCA Hall of Fame 2021
- Damla Senol Cali (Bionano Genomics), SRC TECHCON 2019 Best Student Presentation Award
- Nastaran Hajinazar (ETH Zurich)
- Gagandeep Singh (ETH Zurich), FPL 2020 Best Paper Award Finalist
- Amirali Boroumand (Stanford Univ → Google), SRC TECHCON 2018 Best Student Presentation Award
- Jeremie Kim (ETH Zurich), EDAA Outstanding Dissertation Award 2020; IEEE Micro Top Picks 2019; ISCA/MICRO HoF 2021
- Nandita Vijaykumar (Univ. of Toronto, Assistant Professor), ISCA Hall of Fame 2021
- Kevin Hsieh (Microsoft Research, Senior Researcher)
- Justin Meza (Facebook), HiPEAC 2015 Best Student Presentation Award; ICCD 2012 Best Paper Award
- Mohammed Alser (ETH Zurich), IEEE Turkey Best PhD Thesis Award 2018
- Yixin Luo (Google), HPCA 2015 Best Paper Session
- Kevin Chang (Facebook), SRC TECHCON 2016 Best Student Presentation Award
- Rachata Ausavarungnirun (KMUNTB, Assistant Professor), NOCS 2015 and NOCS 2012 Best Paper Award Finalist
- Gennady Pekhimenko (Univ. of Toronto, Assistant Professor), ISCA Hall of Fame 2021; ASPLOS 2015 SRC Winner
- Vivek Seshadri (Microsoft Research)
- Donghyuk Lee (NVIDIA Research, Senior Researcher), HPCA Hall of Fame 2018
- Yoongu Kim (Software Robotics → Google), TCAD’19 Top Pick Award; IEEE Micro Top Picks’10; HPCA’10 Best Paper Session
- Lavanya Subramanian (Intel Labs → Facebook)

- Samira Khan (Univ. of Virginia, Assistant Professor), HPCA 2014 Best Paper Session
- Saugata Ghose (Univ. of Illinois, Assistant Professor), DFRWS-EU 2017 Best Paper Award
- Jawad Haj-Yahya (Huawei Research Zurich, Principal Researcher)
Principle: Teaching and Research

Teaching drives Research
Research drives Teaching
Focus on learning and scholarship
Principle: Insight and Ideas

Focus on Insight
Encourage New Ideas
Principle: Learning and Scholarship

The quality of your work defines your impact
You can make a good impact on the world
Research & Teaching: Some Overview Talks

https://www.youtube.com/onurmutlulectures

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

- Enabling In-Memory Computation
  - https://www.youtube.com/watch?v=njX_14584Jw&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=16

- Accelerating Genome Analysis
  - https://www.youtube.com/watch?v=r7sn41Ih-4A&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=41

- Rethinking Memory System Design
  - https://www.youtube.com/watch?v=F7xZLNMIY1E&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=3

- Intelligent Architectures for Intelligent Machines
  - https://www.youtube.com/watch?v=c6_LqzuNdkw&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=25

- The Story of RowHammer
  - https://www.youtube.com/watch?v=sgd7PHQQ1AI&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=39
Online Courses & Lectures

- **First Computer Architecture & Digital Design Course**
  - Digital Design and Computer Architecture
  - **Spring 2021 Livestream** Edition: [https://www.youtube.com/watch?v=LbC0EZY8yw4&list=PL5Q2soXY2Zi_uej3aY39YB5pfW4SJ7LiN](https://www.youtube.com/watch?v=LbC0EZY8yw4&list=PL5Q2soXY2Zi_uej3aY39YB5pfW4SJ7LiN)

- **Advanced Computer Architecture Course**
  - Computer Architecture
  - **Fall 2021 Livestream** Edition: [https://www.youtube.com/watch?v=c3mPdZA-Fmc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN](https://www.youtube.com/watch?v=c3mPdZA-Fmc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN)
  - **Fall 2020** Edition: [https://www.youtube.com/watch?v=4yfkM_5EFgo&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAGILkTOF](https://www.youtube.com/watch?v=4yfkM_5EFgo&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAGILkTOF)

SAFARI [https://www.youtube.com/onurmutlulectures](https://www.youtube.com/onurmutlulectures)
Bachelor’s course
- 2nd semester at ETH Zurich
- Rigorous introduction into “How Computers Work”
- Digital Design/Logic
- Computer Architecture
- 10 FPGA Lab Assignments
Comp Arch (Fall 2020)

- [https://www.youtube.com/watch?v=c3mPdZA-Fmc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN](https://www.youtube.com/watch?v=c3mPdZA-Fmc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN)

- Master’s level course
  - Taken by Bachelor’s/Masters/PhD students
  - Cutting-edge research topics + fundamentals in Computer Architecture
  - 5 Simulator-based Lab Assignments
  - Potential research exploration
  - Many research readings
Comp Arch (Current)

- [link](https://safari.ethz.ch/architecture/fall2021/doku.php?id=schedule)

**Youtube Livestream:**
- [link](https://www.youtube.com/watch?v=4yfkM_5EFgo&list=PL5Q2soXYZi-Mnk1PxjEIG32HAGILkTOF)

- Master’s level course
  - Taken by Bachelor’s/Masters/PhD students
  - Cutting-edge research topics + fundamentals in Computer Architecture
  - 5 Simulator-based Lab Assignments
  - Potential research exploration
  - Many research readings
Seminar (Spring’21)

- [https://www.youtube.com/watch?v=t3m93ZpLOyw&list=PL5Q2soXY2Zi_awYdjmWVIUegsbY7TPGW4](https://www.youtube.com/watch?v=t3m93ZpLOyw&list=PL5Q2soXY2Zi_awYdjmWVIUegsbY7TPGW4)

- Critical analysis course
  - Taken by Bachelor’s/Masters/PhD students
  - Cutting-edge research topics + fundamentals in Computer Architecture
  - 20+ research papers, presentations, analyses
Seminar (Current)

- [https://safari.ethz.ch/architecture_seminar/fall2021/doku.php?id=schedule](https://safari.ethz.ch/architecture_seminar/fall2021/doku.php?id=schedule)

- **Youtube Livestream:**
  - [https://www.youtube.com/watch?v=4TcP297mdsI&list=PL5Q2soXY2Zi_7UBNmC9B8Yr5JSwTG9yH4](https://www.youtube.com/watch?v=4TcP297mdsI&list=PL5Q2soXY2Zi_7UBNmC9B8Yr5JSwTG9yH4)

- Critical analysis course
  - Taken by Bachelor’s/Masters/PhD students
  - Cutting-edge research topics + fundamentals in Computer Architecture
  - 20+ research papers, presentations, analyses
Hands-On Projects & Seminars Courses

- https://safari.ethz.ch/projects_and_seminars/doku.php

SAFARI Project & Seminars Courses (Spring 2021)

Welcome to the wiki for Project and Seminar courses SAFARI offers.

Courses we offer:

- Understanding and Improving Modern DRAM Performance, Reliability, and Security with Hands-On Experiments
- Designing and Evaluating Memory Systems and Modern Software Workloads with Ramulator
- Accelerating Genome Analysis with FPGAs, GPUs, and New Execution Paradigms
- Genome Sequencing on Mobile Devices
- Exploring the Processing-in-Memory Paradigm for Future Computing Systems
- Hands-on Acceleration on Heterogeneous Computing Systems
- Understanding and Designing Modern NAND Flash-Based Solid-State Drives (SSDs) by Building a Practical SSD Simulator
PIM Course (Current)

- **Fall 2021 Edition:**

- **Youtube Livestream:**
  - [https://www.youtube.com/watch?v=9e4Chnwdovo&list=PL5Q2soXY2Zi-841fUYUK9EsXKhQKRPyX](https://www.youtube.com/watch?v=9e4Chnwdovo&list=PL5Q2soXY2Zi-841fUYUK9EsXKhQKRPyX)

- **Project course**
  - Taken by Bachelor’s/Master’s students
  - Processing-in-Memory lectures
  - Hands-on research exploration
  - Many research readings
SAFARI Live Seminars (1)

SAFARI Live Seminars in Computer Architecture
Dr. Juan Gómez Luna, ETH Zurich
Understanding a Modern Processing-in-Memory Architecture: Benchmarking and Experimental Characterization

SAFARI Live Seminars in Computer Architecture
Dr. Andrew Walker, Schiltron Corporation & Nexgen Power Systems
An Addiction to Low Cost Per Memory Bit - How to Recognize it and What to Do About It

SAFARI Live Seminars in Computer Architecture
Geraldo F. Oliveira, ETH Zurich
DAMON: A New Methodology and Benchmark Suite for Evaluating Data Movement Bottlenecks

SAFARI Live Seminars in Computer Architecture
Gennady Pakhmuro, University of Toronto
Efficient DNN Training at Scale: from Algorithms to Hardware

SAFARI Live Seminars in Computer Architecture
Jawad Haj-Yahya, Huawei Research Center Zurich
Power Management Mechanisms in Modern Microprocessors and Their Security Implications

SAFARI Live Seminars in Computer Architecture
Ataberk Olgun, TOBB & ETH Zurich
QIAC-FRD: High-Throughput True Random Number Generation Using Quadruple Row Activation in Commercial DRAM Chips

SAFARI Live Seminars in Computer Architecture
Minas Patel, ETH Zurich
Enabling Effective Error Mitigation in Memory Chips That Use On-Die ECCs

SAFARI Live Seminars in Computer Architecture
Christina Giannous, National Technical University of Athens
Efficient Synchronization: Support for Near-Data-Processing Architectures

SAFARI Live Seminars in Computer Architecture
Jawad Haj-Yahya, Huawei Research Center Zurich

https://safari.ethz.ch/safari-seminar-series/
SAFARI Live Seminars (II)

SAFARI Live Seminars in Computer Architecture
Nastaran Hajinazar, ETH Zurich
Data Centric and Data-Aware Frameworks for Fundamentally Efficient Data Handling in Modern Computing Systems
Overview of Our Approach
Data and the efficient computation of data should be the ultimate priority of the system
• Data-Centric Architectures
  - Enable computation with minimal data movement
  - Compute where data resides
• Data-Aware Architectures
  - Understand what they can do with and to each piece of data
  - Make use of different properties of data to improve performance, efficiency, etc.

SAFARI Live Seminar: Nastaran Hajinazar 27 Oct 2021
Posted on October 1, 2021 by event

Join us for our SAFARI Live Seminar with Nastaran Hajinazar.
Wednesday, October 27 at 7:00 pm Zurich time (CET)

SAFARI Live Seminars in Computer Architecture
Damla Senol Cali, Bionano Genomics
Accelerating Genome Sequence Analysis via Efficient Hardware/Algorithm Co-Design
Our Goal & Approach
• Our Goal:
  Accelerating genome sequence analysis by efficient hardware/algorithm co-design
• Our Approach:
  (a) Analyze the multiple steps and the associated tools in the genome sequence analysis pipeline,
  (b) Expose the tradeoffs between accuracy, performance, memory usage and scalability, and
  (c) Co-design fast and efficient algorithms along with scalable and energy-efficient customized hardware accelerators for the key bottleneck steps of the pipeline

SAFARI Live Seminar: Damla Senol Cali 07 Nov 2021
Posted on October 18, 2021 by event

Join us for our SAFARI Live Seminar with Damla Senol Cali.
Sunday, November 07 at 6:00 pm Zurich time (CET)

SAFARI Live Seminars in Computer Architecture
Gennady Pekhimenko, University of Toronto
Machine Learning Tools in Action
SAFARI Research Group
SAFARI

SAFARI Live Seminar: Gennady Pekhimenko 08 Nov 2021
Posted on November 1, 2021 by event

Join us for our SAFARI Live Seminar with Gennady Pekhimenko.
Monday, November 08 at 4:00 pm Zurich time (CET)

SAFARI Live Seminars in Computer Architecture
Sergei Mangul, Mangul Lab, USC
Opportunities and challenges of computational data-driven immunology

SAFARI Live Seminar: Sergei Mangul 11 Nov 2021
Posted on November 5, 2021 by event

Join us for our SAFARI Live Seminar with Sergei Mangul.
Thursday, November 11 at 11:00 am Zurich time (CET), ETH Zentrum ETZ K91

https://www.youtube.com/watch?v=D8Hjy2iU9l4&list=PL5Q2soXY2 Zi_tOTAYm--dYByNPL7JhwR9&index=1
Open-Source Artifacts

https://github.com/CMU-SAFARI
Open Source Tools: SAFARI GitHub

SAFARI Research Group at ETH Zurich and Carnegie Mellon University
Site for source code and tools distribution from SAFARI Research Group at ETH Zurich and Carnegie Mellon University.

ETH Zurich and Carnegie Mellon ...  🌐 https://safari.ethz.ch/  📧 omutlu@gmail.com

Repositories 55  Packages  People 40  Teams 1  Projects 1  Settings

Pinned

- **ramulator**  🔄 Public  ⚙️
  A Fast and Extensible DRAM Simulator, with built-in support for modeling many different DRAM technologies including DDRx, LPDDRx, GDDRx, WIOx, HBMx, and various academic proposals. Described in the...
  🌟 C++  ⭐️ 250  🏼 130

- **prim-benchmarks**  🔄 Public  ⚙️
  PrIM (Processing-In-Memory benchmarks) is the first benchmark suite for a real-world processing-in-memory (PIM) architecture. PrIM is developed to evaluate, analyze, and characterize the first publi...
  🌟 C  ⭐️ 18  🏼 8

- **DAMOV**  🔄 Public  ⚙️
  DAMOV is a benchmark suite and a methodical framework targeting the study of data movement bottlenecks in modern applications. It is intended to study new architectures, such as near-data processin...
  🌟 C++  ⭐️ 12  🏼 1

Repositories

Find a repository...

- **Pythia**
  A Customizable Hardware Prefetching Framework Using Online Reinforcement Learning.
  🌟 C++  ⭐️ 0  🏼 1  ⏰ 0  Updated yesterday

- **BurstLink**
  🌟 0  🏼 0  ⏰ 0  Updated 21 days ago

[https://github.com/CMU-SAFARI/](https://github.com/CMU-SAFARI/)
SAFARI Research Group at ETH Zurich and Carnegie Mellon University
Site for source code and tools distribution from SAFARI Research Group at ETH Zurich and Carnegie Mellon University.

- ETH Zurich and Carnegie Mellon ...
- https://safari.ethz.ch/
- omulu@gmail.com

Repositories 45 Packages People 12 Projects

Find a repository...

COVIDHunter
COVIDHunter is an accurate and flexible COVID-19 outbreak simulation model that forecasts the strength of future mitigation measures and the numbers of cases, hospitalizations, and deaths for a given day, while considering the potential effect of environmental conditions. Described by Alser et al. (preliminary version at https://arxiv.org/abs/2...)

Top languages
- C++
- C
- C#
- AGS Script
- VHDL

Most used topics
- dram
- reliability
- error-correcting-codes
- experimental-data
- pre-alignment-filtering

SNP-Selective-Hiding
An optimization-based mechanism to selectively hide the minimum number of overlapping SNPs among the family members who participated in the genomic studies (i.e. GWAS). Our goal is to distort the dependencies among the family members in the original database for achieving better privacy without significantly degrading the data utility.

People

SneakySnake
SneakySnake is the first and only pre-alignment filtering algorithm that works efficiently and fast on modern CPU, FPGA, and GPU architectures. It greatly (by more than two orders of magnitude) expedites sequence alignment calculation for both short and long reads. Described in the Bioinformatics (2020) by Alser et al. https://arxiv.org/abs/...

ramulator
A Fast and Extensible DRAM Simulator, with built-in support for modeling many different DRAM technologies including DDR4, LPDDR4x, GDDR5x, WIOx, HBMx, and various academic proposals. Described in the IEEE CAL 2015 paper by Kim et al. at http://users.ece.cmu.edu/~omulu/pub/ramulator_dram_simulator-ieee-cal15.pdf

https://github.com/CMU-SAFARI
An Interview on Research and Education

- **Computing Research and Education (@ ISCA 2019)**
  - [https://www.youtube.com/watch?v=8ffSEKZhmvo&list=PL5Q2soXY2Zi_4oP9LdL3cc8G6NIjD2Ydz](https://www.youtube.com/watch?v=8ffSEKZhmvo&list=PL5Q2soXY2Zi_4oP9LdL3cc8G6NIjD2Ydz)

- **Maurice Wilkes Award Speech (10 minutes)**
  - [https://www.youtube.com/watch?v=tcQ3zZ3JpuA&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=15](https://www.youtube.com/watch?v=tcQ3zZ3JpuA&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=15)
More Thoughts and Suggestions

- Onur Mutlu, "Some Reflections (on DRAM)"
  Award Speech for ACM SIGARCH Maurice Wilkes Award, at the ISCA Awards Ceremony, Phoenix, AZ, USA, 25 June 2019.
  [Slides (pptx) (pdf)]
  [Video of Award Acceptance Speech (Youtube; 10 minutes) (Youku; 13 minutes)]
  [Video of Interview after Award Acceptance (Youtube; 1 hour 6 minutes) (Youku; 1 hour 6 minutes)]
  [News Article on "ACM SIGARCH Maurice Wilkes Award goes to Prof. Onur Mutlu"]

- Onur Mutlu, "How to Build an Impactful Research Group"
  57th Design Automation Conference Early Career Workshop (DAC), Virtual, 19 July 2020.
  [Slides (pptx) (pdf)]
More Thoughts and Suggestions (II)

- Onur Mutlu,
  "Computer Architecture: Why Is It So Important and Exciting Today?"
  Invited Lecture at Immir Institute of Technology (IYTE), Virtual, 16 October 2020.
  [Slides (pptx) (pdf)]
  [Talk Video (2 hours 12 minutes)]

- Onur Mutlu,
  "Applying to Graduate School & Doing Impactful Research"
  Invited Panel Talk at the 3rd Undergraduate Mentoring Workshop, held with the 48th International Symposium on Computer Architecture (ISCA), Virtual, 18 June 2021.
  [Slides (pptx) (pdf)]
  [Talk Video (50 minutes)]
A Talk on Impactful Research & Teaching

Applying to Grad School & Doing Impactful Research

Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu
13 June 2020
Undergraduate Architecture Mentoring Workshop @ ISCA 2021

Panel talk at Undergraduate Architecture Mentoring Workshop at ISCA 2021
(https://sites.google.com/wisc.edu/uar...)
An Interview on Computing Futures

Interview with Onur Mutlu @ ISCA 2019 on computing research & education (after Maurice Wilkes Award)

6,749 views • Oct 19, 2019

https://www.youtube.com/watch?v=8ffSEKZhmvo
Papers, Talks, Videos, Artifacts

- All are available at

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

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

  https://www.youtube.com/onurmutlulectures

  https://github.com/CMU-SAFARI/
Fundamental Thinking
"There's Plenty of Room at the Bottom: An Invitation to Enter a New Field of Physics" was a lecture given by physicist Richard Feynman at the annual American Physical Society meeting at Caltech on December 29, 1959. Feynman considered the possibility of direct manipulation of individual atoms as a more powerful form of synthetic chemistry than those used at the time. Although versions of the talk were reprinted in a few popular magazines, it went largely unnoticed and did not inspire the conceptual beginnings of the field. Beginning in the 1980s, nanotechnology advocates cited it to establish the scientific credibility of their work.
Feynman considered some ramifications of a general ability to manipulate matter on an atomic scale. He was particularly interested in the possibilities of denser computer circuitry, and microscopes that could see things much smaller than is possible with scanning electron microscopes. These ideas were later realized by the use of the scanning tunneling microscope, the atomic force microscope and other examples of scanning probe microscopy and storage systems such as Millipede, created by researchers at IBM.

Feynman also suggested that it should be possible, in principle, to make nanoscale machines that "arrange the atoms the way we want", and do chemical synthesis by mechanical manipulation.

He also presented the possibility of "swallowing the doctor", an idea that he credited in the essay to his friend and graduate student Albert Hibbs. This concept involved building a tiny, swallowable surgical robot.
Much of the improvement in computer performance comes from decades of miniaturization of computer components, a trend that was foreseen by the Nobel Prize–winning physicist Richard Feynman in his 1959 address, “There’s Plenty of Room at the Bottom,” to the American Physical Society. In 1975, Intel founder Gordon Moore predicted the regularity of this miniaturization trend, now called Moore’s law, which, until recently, doubled the number of transistors on computer chips every 2 years.

Unfortunately, semiconductor miniaturization is running out of steam as a viable way to grow computer performance—there isn’t much more room at the “Bottom.” If growth in computing power stalls, practically all industries will face challenges to their productivity. Nevertheless, opportunities for growth in computing performance will still be available, especially at the “Top” of the computing-technology stack: software, algorithms, and hardware architecture.
Axiom, Revisited

There is plenty of room both at the top and at the bottom

but much more so

when you

communicate well between and optimize across

the top and the bottom
Hence the Expanded View

Computer Architecture (expanded view)

- Problem
- Algorithm
- Program/Language
- System Software
- SW/HW Interface
- Micro-architecture
- Logic
- Devices
- Electrons
Fundamentally Better Architectures

Data-centric

Data-driven

Data-aware
End of Backup Slides