

Digital Design & Computer Arch.

Lecture 3a: Mysteries in Comp. Arch.

Prof. Onur Mutlu

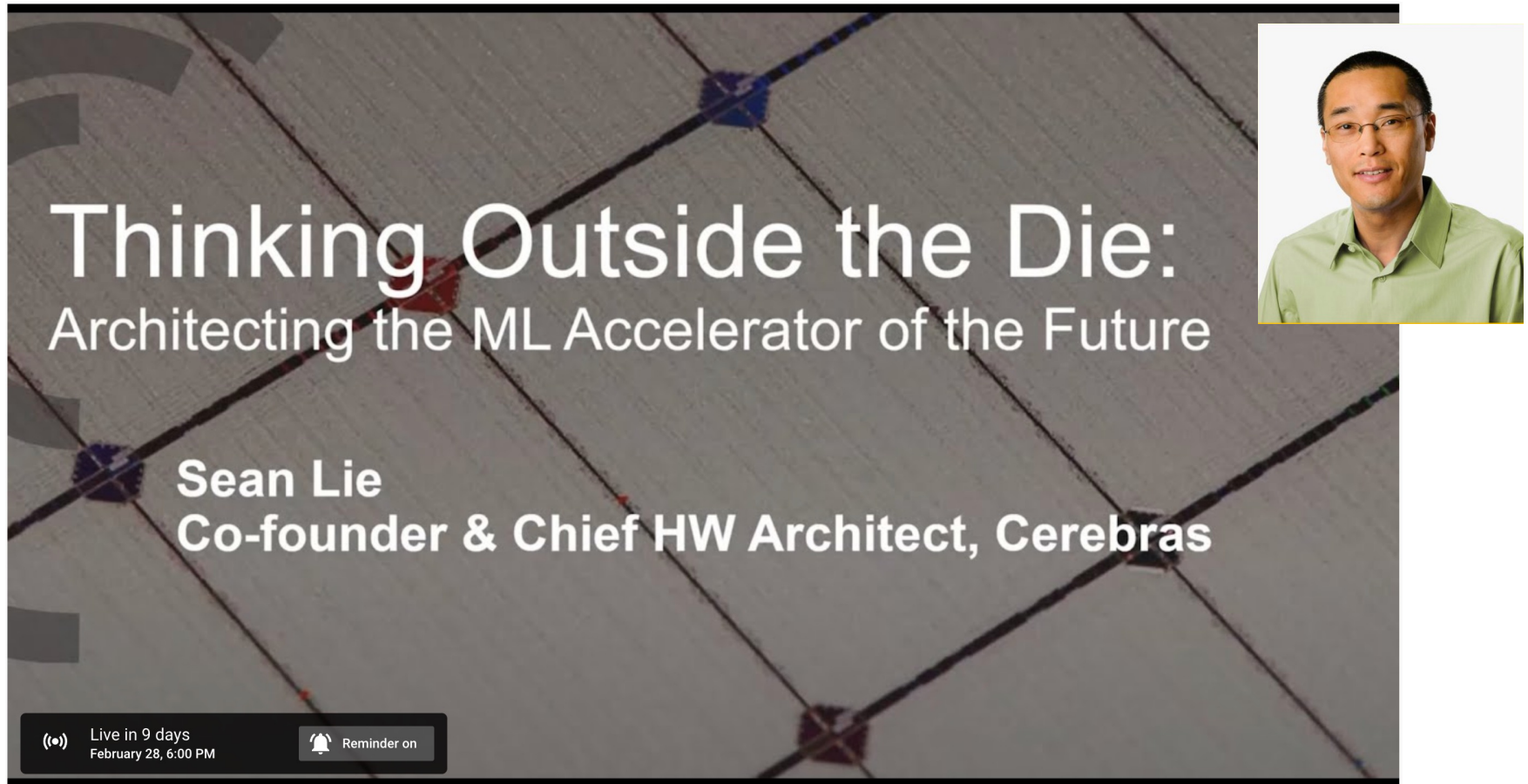
ETH Zürich

Spring 2022

3 March 2022

Recent SAFARI Live Seminar (Feb 28)

<https://www.youtube.com/watch?v=x2-qB0J7KHw>



SAFARI Live Seminar - Thinking Outside the Die: Architecting the ML Accelerator of the Future

1 waiting • Scheduled for Feb 28, 2022

👍 7 🗨 DISLIKE ➦ SHARE ➦+ SAVE ...




Onur Mutlu Lectures
22.6K subscribers

ANALYTICS

EDIT VIDEO

Recent SAFARI Live Seminar (Feb 28)

<https://www.youtube.com/watch?v=x2-qB0J7KHw>



Cerebras CS-2 System

Built from the ground up to run the WSE-2
Single chip equivalent to **56x** traditional chips

Combined with process shrinks...
Imagine more balanced scaling:

- ✓ 2x → **20-100x** Process technology
- 3x → **30-150x** Architecture and march
- 300x: **Scalable** cluster scale-out

Cluster scale performance on a single system

zoom

26:13 / 1:58:55

SAFARI Live Seminar - Thinking Outside the Die: Architecting the ML Accelerator of the Future

2,521 views • Streamed live on Feb 28, 2022

76 DISLIKE SHARE CLIP SAVE ...

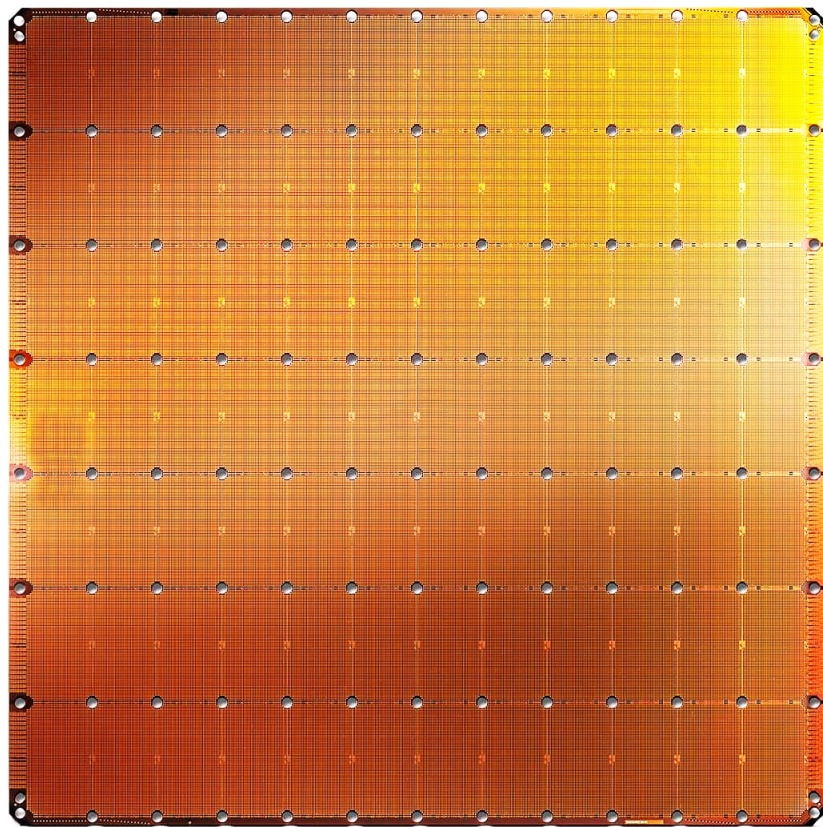


Onur Mutlu Lectures
23K subscribers

ANALYTICS

EDIT VIDEO

Cerebras's Wafer Scale ML Engine (2019)



Cerebras WSE

1.2 Trillion transistors

46,225 mm²

- The largest ML accelerator chip
- 400,000 cores



Largest GPU

21.1 Billion transistors

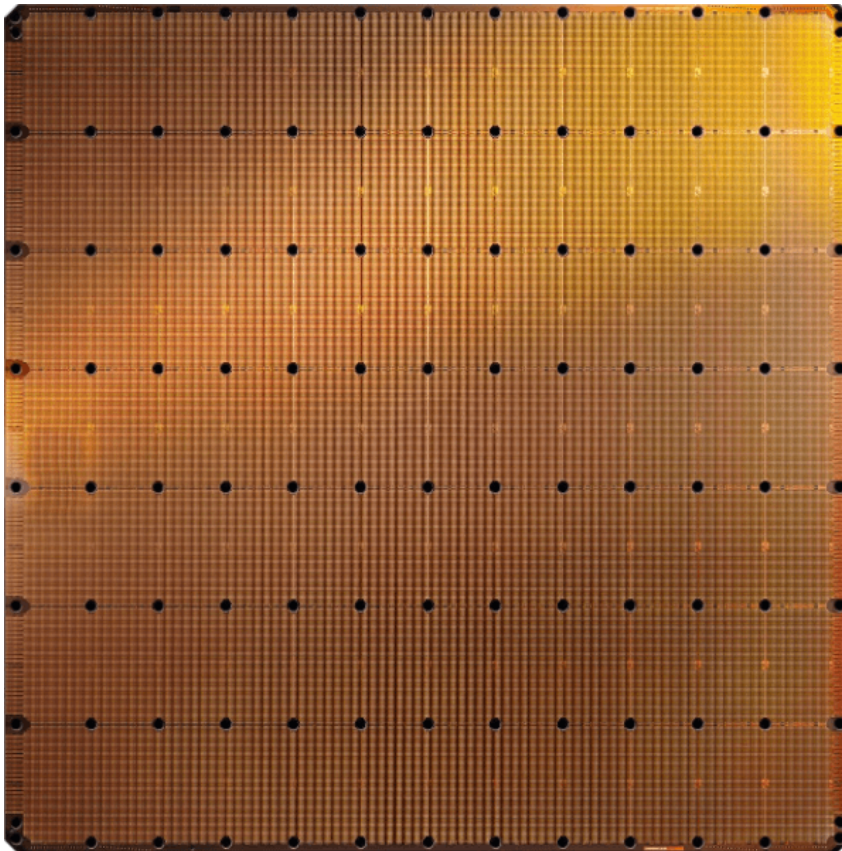
815 mm²

NVIDIA TITAN V

<https://www.anandtech.com/show/14758/hot-chips-31-live-blogs-cerebras-wafer-scale-deep-learning>

<https://www.cerebras.net/cerebras-wafer-scale-engine-why-we-need-big-chips-for-deep-learning>⁴

Cerebras's Wafer Scale ML Engine-2 (2021)



Cerebras WSE-2
2.6 Trillion transistors
46,225 mm²

- The largest ML accelerator chip (2021)
- 850,000 cores



Largest GPU
54.2 Billion transistors
826 mm²

NVIDIA Ampere GA100

<https://www.anandtech.com/show/14758/hot-chips-31-live-blogs-cerebras-wafer-scale-deep-learning>

<https://www.cerebras.net/cerebras-wafer-scale-engine-why-we-need-big-chips-for-deep-learning/>

Extra Credit Assignment: Due March 15

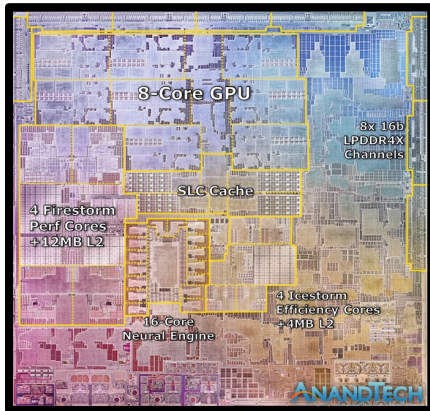
- **Attend and watch Sean Lie's talk on Feb 28**
 - Either on Zoom or Youtube
 - <https://safari.ethz.ch/safari-live-seminar-sean-lie-28-feb-2022/>

- **Optional Assignment – for 1% extra credit**
 - **Write and submit a 1-page summary** of the talk
 - What are the key ideas used in the Cerebras system?
 - What are your key takeaways from the talk?
 - What did you learn?
 - What did you like or dislike?
 - Submit your summary to Moodle: <https://moodle-app2.let.ethz.ch/mod/assign/view.php?id=722952>

General Purpose vs. Special Purpose Systems

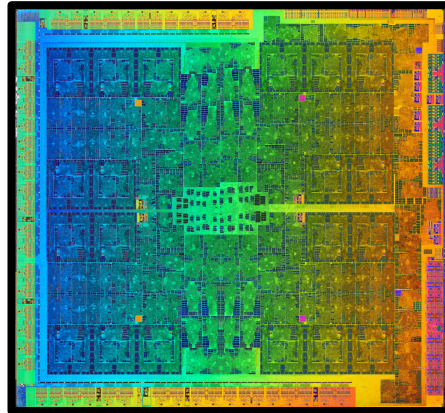
General Purpose

CPU_s



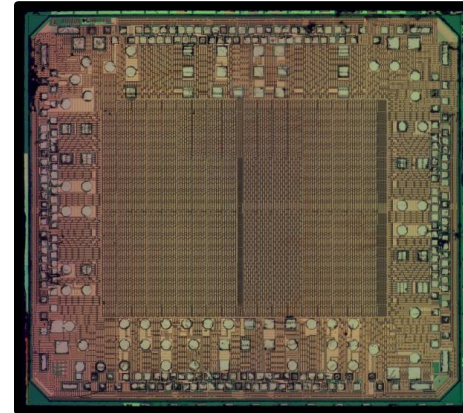
Apple M1

GPU_s



Nvidia GTX 1070

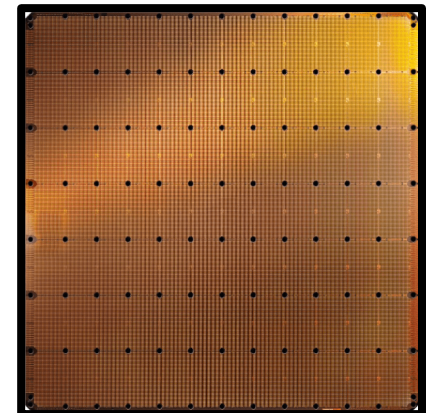
FPGAs



Xilinx Spartan

Special Purpose

ASIC_s



Cerebras WSE-2



Flexible: Can execute any program

Easy to program & use

Not the best performance & efficiency

Efficient & High performance

(Usually) Difficult to program & use

Inflexible: Limited set of programs

General Purpose vs. Special Purpose Systems

General Purpose



Flexible: Can work with any bolt
Easy to use

Not the best fit, results or efficiency

Special Purpose



Efficient & High performance
(Usually) Difficult to use
Inflexible: Only for fitting bolts

Recall: Crossing the Abstraction Layers

- Two goals of this course (especially the second half) are
 - to understand how a processor works underneath the software layer and how decisions made in hardware affect the software/programmer
 - to enable you to comfortably make design and optimization decisions that cross the boundaries of different layers and system components

Some Example “Mysteries”

Four Mysteries: Familiar with Any?

- Rowhammer (2012-2014)
- Meltdown & Spectre (2017-2018)
- Memories Forget: Refresh (2011-2012)
- Memory Performance Attacks (2006-2007)

Mystery #1: RowHammer

Recall: The Story of RowHammer

- One can **predictably induce bit flips** in commodity DRAM chips
 - >80% of the tested DRAM chips are vulnerable
- First example of how a **simple hardware failure mechanism** can create a **widespread system security vulnerability**

WIRED

Forget Software—Now Hackers Are Exploiting Physics

BUSINESS	CULTURE	DESIGN	GEAR	SCIENCE
----------	---------	--------	------	---------

ANDY GREENBERG SECURITY 08.31.16 7:00 AM

SHARE



SHARE
18276



TWEET

FORGET SOFTWARE—NOW HACKERS ARE EXPLOITING PHYSICS

Recall: Security Implications



It's like breaking into an apartment by repeatedly slamming a neighbor's door until the vibrations open the door you were after

Recall: More Security Implications (II)

"Can gain control of a smart phone deterministically"



Drammer: Deterministic Rowhammer
Attacks on Mobile Platforms, CCS'16 ¹⁵

Recall: More Security Implications (VIII)

■ USENIX Security 2020

DeepHammer: Depleting the Intelligence of Deep Neural Networks through Targeted Chain of Bit Flips

Fan Yao

University of Central Florida

fan.yao@ucf.edu

Adnan Siraj Rakin

Arizona State University

asrakin@asu.edu

Deliang Fan

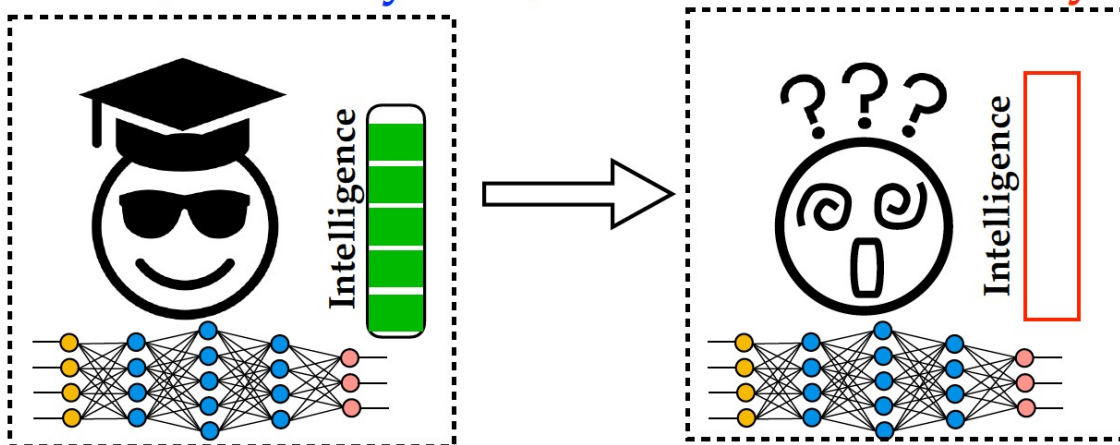
Arizona State University

dfan@asu.edu

Degrade the inference accuracy to the level of Random Guess

Example: ResNet-20 for CIFAR-10, 10 output classes

Before attack, **Accuracy: 90.2%** After attack, **Accuracy: ~10% (1/10)**



How Do We Fix The Problem?

Some Thoughts on RowHammer

How to find, exploit, and fix RowHammer
requires a strong understanding
across the transformation layers

Two Types of RowHammer Solutions

■ Immediate

- ❑ To protect the vulnerable DRAM chips already in the field
- ❑ Limited possibilities

■ Longer-term

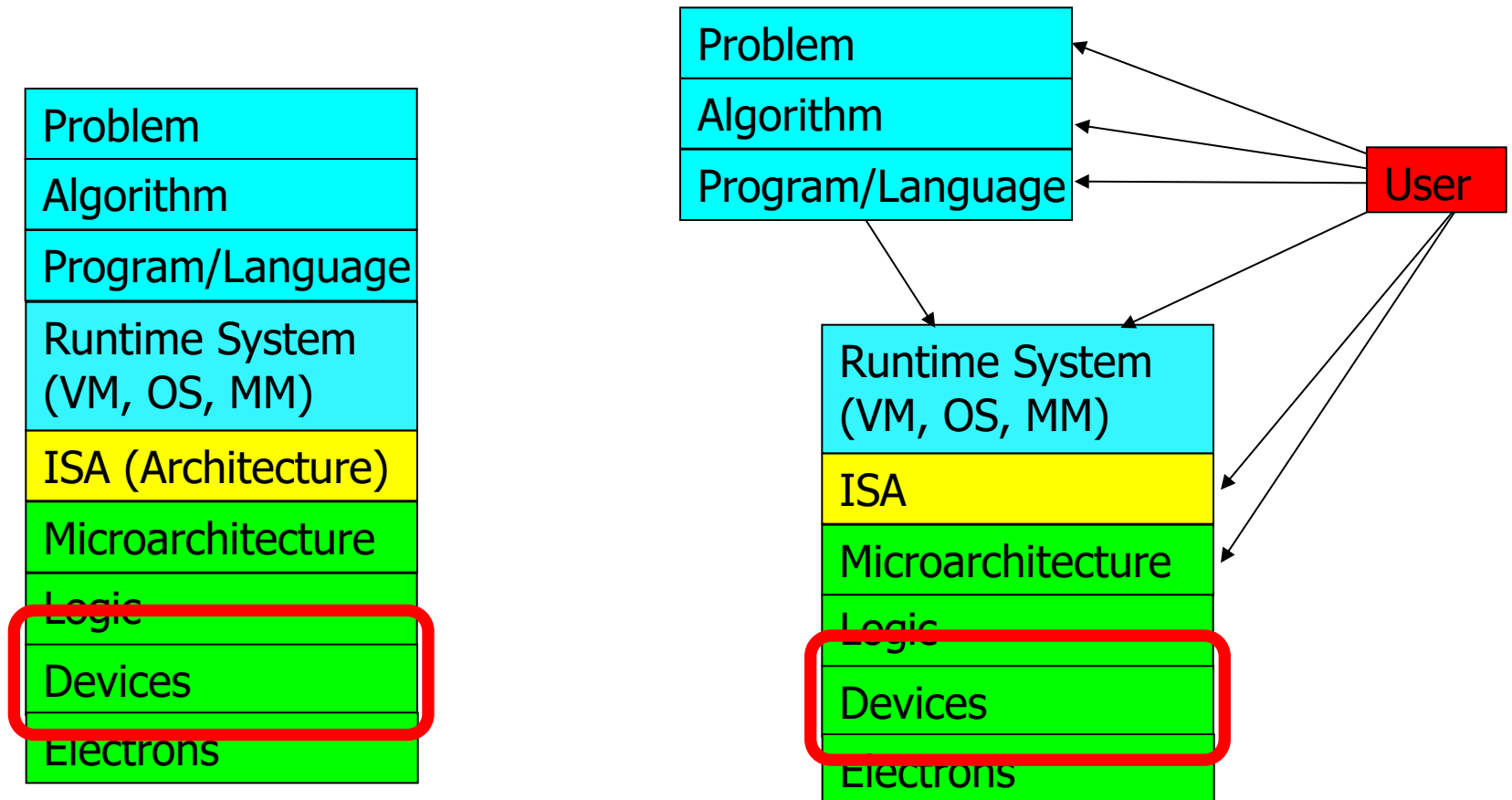
- ❑ To protect future DRAM chips
- ❑ Wider range of protection mechanisms

■ Our ISCA 2014 paper proposes both types of solutions

- ❑ Seven solutions in total
- ❑ PARA proposed as best solution → already employed in the field

Recall Higher-Level Implications

- RowHammer has enormous implications on upper layers of the transformation hierarchy

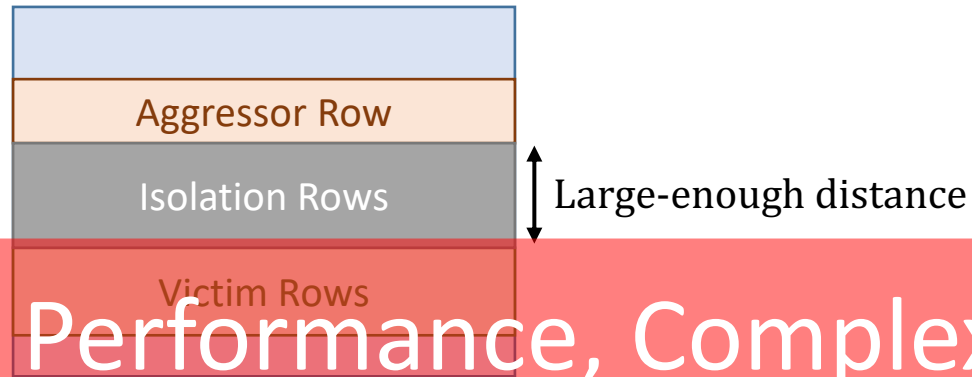


RowHammer Solution Approaches

- More robust DRAM chips **and/or** error-correcting codes
- Increased refresh rate

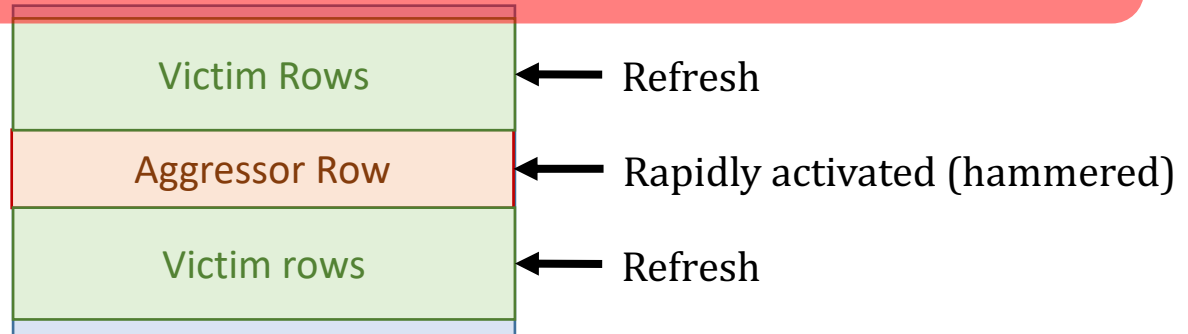


- Physical isolation



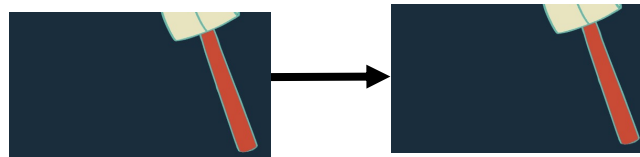
Cost, Power, Performance, Complexity

- Reactive refresh



- Proactive throttling

SAFARI



Fewer activations allowed for aggressive applications

Really Interested?

- Our first detailed study: Rowhammer analysis and solutions (June 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](#)]
- Our Source Code to Induce Errors in Modern DRAM Chips (June 2014)
 - <https://github.com/CMU-SAFARI/rowhammer>
- Google Project Zero's Attack to Take Over a System (March 2015)
 - [Exploiting the DRAM rowhammer bug to gain kernel privileges](#) (Seaborn+, 2015)
 - <https://github.com/google/rowhammer-test>
 - Double-sided Rowhammer

RowHammer: Eight Years Ago...

- 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](#)).

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

RowHammer: 2019 and Beyond...

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

RowHammer: A Retrospective

Onur Mutlu^{§‡} Jeremie S. Kim^{‡§}
§ETH Zürich ‡Carnegie Mellon University

Takeaway

Breaking the abstraction layers
(between components and
transformation hierarchy levels)
and knowing what is underneath
enables you to **understand** and
solve problems

RowHammer in 2020 & 2021

RowHammer is Getting Much Worse

- Jeremie S. Kim, Minesh Patel, A. Giray Yaglikci, Hasan Hassan, Roknoddin Azizi, Lois Orosa, and Onur Mutlu,
"Revisiting RowHammer: An Experimental Analysis of Modern Devices and Mitigation Techniques"
Proceedings of the 47th International Symposium on Computer Architecture (ISCA), Valencia, Spain, June 2020.
[[Slides \(pptx\)](#)] [[pdf](#)]
[[Lightning Talk Slides \(pptx\)](#)] [[pdf](#)]
[[Talk Video](#) (20 minutes)]
[[Lightning Talk Video](#) (3 minutes)]

Revisiting RowHammer: An Experimental Analysis of Modern DRAM Devices and Mitigation Techniques

Jeremie S. Kim^{§†} Minesh Patel[§] A. Giray Yağlıkçı[§]
Hasan Hassan[§] Roknoddin Azizi[§] Lois Orosa[§] Onur Mutlu^{§†}
[§]*ETH Zürich* [†]*Carnegie Mellon University*

Industry-Adopted Solutions Do Not Work

- Pietro Frigo, Emanuele Vannacci, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi,
"TRRespass: Exploiting the Many Sides of Target Row Refresh"
Proceedings of the 41st IEEE Symposium on Security and Privacy (S&P), San Francisco, CA, USA, May 2020.
[[Slides \(pptx\)](#)] [[pdf](#)]
[[Lecture Slides \(pptx\)](#)] [[pdf](#)]
[[Talk Video](#)] (17 minutes)
[[Lecture Video](#)] (59 minutes)
[[Source Code](#)]
[[Web Article](#)]
Best paper award.
Pwnie Award 2020 for Most Innovative Research. [Pwnie Awards 2020](#)

TRRespass: Exploiting the Many Sides of Target Row Refresh

Pietro Frigo^{*†} Emanuele Vannacci^{*†} Hasan Hassan[§] Victor van der Veen[¶]
Onur Mutlu[§] Cristiano Giuffrida^{*} Herbert Bos^{*} Kaveh Razavi^{*}

Hard to Guarantee RowHammer-Free Chips

- Lucian Cojocar, Jeremie Kim, Minesh Patel, Lillian Tsai, Stefan Saroiu, Alec Wolman, and Onur Mutlu,

"Are We Susceptible to Rowhammer? An End-to-End Methodology for Cloud Providers"

Proceedings of the 41st IEEE Symposium on Security and Privacy (S&P), San Francisco, CA, USA, May 2020.

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

[[Talk Video](#) (17 minutes)]

Are We Susceptible to Rowhammer?

An End-to-End Methodology for Cloud Providers

Lucian Cojocar, Jeremie Kim^{§†}, Minesh Patel[§], Lillian Tsai[‡],
Stefan Saroiu, Alec Wolman, and Onur Mutlu^{§†}
Microsoft Research, [§]ETH Zürich, [†]CMU, [‡]MIT

RowHammer Has Many Dimensions

- Lois Orosa, Abdullah Giray Yaglikci, Haocong Luo, Ataberk Olgun, Jisung Park, Hasan Hassan, Minesh Patel, Jeremie S. Kim, and Onur Mutlu,
"A Deeper Look into RowHammer's Sensitivities: Experimental Analysis of Real DRAM Chips and Implications on Future Attacks and Defenses"
*Proceedings of the 54th International Symposium on Microarchitecture (**MICRO**), Virtual, October 2021.*
[[Slides \(pptx\)](#)] [[pdf](#)]
[[Short Talk Slides \(pptx\)](#)] [[pdf](#)]
[[Lightning Talk Slides \(pptx\)](#)] [[pdf](#)]
[[Talk Video](#) (21 minutes)]
[[Lightning Talk Video](#) (1.5 minutes)]
[[arXiv version](#)]

A Deeper Look into RowHammer's Sensitivities: Experimental Analysis of Real DRAM Chips and Implications on Future Attacks and Defenses

Lois Orosa*
ETH Zürich

A. Giray Yağlıkçı*
ETH Zürich

Haocong Luo
ETH Zürich

Ataberk Olgun
ETH Zürich, TOBB ETÜ

Jisung Park
ETH Zürich

Hasan Hassan
ETH Zürich

Minesh Patel
ETH Zürich

Jeremie S. Kim
ETH Zürich

Onur Mutlu
ETH Zürich

Industry-Adopted Solutions Are Poor

- Hasan Hassan, Yahya Can Tugrul, Jeremie S. Kim, Victor van der Veen, Kaveh Razavi, and Onur Mutlu,
"Uncovering In-DRAM RowHammer Protection Mechanisms: A New Methodology, Custom RowHammer Patterns, and Implications"
Proceedings of the 54th International Symposium on Microarchitecture (MICRO), Virtual, October 2021.
[[Slides \(pptx\)](#)] [[pdf](#)]
[[Short Talk Slides \(pptx\)](#)] [[pdf](#)]
[[Lightning Talk Slides \(pptx\)](#)] [[pdf](#)]
[[Talk Video](#) (25 minutes)]
[[Lightning Talk Video](#) (100 seconds)]
[[arXiv version](#)]

Uncovering In-DRAM RowHammer Protection Mechanisms: A New Methodology, Custom RowHammer Patterns, and Implications

Hasan Hassan[†]

[†]ETH Zürich

Yahya Can Tuğrul^{†‡}

Kaveh Razavi[†]
[‡]TOBB University of Economics & Technology

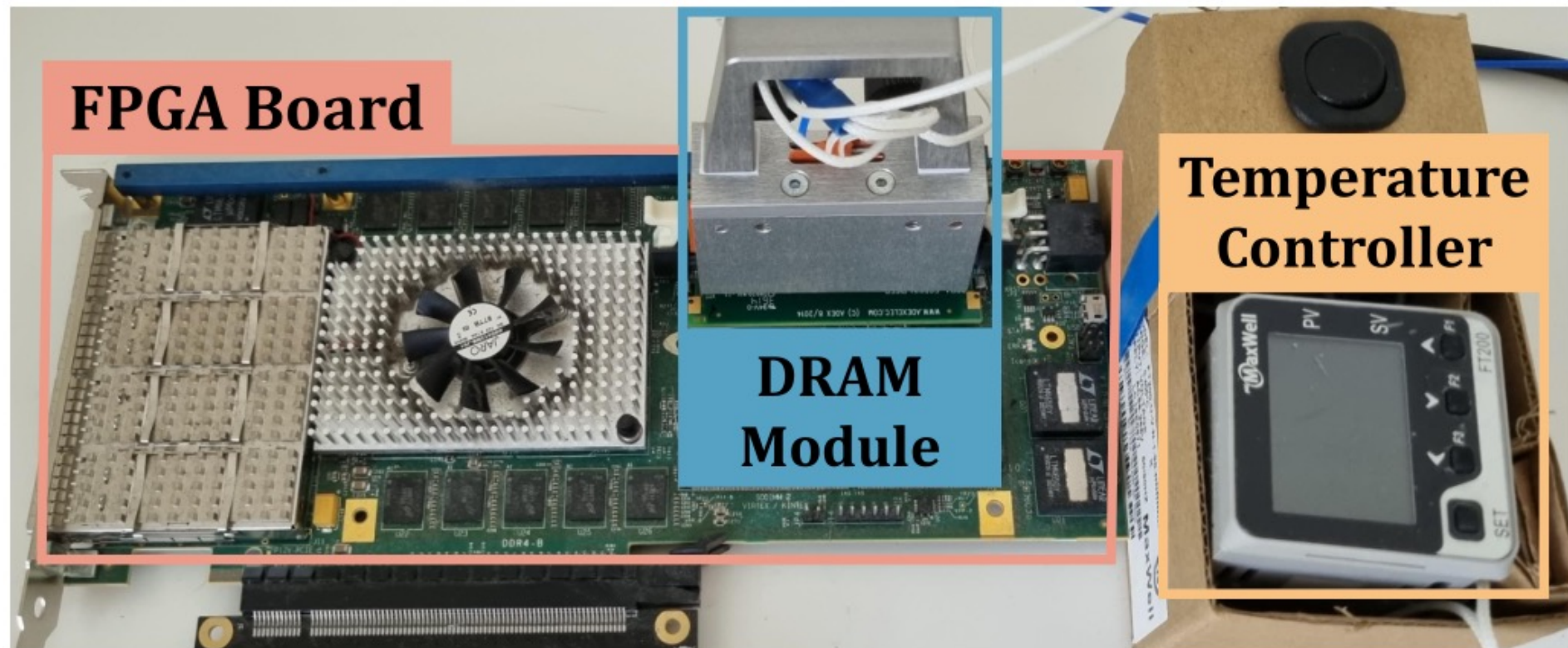
Jeremie S. Kim[†]

Onur Mutlu[†]

Victor van der Veen^σ

^σQualcomm Technologies Inc.

Analyzing “Protected” DDR4 Chips



* *SoftMC [Hassan+, HPCA'17] enhanced for DDR4*

Key Takeaways

All 45 modules we tested are vulnerable

99.9% of rows
experience at least one RowHammer bit flip

Error Correcting Codes (ECC) is ineffective

Module	Date (yy-ww)	Chip Density (Gbit)	Organization			HC_{first}^{\dagger}	Our Key TRR Observations and Results							
			Ranks	Banks	Pins		Version	Aggressor Detection	Aggressor Capacity	Per-Bank TRR	TRR-to-REF Ratio	Neighbors Refreshed	% Vulnerable DRAM Rows [†]	Max. Bit Flips per Row per Hammer [†]
A0	19-50	8	1	16	8	16K	A_{TRR1}	Counter-based	16	✓	1/9	4	73.3%	1.16
A1-5	19-36	8	1	8	16	13K-15K	A_{TRR1}	Counter-based	16	✓	1/9	4	99.2% - 99.4%	2.32 - 4.73
A6-7	19-45	8	1	8	16	13K-15K	A_{TRR1}	Counter-based	16	✓	1/9	4	99.3% - 99.4%	2.12 - 3.86
A8-9	20-07	8	1	16	8	12K-14K	A_{TRR1}	Counter-based	16	✓	1/9	4	74.6% - 75.0%	1.96 - 2.96
A10-12	19-51	8	1	16	8	12K-13K	A_{TRR1}	Counter-based	16	✓	1/9	4	74.6% - 75.0%	1.48 - 2.86
A13-14	20-31	8	1	8	16	11K-14K	A_{TRR2}	Counter-based	16	✓	1/9	2	94.3% - 98.6%	1.53 - 2.78
B0	18-22	4	1	16	8	44K	B_{TRR1}	Sampling-based	1	✗	1/4	2	99.9%	2.13
B1-4	20-17	4	1	16	8	159K-192K	B_{TRR1}	Sampling-based	1	✗	1/4	2	23.3% - 51.2%	0.06 - 0.11
B5-6	16-48	4	1	16	8	44K-50K	B_{TRR1}	Sampling-based	1	✗	1/4	2	99.9%	1.85 - 2.03
B7	19-06	8	2	16	8	20K	B_{TRR1}	Sampling-based	1	✗	1/4	2	99.9%	31.14
B8	18-03	4	1	16	8	43K	B_{TRR1}	Sampling-based	1	✗	1/4	2	99.9%	2.57
B9-12	19-48	8	1	16	8	42K-65K	B_{TRR2}	Sampling-based	1	✗	1/9	2	36.3% - 38.9%	16.83 - 24.26
B13-14	20-08	4	1	16	8	11K-14K	B_{TRR3}	Sampling-based	1	✓	1/2	4	99.9%	16.20 - 18.12
C0-3	16-48	4	1	16	x8	137K-194K	C_{TRR1}	Mix	Unknown	✓	1/17	2	1.0% - 23.2%	0.05 - 0.15
C4-6	17-12	8	1	16	x8	130K-150K	C_{TRR1}	Mix	Unknown	✓	1/17	2	7.8% - 12.0%	0.06 - 0.08
C7-8	20-31	8	1	8	x16	40K-44K	C_{TRR1}	Mix	Unknown	✓	1/17	2	39.8% - 41.8%	9.66 - 14.56
C9-11	20-31	8	1	8	x16	42K-53K	C_{TRR2}	Mix	Unknown	✓	1/9	2	99.7%	9.30 - 32.04
C12-14	20-46	16	1	8	x16	6K-7K	C_{TRR3}	Mix	Unknown	✓	1/8	2	99.9%	4.91 - 12.64

Industry-Adopted Solutions Are Poor

- Hasan Hassan, Yahya Can Tugrul, Jeremie S. Kim, Victor van der Veen, Kaveh Razavi, and Onur Mutlu,
"Uncovering In-DRAM RowHammer Protection Mechanisms: A New Methodology, Custom RowHammer Patterns, and Implications"
Proceedings of the 54th International Symposium on Microarchitecture (MICRO), Virtual, October 2021.
[[Slides \(pptx\)](#)] [[pdf](#)]
[[Short Talk Slides \(pptx\)](#)] [[pdf](#)]
[[Lightning Talk Slides \(pptx\)](#)] [[pdf](#)]
[[Talk Video](#) (25 minutes)]
[[Lightning Talk Video](#) (100 seconds)]
[[arXiv version](#)]

Uncovering In-DRAM RowHammer Protection Mechanisms: A New Methodology, Custom RowHammer Patterns, and Implications

Hasan Hassan[†]

[†]ETH Zürich

Yahya Can Tuğrul^{†‡}

Kaveh Razavi[†]
[‡]TOBB University of Economics & Technology

Jeremie S. Kim[†]

Onur Mutlu[†]

Victor van der Veen^σ

^σQualcomm Technologies Inc.

BlockHammer Solution in 2021

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

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

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](#) (22 minutes)]

[[Short Talk Video](#) (7 minutes)]

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

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

¹ETH Zürich

²University of Illinois at Urbana–Champaign

An Intelligent
Memory Controller
can fix the problem

Detailed Lectures on RowHammer

■ Computer Architecture, Fall 2021, Lecture 5

- RowHammer (ETH Zürich, Fall 2021)
- <https://www.youtube.com/watch?v=7wVKnPj3NVw&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAGILkTOF&index=5>

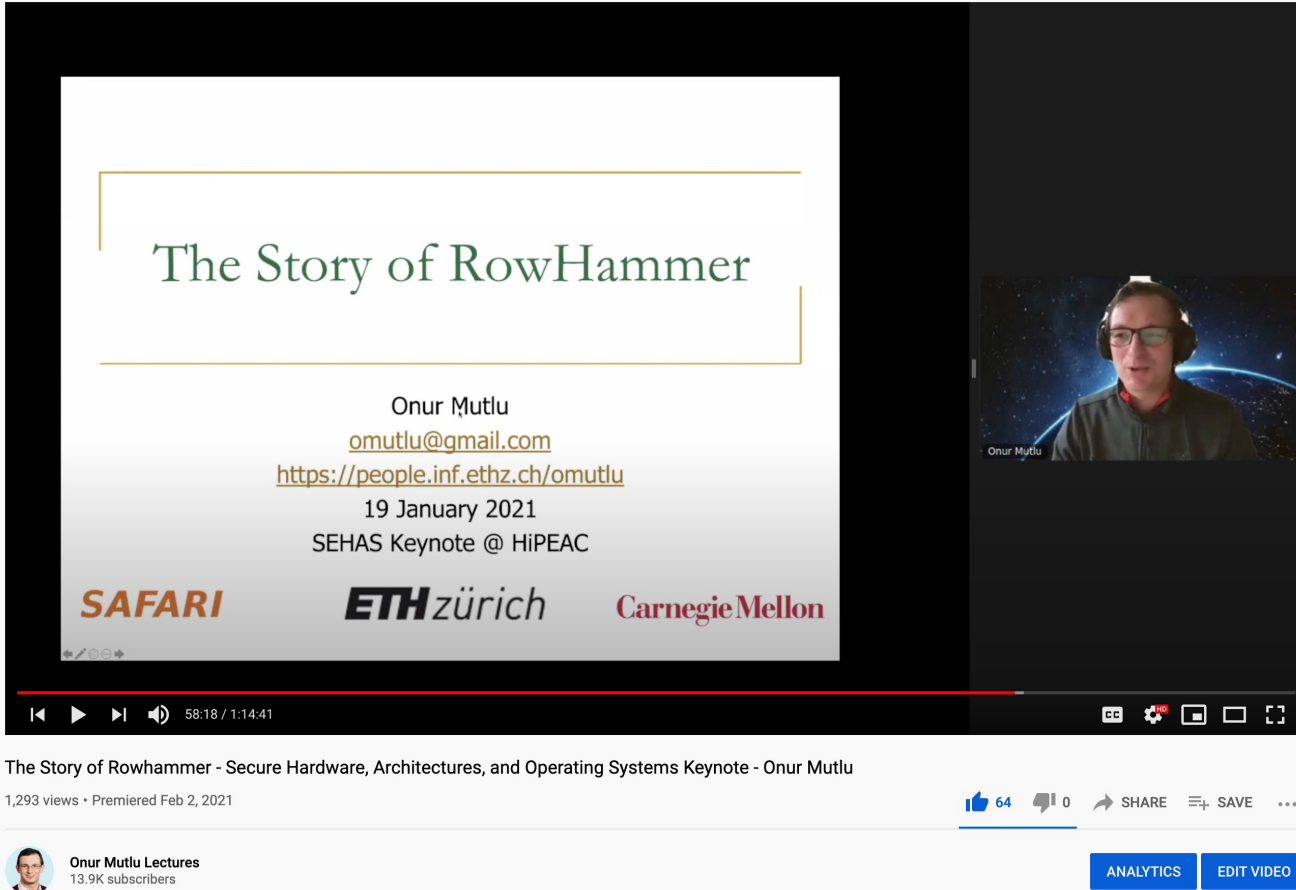
■ Computer Architecture, Fall 2021, Lecture 6

- RowHammer and Secure & Reliable Memory (ETH Zürich, Fall 2021)
- <https://www.youtube.com/watch?v=HNd4skQrt6I&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAGILkTOF&index=6>

<https://www.youtube.com/onurmutlulectures>

The Story of RowHammer Lecture ...

- Onur Mutlu,
"The Story of RowHammer"
Keynote Talk at *Secure Hardware, Architectures, and Operating Systems Workshop (SeHAS)*, held with *HiPEAC 2021 Conference*, Virtual, 19 January 2021.
[Slides (pptx)] (pdf)
[Talk Video] (1 hr 15 minutes, with Q&A)]



The video player shows a presentation slide titled "The Story of RowHammer" by Onur Mutlu. The slide includes contact information: omutlu@gmail.com, <https://people.inf.ethz.ch/omutlu>, and the date 19 January 2021. It also mentions "SEHAS Keynote @ HiPEAC". Logos for SAFARI, ETH zürich, and Carnegie Mellon are at the bottom. The video player interface shows a progress bar at 58:18 / 1:14:41 and a video feed of Onur Mutlu on the right. Below the player, the video title is "The Story of Rowhammer - Secure Hardware, Architectures, and Operating Systems Keynote - Onur Mutlu", with 1,293 views and a premiere date of Feb 2, 2021. The channel "Onur Mutlu Lectures" has 13.9K subscribers. Interaction buttons for likes (64), comments (0), share, save, and analytics are visible.

The Story of RowHammer

Onur Mutlu
omutlu@gmail.com
<https://people.inf.ethz.ch/omutlu>
19 January 2021
SEHAS Keynote @ HiPEAC

SAFARI ETH zürich Carnegie Mellon

58:18 / 1:14:41

The Story of Rowhammer - Secure Hardware, Architectures, and Operating Systems Keynote - Onur Mutlu

1,293 views • Premiered Feb 2, 2021

64 0 SHARE SAVE ...

Onur Mutlu Lectures
13.9K subscribers

ANALYTICS EDIT VIDEO

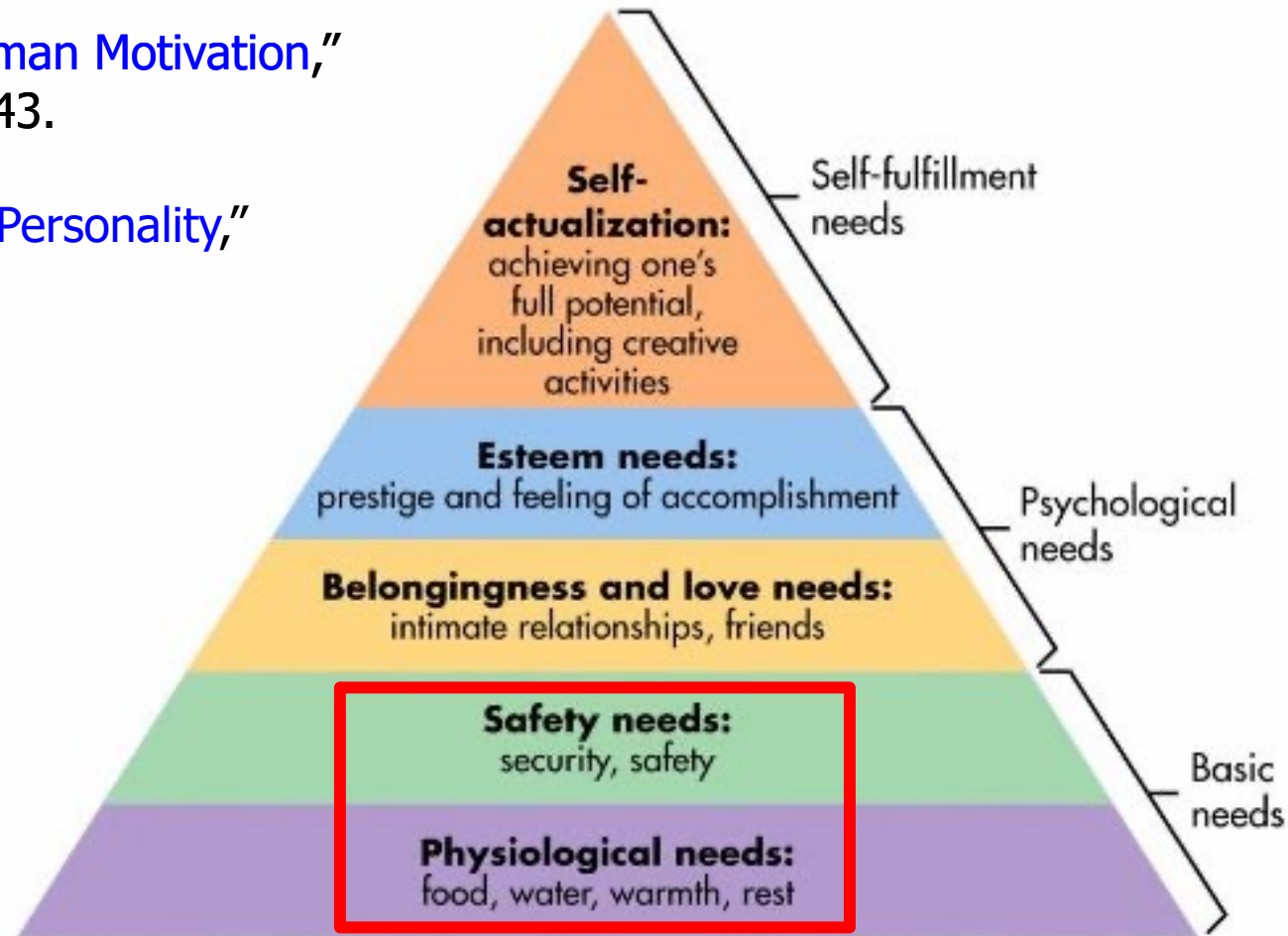


Rowhammer

Maslow's (Human) Hierarchy of Needs

Maslow, "A Theory of Human Motivation,"
Psychological Review, 1943.

Maslow, "Motivation and Personality,"
Book, 1954-1970.



- We need to start with **reliability, safety, and security**...

How Reliable/Safe/Secure is This Bridge?



Collapse of the “Galloping Gertie”



How Safe & Secure Are These People?



Security is about preventing unforeseen consequences

Can We Depend on Computers?



Example Security Implications (VIII)

■ USENIX Security 2020

DeepHammer: Depleting the Intelligence of Deep Neural Networks through Targeted Chain of Bit Flips

Fan Yao
University of Central Florida
fan.yao@ucf.edu

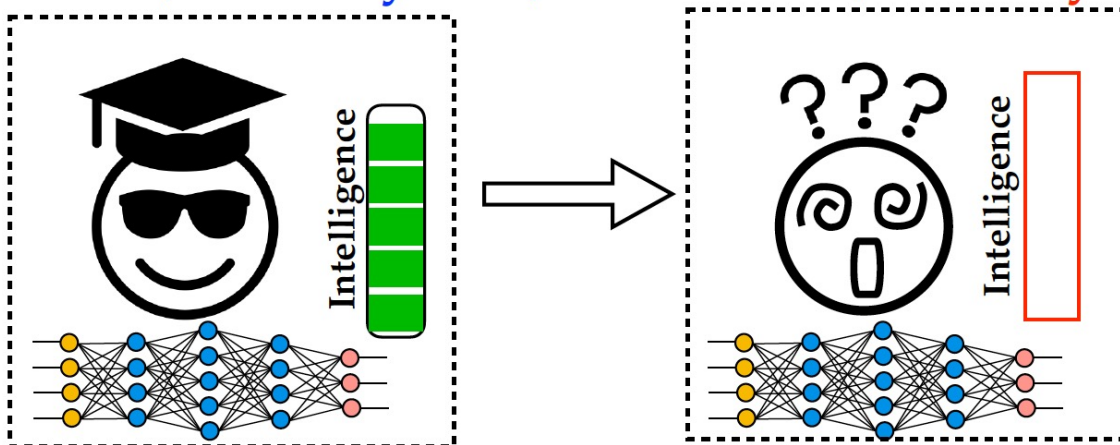
Adnan Siraj Rakin
Arizona State University
asrakin@asu.edu

Deliang Fan
Arizona State University
dfan@asu.edu

Degrade the inference accuracy to the level of Random Guess

Example: ResNet-20 for CIFAR-10, 10 output classes

Before attack, **Accuracy: 90.2%** After attack, **Accuracy: ~10% (1/10)**



Two Other Goals of This Course

- ❑ Enable you to think critically
- ❑ Enable you to think broadly

RowHammer: Retrospective

- New mindset that has enabled a renewed interest in HW security attack research:
 - ❑ Real (memory) chips are vulnerable, in a simple and widespread manner
→ this causes real security problems
 - ❑ Hardware reliability → security connection is now mainstream discourse
- Many new RowHammer attacks...
 - ❑ Tens of papers in top security & architecture venues
 - ❑ **More to come** as RowHammer is getting worse (DDR4 & beyond)
- Many new RowHammer solutions...
 - ❑ Apple security release; Memtest86 updated
 - ❑ Many solution proposals in top venues (latest tomorrow in ASPLOS)
 - ❑ Principled system-DRAM co-design (in original RowHammer paper)
 - ❑ **More to come...**

Perhaps Most Importantly...

- RowHammer enabled a shift of mindset in mainstream security researchers
 - General-purpose hardware is fallible, in a widespread manner
 - Its problems are exploitable
- This mindset has enabled many systems security researchers to examine hardware in more depth
 - And understand HW's inner workings and vulnerabilities
- It is no coincidence that two of the groups that discovered Meltdown and Spectre heavily worked on RowHammer attacks before
 - **More to come...**

A RowHammer Survey Across the Stack

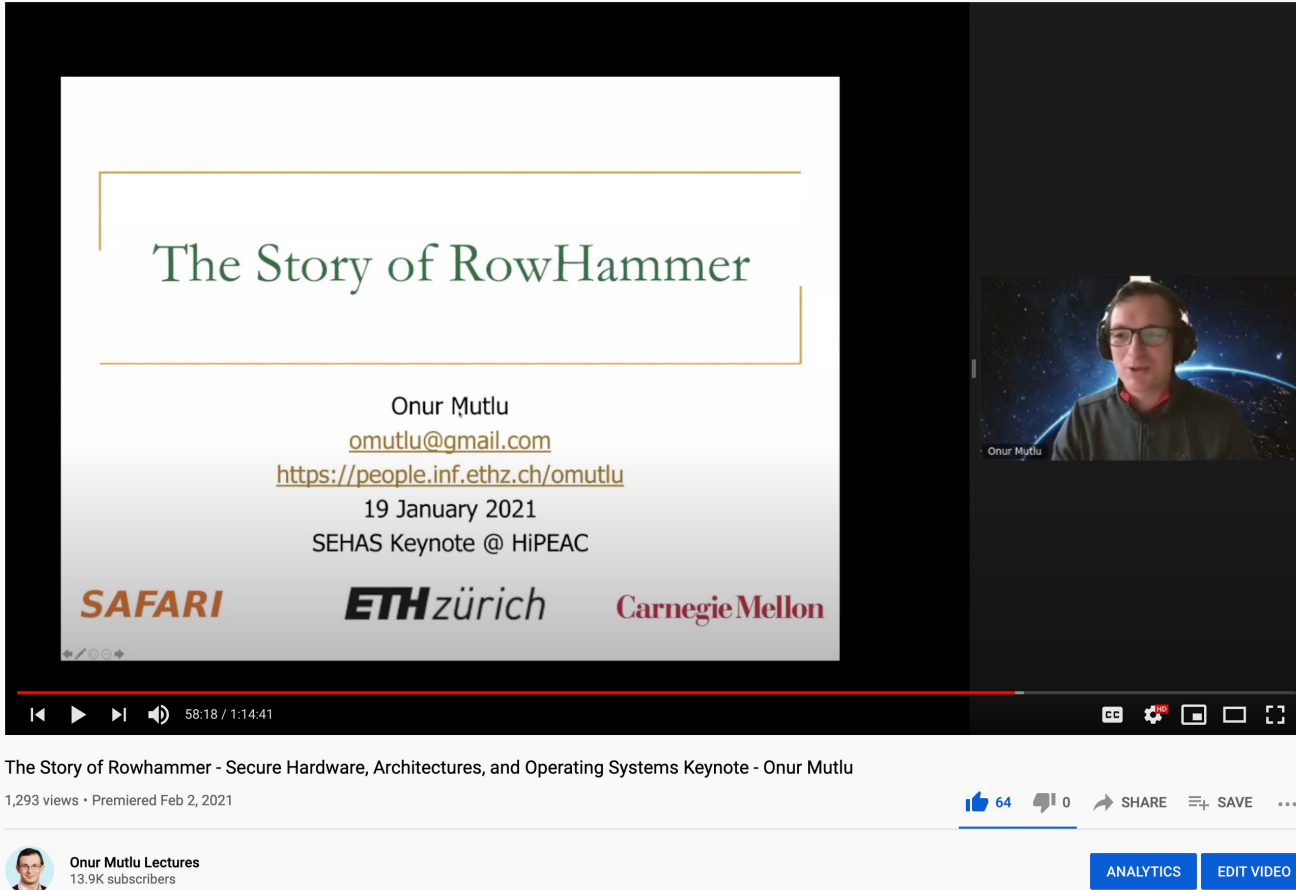
- Onur Mutlu and Jeremie Kim,
[**"RowHammer: A Retrospective"**](#)
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD) Special Issue on Top Picks in Hardware and Embedded Security, 2019.
[[Preliminary arXiv version](#)]
[[Slides from COSADE 2019 \(pptx\)](#)]
[[Slides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

RowHammer: A Retrospective

Onur Mutlu^{§‡} Jeremie S. Kim^{‡§}
[§]ETH Zürich [‡]Carnegie Mellon University

The Story of RowHammer Lecture ...

- Onur Mutlu,
"The Story of RowHammer"
Keynote Talk at *Secure Hardware, Architectures, and Operating Systems Workshop (SeHAS)*, held with *HiPEAC 2021 Conference*, Virtual, 19 January 2021.
[[Slides \(pptx\)](#) ([pdf](#))]
[[Talk Video](#) (1 hr 15 minutes, with Q&A)]



The video player shows a presentation slide titled "The Story of RowHammer" by Onur Mutlu. The slide includes contact information: omutlu@gmail.com, <https://people.inf.ethz.ch/omutlu>, and the date 19 January 2021. It also mentions "SEHAS Keynote @ HiPEAC". Logos for SAFARI, ETH zürich, and Carnegie Mellon are at the bottom. The video player interface shows a progress bar at 58:18 / 1:14:41 and a video feed of Onur Mutlu on the right. Below the player, the video title is "The Story of Rowhammer - Secure Hardware, Architectures, and Operating Systems Keynote - Onur Mutlu", with 1,293 views and a premiere date of Feb 2, 2021. The channel "Onur Mutlu Lectures" has 13.9K subscribers. Interaction buttons for likes (64), comments (0), share, save, and analytics are visible.

The Story of RowHammer

Onur Mutlu
omutlu@gmail.com
<https://people.inf.ethz.ch/omutlu>
19 January 2021
SEHAS Keynote @ HiPEAC

SAFARI ETH zürich Carnegie Mellon

58:18 / 1:14:41

The Story of Rowhammer - Secure Hardware, Architectures, and Operating Systems Keynote - Onur Mutlu

1,293 views • Premiered Feb 2, 2021

64 0 SHARE SAVE ...

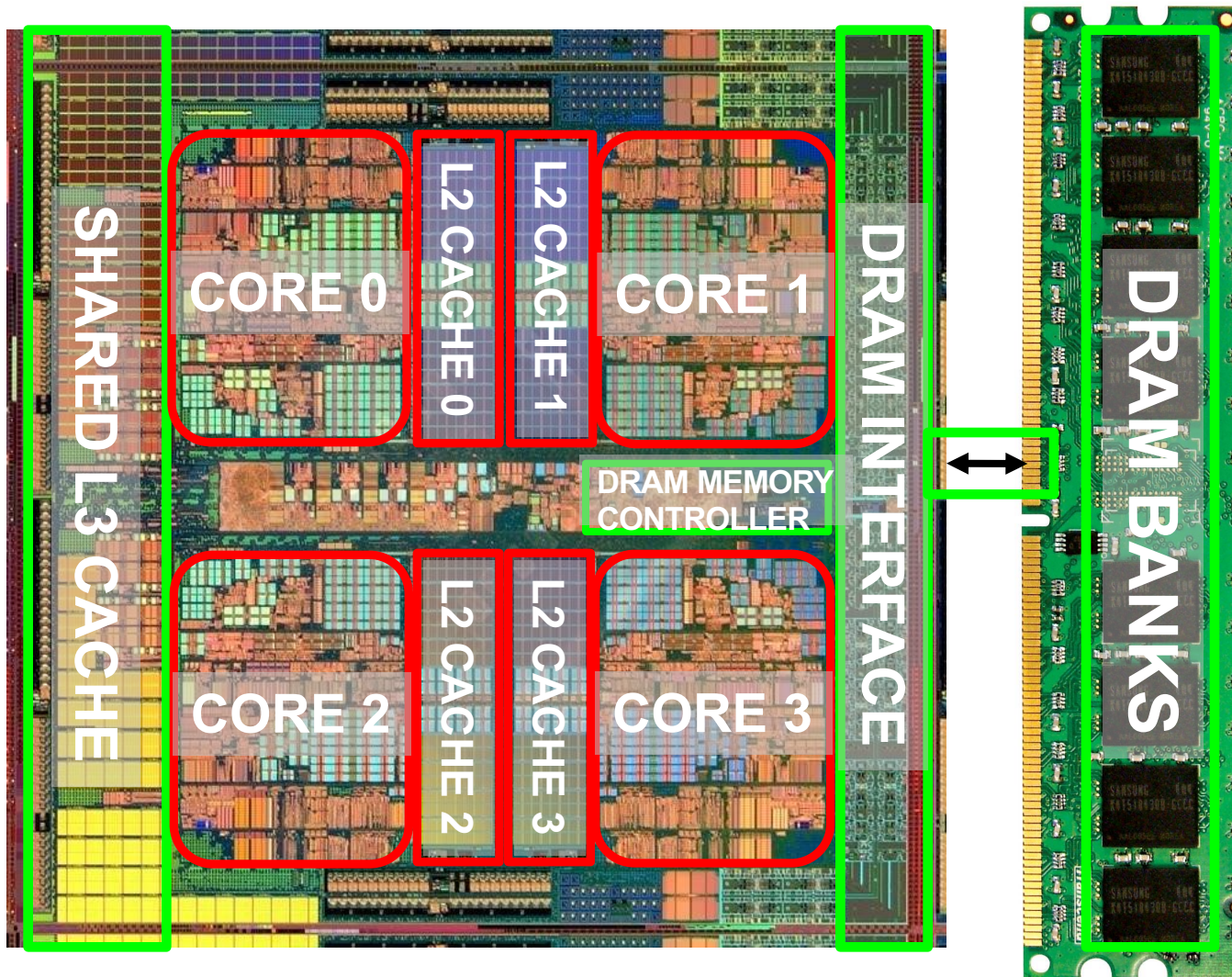
Onur Mutlu Lectures
13.9K subscribers

ANALYTICS EDIT VIDEO

Mystery #3: DRAM Refresh

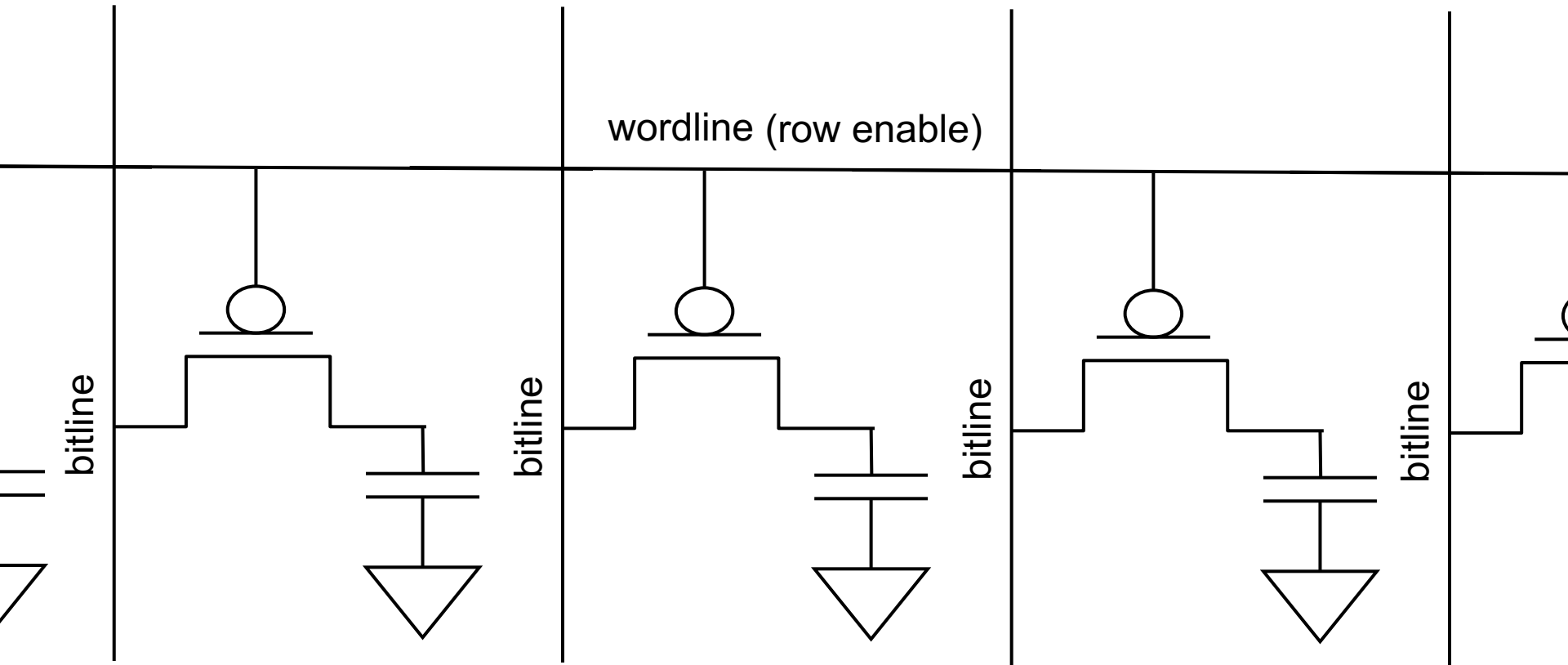
Main Memory (DRAM) in the System

Multi-Core
Chip



*Die photo credit: AMD Barcelona

A DRAM Cell



- A DRAM cell consists of a capacitor and an access transistor
 - It stores data in terms of charge status of the capacitor
 - A DRAM chip consists of (10s of 1000s of) rows of such cells
-

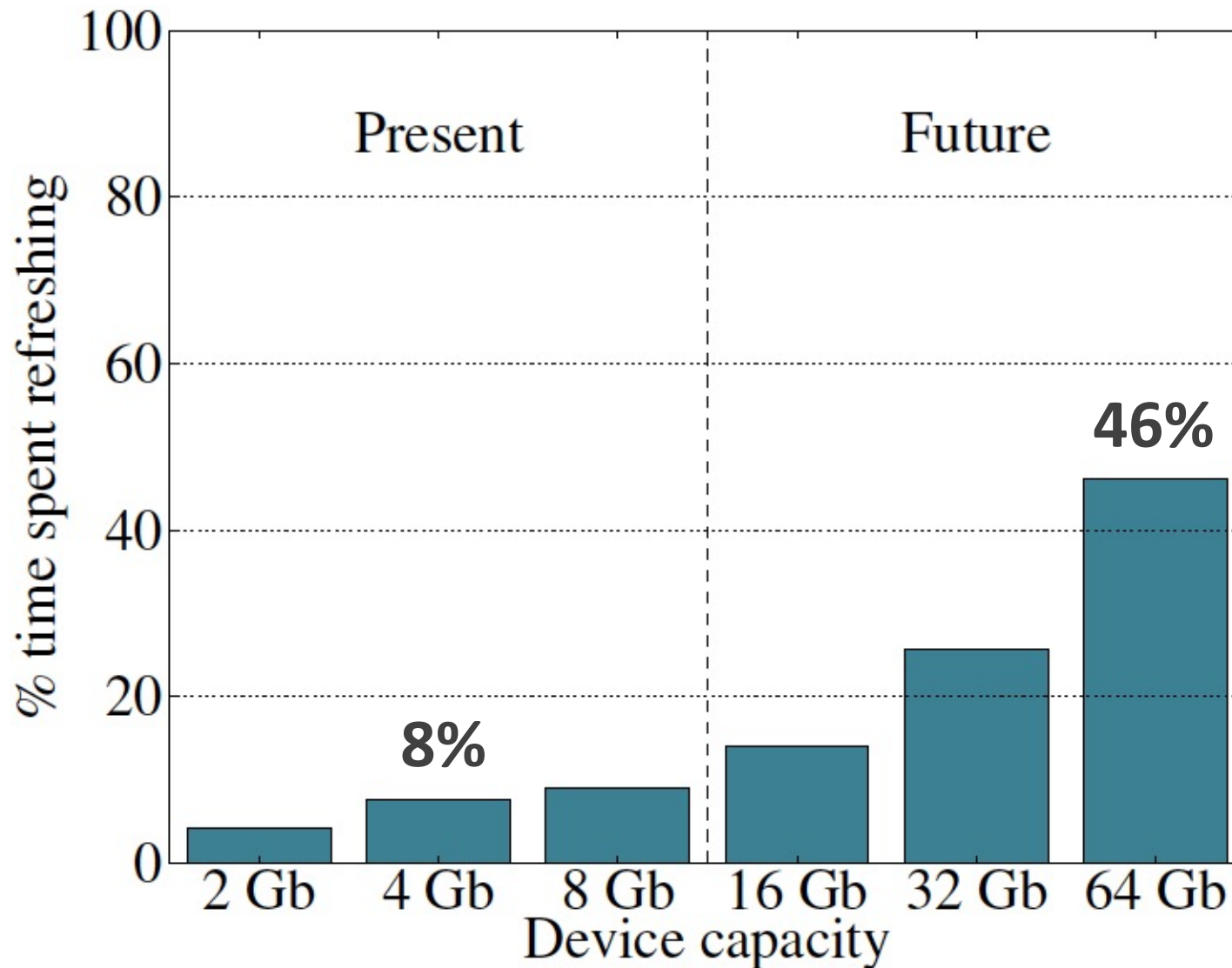
DRAM Refresh

- DRAM capacitor charge leaks over time
- The memory controller needs to refresh each row periodically to restore charge
 - Activate each row every N ms
 - Typical $N = 64$ ms
- Downsides of refresh
 - **Energy consumption**: Each refresh consumes energy
 - **Performance degradation**: DRAM rank/bank unavailable while refreshed
 - **QoS/predictability impact**: (Long) pause times during refresh
 - **Refresh rate limits DRAM capacity scaling**

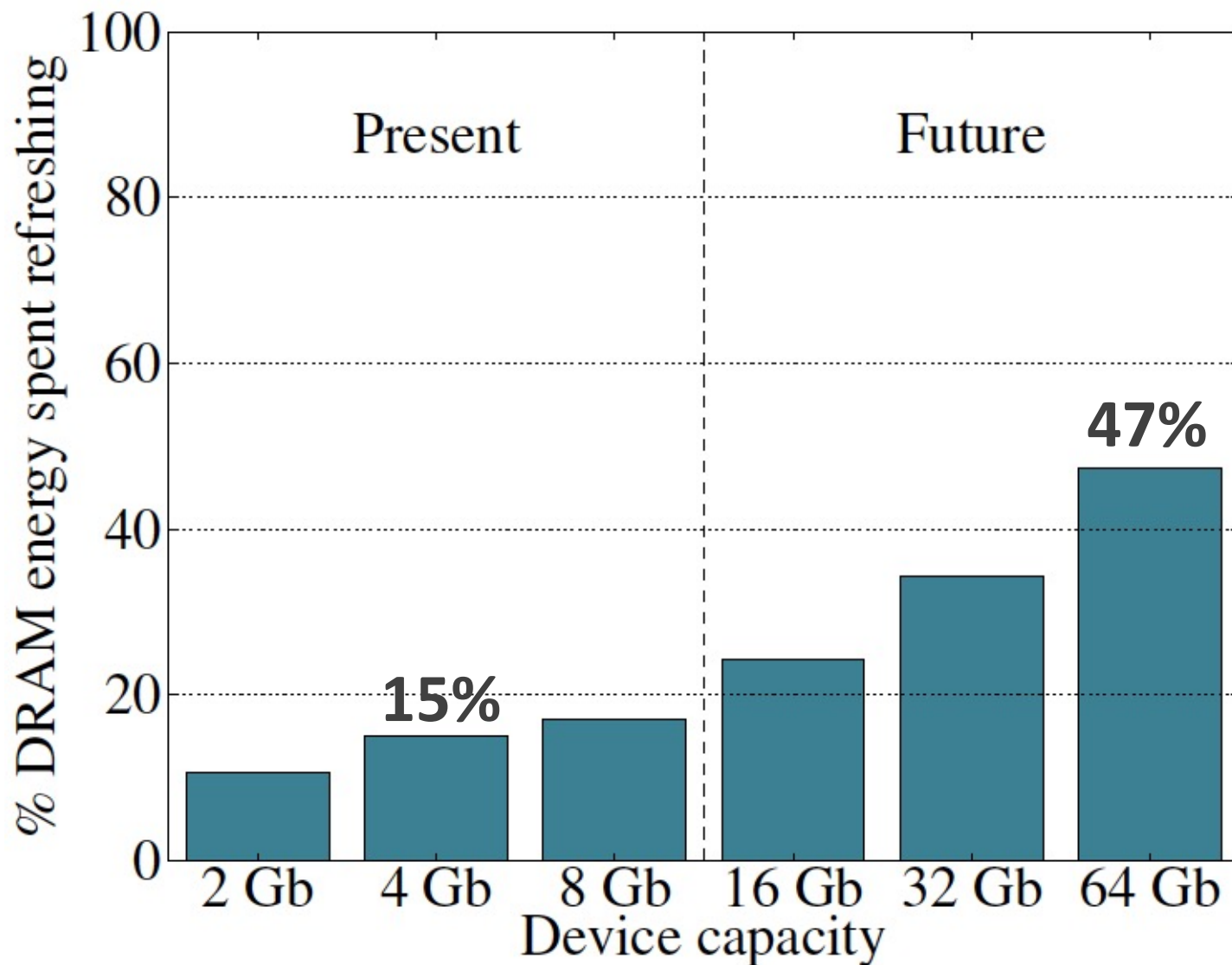
First, Some Analysis

- Imagine a system with 1 ExaByte DRAM (2^{60} bytes)
- Assume a row size of 8 KiloBytes (2^{13} bytes)
- How many rows are there?
- How many refreshes happen in 64ms?
- What is the total power consumption of DRAM refresh?
- What is the total energy consumption of DRAM refresh during one day?
- A good exercise... Optional homework...
- Brownie points from me if you do it...

Refresh Overhead: Performance



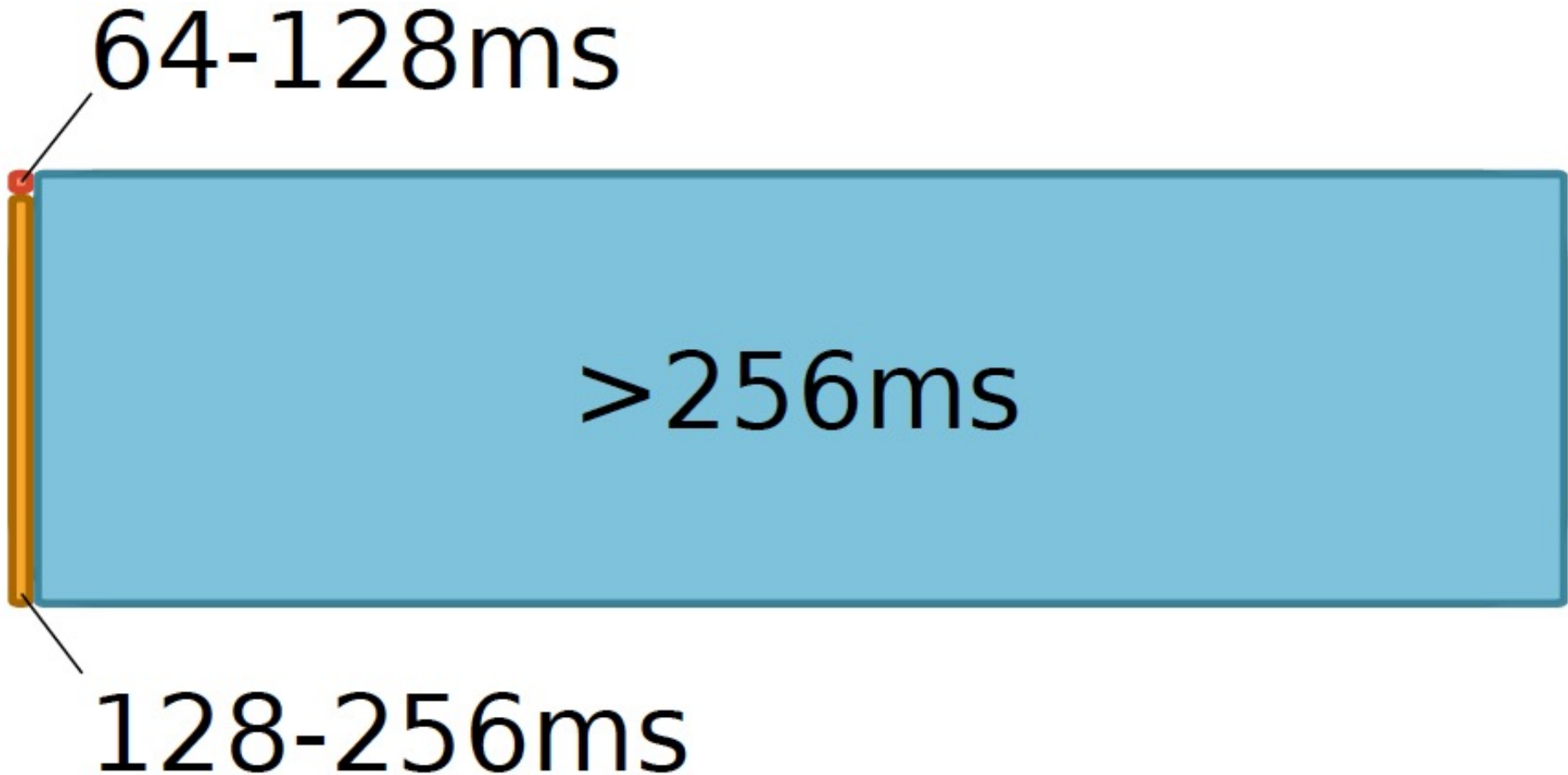
Refresh Overhead: Energy



How Do We Solve the Problem?

- Observation: All DRAM rows are refreshed every 64ms.
- **Critical thinking:** Do we have to refresh all rows every 64ms?
- What if we knew what happened underneath and exposed that information to upper layers?

Underneath: Retention Time Profile of DRAM



Aside: Why Do We Have Such a Profile?

- Answer: Manufacturing is not perfect
- Not all DRAM cells are exactly the same
- Some are leakier than others
- This is called **Manufacturing Process Variation**

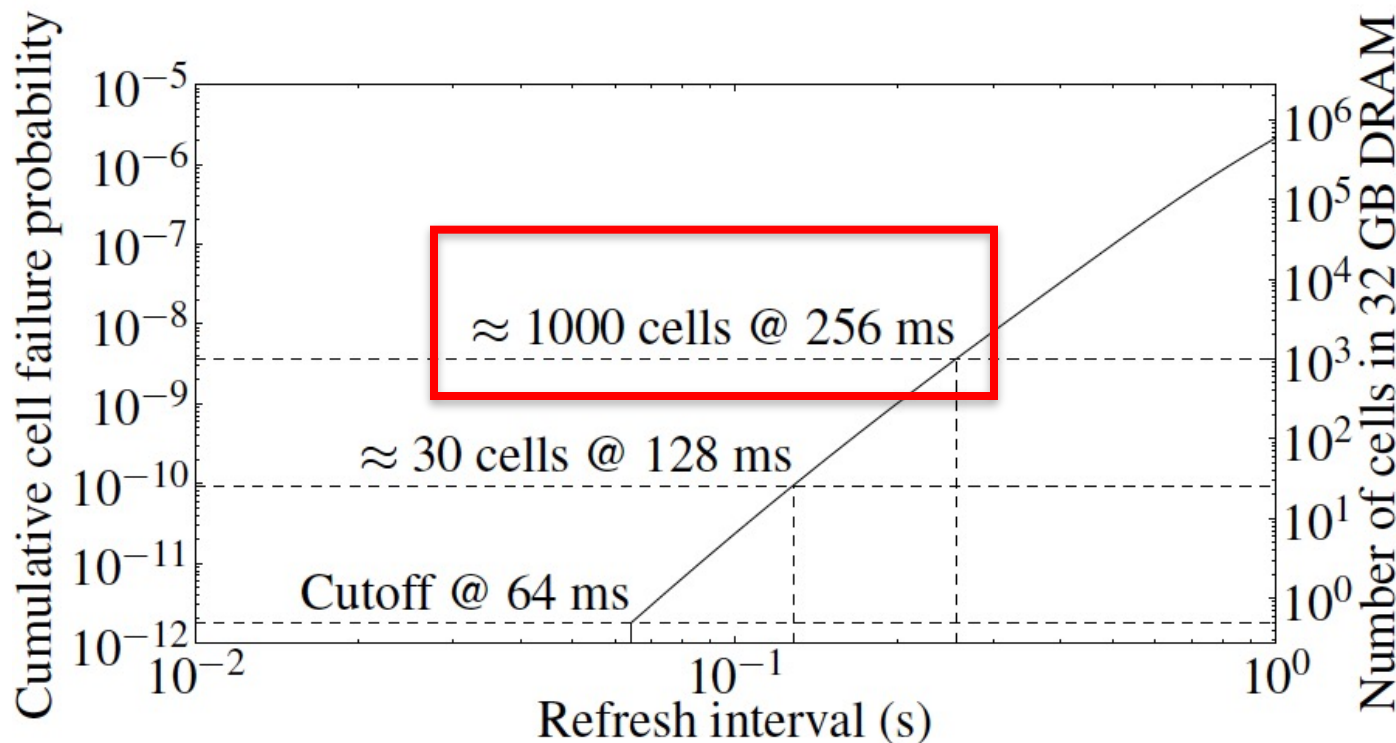
Opportunity: Taking Advantage of This Profile

- Assume we know the retention time of each row exactly
- What can we do with this information?
- Who do we expose this information to?
- How much information do we expose?
 - Affects hardware/software overhead, power consumption, verification complexity, cost
- How do we determine this profile information?
 - Also, who determines it?

Problem
Algorithm
Program/Language
Runtime System (VM, OS, MM)
ISA (Architecture)
Microarchitecture
Logic
Devices
Electrons

Retention Time of DRAM Rows

- Observation: Overwhelming majority of DRAM rows can be refreshed much less often without losing data



Key Idea of RAIDR: Refresh weak rows more frequently,
all other rows less frequently

RAIDR: Eliminating Unnecessary DRAM Refreshes

Liu, Jaiyen, Veras, Mutlu,
[RAIDR: Retention-Aware Intelligent DRAM Refresh](#)
ISCA 2012.

RAIDR: Mechanism

1. **Profiling:** Identify the retention time of all DRAM rows

64-128ms

> 256ms

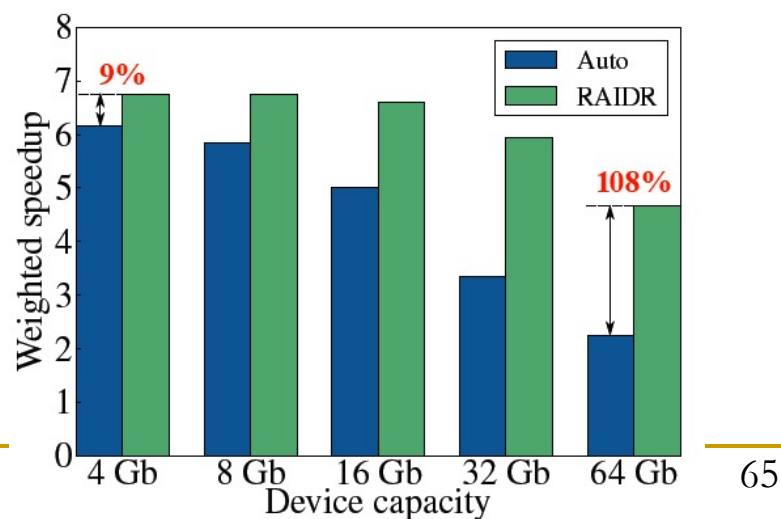
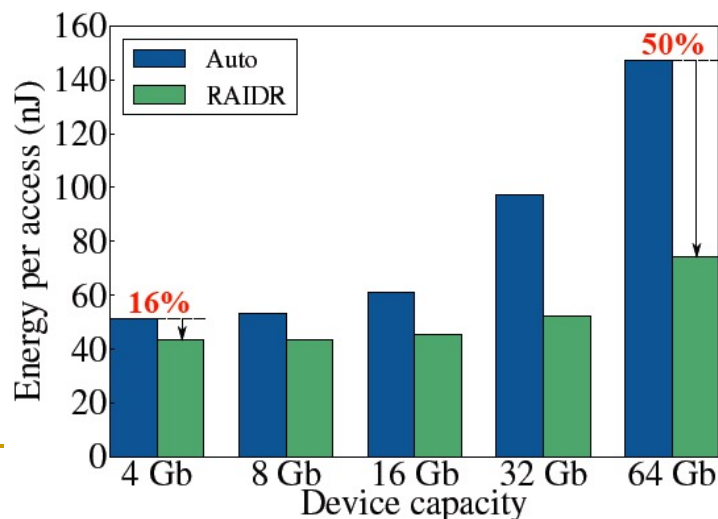
1.25KB storage in controller for 32GB DRAM memory

128-256ms

→ check the bins to determine refresh rate of a row

RAIDR: Results and Takeaways

- System: 32GB DRAM, 8-core; Various workloads
- RAIDR hardware cost: 1.25 kB (2 Bloom filters)
- Refresh reduction: 74.6%
- Dynamic DRAM energy reduction: 16%
- Idle DRAM power reduction: 20%
- Performance improvement: 9%
- Benefits increase as DRAM scales in density



Takeaway

Breaking the abstraction layers
(between components and
transformation hierarchy levels)

and knowing what is underneath

enables you to **understand** and
solve problems

Reading for the Really Interested

- Jamie Liu, Ben Jaiyen, Richard Veras, and Onur Mutlu,
"RAIDR: Retention-Aware Intelligent DRAM Refresh"
Proceedings of the 39th International Symposium on Computer Architecture
(ISCA), Portland, OR, June 2012. [Slides \(pdf\)](#)

RAIDR: Retention-Aware Intelligent DRAM Refresh

Jamie Liu Ben Jaiyen Richard Veras Onur Mutlu
Carnegie Mellon University
{jamiel,bjaiyen,rveras,onur}@cmu.edu

Really Interested? ... Further Readings

- Onur Mutlu,
"Memory Scaling: A Systems Architecture Perspective"
*Technical talk at MemCon 2013 (**MEMCON**), Santa Clara, CA, August 2013.*
Slides (pptx) (pdf) Video
- Kevin Chang, Donghyuk Lee, Zeshan Chishti, Alaa Alameldeen, Chris Wilkerson, Yoongu Kim, and Onur Mutlu,
"Improving DRAM Performance by Parallelizing Refreshes with Accesses"
*Proceedings of the 20th International Symposium on High-Performance Computer Architecture (**HPCA**), Orlando, FL, February 2014. Slides (pptx) (pdf)*

Detailed Lectures on Memory Refresh

■ Computer Architecture, Fall 2020, Lecture 2b

- Data Retention and Memory Refresh (ETH Zürich, Fall 2020)
- <https://www.youtube.com/watch?v=v702wUnaWGE&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=3>

■ Computer Architecture, Fall 2020, Lecture 3b

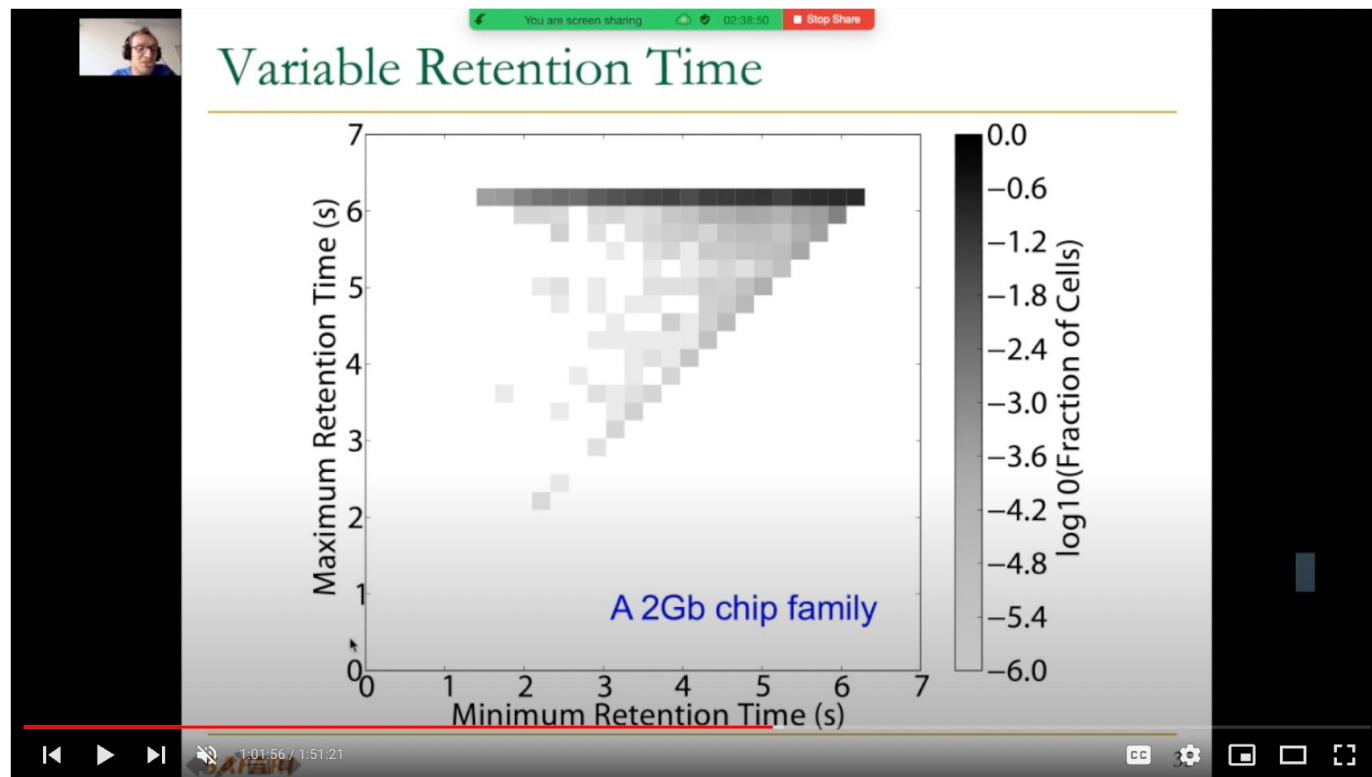
- Memory Systems: Challenges & Opportunities (ETH Zürich, Fall 2020)
- <https://www.youtube.com/watch?v=Q2FbUxD7GHs&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=6>

■ Computer Architecture, Fall 2020, Lecture 4a

- Memory Systems: Solution Directions (ETH Zürich, Fall 2020)
- <https://www.youtube.com/watch?v=PANTCVTYe8M&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=7>

Data Retention & Memory Refresh Lecture

- Computer Architecture, Fall 2020, Lecture 2b
 - Data Retention and Memory Refresh (ETH Zürich, Fall 2020)
 - <https://www.youtube.com/watch?v=v702wUnaWGE&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=3>



ETH ZÜRICH

Computer Architecture - Lecture 2b: Data Retention and Memory Refresh (ETH Zürich, Fall 2020)

3,876 views • Sep 19, 2020

48 DISLIKE SHARE SAVE ...



Onur Mutlu Lectures
22.8K subscribers

ANALYTICS

EDIT VIDEO

Four Mysteries: Familiar with Any?

- Rowhammer (2012-2014)
- Meltdown & Spectre (2017-2018)
- Memories Forget: Refresh (2011-2012)
- Memory Performance Attacks (2006-2007)

We Covered Until This Point
in the Lecture

Digital Design & Computer Arch.

Lecture 3a: Mysteries in Comp. Arch.

Prof. Onur Mutlu

ETH Zürich

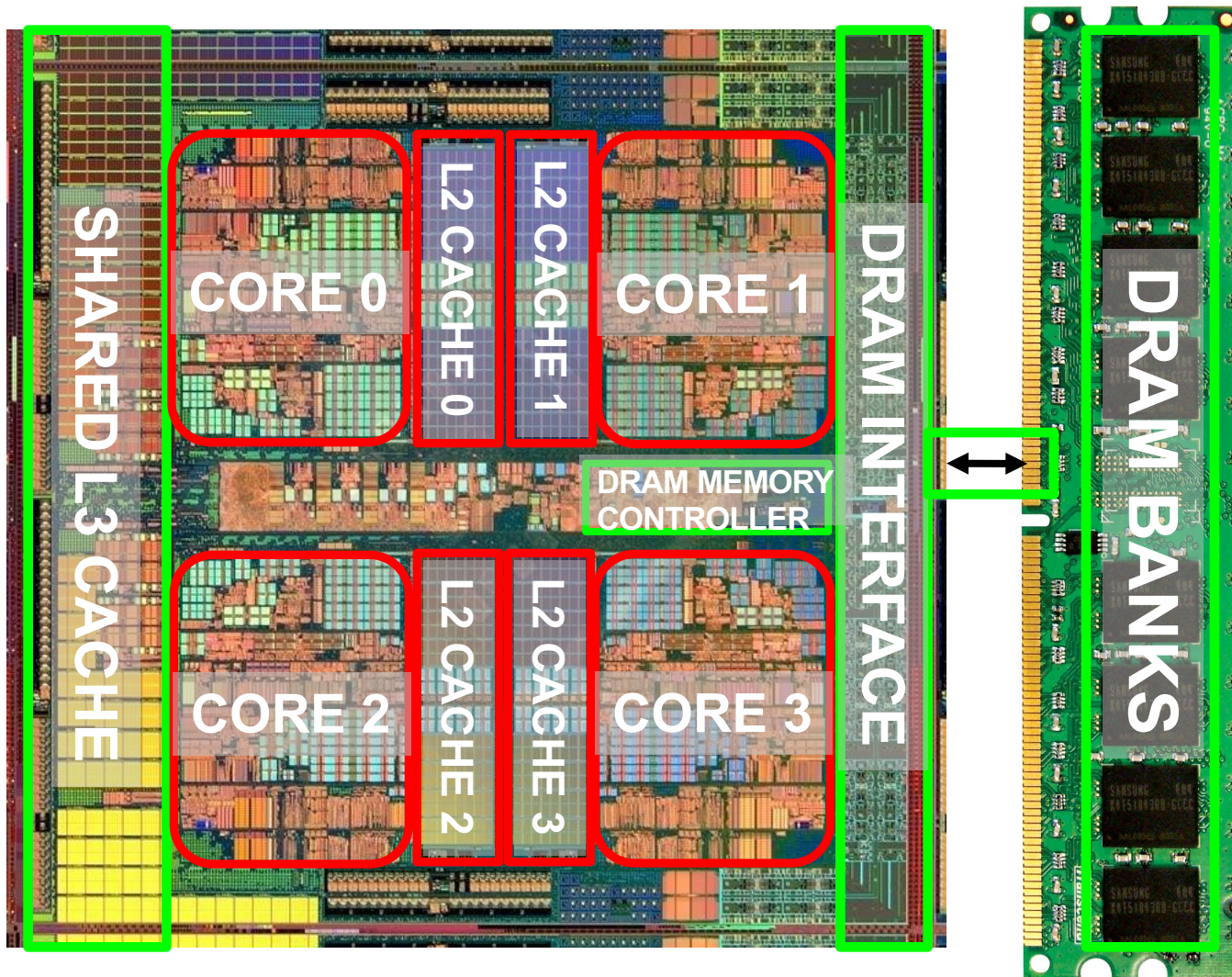
Spring 2022

3 March 2022

Mystery #4: Memory Performance Attacks

Multi-Core Systems

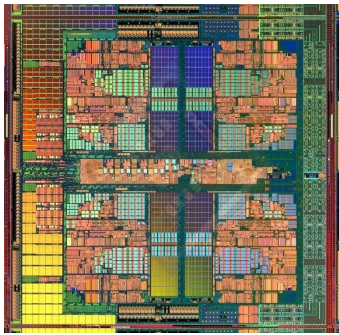
Multi-Core
Chip



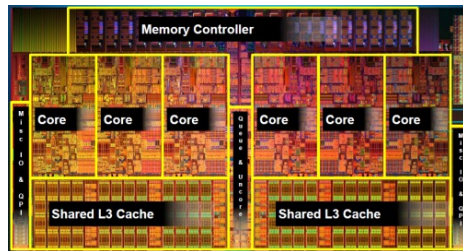
*Die photo credit: AMD Barcelona

A Trend: Many Cores on Chip

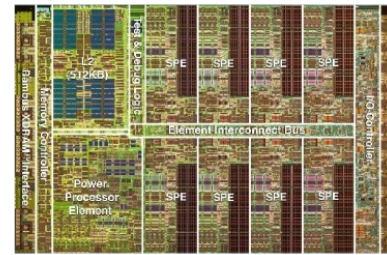
- **Simpler and lower power** than a **single large core**
- Parallel processing on single chip → faster, new applications



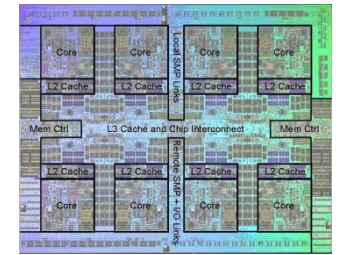
AMD Barcelona
4 cores



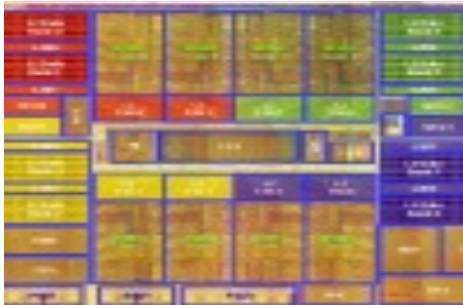
Intel Core i7
8 cores



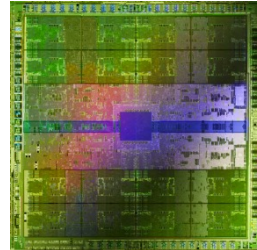
IBM Cell BE
8+1 cores



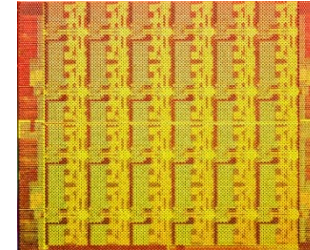
IBM POWER7
8 cores



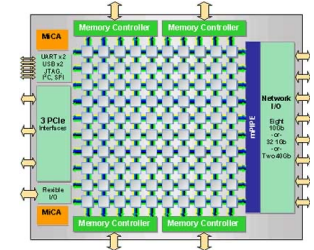
Sun Niagara II
8 cores



Nvidia Fermi
448 "cores"

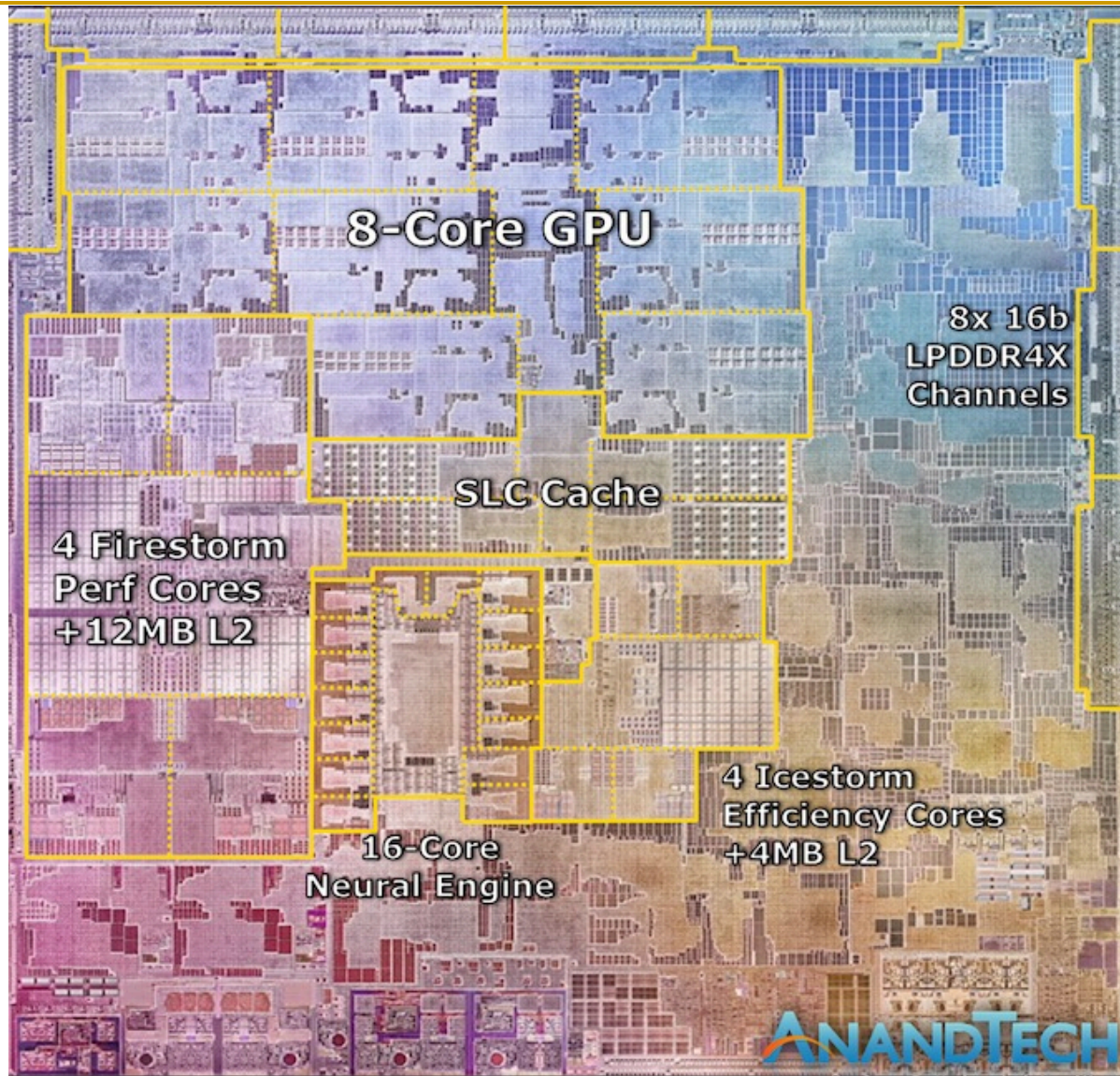


Intel SCC
48 cores, networked



Tiler TILE Gx
100 cores, networked

A Trend: Many Cores on Chip (2021)

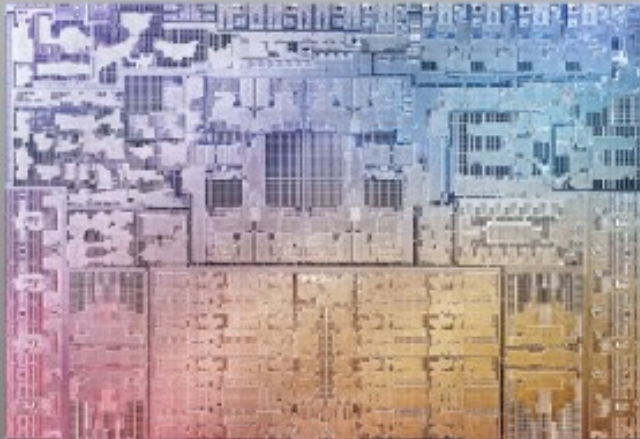


Apple M1,
2021

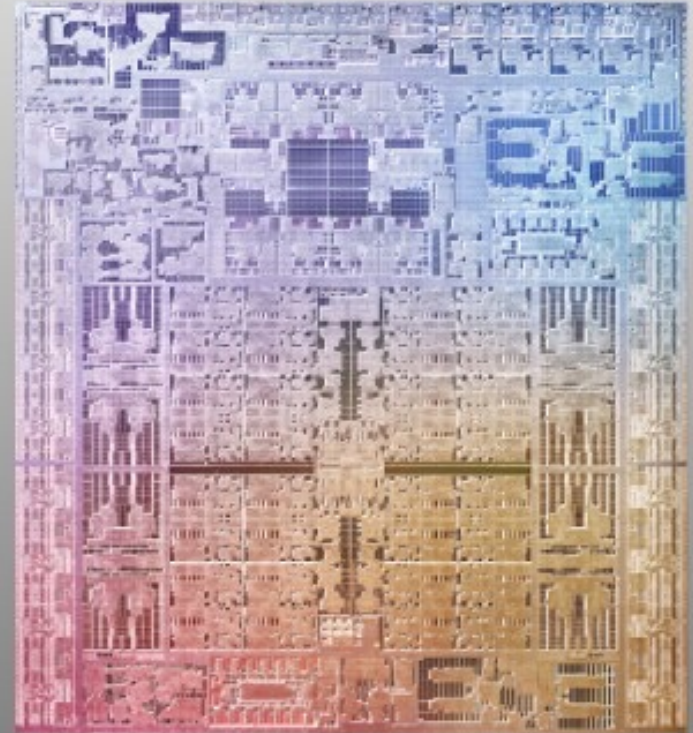
A Trend: Many Cores on Chip (2021)



Apple M1



Apple M1 Pro



Apple M1 Max

A Trend: Many Cores on Chip (2021)



10nm ESF=Intel 7 Alder Lake die shot (~209mm²) from Intel: <https://www.intel.com/content/www/us/en/newsroom/news/12th-gen-core-processors.html>

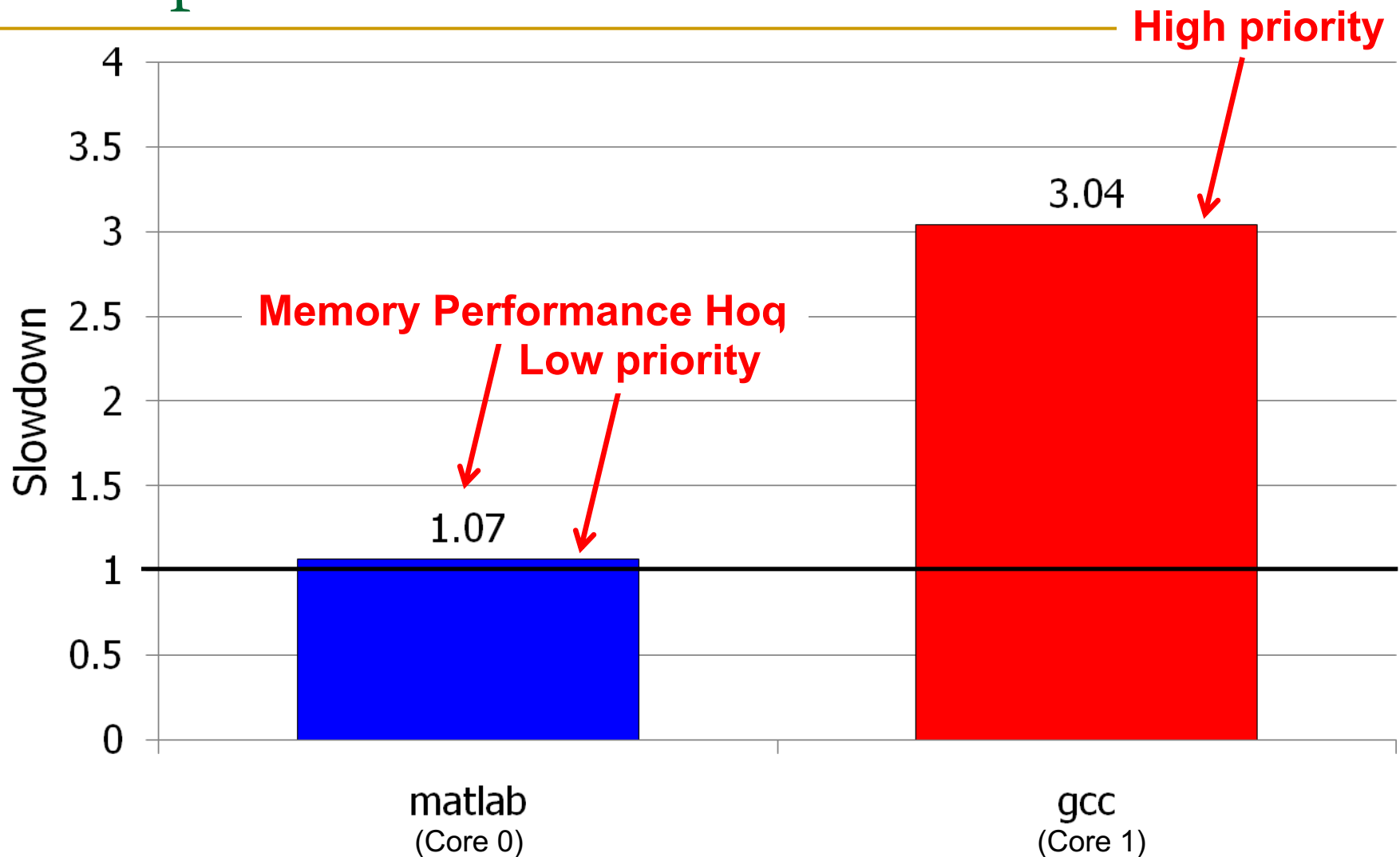
Die shot interpretation by Locuza, October 2021

Intel Alder Lake,
2021

Many Cores on Chip

- What we want:
 - N times the system performance with N times the cores
- What do we get today?

Unexpected Slowdowns in Multi-Core



Moscibroda and Mutlu, “[Memory performance attacks: Denial of memory service in multi-core systems](#),” USENIX Security 2007.

Three Questions

- Can you figure out **why the applications slow down** if you do not know the underlying system and how it works?
- Can you figure out **why there is a disparity in slowdowns** if you do not know how the system executes the programs?
- Can you **fix the problem** without knowing what is happening “underneath”?

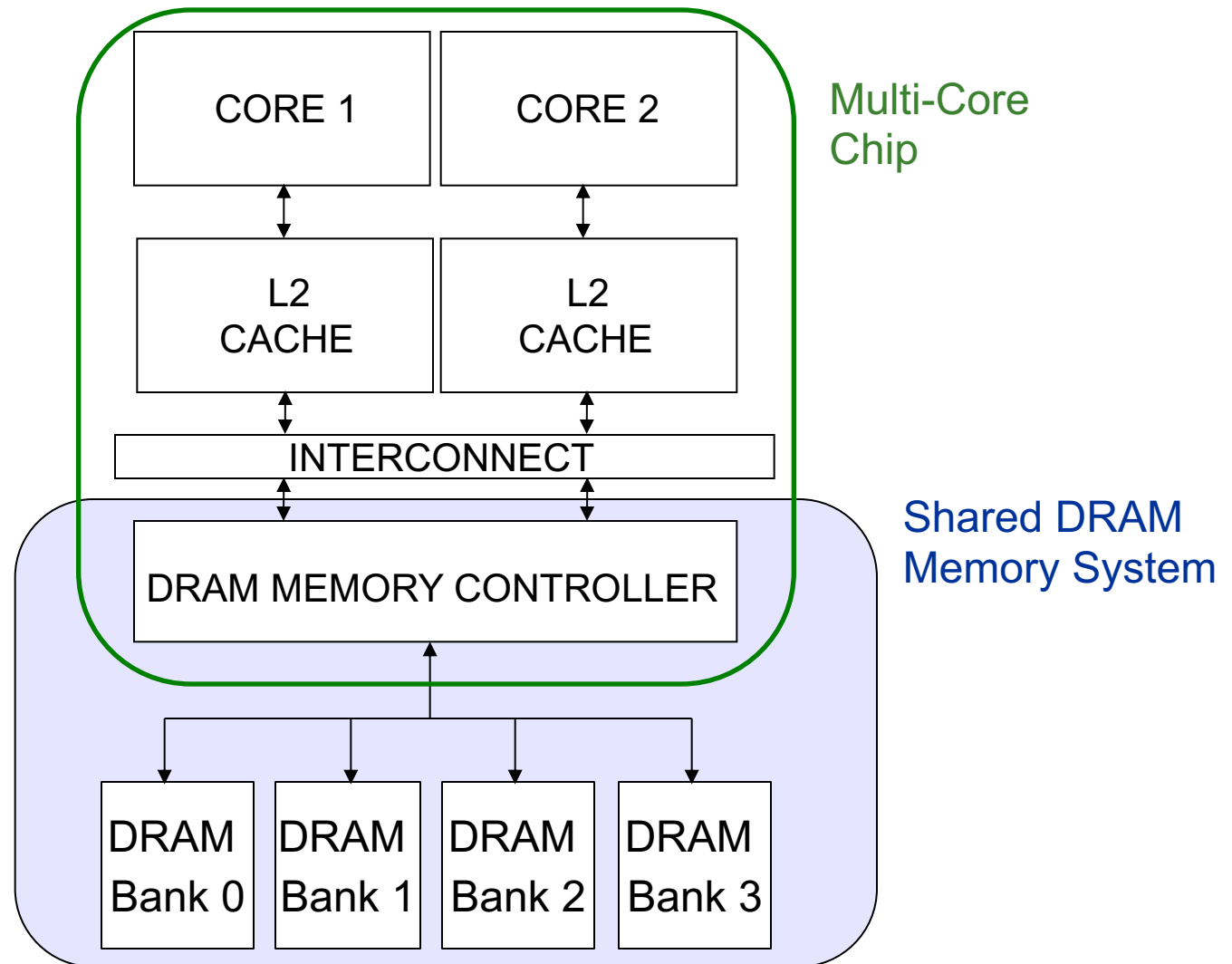
Three Questions

- Why is there any slowdown?
- Why is there a disparity in slowdowns?
- How can we solve the problem if we do not want that disparity?
 - What do we want (the system to provide)?

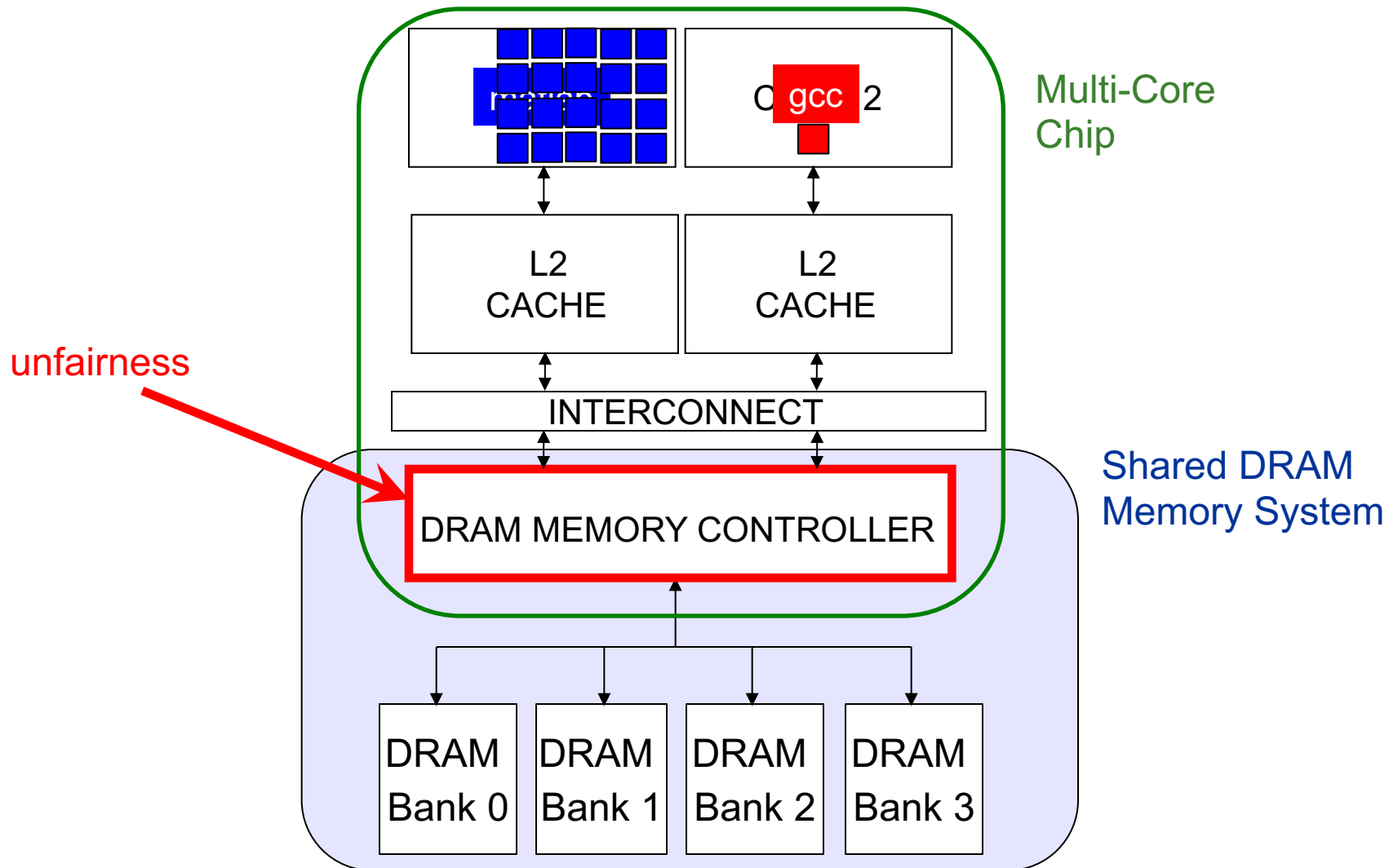
Why Is This Important?

- We want to execute applications concurrently in multi-core systems → consolidate more and more
 - Cloud computing
 - Mobile phones
- We want to mix different types of applications together
 - those requiring QoS guarantees (e.g., video, pedestrian detection)
 - those that are important but less so
 - those that are less important
- We want the system to be **controllable and high performance**

Why the Disparity in Slowdowns?



Why the Disparity in Slowdowns?



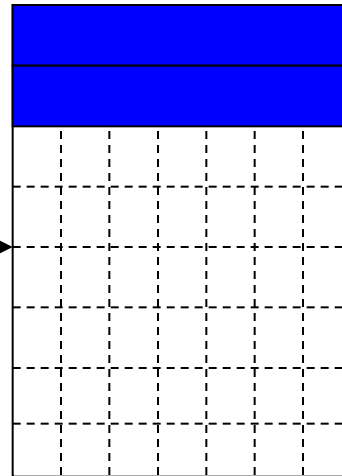
Digging Deeper: DRAM Bank Operation

Access Address:
(Row 0, Column 0)
(Row 0, Column 1)
(Row 0, Column 85)
(Row 1, Column 0)

Row address 0

Row decoder

Columns



Rows

This view of a bank is an abstraction.

Internally, a bank consists of many cells (transistors & capacitors) and other structures that enable access to cells



Row Buffer ~~CONFLICT!~~

Column address 05

Column mux

Data

DRAM Controllers

- A row-conflict memory access takes significantly longer than a row-hit access
- Current controllers take advantage of this fact
- Commonly used scheduling policy (FR-FCFS) [Rixner 2000]*
 - (1) Row-hit first: Service row-hit memory accesses first
 - (2) Oldest-first: Then service older accesses first
- This scheduling policy aims to maximize DRAM throughput

*Rixner et al., “Memory Access Scheduling,” ISCA 2000.

*Zuravleff and Robinson, “Controller for a synchronous DRAM ...,” US Patent 5,630,096, May 1997.

The Problem

- Multiple applications share the DRAM controller
- DRAM controllers designed to maximize DRAM data throughput
- DRAM scheduling policies are unfair to some applications
 - Row-hit first: unfairly prioritizes apps with high row buffer locality
 - Threads that keep on accessing the same row
 - Oldest-first: unfairly prioritizes memory-intensive applications
- DRAM controller vulnerable to denial of service attacks
 - Can write programs to exploit unfairness

A Memory Performance Hog

```
// initialize large arrays A, B

for (j=0; j<N; j++) {
    index = j*linesize; streaming
    A[index] = B[index]; (in sequence)
    ...
}
```

STREAM

- Sequential memory access
- Very high row buffer locality (96% hit rate)
- Memory intensive

```
// initialize large arrays A, B

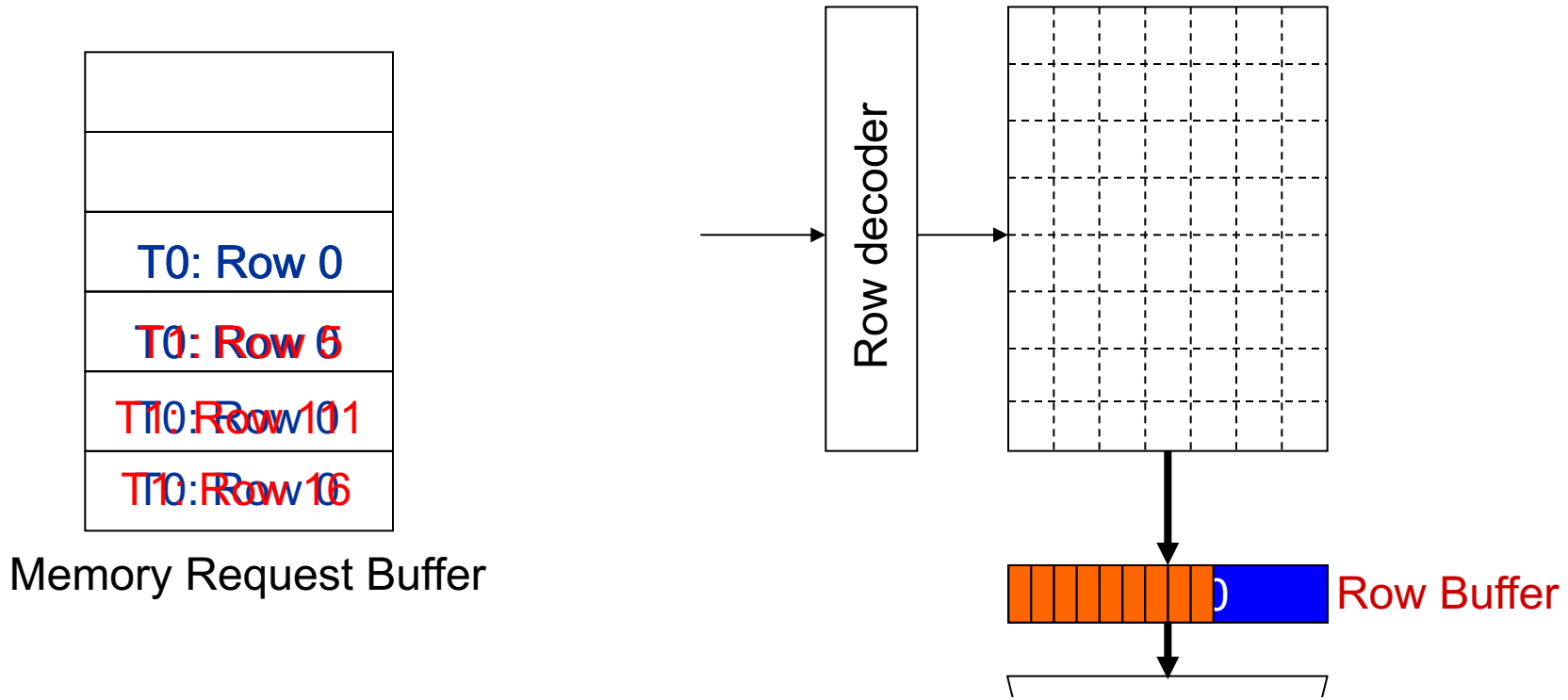
for (j=0; j<N; j++) {
    index = rand(); random
    A[index] = B[index];
    ...
}
```

RANDOM

- Random memory access
- Very low row buffer locality (3% hit rate)
- Similarly memory intensive

Moscibroda and Mutlu, “[Memory Performance Attacks](#),” USENIX Security 2007.

What Does the Memory Hog Do?



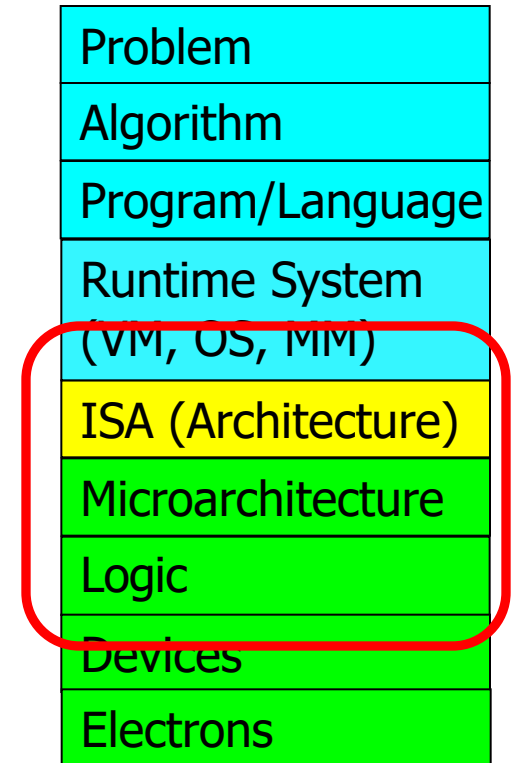
Row size: 8KB, request size: 64B

128 (8KB/64B) requests of STREAM serviced
before a single request of RANDOM

Moscibroda and Mutlu, “Memory Performance Attacks,” USENIX Security 2007.

Now That We Know What Happens Underneath

- How would you solve the problem?
- What is the right place to solve the problem?
 - Programmer?
 - System software?
 - Compiler?
 - Hardware (Memory controller)?
 - Hardware (DRAM)?
 - Circuits?
- Two major goals of this course:
 - Enable you to **think critically**
 - Enable you to **think broadly**



For the Really Interested...

- Thomas Moscibroda and Onur Mutlu,
"Memory Performance Attacks: Denial of Memory Service in Multi-Core Systems"
*Proceedings of the 16th USENIX Security Symposium (**USENIX SECURITY**),*
pages 257-274, Boston, MA, August 2007. [Slides \(ppt\)](#)

Memory Performance Attacks: Denial of Memory Service in Multi-Core Systems

Thomas Moscibroda Onur Mutlu
Microsoft Research
{moscitho,onur}@microsoft.com

Really Interested? ... Further Readings

- Onur Mutlu and Thomas Moscibroda,
"Stall-Time Fair Memory Access Scheduling for Chip Multiprocessors"
Proceedings of the 40th International Symposium on Microarchitecture (MICRO), pages 146-158, Chicago, IL, December 2007. [Slides \(ppt\)](#)
- Onur Mutlu and Thomas Moscibroda,
"Parallelism-Aware Batch Scheduling: Enhancing both Performance and Fairness of Shared DRAM Systems"
Proceedings of the 35th International Symposium on Computer Architecture (ISCA) [[Slides \(ppt\)](#)]
- Sai Prashanth Muralidhara, Lavanya Subramanian, Onur Mutlu, Mahmut Kandemir, and Thomas Moscibroda,
"Reducing Memory Interference in Multicore Systems via Application-Aware Memory Channel Partitioning"
Proceedings of the 44th International Symposium on Microarchitecture (MICRO), Porto Alegre, Brazil, December 2011. [Slides \(pptx\)](#)

Detailed Lectures on Memory Controllers

- **Computer Architecture, Fall 2020, Lecture 2a**
 - ❑ Memory Performance Attacks (ETH Zürich, Fall 2020)
 - ❑ <https://www.youtube.com/watch?v=VJzZbwgBfy8&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=2>
- **Computer Architecture, Fall 2020, Lecture 11a**
 - ❑ Memory Controllers (ETH Zürich, Fall 2020)
 - ❑ <https://www.youtube.com/watch?v=TeG773OgiMQ&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=20>
- **Computer Architecture, Fall 2020, Lecture 11b**
 - ❑ Memory Interference and QoS (ETH Zürich, Fall 2020)
 - ❑ <https://www.youtube.com/watch?v=0nnI807nCkc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=21>
- **Computer Architecture, Fall 2020, Lecture 13**
 - ❑ Memory Interference and QoS II (ETH Zürich, Fall 2020)
 - ❑ <https://www.youtube.com/watch?v=Axye9VqQT7w&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=26>

Memory Performance Attacks Lecture ...

- Computer Architecture, Fall 2020, Lecture 2a
 - Memory Performance Attacks (ETH Zürich, Fall 2020)
 - <https://www.youtube.com/watch?v=VJzZbwgBfy8&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=2>

You are screen sharing 01:18:36 Stop Share

Distributed DoS in Networked Multi-Core Systems

Attackers (Cores 1-8) Stock option pricing application (Cores 9-64)

Cores connected via packet-switched routers on chip

Grot, Hestness, Keckler, Mutlu, "Preemptive virtual clock: A Flexible, Efficient, and Cost-effective QoS Scheme for Networks-on-Chip," MICRO 2009.

1:06:47 / 1:13:48 • Reading on Memory Performance Attacks

ETH ZÜRICH

Computer Architecture - Lecture 2a: Memory Performance Attacks (ETH Zürich, Fall 2020)

5,903 views • Sep 19, 2020

101 DISLIKE SHARE SAVE ...

SAFARI



Onur Mutlu Lectures
22.8K subscribers

ANALYTICS

EDIT VIDEO

Takeaway I

Breaking the abstraction layers
(between components and
transformation hierarchy levels)
and knowing what is underneath
enables you to **understand** and
solve problems

Takeaway II

Cooperation between
multiple components and layers
can enable
more effective
solutions and systems

Recap: Mysteries No Longer!

- Rowhammer (2012-2014)
- Meltdown & Spectre (2017-2018)
- Memories Forget: Refresh (2011-2012)
- Memory Performance Attacks (2006-2007)

Takeaways

Takeaways

- It is an exciting time to be understanding and designing computing architectures
- Many challenging and exciting problems in platform design
 - That no one has tackled (or thought about) before
 - That can have huge impact on the world's future
- Driven by huge hunger for data (Big Data), new applications (ML/AI, graph analytics, genomics), ever-greater realism, ...
 - We can easily collect more data than we can analyze/understand
- Driven by significant difficulties in keeping up with that hunger at the technology layer
 - Five walls: Energy, reliability, complexity, security, scalability

Computer Architecture as an Enabler of the Future

Assignment: Required Lecture Video

- Why study computer architecture? Why is it important?
- Future Computing Platforms: Challenges & Opportunities
- **Required Assignment**
 - ❑ **Watch one of** Prof. Mutlu's lectures and analyze either (or both)
 - ❑ <https://www.youtube.com/watch?v=kgiZISOcGFM> (May 2017)
 - ❑ <https://www.youtube.com/watch?v=mskTeNnf-i0> (Feb 2021)
- **Optional Assignment – for 1% extra credit**
 - ❑ **Write a 1-page summary** of one of the lectures and email us
 - What are your key takeaways?
 - What did you learn?
 - What did you like or dislike?
 - Submit your summary to [Moodle](#)

Bloom Filters

Approximate Set Membership

- Suppose you want to quickly find out:
 - whether an element belongs to a set
- And, you can tolerate mistakes of the sort:
 - The element is actually **not** in the set, but you are incorrectly told that it is → false positive
- But, you cannot tolerate mistakes of the sort:
 - The element is actually in the set, but you are incorrectly told that it is **not** → false negative
- Example task: You want to quickly identify all Mobile Phone Model X owners among all possible people in the world
 - Perhaps you want to give them free replacement phones

Example Task

- World population
 - ~8 billion (and growing)
 - 1 bit per person to indicate Model X owner or not
 - 2^{33} bits needed to represent the entire set accurately
 - 8 Gigabits → large storage cost, slow access
- Mobile Phone Model X owner population
 - Say 1 million (and growing)
- Can we represent the Model X owner set approximately, using a much smaller number of bits?
 - Record the ID's of owners in a much smaller Bloom Filter

Example Task II

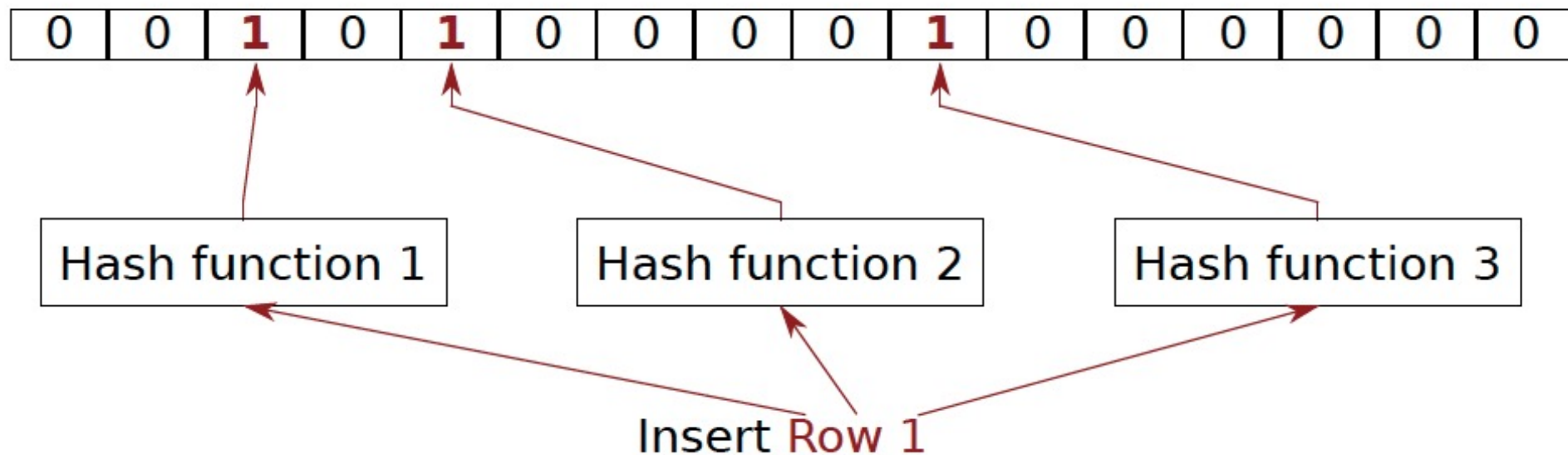
- DRAM row population
 - ~8 billion (and growing)
 - 1 bit per row to indicate Refresh-often or not
 - 2^{33} bits needed to represent the entire set accurately
 - 8 Gigabits → large storage cost, slow access
- Refresh-often population
 - Say 1 million
- Can we represent Refresh-often set approximately, using a much smaller number of bits?
 - Record the ID's of Refresh-Often rows in a much smaller Bloom Filter

Bloom Filter

- [Bloom, CACM 1970]
- Probabilistic data structure that compactly represents set membership (presence or absence of element in a set)
- Non-approximate set membership: Use 1 bit per element to indicate absence/presence of each element from an element space of N elements
- Approximate set membership: use a much smaller number of bits and indicate each element's presence/absence with a subset of those bits
 - Some elements map to the bits other elements also map to
- Operations: 1) insert, 2) test, 3) remove all elements

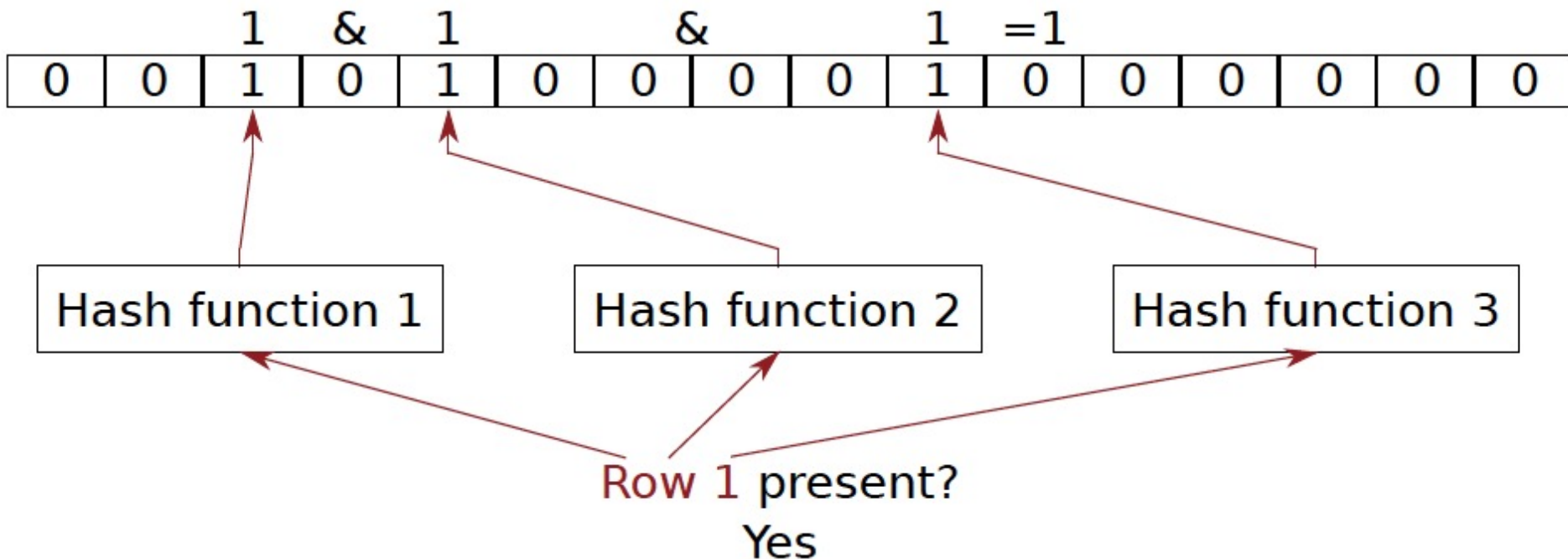
Bloom Filter Operation Example

Example with 64-128ms bin:



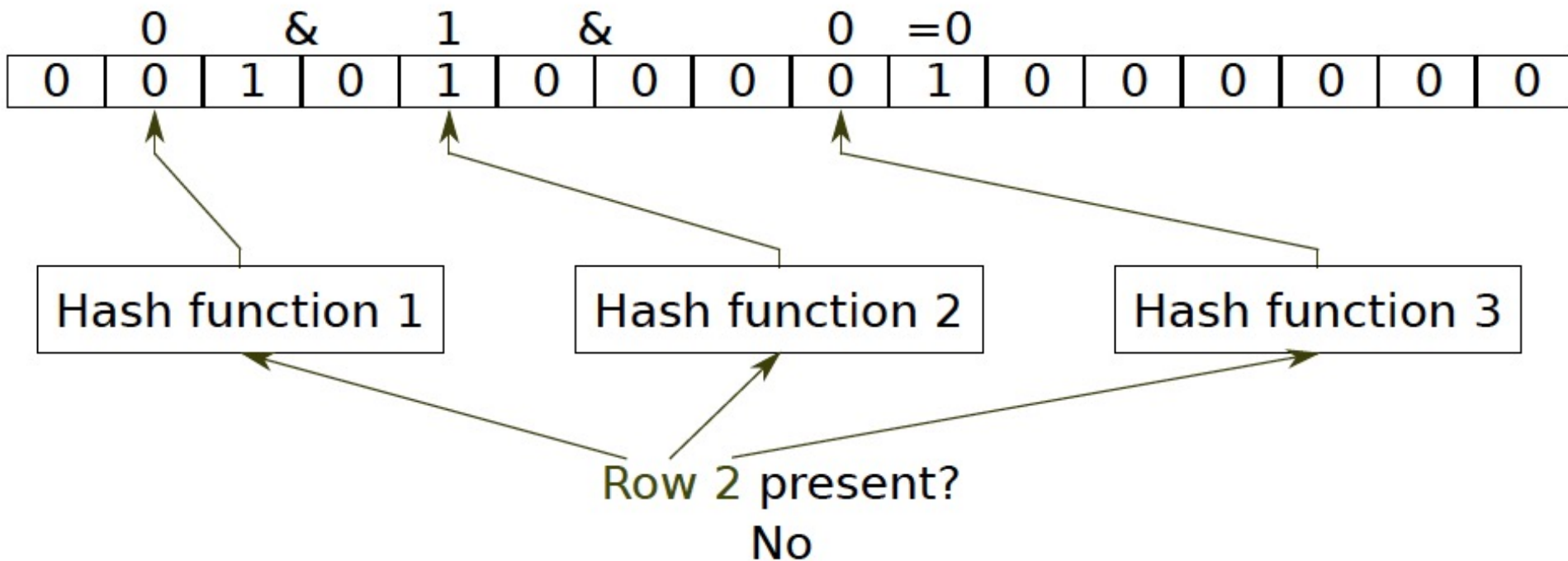
Bloom Filter Operation Example

Example with 64-128ms bin:



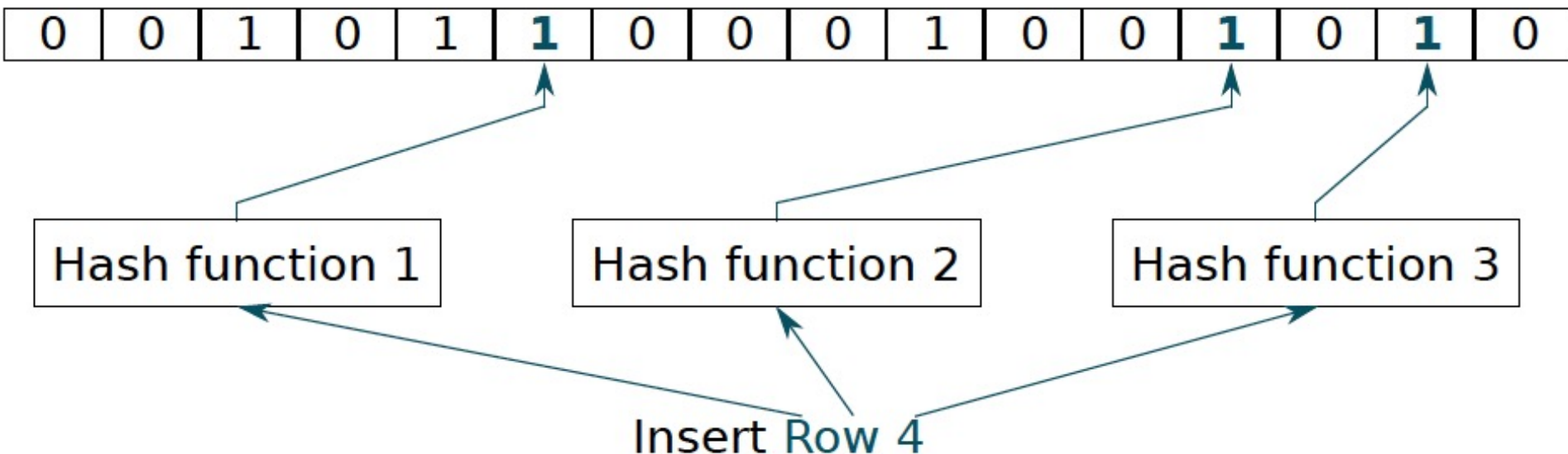
Bloom Filter Operation Example

Example with 64-128ms bin:



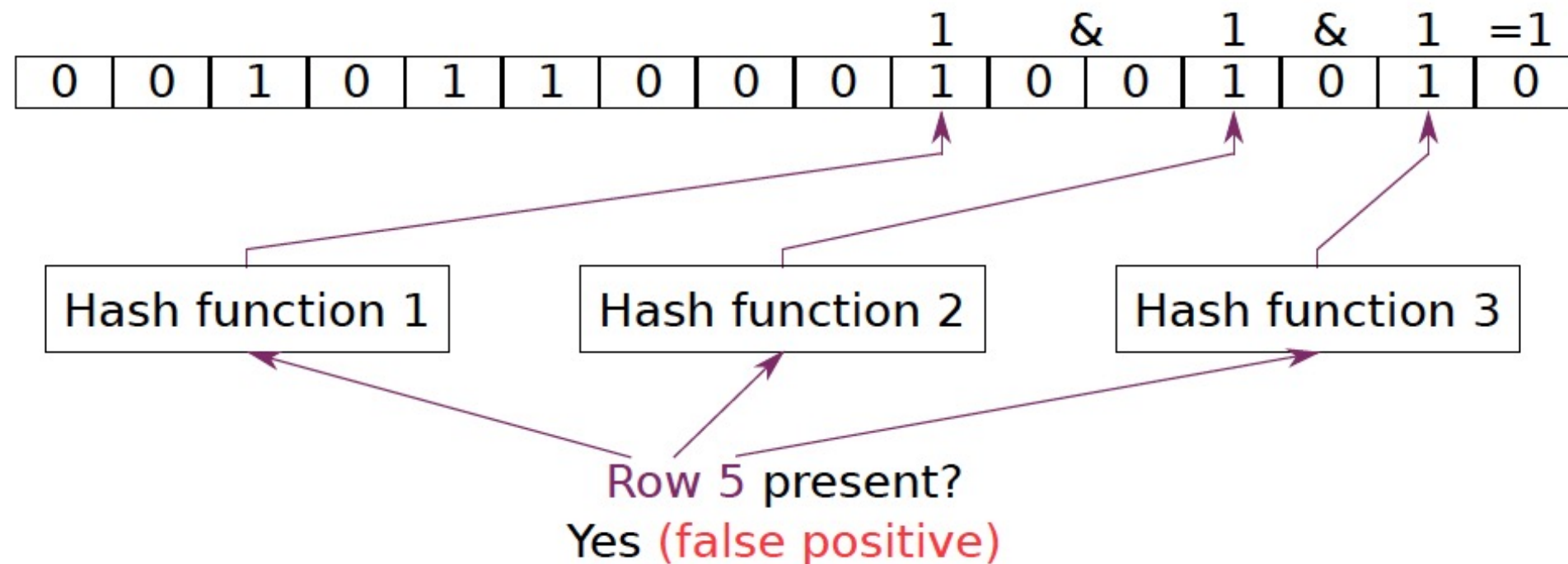
Bloom Filter Operation Example

Example with 64-128ms bin:



Bloom Filter Operation Example

Example with 64–128ms bin:



Bloom Filters

Space/Time Trade-offs in Hash Coding with Allowable Errors

BURTON H. BLOOM

Computer Usage Company, Newton Upper Falls, Mass.

In such applications, it is envisaged that overall performance could be improved by using a smaller core resident hash area in conjunction with the new methods and, when necessary, by using some secondary and perhaps time-consuming test to "catch" the small fraction of errors associated with the new methods. An example is discussed which illustrates possible areas of application for the new methods.

In this paper trade-offs among certain computational factors in hash coding are analyzed. The paradigm problem considered is that of testing a series of messages one-by-one for membership in a given set of messages. Two new hash-coding methods are examined and compared with a particular conventional hash-coding method. The computational factors considered are the size of the hash area (space), the time required to identify a message as a nonmember of the given set (reject time), and an allowable error frequency.

Bloom Filters: Pros and Cons

■ Advantages

- + Enables **storage-efficient** representation of set membership
- + Insertion and testing for set membership (presence) are **fast**
- + **No false negatives**: If Bloom Filter says an element is not present in the set, the element must not have been inserted
- + Enables **tradeoffs** between **time** & **storage efficiency** & **false positive rate** (via sizing and hashing)

■ Disadvantages

- **False positives**: An element may be deemed to be present in the set by the Bloom Filter but it may never have been inserted

Not the right data structure when you cannot tolerate false positives

Benefits of Bloom Filters as Refresh Rate Bins

- **False positives:** a row may be declared present in the Bloom filter even if it was never inserted
 - **Not a problem:** Refresh some rows more frequently than needed
- **No false negatives:** rows are never refreshed less frequently than needed (no correctness problems)
- **Scalable:** a Bloom filter never overflows (unlike a fixed-size table)
- **Efficient:** No need to store info on a per-row basis; simple hardware → 1.25 KB for 2 filters for 32 GB DRAM system

Use of Bloom Filters in Hardware

- Useful & cost-effective when you can tolerate false positives in set membership tests
- See the following recent examples for clear descriptions of how Bloom Filters are used
 - Liu et al., “[RAIDR: Retention-Aware Intelligent DRAM Refresh](#),” ISCA 2012.
 - Seshadri et al., “[The Evicted-Address Filter: A Unified Mechanism to Address Both Cache Pollution and Thrashing](#),” PACT 2012.
 - Yaglikci et al., “[BlockHammer: Preventing RowHammer at Low Cost by Blacklisting Rapidly-Accessed DRAM Rows](#),” HPCA 2021.

Bloom Filters for Eliminating Refresh

- Jamie Liu, Ben Jaiyen, Richard Veras, and Onur Mutlu,
"RAIDR: Retention-Aware Intelligent DRAM Refresh"
Proceedings of the 39th International Symposium on Computer Architecture (ISCA), Portland, OR, June 2012. [Slides \(pdf\)](#)

RAIDR: Retention-Aware Intelligent DRAM Refresh

Jamie Liu Ben Jaiyen Richard Veras Onur Mutlu
Carnegie Mellon University
{jamel, bjaiyen, rveras, onur}@cmu.edu

Bloom Filters for Better Cache Performance

- Vivek Seshadri, Onur Mutlu, Michael A. Kozuch, and Todd C. Mowry, **"The Evicted-Address Filter: A Unified Mechanism to Address Both Cache Pollution and Thrashing"**
Proceedings of the 21st International Conference on Parallel Architectures and Compilation Techniques (PACT), Minneapolis, MN, September 2012. [Slides \(pptx\)](#) [Source Code](#)

The Evicted-Address Filter: A Unified Mechanism to Address Both Cache Pollution and Thrashing

Vivek Seshadri[†]
vseshadr@cs.cmu.edu

Onur Mutlu[†]
onur@cmu.edu

Michael A Kozuch^{*}
michael.a.kozuch@intel.com

Todd C Mowry[†]
tcm@cs.cmu.edu

[†]Carnegie Mellon University

^{*}Intel Labs Pittsburgh

Bloom Filters for Solving RowHammer

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

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

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](#) (22 minutes)]

[[Short Talk Video](#) (7 minutes)]

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

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

¹*ETH Zürich*

²*University of Illinois at Urbana–Champaign*

Mystery #2:

Meltdown & Spectre

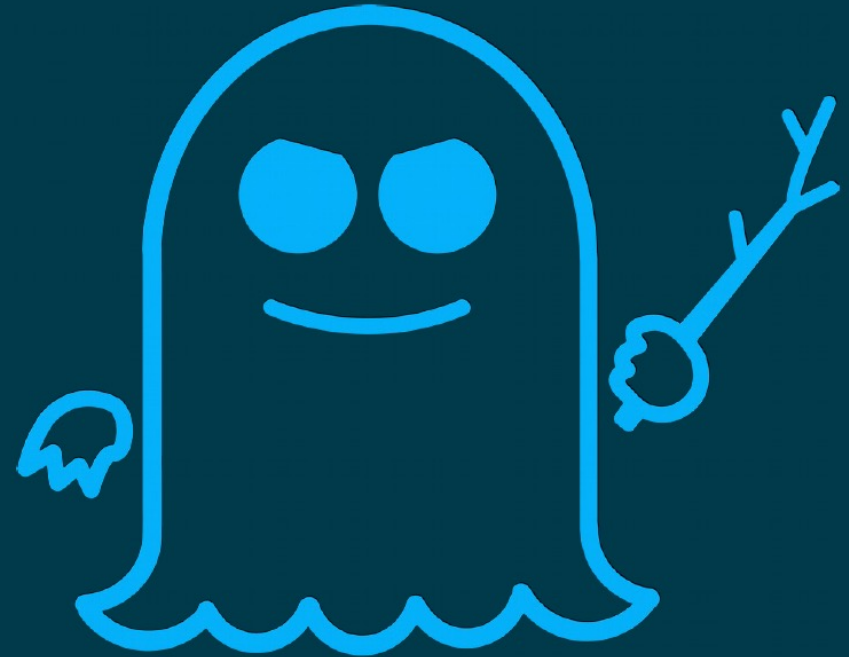
Four Mysteries: Familiar with Any?

- Rowhammer (2012-2014)
- Meltdown & Spectre (2017-2018)
- Memories Forget: Refresh (2011-2012)
- Memory Performance Attacks (2006-2007)

What Are These?



MELTDOWN



SPECTRE

Meltdown and Spectre Attacks

- Someone can steal secret data from the system even though
 - your program and data are perfectly correct and
 - your hardware behaves according to the specification and
 - there are no software vulnerabilities/bugs

Meltdown and Spectre

- Hardware security vulnerabilities that essentially effect almost all computer chips that were manufactured in the past two decades
- They exploit “speculative execution”
 - A technique employed in modern processors for high performance
- **Speculative execution:** Doing something before you know that it is needed
 - We do it all the time in life, to save time & be faster
 - Guess what will happen and act based on that guess
 - Processors do it, too, to run programs fast
 - They guess and execute code before they know it should be executed

Speculative Execution (I)

- Modern processors “speculatively execute” code to improve performance:

```
if (account-balance <= 0) {  
    // do something  
} else if (account-balance < 1M) {  
    // do something else  
} else {  
    // do something else  
}
```

Guess what code will be executed and execute it speculatively

- Improves performance, if it takes a long time to access account-balance

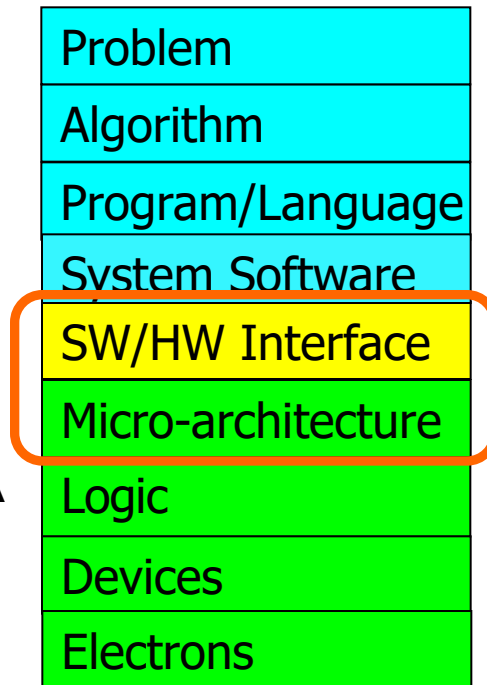
If the guess was wrong, flush the wrong instructions and execute the correct code

Speculative Execution is Invisible to the User

ISA (Instruction Set Architecture)

Interface/contract between
SW and HW.

What the programmer
assumes hardware will
satisfy.



Programmer assumes their code
will be executed in sequential order

Microarchitecture

An implementation of the ISA

Microarchitecture executes
instructions in a different order,
speculatively – but reports the results
as expected by the programmer

Meltdown and Spectre

- Someone can steal secret data from the system even though
 - ❑ your program and data are perfectly correct and
 - ❑ your hardware behaves according to the specification and
 - ❑ there are no software vulnerabilities/bugs

- Why?
 - ❑ Speculative execution leaves traces of secret data in the processor's cache (internal storage)
 - It brings data that is not supposed to be brought/accessed if there was no speculative execution
 - ❑ A malicious program can inspect the contents of the cache to "infer" secret data that it is not supposed to access
 - ❑ A malicious program can actually force another program to speculatively execute code that leaves traces of secret data

Processor Cache as a Side Channel

- Speculative execution leaves traces of data in processor cache
 - ❑ **Architecturally correct behavior w.r.t. specification**
 - ❑ However, **this leads to a side channel**: a channel through which someone sophisticated can extract information
- Processor cache leaks information by storing speculatively-accessed data
 - ❑ A clever attacker can probe the cache and infer the secret data values
 - by measuring how long it takes to access the data
 - ❑ A clever attacker can force a program to speculatively execute code and leave traces of secret data in the cache

More on Meltdown/Spectre Side Channels

Project Zero

News and updates from the Project Zero team at Google

Wednesday, January 3, 2018

Reading privileged memory with a side-channel

Posted by Jann Horn, Project Zero

We have discovered that CPU data cache timing can be abused to efficiently leak information out of mis-speculated execution, leading to (at worst) arbitrary virtual memory read vulnerabilities across local security boundaries in various contexts.

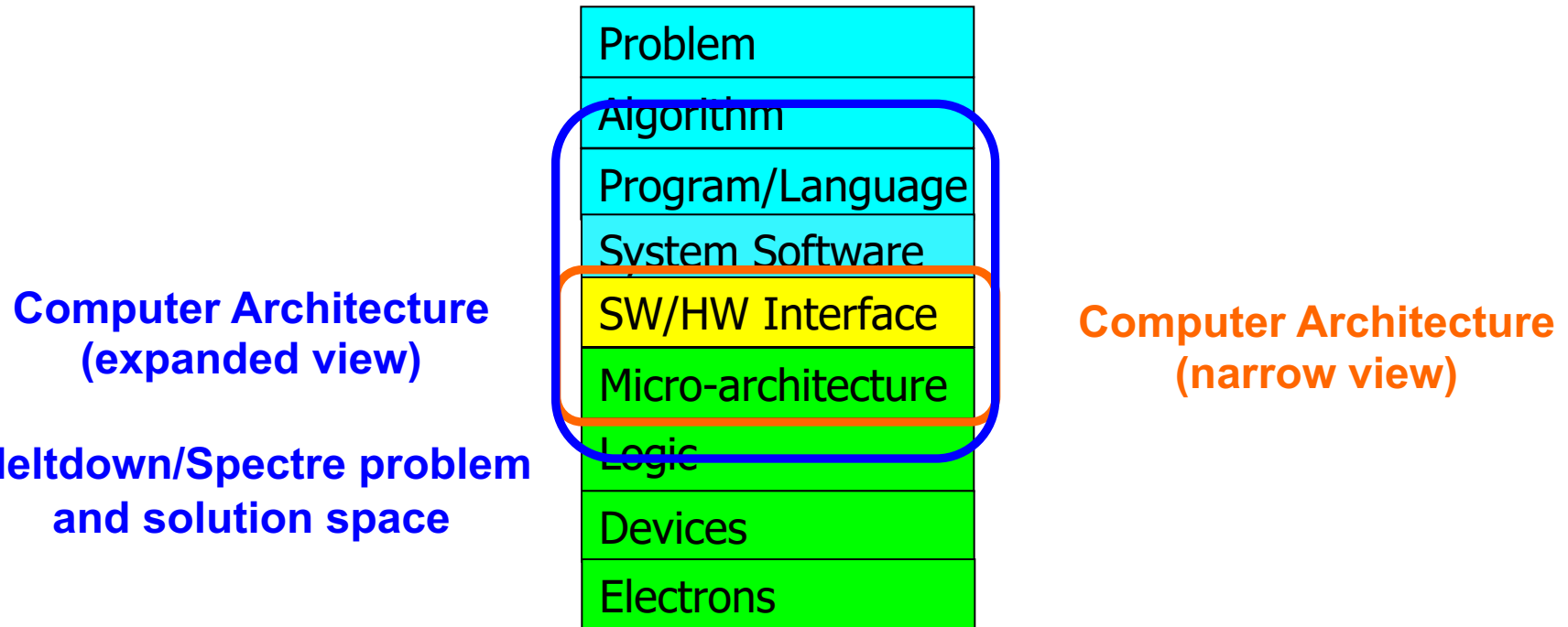
Three Questions

- Can you figure out **why someone stole your secret data** if you do not know how the processor executes a program?
- Can you **fix the problem** without knowing what is happening “underneath”, i.e., inside the microarchitecture?
- Can you **fix the problem well/fundamentally** without knowing both software and hardware design?
- Can you **construct this attack or similar attacks** without knowing what is happening underneath?

Three Other Questions

- What are the causes of Meltdown and Spectre?
- How can we prevent them (while keeping the performance benefits of speculative execution)?
 - Software changes?
 - Operating system changes?
 - Instruction set architecture changes?
 - Microarchitecture/hardware changes?
 - Changes at multiple layers, done cooperatively?
 - ...
- How do we design high-performance processors that do not leak information via side channels?

Meltdown/Spectre Span Across the Hierarchy



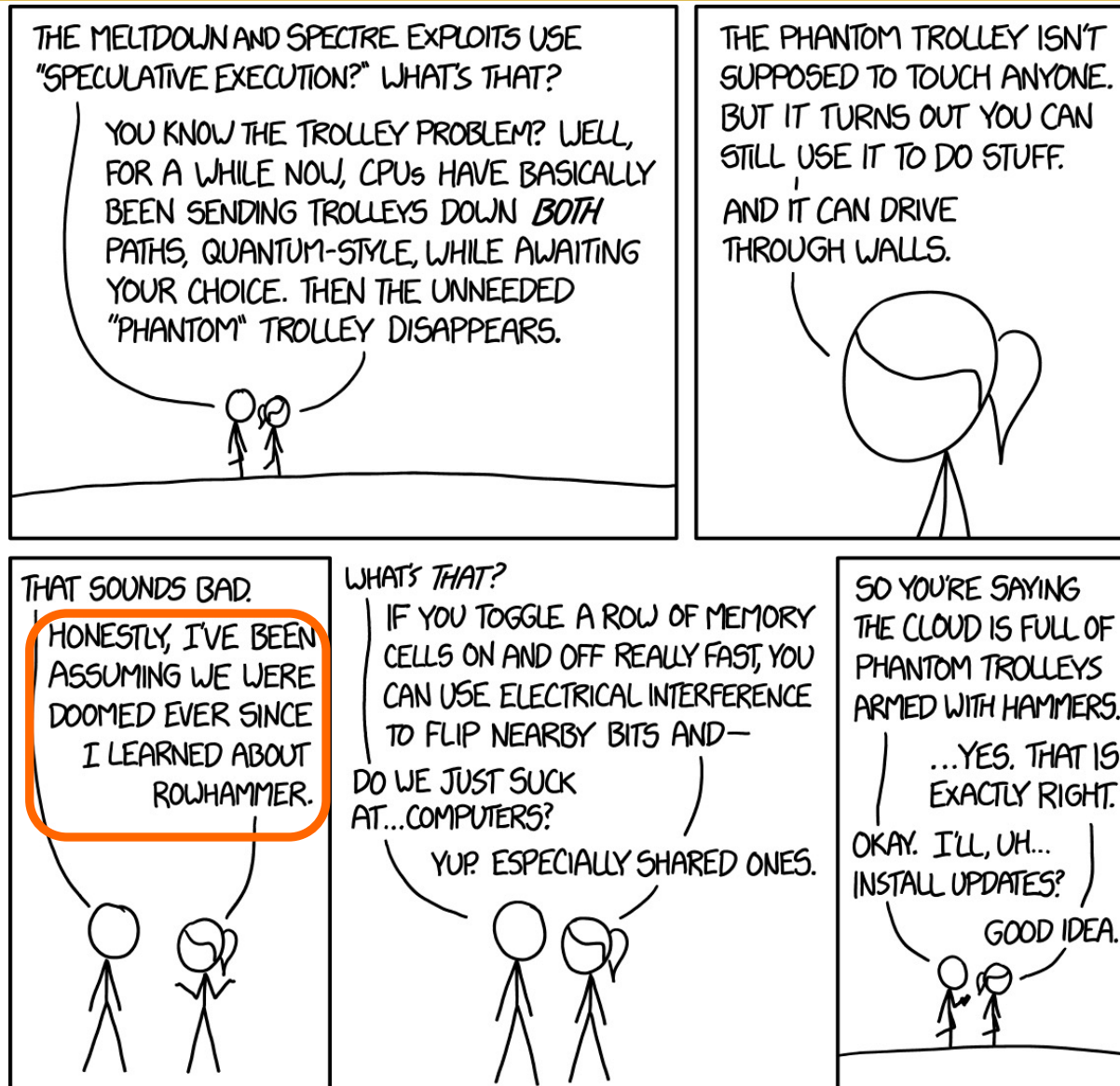
Takeaway

Breaking the abstraction layers
(between components and
transformation hierarchy levels)

and knowing what is underneath

enables you to **understand** and
solve problems

... and Also Understand/Critique Cartoons!



An Important Note: Design Goal and Mindset

- Design goal of a system determines the design mindset and evaluation metrics
- Meltdown and Spectre are there because the design goal of cutting-edge processors (employed everywhere in our lives)
 - has mainly been focused on **high performance and low energy** (relatively recently)
 - has **not included security** (or information leakage) as an important constraint
- Incorporating security as a first-class constraint and “metric” into (hardware) design and education is critical in today’s world

Design Mindset



Security is about preventing unforeseen consequences

Two Other Goals of This Course

- ❑ Enable you to think critically
- ❑ Enable you to think broadly

To Learn and Discover Further

- High-level Video by RedHat
 - <https://www.youtube.com/watch?v=syAdX44pokE>
- A bit lower-level, comprehensive explanation by Y. Vigfusson
 - <https://www.youtube.com/watch?v=mgAN4w7LH2o>
- Keep attending lectures and taking in all the material
- Talk with me & the TAs in the future
 - I have many bachelor's/master's projects on hardware security
 - “Fundamentally secure computing architectures” is a key direction of scientific investigation and design