

Memory-Centric Computing

Enabling Fundamentally Efficient & Intelligent Machines

Onur Mutlu

omutlu@gmail.com

<https://people.inf.ethz.ch/omutlu>

15 July 2025

APPT 2025 Keynote Speech

SAFARI

ETH zürich

Computing

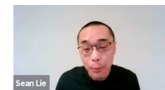
is Bottlenecked by Data

Data is Key for AI, ML, Genomics, ...

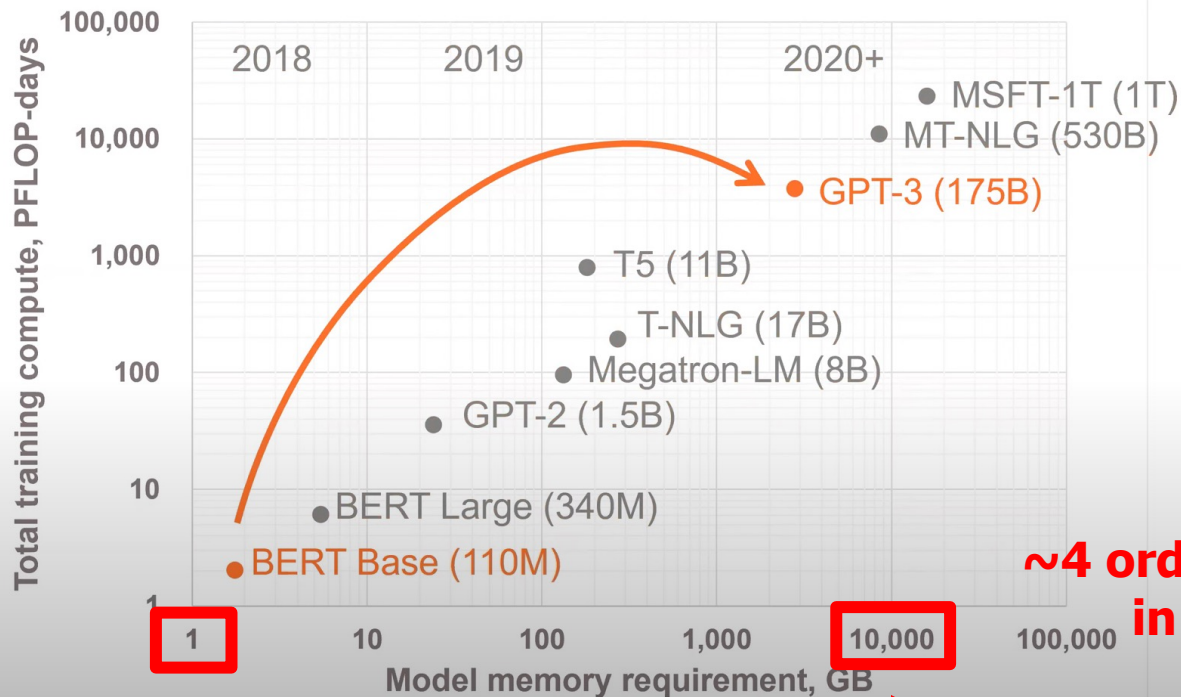
- Important workloads are all data intensive
- They require rapid and efficient processing of large amounts of data
- Data is increasing
 - We can generate more than we can process
 - We need to perform more sophisticated analyses on more data

Huge Demand for Performance & Efficiency

Exponential Growth of Neural Networks



Memory and compute requirements

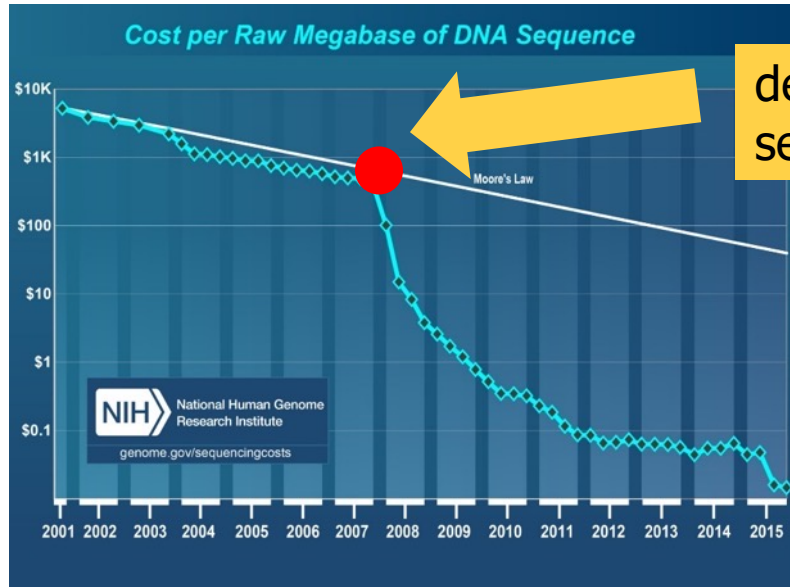


1800x more compute
In just 2 years

Tomorrow, **multi-trillion** parameter models

~4 orders of magnitude increase
in memory requirement
in just a few years!

Huge Demand for Performance & Efficiency

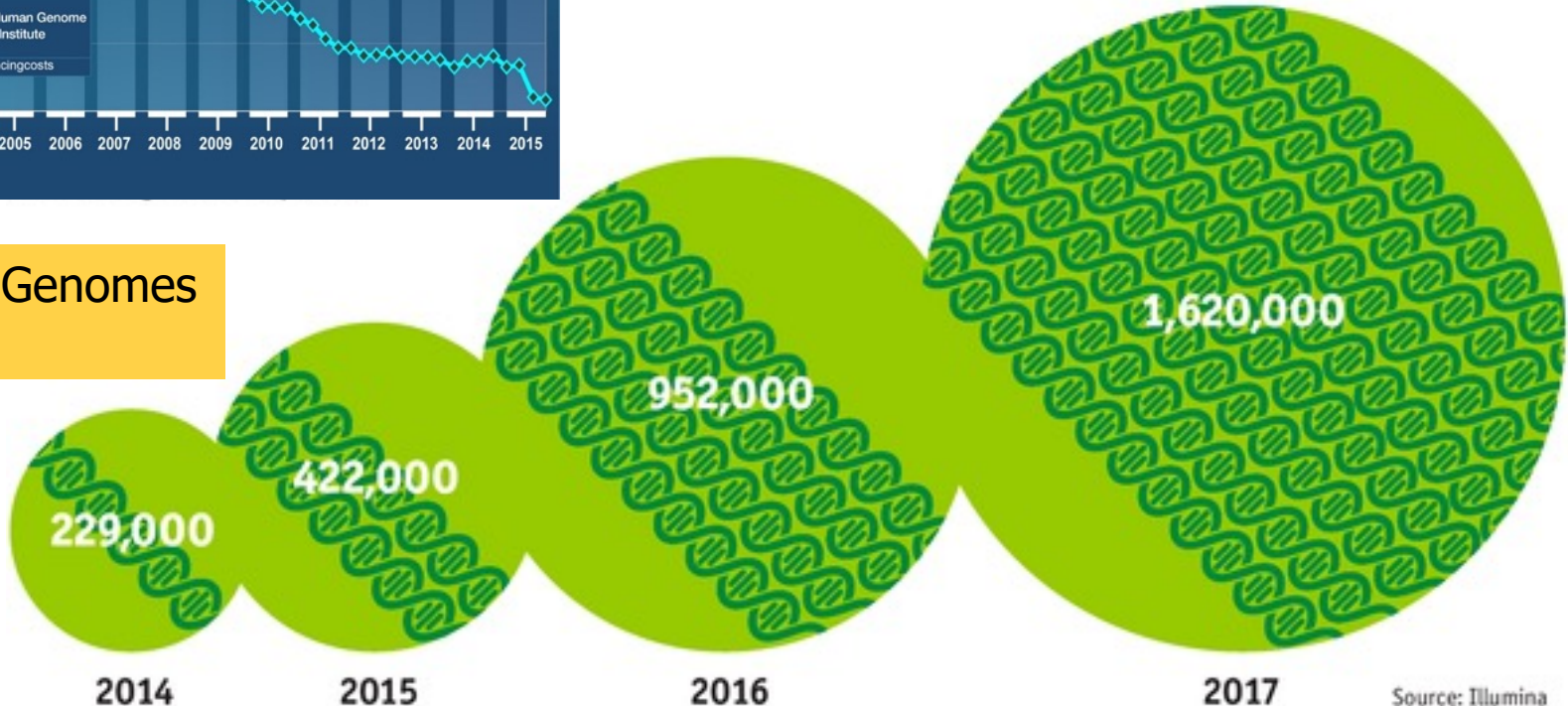


development of new sequencing technologies



Oxford Nanopore MinION

Number of Genomes Sequenced



The Economist

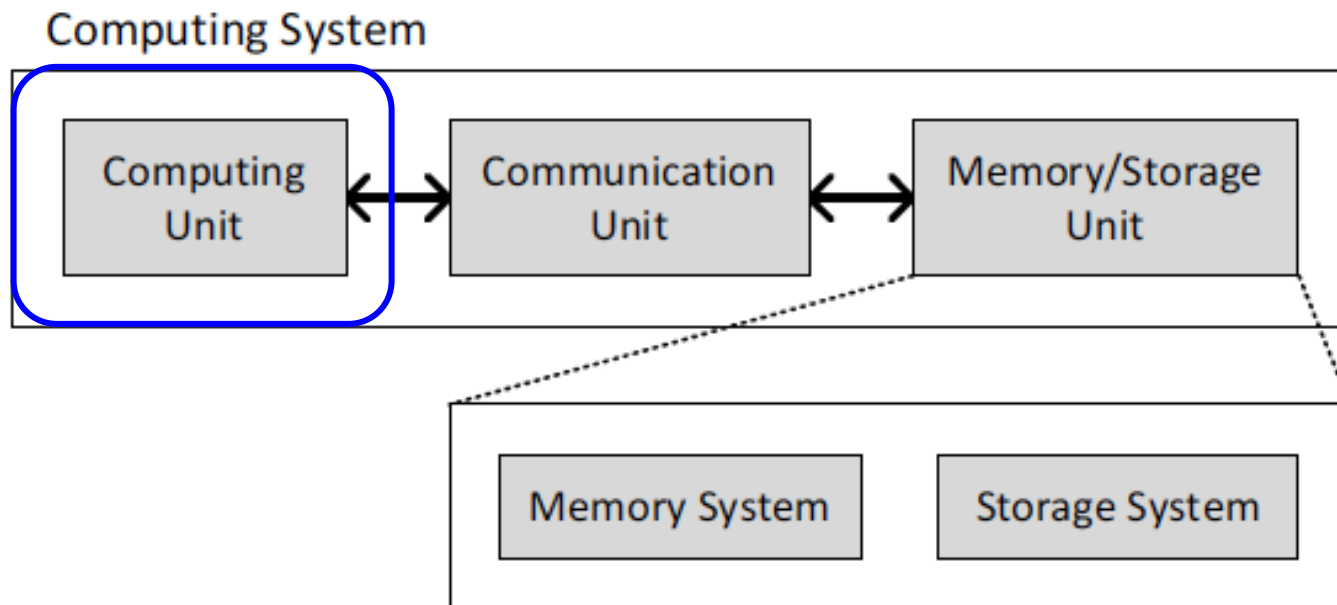
The Problem

Data access is the major performance and energy bottleneck

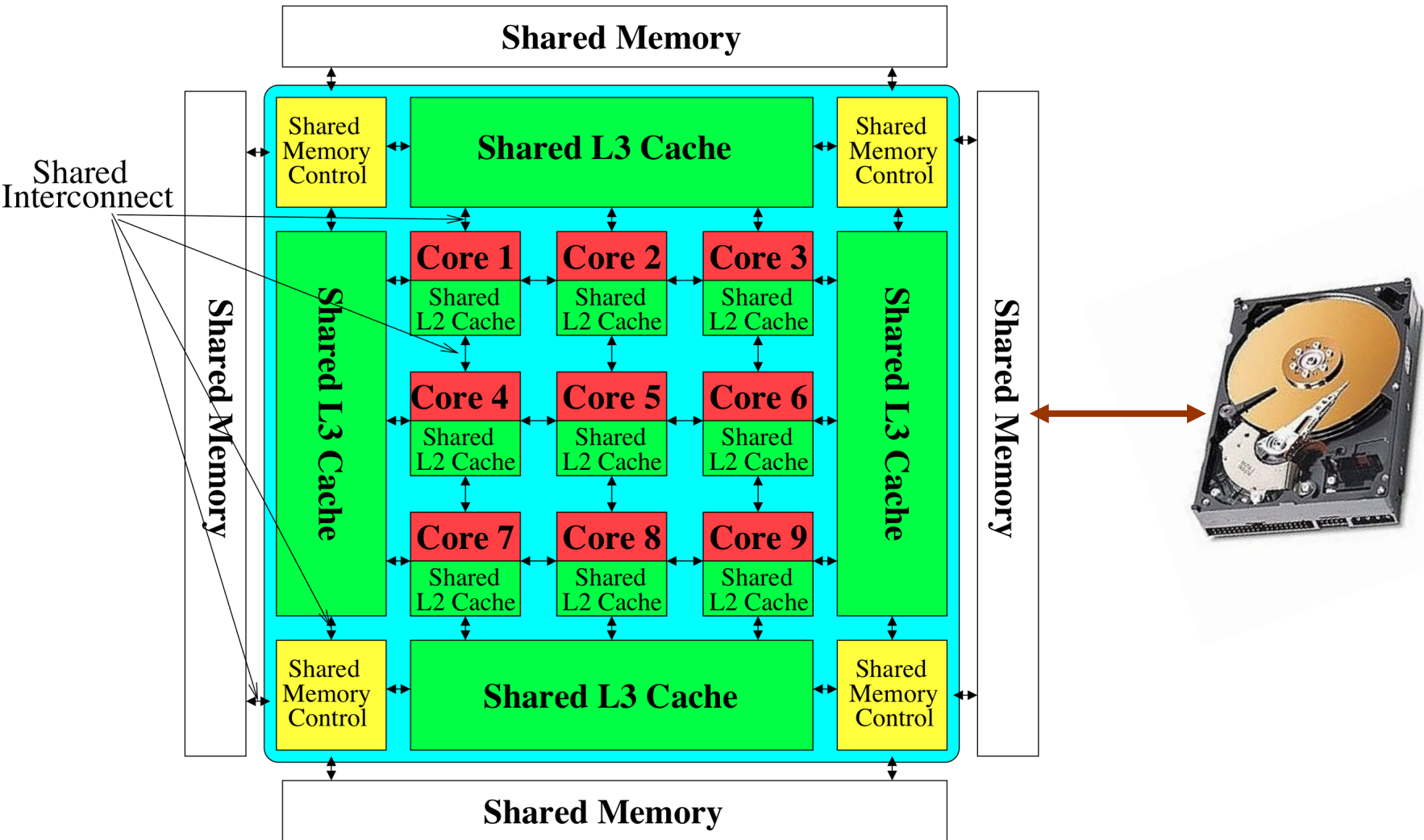
Our current
design principles
cause great energy waste
(and great performance loss)

Today's Computing Systems

- Processor centric
- All data processed in the processor → at great system cost



Perils of Processor-Centric Design

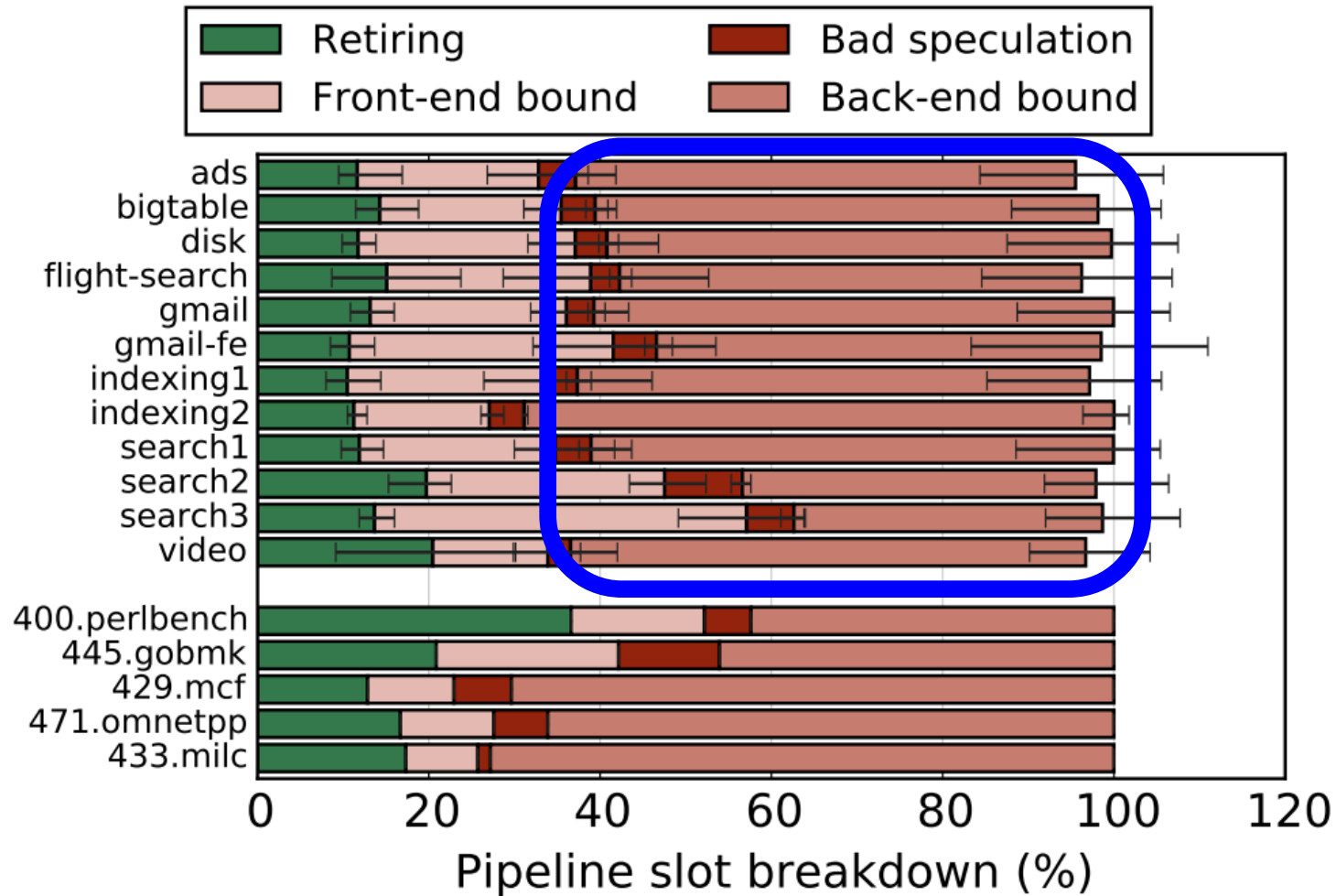


Most of the system is dedicated to storing and moving data

Yet, system is still bottlenecked by memory

Processor-Centric System Performance

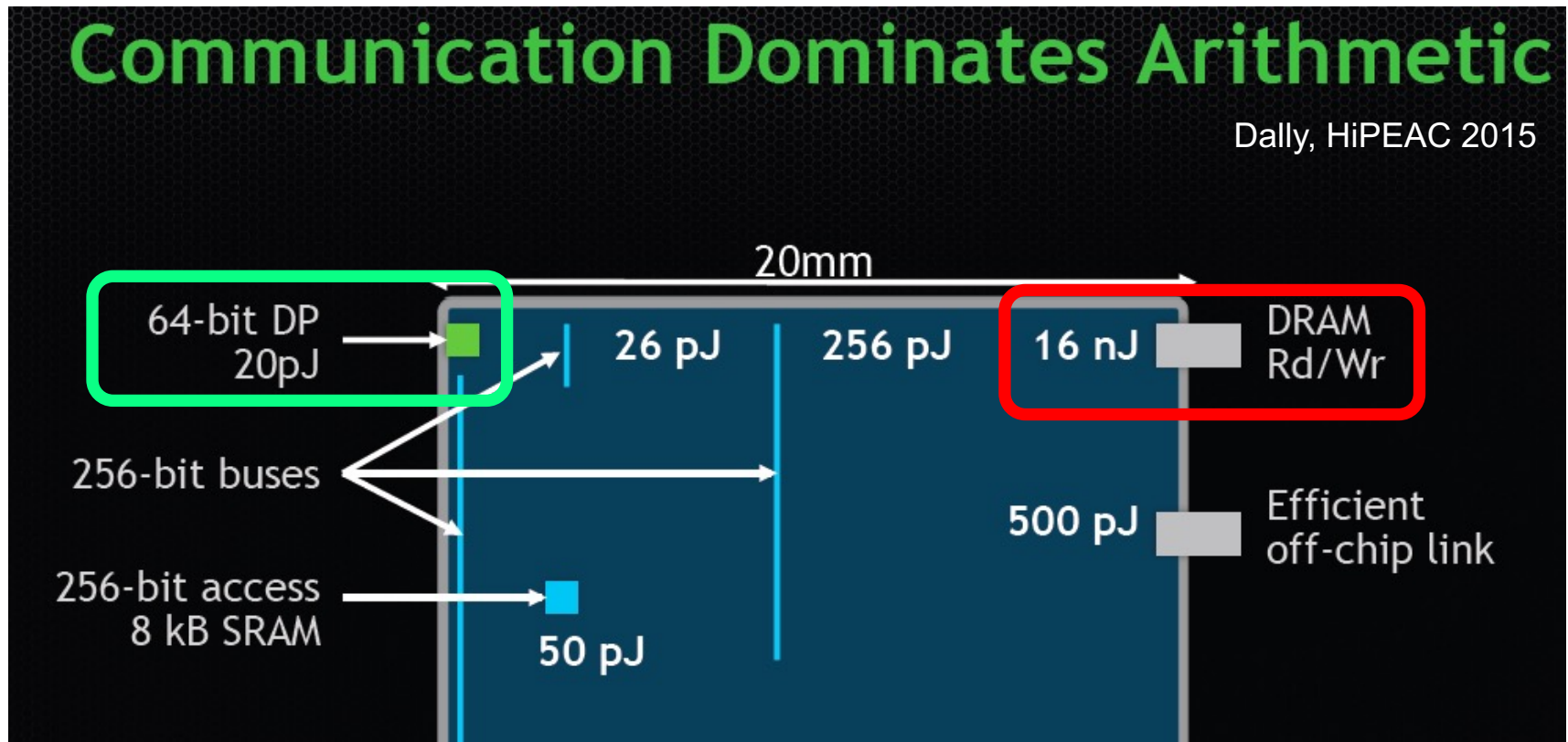
- All of Google's Data Center Workloads (2015):



Data Movement vs. Computation Energy

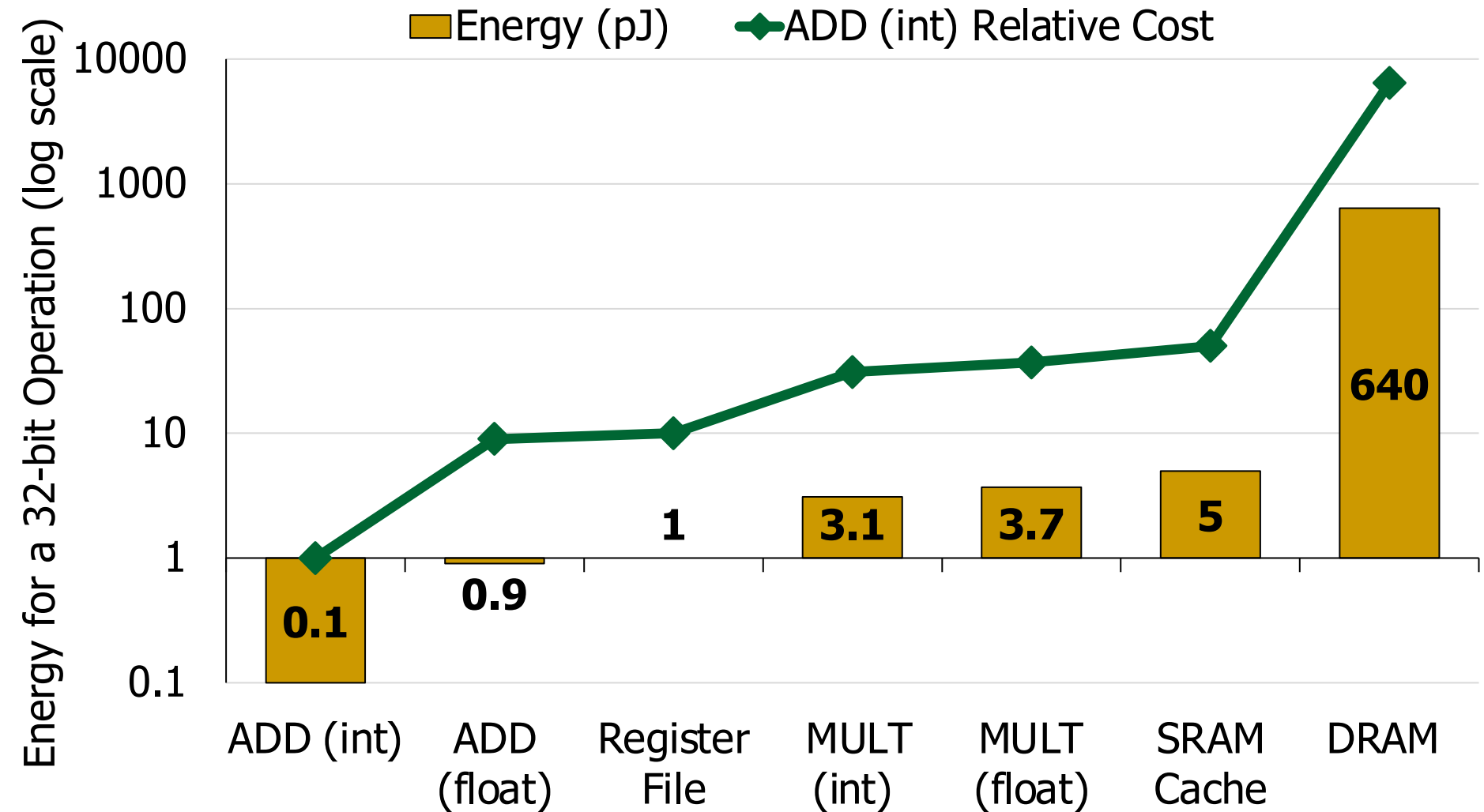
Communication Dominates Arithmetic

Dally, HiPEAC 2015

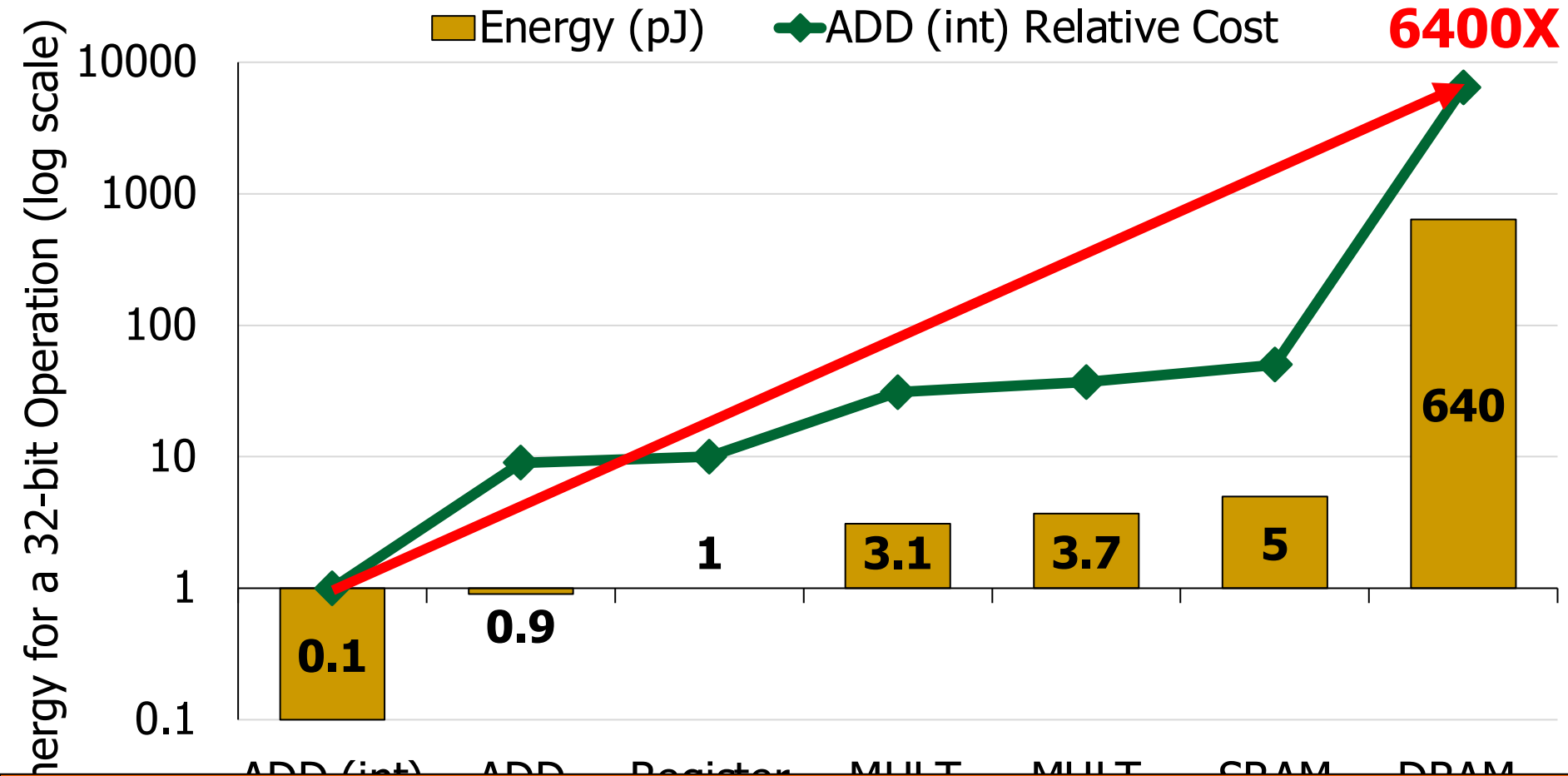


A memory access consumes $\sim 100-1000X$ the energy of a complex addition

Data Movement vs. Computation Energy



Data Movement vs. Computation Energy



A memory access consumes 6400X the energy of a simple integer addition

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"](#) *Proceedings of the 23rd International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Williamsburg, VA, USA, March 2018.

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

Google Workloads for Consumer Devices: Mitigating Data Movement Bottlenecks

Amirali Boroumand¹

Saugata Ghose¹

Youngsok Kim²

Rachata Ausavarungnirun¹

Eric Shiu³

Rahul Thakur³

Daehyun Kim^{4,3}

Aki Kuusela³

Allan Knies³

Parthasarathy Ranganathan³

Onur Mutlu^{5,1}

Energy Waste in Accelerators

- 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)]

**> 90% of the total system energy
is spent on **memory** in large ML models**

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.

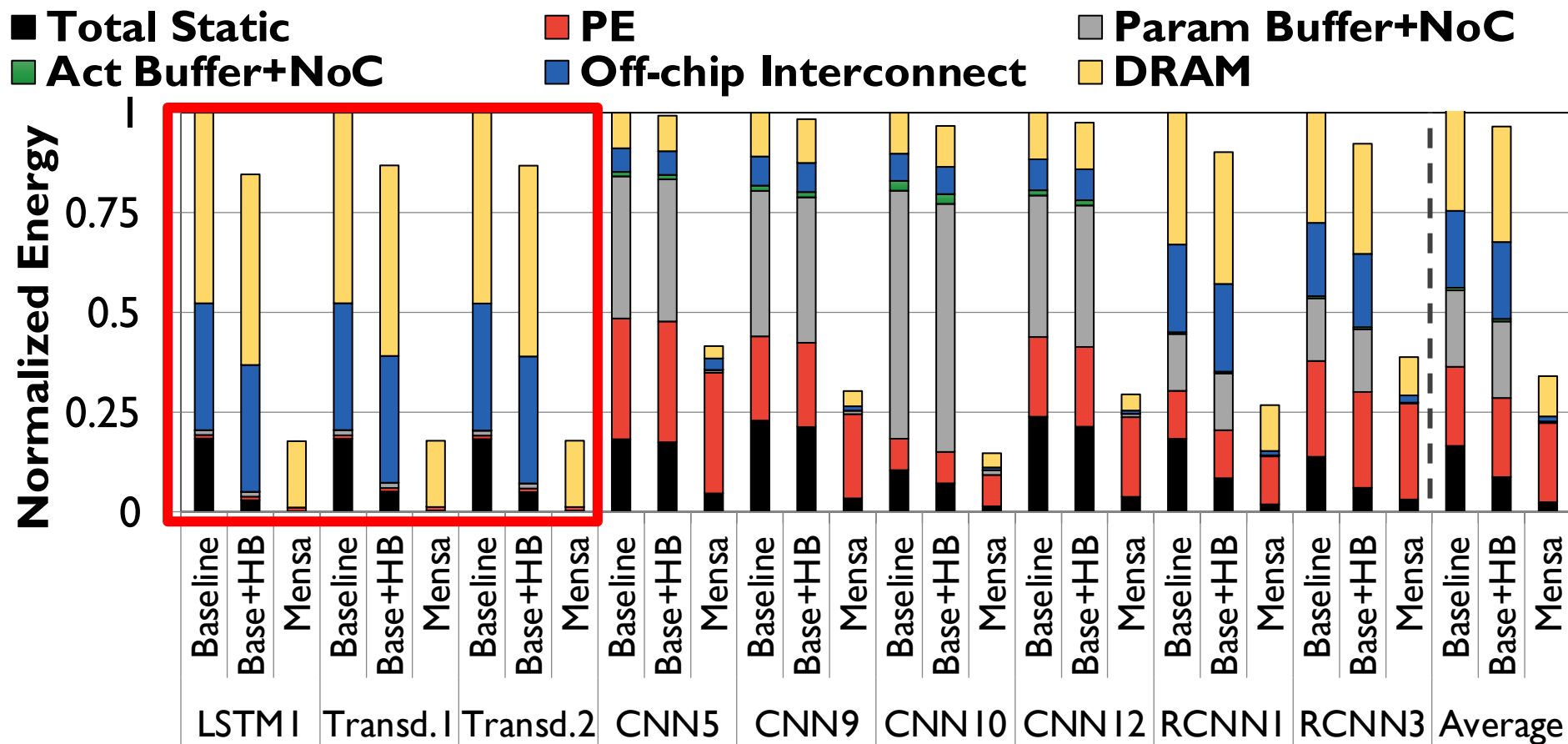
[◇]Stanford Univ.

[‡]Univ. of Illinois Urbana-Champaign

[§]Google

^{*}ETH Zürich

Energy Wasted on Data Movement



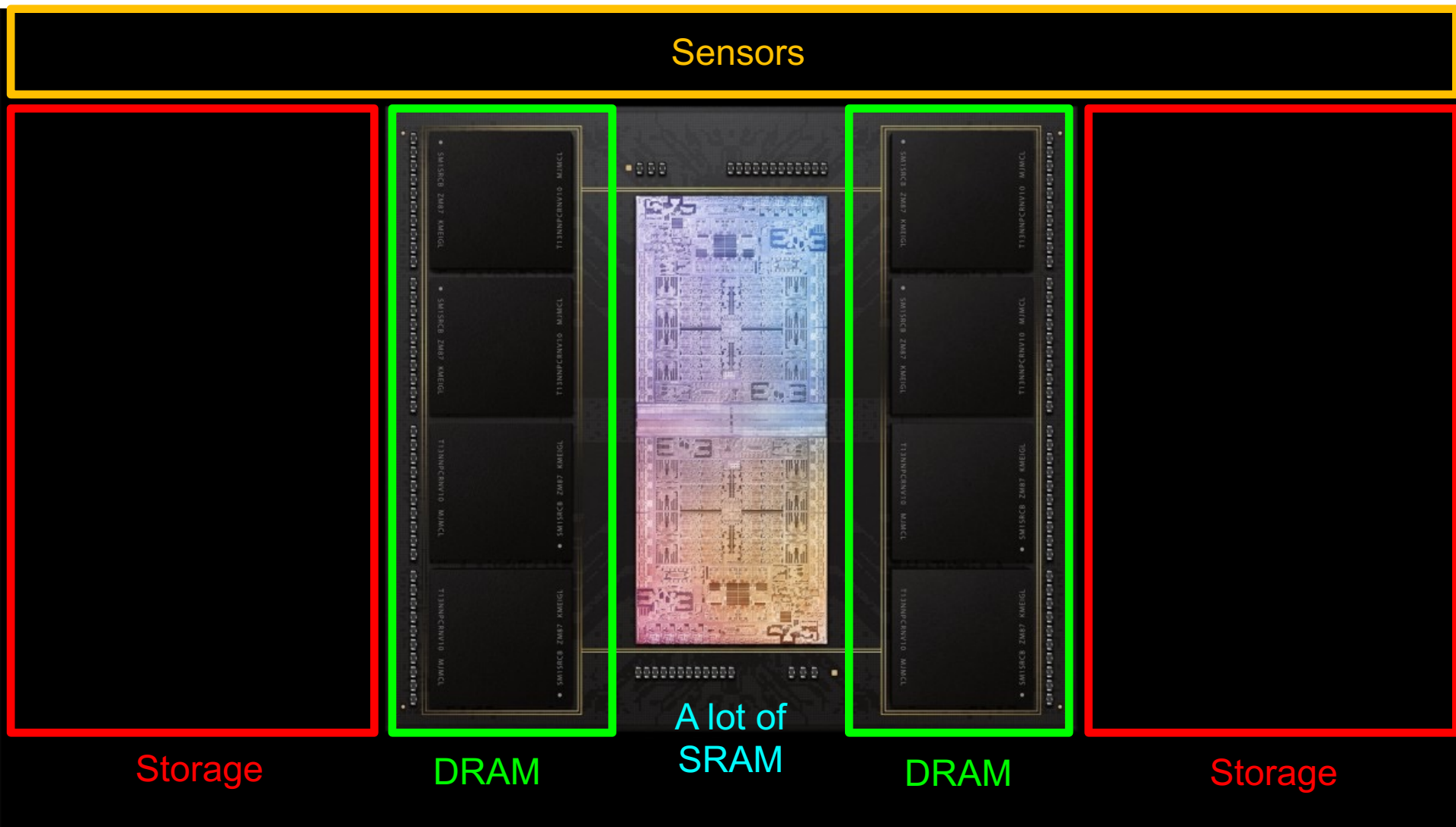
**In LSTMs and Transducers used by Google,
>90% energy spent on off-chip interconnect and DRAM**

Processing of data
is performed
far away from the data

We Need A Paradigm Shift To ...

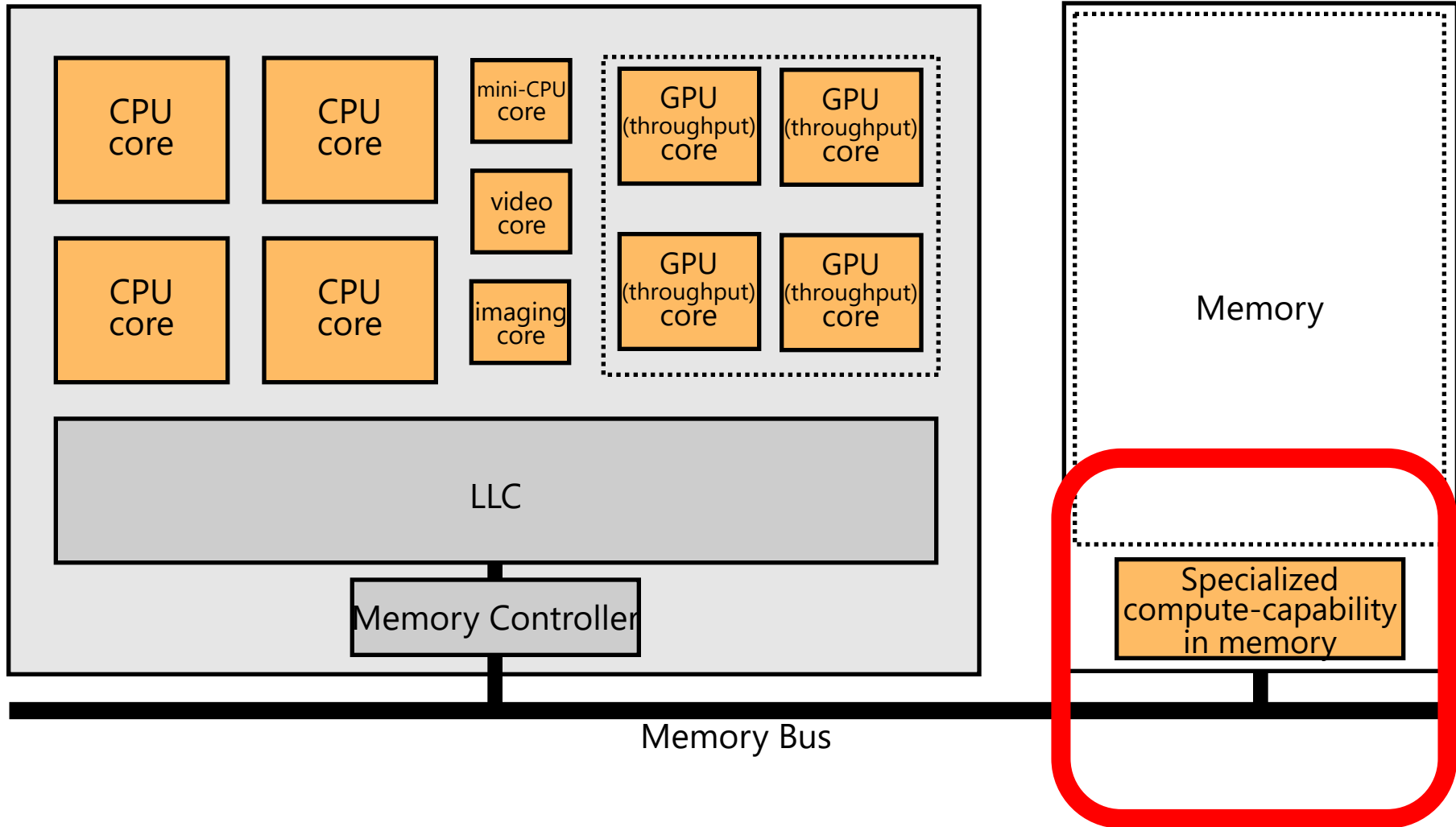
- Enable computation with minimal data movement
- Compute where it makes sense (where data resides)
- Make computing architectures more data-centric

Process Data Where It Makes Sense



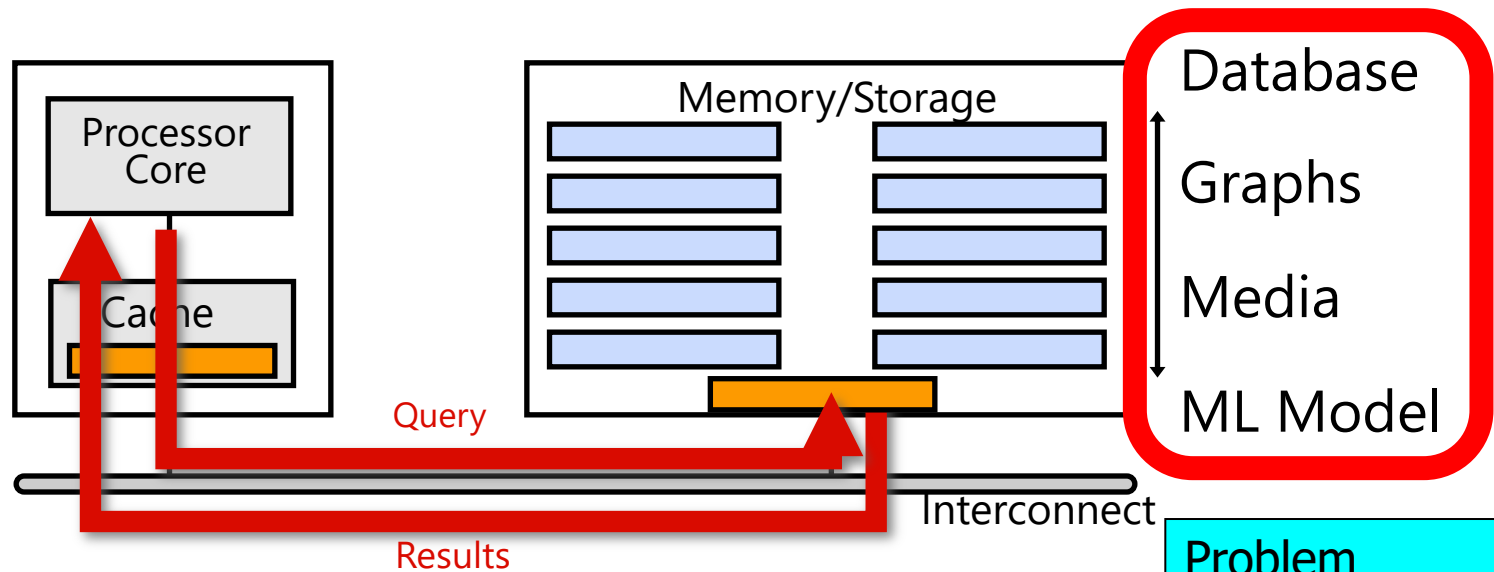
Apple M1 Ultra System (2022)

Memory as an Accelerator



Memory similar to a “conventional” accelerator

Goal: Processing Inside Memory/Storage



- Many questions ... How do we design the:
 - ❑ compute-capable memory & controllers?
 - ❑ processors & communication units?
 - ❑ software & hardware interfaces?
 - ❑ system software, compilers, languages?
 - ❑ algorithms & theoretical foundations?

Problem

Algorithm

Program/Language

System Software

SW/HW Interface

Micro-architecture

Logic

Devices

Electrons

Processing in/near Memory: An Old Idea

- Kautz, "Cellular Logic-in-Memory Arrays", IEEE TC 1969.

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

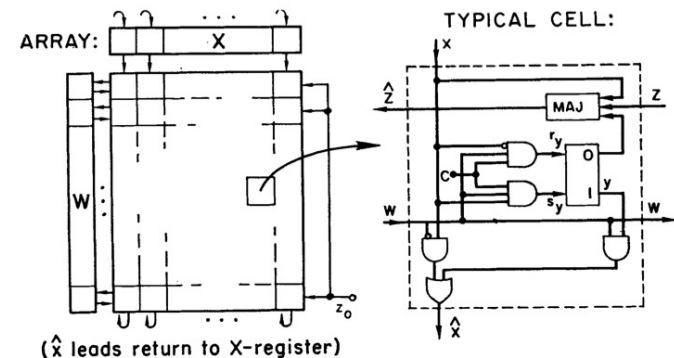
Cellular Logic-in-Memory Arrays

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.



$$\begin{aligned}\hat{x} &= \bar{w}x + wy \\ s_y &= wcx, r_y = wc\bar{x} \\ \hat{z} &= M(x, \bar{y}, z) = x\bar{y} + z(x + \bar{y})\end{aligned}$$

Fig. 1. Cellular sorting array I.

Processing in/near Memory: An Old Idea

- Stone, "A Logic-in-Memory Computer," IEEE TC 1970.

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 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

■ Huge problems with Memory Technology

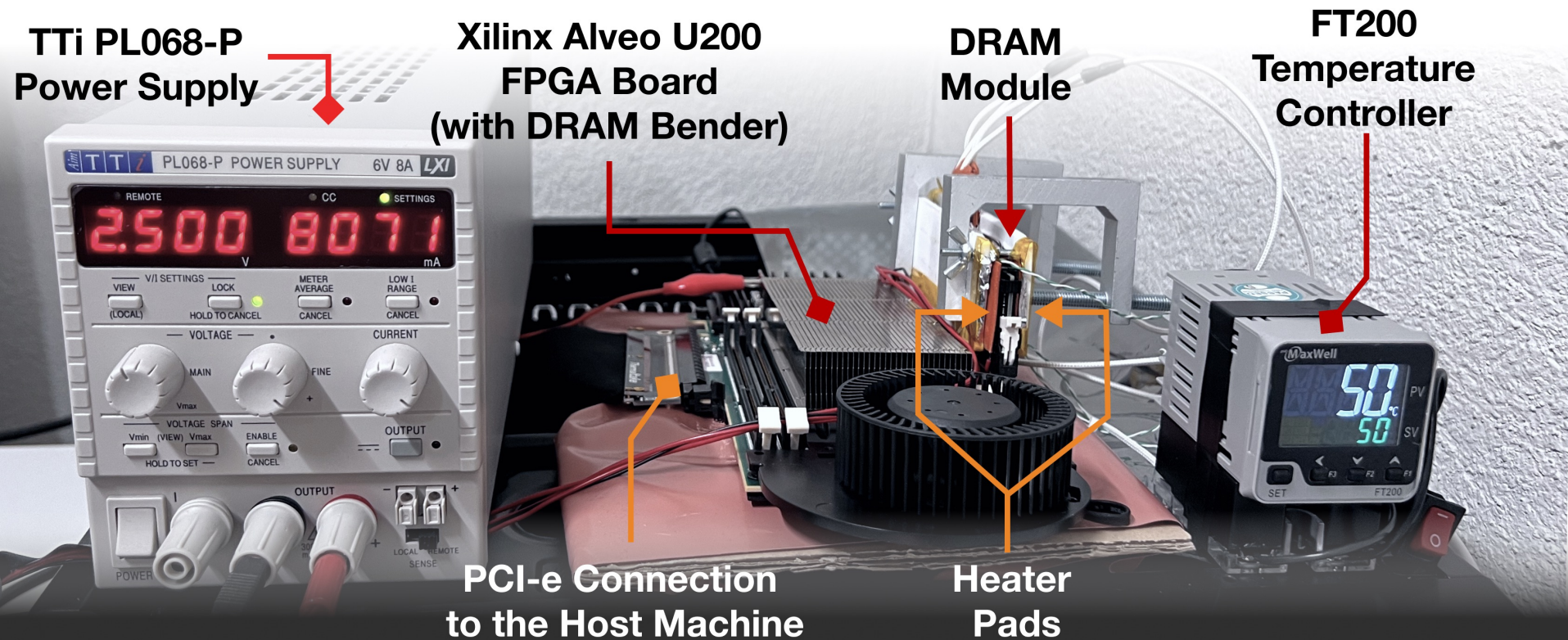
- ❑ Memory technology scaling is not going well (e.g., RowHammer)
- ❑ Many scaling issues demand intelligence in memory
- ❑ Emerging technologies can enable new functions in memory

■ Designs are squeezed in the middle

Memory Technology Scaling

Infrastructures to Understand Scaling Issues

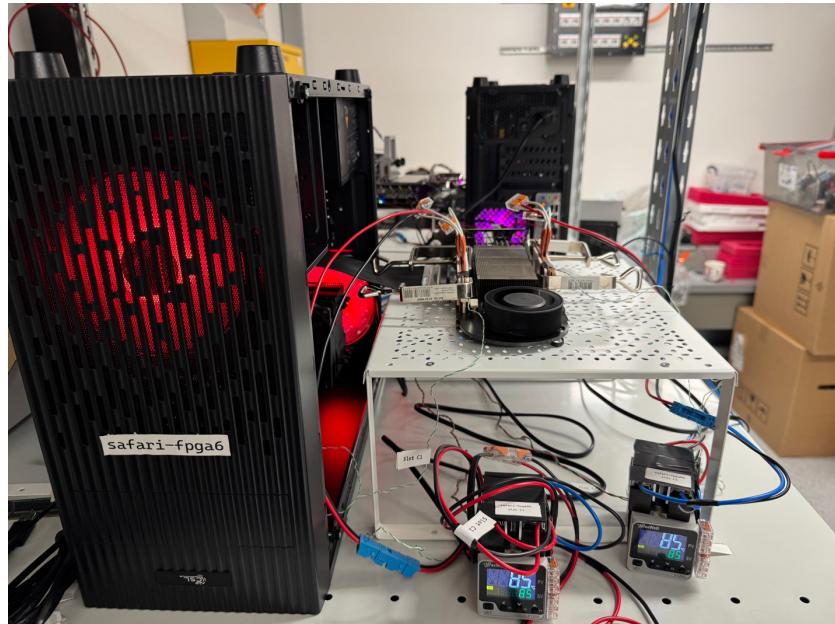
DRAM Bender on a Xilinx Virtex UltraScale+ XCU200



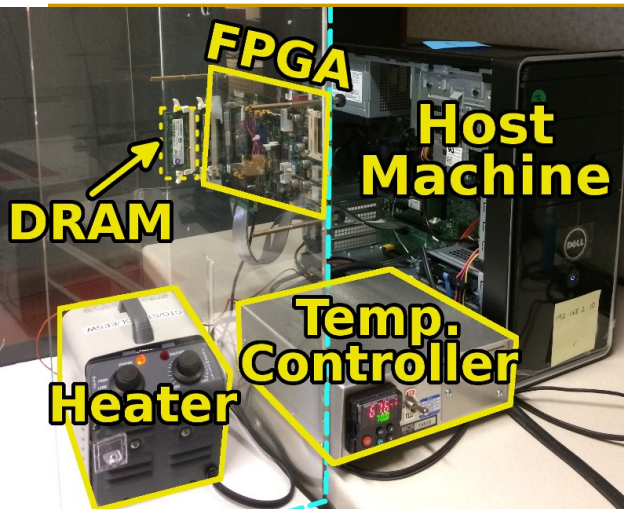
Fine-grained control over **DRAM** commands,
timing parameters ($\pm 1.5\text{ns}$), **temperature ($\pm 0.5^\circ\text{C}$)**,
and **voltage ($\pm 1\text{mV}$)**

*Olgun et al., “**DRAM Bender: An Extensible and Versatile FPGA-based Infrastructure to Easily Test State-of-the-art DRAM Chips**,” in TCAD, 2023. [GitHub: <https://github.com/CMU-SAFARI/DRAM-Bender>]

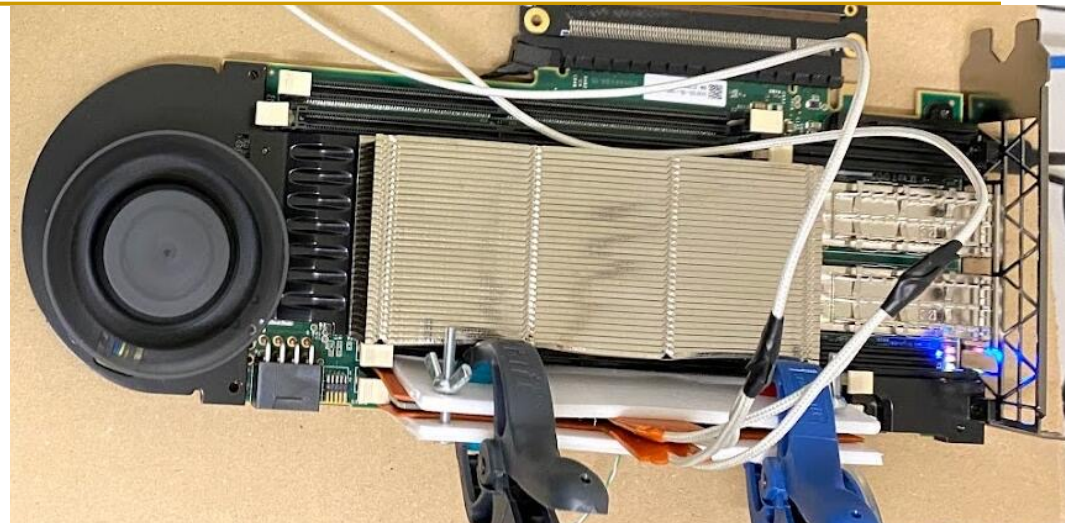
Laboratory for Understanding Memory



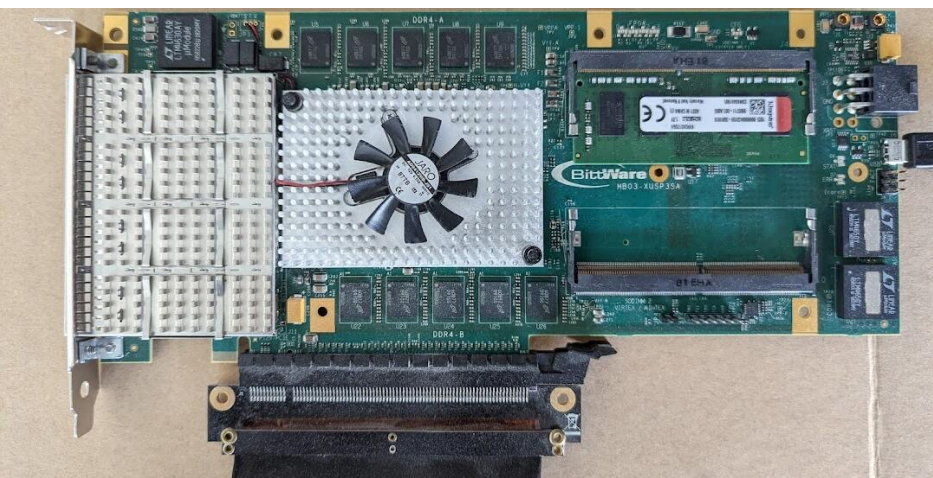
DRAM Testing Infrastructures (I)



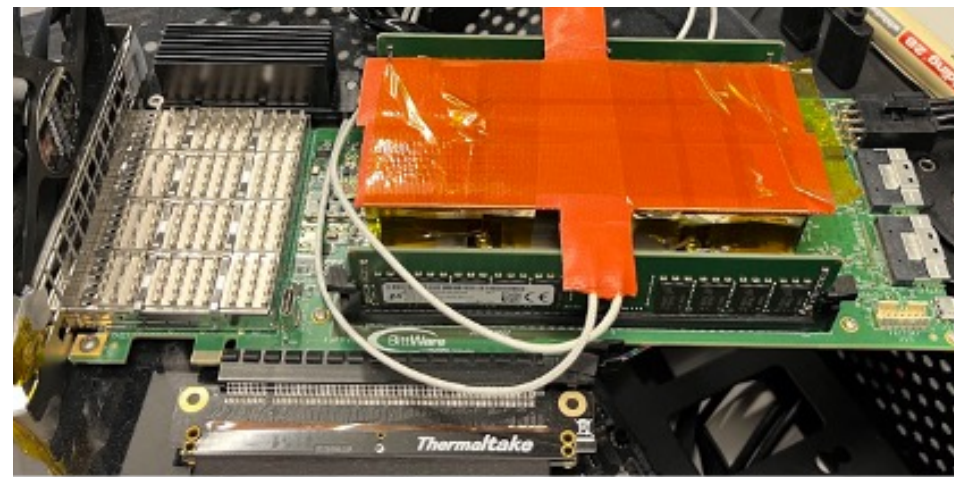
DDR3 DRAM SODIMMs
Xilinx ML605



DDR4 DRAM R/UDIMMs
Xilinx Alveo U200

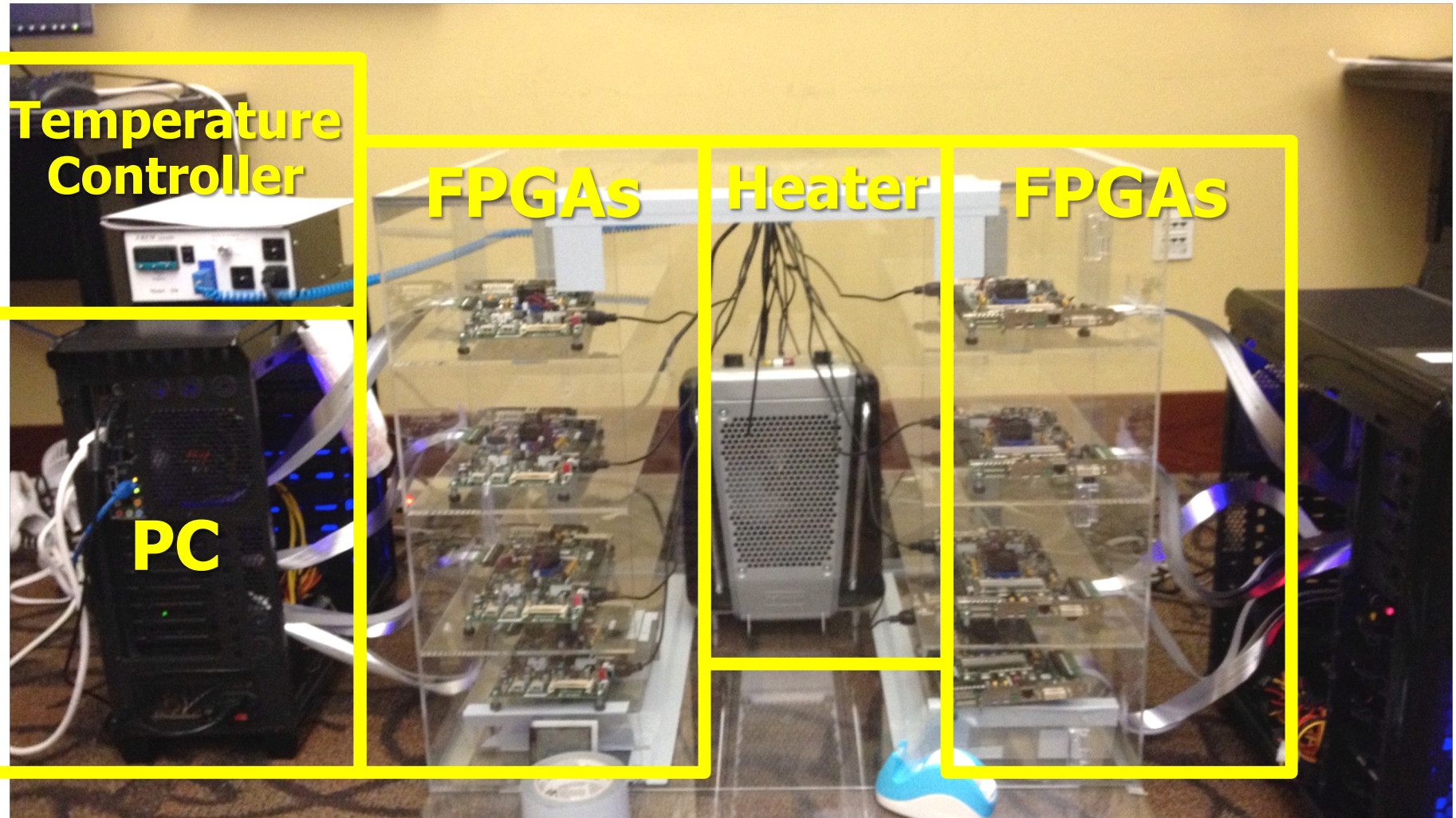


DDR4 DRAM (SODIMM)
Bittware XUSP3S



HBM2 DRAM Chips
Xilinx Alveo U50

DRAM Testing Infrastructures (II)



DRAM Testing Infrastructures (III)



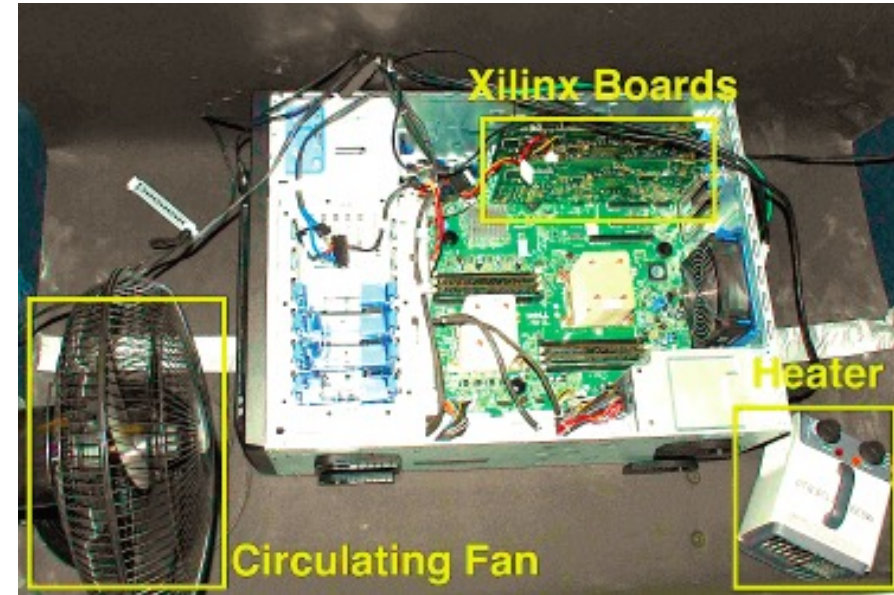
An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms (Liu et al., ISCA 2013)

The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study (Khan et al., SIGMETRICS 2014)

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

Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case (Lee et al., HPCA 2015)

AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems (Qureshi et al., DSN 2015)



SoftMC: Open Source DRAM Infrastructure

- Hasan Hassan, Nandita Vijaykumar, Samira Khan, Saugata Ghose, Kevin Chang, Gennady Pekhimenko, Donghyuk Lee, Oguz Ergin, and Onur Mutlu,
"SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies"
Proceedings of the 23rd International Symposium on High-Performance Computer Architecture (HPCA), Austin, TX, USA, February 2017.
[[Slides \(pptx\) \(pdf\)](#)] [[Lightning Session Slides \(pptx\) \(pdf\)](#)]
[[Full Talk Lecture](#) (39 minutes)]
[[Source Code](#)]

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

Hasan Hassan^{1,2,3} Nandita Vijaykumar³ Samira Khan^{4,3} Saugata Ghose³ Kevin Chang³
Gennady Pekhimenko^{5,3} Donghyuk Lee^{6,3} Oguz Ergin² Onur Mutlu^{1,3}

¹ETH Zürich ²TOBB University of Economics & Technology ³Carnegie Mellon University
⁴University of Virginia ⁵Microsoft Research ⁶NVIDIA Research

DRAM Bender

- Ataberk Olgun, Hasan Hassan, A Giray Yağlıkçı, Yahya Can Tuğrul, Lois Orosa, Haocong Luo, Minesh Patel, Oğuz Ergin, and Onur Mutlu,
"DRAM Bender: An Extensible and Versatile FPGA-based Infrastructure to Easily Test State-of-the-art DRAM Chips"
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), 2023.
[[Extended arXiv version](#)]
[[DRAM Bender Source Code](#)]
[[DRAM Bender Tutorial Video](#) (43 minutes)]

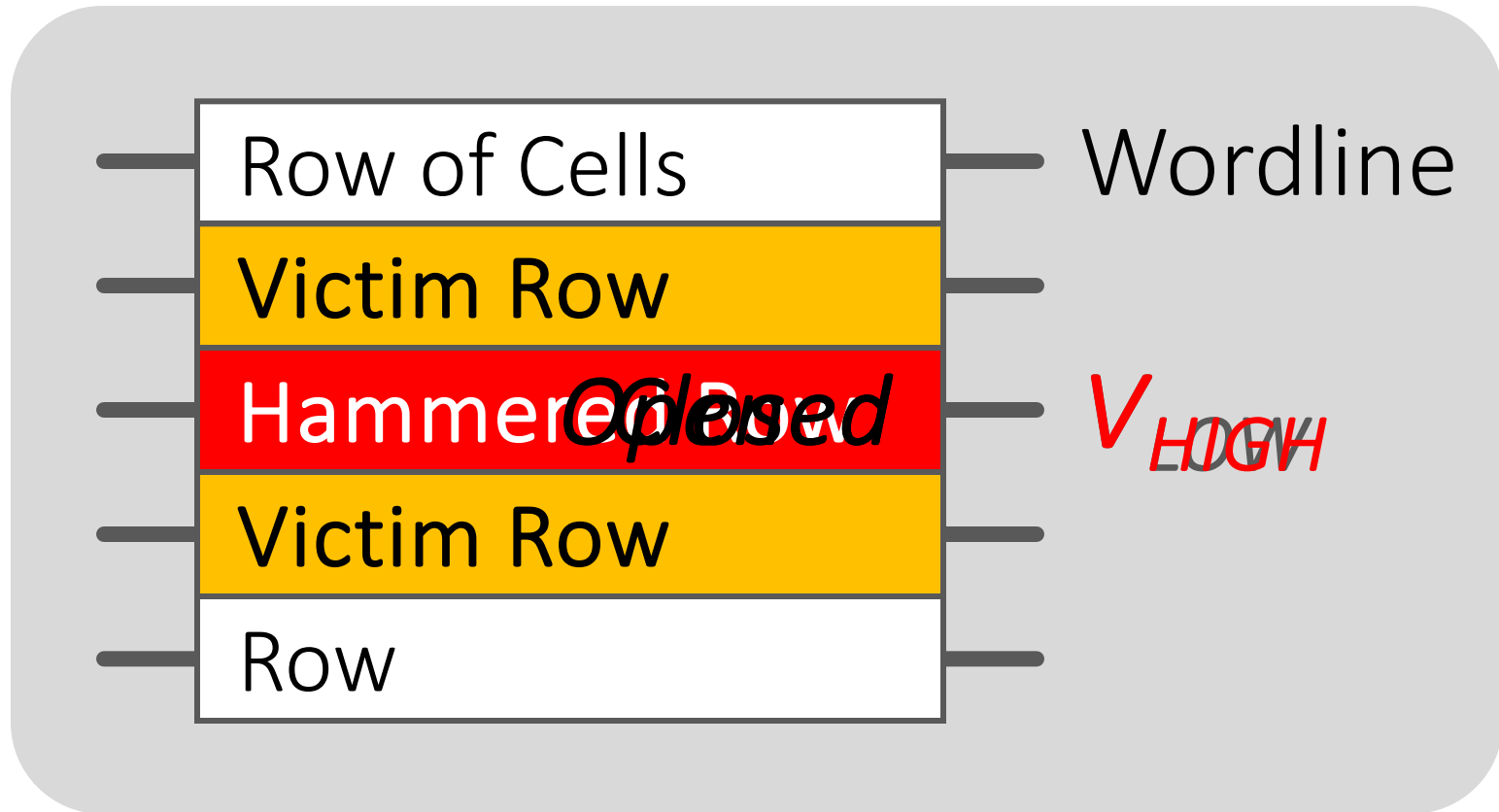
DRAM Bender: An Extensible and Versatile FPGA-based Infrastructure to Easily Test State-of-the-art DRAM Chips

Ataberk Olgun[§] Hasan Hassan[§] A. Giray Yağlıkçı[§] Yahya Can Tuğrul^{§†}
Lois Orosa^{§⊙} Haocong Luo[§] Minesh Patel[§] Oğuz Ergin[†] Onur Mutlu[§]
 [§]*ETH Zürich* [†]*TOBB ETÜ* [⊙]*Galician Supercomputing Center*



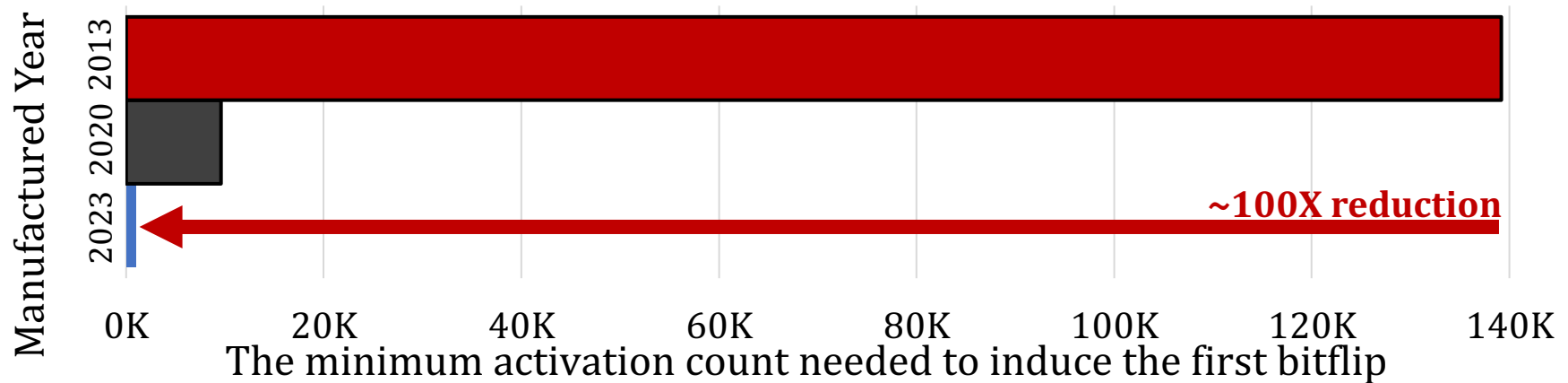
Rowhammer

Modern DRAM is Prone to Disturbance Errors



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**

Read Disturbance Worsens with Scaling



RowHammer [ISCA 2014]

- 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"
Proceedings of the 41st International Symposium on Computer Architecture (ISCA), Minneapolis, MN, June 2014.
[[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](#)). Selected to the ISCA-50 25-Year Retrospective Issue covering 1996-2020 in 2023 ([Retrospective \(pdf\)](#) [Full Issue](#)). Winner of the 2024 IFIP Jean-Claude Laprie Award in dependable computing ([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

Many RowHammer Security Exploits

- One can exploit RowHammer to
- Take over a system
- Read data they do not have access to
- Break out of virtual machine sandboxes
- Corrupt important data → render ML inference useless
- Steal secret data (e.g., crypto keys & ML model parameters)



- Haocong Luo, Ataberk Olgun, Giray Yaglikci, Yahya Can Tugrul, Steve Rhyner, M. Banu Cavlak, Joel Lindegger, Mohammad Sadrosadati, and Onur Mutlu, **"RowPress: Amplifying Read Disturbance in Modern DRAM Chips"**

Proceedings of the 50th International Symposium on Computer Architecture (ISCA), Orlando, FL, USA, June 2023.

[[Slides \(pptx\)](#) ([pdf](#))]

[[Lightning Talk Slides \(pptx\)](#) ([pdf](#))]

[[Lightning Talk Video](#) (3 minutes)]

[[RowPress Source Code and Datasets \(Officially Artifact Evaluated with All Badges\)](#)]

***Officially artifact evaluated as available, reusable and reproducible.
Best artifact award at ISCA 2023. IEEE Micro Top Pick in 2024.***

RowPress: Amplifying Read-Disturbance in Modern DRAM Chips

Haocong Luo Ataberk Olgun A. Giray Yağlıkçı Yahya Can Tuğrul Steve Rhyner
Meryem Banu Cavlak Joël Lindegger Mohammad Sadrosadati Onur Mutlu

ETH Zürich

RowPress vs. RowHammer

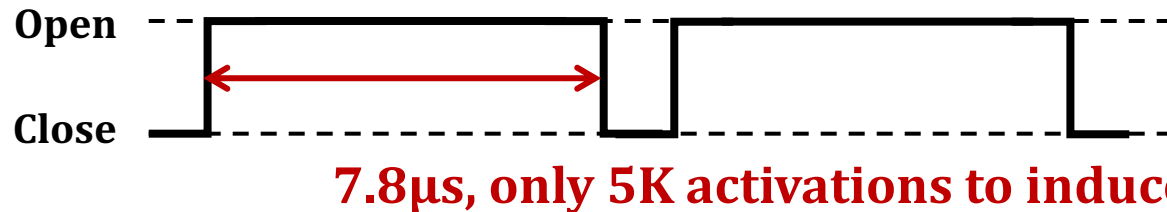
Instead of using a high activation count,

☞ increase the time that the aggressor row stays open

RowHammer
Aggressor Row



RowPress
Aggressor Row



RowPress reduces the number of activations to induce a bitflip by 1-2 orders of magnitude

Main Memory Needs Intelligent Controllers

An “Early” Position Paper [IMW 2013]

- 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/>

Updated Paper 12 Years Later [IMW 2025]

- Onur Mutlu, Ataberk Olgun, and İsmail Emir Yüksel,
**"Memory-Centric Computing: Solving Computing's
Memory Problem"**

*Invited Paper in Proceedings of the 17th IEEE International
Memory Workshop (**IMW**), Monterey, CA, USA, May 2025.*

[Slides (pptx) (pdf)]

Memory-Centric Computing: Solving Computing's Memory Problem

Onur Mutlu Ataberk Olgun İsmail Emir Yüksel

ETH Zürich

<https://www.arxiv.org/pdf/2505.00458>

Industry's Intelligent DRAM Controllers (I)

ISSCC 2023 / SESSION 28 / HIGH-DENSITY MEMORIES /

28.8 A 1.1V 16Gb DDR5 DRAM with Probabilistic-Aggressor Tracking, Refresh-Management Functionality, Per-Row Hammer Tracking, a Multi-Step Precharge, and Core-Bias Modulation for Security and Reliability Enhancement

Woongrae Kim, Chulmoon Jung, Seongnyuh Yoo, Duckhwa Hong, Jeongjin Hwang, Jungmin Yoon, Ohyong Jung, Joonwoo Choi, Sanga Hyun, Mankeun Kang, Sangho Lee, Dohong Kim, Sanghyun Ku, Donhyun Choi, Nogeun Joo, Sangwoo Yoon, Junseok Noh, Byeongyong Go, Cheolhoe Kim, Sunil Hwang, Mihyun Hwang, Seol-Min Yi, Hyungmin Kim, Sanghyuk Heo, Yeonsu Jang, Kyoungchul Jang, Shinho Chu, Yoonna Oh, Kwidong Kim, Junghyun Kim, Soohwan Kim, Jeongtae Hwang, Sangil Park, Junphyo Lee, Inchul Jeong, Joohwan Cho, Jonghwan Kim

SK hynix Semiconductor, Icheon, Korea

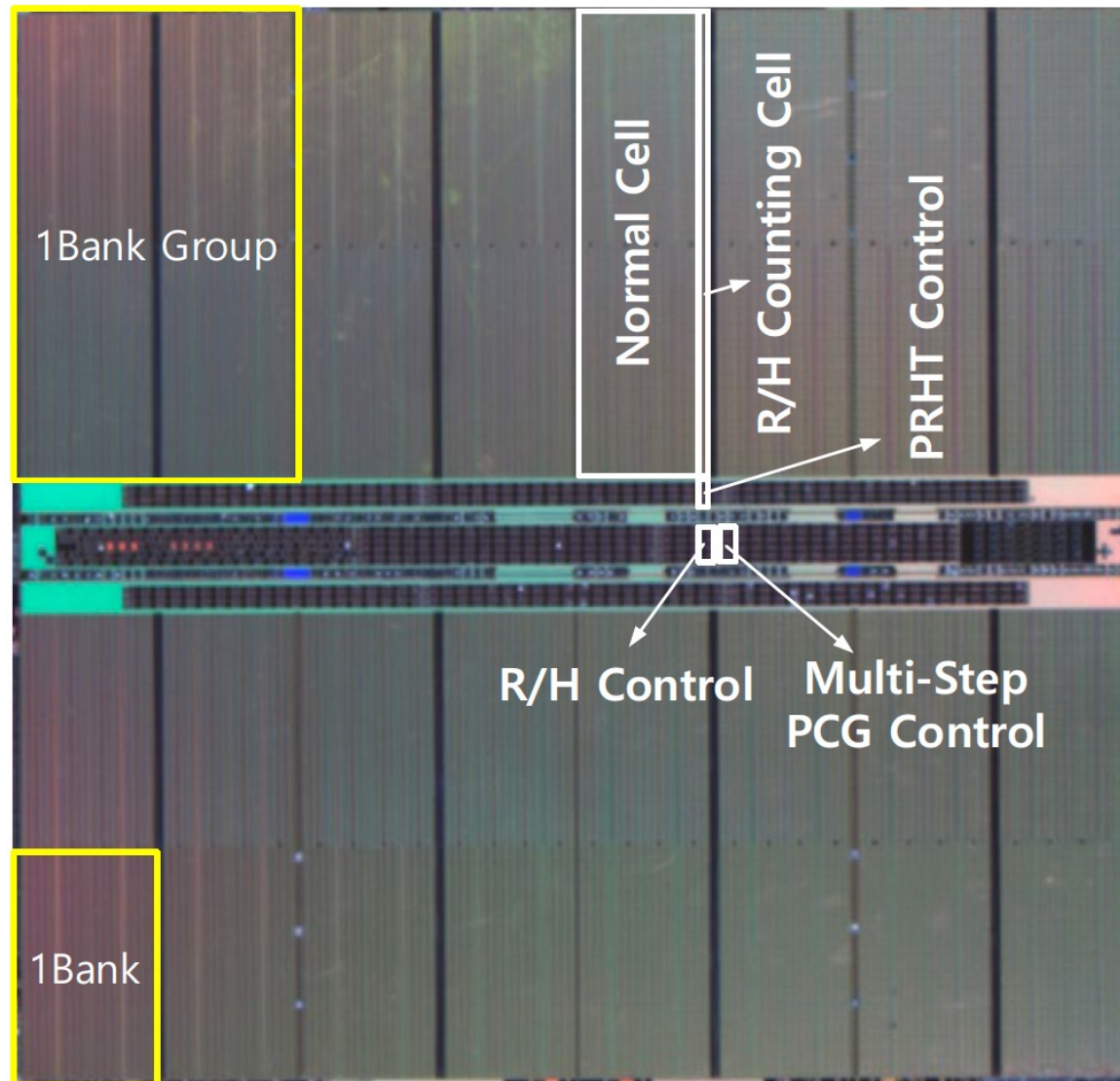


Industry's Intelligent DRAM Controllers (II)

SK hynix Semiconductor, Icheon, Korea

DRAM products have been recently adopted in a wide range of high-performance computing applications: such as in cloud computing, in big data systems, and IoT devices. This demand creates larger memory capacity requirements, thereby requiring aggressive DRAM technology node scaling to reduce the cost per bit [1,2]. However, DRAM manufacturers are facing technology scaling challenges due to row hammer and refresh retention time beyond 1a-nm [2]. Row hammer is a failure mechanism, where repeatedly activating a DRAM row disturbs data in adjacent rows. Scaling down severely threatens reliability since a reduction of DRAM cell size leads to a reduction in the intrinsic row hammer tolerance [2,3]. To improve row hammer tolerance, there is a need to probabilistically activate adjacent rows with carefully sampled active addresses and to improve intrinsic row hammer tolerance [2]. In this paper, row-hammer-protection and refresh-management schemes are presented to guarantee DRAM security and reliability despite the aggressive scaling from 1a-nm to sub 10-nm nodes. The probabilistic-aggressor-tracking scheme with a refresh-management function (RFM) and per-row hammer tracking (PRHT) improve DRAM resilience. A multi-step precharge reinforces intrinsic row-hammer tolerance and a core-bias modulation improves retention time: even in the face of cell-transistor degradation due to technology scaling. This comprehensive scheme leads to a reduced probability of failure, due to row hammer attacks, by 93.1% and an improvement in retention time by 17%.

Industry's Intelligent DRAM Controllers (III)



ISSCC 2023 / SESSION 28 / HIGH-DENSITY MEMORIES

28.8 A 1.1V 16Gb DDR5 DRAM with Probabilistic-Aggressor Tracking, Refresh-Management Functionality, Per-Row Hammer Tracking, a Multi-Step Precharge, and Core-Bias Modulation for Security and Reliability Enhancement

Woongrae Kim, Chulmoon Jung, Seongnyuh Yoo, Duckhwa Hong, Jeongjin Hwang, Jungmin Yoon, Ohyoung Jung, Joonwoo Choi, Sanga Hyun, Mankeun Kang, Sangho Lee, Dohong Kim, Sanghyun Ku, Donhyun Choi, Nogeun Joo, Sangwoo Yoon, Junseok Noh, Byeongyong Go, Cheolhoe Kim, Sunil Hwang, Mihyun Hwang, Seol-Min Yi, Hyungmin Kim, Sanghyuk Heo, Yeonsu Jang, Kyoungchul Jang, Shinho Chu, Yoonna Oh, Kwidong Kim, Junghyun Kim, Soohwan Kim, Jeongtae Hwang, Sangil Park, Junphyo Lee, Inchul Jeong, Joohwan Cho, Jonghwan Kim

SK hynix Semiconductor, Icheon, Korea

Recent Improvements in JEDEC (2024)



Global Standards for the Microelectronics Industry

STANDARDS & DOCUMENTS

COMMITTEES

NEWS

EVENTS & MEETINGS

JOIN

DDR5 SDRAM

JESD79-5C

Apr 2024

Release Number: Version 1.30

Version 1.30

This standard defines the DDR5 SDRAM specification, including features, functionalities, AC and DC characteristics, packages, and ball/signal assignments. The purpose of this Standard is to define the minimum set of requirements for JEDEC compliant 8 Gb through 32 Gb for x4, x8, and x16 DDR5 SDRAM devices. This standard was created based on the DDR4 standards (JESD79-4) and some aspects of the DDR, DDR2, DDR3, and LPDDR4 standards (JESD79, JESD79-2, JESD79-3, and JESD209-4).

Committee(s): [JC-42](#), [JC-42.3](#)

Are Solutions Good?



Evaluation of Industry's Recent Solutions

- **Appears at DRAMSec 2024**

Understanding the Security Benefits and Overheads of Emerging Industry Solutions to DRAM Read Disturbance

Oğuzhan Canpolat^{§†}

A. Giray Yağlıkçı[§]

Geraldo F. Oliveira[§]

Ataberk Olgun[§]

Oğuz Ergin[†]

Onur Mutlu[§]

[§]*ETH Zürich*

[†]*TOBB University of Economics and Technology*

<https://arxiv.org/pdf/2406.19094>

<https://github.com/CMU-SAFARI/ramulator2>

Evaluation of Industry's Recent Solutions

- Oguzhan Canpolat, Abdullah Giray Yaglikci, Geraldo Francisco de Oliveira, Ataberk Olgun, Nisa Bostanci, Ismail Emir Yuksel, Haocong Luo, Oguz Ergin, and Onur Mutlu,
"Chronus: Understanding and Securing the Cutting-Edge Industry Solutions to DRAM Read Disturbance"

Proceedings of the 31st International Symposium on High-Performance Computer Architecture (HPCA), Las Vegas, NV, USA, March 2025.

[Chronus Source Code (Officially Artifact Evaluated with All Badges)]

Officially artifact evaluated as available, functional, and reproduced.

2025 IEEE International Symposium on High-Performance Computer Architecture (HPCA)



Chronus: Understanding and Securing the Cutting-Edge Industry Solutions to DRAM Read Disturbance

Oğuzhan Canpolat^{§†} A. Giray Yağlıkçı[§] Geraldo F. Oliveira[§] Ataberk Olgun[§]
Nisa Bostancı[§] Ismail Emir Yuksel[§] Haocong Luo[§] Oğuz Ergin^{‡†} Onur Mutlu[§]
[§]*ETH Zürich* [†]*TOBB University of Economics and Technology* [‡]*University of Sharjah*

<https://arxiv.org/pdf/2502.12650>

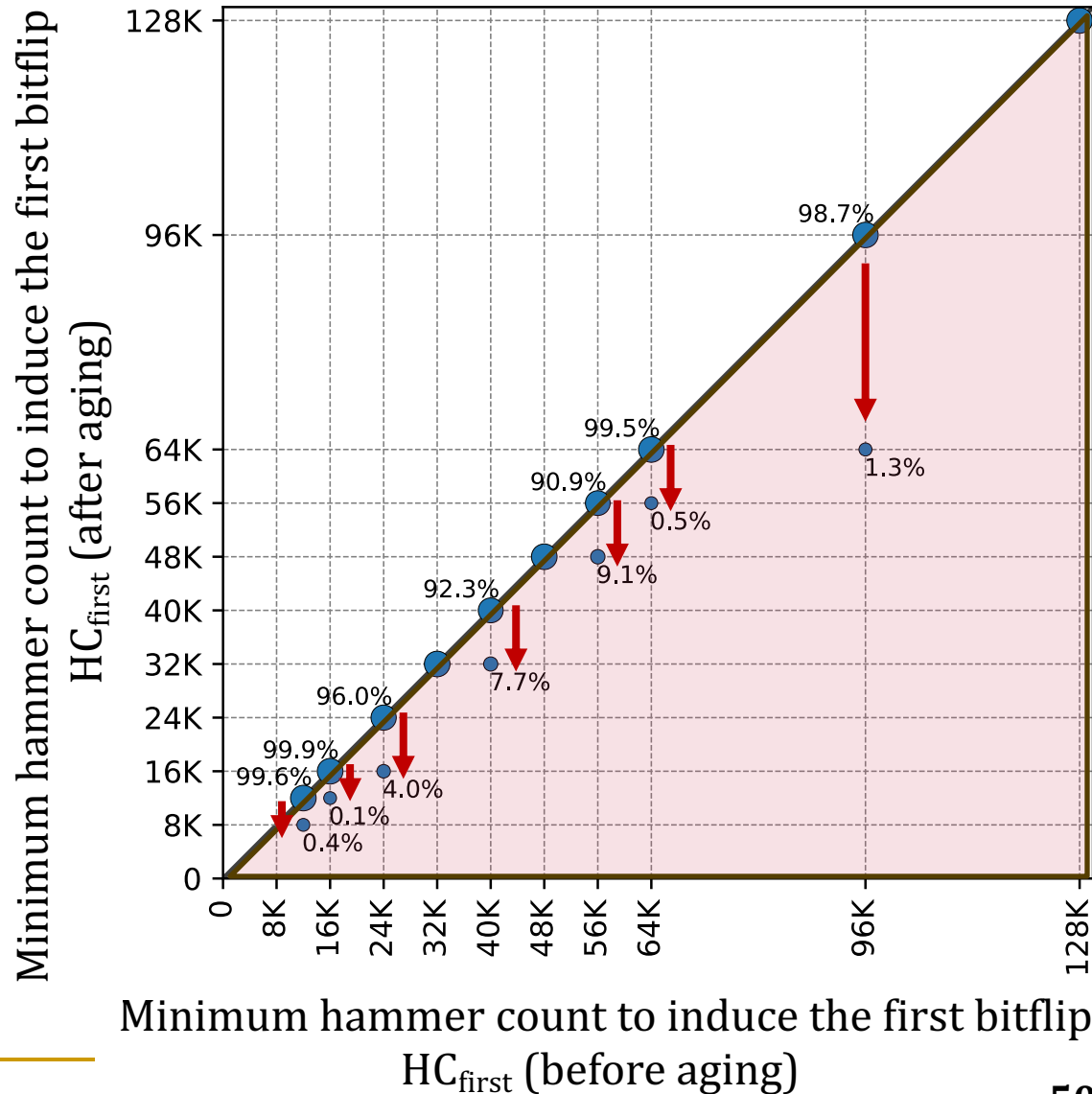
<https://github.com/CMU-SAFARI/Chronus>

More to Come...

RowHammer Becomes Worse with Aging

Preliminary data on aging via 68-day of continuous hammering

Aging can lead to read disturbance bitflips at **smaller** hammer counts



RowHammer (Spatial Variation) Analysis (2024)

- **Appears at HPCA 2024**

Spatial Variation-Aware Read Disturbance Defenses: Experimental Analysis of Real DRAM Chips and Implications on Future Solutions

Abdullah Giray Yağlıkçı Yahya Can Tuğrul Geraldo F. Oliveira
İsmail Emir Yüksel Ataberk Olgun Haocong Luo Onur Mutlu
ETH Zürich

<https://arxiv.org/pdf/2402.18652>

Variable Read Disturbance (2025)

Key Takeaway

The Read Disturbance Threshold (RDT) of a row
changes randomly and unpredictably over time

Accurately identifying RDT is challenging

Variable Read Disturbance (2025)

- **Appears at HPCA 2025**

Variable Read Disturbance:

An Experimental Analysis of Temporal Variation in DRAM Read Disturbance

Ataberk Olgun[†] F. Nisa Bostancı[†] İsmail Emir Yüksel[†] Oğuzhan Canpolat[†] Haocong Luo[†]
Geraldo F. Oliveira[†] A. Giray Yağlıkçı[†] Minesh Patel[‡] Onur Mutlu[†]
ETH Zurich[†] Rutgers University[‡]

Emerging Memories Also Need Intelligent Controllers

- Benjamin C. Lee, Engin Ipek, Onur Mutlu, and Doug Burger,
"Architecting Phase Change Memory as a Scalable DRAM Alternative"
Proceedings of the 36th International Symposium on Computer Architecture (ISCA), pages 2-13, Austin, TX, June 2009. Slides (pdf)
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.

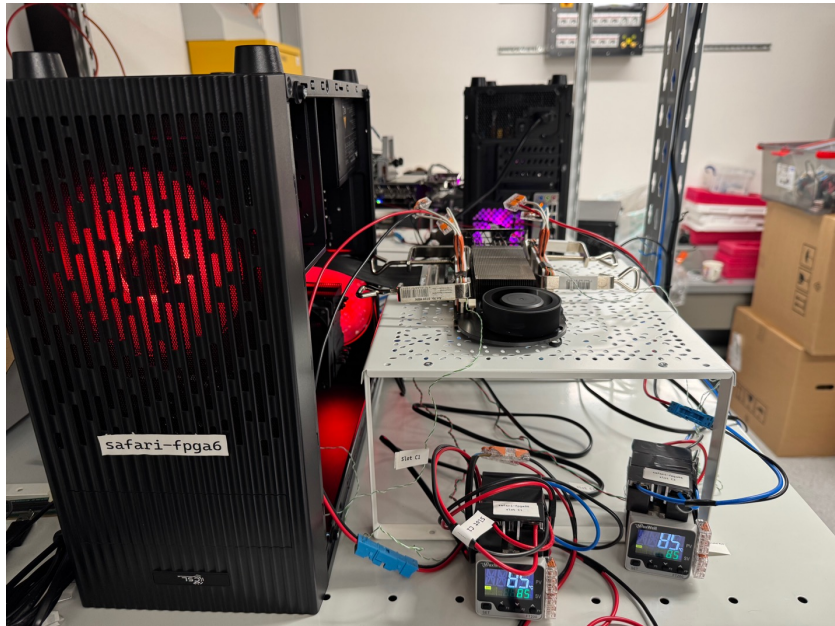
Architecting Phase Change Memory as a Scalable DRAM Alternative

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

Laboratory for Understanding Memory



Read Disturbance Sessions @ HPCA 2025

HPCA 2025

2025 IEEE International Symposium on High-Performance Computer Architecture,
3/1/2025-3/5/2025, Las Vegas, NV, USA



Session 7A (Acacia A and B): Hammering the Odds – 1

Session Chair: Gururaj Saileshwar (Toronto)

- **Variable Read Disturbance: An Experimental Analysis of Temporal Variation in DRAM Read Disturbance**
Ataberk Olgun (ETH Zürich), Nisa Bostanci (ETH Zürich), Ismail Emir Yuksel (ETH Zürich), Giray Yaglikci (ETH Zürich), Geraldo F. Oliveira (ETH Zürich), Haocong Luo (ETH Zürich), Oguzhan Canpolat (ETH Zürich), Minesh Patel (Rutgers University), Onur Mutlu (ETH Zürich)
- **Understanding RowHammer Under Reduced Refresh Latency: Experimental Analysis of Real DRAM Chips and Implications on Future Solutions**
Yahya Can Tuğrul (TOBB ETÜ & ETH Zürich), Giray Yaglikci (ETH Zürich), Ismail Emir Yuksel (ETH Zürich), Ataberk Olgun (ETH Zürich), Oğuzhan Canpolat (TOBB ETÜ & ETH Zürich), Nisa Bostanci (ETH Zürich), Mohammad Sadrosadati (ETH Zürich), Oguz Ergin (TOBB ETÜ), Onur Mutlu (ETH Zürich)
- **Chronus: Understanding and Securing the Cutting-Edge Industry Solutions to DRAM Read Disturbance**
Oğuzhan Canpolat (TOBB ETÜ & ETH Zürich), Giray Yaglikci (ETH Zürich), Geraldo Francisco de Oliveira (ETH Zürich), Ataberk Olgun (ETH Zürich), Nisa Bostanci (ETH Zürich), Ismail Emir Yuksel (ETH Zürich), Haocong Luo (ETH Zürich), Oğuz Ergin (TOBB ETÜ), Onur Mutlu (ETH Zürich)

Session 8A (Acacia A and B): Hammering the Odds – 2

Session Chair: Sudhanva Gurumurthi (AMD)

- **AutoRFM: Scaling Low-Cost In-DRAM Trackers to Ultra-Low Rowhammer Thresholds**
Moinuddin Qureshi (Georgia Tech)
- **DAPPER: A Performance-Attack-Resilient Tracker for RowHammer Defense**
Jeonghyun Woo (The University of British Columbia (UBC)), Prashant J. Nair (The University of British Columbia (UBC))
- **QPRAC: Towards Secure and Practical PRAC-based Rowhammer Mitigation using Priority Queues**
Jeonghyun Woo (The University of British Columbia (UBC)), Shaopeng (Chris) Lin (University of Toronto), Prashant J. Nair (The University of British Columbia (UBC)), Aamer Jaleel (NVIDIA), Gururaj Saileshwar (University of Toronto)

Tuesday, March 4th, 11am and 2pm

Read Disturbance Papers @ ASPLOS 2025



Rotterdam, The Netherlands — March 30- April 3, 2025.

Session 4B: Memory & Storage +

LOCATION: VAN OLDENBARNEVELD

Marionette: A RowHammer Attack via Row Coupling

Seungmin Baek (Seoul National University),
Minbok Wi (Seoul National University),
Seonyong Park (Seoul National University),
Hwayong Nam (Seoul National University),
Michael Jaemin Kim (Seoul National University),
Nam Sung Kim (University of Illinois),
Jung Ho Ahn (Seoul National University)

[Paper](#)

MOAT: Securely Mitigating Rowhammer with Per-Row Activation Counters

Moinuddin Qureshi (Georgia Institute of Technology),
Salman Qazi (Google)

[Paper](#)

HyperHammer: Breaking Free from KVM-Enforced Isolation

Wei Chen (Peking University), Zhi Zhang (University of Western Australia), Xin Zhang (Peking University), Qingni Shen (Peking University), Yuval Yarom (Ruhr University Bochum), Daniel Genkin (Georgia Institute of Technology), Chen Yan (Peking University), Zhe Wang (SKLP, Institute of Computing Technology, Chinese Academy of Sciences, Zhongguancun Laboratory)

[Paper](#)

Read Disturbance Session @ ISCA 2025



Session 5A: RowHammer

Location: Okuma Auditorium (Main)

Session Chair: TBA

08:30 AM – 08:50 AM

MoPAC: Efficiently Mitigating Rowhammer with Probabilistic Activation Counting

Suhas Vittal, Salman Qazi, Poulami Das, Moin Qureshi

08:50 AM – 09:10 AM

When Mitigations Backfire: Timing Channel Attacks and Defense for PRAC-Based Rowhammer Mitigations

Jeonghyun Woo, Joyce Qu, Gururaj Saileshwar, Prashant Nair

09:10 AM – 09:30 AM

PuDHammer: Experimental Analysis of Read Disturbance Effects of Processing-using-DRAM in Real DRAM Chips

Ismail Emir Yuksel, Akash Sood, Ataberk Olgun, O?uzhan Canpolat, Haocong Luo, Nisa Bostanci, Mohammad Sadrosadati, Giray Yaglikci, Onur Mutlu

09:30 AM – 09:50 AM

DREAM: Enabling Low-Overhead Rowhammer Mitigation via Directed Refresh Management

Hritvik Taneja, Moin Qureshi

Read Disturbance Papers @ DRAMSec 2025

Accepted papers

Softhammer: Exploiting Rowhammer Bit Flips without Crashing

Finn de Ridder, Patrick Jattke, Kaveh Razavi

Rubber Mallet: A Study of High Frequency Localized Bit Flips and Their Impact on Security

Andrew J. Adiletta, Zane Weissman, Fatemeh Khojasteh Dana, Berk Sunar, Shahin Tajik

CnC-PRAC: Coalesce, not Cache, Per Row Activation Counts for an Efficient in-DRAM Rowhammer Mitigation

Chris S. Lin, Jeonghyun Woo, Prashant J. Nair, Gururaj Saileshwar

A Simulation-based Evaluation Framework for Inter-VM RowHammer Mitigation Techniques

Hidemasa Kawasaki, Soramichi Akiyama

Sudoku: Decomposing DRAM Address Mapping into Component Functions

Minbok Wi, Seungmin Baek, Seonyong Park, Mattan Erez, Jung Ho Ahn

Counterpoint: One-Hot Counting for PRAC-Based RowHammer Mitigation

Shih-Lien Lu, Jeonghyun Woo, Prashant J. Nair

DRFM and the Art of Rowhammer Sampling

Salman Qazi, Moinuddin Qureshi

Keynote

Panel

*Is PRAC a good solution to DRAM read disturbance? Are we missing anything?
Can we (and should we) do much better (and hopefully not worse)?*

Workshop chairs

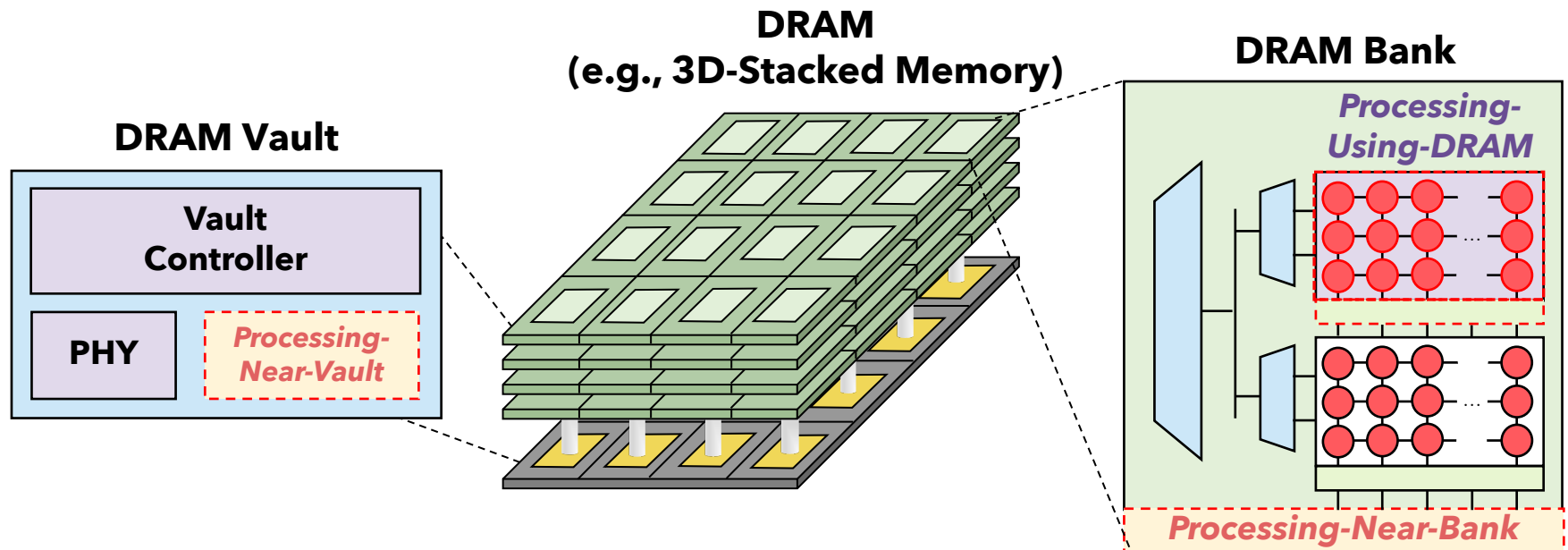
- Onur Mutlu, ETH Zürich
- Kuljit Bains, NVIDIA

Processing in Memory: Two Types

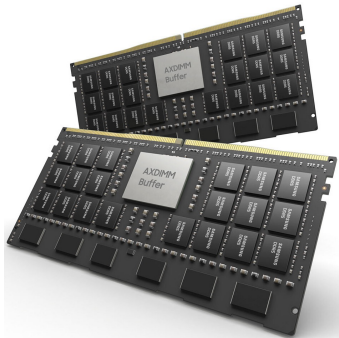
1. Processing **near** Memory
2. Processing **using** Memory

Processing-in-Memory: Two Types

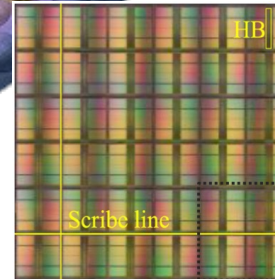
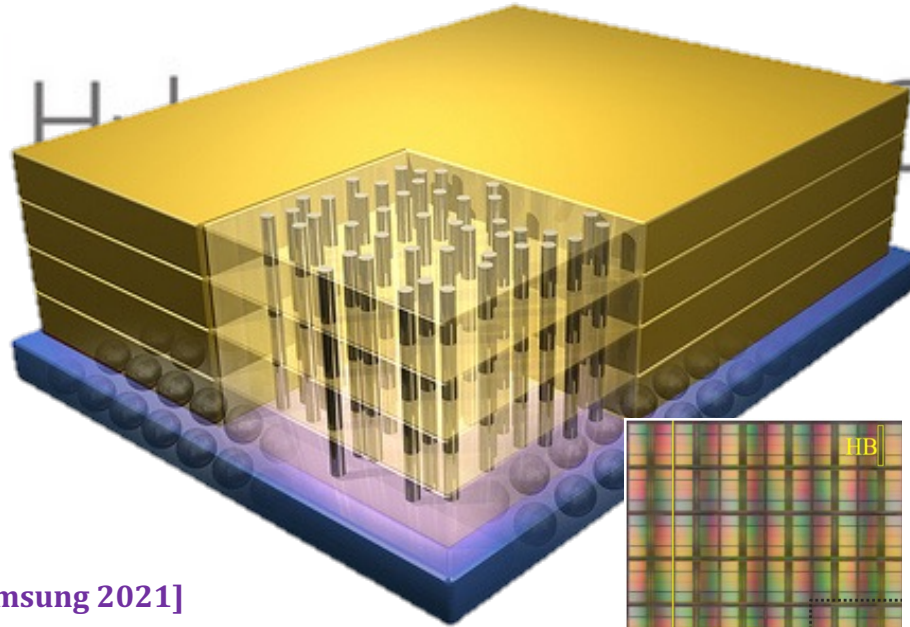
- 1 Processing-Near-Memory:** Computation logic is added to the same die as memory or to the logic layer of 3D-stacked memory
- 2 Processing-Using-Memory:** uses the operational principles of memory cells & circuitry to perform computation



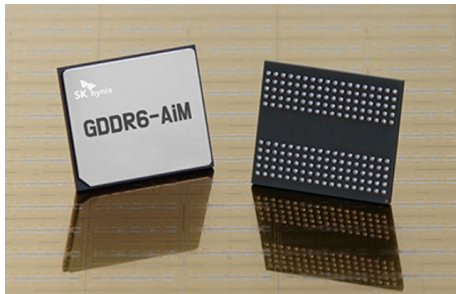
Processing-in-Memory Landscape Today



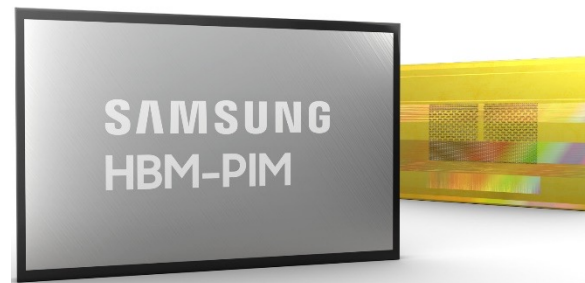
[Samsung 2021]



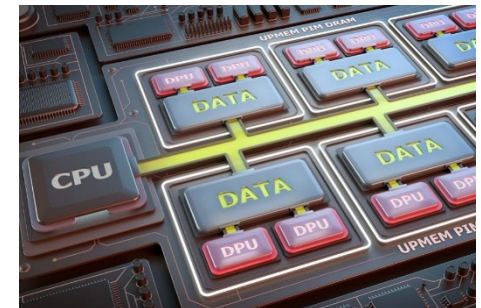
[Alibaba 2022]



[SK Hynix 2022]



[Samsung 2021]



[UPMEM 2019]

Processing-in-Memory Landscape Today

IEEE COMPUTER ARCHITECTURE LETTERS, VOL. 22, NO. 1, JANUARY-JUNE

Computational CXL-Memory Solution for Accelerating Memory-Intensive Applications

Joonseop Sim^{ID}, Soohong Ahn^{ID}, Taeyoung Ahn^{ID},
Seungyong Lee^{ID}, Myunghyun Rhee, Jooyoung Kim^{ID},
Kwangsik Shin, Donguk Moon^{ID},
Euseok Kim, and Kyoung Park^{ID}

Abstract—CXL interface is the up-to-date technology that enables effective memory expansion by providing a memory-sharing protocol in configuring heterogeneous devices. However, its limited physical bandwidth can be a significant bottleneck for emerging data-intensive applications. In this work, we propose a novel CXL-based memory disaggregation architecture with a real-world prototype demonstration, which overcomes the bandwidth limitation of the CXL interface using near-data processing. The experimental results demonstrate that our design achieves up to $1.9\times$ better performance/power efficiency than the existing CPU system.

Index Terms—Compute express link (CXL), near-data-processing (NDP)

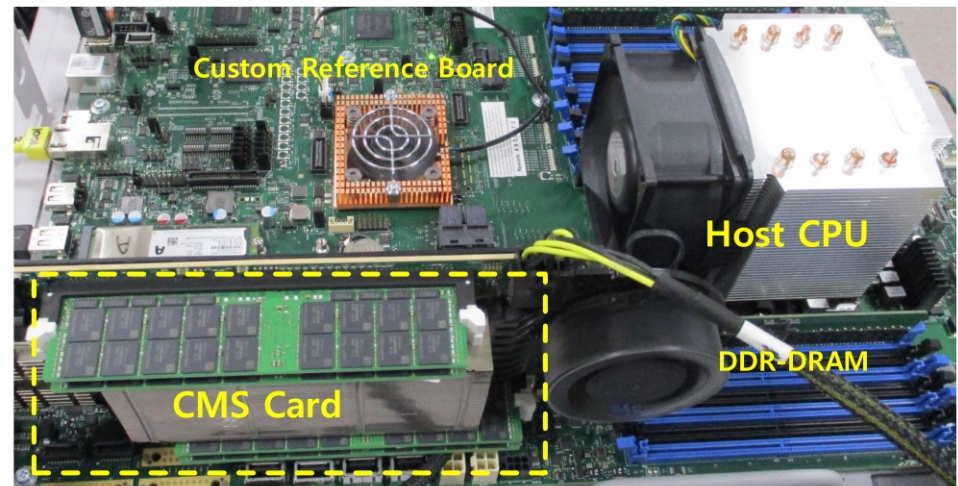


Fig. 6. FPGA prototype of proposed CMS card.

Processing-in-Memory Landscape Today

Samsung Processing in Memory Technology at Hot Chips 2023

By Patrick Kennedy - August 28, 2023

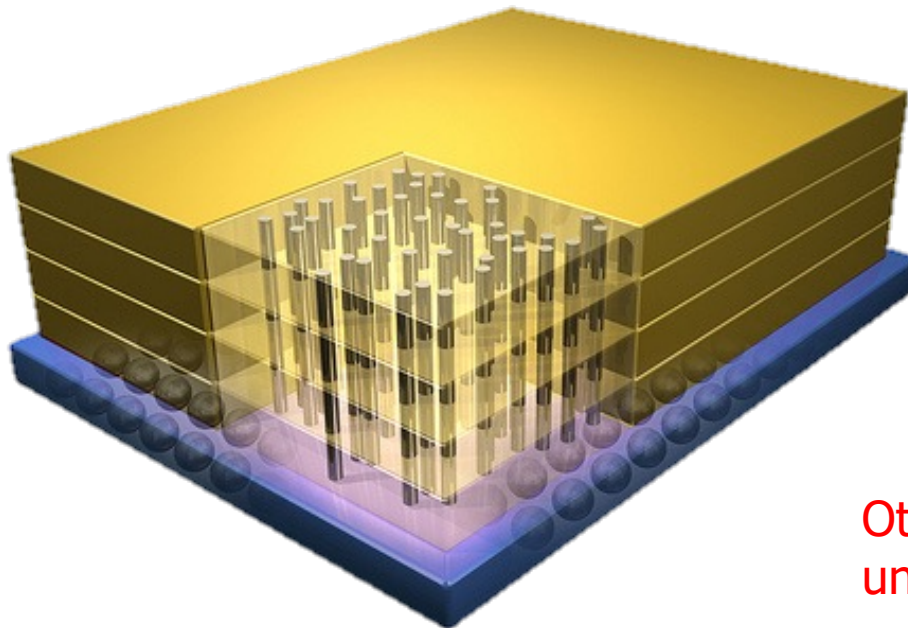


Samsung PIM PNM For Transformer Based AI HC35_Page_24

Opportunity: 3D-Stacked Logic+Memory



Hybrid Memory Cube
C O N S O R T I U M



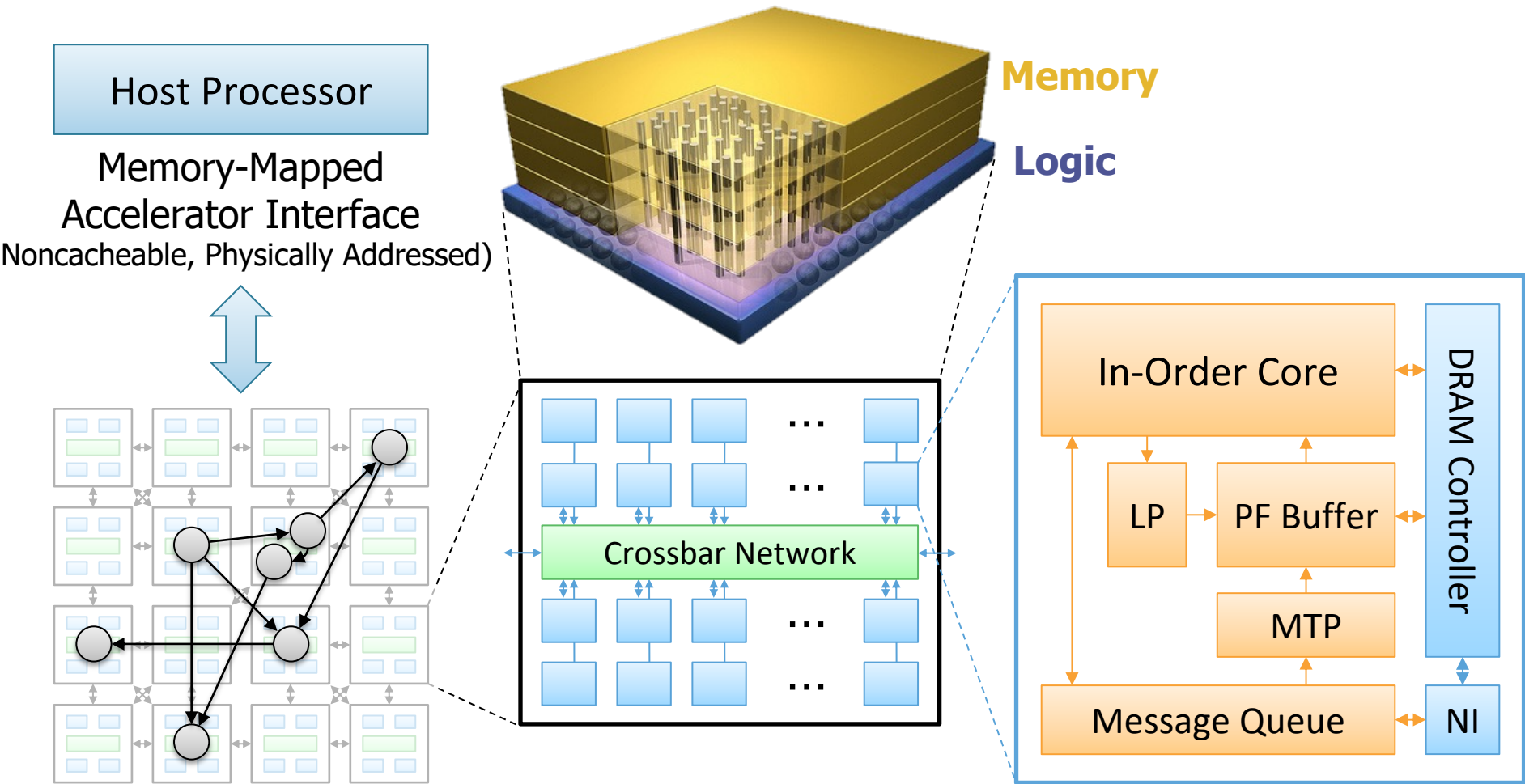
Memory

Logic

Other "True 3D" technologies
under development

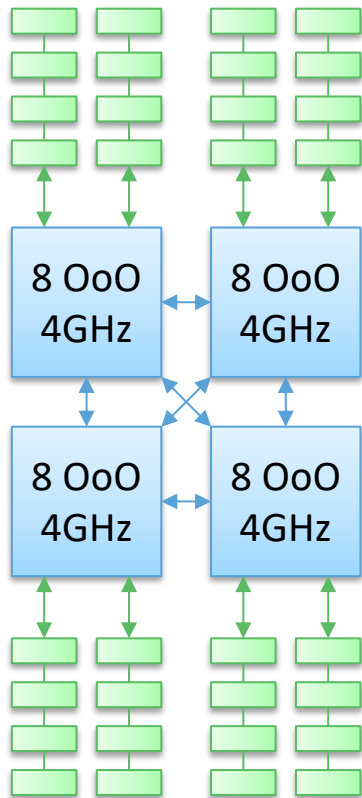
Tesseract System for Graph Processing

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



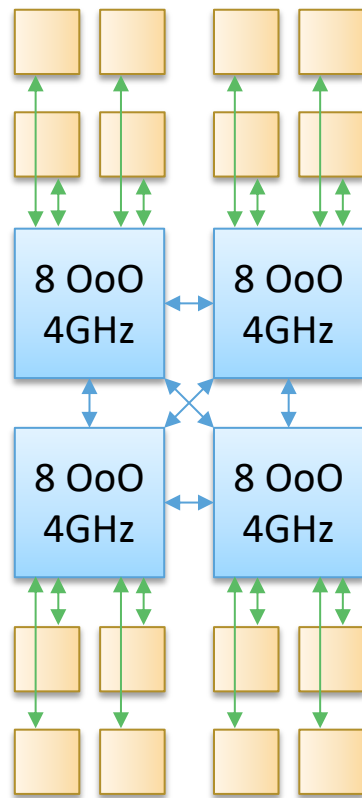
Evaluated Systems

DDR3-OoO



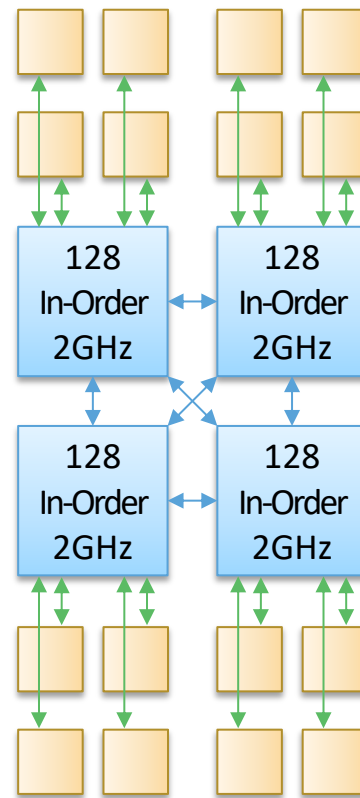
102.4GB/s

HMC-OoO



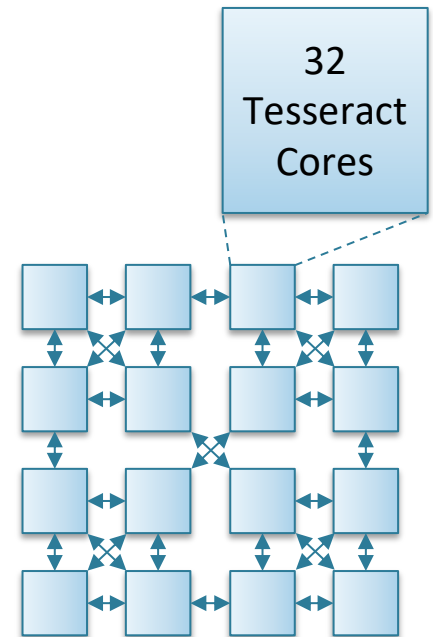
640GB/s

HMC-MC



640GB/s

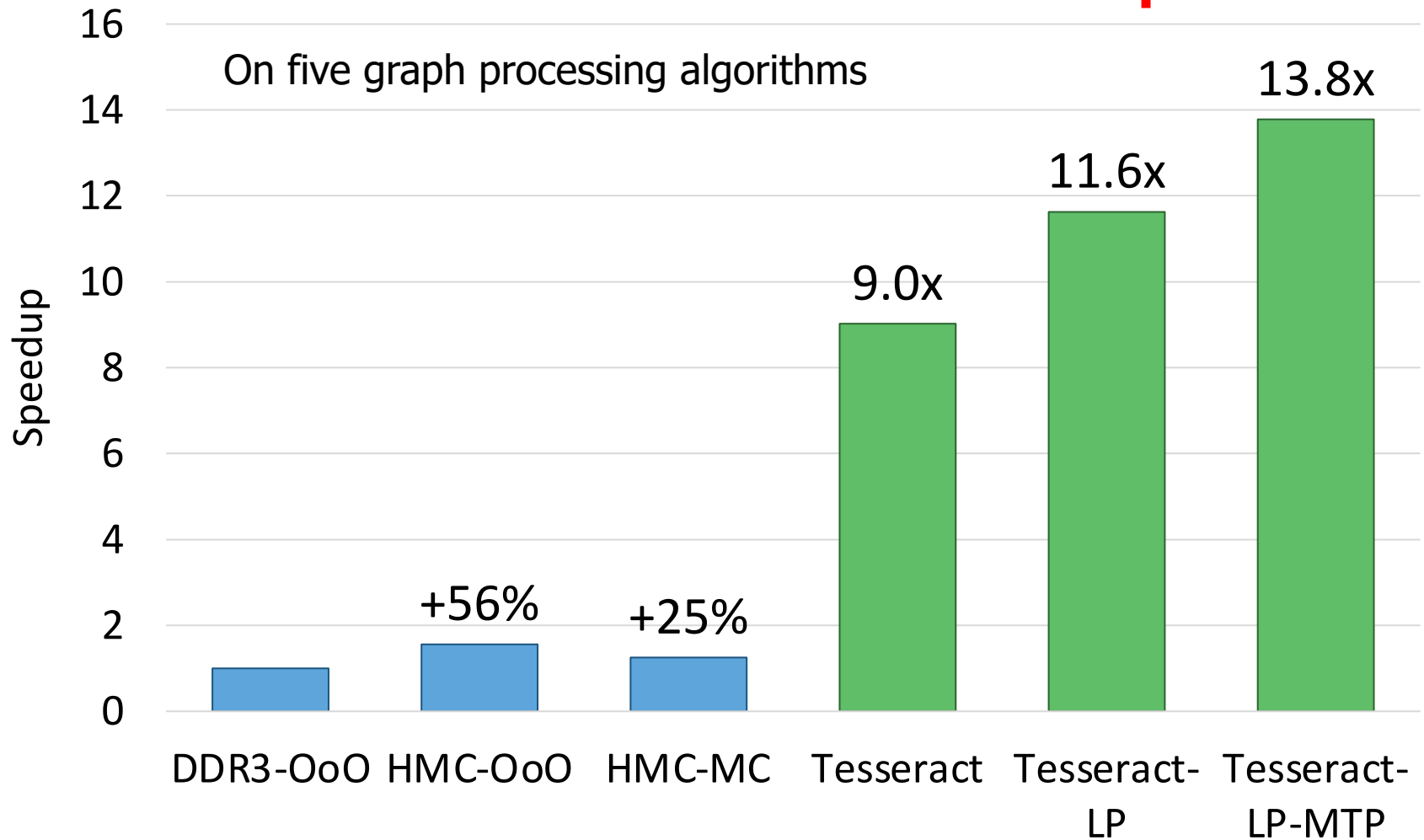
Tesseract



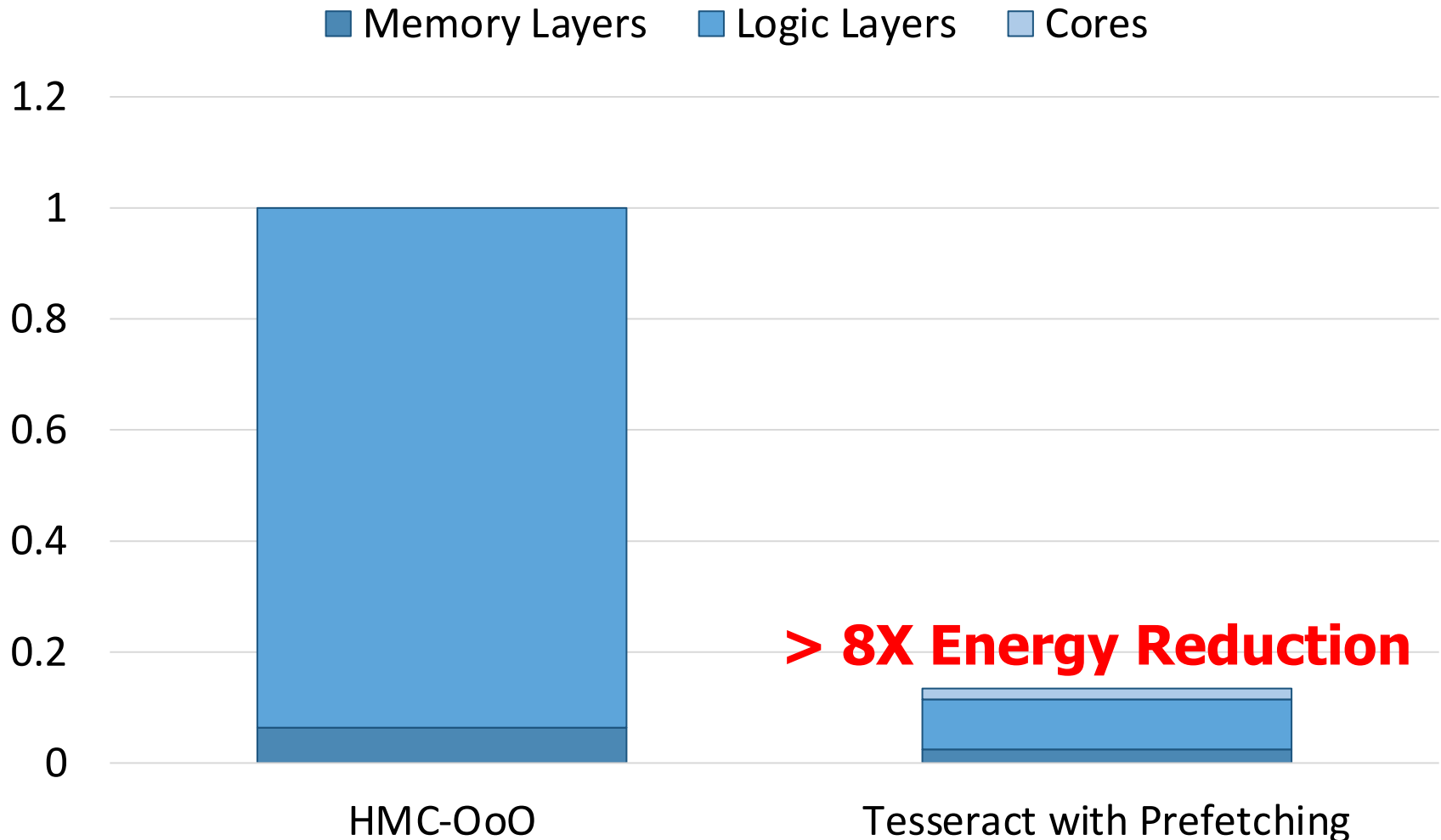
8TB/s

Tesseract Graph Processing Performance

>13X Performance Improvement



Tesseract Graph Processing System Energy



More on Tesseract

- Junwhan Ahn, Sungpack Hong, Sungjoo Yoo, Onur Mutlu, and Kiyoun Choi,
"A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing"
Proceedings of the 42nd International Symposium on Computer Architecture (ISCA), Portland, OR, June 2015.
[Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)]
Top Picks Honorable Mention by IEEE Micro.
Selected to the ISCA-50 25-Year Retrospective Issue covering 1996-2020 in 2023 (Retrospective (pdf) Full Issue).

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

Junwhan Ahn Sungpack Hong[§] Sungjoo Yoo Onur Mutlu[†] Kiyoun 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

A Short Retrospective @ 50 Years of ISCA

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

Junwhan Ahn[†] Sungpack Hong[‡] Sungjoo Yoo[▽] Onur Mutlu[§] Kiyoung Choi[▽]
[†]Google DeepMind [‡]Oracle Labs [§]ETH Zürich [▽]Seoul National University

Abstract—Our ISCA 2015 paper [1] provides a new programmable processing-in-memory (PIM) architecture and system design that can accelerate key data-intensive applications, with a focus on graph processing workloads. Our major idea was to completely rethink the system, including the programming model, data partitioning mechanisms, system support, instruction set architecture, along with near-memory execution units and their communication architecture, such that an important workload can be accelerated at a maximum level using a distributed system of well-connected near-memory accelerators. We built our accelerator system, Tesseract, using 3D-stacked memories with logic layers, where each logic layer contains general-purpose processing cores and control logic for each core using a message-passing programming model. Cores could be specialized for graph processing (or any other application to be accelerated).

To our knowledge, our paper was the first to completely design a near-memory accelerator system from scratch such that it is both generally programmable and specifically customizable to accelerate important applications, with a case study on major graph processing workloads. Existing work in academia and industry showed that similar approaches to system design can greatly benefit both graph processing workloads and other applications, such as machine learning, for which ideas from Tesseract seem to have been influential.

This short retrospective provides a brief analysis of our ISCA 2015 paper and its impact. We briefly describe the major ideas and contributions of the work, discuss later works that built on it or were influenced by it, and make some educated guesses on what the future may bring on PIM and accelerator systems.

I. BACKGROUND, APPROACH & MINDSET

We started our research when 3D-stacked memories (e.g., [2–4]) were viable and seemed to have promise for building effective and practical processing-near-memory systems. Such near-memory systems could lead to improvements, but there was little to no research that examined how an accelerator could be completely (re-)designed using such near-memory technology, from its hardware architecture to its programming model and software system, and what the performance and energy benefits could be of such a re-design. We set out to answer these questions in our ISCA 2015 paper [1].

We followed several major principles to design our accelerator from the ground up. We believe these principles are still important: a major contribution and influence of our work was in putting all of these together in a cohesive full-system design and demonstrating the large performance and energy benefits that can be obtained from such a design. We see a similar approach in many modern large-scale accelerator systems in machine learning today (e.g., [5–9]). Our principles are:

1. *Near-memory execution* to enable/exploit the high data access bandwidth modern workloads (e.g., graph processing) need and to reduce data movement and access latency.
2. *General programmability* so that the system can be easily adopted, extended, and customized for many workloads.
3. *Maximal acceleration capability* to maximize the performance and energy benefits. We set ourselves free from backward compatibility and cost constraints. We aimed to completely re-design the system stack. Our goal was to explore the maximal performance and energy efficiency benefits we can gain from a near-memory accelerator if we had complete freedom to change things as much as we needed. We contrast this approach to the *minimal intrusion* approach we also explored in a separate ISCA 2015 paper [10].
4. *Customizable to specific workloads*, such that we can maximize acceleration benefits. Our focus workload was graph

analytics/processing, a key workload at the time and today. However, our design principles are not limited to graph processing and the system we built is customizable to other workloads as well, e.g., machine learning, genome analysis.

5. *Memory-capacity-proportional performance*, i.e., processing capability should proportionally grow (i.e., scale) as memory capacity increases and vice versa. This enables scaling of data-intensive workloads that need both memory and compute.

6. *Exploit new technology (3D stacking)* that enables tight integration of memory and logic and helps multiple above principles (e.g., enables customizable near-memory acceleration capability in the logic layer of a 3D-stacked memory chip).

7. *Good communication and scaling capability* to support scalability to large dataset sizes and to enable memory-capacity-proportional performance. To this end, we provided scalable communication mechanisms between execution cores and carefully interconnected small accelerator chips to form a large distributed system of accelerator chips.

8. *Maximal and efficient use of memory bandwidth* to supply the high-bandwidth data access that modern workloads need. To this end, we introduced new, specialized mechanisms for prefetching and a programming model that helps leverage application semantics for hardware optimization.

II. CONTRIBUTIONS AND INFLUENCE

We believe the major contributions of our work were 1) complete rethinking of how an accelerator system should be designed to enable maximal acceleration capability, and 2) the design and analysis of such an accelerator with this mindset and using the aforementioned principles to demonstrate its effectiveness in an important class of workloads.

One can find examples of our approach in modern large-scale machine learning (ML) accelerators, which are perhaps the most successful incarnation of scalable near-memory execution architectures. ML infrastructure today (e.g., [5–9]) consists of accelerator chips, each containing compute units and high-bandwidth memory tightly packaged together, and features scale-up capability enabled by connecting thousands of such chips with high-bandwidth interconnection links. The system-wide rethinking that was done to enable such accelerators and many of the principles used in such accelerators resemble our ISCA 2015 paper's approach.

The “memory-capacity-proportional performance” principle we explored in the paper shares similarities with how ML workloads are scaled up today. Similar to how we carefully sharded graphs across our accelerator chips to greatly improve effective memory bandwidth in our paper, today's ML workloads are sharded across a large number of accelerators by leveraging data/model parallelism and optimizing the placement to balance communication overheads and compute scalability [11, 12]. With the advent of large generative models requiring high memory bandwidth for fast training and inference, the scaling behavior where capacity and bandwidth are scaled together has become an essential architectural property to support modern data-intensive workloads.

The “maximal acceleration capability” principle we used in Tesseract provides much larger performance and energy improvements and better customization than the “minimalist” approach that our other ISCA 2015 paper on *PIM-Enabled Instructions* [10] explored: “minimally change” an existing

system to incorporate (near-memory) acceleration capability to ease programming and keep costs low. So far, the industry has more widely adopted the maximal approach to overcome the pressing scaling bottlenecks of major workloads. The key enabler that bridges the programmability gap between the maximal approach favoring large performance & energy benefits and the minimal approach favoring ease of programming is compilation techniques. These techniques lower well-defined high-level constructs into lower-level primitives [12, 13]; our ISCA 2015 papers [1, 10] and a follow-up work [14] explore them lightly. We believe that a good programming model that enables large benefits coupled with support for it across the entire system stack (including compilers & hardware) will continue to be important for effective near-memory system and accelerator designs [14]. We also believe that the maximal versus minimal approaches that are initially explored in our two ISCA 2015 papers is a useful way of exploring emerging technologies (e.g., near-memory accelerators) to better understand the tradeoffs of system designs that exploit such technologies.

III. INFLUENCE ON LATER WORKS

Our paper was at the beginning of a proliferation of scalable near-memory processing systems designed to accelerate key applications (see [15] for many works on the topic). Tesseract has inspired many near-memory system ideas (e.g., [16–28]) and served as the de facto comparison point for such systems, including near-memory graph processing accelerators that built on Tesseract and improved various aspects of Tesseract. Since machine learning accelerators that use high-bandwidth memory (e.g., [5, 29]) and industrial PIM prototypes (e.g., [30–41]) are now in the market, near-memory processing is no longer an “eccentric” architecture it used to be when Tesseract was originally published.

Graph processing & analytics workloads remain as an important and growing class of applications in various forms, ranging from large-scale industrial graph analysis engines (e.g., [42]) to graph neural networks [43]. Our focus on large-scale graph processing in our ISCA 2015 paper increased attention to this domain in the computer architecture community, resulting in subsequent research on efficient hardware architectures for graph processing (e.g., [44–46]).

IV. SUMMARY AND FUTURE OUTLOOK

We believe that our ISCA 2015 paper's principled rethinking of system design to accelerate an important class of data-intensive workloads provided significant value and enabled/influenced a large body of follow-on works and ideas. We expect that such rethinking of system design for key workloads, especially with a focus on “maximal acceleration capability,” will continue to be critical as pressing technology and application scaling challenges increasingly require us to think differently to substantially improve performance and energy (as well as other metrics). We believe the principles exploited in Tesseract are fundamental and they will remain useful and likely become even more important as systems memory access and computation demands of future workloads. We also project that as hardware substrates for near-memory acceleration (e.g., 3D stacking, in-DRAM computation, NVM-based PIM, processing using memory [15]) evolve and mature, systems will take advantage of them even more, likely using principles similar to those used in the design of Tesseract.

REFERENCES

- [1] J. Ahn *et al.*, “A Scalable Processing-in-Memory Accelerator for Parallel Graph Processing,” in *ISCA*, 2015.
- [2] Hybrid Memory Cube Consortium, “HMC Specification 1.1,” 2013.
- [3] J. Jeddah and B. Keeth, “Hybrid Memory Cube: New DRAM Architecture Increases Density and Performance,” in *VLSIT*, 2012.
- [4] JEDEC, “High Bandwidth Memory (HBM) DRAM,” Standard No. JESD235, 2013.

- [5] N. Jouppi *et al.*, “TPU v4: An Optically Reconfigurable Supercomputer for Machine Learning with Hardware Support for Embedding,” in *ISCA*, 2023.
- [6] J. Fowers *et al.*, “A Configurable Cloud-Scale DNN Processor for Real-Time AI,” in *ISCA*, 2018.
- [7] S. Lie, “Cerebras Architecture Deep Dive: First Look Inside the Hardware/Software Co-Design for Deep Learning,” in *IEEE Micro*, 2023.
- [8] E. Talpes *et al.*, “The Microarchitectural Design of DIO, Tesla's Exa-Scale Computer,” in *IEEE Micro*, 2023.
- [9] A. Ishii and R. Wells, “NVLink-Network Switch - NVIDIA's Switch Chip for High Communication-Bandwidth SuperPODs,” in *Hot Chips*, 2022.
- [10] J. Ahn *et al.*, “PIM-Enabled Instructions: A Low-Overhead, Locality-Aware Processing-in-Memory Architecture,” in *ISCA*, 2015.
- [11] R. Pope *et al.*, “Efficiently Scaling Transformer Inference,” in *MLSys*, 2023.
- [12] D. Lepikhin *et al.*, “GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding,” in *ICLR*, 2021.
- [13] S. Wang *et al.*, “Overlap Communication with Dependent Computation via Decomposition in Large Deep Learning Models,” in *ASPLOS*, 2023.
- [14] Ahn *et al.*, “AIM: Energy-Efficient Aggregation Inside the Memory Hierarchy,” *ACM TACD*, vol. 13, no. 4, 2016.
- [15] O. Mutlu *et al.*, “A Modern Primer on Processing in Memory,” *Emerging Computing: From Devices to Systems*, 2021, <https://arxiv.org/abs/2012.03192>.
- [16] M. Zhang *et al.*, “GraphR: Reducing Communication for PIM-Based Graph Processing with Efficient Data Partition,” in *HPCA*, 2018.
- [17] L. Song, “GraphR: Accelerating Graph Processing Using ReRAM,” in *HPCA*, 2018.
- [18] Z. Zhuo *et al.*, “GraphQ: Scalable PIM-Based Graph Processing,” in *MICRO*, 2019.
- [19] G. Dai *et al.*, “GraphA: A Processing-in-Memory Architecture for Large-Scale Graph Processing,” in *IEEE TCAD*, 2018.
- [20] G. Li *et al.*, “GraphIA: An In-situ Accelerator for Large-Scale Graph Processing,” in *MEMSYS*, 2018.
- [21] S. Rheindt *et al.*, “NEMESIS: Near-Memory Graph Copy Enhanced System-Software,” in *MEMSYS*, 2019.
- [22] L. Belayneh and V. Bertacco, “GraphVine: Exploiting Multicast for Graph Analytics,” in *DATE*, 2020.
- [23] N. Challapalle *et al.*, “Gauss: Graph Analytics Accelerator Supporting Sparse Rank-Level Parallelism using Crossbar,” in *ISCA*, 2020.
- [24] M. Zhou *et al.*, “Ultra Efficient Accelerator for De Novo Genome Assembly by Near-Memory Computing,” in *ISCA*, 2021.
- [25] X. Xie *et al.*, “SpaceA: Sparse Matrix Vector Multiplication on Processing-in-Memory Accelerator,” in *HPCA*, 2021.
- [26] M. Zhou *et al.*, “HyGraph: Accelerating Graph Processing with Hybrid Memory-Centric Computing,” in *WATFOPS*, 2021.
- [27] M. Lenjani *et al.*, “Gearbox: A Case for Supporting Accumulation Dispatching and Hybrid Partitioning in PIM-based Accelerators,” in *ISCA*, 2022.
- [28] M. Orenes-Vera *et al.*, “Dalorex: A Data-Local Program Execution and Architecture for Memory-Bound Applications,” in *HPCA*, 2023.
- [29] J. Choquette, “Nvidia Hopper GPU: Scaling Performance,” in *Hot Chips*, 2022.
- [30] F. Devaux, “The True Processing in Memory Accelerator,” in *Hot Chips*, 2019.
- [31] J. Gómez-Luna *et al.*, “Benchmarking a New Paradigm: Experimental Analysis and Characterization of a Real Processing-in-Memory System,” *IEEE Access*, 2022.
- [32] J. Gomez-Luna *et al.*, “Evaluating Machine Learning Workloads on Memory-Centric Computing Systems,” in *ISPASS*, 2023.
- [33] S. Lee *et al.*, “Hardware Architecture and Software Stack for PIM Based on Commercial DRAM Technology: Industrial Product,” in *ISCA*, 2021.
- [34] Y.-C. Kwon *et al.*, “25.4 A 20nm 6GB Function-In-Memory DRAM, Based on HBM2 with a 1.2 TTOPS Programmable Computing Unit Using Bank-Level Parallelism, for Machine Learning Applications,” in *ISSCC*, 2021.
- [35] L. Ke *et al.*, “Near-Memory Processing in Action: Accelerating Personalized Recommendation with AxDIMM,” in *IEEE Micro*, 2021.
- [36] D. Lee *et al.*, “Improving In-Memory Database Operations with Acceleration DIMM (AxDIMM),” in *DaMoN*, 2022.
- [37] S. Lee *et al.*, “A 1nm 1.25V 8Gb/s/pin GDDR6-based Accelerator-in-Memory supporting 1T TOPS MAC Operation and Various Activation Functions for Deep-Learning Applications,” in *ISSCC*, 2022.
- [38] D. Niu *et al.*, “184QPS/W 64Mb/mm² 3D Logic-to-DRAM Hybrid Bonding with Process-Near-Memory Engine for Recommendation System,” in *ISSCC*, 2022.
- [39] Y. Kwon, “System Architecture and Software Stack for GDDR6-AIM,” in *HCS*, 2022.
- [40] G. Singh *et al.*, “FPGA-based Near-Memory Acceleration of Modern Data-Intensive Applications,” *IEEE Micro*, 2021.
- [41] G. Singh *et al.*, “Accelerating Weather Prediction using Near-Memory Reconfigurable Fabric,” *ACM TMTS*, 2021.
- [42] S. Hong *et al.*, “PGX-D: A Fast Distributed Graph Processing Engine,” in *SC*, 2015.
- [43] T. N. Kipf and M. Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” in *ICLR*, 2017.
- [44] L. Nai *et al.*, “GraphPIM: Enabling Instruction-Level PIM Offloading in Graph Computing Frameworks,” in *HPCA*, 2017.
- [45] A. Besta *et al.*, “NISA: Set-Centric Instruction Set Architecture for Graph Mining on Processing-in-Memory Systems,” in *MICRO*, 2021.
- [46] T. J. Ham *et al.*, “Graphicionado: A High-Performance and Energy-Efficient Accelerator for Graph Analytics,” in *MICRO*, 2016.

Accelerating Graph Pattern Mining

- 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
Thailand

²IIT Tirupati, India

³King Mongkut's University of Technology North Bangkok,
⁴Technical University of Ostrava, Czech Republic

⁵AGH-UST, Poland

Accelerating Machine Learning 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.

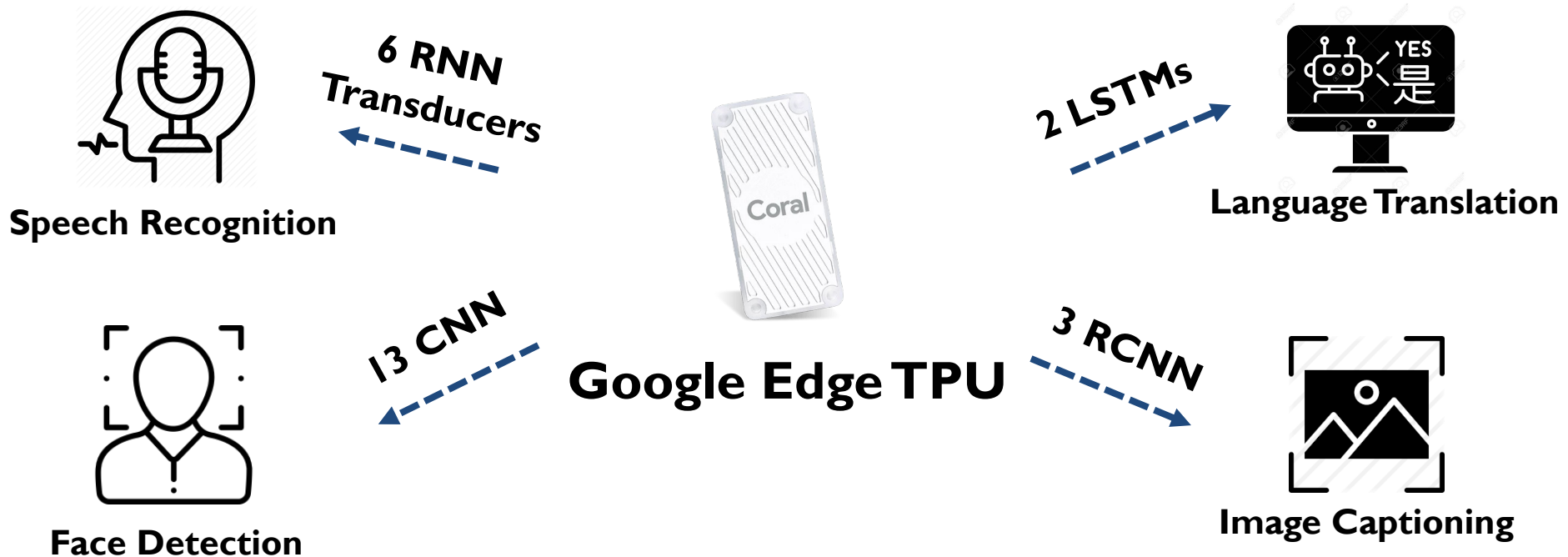
[‡]Univ. of Illinois Urbana-Champaign

[§]Google

^{*}ETH Zürich

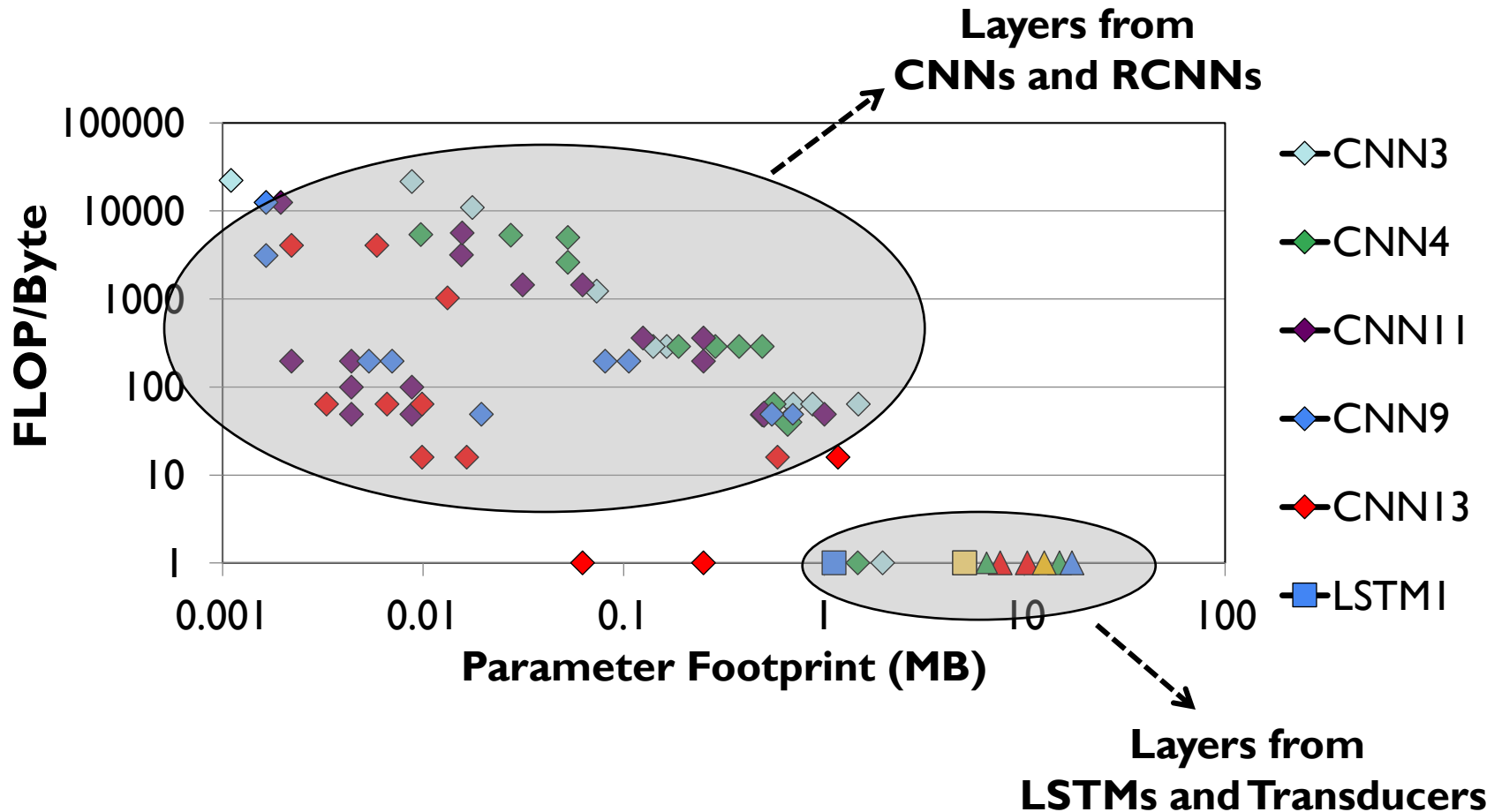
Google Edge Neural Network Models

We analyze inference execution using 24 edge NN models



Diversity Across the Models

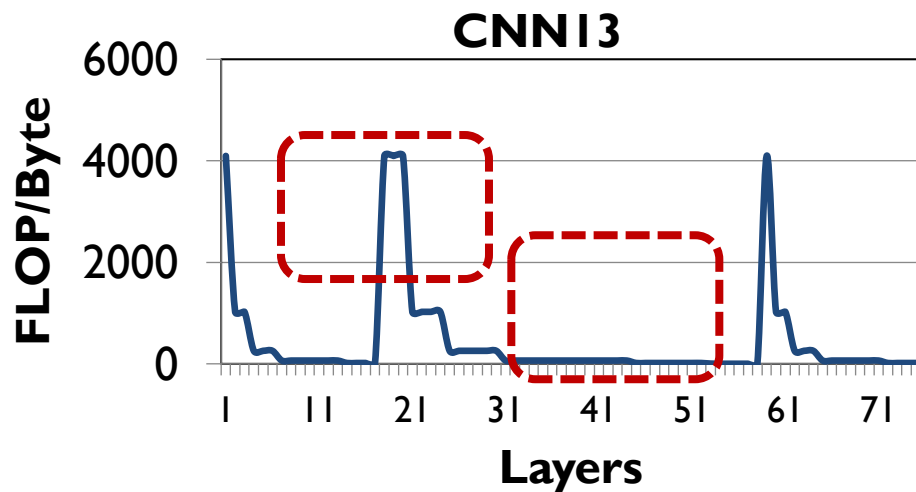
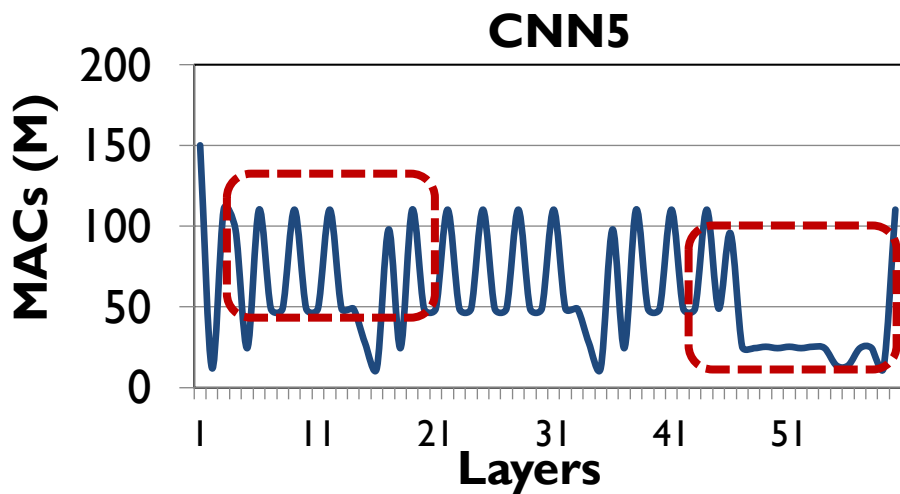
Insight I: there is **significant variation** in terms of layer characteristics **across the models**



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:



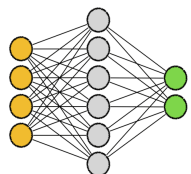
Variation in **MAC intensity**: up to **200x** across layers

Variation in **FLOP/Byte**: up to **244x** across layers

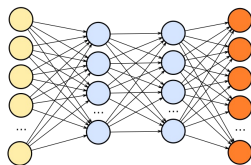
Mensa High-Level Overview

Edge TPU Accelerator

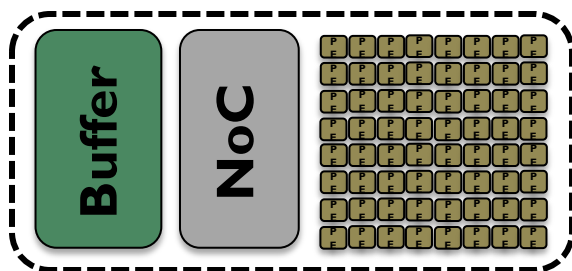
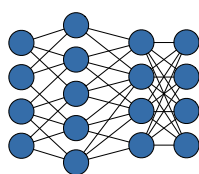
Model A



Model B



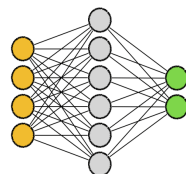
Model C



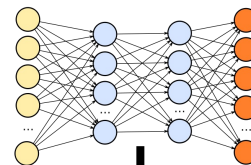
Monolithic Accelerator

Mensa

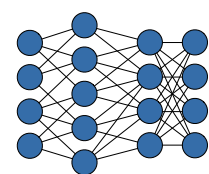
Model A



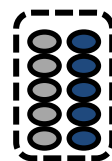
Model B



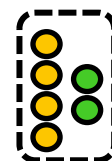
Model C



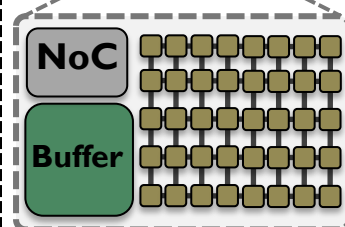
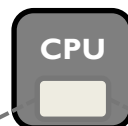
Family 1



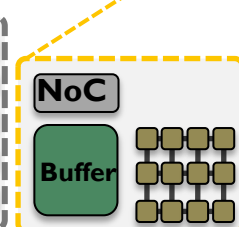
Family 2



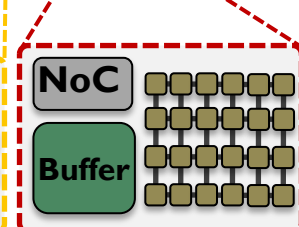
Family 3



Acc. 1



Acc. 2

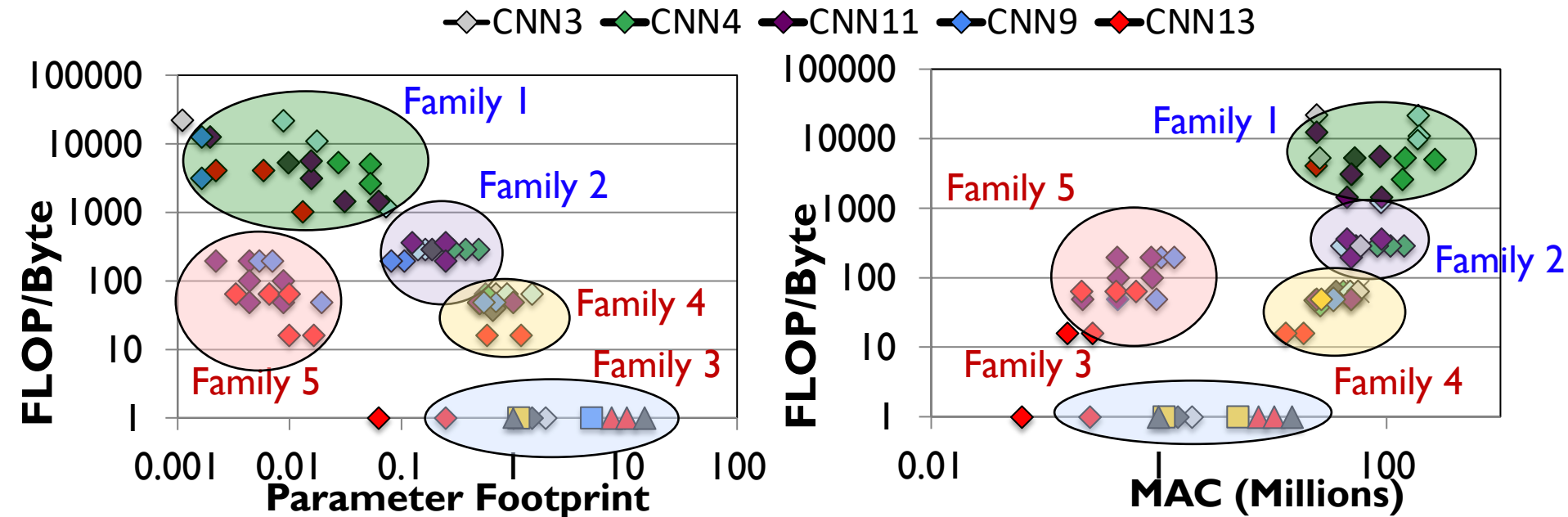


Acc. 3

Heterogeneous Accelerators

Identifying Layer Families

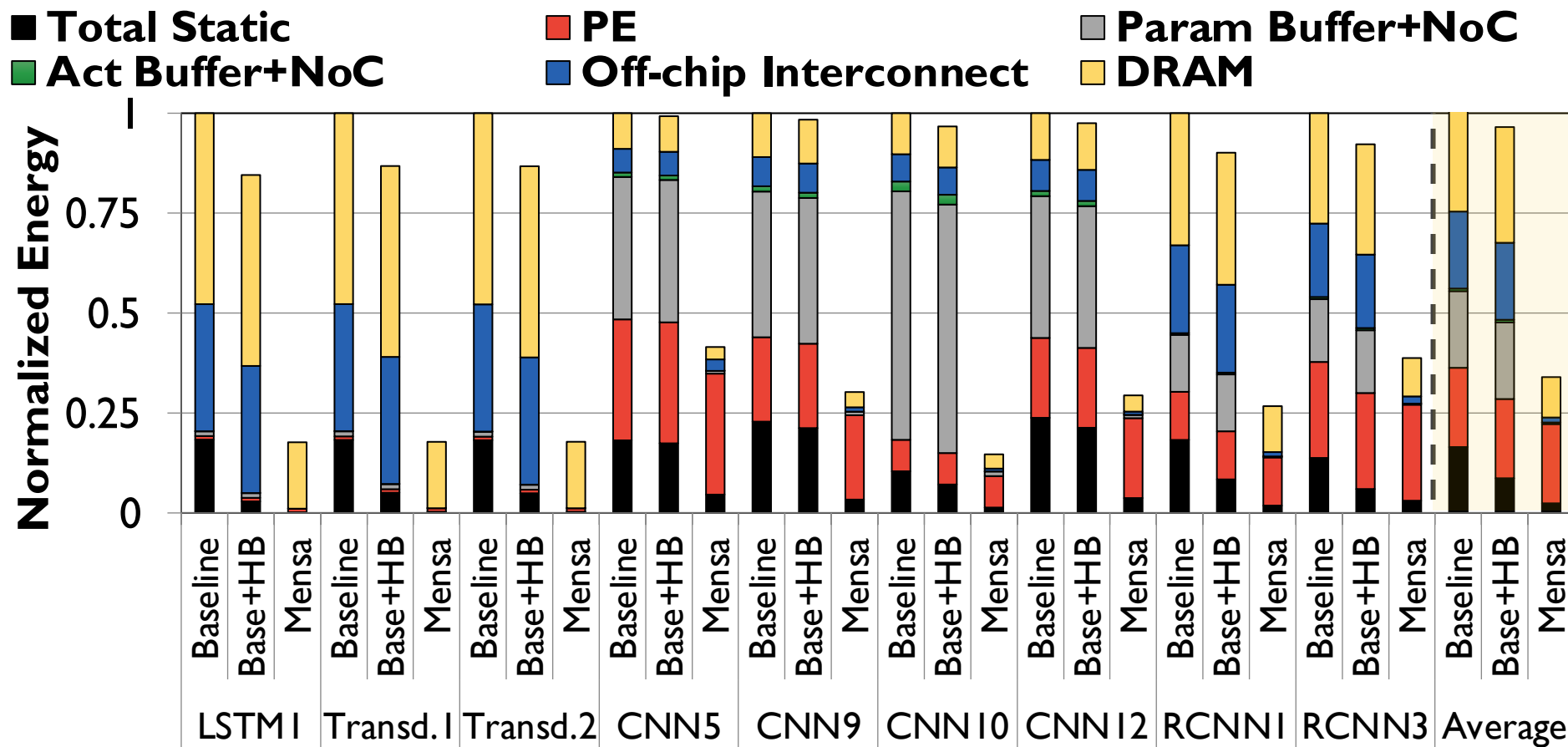
Key observation: the majority of layers group into a small number of layer families



Families 1 & 2: low parameter footprint, high data reuse and **MAC intensity**
→ compute-centric layers

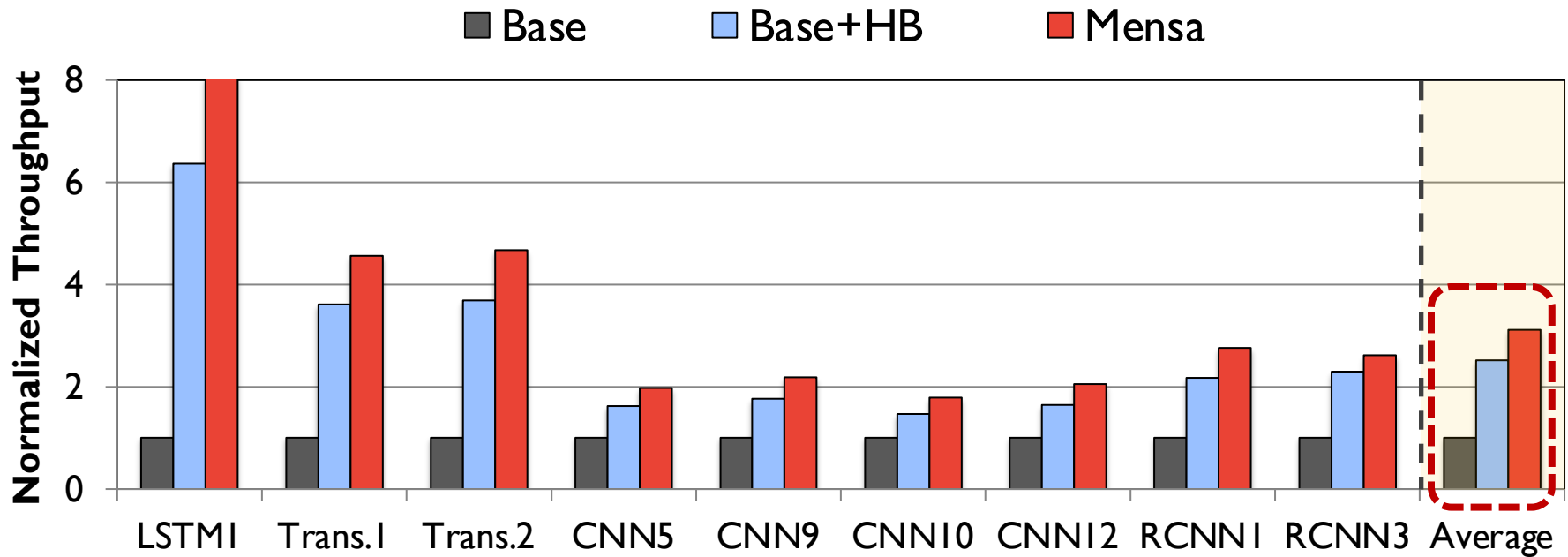
Families 3, 4 & 5: high parameter footprint, low data reuse and **MAC intensity**
→ data-centric layers

Mensa: Energy Reduction



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

Mensa: Throughput Improvement



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*

Accelerating Mobile Workloads

- 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"

Proceedings of the 23rd International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), Williamsburg, VA, USA, March 2018.

[[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 Boroumand¹

Saugata Ghose¹

Youngsok Kim²

Rachata Ausavarungnirun¹

Eric Shiu³

Rahul Thakur³

Daehyun Kim^{4,3}

Aki Kuusela³

Allan Knies³

Parthasarathy Ranganathan³

Onur Mutlu^{5,1}

Accelerating DNA Read Mapping

- Jeremie S. Kim, Damla Senol Cali, Hongyi Xin, Donghyuk Lee, Saugata Ghose, Mohammed Alser, Hasan Hassan, Oguz Ergin, Can Alkan, and Onur Mutlu,
["GRIM-Filter: Fast Seed Location Filtering in DNA Read Mapping Using Processing-in-Memory Technologies"](#)
[*BMC Genomics*](#), 2018.
Proceedings of the 16th Asia Pacific Bioinformatics Conference (APBC), Yokohama, Japan, January 2018.
[[Slides \(pptx\) \(pdf\)](#)]
[[Source Code](#)]
[[arxiv.org Version \(pdf\)](#)]
[[Talk Video at AACBB 2019](#)]

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

Jeremie S. Kim^{1,6*}, Damla Senol Cali¹, Hongyi Xin², Donghyuk Lee³, Saugata Ghose¹, Mohammed Alser⁴, Hasan Hassan⁶, Oguz Ergin⁵, Can Alkan^{4*} and Onur Mutlu^{6,1*}

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"
Proceedings of the 27th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), Virtual, February-March 2022.
[[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

In-Storage Metagenomics [ISCA 2024]

- Nika Mansouri Ghiasi, Mohammad Sadrosadati, Harun Mustafa, Arvid Gollwitzer, Can Firtina, Julien Eudine, Haiyu Mao, Joel Lindegger, Meryem Banu Cavlak, Mohammed Alser, Jisung Park, and Onur Mutlu,

"MegIS: High-Performance and Low-Cost Metagenomic Analysis with In-Storage Processing"

Proceedings of the 51st Annual International Symposium on Computer Architecture (ISCA), Buenos Aires, Argentina, July 2024.

[[Slides \(pptx\)](#)] [[pdf](#)]

[[arXiv version](#)]

MegIS: High-Performance, Energy-Efficient, and Low-Cost Metagenomic Analysis with In-Storage Processing

Nika Mansouri Ghiasi¹ Mohammad Sadrosadati¹ Harun Mustafa¹ Arvid Gollwitzer¹
Can Firtina¹ Julien Eudine¹ Haiyu Mao¹ Joël Lindegger¹ Meryem Banu Cavlak¹
Mohammed Alser¹ Jisung Park² Onur Mutlu¹
¹ETH Zürich ²POSTECH

Many More Examples ...

A Modern Primer on Processing-In-Memory

Onur Mutlu^a, Saugata Ghose^b, Juan Gómez-Luna^c, Rachata Ausavarungnirun^d,
Mohammad Sadrosadati^a, Geraldo F. Oliveira^a

SAFARI Research Group

^a*ETH Zürich*

^b*University of Illinois Urbana-Champaign*

^c*NVIDIA Research*

^d*MangoBoost Inc.*

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, Rachata Ausavarungnirun,
Mohammad Sadrosadati, and Geraldo F. Oliveira,

"A Modern Primer on Processing in Memory"

*Invited Book Chapter in **Emerging Computing: From Devices to Systems - Looking Beyond Moore and Von Neumann**, Springer, 2022.*

PAPI: Hybrid System for Near-Memory LLM Inference

- Yintao He, Haiyu Mao, Christina Giannoula, Mohammad Sadrosadati, Juan Gomez-Luna, Huawei Li, Xiaowei Li, Ying Wang, and Onur Mutlu,
"PAPI: Exploiting Dynamic Parallelism in Large Language Model Decoding with a Processing-In-Memory-Enabled Computing System,"
Proceedings of the 30th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), Rotterdam, Netherlands, April 2025.

PAPI: Exploiting Dynamic Parallelism in Large Language Model Decoding with a Processing-In-Memory-Enabled Computing System

Yintao He^{1,2} Haiyu Mao^{3,4} Christina Giannoula^{5,6,4} Mohammad Sadrosadati⁴
Juan Gómez-Luna⁷ Huawei Li^{1,2} Xiaowei Li^{1,2} Ying Wang¹ Onur Mutlu⁴

¹SKLP, Institute of Computing Technology, CAS ²University of Chinese Academy of Sciences ³King's College London

⁴ETH Zürich ⁵University of Toronto ⁶Vector Institute ⁷NVIDIA

CENT: GPU-Free System for Near-Memory LLM Inference

- Yufeng Gu, Alireza Khadem, Sumanth Umesh, Ning Liang, Xavier Servot, Onur Mutlu, Ravi Iyer, and Reetuparna Das,
"PIM Is All You Need: A CXL-Enabled GPU-Free System for Large Language Model Inference,"
Proceedings of the 30th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), Rotterdam, Netherlands, April 2025.
Officially artifact evaluated as available, functional, and reproducible.

PIM Is All You Need: A CXL-Enabled GPU-Free System for Large Language Model Inference

Yufeng Gu*
University of Michigan
Ann Arbor, USA
yufenggu@umich.edu

Alireza Khadem*
University of Michigan
Ann Arbor, USA
arkhadem@umich.edu

Sumanth Umesh
University of Michigan
Ann Arbor, USA
sumanthu@umich.edu

Ning Liang
University of Michigan
Ann Arbor, USA
nliang@umich.edu

Xavier Servot
ETH Zürich
Zürich, Switzerland
xservot@student.ethz.ch

Onur Mutlu
ETH Zürich
Zürich, Switzerland
omutlu@gmail.com

Ravi Iyer[†]
Google
Mountain View, USA
raviiyer20@gmail.com

Reetuparna Das
University of Michigan
Ann Arbor, USA
reetudas@umich.edu

PAPI LLM Inference System [ASPLOS 2025]

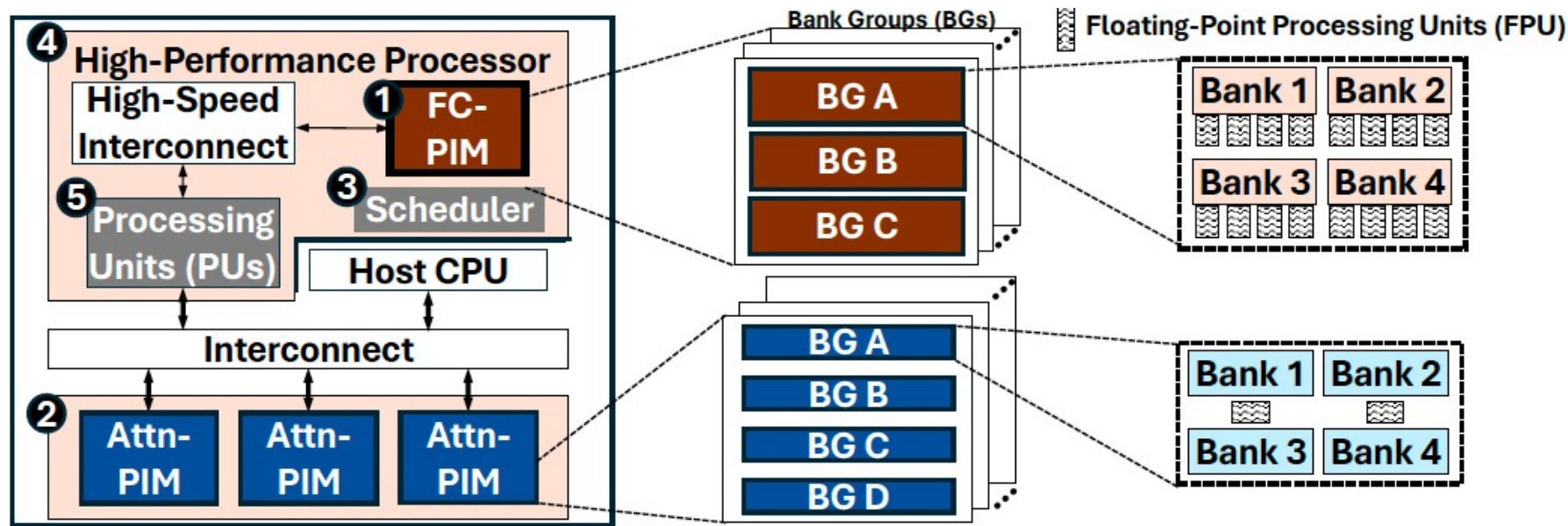


Fig. 5: Overview of the PAPI LLM Inference System. Adapted from [18].

PAPI over best prior LLM decoding system

- **1.8×** speedup
- **3.4×** energy efficiency increase

CENT LLM Inference System [ASPLOS 2025]

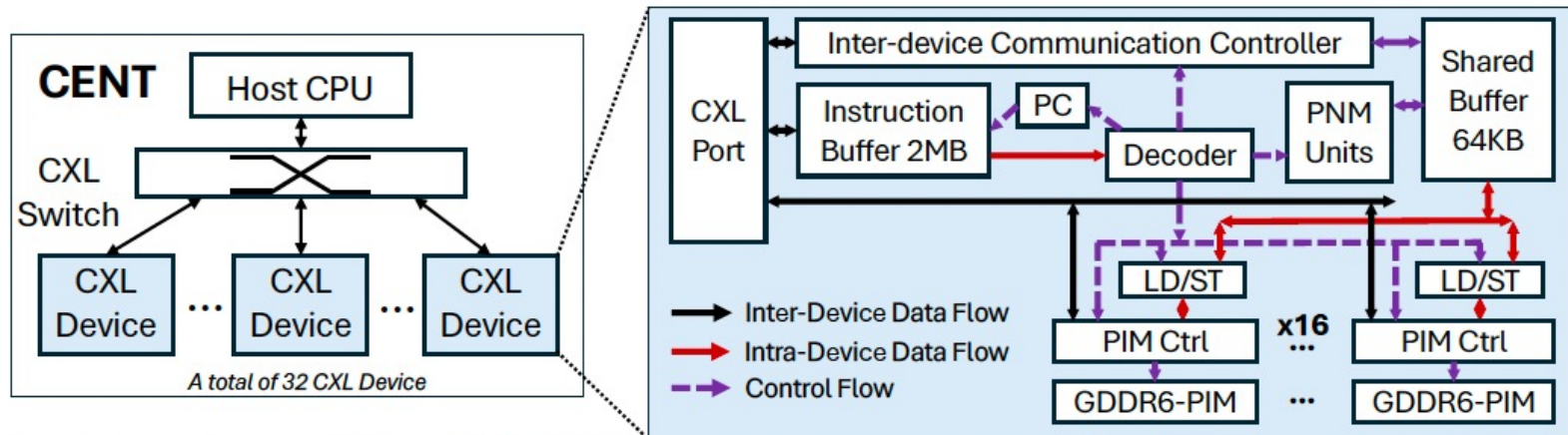


Fig. 6: **Overview of the CENT LLM Inference System.** Host CPU drives 32 CXL devices, each having a CXL controller, PNM units, and 16 GDDR6-PIM chips. The LLM inference task is partitioned between PNM units and GDDR6-PIM chips. CENT provides communication mechanisms within and across CXL devices to coordinate and scale computation. Adapted from [19].

CENT over best prior GPU LLM inference system

- **2.3×** higher throughput
- **5.2×** higher tokens per dollar
- **2.4×** lower hardware cost

Processing in Memory: Two Types

1. Processing **near** Memory
2. Processing **using** Memory

Focus: Processing using DRAM

- We can natively support
 - Bulk bitwise COPY and INIT/ZERO
 - Bulk bitwise AND, OR, NOT, MAJ, NOR, NAND
 - True Random Number Generation; Physical Unclonable Functions
 - More complex computation using Lookup Tables
- At low cost
- Using analog computation capability of DRAM
 - Idea: activating (multiple) rows performs computation
 - Even in commodity off-the-shelf DRAM chips!
- 30X-257X performance and energy improvements

Seshadri+, "RowClone: Fast and Efficient In-DRAM Copy and Initialization of Bulk Data," MICRO 2013.

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

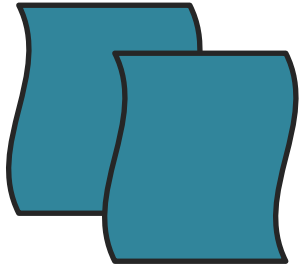
Seshadri+, "Ambit: In-Memory Accelerator for Bulk Bitwise Operations Using Commodity DRAM Technology," MICRO 2017.

Hajinazar+, "SIMDRAM: A Framework for Bit-Serial SIMD Processing using DRAM," ASPLOS 2021.

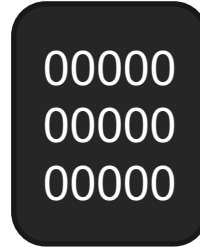
Oliveira+, "MIMDRAM: An End-to-End Processing-Using-DRAM System for High-Throughput, Energy-Efficient and Programmer-Transparent Multiple-Instruction Multiple-Data Processing," HPCA 2024.

Starting Simple: Data Copy and Initialization

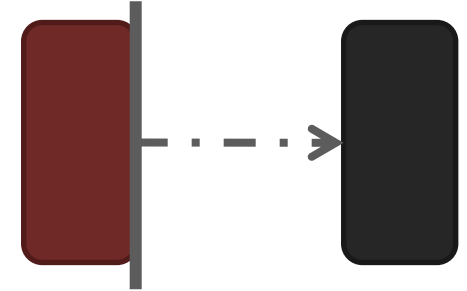
memmove & memcpy: 5% cycles in Google's datacenter [Kanev+ ISCA'15]



Forking



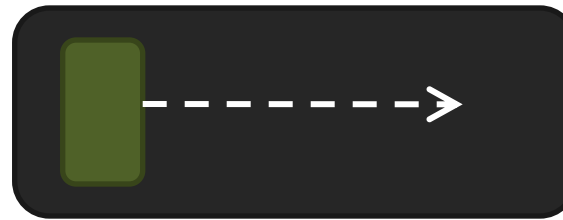
**Zero initialization
(e.g., security)**



Checkpointing



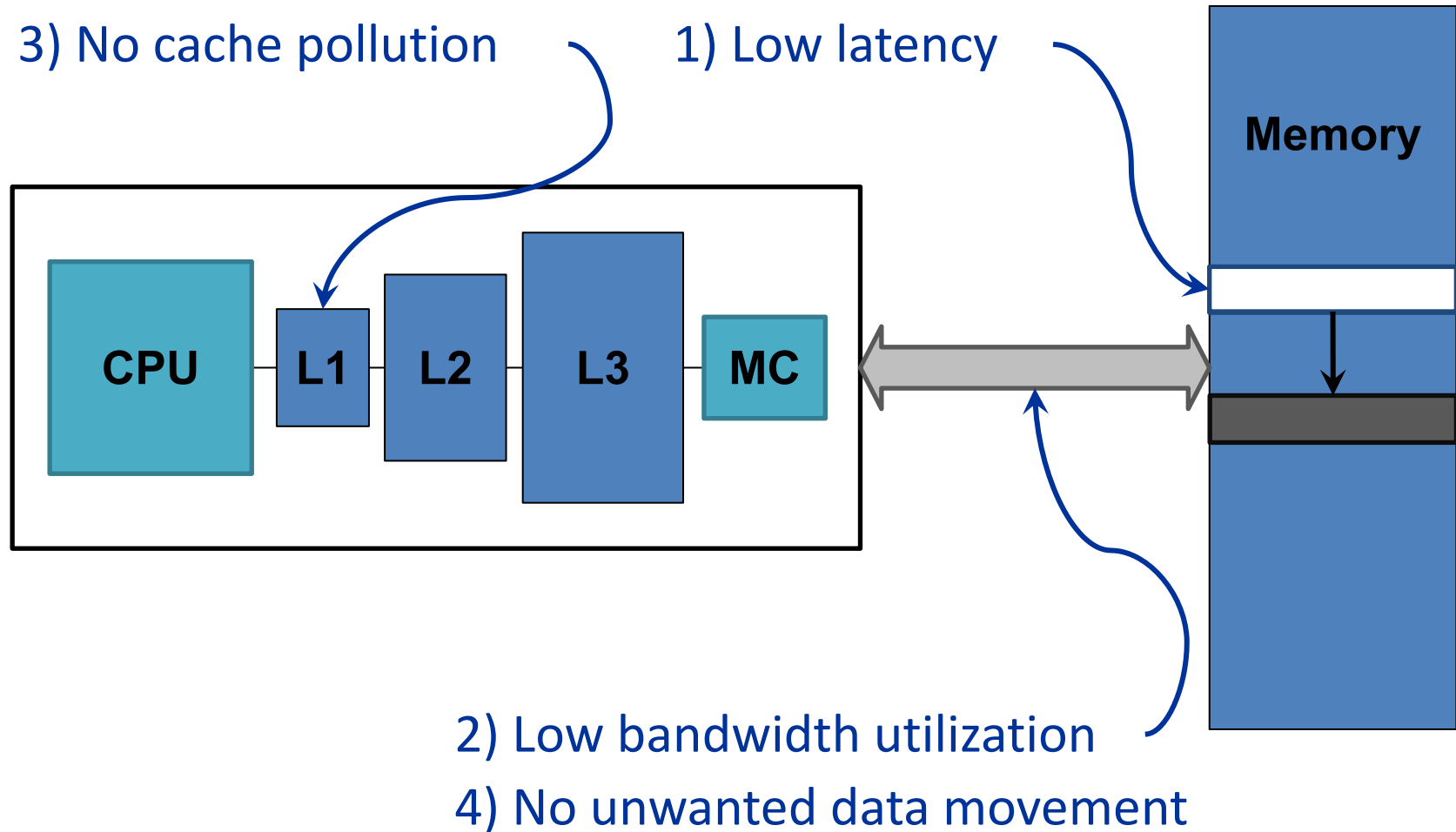
**VM Cloning
Deduplication**



Page Migration

...
Many more

Future Systems: In-Memory Copy

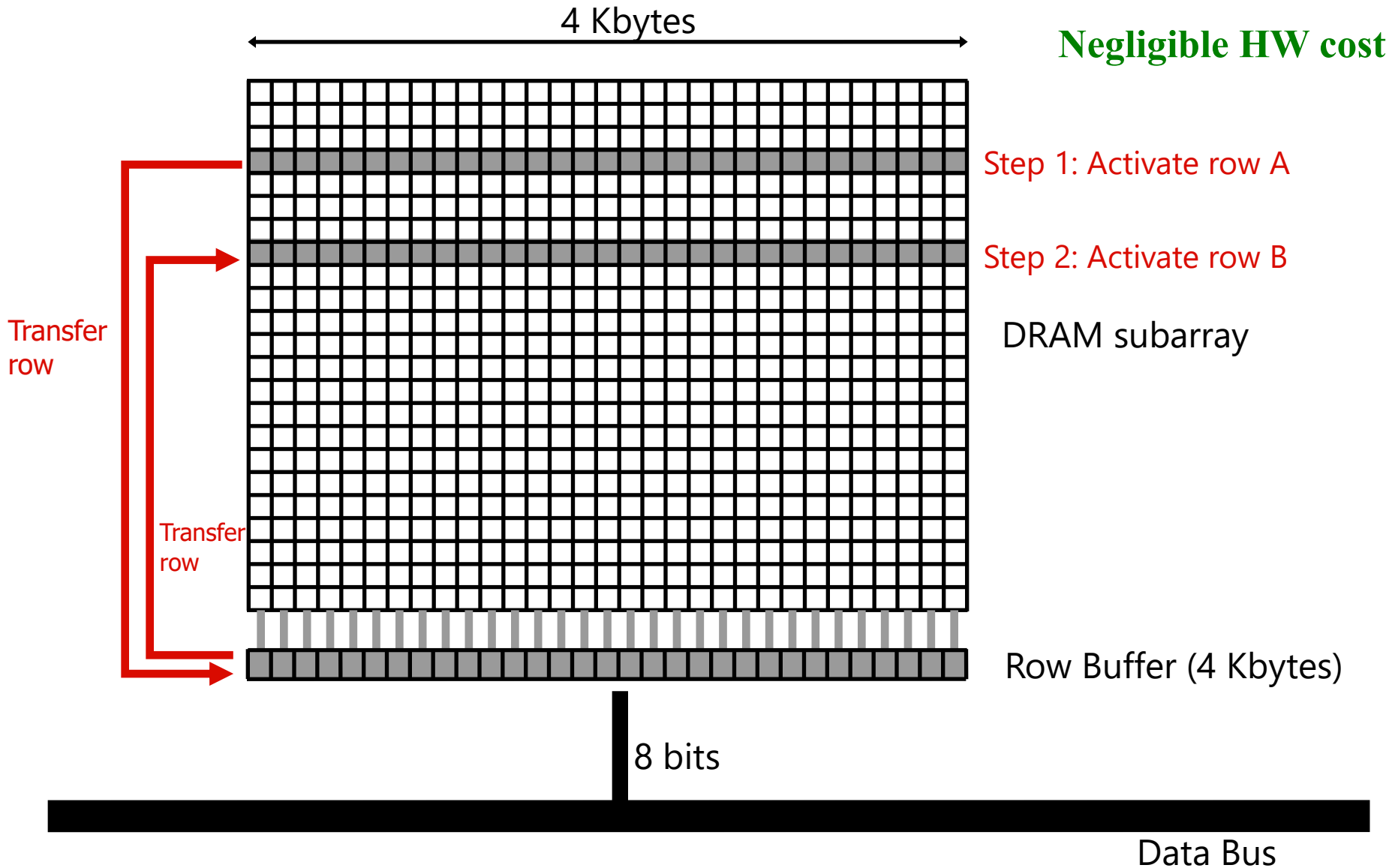


1046ns, 3.6uJ → 90ns, 0.04uJ

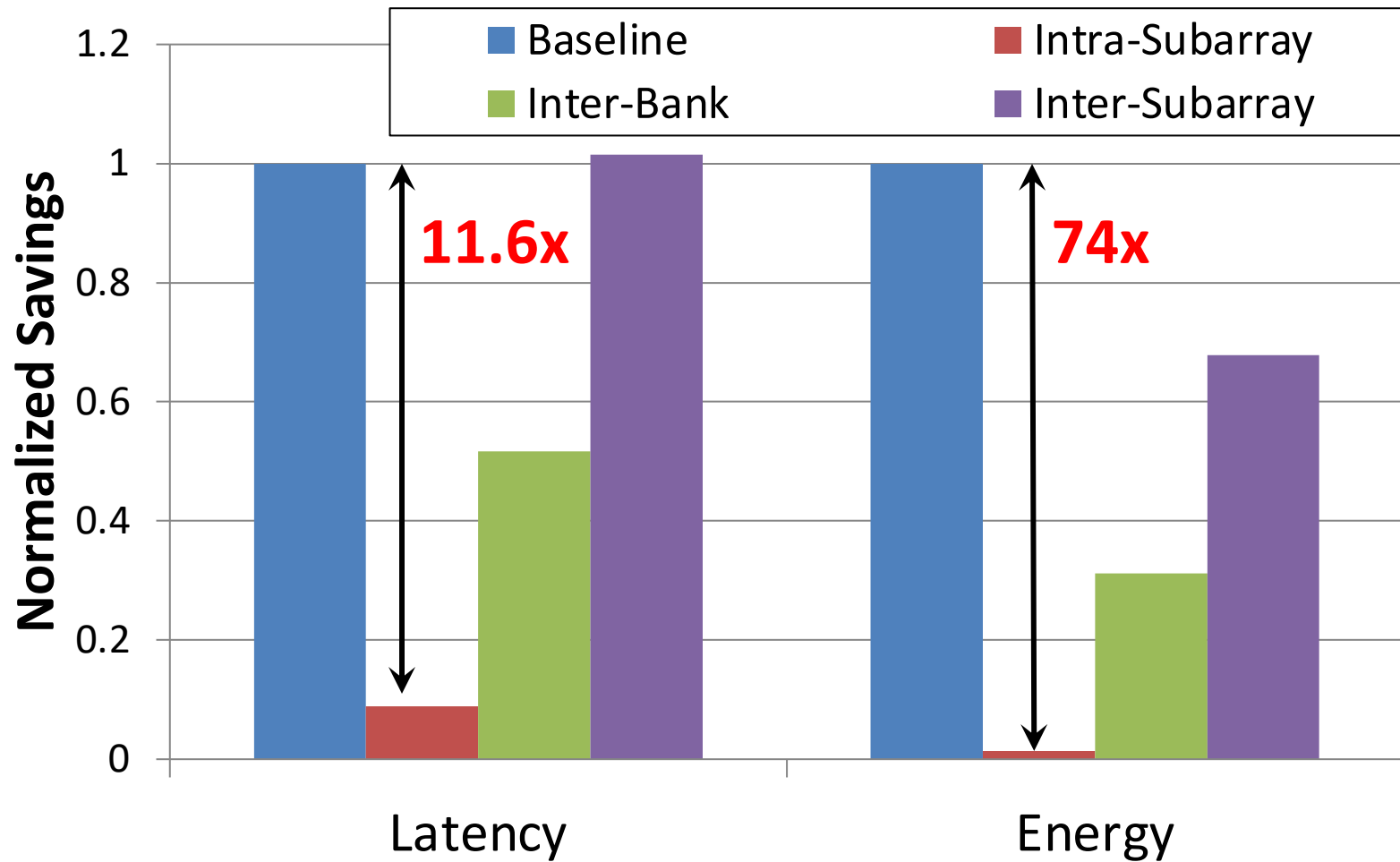
RowClone: In-DRAM Row Copy

Idea: Two consecutive ACTivates

Negligible HW cost



RowClone: Latency and Energy Savings



Seshadri et al., "RowClone: Fast and Efficient In-DRAM Copy and Initialization of Bulk Data," MICRO 2013.

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

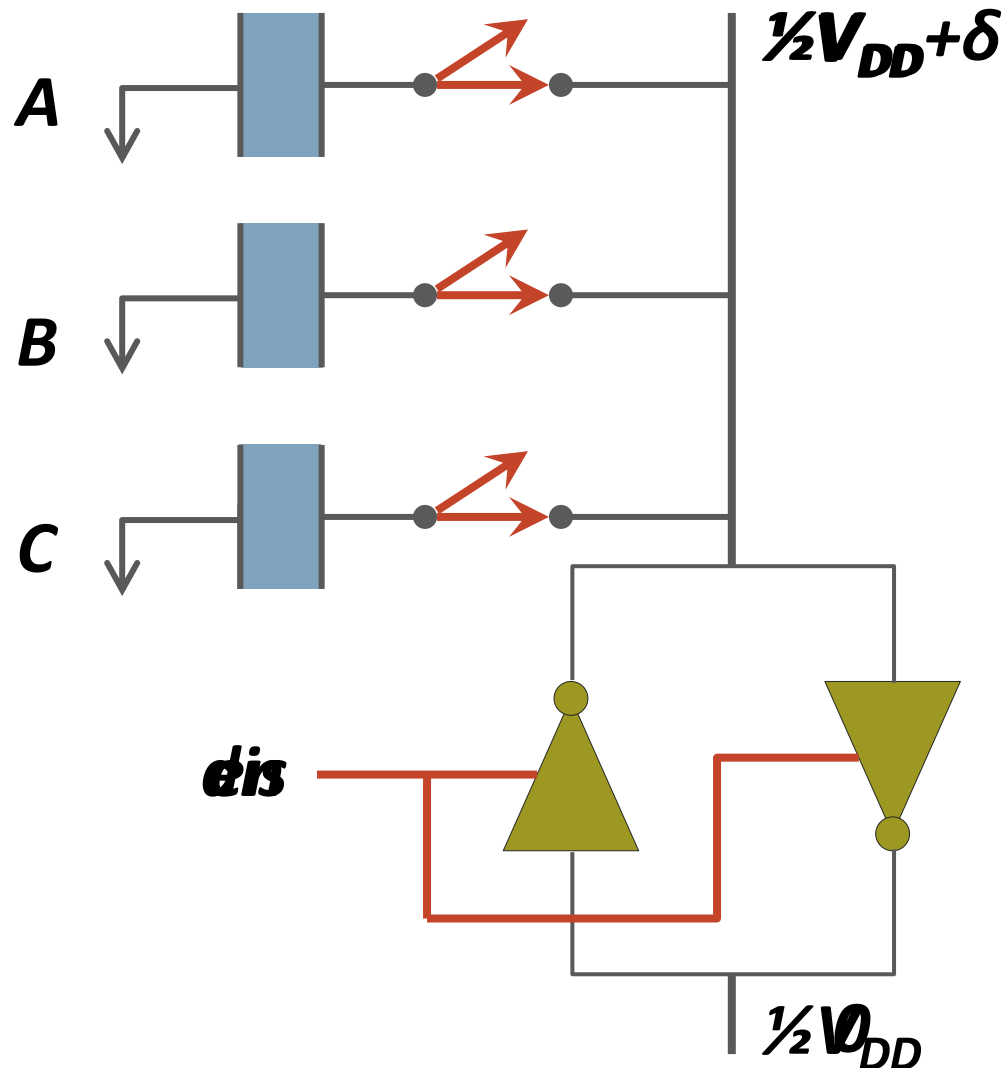
Vivek Seshadri Yoongu Kim Chris Fallin* Donghyuk Lee
vseshadr@cs.cmu.edu yoongukim@cmu.edu cfallin@c1f.net donghyuk1@cmu.edu

Rachata Ausavarungnirun Gennady Pekhimenko Yixin Luo
rachata@cmu.edu gpekhime@cs.cmu.edu yixinluo@andrew.cmu.edu

Onur Mutlu Phillip B. Gibbons† Michael A. Kozuch† Todd C. Mowry
onur@cmu.edu phillip.b.gibbons@intel.com michael.a.kozuch@intel.com tcm@cs.cmu.edu

Carnegie Mellon University †Intel Pittsburgh

In-DRAM AND/OR: Triple Row Activation



Final State
 $AB + BC + AC$

**$C(A + B) +$
 $\sim C(AB)$**

In-DRAM Acceleration of Database Queries

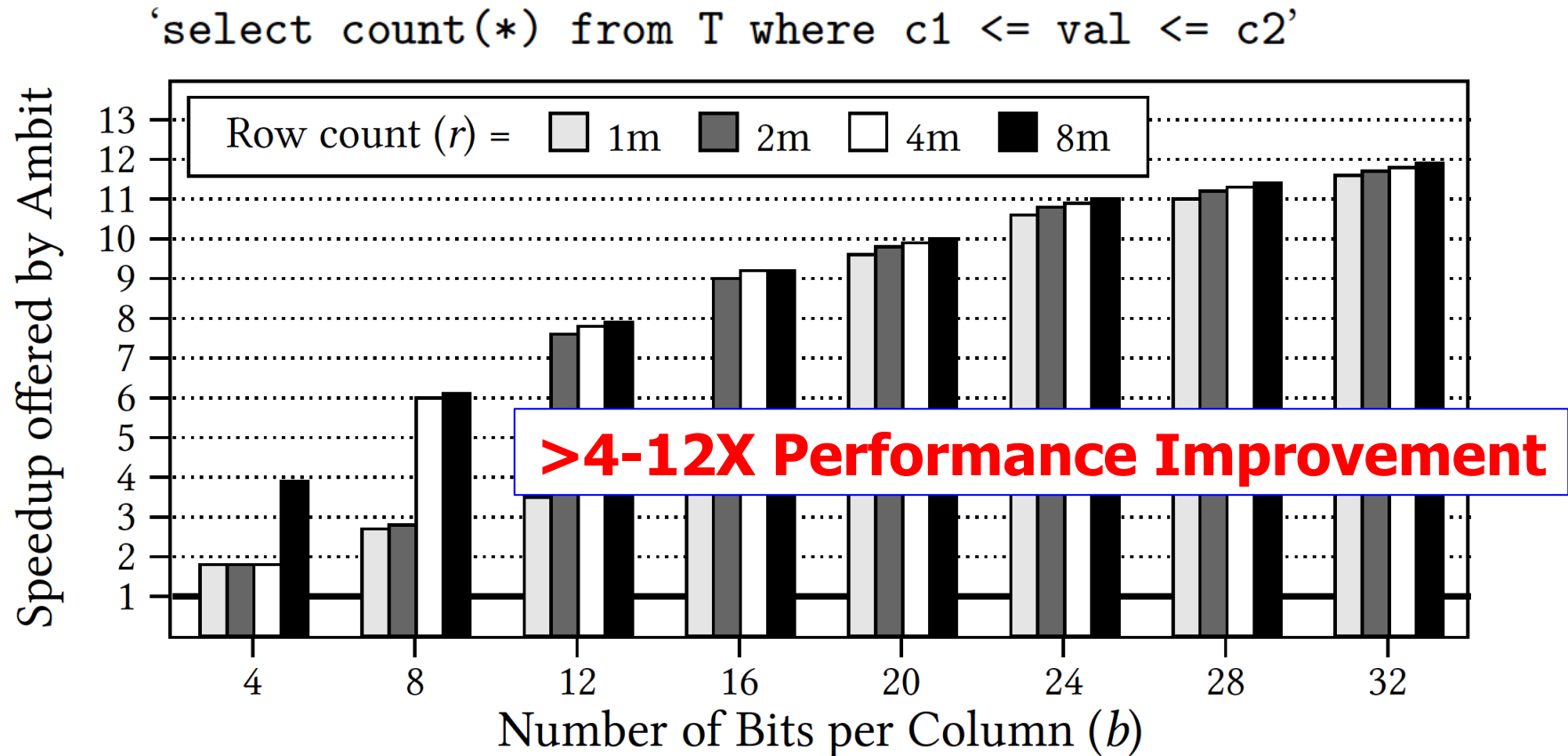
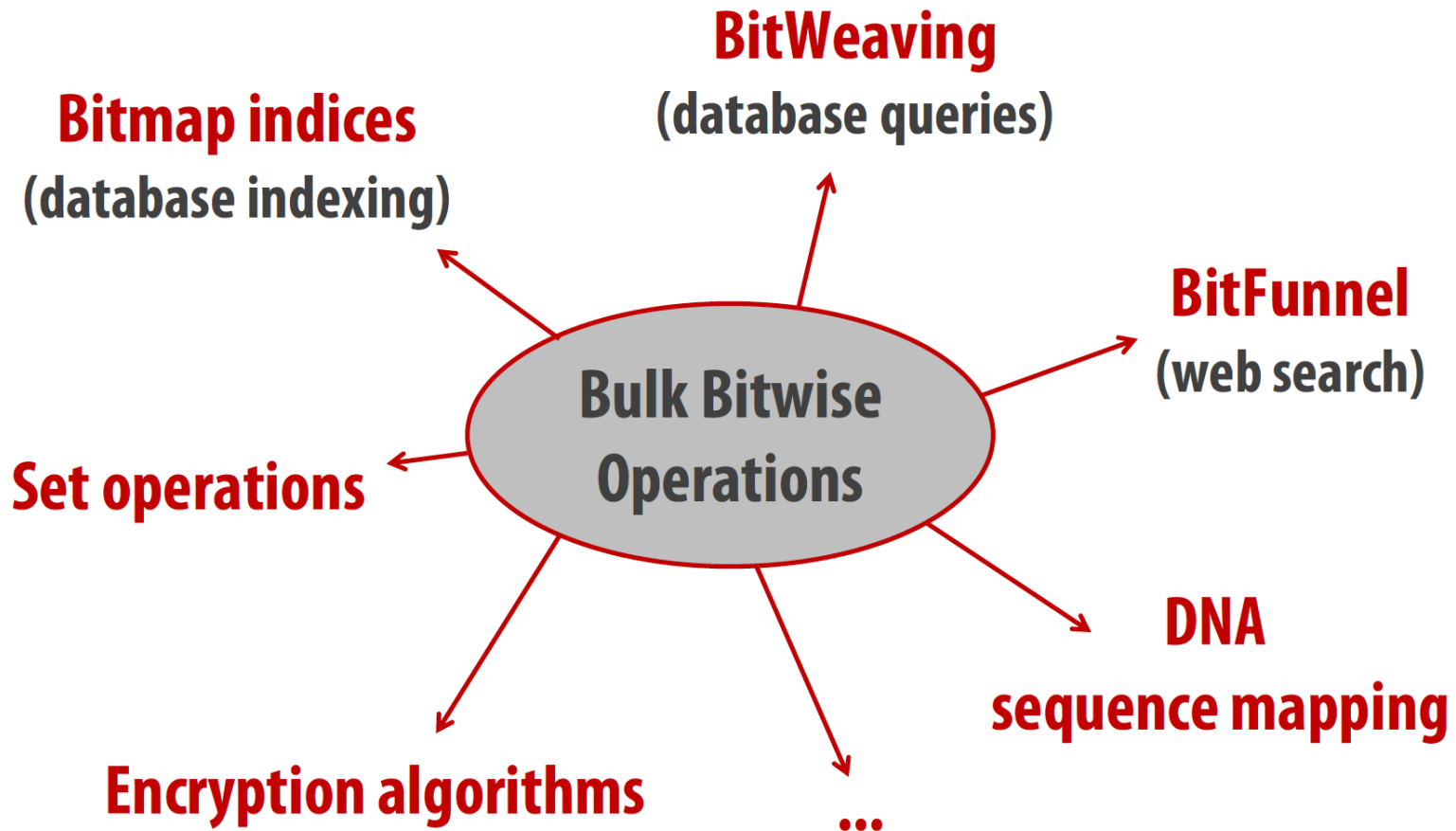


Figure 11: Speedup offered by Ambit over baseline CPU with SIMD for BitWeaving

Seshadri+, "Ambit: In-Memory Accelerator for Bulk Bitwise Operations using Commodity DRAM Technology," MICRO 2017.

Bulk Bitwise Operations in Workloads



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^{1,5} Donghyuk Lee^{2,5} Thomas Mullins^{3,5} Hasan Hassan⁴ Amirali Boroumand⁵
Jeremie Kim^{4,5} Michael A. Kozuch³ Onur Mutlu^{4,5} Phillip B. Gibbons⁵ Todd C. Mowry⁵

¹Microsoft Research India ²NVIDIA Research ³Intel ⁴ETH Zürich ⁵Carnegie Mellon University

SIMDRAM Framework

- 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"](#)** *Proceedings of the 26th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Virtual, March-April 2021.
[[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 Ghiasi¹

*Geraldo F. Oliveira¹

Minesh Patel¹

Juan Gómez-Luna¹

Sven Gregorio¹

Mohammed Alser¹

Onur Mutlu¹

João Dinis Ferreira¹

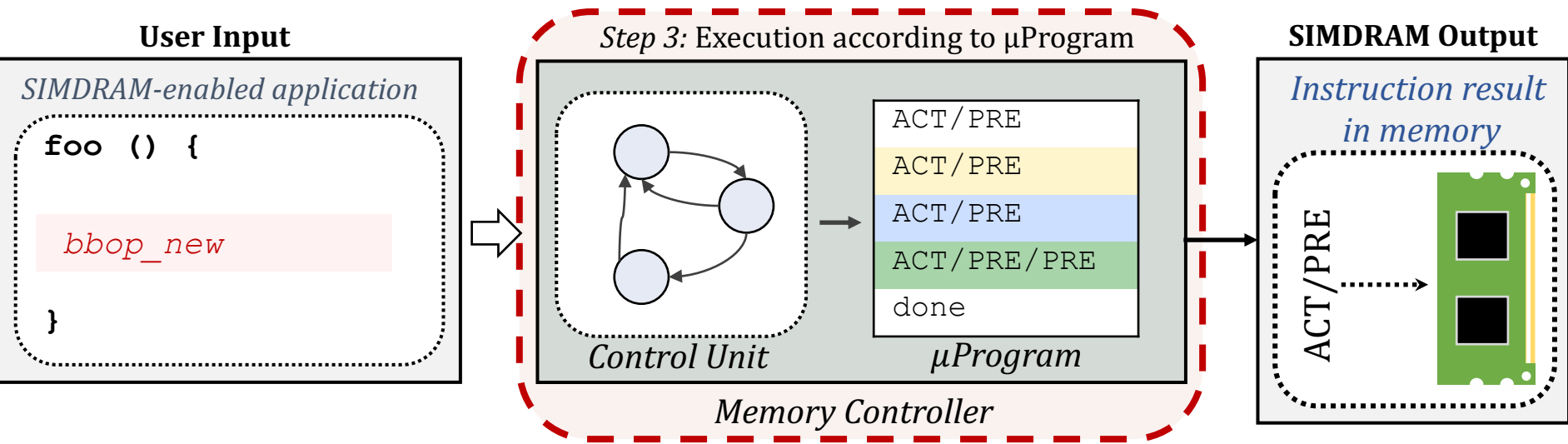
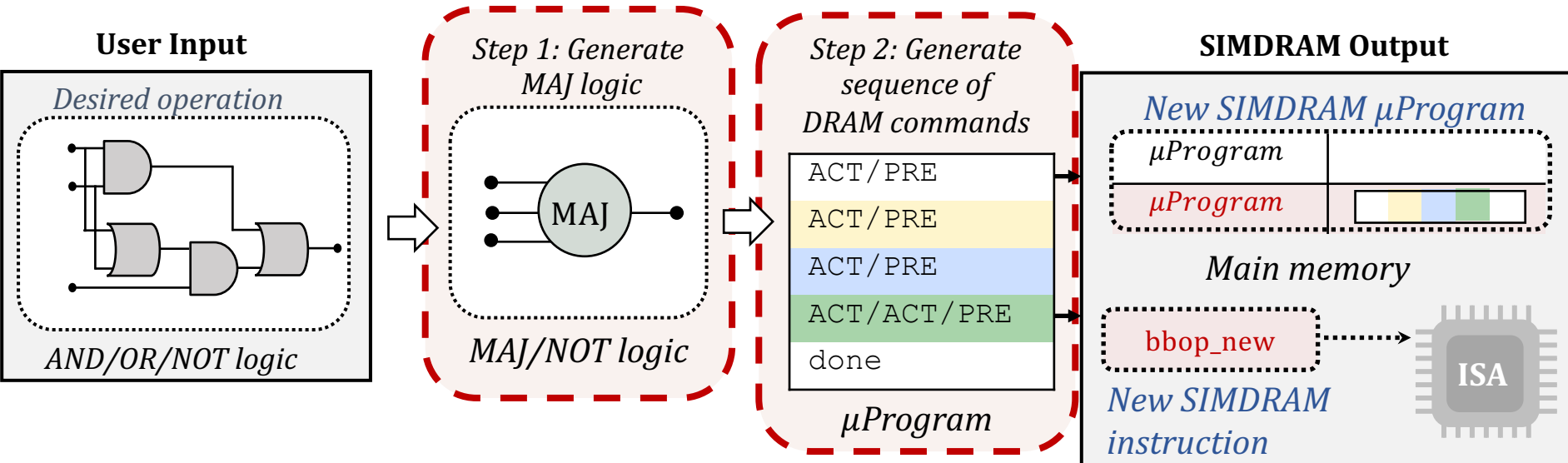
Saugata Ghose³

¹ETH Zürich

²Simon Fraser University

³University of Illinois at Urbana-Champaign

SIMDRAM Framework: Overview



SIMDRAM Key Results

Large improvements over **CPU** & **high-end GPU**:

Throughput: **88×** and **5.8×**

(16 complex operations)

Energy: **257×** and **31×**

(16 complex operations)

Application Performance: **21×** and **2.1×**

(seven common real-world applications)

More on SIMD RAM

- 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"**](#) *Proceedings of the 26th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Virtual, March-April 2021.
[[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}	*Geraldo F. Oliveira ¹	Sven Gregorio ¹	João Dinis Ferreira ¹
Nika Mansouri Ghiasi ¹	Minesh Patel ¹	Mohammed Alser ¹	Saugata Ghose ³
	Juan Gómez-Luna ¹	Onur Mutlu ¹	

¹ETH Zürich

²Simon Fraser University

³University of Illinois at Urbana–Champaign

MIMDRAM: More Flexible Processing using DRAM

- **Appears at HPCA 2024** <https://arxiv.org/pdf/2402.19080.pdf>

MIMDRAM: An End-to-End Processing-Using-DRAM System for High-Throughput, Energy-Efficient and Programmer-Transparent Multiple-Instruction Multiple-Data Computing

Geraldo F. Oliveira[†] Ataberk Olgun[†] Abdullah Giray Yağlıkçı[†] F. Nisa Bostancı[†]
Juan Gómez-Luna[†] Saugata Ghose[‡] Onur Mutlu[†]

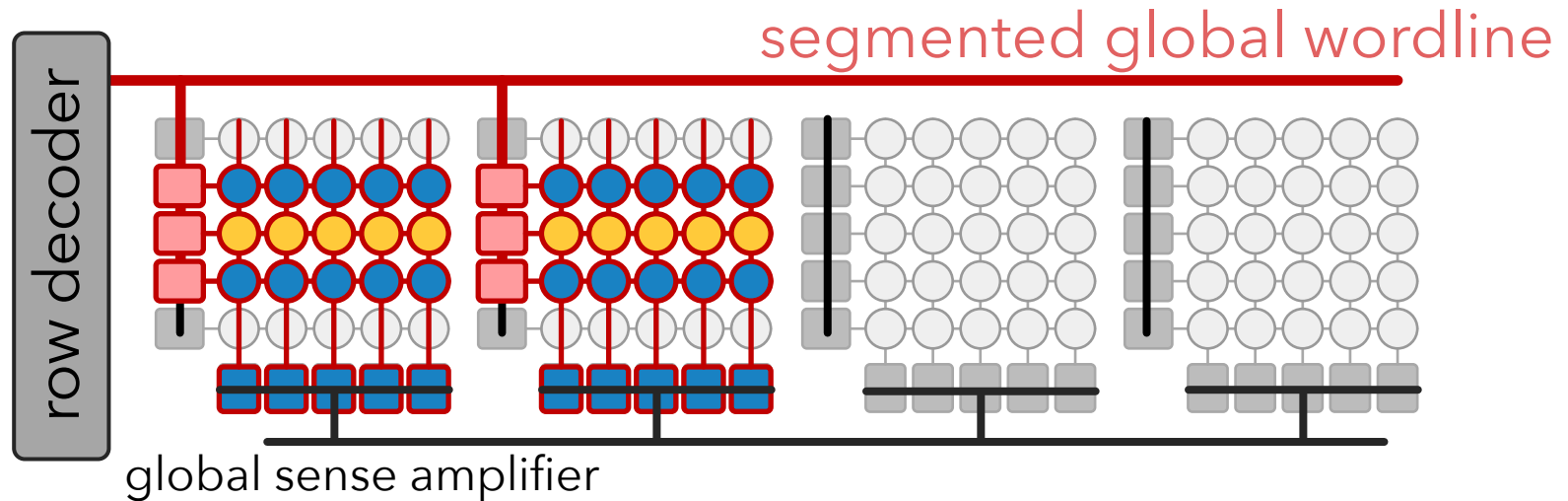
[†] *ETH Zürich*

[‡] *Univ. of Illinois Urbana-Champaign*

*Our **goal** is to design a flexible PUD system that overcomes the limitations caused by the large and rigid granularity of PUD. To this end, we propose MIMDRAM, a hardware/software co-designed PUD system that introduces new mechanisms to allocate and control only the necessary resources for a given PUD operation. The key idea of MIMDRAM is to leverage fine-grained DRAM (i.e., the ability to independently access smaller segments of a large DRAM row) for PUD computation. MIMDRAM exploits this key idea to enable a multiple-instruction multiple-data (MIMD) execution model in each DRAM subarray (and SIMD execution within each DRAM row segment).*

MIMDRAM: Key Idea

Enable narrower-width operations than a DRAM row



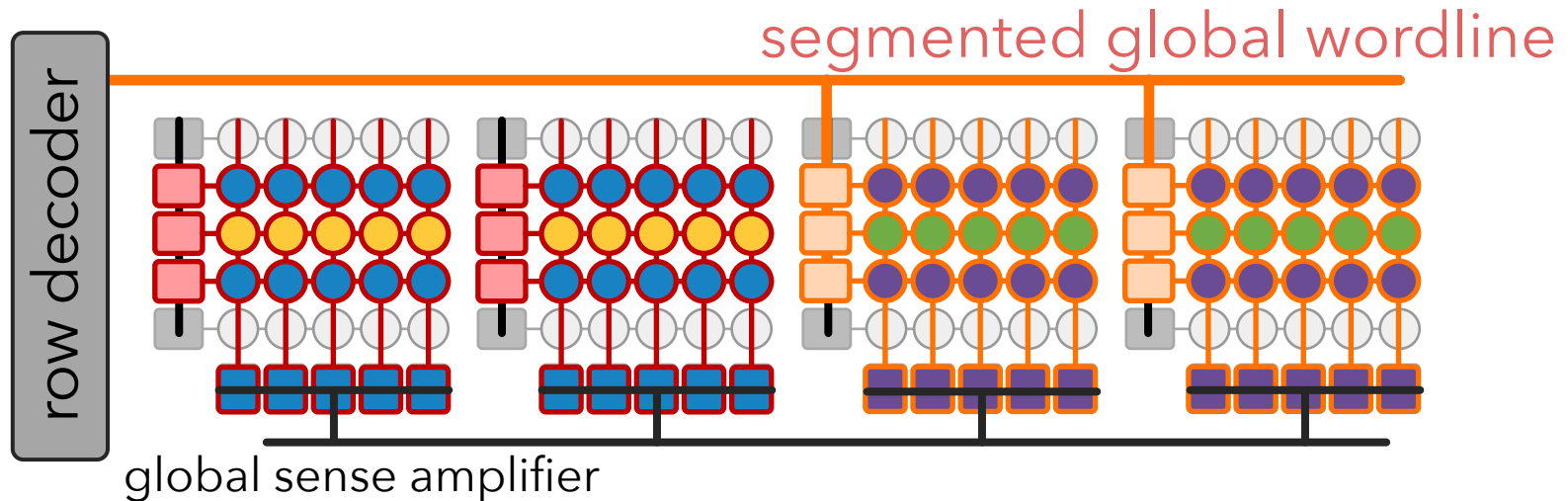
Use fine-grained DRAM for processing-using-DRAM:

1 Improves SIMD utilization

- for a single PUD operation, only access the DRAM mats with target data

MIMDRAM: Key Idea

Enable narrower-width operations than a DRAM row



Use fine-grained DRAM for processing-using-DRAM:

1 Improves SIMD utilization

- for a single PUD operation, only access the DRAM mats with target data
 - for multiple PUD operations, execute independent operations concurrently
- **multiple instruction, multiple data (MIMD) execution model**

Sectored DRAM

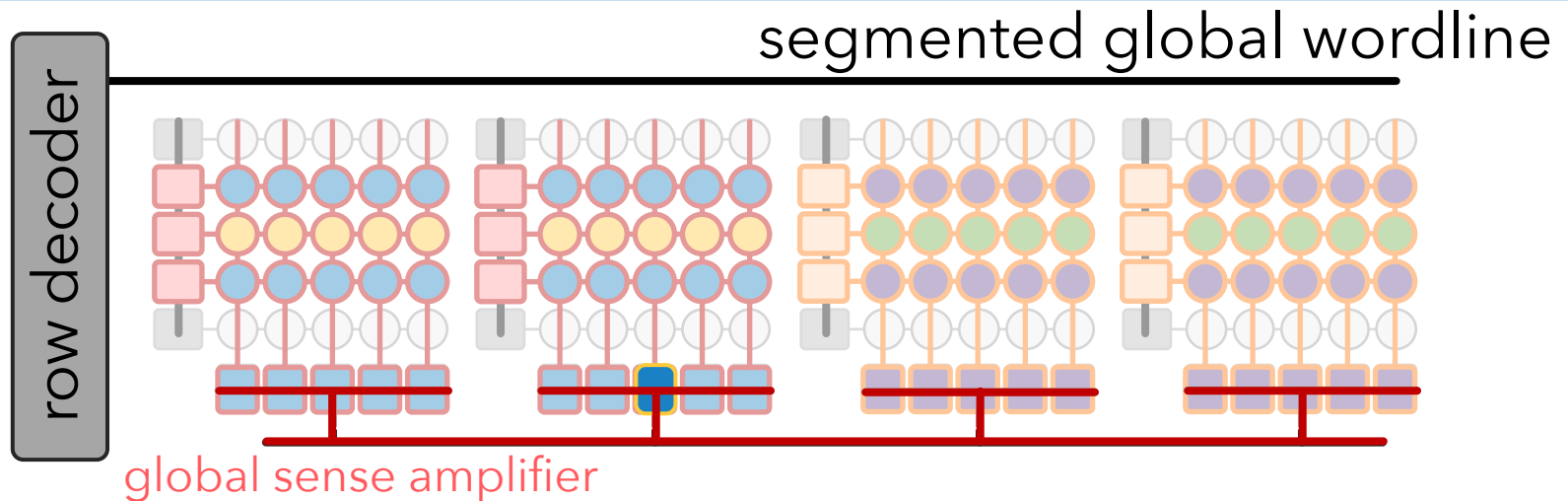
- Ataberk Olgun, F. Nisa Bostanci, Geraldo F. Oliveira, Yahya Can Tugrul, Rahul Bera, A. Giray Yaglikci, Hasan Hassan, Oguz Ergin, and Onur Mutlu, **"Sectored DRAM: A Practical Energy-Efficient and High-Performance Fine-Grained DRAM Architecture"**
ACM Transactions on Architecture and Code Optimization (TACO),
[online] June 2024.
[[arXiv version](#)]
[[ACM Digital Library version](#)]

Sectored DRAM: A Practical Energy-Efficient and High-Performance Fine-Grained DRAM Architecture

Ataberk Olgun[§] F. Nisa Bostanci^{§†} Geraldo F. Oliveira[§] Yahya Can Tuğrul^{§†} Rahul Bera[§]
A. Giray Yağlıkcı[§] Hasan Hassan[§] Oğuz Ergin[†] Onur Mutlu[§]

MIMDRAM: Key Idea

Enable narrower-width operations than a DRAM row



Use fine-grained DRAM for processing-using-DRAM:

1 Improves SIMD utilization

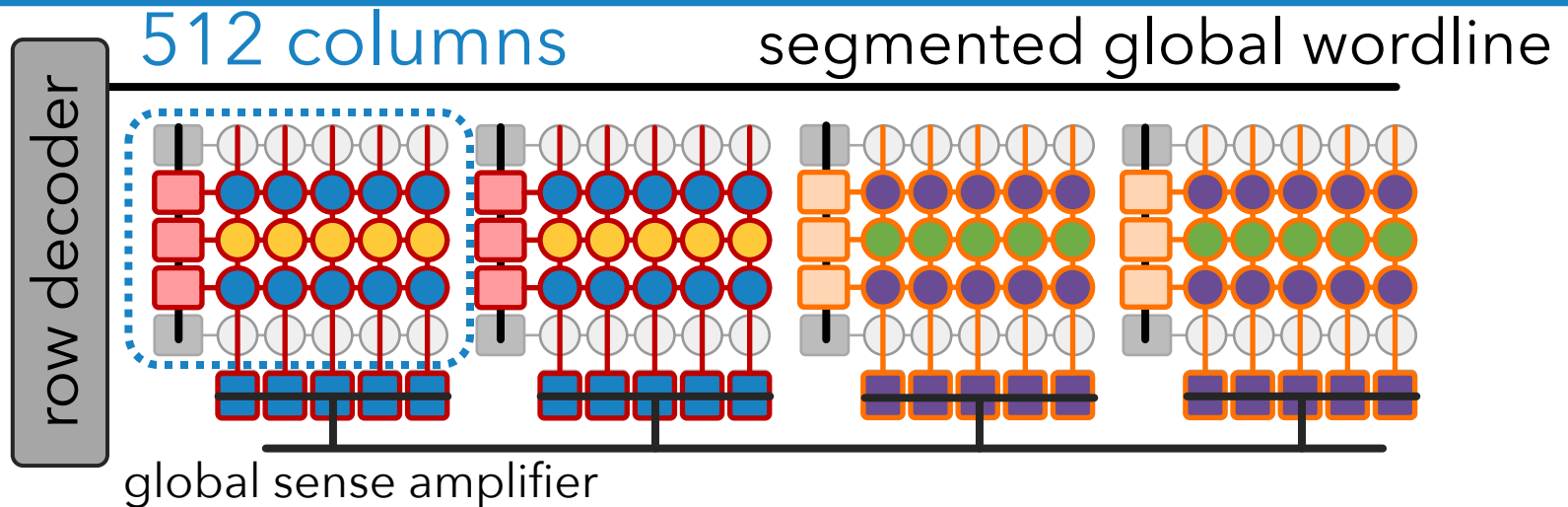
- for a single PUD operation, only access the DRAM mats with target data
- for multiple PUD operations, execute independent operations concurrently
→ **multiple instruction, multiple data (MIMD) execution model**

2 Enables low-cost interconnects for vector reduction

- global and local data buses can be used for inter-/intra-mat communication

MIMDRAM: Key Idea

Enable narrower-width operations than a DRAM row



Use fine-grained DRAM for processing-using-DRAM:

1 Improves SIMD utilization

- for a single PUD operation, only access the DRAM mats with target data
- for multiple PUD operations, execute independent operations concurrently
→ **multiple instruction, multiple data (MIMD) execution model**

2 Enables low-cost interconnects for vector reduction

- global and local data buses can be used for inter-/intra-mat communication

3 Eases programmability

- SIMD parallelism in a DRAM mat is on par with vector ISAs' SIMD width

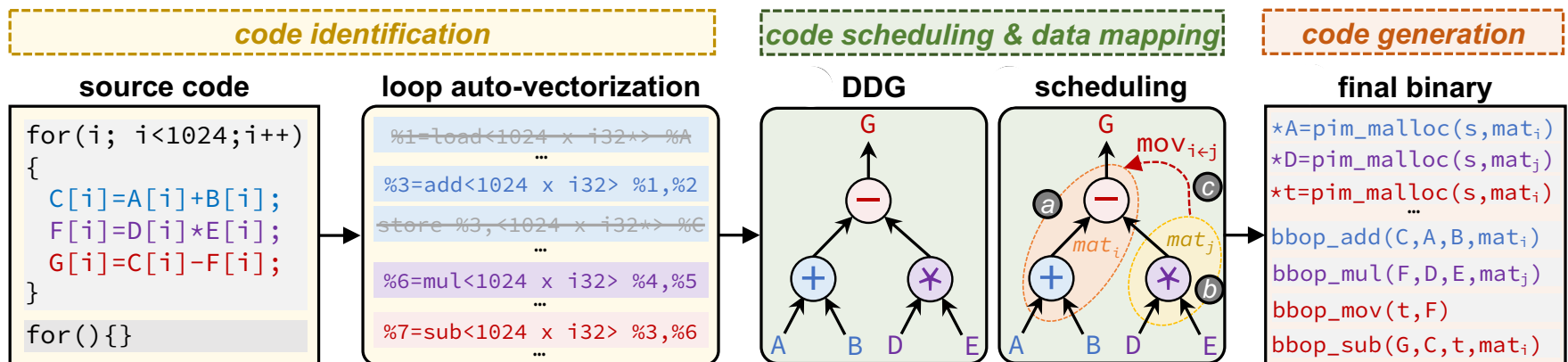
MIMDRAM: Compiler Support

Goal

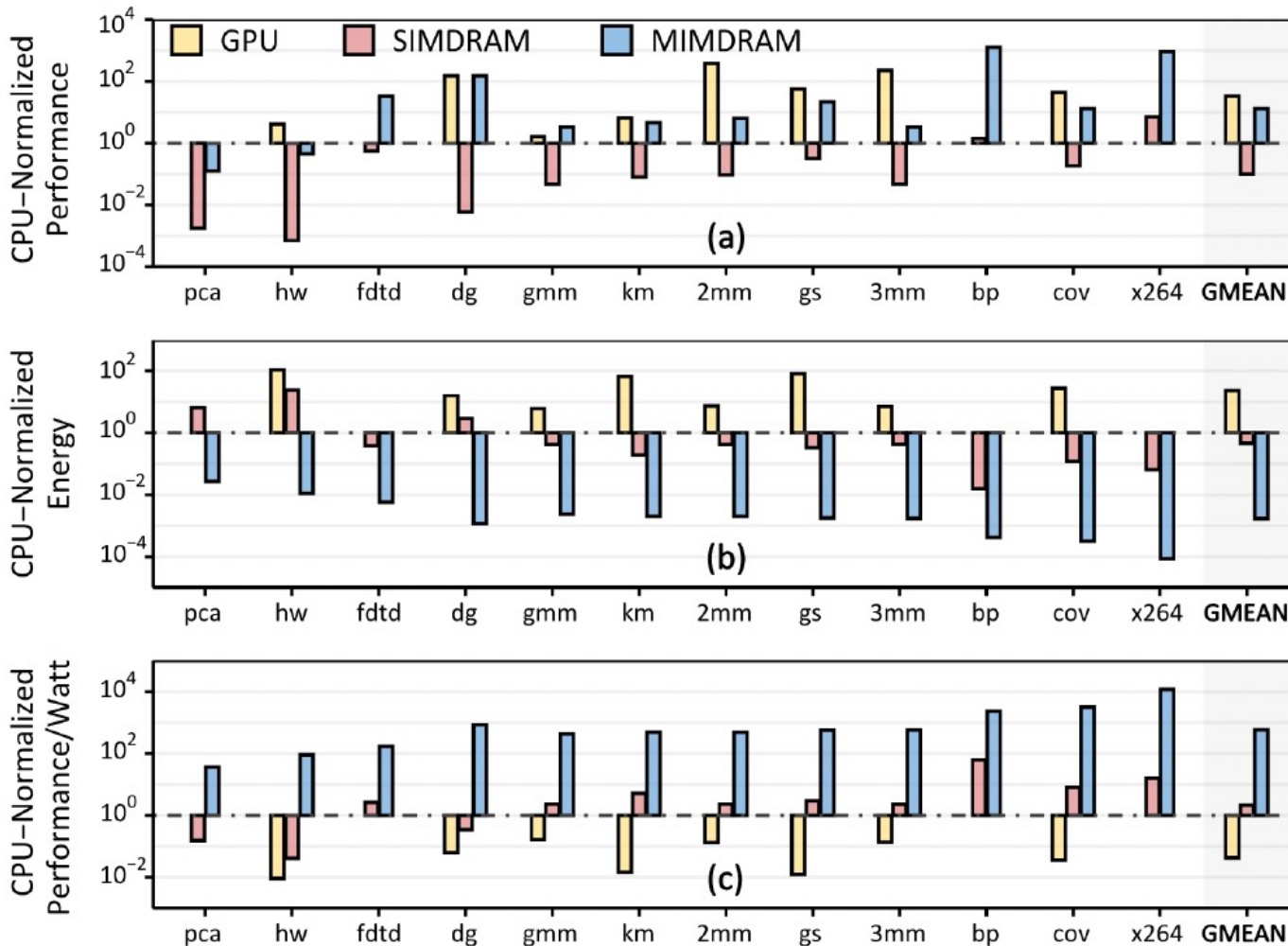
Transparently to programmer:
extract SIMD parallelism from an application, and
schedule PUD instructions while maximizing utilization



Three new LLVM-based passes targeting PUD execution



MIMDRAM Perf, Energy, Perf/Watt



582X and 13,612X the energy efficiency of CPU and GPU, respectively

Existing DRAM Chips
Are Already Quite Capable

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[§]
Hasan Hassan[§]

Konstantinos Kanellopoulos[§]
Oğuz Ergin[†] Onur Mutlu[§]

Behzad Salami^{§*}

[§]ETH Zürich

[†]TOBB ETÜ

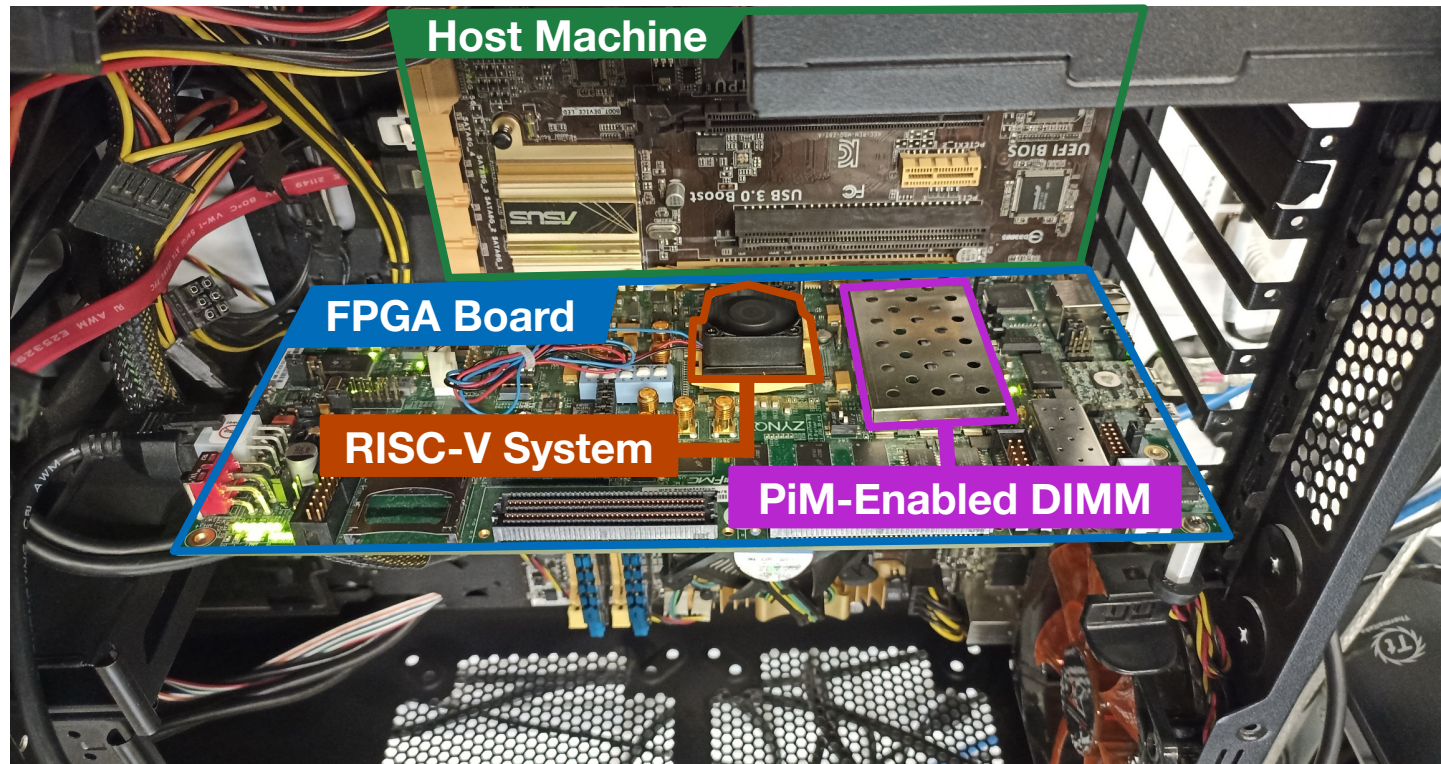
^{*}BSC

<https://arxiv.org/pdf/2111.00082.pdf>

<https://github.com/cmu-safari/pidram>

<https://www.youtube.com/watch?v=qeukNs5XI3g&t=4192s>

Real Processing-using-Memory Prototype



<https://arxiv.org/pdf/2111.00082.pdf>

<https://github.com/cmu-safari/pidram>

<https://www.youtube.com/watch?v=qeukNs5XI3g&t=4192s>

Real Processing-using-Memory Prototype

☰ README.md

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` 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`

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:

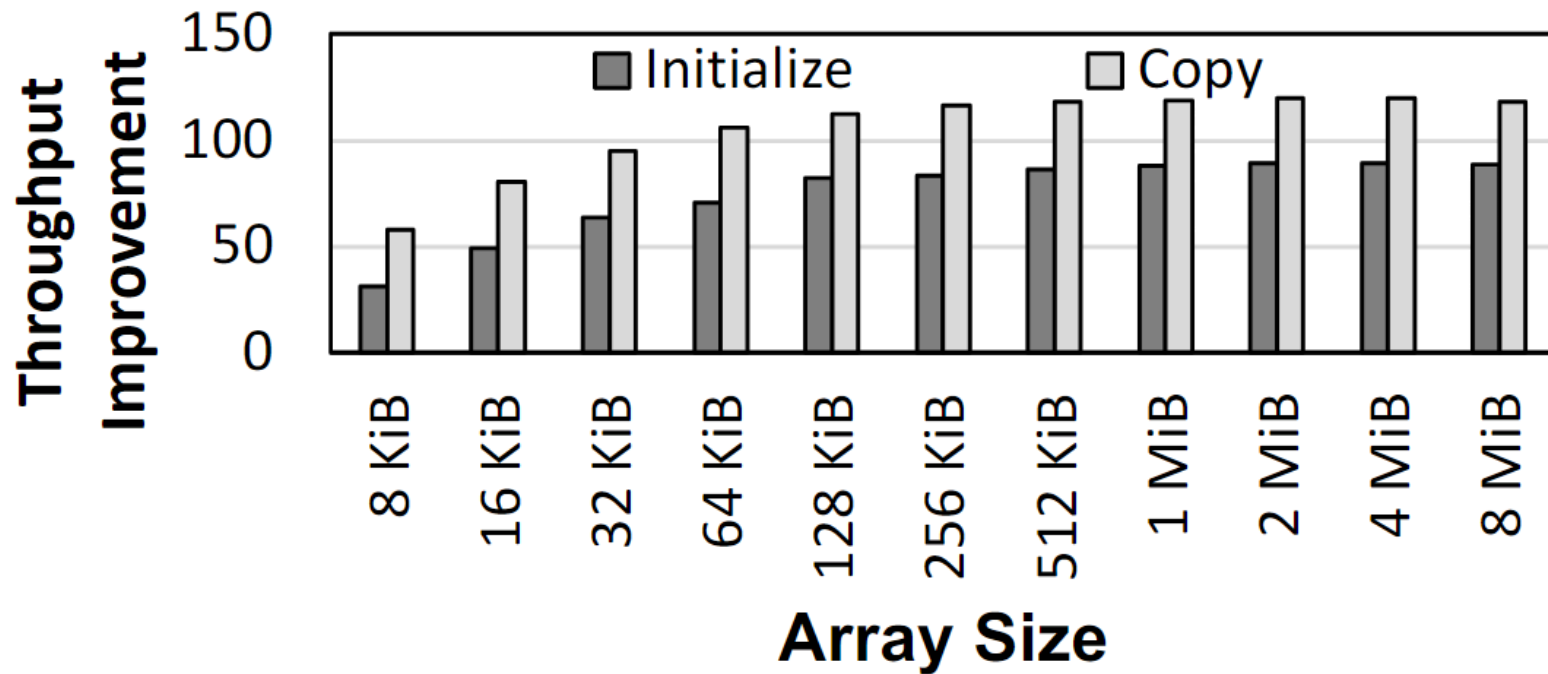
- 1- Open IP Catalog
- 2- Find "Memory Interface Generator (MIG 7 Series)" IP and double click

<https://arxiv.org/pdf/2111.00082.pdf>

<https://github.com/cmu-safari/pidram>

<https://www.youtube.com/watch?v=qeukNs5XI3g&t=4192s>

Microbenchmark Copy/Initialization Throughput



**In-DRAM Copy and Initialization
improve throughput by 119x and 89x**

More on PiDRAM

- Ataberk Olgun, Juan Gomez Luna, Konstantinos Kanellopoulos, Behzad Salami, Hasan Hassan, Oguz Ergin, and Onur Mutlu,
["PiDRAM: A Holistic End-to-end FPGA-based Framework for Processing-in-DRAM"](#)
[ACM Transactions on Architecture and Code Optimization \(TACO\)](#), March 2023.
[\[arXiv version\]](#)
Presented at the [18th HiPEAC Conference](#), Toulouse, France, January 2023.
[\[Slides \(pptx\) \(pdf\)\]](#)
[\[Longer Lecture Slides \(pptx\) \(pdf\)\]](#)
[\[Lecture Video \(40 minutes\)\]](#)
[\[PiDRAM Source Code\]](#)

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 University of Economics and Technology*

DRAM Chips Are Already (Quite) Capable!

- **Appears at HPCA 2024** <https://arxiv.org/pdf/2402.18736.pdf>

Functionally-Complete Boolean Logic in Real DRAM Chips: Experimental Characterization and Analysis

İsmail Emir Yüksel Yahya Can Tuğrul Ataberk Olgun F. Nisa Bostancı A. Giray Yağlıkçı
Geraldo F. Oliveira Haocong Luo Juan Gómez-Luna Mohammad Sadrosadati Onur Mutlu

ETH Zürich

We experimentally demonstrate that COTS DRAM chips are capable of performing 1) functionally-complete Boolean operations: NOT, NAND, and NOR and 2) many-input (i.e., more than two-input) AND and OR operations. We present an extensive characterization of new bulk bitwise operations in 256 off-the-shelf modern DDR4 DRAM chips. We evaluate the reliability of these operations using a metric called success rate: the fraction of correctly performed bitwise operations. Among our 19 new observations, we highlight four major results. First, we can perform the NOT operation on COTS DRAM chips with 98.37% success rate on average. Second, we can perform up to 16-input NAND, NOR, AND, and OR operations on COTS DRAM chips with high reliability (e.g., 16-input NAND, NOR, AND, and OR with average success rate of 94.94%, 95.87%, 94.94%, and 95.85%, respectively). Third, data pattern only slightly

The Capability of COTS DRAM Chips

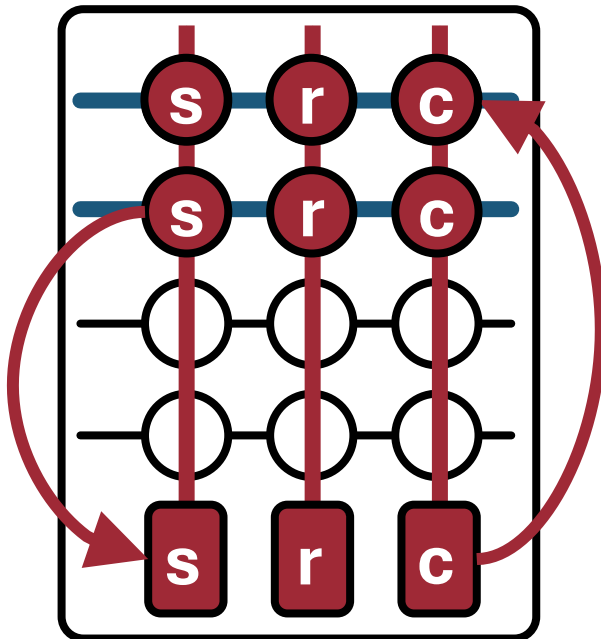
We demonstrate that COTS DRAM chips:

- 1 Can copy one row into up to 31 other rows with **>99.98%** success rate
- 2 Can perform **NOT operation** with up to **32 output operands**
- 3 Can perform up to **16-input AND, NAND, OR, and NOR** operations

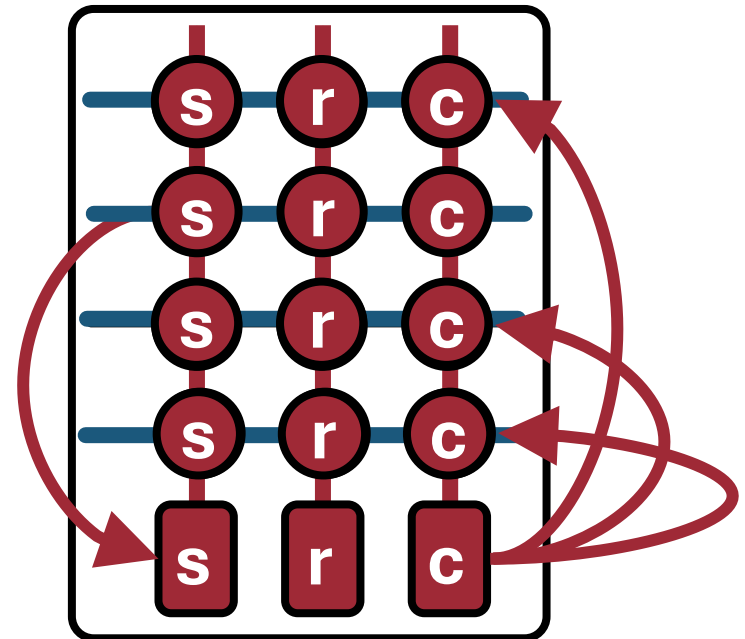
In-DRAM Multiple Row Copy (Multi-RowCopy)

Simultaneously activate many rows to copy **one row's content** to **multiple destination rows**

RowClone

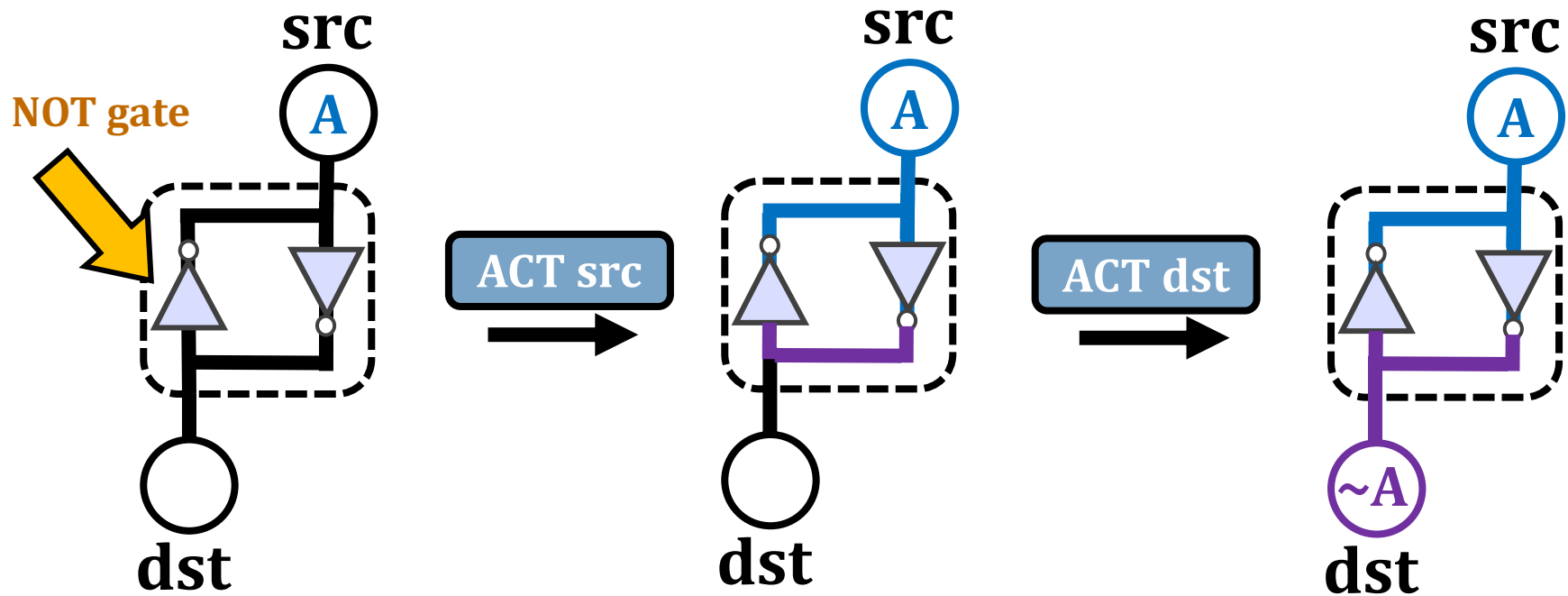


Multi-RowCopy



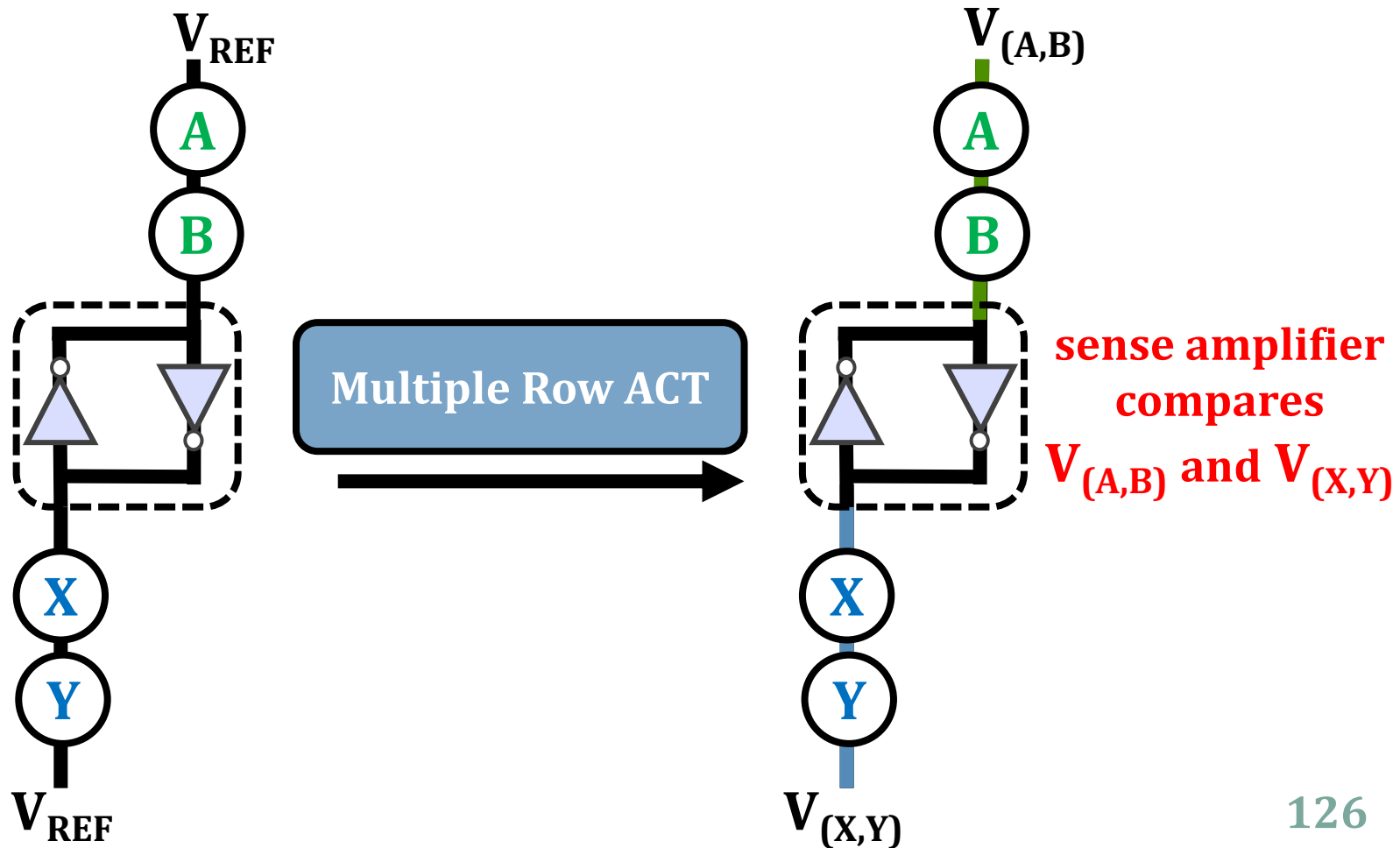
Key Idea: NOT Operation

Connect rows in neighboring subarrays through a **NOT gate** by consecutively activating rows

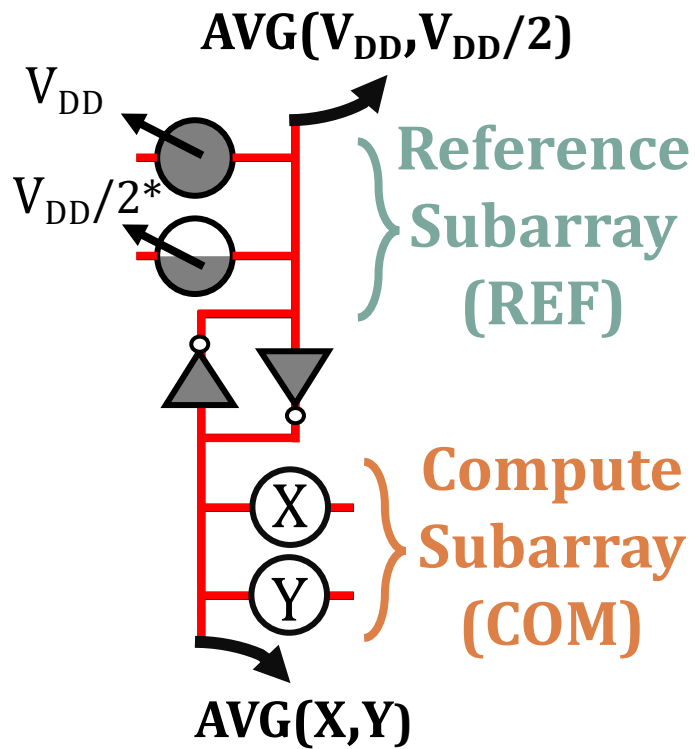


Key Idea: NAND, NOR, AND, OR

Manipulate the bitline voltage to express
a wide variety of functions using
simultaneous multi-row activation in neighboring subarrays



Two-Input AND and NAND Operations



$V_{DD}=1$ & $\text{GND} = 0$

X	Y	COM	REF
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0
		AND	NAND

Many-Input AND, NAND, OR, and NOR Operations

We can express AND, NAND, OR, and NOR operations by carefully manipulating the reference voltage

Functionally-Complete Boolean Logic in Real DRAM Chips: Experimental Characterization and Analysis

İsmail Emir Yüksel Yahya Can Tuğrul Ataberk Olgun F. Nisa Bostancı A. Giray Yağlıkçı
Geraldo F. Oliveira Haocong Luo Juan Gómez-Luna Mohammad Sadrosadati Onur Mutlu

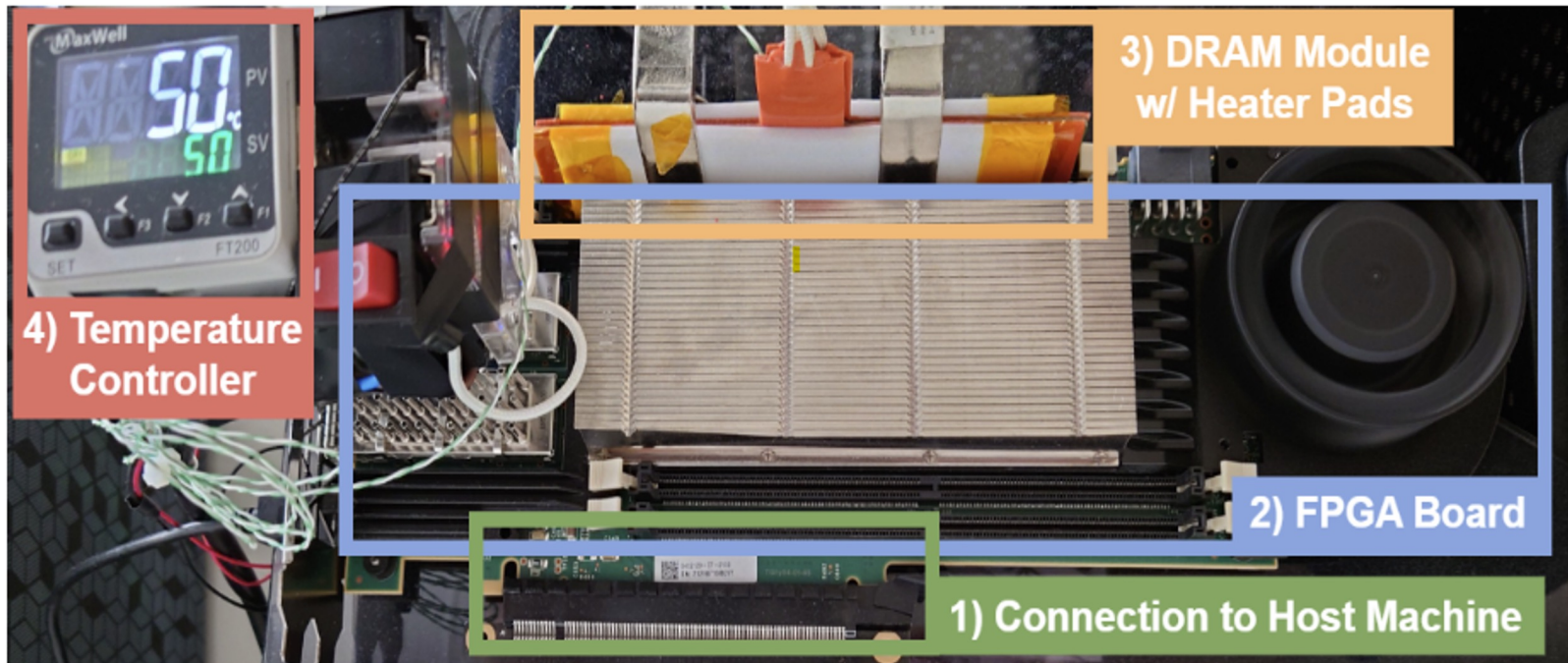
ETH Zürich

(More details in the paper)

<https://arxiv.org/pdf/2402.18736.pdf>

DRAM Testing Infrastructure

- Developed from [DRAM Bender \[Olgun+, TCAD'23\]*](#)
- **Fine-grained control** over DRAM commands, timings, and temperature

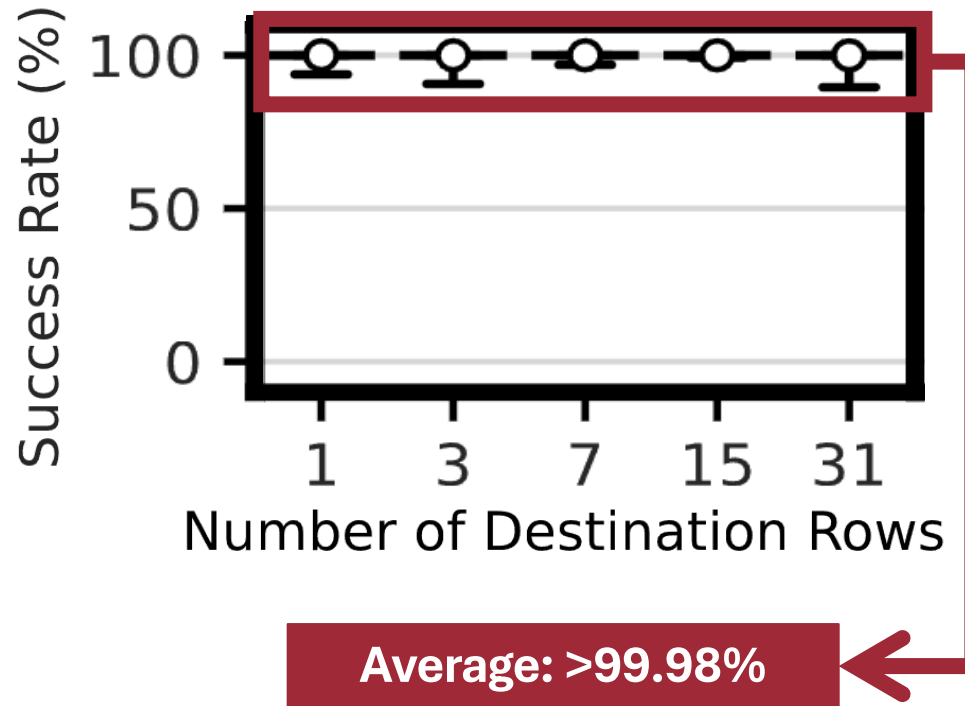


DRAM Chips Tested

- 256 DDR4 chips from two major DRAM manufacturers
- Covers different die revisions and chip densities

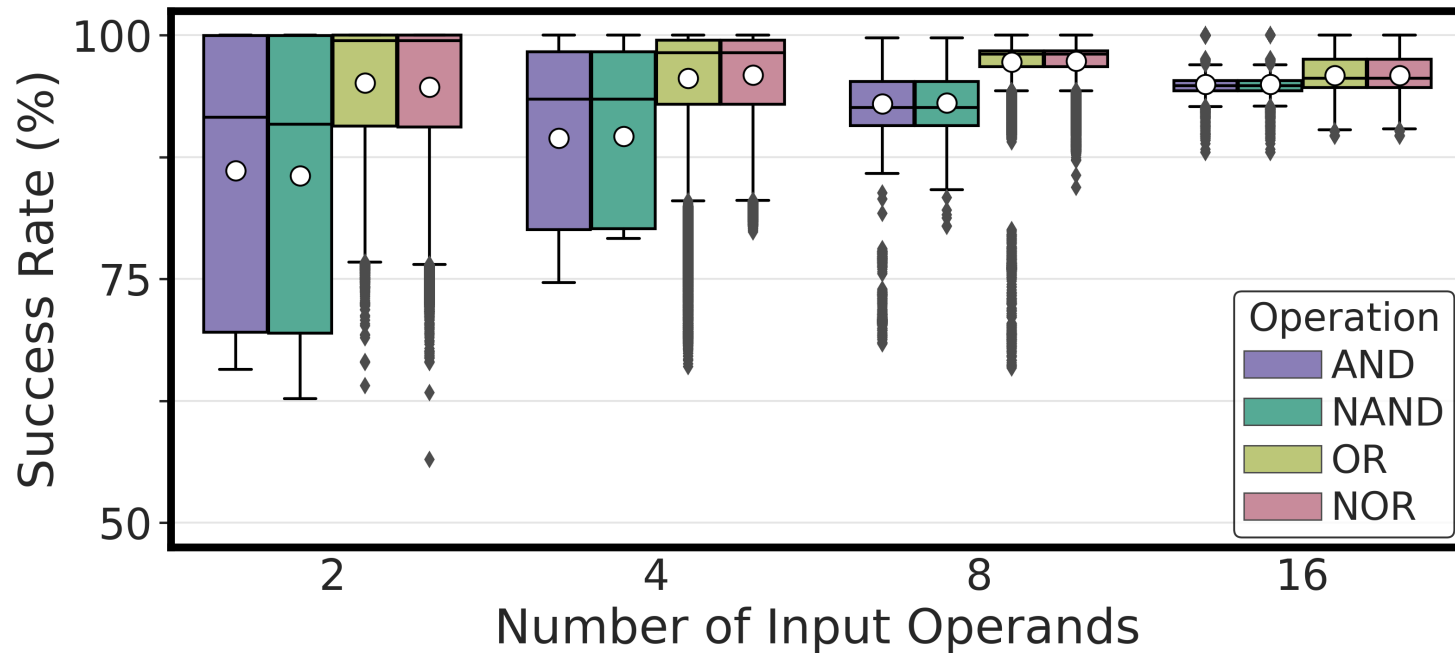
Chip Mfr.	#Modules (#Chips)	Die Rev.	Mfr. Date ^a	Chip Density	Chip Org.	Speed Rate
SK Hynix	9 (72)	M	N/A	4Gb	x8	2666MT/s
	5 (40)	A	N/A	4Gb	x8	2133MT/s
	1 (16)	A	N/A	8Gb	x8	2666MT/s
	1 (32)	A	18-14	4Gb	x4	2400MT/s
	1 (32)	A	16-49	8Gb	x4	2400MT/s
	1 (32)	M	16-22	8Gb	x4	2666MT/s
Samsung	1 (8)	F	21-02	4Gb	x8	2666MT/s
	2 (16)	D	21-10	8Gb	x8	2133MT/s
	1 (8)	A	22-12	8Gb	x8	3200MT/s

Robustness of Multi-RowCopy



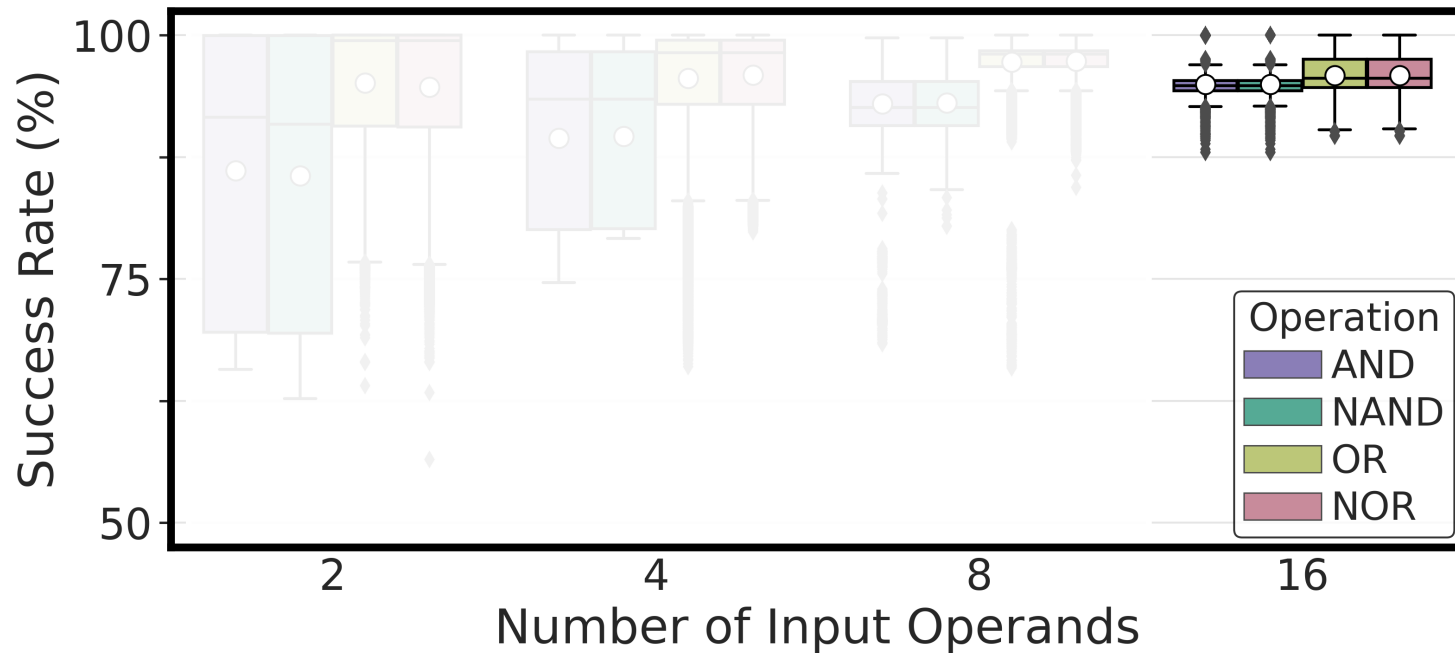
COTS DRAM chips can copy one row's content to up to 31 rows with a very high success rate

Performing AND, NAND, OR, and NOR



COTS DRAM chips can perform {2, 4, 8, 16}-input AND, NAND, OR, and NOR operations

Performing AND, NAND, OR, and NOR



**COTS DRAM chips can perform
16-input AND, NAND, OR, and NOR operations
with very high success rate (>94%)**

More on Functionally-Complete DRAM

- Ismail Emir Yüksel, Yahya Can Tuğrul, Ataberk Olgun, F. Nisa Bostancı, A. Giray Yağlıkçı, Geraldo F. Oliveira, Haocong Luo, Juan Gomez-Luna, Mohammad Sadrosadati, and Onur Mutlu,
"Functionally-Complete Boolean Logic in Real DRAM Chips: Experimental Characterization and Analysis"
Proceedings of the 30th International Symposium on High-Performance Computer Architecture (HPCA), April 2024.
[[Slides \(pptx\)](#)] [[pdf](#)]
[[arXiv version](#)]
[[FCDRAM Source Code](#)]

Functionally-Complete Boolean Logic in Real DRAM Chips: Experimental Characterization and Analysis

Ismail Emir Yüksel Yahya Can Tuğrul Ataberk Olgun F. Nisa Bostancı A. Giray Yağlıkçı
Geraldo F. Oliveira Haocong Luo Juan Gómez-Luna Mohammad Sadrosadati Onur Mutlu

ETH Zürich

More on Multi-Row Copy

- Ismail Emir Yuksel, Yahya Can Tugrul, F. Nisa Bostanci, Geraldo F. Oliveira, A. Giray Yaglikci, Ataberk Olgun, Melina Soysal, Haocong Luo, Juan Gomez-Luna, Mohammad Sadrosadati, and Onur Mutlu,

"Simultaneous Many-Row Activation in Off-the-Shelf DRAM Chips: Experimental Characterization and Analysis"

Proceedings of the 54th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Brisbane, Australia, June 2024.

[[Slides \(pptx\)](#) ([pdf](#))]

[[arXiv version](#)]

[[SiMRA-DRAM Source Code](#) (Officially Artifact Evaluated with All Badges)]

Officially artifact evaluated as both code and dataset available, reviewed and reproducible.



Simultaneous Many-Row Activation in Off-the-Shelf DRAM Chips: Experimental Characterization and Analysis

İsmail Emir Yüksel¹ Yahya Can Tuğrul^{1,2} F. Nisa Bostancı¹ Geraldo F. Oliveira¹
A. Giray Yağlıkçı¹ Ataberk Olgun¹ Melina Soysal¹ Haocong Luo¹
Juan Gómez-Luna¹ Mohammad Sadrosadati¹ Onur Mutlu¹

¹ETH Zürich

²TOBB University of Economics and Technology

What Else Can We Do Using Commodity Memories?

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"

Proceedings of the 25th International Symposium on High-Performance Computer Architecture (HPCA), Washington, DC, USA, February 2019.

[[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"
Proceedings of the 48th International Symposium on Computer Architecture (ISCA), Virtual, June 2021.
[[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^{§⊙}

Oğuz Ergin[†]

Onur Mutlu[§]

[§]ETH Zürich

[†]TOBB University of Economics and Technology

[⊙]University of Toronto

In-DRAM TRNG: Recent Results

■ N-row Activation

- initialize cell values to sample random values in sense amplifiers

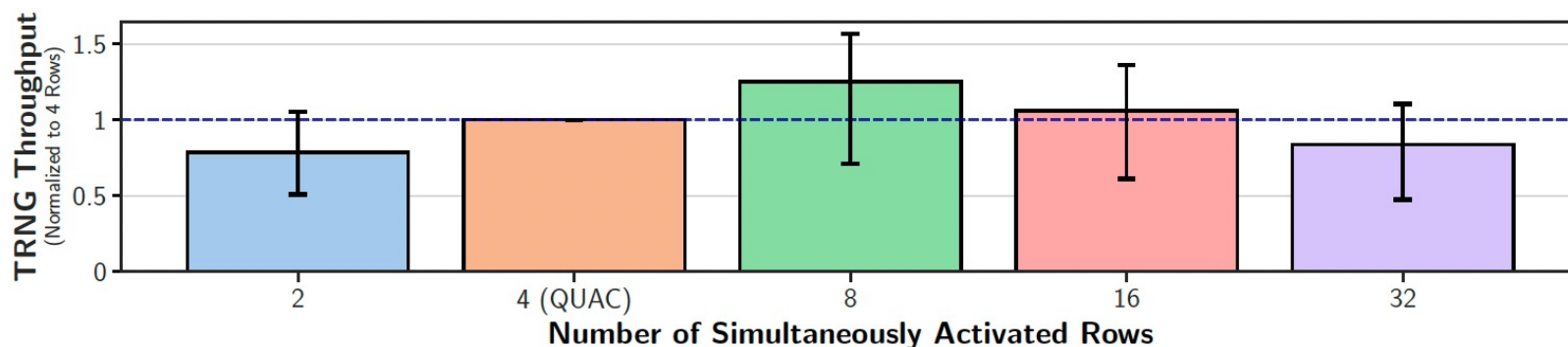


Fig. 11: **Throughput of generating true random numbers, as measured in 96 COTS DRAM chips using multiple-row activation, normalized to state-of-the-art DRAM-based TRNG, QUAC-TRNG (i.e., 4-row activation) [135].** Each error bar shows the range across all tested chips. We observe that random numbers that are generated with multiple-row activation and then post-processed with the SHA-256 function [221] pass *all* NIST STS tests [222], which means 2-, 4-, 8-, 16-, and 32-row activation generates high-quality true random bitstreams. On average, 8- and 16-row activation-based TRNG outperforms the state-of-the-art by $1.25\times$ and $1.06\times$, respectively, while 2- and 32-row activation-based TRNG provides $0.69\times$ and $0.84\times$ the throughput of the state-of-the-art.

Mutlu+, “[Memory-Centric Computing: Recent Advances in Processing-in-DRAM](#),” IEDM 2024.

In-DRAM True Random Number Generation

- F. Nisa Bostanci, Ataberk Olgun, Lois Orosa, A. Giray Yaglikci, Jeremie S. Kim, Hasan Hassan, Oguz 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-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"
Proceedings of the 24th International Symposium on High-Performance Computer Architecture (HPCA), Vienna, Austria, February 2018.
[[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 Lookup-Table Based Execution

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

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

Proceedings of the 55th International Symposium on Microarchitecture (MICRO), Chicago, IL, USA, October 2022.

[[Slides \(pptx\)](#) ([pdf](#))]

[[Longer Lecture Slides \(pptx\)](#) ([pdf](#))]

[[Lecture Video](#) (26 minutes)]

[[arXiv version](#)]

[[Source Code](#) (Officially Artifact Evaluated with All Badges)]

Officially artifact evaluated as available, reusable and reproducible.



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

In-Flash Bulk Bitwise Execution

- Jisung Park, Roknoddin Azizi, Geraldo F. Oliveira, Mohammad Sadrosadati, Rakesh Nadig, David Novo, Juan Gómez-Luna, Myungsuk Kim, and Onur Mutlu,
"Flash-Cosmos: In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory"
Proceedings of the 55th International Symposium on Microarchitecture (MICRO),
Chicago, IL, USA, October 2022.
[[Slides \(pptx\)](#)] [[pdf](#)]
[[Longer Lecture Slides \(pptx\)](#)] [[pdf](#)]
[[Lecture Video](#) (44 minutes)]
[[arXiv version](#)]

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-Flash Homomorphic Encryption

- Mayank Kabra, Rakesh Nadig, Harshita Gupta, Rahul Bera, Manos Frouzakis, Vamanan Arulchelvan, Yu Liang, Haiyu Mao, Mohammad Sadrosadati, and Onur Mutlu, **"CIPHERMATCH: Accelerating Homomorphic Encryption based String Matching via Memory-Efficient Data Packing and In-Flash Processing"** *Proceedings of the 30th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Rotterdam, Netherlands, April 2025.
[[Slides \(pptx\)](#)] [[pdf](#)]

CIPHERMATCH: Accelerating Homomorphic Encryption-Based String Matching via Memory-Efficient Data Packing and In-Flash Processing

Mayank Kabra[†] Rakesh Nadig[†] Harshita Gupta[†] Rahul Bera[†] Manos Frouzakis[†]
Vamanan Arulchelvan[†] Yu Liang[†] Haiyu Mao[‡] Mohammad Sadrosadati[†] Onur Mutlu[†]
ETH Zurich[†] King's College London[‡]

Processing in Memory: Two Types

1. Processing **near** Memory
2. Processing **using** Memory

PIM Review and Open Problems

A Modern Primer on Processing-In-Memory

Onur Mutlu^a, Saugata Ghose^b, Juan Gómez-Luna^c, Rachata Ausavarungnirun^d,
Mohammad Sadrosadati^a, Geraldo F. Oliveira^a

SAFARI Research Group

^a*ETH Zürich*

^b*University of Illinois Urbana-Champaign*

^c*NVIDIA Research*

^d*MangoBoost Inc.*

Onur Mutlu, Saugata Ghose, Juan Gomez-Luna, Rachata Ausavarungnirun,
Mohammad Sadrosadati, and Geraldo F. Oliveira,

"A Modern Primer on Processing in Memory"

*Invited Book Chapter in **Emerging Computing: From Devices to Systems - Looking Beyond Moore and Von Neumann**, Springer, 2022.*

A Recent Short Paper [IMW 2025]

- Onur Mutlu, Ataberk Olgun, and İsmail Emir Yüksel,
**"Memory-Centric Computing: Solving Computing's
Memory Problem"**

*Invited Paper in Proceedings of the 17th IEEE International
Memory Workshop (**IMW**), Monterey, CA, USA, May 2025.*
[Slides (pptx) (pdf)]

Memory-Centric Computing: Solving Computing's Memory Problem

Onur Mutlu Ataberk Olgun İsmail Emir Yüksel

ETH Zürich

How to Enable Adoption of Processing in Memory

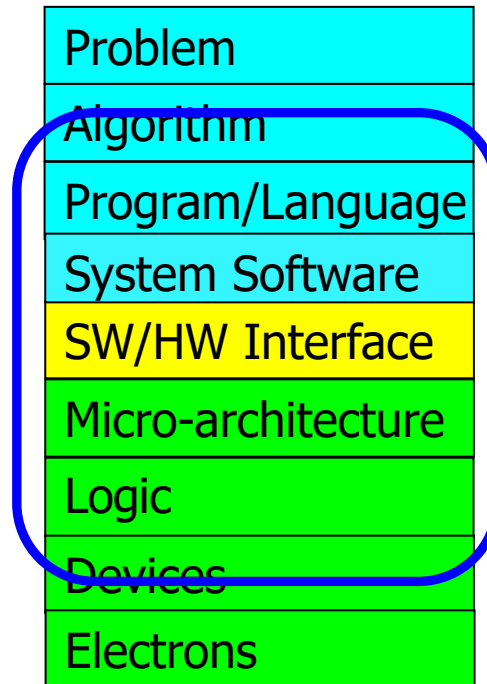
Potential Barriers to Adoption of PIM

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

- With a **memory-centric mindset**



We can get there step by step

A Very Recent PhD Thesis

- <https://safari.ethz.ch/geraldo-francisco-de-oliveira-junior-successfully-defends-his-phd/>

New Tools, Programming Models, and System Support for Processing-in-Memory Architectures

Geraldo F. Oliveira

Doctoral Examination

29 April 2025

Advisor:

Onur Mutlu (ETH Zürich)

Co-Examiners:

Christian Weis (RPTU)

Donghyuk Lee (NVIDIA Research)

Reetuparna Das (University of Michigan)

Tony Nowatzki (UCLA)

Concluding Remarks

Fundamentally Energy-Efficient **(Data-Centric)** Computing Architectures

Fundamentally High-Performance **(Data-Centric)** Computing Architectures

Computing Architectures with Minimal Data Movement

Concluding Remarks

- Goal: Enable computation capability in memory
- We highlighted major recent advances in Processing-in-DRAM
 - ❑ Can lead to **orders-of-magnitude energy & perf** improvements
 - ❑ **Unmodified DRAM chips are already capable of computation**
- Memory should be designed as a **combined computation and storage substrate**
 - ❑ Not as an inactive storage substrate
 - ❑ Design mindset and flow should change
- Future of **truly memory-centric computing** is bright
 - ❑ We need to do research & design across the computing stack
 - ❑ With a proper mindset and infrastructure shift



Data-centric

Data-driven

Data-aware

A Blueprint for Fundamentally Better Architectures

- Onur Mutlu,
"Intelligent Architectures for Intelligent Computing Systems"
*Invited Paper in Proceedings of the Design, Automation, and Test in Europe Conference (**DATE**), Virtual, February 2021.*
[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

PIM Tutorial November 2024 Edition

MICRO 2024 - Tutorial on Memory-Centric Computing Systems

Saturday, November 2nd, Austin, Texas, USA

Organizers: Geraldo F. Oliveira, Dr. Mohammad Sadrosadati,
Ataberk Olgun, Professor Onur Mutlu

Program: <https://events.safari.ethz.ch/micro24-memorycentric-tutorial/>

Overview of PIM | PIM taxonomy
PIM in memory & storage
Real-world PNM systems
PUM for bulk bitwise operations
Programming techniques & tools
Infrastructures for PIM Research
Research challenges & opportunities



<https://www.youtube.com/watch?v=KV2MXvcBgb0>

<https://events.safari.ethz.ch/micro24-memorycentric-tutorial/>

PIM Tutorial @ PPOPP/HPCA/CGO/CC

PPoPP 2025 - Tutorial on Memory-Centric Computing Systems

March 1st, Las Vegas, Nevada, USA

Organizers: Geraldo F. Oliveira, Dr. Mohammad Sadrosadati,
Ataberk Olgun, Professor Onur Mutlu

Program: <https://events.safari.ethz.ch/ppopp25-memorycentric-tutorial/>

Overview of PIM | PIM taxonomy
PIM in memory & storage
Real-world PNM systems
PUM for bulk bitwise operations
Programming techniques & tools
Infrastructures for PIM Research
Research challenges & opportunities



<https://www.youtube.com/live/NkDY6osus6g>

<https://events.safari.ethz.ch/ppopp25-memorycentric-tutorial/> 161

PIM Tutorial/Workshop @ ASPLOS 2025

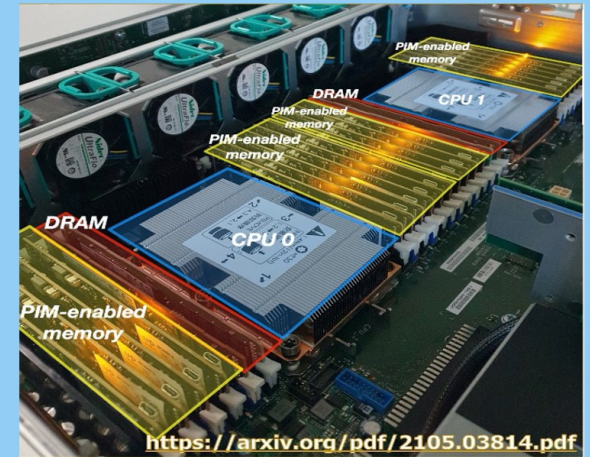
ASPLOS 2025 - 1st Workshop on Memory-Centric Computing Systems

Sunday, March 30th, Rotterdam, The Netherlands

Organizers: Geraldo F. Oliveira, Dr. Mohammad Sadrosadati, Ataberk Olgun, Professor Onur Mutlu

Program: <https://events.safari.ethz.ch/asplos25-MCCSys/doku.php>

Overview of PIM | PIM taxonomy
PIM in memory & storage
Real-world PNM systems
PUM for bulk bitwise operations
Programming techniques & tools
Infrastructures for PIM Research
Research challenges & opportunities



<https://events.safari.ethz.ch/asplos25-MCCSys/doku.php>

PIM Tutorial/Workshop @ ICS 2025

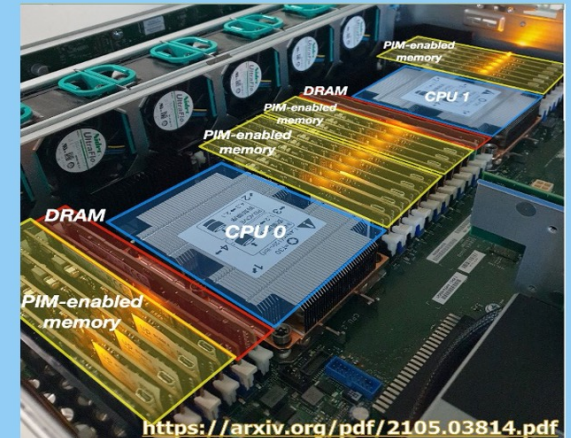
ICS 2025 - 2nd Workshop on Memory-Centric Computing Systems

Sunday, June 8th, Salt Lake City, USA

Organizers: Geraldo F. Oliveira, Dr. Mohammad Sadrosadati,
Ataberk Olgun, Professor Onur Mutlu

Program: <https://events.safari.ethz.ch/ics25-MCCSys/doku.php>

Overview of PIM | PIM taxonomy
PIM in memory & storage
Real-world PNM systems
PUM for bulk bitwise operations
Programming techniques & tools
Infrastructures for PIM Research
Research challenges & opportunities



<https://events.safari.ethz.ch/ics25-MCCSys/doku.php>

PIM Tutorial/Workshop @ ISCA 2025

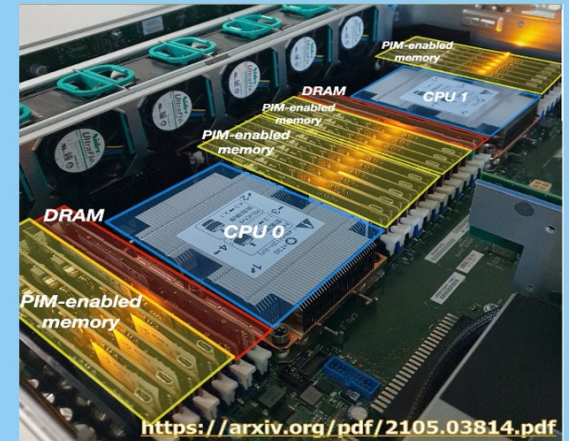
ISCA 2025 - 3rd Workshop on Memory-Centric Computing Systems

Saturday, 21st June, 2025, Tokyo, Japan

Organizers: Geraldo F. Oliveira, Dr. Mohammad Sadrosadati,
Ataberk Olgun, Professor Onur Mutlu

Program: <https://events.safari.ethz.ch/isca25-MCCSys/doku.php>

Overview of PIM | PIM taxonomy
PIM in memory & storage
Real-world PNM systems
PUM for bulk bitwise operations
Programming techniques & tools
Infrastructures for PIM Research
Research challenges & opportunities



<https://events.safari.ethz.ch/isca25-MCCSys/doku.php>

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.

👤 440 followers

📍 ETH Zurich and Carnegie Mellon U...

🔗 <https://safari.ethz.ch/>

✉ omutlu@gmail.com

🏠 Overview

📁 Repositories 80

📁 Projects

📁 Packages

👤 People 13

📁 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++ ☆ 583 🍴 209

📁 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 ☆ 137 🍴 50

📁 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++ ☆ 277 🍴 149

📁 rowhammer Public

Source code for testing the Row Hammer error mechanism in DRAM devices. Described in the ISCA 2014 paper by Kim et al. at http://users.ece.cmu.edu/~omutlu/pub/dram-row-hammer_isca14.pdf.

● C ☆ 217 🍴 42

📁 SoftMC Public

SoftMC is an experimental FPGA-based memory controller design that can be used to develop tests for DDR3 SODIMMs using a C++ based API. The design, the interface, and its capabilities and limitatio...

● Verilog ☆ 127 🍴 28

📁 Pythia Public

A customizable hardware prefetching framework using online reinforcement learning as described in the MICRO 2021 paper by Bera et al. (<https://arxiv.org/pdf/2109.12021.pdf>).

● C++ ☆ 117 🍴 36

<https://github.com/CMU-SAFARI/>

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/>

Funding Acknowledgments

- Alibaba, AMD, ASML, [Bytedance](#), [Google](#), Facebook, [Futurewei](#), [Hi-Silicon](#), HP Labs, [Huawei](#), IBM, [Intel](#), [Microsoft](#), Nvidia, Oracle, Qualcomm, Rambus, Samsung, Seagate, [VMware](#), [Xilinx](#)
- NSF
- NIH
- GSRC
- [SRC](#)
- CyLab
- [EFCL](#)
- [SNSF](#)
- [ACCESS](#)

Thank you!

Acknowledgments

SAFARI

SAFARI Research Group

safari.ethz.ch

Think BIG, Aim HIGH!

<https://safari.ethz.ch>

SAFARI Newsletter July 2024 Edition

- <https://safari.ethz.ch/safari-newsletter-july-2024/>



Memory-Centric Computing

Enabling Fundamentally Efficient & Intelligent Machines

Onur Mutlu

omutlu@gmail.com

<https://people.inf.ethz.ch/omutlu>

15 July 2025

APPT 2025 Keynote Speech

SAFARI

ETH zürich

Backup Slides

Concluding Remarks

- **Computing has a huge memory problem**
- We can solve it by designing **memory-centric systems**
 - **Memory autonomously manages itself** → technology scaling
 - **Memory performs computation** → app & system scaling
- Major advances in **memory-centric DRAM systems**
 - Can lead to **orders-of-magnitude energy & perf** improvements
 - **Unmodified DRAM chips are already capable of computation**
- Memory → **combined computation and storage substrate**
 - Design mindset and flow should change
 - Need research & design across the computing stack

Self-Managing DRAM

Better Partitioning of DRAM & Controller

- Hasan Hassan, Ataberk Olgun, A. Giray Yaglikci, Haocong Luo, and Onur Mutlu,
"Self-Managing DRAM: A Low-Cost Framework for Enabling Autonomous and Efficient DRAM Maintenance Operations"
Proceedings of the 57th International Symposium on Microarchitecture (MICRO), Austin, TX, USA, November 2024.
[[Slides \(pptx\)](#)] [[pdf](#)]
[[SelfManagingDRAM Source Code](#)]

Self-Managing DRAM: A Low-Cost Framework for Enabling Autonomous and Efficient DRAM Maintenance Operations

Hasan Hassan[†]

Ataberk Olgun[†]

A. Giray Yağlıkçı

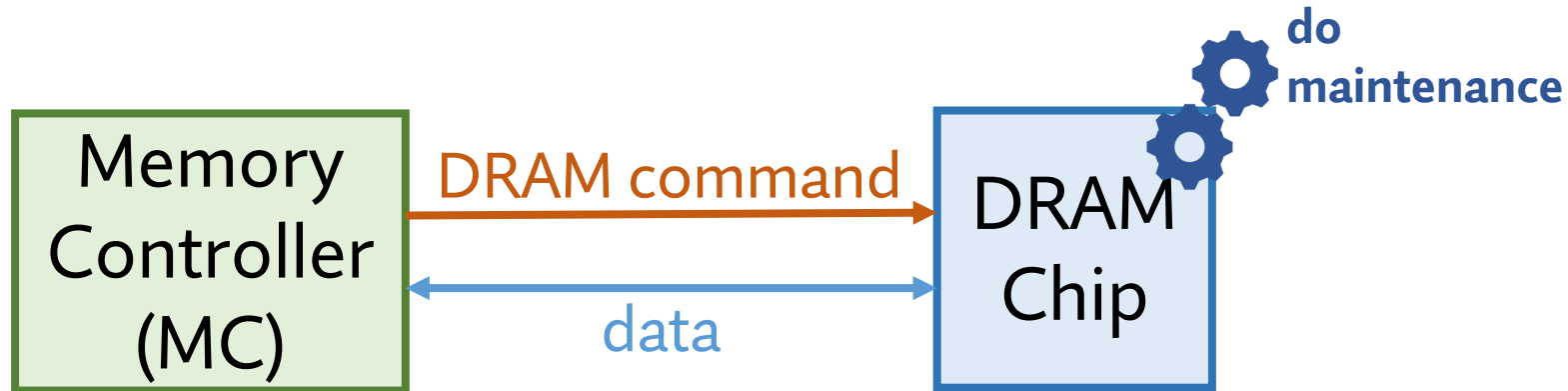
Haocong Luo

Onur Mutlu

ETH Zürich

SMD Key Idea: Autonomous Maintenance

DRAM chip controls in-DRAM maintenance operations

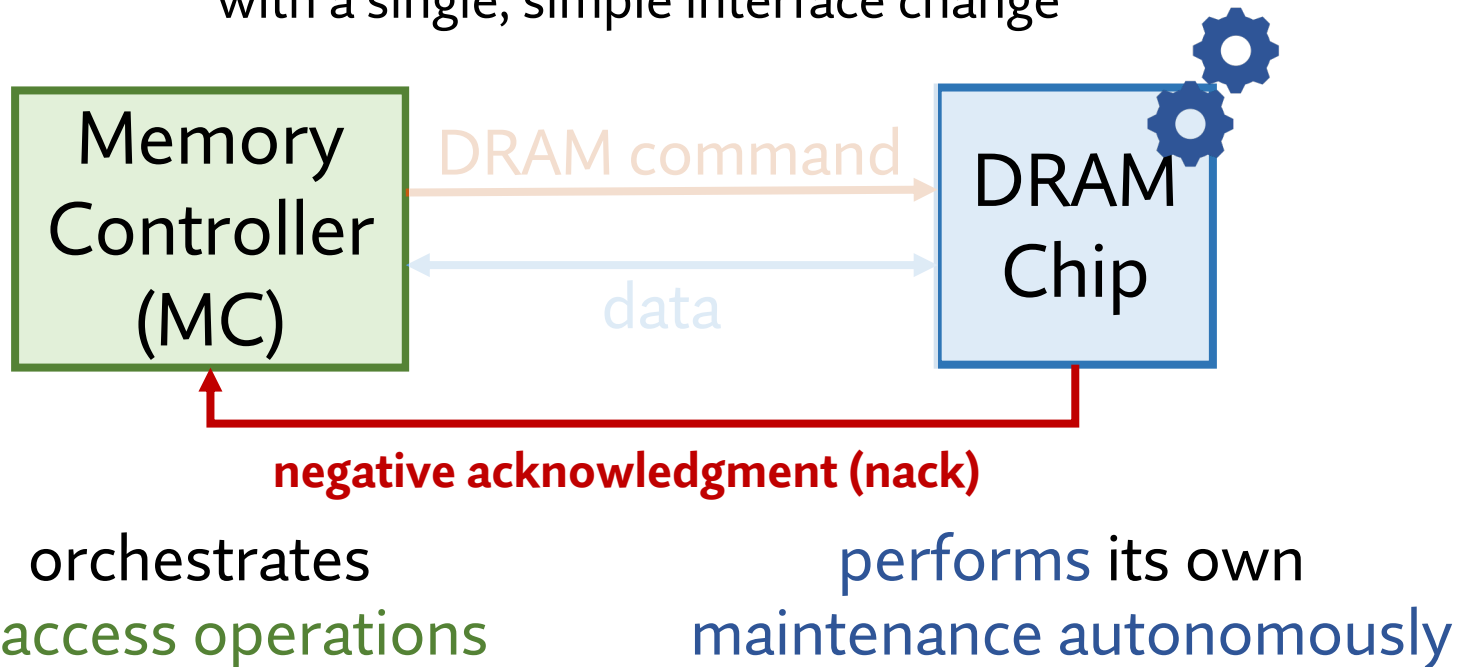


Enable implementing **new maintenance mechanisms** **without** modifying the standard and exposing **DRAM-internal proprietary** information

SMD Key Contribution

DRAM chip controls in-DRAM maintenance operations

with a single, simple interface change



Partition the work nicely between the memory controller and the DRAM chip

SMD-Based Maintenance Mechanisms

DRAM Refresh

Fixed Rate (SMD-FR)

uniformly refreshes **all** DRAM rows with a **fixed** refresh period

Variable Rate (SMD-VR)

skips refreshing rows that can **retain their data for longer** than the default refresh period

RowHammer Protection

Probabilistic (SMD-PRP)

Performs **neighbor** row refresh with a **small probability** on every row activation

Deterministic (SMD-DRP)

keeps track of most **frequently activated** rows and performs **neighbor** row refresh when activation count threshold is exceeded

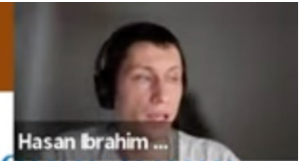
Memory Scrubbing

Periodic Scrubbing (SMD-MS)

periodically **scans** the **entire** DRAM for errors and corrects them

Talk on Self-Managing DRAM

Problem: The Rigid DRAM Interface



The **Memory Controller** manages DRAM maintenance operations



Changes to maintenance operations are often reflected to the memory controller design, DRAM interface, and other system components



Implementing new maintenance operations
(or modifying the existing ones) is difficult-to-realize



SAFARI Live Seminars 2022

SAFARI Live Seminar - Improving DRAM Performance, Reliability, and Security by Understanding DRAM

1,039 views • Streamed live on Sep 15, 2022

37 DISLIKE SHARE DOWNLOAD CLIP SAVE ...



Onur Mutlu Lectures
27.6K subscribers

ANALYTICS EDIT VIDEO

Self-Managing DRAM

- Hasan Hassan, Ataberk Olgun, A. Giray Yaglikci, Haocong Luo, and Onur Mutlu,
"Self-Managing DRAM: A Low-Cost Framework for Enabling Autonomous and Efficient DRAM Maintenance Operations"
Proceedings of the 57th International Symposium on Microarchitecture (MICRO), Austin, TX, USA, November 2024.
[[Slides \(pptx\)](#)] [[pdf](#)]
[[SelfManagingDRAM Source Code](#)]

Self-Managing DRAM: A Low-Cost Framework for Enabling Autonomous and Efficient DRAM Maintenance Operations

Hasan Hassan[†]

Ataberk Olgun[†]

A. Giray Yağlıkçı

Haocong Luo

Onur Mutlu

ETH Zürich

Adoption Issues

Adoption: How to Ease Programmability? (I)

- Kevin Hsieh, Eiman Ebrahimi, Gwangsun Kim, Niladrish Chatterjee, Mike O'Connor, Nandita Vijaykumar, Onur Mutlu, and Stephen W. Keckler, ["Transparent Offloading and Mapping \(TOM\): Enabling Programmer-Transparent Near-Data Processing in GPU Systems"](#)

Proceedings of the [43rd International Symposium on Computer Architecture \(ISCA\)](#), Seoul, South Korea, June 2016.

[\[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

Adoption: How to Ease Programmability? (II)

- Geraldo F. Oliveira, Alain Kohli, David Novo, Juan Gómez-Luna, Onur Mutlu,
“DaPPA: A Data-Parallel Framework for Processing-in-Memory Architectures,”
in *PACT SRC Student Competition*, Vienna, Austria, October 2023.

DaPPA: A Data-Parallel Framework for Processing-in-Memory Architectures

Geraldo F. Oliveira*

Alain Kohli*

David Novo‡

Juan Gómez-Luna*

Onur Mutlu*

**ETH Zürich*

‡*LIRMM, Univ. Montpellier, CNRS*

Adoption: How to Ease Programmability? (III)

- Jinfan Chen, Juan Gómez-Luna, Izzat El Hajj, YuXin Guo, and Onur Mutlu,
"SimplePIM: A Software Framework for Productive and Efficient Processing in Memory"
Proceedings of the 32nd International Conference on Parallel Architectures and Compilation Techniques (PACT), Vienna, Austria, October 2023.

SimplePIM: A Software Framework for Productive and Efficient Processing-in-Memory

Jinfan Chen¹ Juan Gómez-Luna¹ Izzat El Hajj² Yuxin Guo¹ Onur Mutlu¹
¹ETH Zürich ²American University of Beirut

Adoption: How to Ease Programmability? (IV)

- 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"
IEEE Access, 8 September 2021.
Preprint in arXiv, 8 May 2021.
[[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

Adoption: How to Ease Programmability? (V)

■ Appears in IEEE TETC 2023

ALP: Alleviating CPU-Memory Data Movement Overheads in Memory-Centric Systems

Nika Mansouri Ghiasi, Nandita Vijaykumar, Geraldo F. Oliveira, Lois Orosa, Ivan Fernandez, Mohammad Sadrosadati, Konstantinos Kanellopoulos, Nastaran Hajinazar, Juan Gómez Luna, Onur Mutlu

Abstract—Recent advances in memory technology have enabled near-data processing (NDP) to tackle main memory bottlenecks in modern systems. Prior works partition applications into segments (e.g., instructions, loops, functions) and execute memory-bound segments of the applications on NDP computation units, while mapping the cache-friendly application segments to host CPU cores that access a deeper cache hierarchy. Partitioning applications between NDP and host cores causes inter-segment data movement overhead, which is the overhead from moving data generated from one segment and used in the consecutive segments. This overhead can be large if the segments map to cores in different parts of the system (i.e., host and NDP). Prior works take two approaches to the inter-segment data movement overhead when partitioning applications between NDP and host cores. The first class of works maps segments to NDP or host cores based on the properties of each segment, neglecting the performance impact of the inter-segment data movement. Such partitioning techniques suffer from inter-segment data movement overhead. The second class of works maps segments to host or NDP cores based on the overall memory bandwidth savings of each segment (which depends on the memory bandwidth savings within each segment and the inter-segment data movement overhead between other segments). These works do not offload each segment to the best-fitting core if they incur high inter-segment data movement overhead. Therefore these works miss some of the potential NDP performance benefits. We show that mapping each segment (here basic block) to its best-fitting core based on the properties of each segment, assuming no inter-segment data movement, can provide substantial performance benefits. However, we show that the inter-segment data movement reduces this benefit significantly.

To this end, we introduce ALP, a new programmer-transparent technique to leverage the performance benefits of NDP by *alleviating* the performance impact of inter-segment data movement between host and memory and enabling efficient partitioning of applications between host and NDP cores. ALP alleviates the inter-segment data movement overhead by *proactively and accurately* transferring the required data between the segments mapped on host and NDP cores. This is based on the key observation that the instructions that generate the inter-segment data stay the same across different executions of a program on different input sets. ALP uses a compiler pass to identify these instructions and uses specialized hardware support to transfer data between the host and NDP cores at runtime. Using both the compiler and runtime information, ALP efficiently maps application segments to either host or NDP cores considering 1) the properties of each segment, 2) the inter-segment data movement overhead between different segments, and 3) whether this inter-segment data movement overhead can be alleviated proactively and in a timely manner. We evaluate ALP across a wide range of workloads and show on average 54.3% and 45.4% speedup compared to executing the application only on the host CPU or only the NDP cores, respectively.

Adoption: How to Maintain Coherence? (I)

- Amirali Boroumand, Saugata Ghose, Minesh Patel, Hasan Hassan, Brandon Lucia, Kevin Hsieh, Krishna T. Malladi, Hongzhong Zheng, and Onur Mutlu,
"LazyPIM: An Efficient Cache Coherence Mechanism for Processing-in-Memory"
IEEE Computer Architecture Letters (CAL), June 2016.

LazyPIM: An Efficient Cache Coherence Mechanism for Processing-in-Memory

Amirali Boroumand[†], Saugata Ghose[†], Minesh Patel[†], Hasan Hassan^{†§}, Brandon Lucia[†],
Kevin Hsieh[†], Krishna T. Malladi^{*}, Hongzhong Zheng^{*}, and Onur Mutlu^{††}

[†]Carnegie Mellon University ^{*}Samsung Semiconductor, Inc. [§]TOBB ETÜ [‡]ETH Zürich

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"

Proceedings of the 46th International Symposium on Computer Architecture (ISCA), Phoenix, AZ, USA, June 2019.

CoNDA: Efficient Cache Coherence Support for Near-Data Accelerators

Amirali Boroumand[†]

Saugata Ghose[†]

Minesh Patel[★]

Hasan Hassan[★]

Brandon Lucia[†]

Rachata Ausavarungnirun^{†‡}

Kevin Hsieh[†]

Nastaran Hajinazar^{◇†}

Krishna T. Malladi[§]

Hongzhong Zheng[§]

Onur Mutlu^{★†}

[†]Carnegie Mellon University

[★]ETH Zürich

[‡]KMUTNB

[◇]Simon Fraser University

[§]Samsung Semiconductor, Inc.

Adoption: How to Support Synchronization?

- Christina Giannoula, Nandita Vijaykumar, Nikela Papadopoulou, Vasileios Karakostas, Ivan Fernandez, Juan Gómez-Luna, Lois Orosa, Nectarios Koziris, Georgios Goumas, Onur Mutlu, **"SynCron: Efficient Synchronization Support for Near-Data-Processing Architectures"**
Proceedings of the 27th International Symposium on High-Performance Computer Architecture (HPCA), Virtual, February-March 2021.
[[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*

Adoption: Evaluation Infrastructures (I)

- 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"
IEEE Access, 8 September 2021.
Preprint in arXiv, 8 May 2021.
[[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

Adoption: Evaluation Infrastructures (II)

- Ataberk Olgun, Juan Gomez Luna, Konstantinos Kanellopoulos, Behzad Salami, Hasan Hassan, Oguz Ergin, and Onur Mutlu,
["PiDRAM: A Holistic End-to-end FPGA-based Framework for Processing-in-DRAM"](#)
[ACM Transactions on Architecture and Code Optimization \(TACO\)](#), March 2023.
[\[arXiv version\]](#)
Presented at the [18th HiPEAC Conference](#), Toulouse, France, January 2023.
[\[Slides \(pptx\) \(pdf\)\]](#)
[\[Longer Lecture Slides \(pptx\) \(pdf\)\]](#)
[\[Lecture Video \(40 minutes\)\]](#)
[\[PiDRAM Source Code\]](#)

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 University of Economics and Technology

Adoption: Evaluation Infrastructures (III)

- Haocong Luo, Yahya Can Tugrul, F. Nisa Bostanci, Ataberk Olgun, A. Giray Yaglikci, and Onur Mutlu,
"Ramulator 2.0: A Modern, Modular, and Extensible DRAM Simulator"
*Preprint on **arxiv**, August 2023.*
[[arXiv version](#)]
[[Ramulator 2.0 Source Code](#)]

Ramulator 2.0: A Modern, Modular, and Extensible DRAM Simulator

Haocong Luo, Yahya Can Tuğrul, F. Nisa Bostancı, Ataberk Olgun, A. Giray Yağlıkçı, and Onur Mutlu

<https://arxiv.org/pdf/2308.11030.pdf>

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/>

SAFARI Newsletter June 2023 Edition

- <https://safari.ethz.ch/safari-newsletter-june-2023/>

SAFARI
SAFARI Research Group

Think Big, Aim High



ETH zürich

View in your browser

June 2023



Recall: DRAM Testing Infrastructure



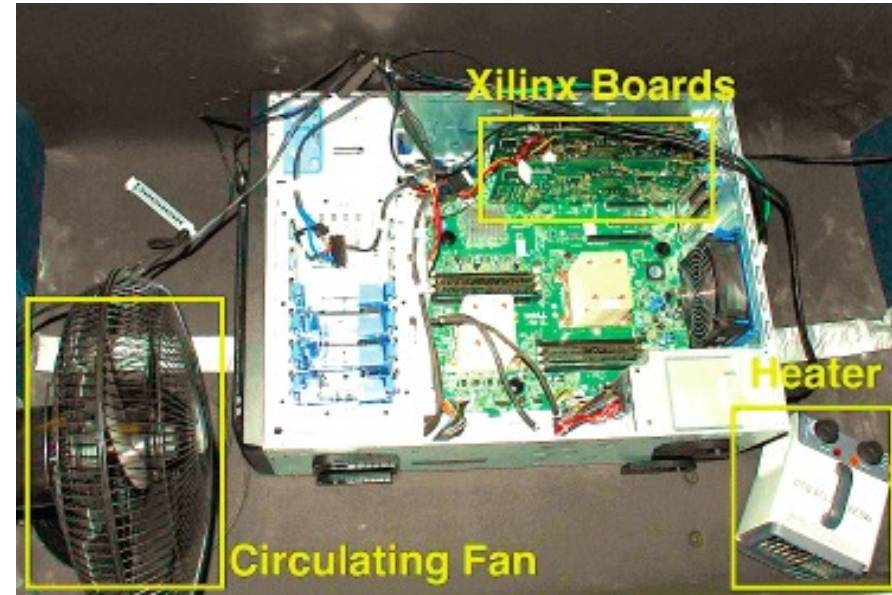
An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms (Liu et al., ISCA 2013)

The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study (Khan et al., SIGMETRICS 2014)

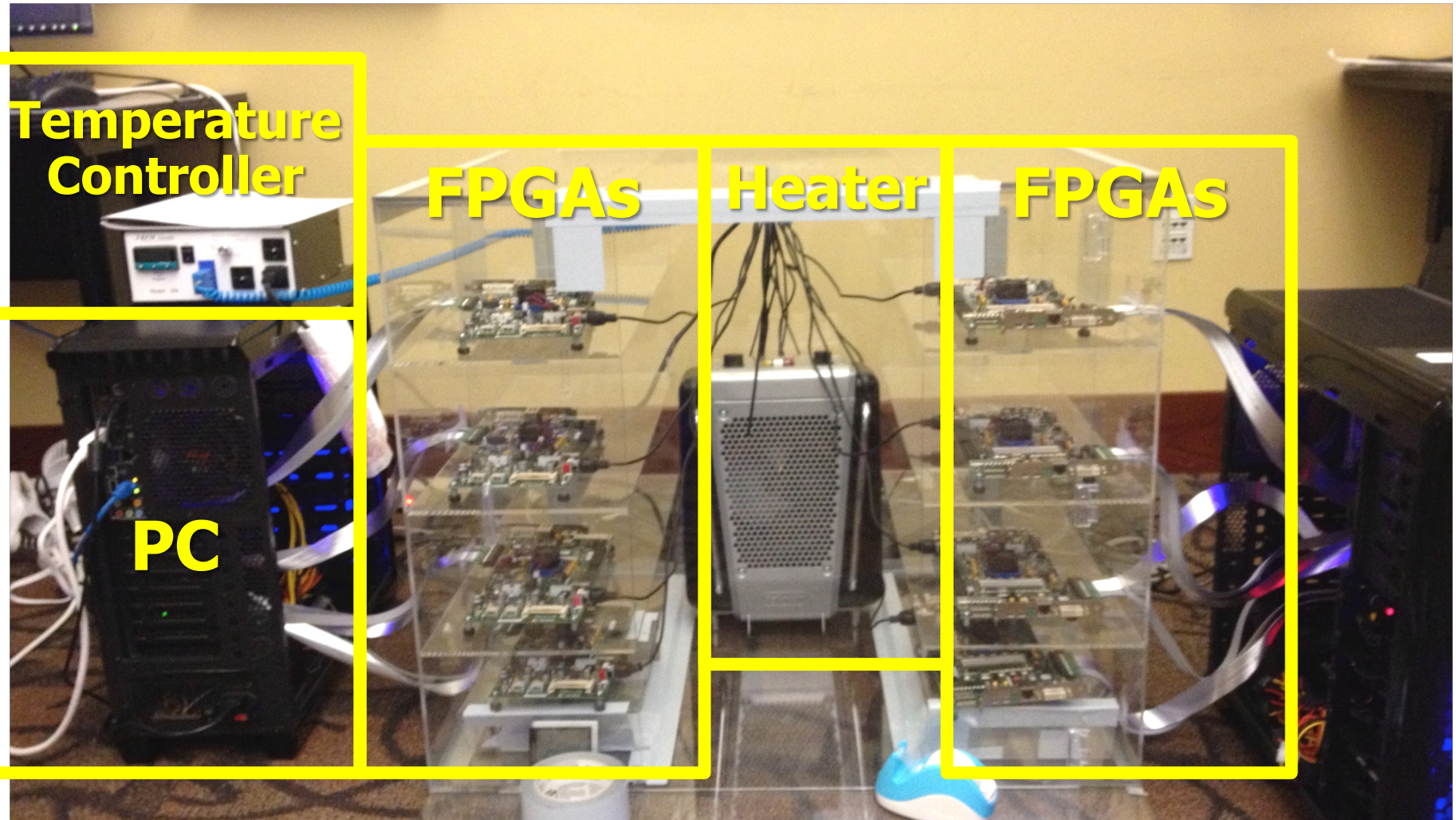
Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors (Kim et al., ISCA 2014)

Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case (Lee et al., HPCA 2015)

AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems (Qureshi et al., DSN 2015)

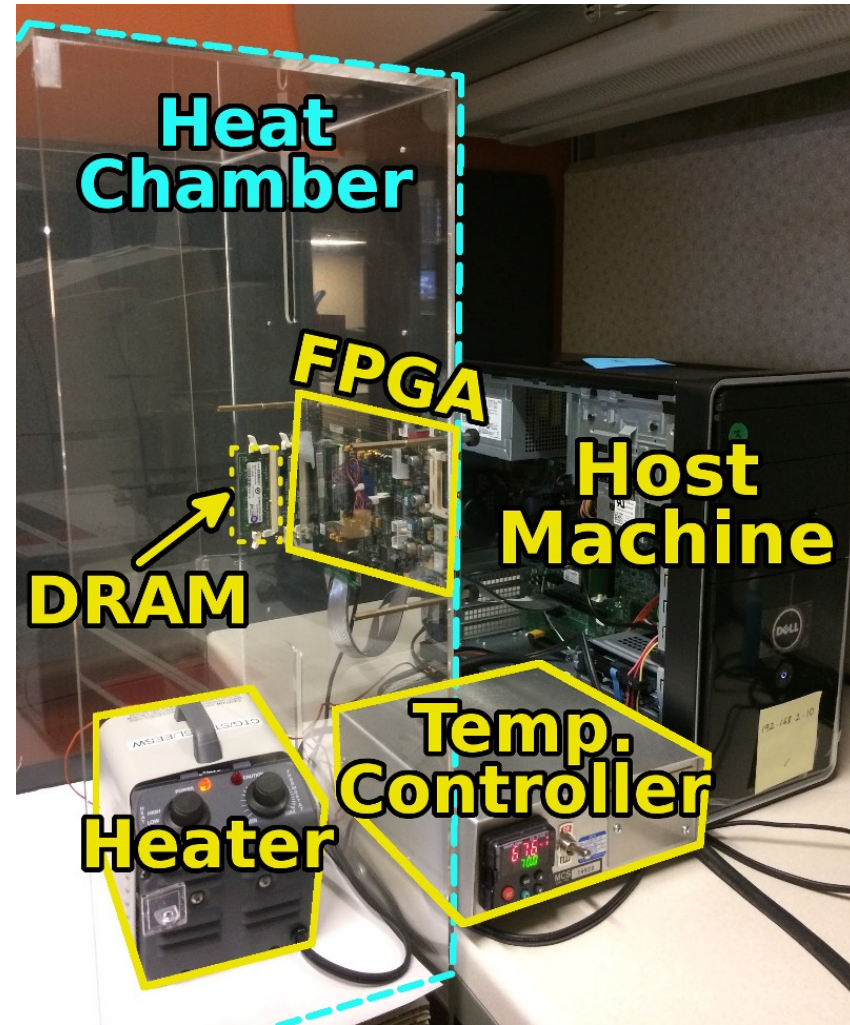


Recall: DRAM Testing Infrastructure



SoftMC: Open Source DRAM Infrastructure

- Hasan Hassan et al., “[SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies](#),” HPCA 2017.
- Flexible
- Easy to Use (C++ API)
- Open-source
github.com/CMU-SAFARI/SoftMC



SoftMC: Open Source DRAM Infrastructure

- Hasan Hassan, Nandita Vijaykumar, Samira Khan, Saugata Ghose, Kevin Chang, Gennady Pekhimenko, Donghyuk Lee, Oguz Ergin, and Onur Mutlu,

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

Proceedings of the 23rd International Symposium on High-Performance Computer Architecture (HPCA), Austin, TX, USA, February 2017.

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

[Full Talk Lecture (39 minutes)]

[Source Code]

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

Hasan Hassan^{1,2,3} Nandita Vijaykumar³ Samira Khan^{4,3} Saugata Ghose³ Kevin Chang³
Gennady Pekhimenko^{5,3} Donghyuk Lee^{6,3} Oguz Ergin² Onur Mutlu^{1,3}

¹ETH Zürich ²TOBB University of Economics & Technology ³Carnegie Mellon University
⁴University of Virginia ⁵Microsoft Research ⁶NVIDIA Research

DRAM Bender

- Ataberk Olgun, Hasan Hassan, A Giray Yağlıkçı, Yahya Can Tuğrul, Lois Orosa, Haocong Luo, Minesh Patel, Oğuz Ergin, and Onur Mutlu,
"DRAM Bender: An Extensible and Versatile FPGA-based Infrastructure to Easily Test State-of-the-art DRAM Chips"
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), 2023.
[[Extended arXiv version](#)]
[[DRAM Bender Source Code](#)]
[[DRAM Bender Tutorial Video](#) (43 minutes)]

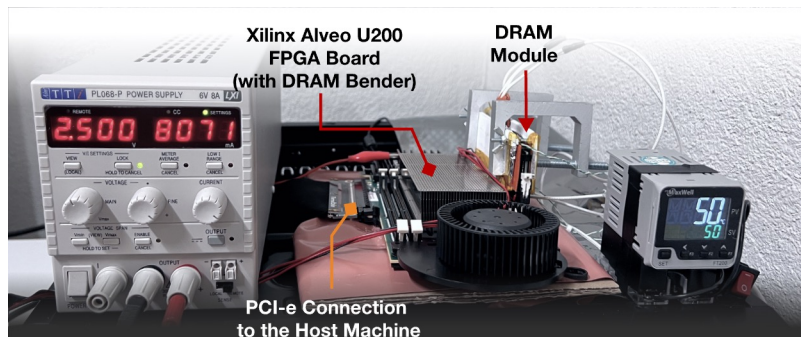
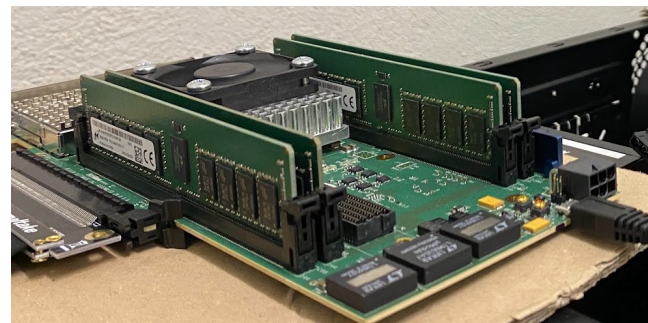
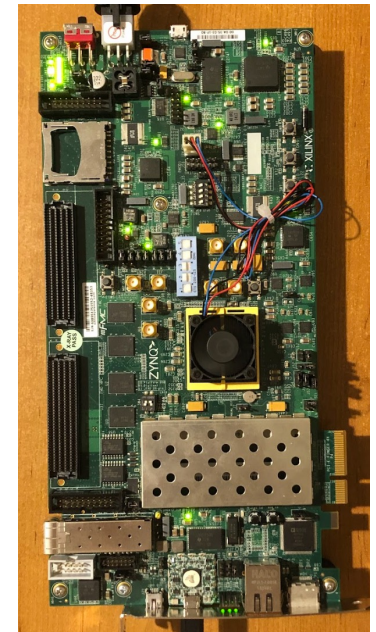
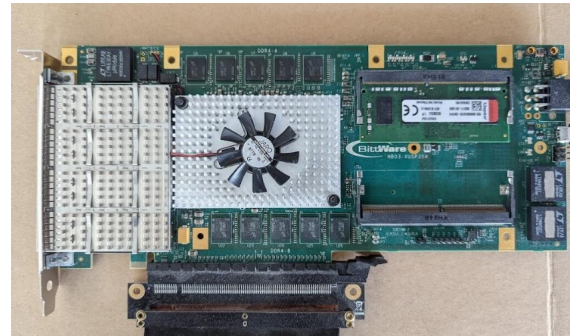
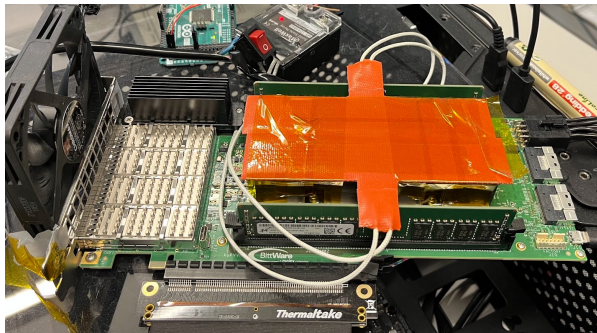
DRAM Bender: An Extensible and Versatile FPGA-based Infrastructure to Easily Test State-of-the-art DRAM Chips

Ataberk Olgun[§] Hasan Hassan[§] A. Giray Yağlıkçı[§] Yahya Can Tuğrul^{§†}
Lois Orosa^{§⊙} Haocong Luo[§] Minesh Patel[§] Oğuz Ergin[†] Onur Mutlu[§]
 [§]*ETH Zürich* [†]*TOBB ETÜ* [⊙]*Galician Supercomputing Center*

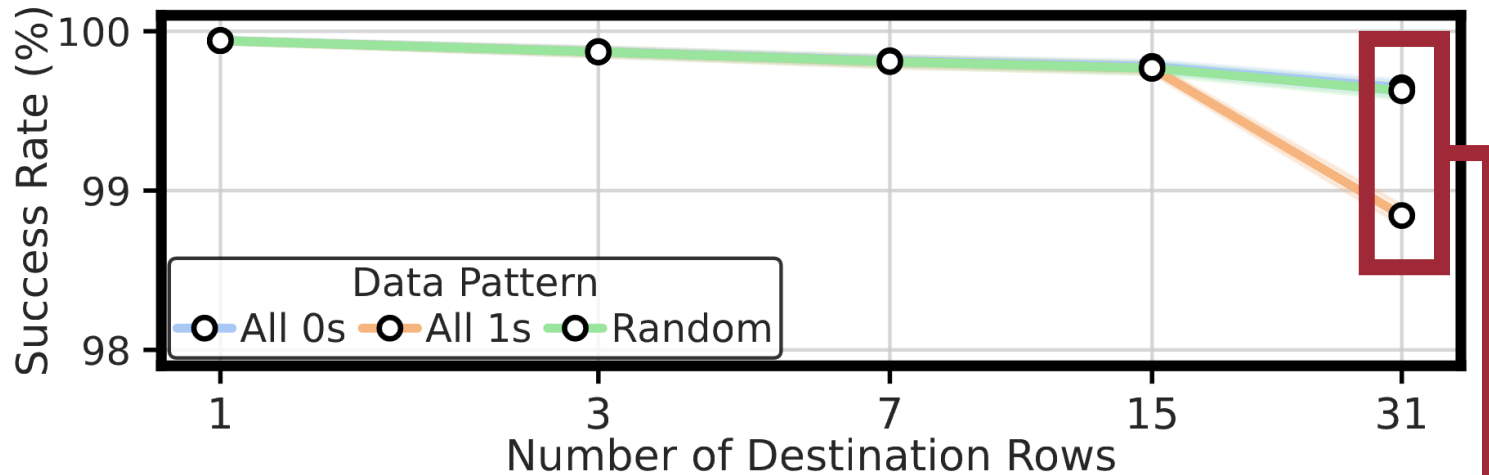
DRAM Bender: Prototypes

Testing Infrastructure	Protocol Support	FPGA Support
SoftMC [134]	DDR3	One Prototype
LiteX RowHammer Tester (LRT) [17]	DDR3/4, LPDDR4	Two Prototypes
DRAM Bender (this work)	DDR3/DDR4	Five Prototypes

Five out of the box FPGA-based prototypes



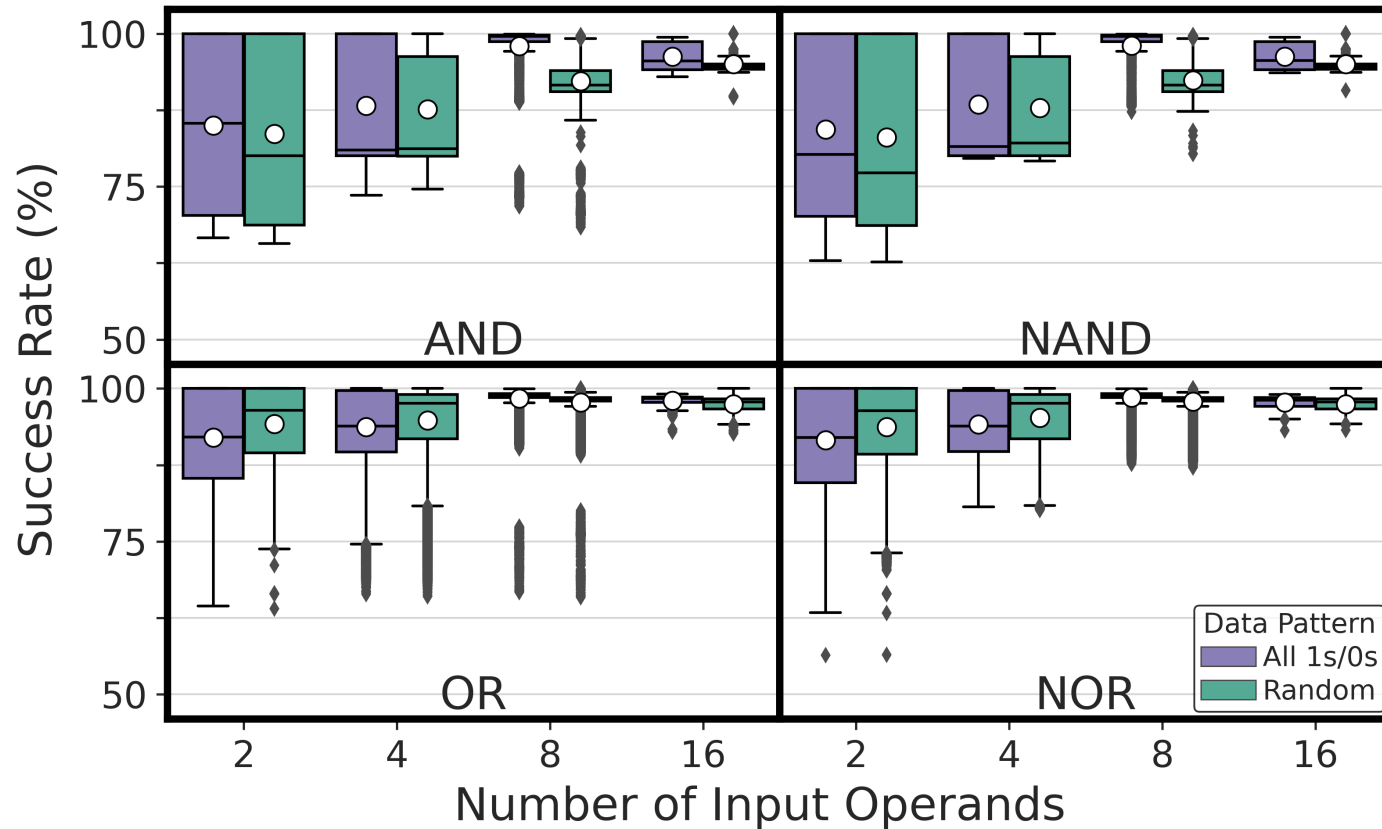
Impact of Data Pattern



**At most 0.79% decrease in
average success rate**

**Data pattern has a small effect
on the success rate of the Multi-RowCopy operation**

Impact of Data Pattern



Data pattern slightly affects the reliability of AND, NAND, OR, and NOR operations

What About Other Types of Memories?

In-Flash Bulk Bitwise Execution

- Jisung Park, Roknoddin Azizi, Geraldo F. Oliveira, Mohammad Sadrosadati, Rakesh Nadig, David Novo, Juan Gómez-Luna, Myungsuk Kim, and Onur Mutlu, **"Flash-Cosmos: In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory"**
Proceedings of the 55th International Symposium on Microarchitecture (MICRO), Chicago, IL, USA, October 2022.
[[Slides \(pptx\)](#)] [[pdf](#)]
[[Longer Lecture Slides \(pptx\)](#)] [[pdf](#)]
[[Lecture Video](#) (44 minutes)]
[[arXiv version](#)]

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

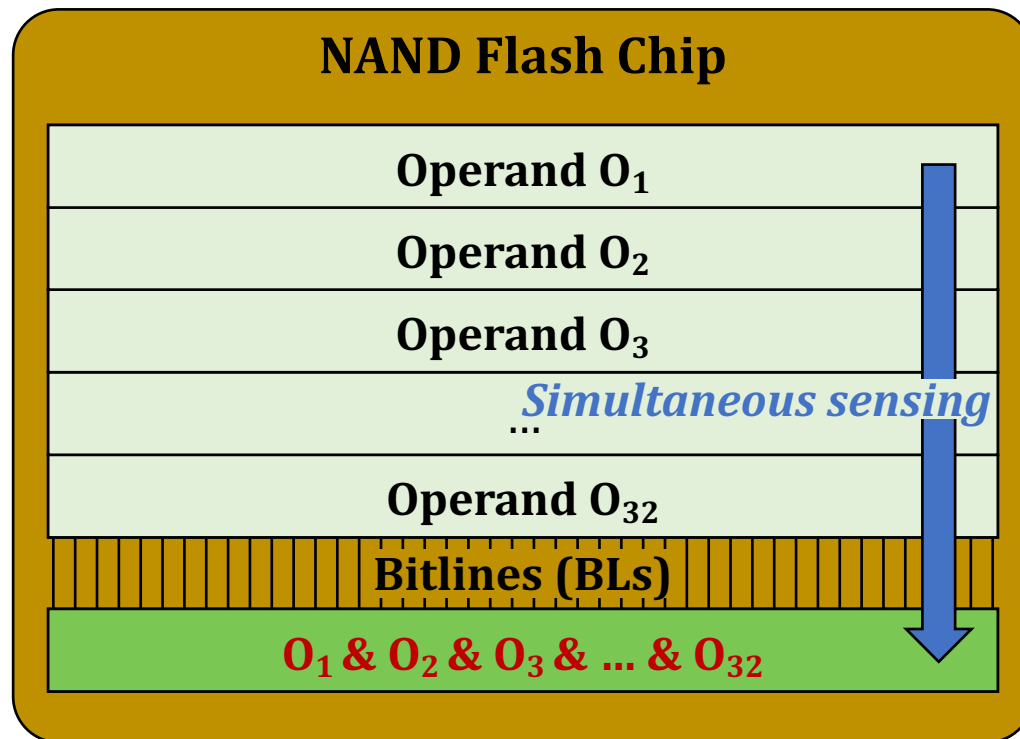
Jisung Park^{§∇} Roknoddin Azizi[§] Geraldo F. Oliveira[§] Mohammad Sadrosadati[§]
Rakesh Nadig[§] David Novo[†] Juan Gómez-Luna[§] Myungsuk Kim[‡] Onur Mutlu[§]

[§]ETH Zürich [∇]POSTECH [†]LIRMM, Univ. Montpellier, CNRS [‡]Kyungpook National University

Flash-Cosmos: Basic Ideas

▪ Flash-Cosmos enables

- Computation on multiple operands with a single sensing operation
- Accurate computation results by eliminating raw bit errors in stored data



Multi-Wordline Sensing (MWS): Bitwise AND

■ Intra-Block MWS:

Simultaneously activates multiple WLs in the same block

→ Bitwise AND of the stored data in the WLs

A bitline reads as '**1**' only when all the target cells store '**1**'
→ Equivalent to the bitwise AND of all the target cells

*Operate
as a resistance (1)
or an open switch (0)*

WL₂
WL₃
WL₄

Result: 0

0

0

0

BL₁

BL₂

BL₃

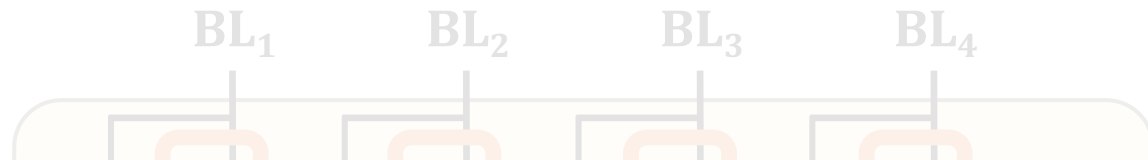
BL₄

Multi-Wordline Sensing (MWS): Bitwise AND

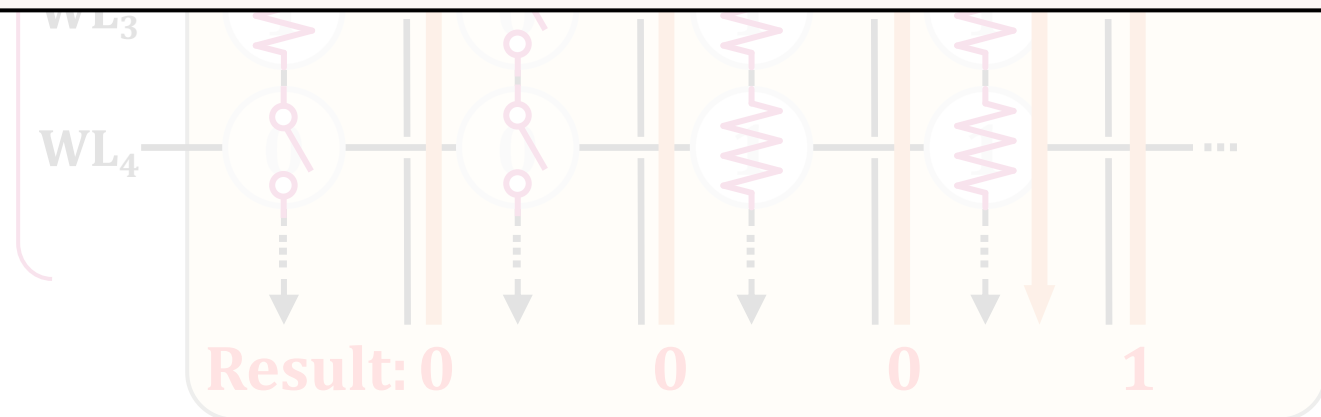
- Intra-Block MWS:

Simultaneously activates multiple WLs in the same block

→ Bitwise AND of the stored data in the WLs



Flash-Cosmos (Intra-Block MWS) enables bitwise AND of multiple pages in the same block via a single sensing operation



Other Types of Bitwise Operations

Flash-Cosmos also enables
other types of bitwise operations
(NOT/NAND/NOR/XOR/XNOR)
leveraging **existing features** of NAND flash memory

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*



<https://arxiv.org/abs/2209.05566.pdf>

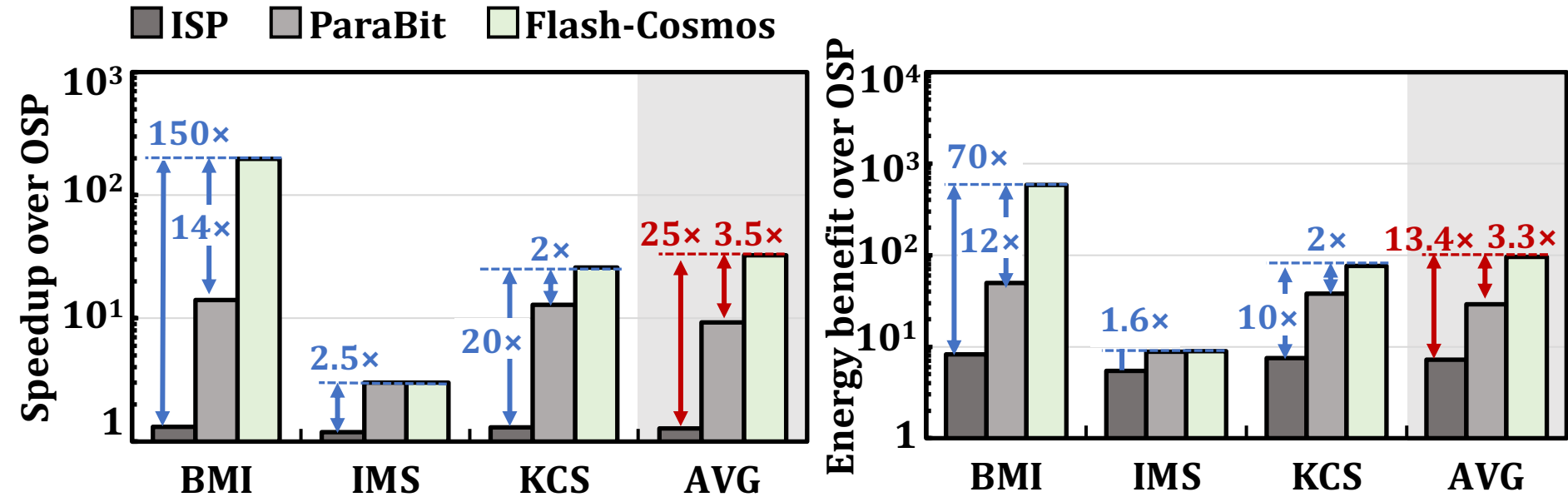
Results: Real-Device Characterization

No changes to the cell array
of commodity NAND flash chips

Can have many operands
(AND: up to 48, OR: up to 4)
with small increase in sensing latency ($< 10\%$)

ESP significantly improves
the reliability of computation results
(no observed bit error in the tested flash cells)

Results: Performance & Energy



Flash-Cosmos provides **significant performance & energy benefits** over all the baselines

The larger the number of operands,
the higher the performance & energy benefits

Flash-Cosmos: In-Flash Bulk Bitwise Execution

- Jisung Park, Roknoddin Azizi, Geraldo F. Oliveira, Mohammad Sadrosadati, Rakesh Nadig, David Novo, Juan Gómez-Luna, Myungsuk Kim, and Onur Mutlu,
"Flash-Cosmos: In-Flash Bulk Bitwise Operations Using Inherent Computation Capability of NAND Flash Memory"
Proceedings of the 55th International Symposium on Microarchitecture (MICRO),
Chicago, IL, USA, October 2022.
[[Slides \(pptx\)](#)] [[pdf](#)]
[[Longer Lecture Slides \(pptx\)](#)] [[pdf](#)]
[[Lecture Video](#) (44 minutes)]
[[arXiv version](#)]

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[§]

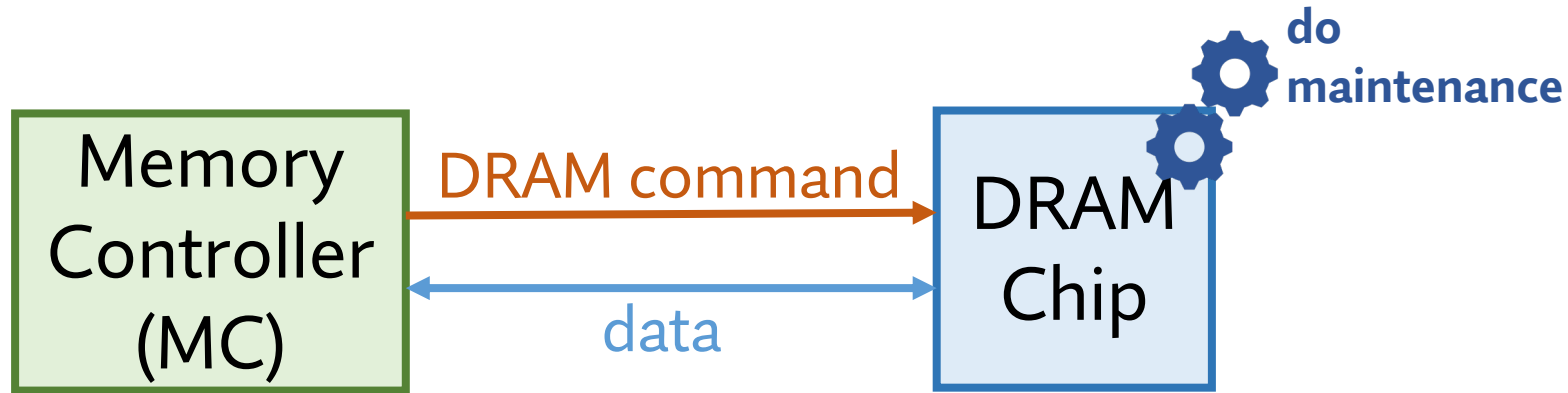
[§]ETH Zürich [∇]POSTECH [†]LIRMM, Univ. Montpellier, CNRS [‡]Kyungpook National University

Self-Managing DRAM

[MICRO 2024]

SMD Key Idea: Autonomous Maintenance

DRAM chip controls in-DRAM maintenance operations

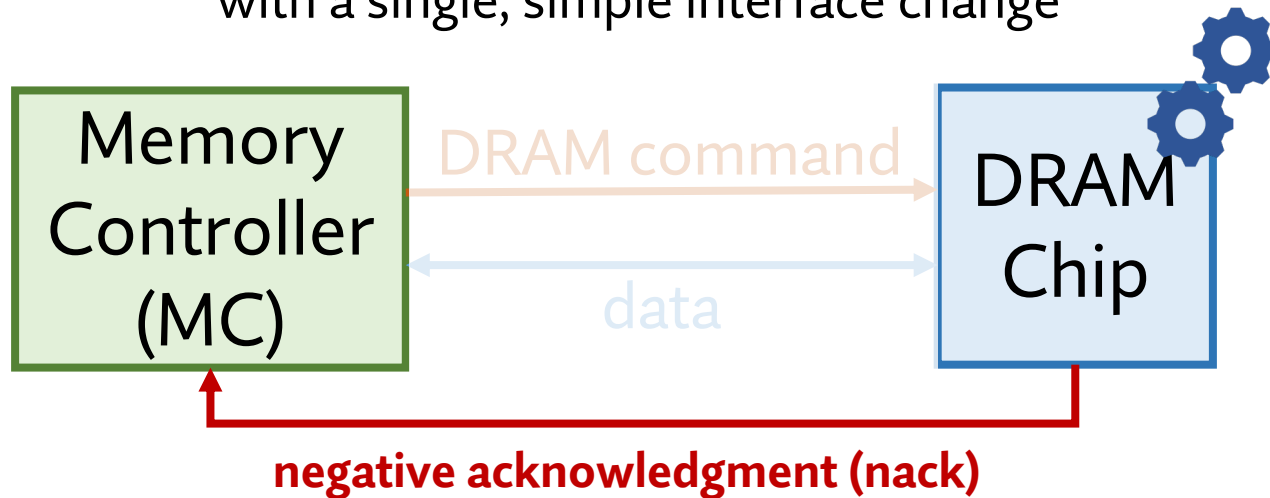


Enable implementing **new maintenance mechanisms** **without** modifying the standard and exposing **DRAM-internal proprietary** information

SMD Key Contribution

DRAM chip controls in-DRAM maintenance operations

with a single, simple interface change



orchestrates
all access operations

can now perform its own
maintenance autonomously

Partition the work nicely between the
memory controller and the DRAM chip

PAPI LLM Inference System

[ASPLOS 2025]

PAPI's Key Idea

Enable **online dynamic task scheduling** in a **heterogeneous PIM-enabled architecture** via online identification of kernel properties in LLM decoding

PAPI's Key Components

A new PIM-enabled computing system design

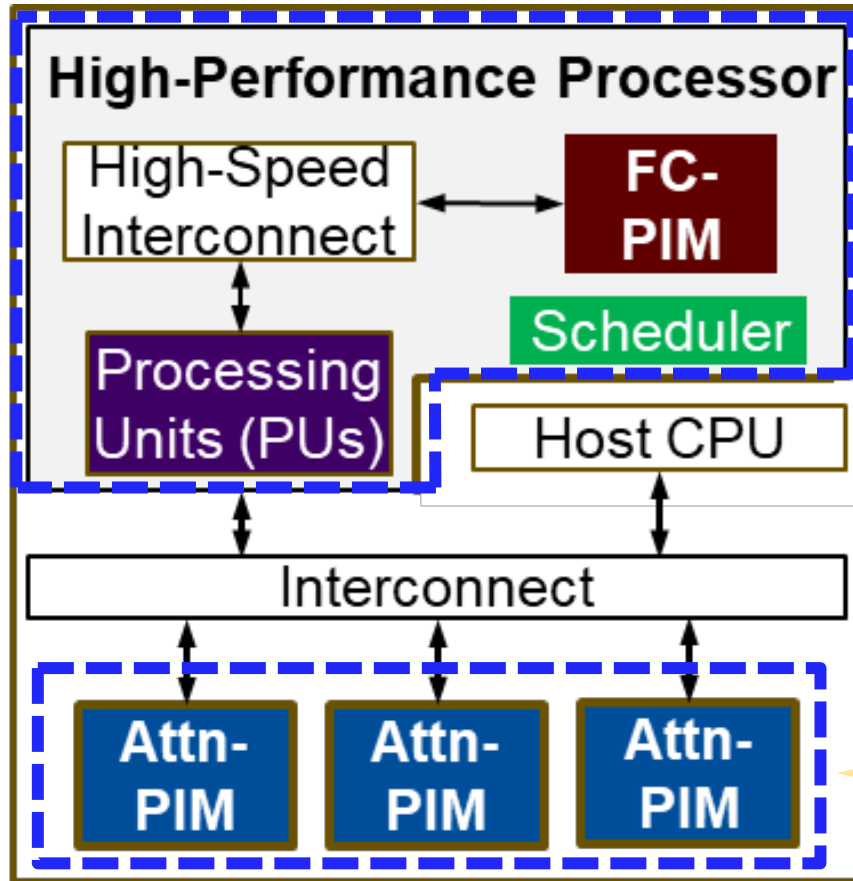
Hybrid PIM units

to cater to different parallelism levels of
FC and attention kernels

Dynamic LLM kernel scheduling

to cater to dynamically changing
parallelism levels

PAPI's Architecture

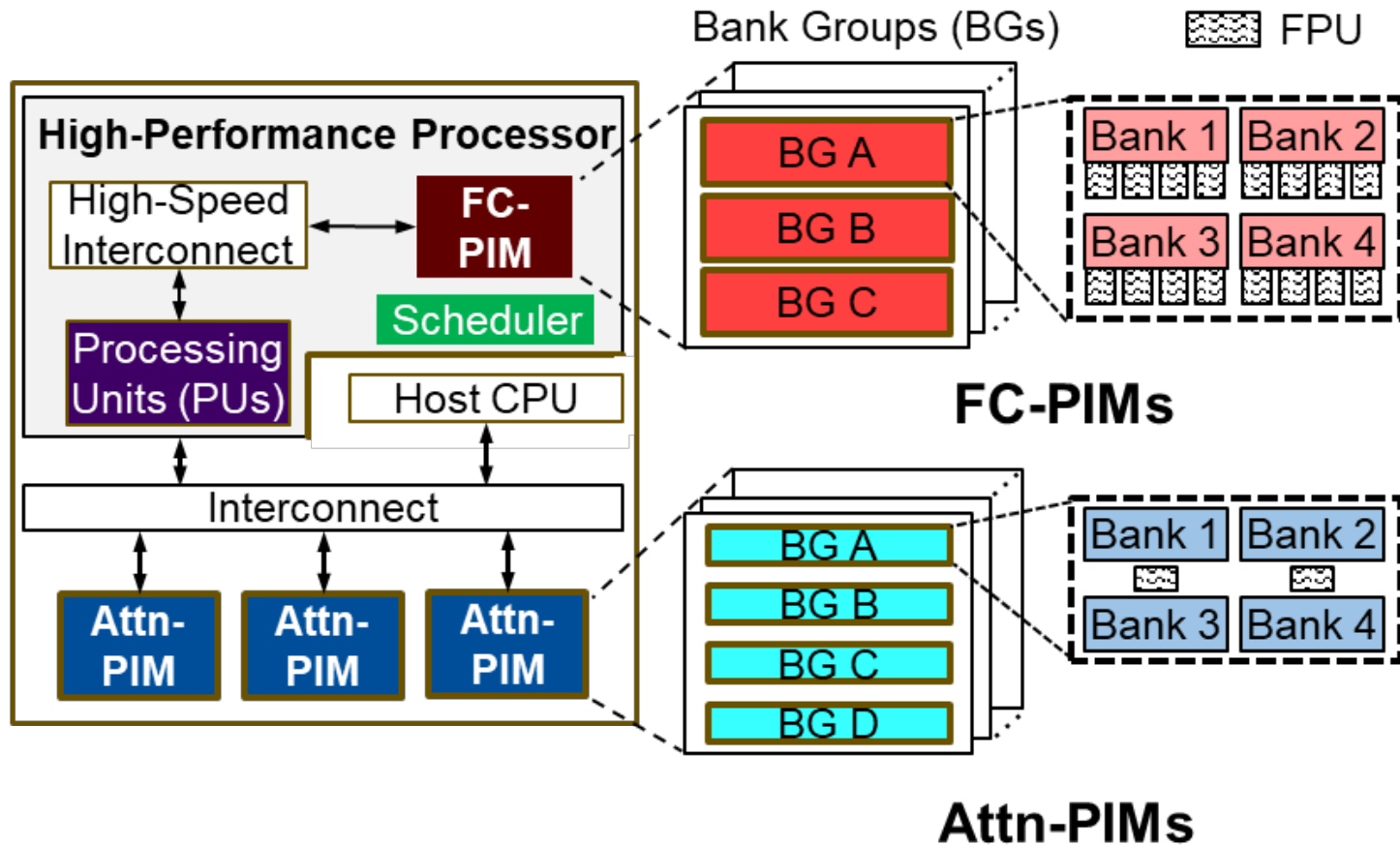


Handles memory-bound or compute-bound **FC kernels**

- Execution of FC kernels
- Dynamic scheduling

Handles memory-bound **attention kernels**

PAPI's Architecture



Hybrid PIM units handle memory-bound FC & attention kernels with **different computational and memory demands**

Outline

1 Background

2 Observations & Motivation

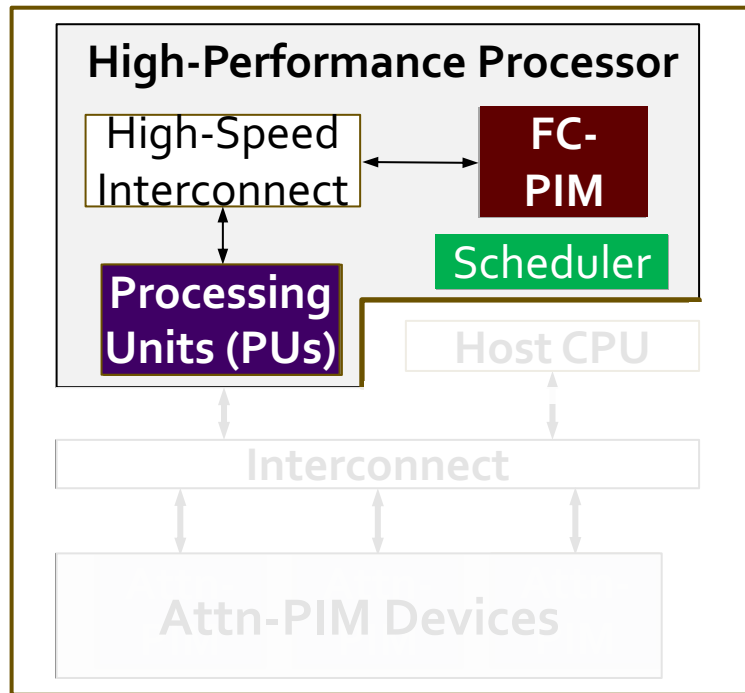
3 PAPI's Overview

4 PAPI's Implementation

5 Evaluation

6 Conclusion

High-Performance Processor



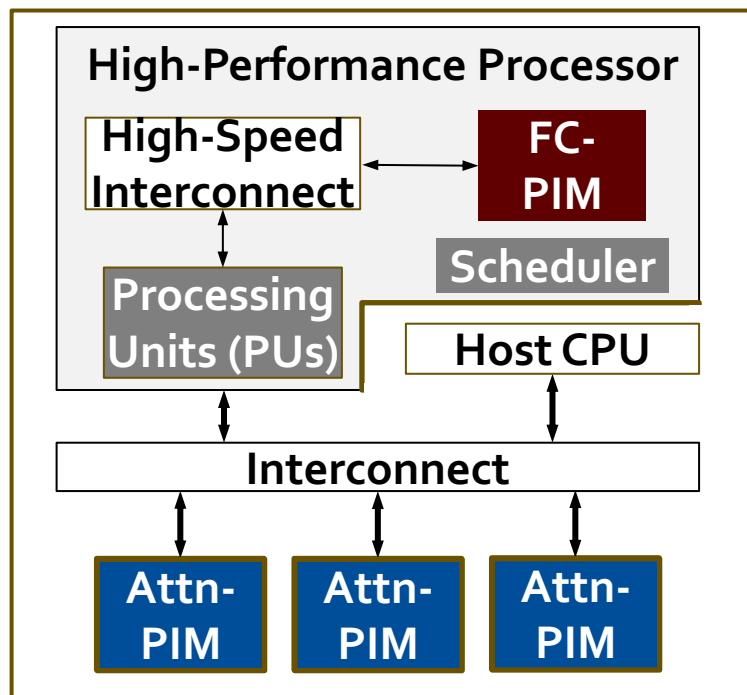
When FC kernels are compute-bound:

Assign FC kernels to PUs

When FC kernels are memory-bound:

Assign FC kernels to FC-PIM

Hybrid PIM Units (I)



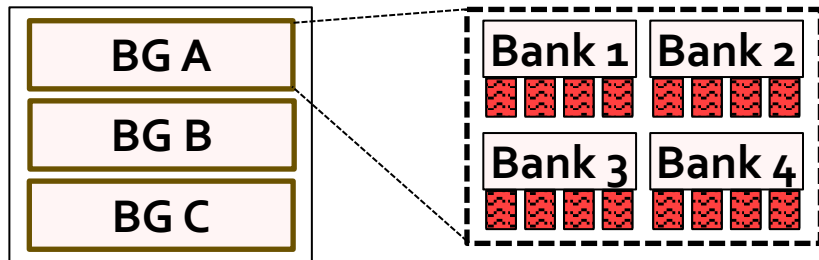
FC-PIM device **placed in**
the High-Performance Processor

Attn-PIM devices store KV cache;
separated from
the High-Performance Processor

Hybrid PIM Units (II)

 Floating-Point Processing Units (FPU)

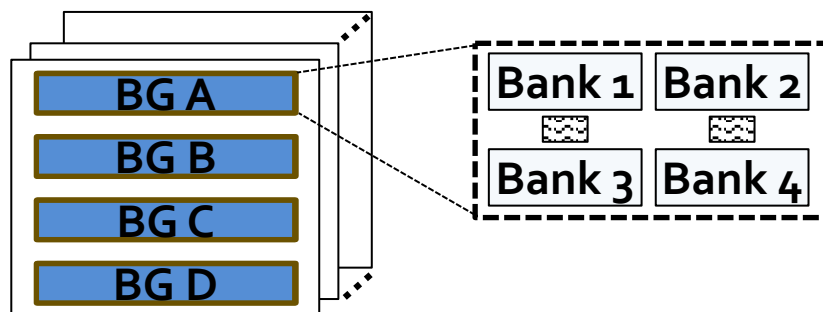
Bank Groups (BGs)



FC-PIM

More FPUs per Bank

Higher Computation Capability
to cater to FC kernels



Attn-PIMs

More Bank Groups per Stack
More Attn-PIM Devices

Higher Memory Capacity
to cater to attention kernels

PAPI Runtime Scheduler

Offline: identify memory-boundedness threshold

① Monitor Parallelism Levels

- RLP & TLP



② Arithmetic Intensity Predictor

- Estimate arithmetic intensity of FC kernels
- Compare with memory-boundedness threshold



③ Schedule the FC Kernels

- Map FC kernels to either FC-PIM or PUs

Evaluation Methodology

Performance and Energy Analysis:

- Simulation using AttAcc [ASPLOS'24] and Ramulator 2 [IEEE CAL'23]

Baselines:

- **AttAcc** [ASPLOS'24]
- **GPU+HBM-PIM** (NVIDIA A100 GPU + Samsung's HBM-PIM)
- **PIM-only** (PIM devices in AttAcc)

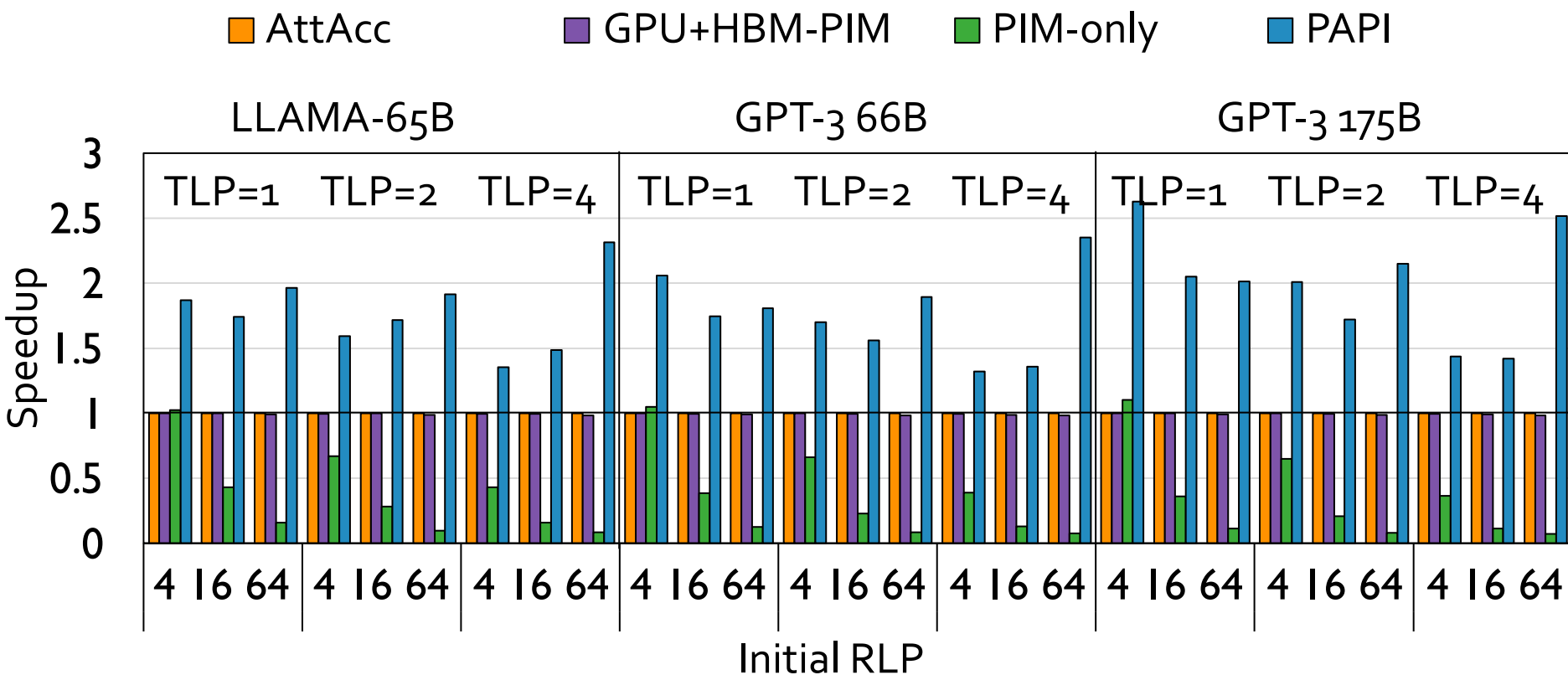
Workloads: **Three** transformer-based LLMs

- LLaMA-65B, GPT-3 66B, GPT-3 175B

Datasets: Dolly

- Creative-writing tasks
- General-QA tasks

Performance Analysis



PAPI improves **performance** by **1.8X**, **1.9X**, and **11.1X** compared to AttAcc, GPU+HBM-PIM, and PIM-only, respectively

Energy Analysis

AttAcc

GPU+HBM-PIM

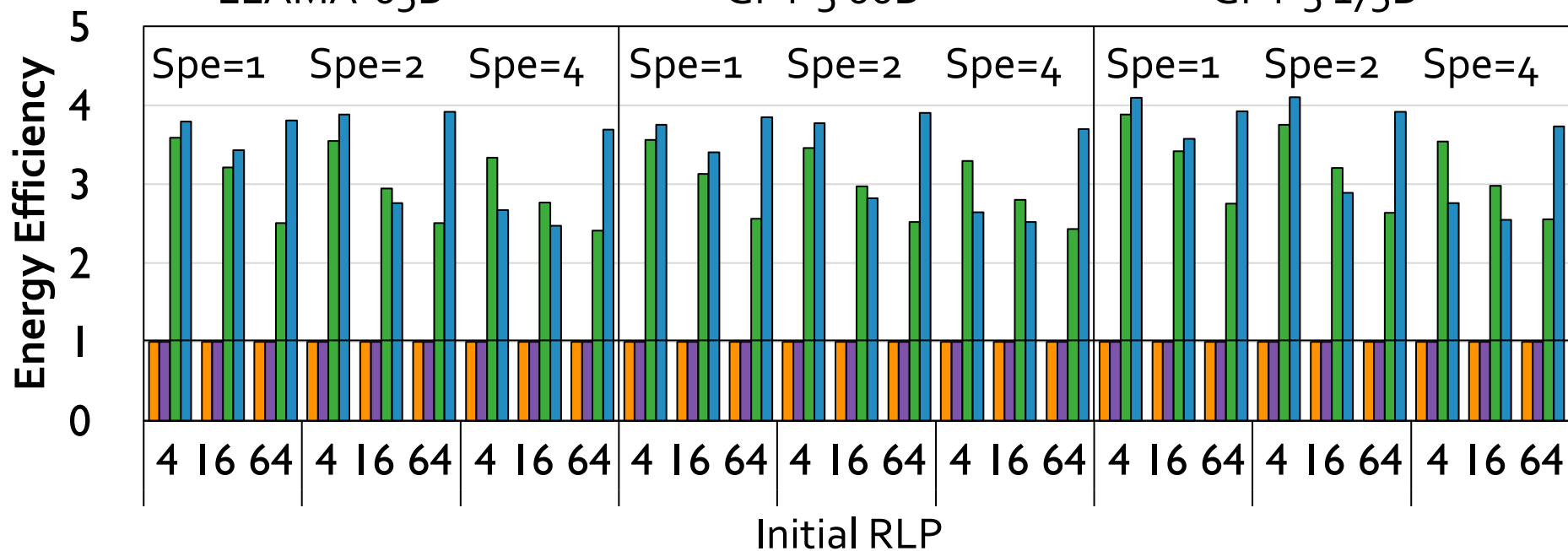
PIM-only

PAPI

LLAMA-65B

GPT-3 66B

GPT-3 175B



PAPI improves **energy efficiency** by **3.4X**, **3.4X**, and **1.2X** compared to AttAcc, GPU+HBM-PIM, and PIM-only, respectively

More in the Paper

- **Details on PAPI's implementation**
 - PAPI's heterogeneous architecture
 - PAPI's runtime scheduler
 - System integration
 - Data partitioning across PIM devices (both Attn-PIM & FC-PIM)
- **Detailed evaluation results**
 - PAPI's speedup across different RLP & TLP levels
 - Ablation study for PAPI's speedup
- **Area/power analysis**

More in the Paper

PAPI: Exploiting Dynamic Parallelism in Large Language Model Decoding with a Processing-In-Memory-Enabled Computing System

Yintao He^{1,2} Haiyu Mao^{3,4} Christina Giannoula^{5,6,4} Mohammad Sadrosadati⁴
Juan Gómez-Luna⁷ Huawei Li^{1,2} Xiaowei Li^{1,2} Ying Wang¹ Onur Mutlu⁴

¹SKLP, Institute of Computing Technology, CAS ²University of Chinese Academy of Sciences ³King's College London

⁴ETH Zürich ⁵University of Toronto ⁶Vector Institute ⁷NVIDIA

<https://arxiv.org/pdf/2502.15470>



Conclusion

Key Findings

- 1 LLM kernels have **different computation and memory bandwidth demands** across **different RLP & TLP levels**
- 2 **Memory-bound kernels** exhibit **different** computation demands depending on kernel type
- 3 LLM kernels have **dynamically changing** RLP and TLP levels

Conclusion

Key Contribution

PAPI

A new **PIM-enabled heterogeneous** system design that caters to **varying demands** of LLM kernels by scheduling them **dynamically** to computation-centric processing units and hybrid PIM units

Key Results

PAPI largely improves both performance and energy efficiency over best prior LLM decoding system

- **1.8×** speedup
- **3.4×** energy efficiency increase