Intelligent Architectures for Intelligent Machines

Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu
1 October 2022
INSAIT Conference
How Intelligent Are Our Platforms?

Source: https://taxistartup.com/wp-content/uploads/2015/03/UK-Self-Driving-Cars.jpg
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

Memory and compute requirements

- **2018**
  - BERT Base (110M)
  - BERT Large (340M)
  - GPT-2 (1.5B)
  - T-NLG (17B)
- **2019**
  - GPT-3 (175B)
  - MSFT-1T (1T)
  - MT-NLG (530B)
- **2020+**
  - T5 (11B)
  - Megatron-LM (8B)

Total training compute, PFLOP-days

Model memory requirement, GB

- **1800x more compute**
  - In just 2 years

Tomorrow, multi-trillion parameter models

[https://www.youtube.com/watch?v=x2-qB0J7KHW](https://www.youtube.com/watch?v=x2-qB0J7KHW)
Data is Key for Future Workloads

In-memory Databases
[Mao+ (EuroSys’12; 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
Google’s video codec

Video Playback

Video Capture
Data Overwhelms Modern Machines

Chrome

TensorFlow Mobile

Data → performance & energy bottleneck

VP9

Video Playback

Google’s video codec

Video Capture

Google’s video codec
Data is Key for Future Workloads

Development of high-throughput sequencing (HTS) technologies

Number of Genomes Sequenced

We Need Faster & Scalable Genome Analysis

Understanding **genetic variations**, species, evolution, ...

Predicting the **presence** and relative **abundance** of **microbes** in a sample

Rapid surveillance of **disease outbreaks**

Developing **personalized medicine**

And, many, many other applications ...
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 ▼

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

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

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

- Gagandeep Singh, Mohammed Alser, Damla Senol Cali, Dionysios Diamantopoulos, Juan Gómez-Luna, Henk Corporaal, and Onur Mutlu,

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


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

Gagandeep Singh\(^\d\) Mohammed Alser\(^\d\) Damla Senol Cali\(^\d\)

Dionysios Diamantopoulos\(^\d\) Juan Gómez-Luna\(^\d\)

Henk Corporaal\(^*\) Onur Mutlu\(^\d,\d\)

\(^\d\)ETH Zürich \(^\d\)Carnegie Mellon University

\(^*\)Eindhoven University of Technology \(^\d\)IBM Research Europe

SAFARI
Beginner Reading on Genome Analysis

Mohammed Alser, Joel Lindegger, Can Firtina, Nour Almadhoun, Haiyu Mao, Gagandeep Singh, Juan Gomez-Luna, Onur Mutlu

“From Molecules to Genomic Variations to Scientific Discovery: Intelligent Algorithms and Architectures for Intelligent Genome Analysis”
Computational and Structural Biotechnology Journal, 2022

[Source code]

Review

From molecules to genomic variations: Accelerating genome analysis via intelligent algorithms and architectures

Mohammed Alser *, Joel Lindegger, Can Firtina, Nour Almadhoun, Haiyu Mao, Gagandeep Singh, Juan Gomez-Luna, Onur Mutlu *

ETH Zurich, Gloriastrasse 35, 8092 Zürich, Switzerland

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

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

AMD Ryzen 5000, 2020

Core Count:
8 cores/16 threads

L1 Caches:
32 KB per core

L2 Caches:
512 KB per core

L3 Cache:
32 MB shared

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
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)

https://www.gsmarena.com/apple_announces_m1_ultra_with_20core_cpu_and_64core_gpu-news-53481.php
Data Overwhelms Modern Machines

Data → performance & energy bottleneck

Chrome

TensorFlow Mobile

Video Playback

Video Capture

VP9

Google’s video codec

Google’s video codec
Data Movement Overwhelms Modern Machines


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

Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks

Amirali Boroumand\textsuperscript{1}, Rachata Ausavarunngirun\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}
Axiom

An Intelligent Architecture
Handles Data Well
How to Handle Data Well

- Ensure data does not overwhelm the components
  - via intelligent algorithms, architectures & 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
Architectures for Intelligent Machines

Data-centric

Data-driven

Data-aware
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
We Need to Revisit the Entire Stack

We can get there step by step
Data-Centric (Memory-Centric) Architectures
Data-Centric Architectures: Properties

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

- **Low-latency & low-energy data access**

- **Low-cost data storage & processing**

- **Intelligent data management**
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 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.

https://doi.org/10.1109/T-C.1969.222754
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?

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

- **Huge demand from Applications & Systems**
  - Data access bottleneck
  - Energy & power bottlenecks
  - Data movement energy dominates computation energy
  - Need all at the same time: performance, energy, sustainability
  - We can improve all metrics by minimizing data movement

- **Designs are squeezed in the middle**
Processing-in-Memory Landscape Today

And, many other experimental chips and startups
Memory Scaling Issues Are 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/

Infrastructures to Understand Such Issues

Memory Testing Infrastructures

*SoftMC [Hassan+, HPCA’17] enhanced for DDR4*
Updated Memory Testing Infrastructure

FPGA-based SoftMC (Xilinx Virtex UltraScale+ XCU200)

Fine-grained control over **DRAM commands**, timing (±1.5ns), **temperature** (±0.1°C), and **voltage** (±1mV)

A Curious Phenomenon [Kim et al., ISCA 2014]

One can predictably induce errors in most DRAM memory chips

RowHammer

A simple hardware failure mechanism can create a widespread system security vulnerability.
Memory Scaling Issues Are Real

- Onur Mutlu,
  "The RowHammer Problem and Other Issues We May Face as Memory Becomes Denser"
  [Slides (pptx) (pdf)]

The RowHammer Problem
and Other Issues We May Face as Memory Becomes Denser

Onur Mutlu
ETH Zürich
onur.mutlu@inf.ethz.ch
https://people.inf.ethz.ch/omutlu

Memory Scaling Issues Are Real

- Onur Mutlu and Jeremie Kim, "RowHammer: A Retrospective"
  
  IEEE Transactions on Computer-Aided Design of Integrated Circuits and
  Systems (TCAD) Special Issue on Top Picks in Hardware and
  
  [Preliminary arXiv version]
  [Slides from COSADE 2019 (pptx)]
  [Slides from VLSI-SOC 2020 (pptx) (pdf)]
  [Talk Video (1 hr 15 minutes, with Q&A)]

RowHammer: A Retrospective

Onur Mutlu§‡
§ETH Zürich

Jeremie S. Kim‡§
‡Carnegie Mellon University
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)]
The Push from Circuits and Devices

Main Memory Needs
Intelligent Controllers
Intel Optane Persistent Memory (2019)

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

https://www.storagereview.com/intel_optane_dc_persistent_memory_module_pmm

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.
The Takeaway

Intelligent Memory Controllers Can Avoid Many Failures & Enable Better Scaling
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
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

- Processor centric

- All data processed in the processor → at great system cost
“It’s the Memory, Stupid!” (Richard Sites, MPR, 1996)

It’s the Memory, Stupid!

When we started the Alpha architecture design in 1988, we estimated a 25-year lifetime and a relatively modest 32% per year compounded performance improvement of implementations over that lifetime (1,000× total). We guestimated about 10× would come from CPU clock improvement, 10× from multiple instruction issue, and 10× from multiple processors.

5, 1996 MICROPROCESSOR REPORT

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

The Performance Perspective

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 Energy Perspective

Communication Dominates Arithmetic

- 64-bit DP 20pJ
- 256-bit buses
- 256-bit access 8 kB SRAM
- 26 pJ
- 256 pJ
- 16 nJ DRAM Rd/WR
- 500 pJ Efficient off-chip link
- 50 pJ
- 1 nJ

Dally, HiPEAC 2015
A memory access consumes $\sim 100\text{-}1000\times$ the energy of a complex addition.
Data Movement vs. Computation Energy

Energy for a 32-bit Operation (log scale)

- **ADD (int)**: 0.1 pJ
- **ADD (float)**: 0.9 pJ
- **Register File**: 1
- **MULT (int)**: 3.1 pJ
- **MULT (float)**: 3.7 pJ
- **SRAM Cache**: 5
- **DRAM**: 640 pJ

- **Energy (pJ)**
- **ADD (int) Relative Cost**

A memory access consumes 6400X the energy of a simple integer addition.
Energy Waste in Mobile Devices


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 Aki Kuusela³
Saugata Ghose¹ Eric Shiu³ Rahul Thakur³ Parthasarathy Ranganathan³
Youngsok Kim² Daehyun Kim⁴,³ Onur Mutlu⁵,¹

SAFARI
We Do Not Want to Move Data!

Communication Dominates Arithmetic

A memory access consumes $\sim 100\text{-}1000\times$ 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?
A Modern Primer on Processing in Memory

Onur Mutlu<sup>a,b</sup>, Saugata Ghose<sup>b,c</sup>, Juan Gómez-Luna<sup>a</sup>, Rachata Ausavarungnirun<sup>d</sup>

SAFARI Research Group

<sup>a</sup>ETH Zürich  
<sup>b</sup>Carnegie Mellon University  
<sup>c</sup>University of Illinois at Urbana-Champaign  
<sup>d</sup>King Mongkut’s University of Technology North Bangkok


SAFARI

We Need to Think Differently from the Past Approaches
Processing in Memory: Two Approaches

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

Memory similar to a “conventional” accelerator
Accelerating In-Memory Graph Analytics

- Large graphs are everywhere (circa 2015)

- Scalable large-scale graph processing is challenging

![Graph Analytics Diagram](image-url)

- 36 Million Wikipedia Pages
- 1.4 Billion Facebook Users
- 300 Million Twitter Users
- 30 Billion Instagram Photos

![Speedup Chart](chart-url)

- 32 Cores
- 128...

+42% Speedup
Key Bottlenecks in Graph Processing

```java
for (v: graph.vertices) {
    for (w: v.successors) {
        w.next_rank += weight * v.rank;
    }
}
```

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)

Memory
Logic

In-Order Core
LP
PF Buffer
MTP
Message Queue

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

- Logic
  - Crossbar Network

- Memory
  - In-Order Core
  - DRAM Controller
  - NI

- Communications via Remote Function Calls
  - Message Queue

SAFARI
Tesseract System for Graph Processing

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

Memory

Logic

Crossbar Network

Prefetching

DRAM Controller

Message Queue

NI

Host Processor

LP

PF Buffer

MTP

Message Queue

Host Processor
Evaluated Systems

**DDR3-OoO**
- 8 OoO 4GHz
- 102.4GB/s

**HMC-OoO**
- 8 OoO 4GHz
- 640GB/s

**HMC-MC**
- 128 In-Order 2GHz
- 640GB/s

**Tesseract**
- 32 Tesseract Cores
- 8TB/s

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

Speedup

DDR3-OoO   HMC-OoO   HMC-MC   Tesseract   Tesseract-LP   Tesseract-LP-MTP

0          2          4          6          8          10          12          14          16

<0% +25% +56% 9.0x 11.6x 13.8x

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

> 8X Energy Reduction

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

Top Picks Honorable Mention by IEEE Micro.

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

Junwhan Ahn  Sungpack Hong§  Sungjoo Yoo  Onur Mutlu†  Kiyoung 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

SAFARI
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)] [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
Genome Sequence Analysis

Data Movement from Storage

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

Alignment

Computation overhead

Data movement overhead
Accelerating Genome Sequence Analysis

- **Heuristics**
- **Accelerators**
- **Filters**

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

- ![Checkmark] Computation overhead
- ![X] Data movement overhead

**SAFARI**
Key Idea

Filter reads that do not require alignment inside the storage system

Filtered Reads

Filter out Exactly-matching reads

Filter out Non-matching reads

SAFARI
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
GenStore Talk Video

GenStore-EM: Not Finding a Match

Sorted Read Table

<table>
<thead>
<tr>
<th>Read</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAAA</td>
</tr>
<tr>
<td>AAAA</td>
</tr>
<tr>
<td>AAAA</td>
</tr>
<tr>
<td>AAAA</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

Sorted K-mer Index

<table>
<thead>
<tr>
<th>K-mer</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAAA</td>
</tr>
<tr>
<td>AAAA</td>
</tr>
<tr>
<td>AAAA</td>
</tr>
<tr>
<td>AAAA</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>

Comparator

Read < K-mer

---

GenStore: A High-Performance In-Storage Processing System for Genome Analysis -- ASPLOS'22 Talk

https://www.youtube.com/watch?v=kIU44FxjbFk
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
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 Memory Modules

- E19: 8 chips DIMM (1 rank). DRAM PUs @ 267 MHz
- P21: 16 chips DIMM (2 ranks). DRAM PUs @ 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 Fernández, ETH Zürich 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, oblivious the 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 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 UPNEM company has designed and manufactured the first publicly available real-world PIM architecture. The UPNEM PIM architecture combines traditional DRAM memory arrays with general-purpose in-order cores, called DRAM Processing Units (DPU), 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 UPNEM-based PIM system using microbenchmarks to assess various architecture limits such as compute throughput and memory bandwidth, yielding new insights. Second, we present PIM benchmark (processing-in-memory benchmark), a benchmark suite of 16 workloads from different application domains (e.g., dense-space linear algebra, databases, data analytics, graph processing, neural networks, bioinformatics, image processing), which we identify as memory-bound. We evaluate the performance and energy consumption of PIM benchmarks on the UPNEM 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 UPNEM-based PIM systems with 140 and 2,560 CPUs 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

https://www.youtube.com/watch?v=Sscy1Wrr22A&list=PL5Q2soXY2Zj9xidy1qBxUz7xRPS-wisBN&index=26
Benchmarking Memory-Centric Computing Systems: Analysis of Real Processing-in-Memory Hardware
## 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

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.
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 GIANNOULA¹,⁴, 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
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=qeukNs5XI3g&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 access

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

https://www.youtube.com/watch?v=qeukNs5XI3g&t=11226s
Samsung Develops Industry’s First High Bandwidth Memory with AI Processing Power

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)
- 1.2 TFLOPS (4H)
- FP16 ADD /
  Multiply (MUL) /
  Multiply-Accumulate (MAC) /
  Multiply-and-Add (MAD)

**ISSCC 2021 / SESSION 25 / DRAM / 25.4**

25.4 A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2TFLOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications

Young-Heeun Kwon1, Suk Han Lee1, Jaeheon Lee1, Sang-Hyuk Kwon1, Ja Min Ryu1, Jong-Pil Son1, Seonggil O1, Hak-Soo Yu1, Hassuk Lee1, Soo Young Kim1, Youngmin Cho1, Jin Guk Kim1, Jongyoon Choi1, Hyun-Sung Shin1, Jin Kim1, BengSeng Phuah1, HyungMin Kim1, Myeong Jun Song1, Ahn Choi1, Daeho Kim1, SooYoung Kim1, Eun-Bong Kim1, David Wang1, Shinheung Kang1, Yuwan Ro1, Seungwoo Seo1, JoonHo Song1, Jaryoun Youn1, Kyomin Sohn1, Nam Sung Kim1

1Samsung Electronics, Hwasung, Korea
2Samsung Electronics, San Jose, CA
3Samsung Electronics, Suwon, Korea
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
Mixed design methodology to implement FIMDRAM

- Full-custom + Digital RTL
Samsung & Meta AxDIMM (2021)

- **DDRx-PIM**
  - Deep learning 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) 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
Eliminating the Adoption Barriers

Processing-in-Memory in the Real World
PIM Course (Spring 2022)

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

- **Youtube Livestream:**
  - [https://www.youtube.com/watch?v=9e4Chnwdovo&list=PL5Q2soXY2Zi841fUYYUK9EsXKhQKRPyX](https://www.youtube.com/watch?v=9e4Chnwdovo&list=PL5Q2soXY2Zi841fUYYUK9EsXKhQKRPyX)

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

[https://www.youtube.com/onurmutlulectures](https://www.youtube.com/onurmutlulectures)
Processing in Memory:
Two Approaches

1. Processing near Memory
2. Processing using 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
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
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
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

Thank you!
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!

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

https://safari.ethz.ch/safari-newsletter-december-2021/
Referenced 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 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
GitHub Stars: 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 publi...

C: 53
GitHub Stars: 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
GitHub Stars: 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
GitHub Stars: 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
GitHub Stars: 93

rowhammer (Public)

C: 189
GitHub Stars: 41

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

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

https://www.youtube.com/watch?v=qeukNs5XI3g
Comp Arch (Fall 2021)

- 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_5EFgO&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAGILkTOF](https://www.youtube.com/watch?v=4yfkM_5EFgO&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAGILkTOF)

- Youtube Livestream (2020):
  - [https://www.youtube.com/watch?v=c3mPdZA-Fmc&list=PL5Q2soXY2Zi9xydyIgBxUz7xRPS-wisBN](https://www.youtube.com/watch?v=c3mPdZA-Fmc&list=PL5Q2soXY2Zi9xydyIgBxUz7xRPS-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)
DDCA (Spring 2022)

**Spring 2022 Edition:**

**Spring 2021 Edition:**

**Youtube Livestream (Spring 2022):**
- [https://www.youtube.com/watch?v=cpXdE3HwyK0&list=PL5Q2soXy2Zi97Ya5DEUpMpo2bbAoaG7c6](https://www.youtube.com/watch?v=cpXdE3HwyK0&list=PL5Q2soXy2Zi97Ya5DEUpMpo2bbAoaG7c6)

**Youtube Livestream (Spring 2021):**
- [https://www.youtube.com/watch?v=LbC0EZYY8yw4&list=PL5Q2soXy2Zi_uj3aY39YB5pW4SJ7L1N](https://www.youtube.com/watch?v=LbC0EZYY8yw4&list=PL5Q2soXy2Zi_uj3aY39YB5pW4SJ7L1N)

**Bachelor’s course**
- 2nd semester at ETH Zurich
- Rigorous introduction into “How Computers Work”
- Digital Design/Logic
- Computer Architecture
- 10 FPGA Lab Assignments

[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=PL5Q2soXY2Zi841fUYYUK9EsXKhQKRPyX

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

https://www.youtube.com/onurmutlulectures
Genomics (Spring 2022)

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

- **Youtube Livestream:**
  - https://www.youtube.com/watch?v=DEL_5A_Y3TI&list=PL5Q2soXY2Zi8NrdGOr1yRU_Cxxjw-u18

- Project course
  - Taken by Bachelor’s/Master’s students
  - Genomics lectures
  - Hands-on research exploration
  - Many research readings

**Spring 2022 Meetings/Schedule**

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Livestream</th>
<th>Meeting</th>
<th>Learning Materials</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>11.3</td>
<td>YouTube Live</td>
<td>M1: P&amp;S Accelerating Genomics Course Introduction &amp; Project Proposal (PDF) (PPT)</td>
<td>Required Materials</td>
</tr>
<tr>
<td></td>
<td>Fri.</td>
<td></td>
<td></td>
<td>Recommended Materials</td>
</tr>
<tr>
<td>W2</td>
<td>18.3</td>
<td>YouTube Live</td>
<td>M2: Introduction to Sequencing (PDF) (PPT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fri.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W3</td>
<td>25.3</td>
<td>YouTube Premiere</td>
<td>M3: Read Mapping (PDF) (PPT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fri.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W4</td>
<td>01.04</td>
<td>YouTube Premiere</td>
<td>M4: GateKeeper (PDF) (PPT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fri.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W5</td>
<td>08.04</td>
<td>YouTube Premiere</td>
<td>M5: MAGNET &amp; Shouji (PDF) (PPT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fri.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W6</td>
<td>15.4</td>
<td>YouTube Premiere</td>
<td>M6: SneakySnake (PDF) (PPT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fri.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W7</td>
<td>29.4</td>
<td>YouTube Premiere</td>
<td>M7: GenStore (PDF) (PPT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fri.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W8</td>
<td>06.05</td>
<td>YouTube Premiere</td>
<td>M8: GRIM-Filter (PDF) (PPT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fri.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W9</td>
<td>13.05</td>
<td>YouTube Premiere</td>
<td>M9: Genome Assembly (PDF) (PPT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fri.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W10</td>
<td>20.05</td>
<td>YouTube Live</td>
<td>M10: Genomic Data Sharing Under Differential Privacy (PDF) (PPT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fri.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>W11</td>
<td>10.06</td>
<td>YouTube Premiere</td>
<td>M11: Accelerating Genome Sequence Analysis (PDF) (PPT)</td>
<td></td>
</tr>
</tbody>
</table>
Hetero. Systems (Spring’22)

- **Spring 2022 Edition:**
  - 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

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

https://www.youtube.com/onurmutlulectures
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=PL5Q2osoXY2Zi8nH7un3ghD2nutKWWDk-NK](https://youtube.com/playlist?list=PL5Q2osoXY2Zi8nH7un3ghD2nutKWWDk-NK)

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

[https://www.youtube.com/onurmutlulectures](https://www.youtube.com/onurmutlulectures)
Solid-State Drives (Spring 2022)

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

- **Youtube Livestream:**
  - https://www.youtube.com/watch?v=_q4r71DsY4&list=PL5Q2soXY2Zi8vabcse1kL22DEcgMl2RAq

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

https://www.youtube.com/onurmutlulectures
RowHammer & DRAM Exploration (Fall 2022)

- **Fall 2022 Edition:**
  - https://safari.ethz.ch/projects_and_seminars/fall2022/doku.php?id=softmc

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

- **Youtube Livestream (Spring 2022):**
  - https://www.youtube.com/watch?v=r5QxuoJWttg&list=PL5Q2soXY2Zi_1trfCcker6PTN8WR72icUO

- Bachelor’s course
  - Elective at ETH Zurich
  - Introduction to DRAM organization & operation
  - Tutorial on using FPGA-based infrastructure
  - Verilog & C++
  - Potential research exploration

https://www.youtube.com/onurmutlulectures
Exploration of Emerging Memory Systems (Fall 2022)

- **Fall 2022 Edition:**
  - [https://safari.ethz.ch/projects_and_seminars/fall2022/doku.php?id=ramulator](https://safari.ethz.ch/projects_and_seminars/fall2022/doku.php?id=ramulator)

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

- **Youtube Livestream (Spring 2022):**
  - [https://www.youtube.com/watch?v=aMIIXRQd3s&list=PL5Q2soXY2Zi_TlmLGw_Z8hBo2925ZApqV](https://www.youtube.com/watch?v=aMIIXRQd3s&list=PL5Q2soXY2Zi_TlmLGw_Z8hBo2925ZApqV)

- Bachelor’s course
  - Elective at ETH Zurich
  - Introduction to memory system simulation
  - Tutorial on using Ramulator
  - C++
  - Potential research exploration

https://www.youtube.com/onurmutlulectures
Intelligent Architectures for Intelligent Machines

Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu
1 October 2022
INSAIT Conference
Processing Using Memory: Extra Slides
Mindset: Memory as an Accelerator

CPU core
CPU core
CPU core
CPU core
mini-CPU core
video core
imaging core
GPU (throughput) core
GPU (throughput) core
GPU (throughput) core
GPU (throughput) core
LLC
Memory Controller
Memory Bus
Memory similar to a “conventional” accelerator

Specialized compute-capability in memory
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
Future Systems: In-Memory Copy

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

1046ns, 3.6uJ → 90ns, 0.04uJ
RowClone: In-DRAM Row Copy

Idea: Two consecutive ACTivates
Negligible HW cost

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

DRAM subarray

Row Buffer (4 Kbytes)

4 Kbytes

8 bits

Data Bus
RowClone: Latency and Energy Savings

![Bar chart showing latency and energy savings](image)

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"
  [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 Wentzlaff  
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§†  Juan Gómez Luna§  Konstantinos Kanellopoulos§  Behzad Salami§*
Hasan Hassan§  Oğuz Ergin†  Onur Mutlu§
§ETH Zürich  †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=queukNs5XI3g&t=4192s
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 `ZynqCopyFPGAConfig`
   - Navigate into `zc706`, then run `make rocket CONFIG=ZynqCopyFPGAConfig -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` and use 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`

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`.

**Generating DDR3 Controller IP sources**

We cannot provide the sources for the Xilinx PHY IP we use in PiDRAM's memory controller due to licensing issues. We describe here how to regenerate them using Vivado 2016.2. First, you need to generate the IP RTL files:

1. Open IP Catalog
2. Find "Memory Interface Generator (MIG 7 Series)" IP and double click

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
PiDRAM is the first flexible end-to-end framework that enables system integration studies and evaluation of real Processing-using-Memory (PuM) techniques. PiDRAM, at a high level, comprises a RISC-V system and a custom memory controller that can perform PuM operations in real DDR3 chips. This repository contains all sources required to build PiDRAM and develop its prototype on the Xilinx ZC706 FPGA boards.

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

PiDRAM is the first flexible end-to-end framework that enables system integration studies and evaluation of real Processing-using-Memory techniques. Prototype on a RISC-V rocket chip system implemented on an FPGA. Described in our preprint: https://arxiv.org/abs/2111.00082
PiDRAM: A Holistic End-to-end FPGA-based Framework for Processing-in-DRAM

Ataberk Olgun, Juan Gómez Luna, Konstantinos Kanellopoulos, Behzad Salami, Hasan Hassan, Oğuz Ergin, Onur Mutlu

Processing-using-memory (PuM) techniques leverage the analog operation of memory cells to perform computation. Several recent works have demonstrated PuM techniques in off-the-shelf DRAM devices. Since DRAM is the dominant memory technology as main memory in current computing systems, these PuM techniques represent an opportunity for alleviating the data movement bottleneck at very low cost. However, system integration of PuM techniques imposes non-trivial challenges that are yet to be solved. Design space exploration of potential solutions to the PuM integration challenges requires appropriate tools to develop necessary hardware and software components. Unfortunately, current specialized DRAM-testing platforms, or system simulators do not provide the flexibility and/or the holistic system view that is necessary to deal with PuM integration challenges.

We design and develop PiDRAM, the first flexible end-to-end framework that enables system integration studies and evaluation of real PuM techniques. PiDRAM provides software and hardware components to rapidly integrate PuM techniques across the whole system software and hardware stack (e.g., necessary modifications in the operating system, memory controller). We implement PiDRAM on an FPGA-based platform along with an open-source RISC-V system. Using PiDRAM, we implement and evaluate two state-of-the-art PuM techniques: in-DRAM (i) copy and initialization, (ii) true random number generation. Our results show that the in-memory copy and initialization techniques can improve the performance of bulk copy operations by 12.6x and bulk initialization operations by 14.6x on a real system. Implementing the true random number generator requires only 190 lines of Verilog and 74 lines of C code using PiDRAM’s software and hardware components.
Long Talk + Tutorial on Youtube

https://youtu.be/s_z_S6FYpC8
(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

Seshadri+, "Fast Bulk Bitwise AND and OR in DRAM", IEEE CAL 2015.
Bulk Bitwise Operations in Workloads

- Bitmap indices (database indexing)
- Set operations
- Encryption algorithms
- BitWeaving (database queries)
- BitFunnel (web search)
- DNA sequence mapping
- ...
In-DRAM Acceleration of Database Queries

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)]

Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology

Vivek Seshadri\textsuperscript{1,5} Donghyuk Lee\textsuperscript{2,5} Thomas Mullins\textsuperscript{3,5} Hasan Hassan\textsuperscript{4} Amirali Boroumand\textsuperscript{5} Jeremie Kim\textsuperscript{4,5} Michael A. Kozuch\textsuperscript{3} Onur Mutlu\textsuperscript{4,5} Phillip B. Gibbons\textsuperscript{5} Todd C. Mowry\textsuperscript{5}

\textsuperscript{1}Microsoft Research India \textsuperscript{2}NVIDIA Research \textsuperscript{3}Intel \textsuperscript{4}ETH Zürich \textsuperscript{5}Carnegie Mellon University
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
Nastaran Hajinazar, Geraldo F. Oliveira, Sven Gregorio, Joao Dinis Ferreira, Nika Mansouri Ghiasi, Minesh Patel, Mohammed Alser, Saugata Ghose, Juan Gomez-Luna, and Onur Mutlu,
"SIMDRAM: An End-to-End Framework for Bit-Serial SIMD Computing in DRAM"

[2-page Extended Abstract]
[Short Talk Slides (pptx) (pdf)]
[Talk Slides (pptx) (pdf)]
[Short Talk Video (5 mins)]
[Full Talk Video (27 mins)]

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

*Nastaran Hajinazar$^{1,2}$  
Nika Mansouri Ghiasi$^1$  
Minesh Patel$^1$  
Juan Gómez-Luna$^1$

*Geraldo F. Oliveira$^1$  
Sven Gregorio$^1$  
Mohammed Alser$^1$  
Onur Mutlu$^1$  
João Dinis Ferreira$^1$  
Saugata Ghose$^3$

$^1$ETH Zürich  
$^2$Simon Fraser University  
$^3$University of Illinois at Urbana–Champaign
**SIMDRAM Framework: Overview**

**Step 1: Generate MAJ logic**

**Desired operation**

- AND/OR/NOT logic

**MAJ/NOR logic**

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

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

**Step 3: Execution according to µProgram**

**SIMDRAM-enabled application**

```plaintext
foo () {
    bbop_new
}
```

**Control Unit**

**µProgram**

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

**SIMDRAM Output**

**Main memory**

- New SIMDRAM µProgram
  - µProgram
  - bbop_new
  - Instruction result in memory

**SAFARI**
SIMDRAM Key Results

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

SIMDRAM provides:

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

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

• $21 \times$ and $2.1 \times$ the performance of a CPU and a high-end GPU, over seven real-world applications
More on SIMDREAM


[2-page Extended Abstract]
[Short Talk Slides (pptx) (pdf)]
[Talk Slides (pptx) (pdf)]
[Short Talk Video (5 mins)]
[Full Talk Video (27 mins)]

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

*Nastaran Hajinazar¹,² Nika Mansouri Ghiasi¹ Geraldo F. Oliveira¹ Minesh Patel¹ Juan Gómez-Luna¹ Sven Gregorio¹ Mohammed Alser¹ Onur Mutlu¹ João Dinis Ferreira¹ Saugata Ghose³

¹ETH Zürich ²Simon Fraser University ³University of Illinois at Urbana–Champaign
In-DRAM Physical Unclonable Functions


[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†§ Minesh Patel§ Hasan Hassan§ Onur Mutlu§†
†Carnegie Mellon University §ETH Zürich
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 Yaglıkçı§ Haocong Luo§
Jeremie S. Kim§ F. Nisa Bostanci§† Nandita Vijaykumar§⊙ Oguz Ergin† Onur Mutlu§

§ETH Zürich †TOBB University of Economics and Technology ⊙University of Toronto
DR-STRaNGe: End-to-End System Design for DRAM-based True Random Number Generators

F. Nisa Bostancı†§ 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-DRAM True Random Number Generation

F. Nisa Bostancı, 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)]

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§ V 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