Computer Architecture

Lecture 6: The Story of RowHammer

Memory Security & Reliability

Prof. Onur Mutlu

ETH Zürich

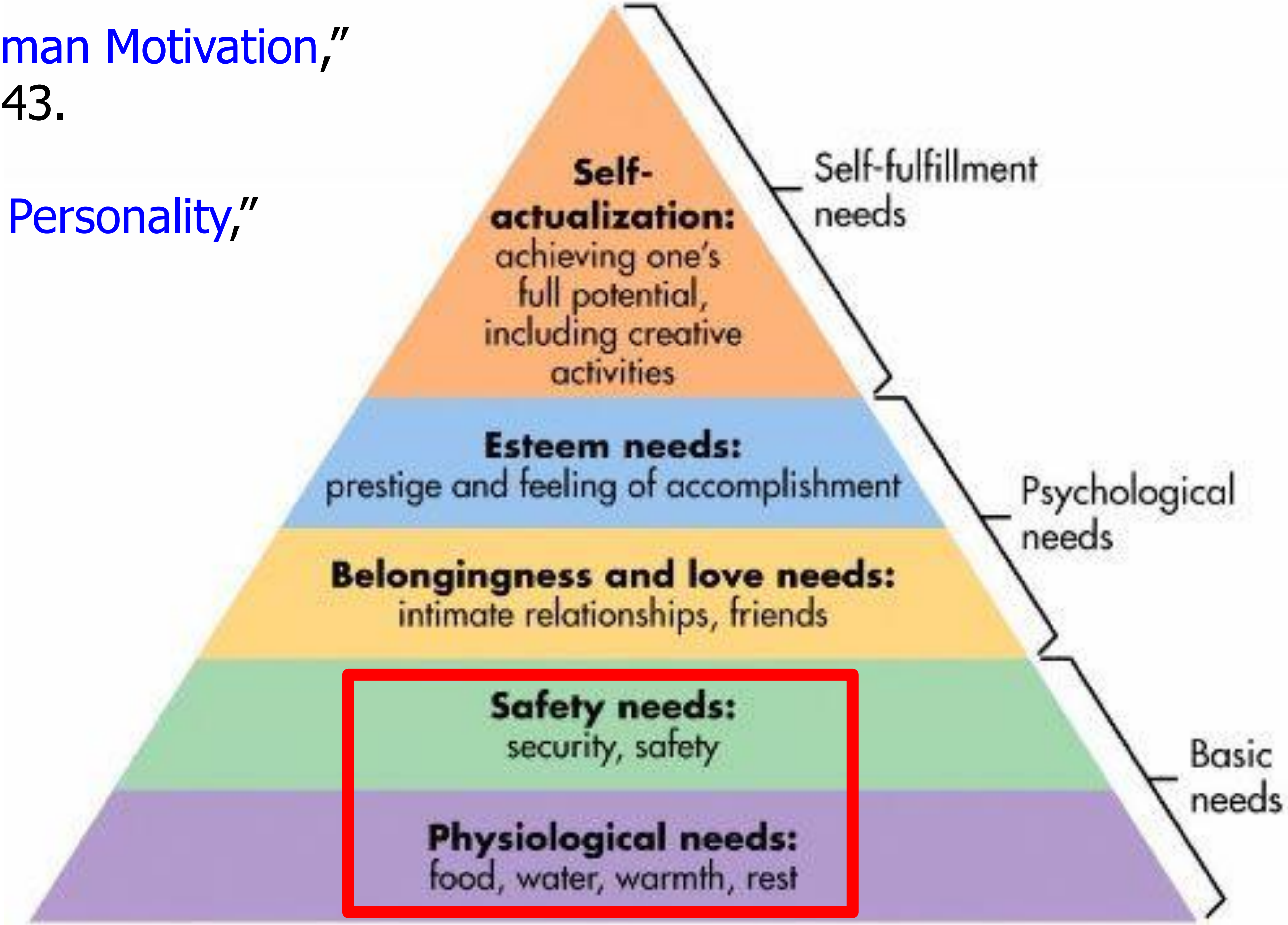
Fall 2022

14 October 2022

Maslow's (Human) Hierarchy of Needs

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

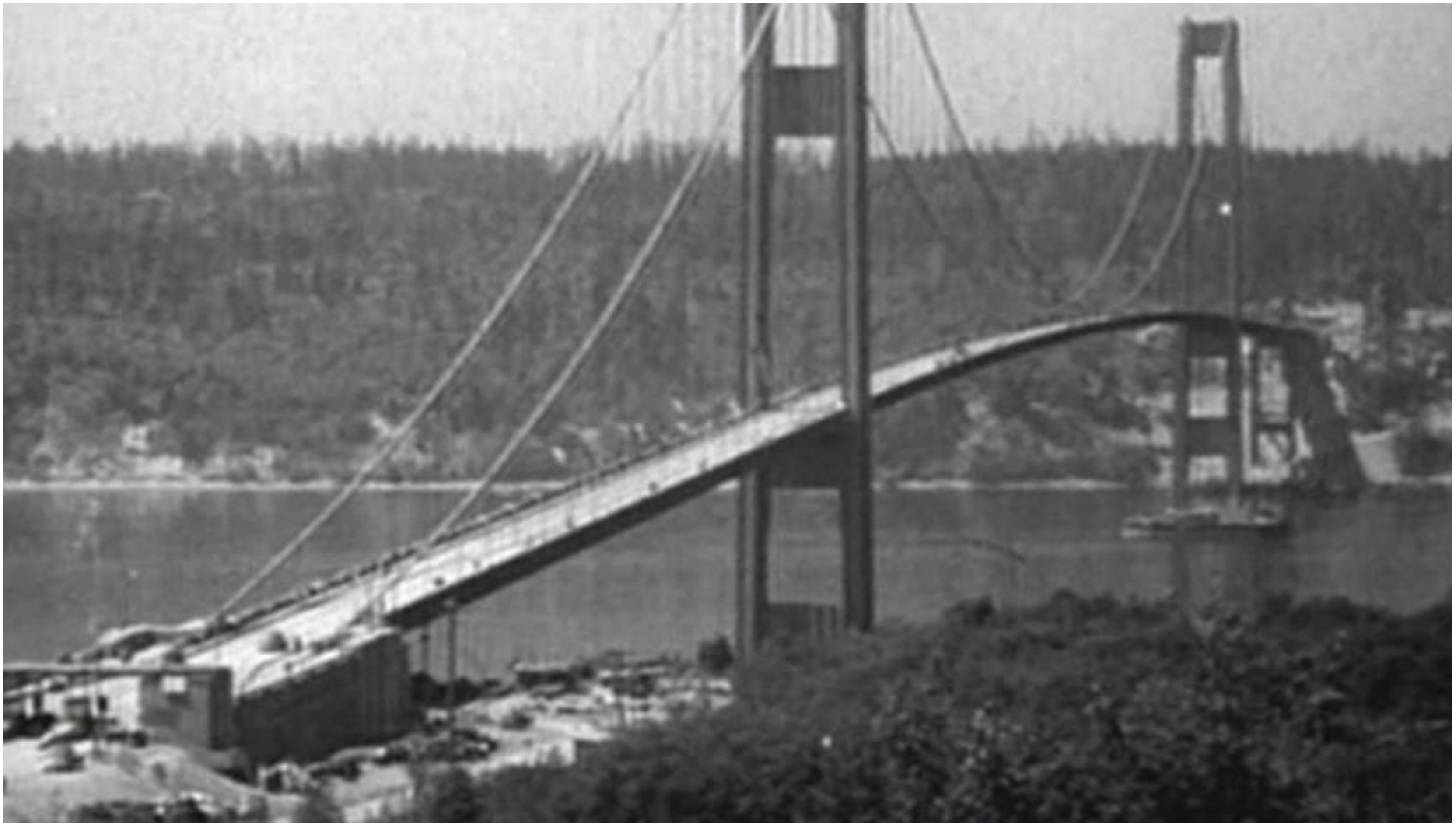
Maslow, "Motivation and Personality," Book, 1954-1970.


Self-actualization: achieving one's full potential, including creative activities
Esteem needs: prestige and feeling of accomplishment
Belongingness and love needs: intimate relationships, friends
Safety needs: security, safety
Physiological needs: food, water, warmth, rest
Self-fulfillment needs
Psychological needs
Basic needs

- We need to start with reliability and security…

Source: https://www.simplypsychology.org/maslow.html
Source: By User:Factoryjoe - Mazlow's Hierarchy of Needs.svg, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=7964065

How Reliable/Secure/Safe is This Bridge?



Source: http://www.technologystudent.com/struct1/tacom1.png

Collapse of the "Galloping Gertie"

Source: AP
http://www.wsdot.wa.gov/tnbhistory/connections/connections3.htm

How Secure Are These People?

Security is about preventing unforeseen consequences

Source: https://s-media-cache-ak0.pinimg.com/originals/48/09/54/4809543a9c7700246a0cf8acdae27abf.jpg

How Safe & Secure Are Our Platforms?

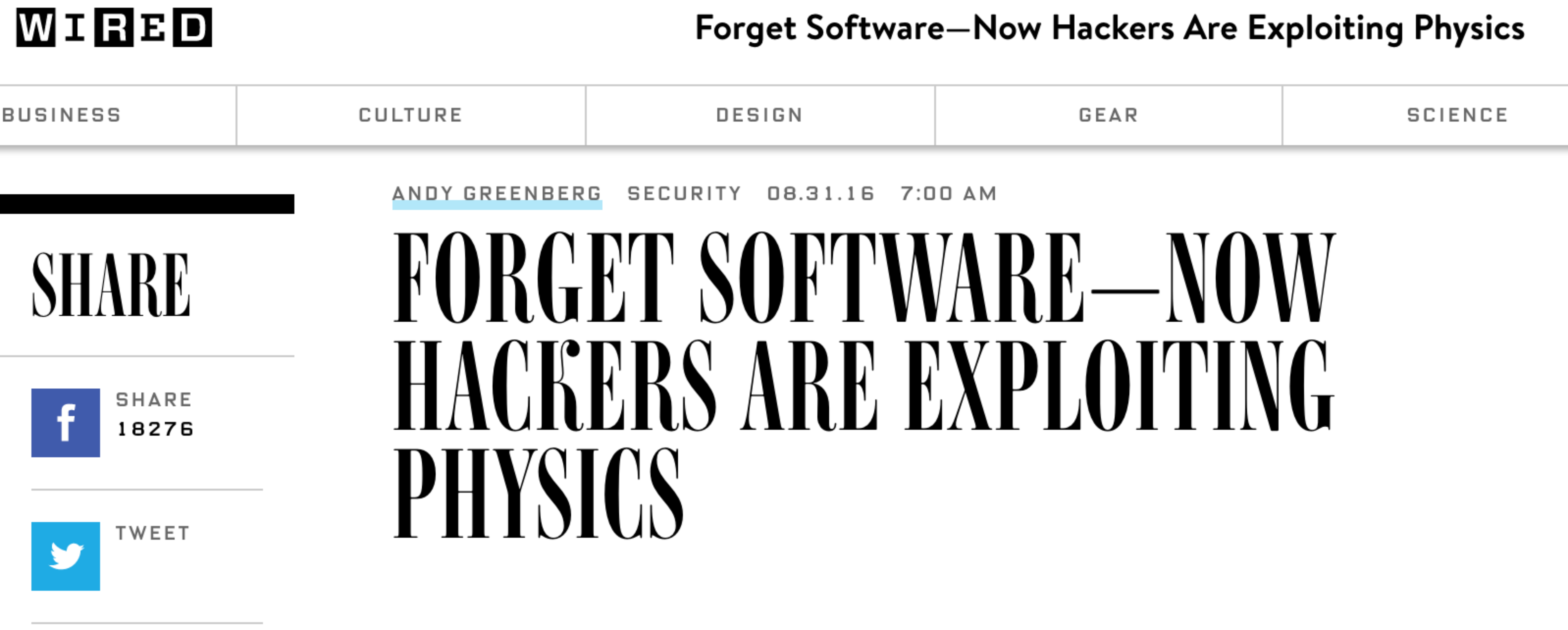
Security is about preventing unforeseen consequences

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

What Is 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



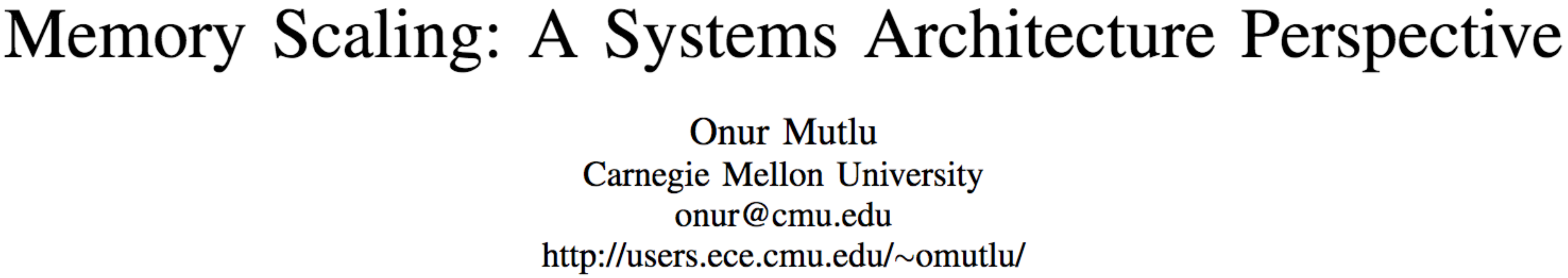
Forget Software—Now Hackers Are Exploiting Physics

ANDY GREENBERG SECURITY 08.31.16 7:00 AM

FORGET SOFTWARE—NOW HACKERS ARE EXPLOITING PHYSICS

An “Early” Position Paper **[IMW’13]**

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



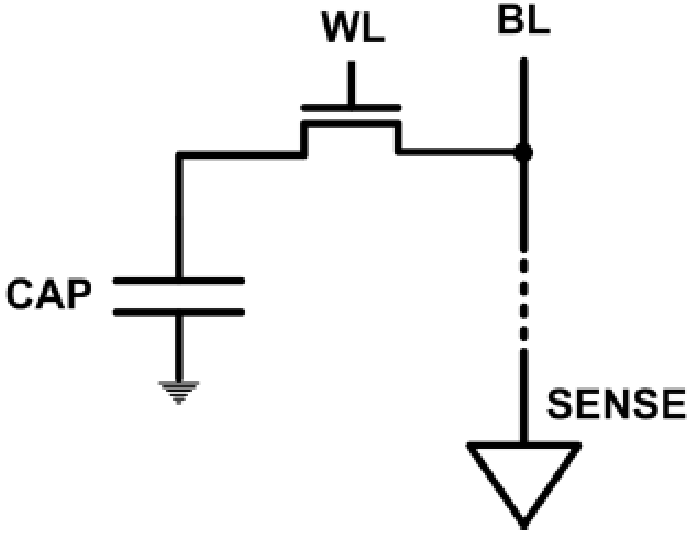
Memory Scaling: A Systems Architecture Perspective

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

https://people.inf.ethz.ch/omutlu/pub/memory-scaling_memcon13.pdf

The DRAM Scaling Problem

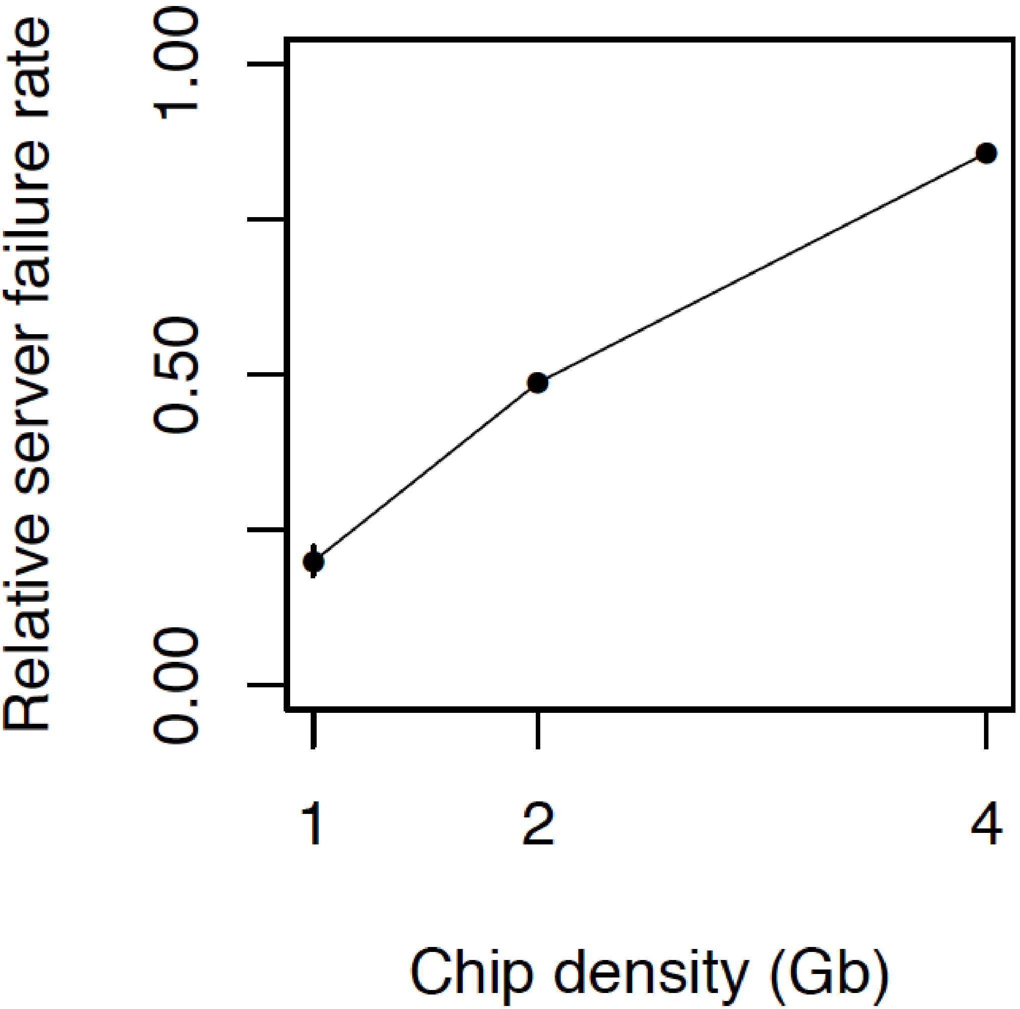
- DRAM stores charge in a capacitor (charge-based memory)
  - Capacitor must be large enough for reliable sensing
  - Access transistor should be large enough for low leakage and high retention time
  - Scaling beyond 40-35nm (2013) is challenging [ITRS, 2009]



- DRAM capacity, cost, and energy/power hard to scale

As Memory Scales, It Becomes Unreliable

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

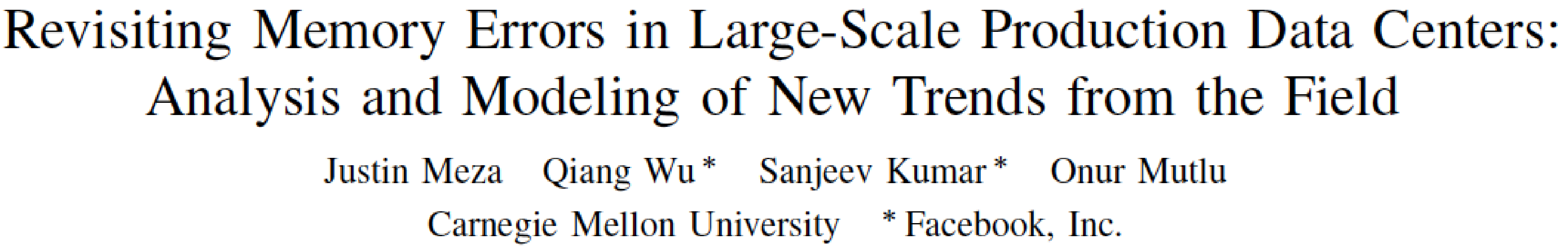


Intuition: quadratic increase in capacity

Large-Scale Failure Analysis of DRAM Chips

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

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



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

Justin Meza Qiang Wu * Sanjeev Kumar * Onur Mutlu
Carnegie Mellon University * Facebook, Inc.

Infrastructures to Understand Such Issues

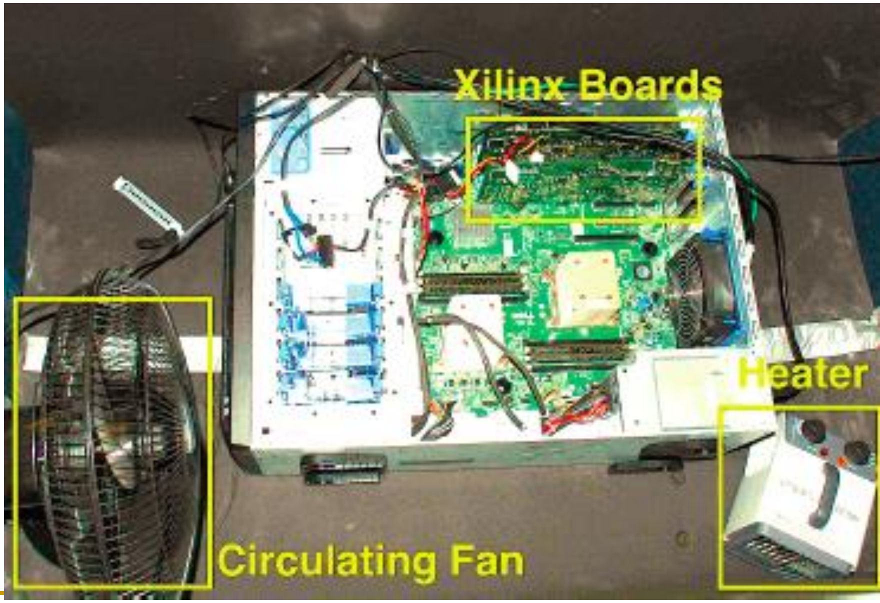
An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms (Liu et al., ISCA 2013)

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

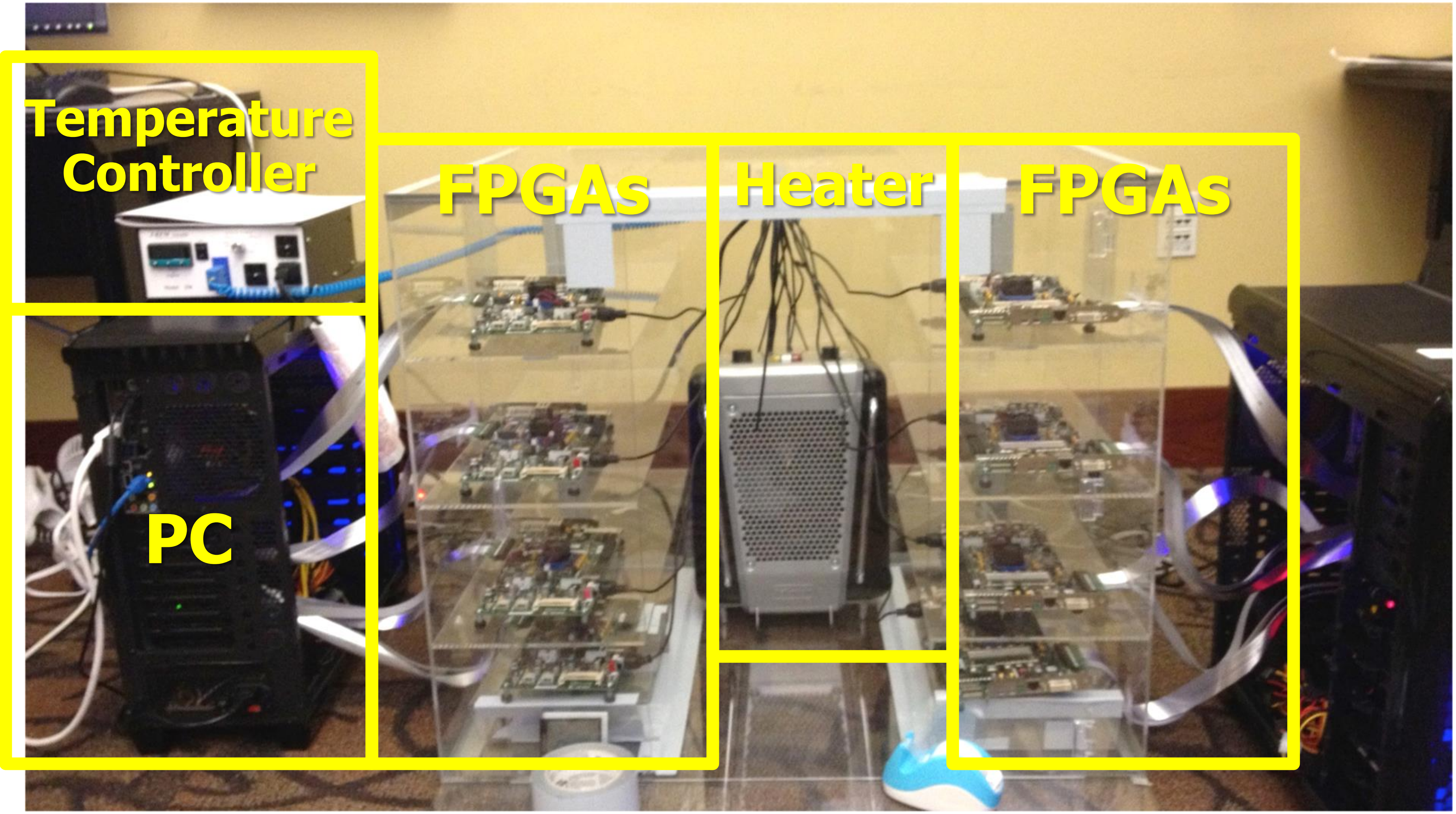
Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors (Kim et al., ISCA 2014)

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)



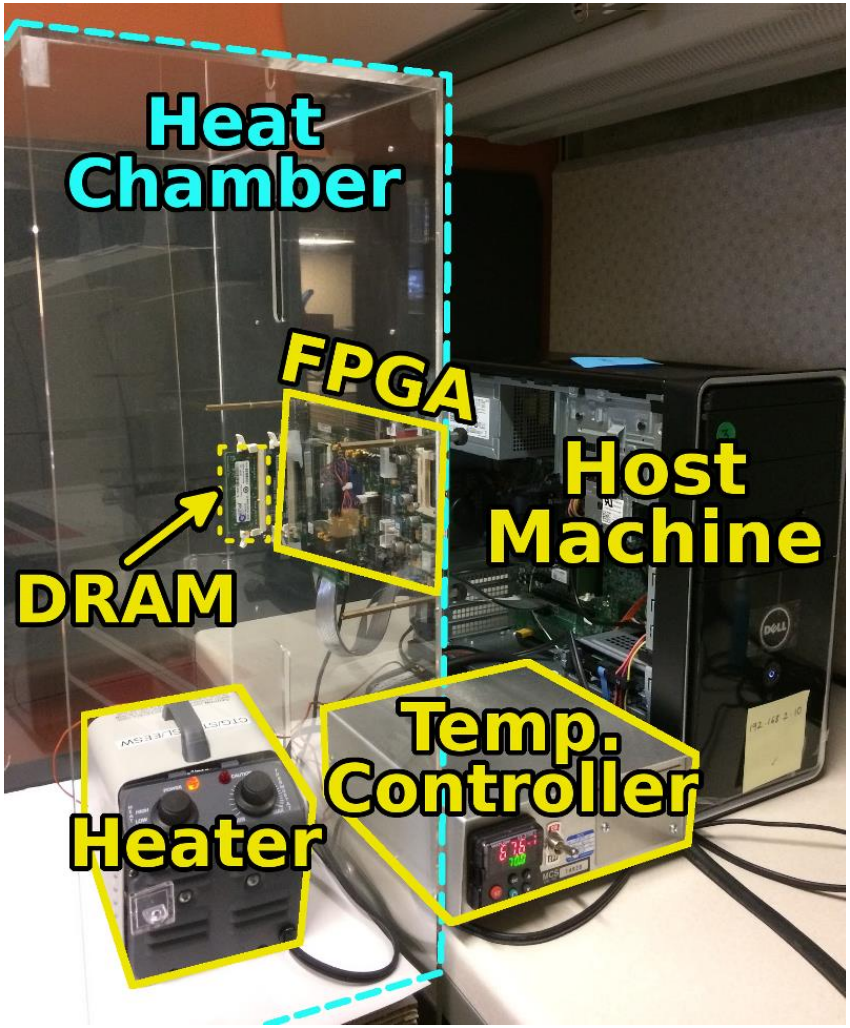
Infrastructures to Understand Such Issues



Kim+, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," ISCA 2014.

SoftMC: Open Source DRAM Infrastructure

- Hasan Hassan et al., "**SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies**," HPCA 2017.

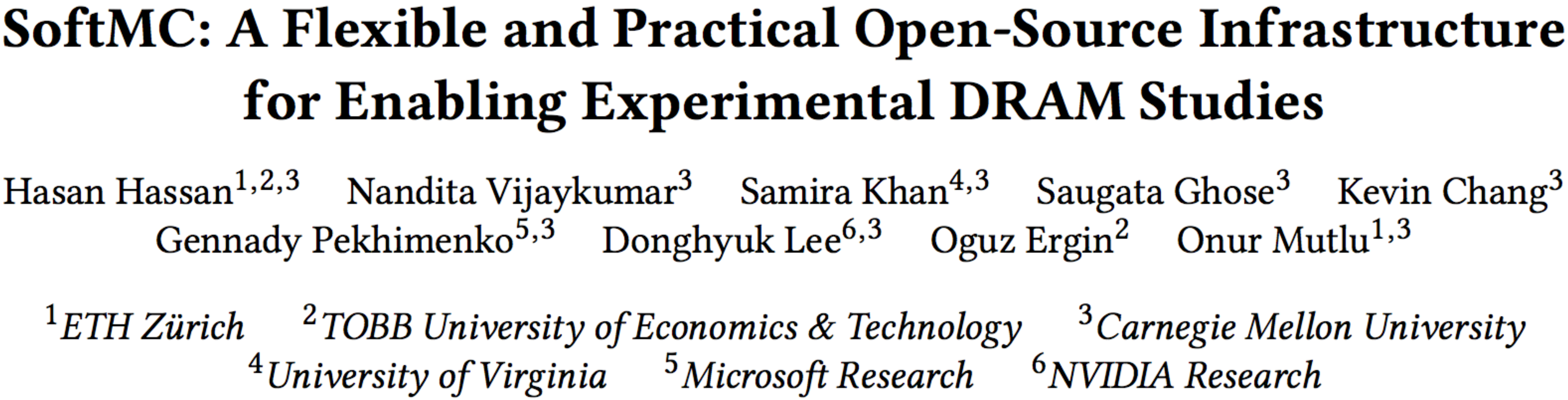


- **Flexible**
- **Easy to Use (C++ API)**
- **Open-source**

github.com/CMU-SAFARI/SoftMC

SoftMC: Open Source DRAM Infrastructure

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



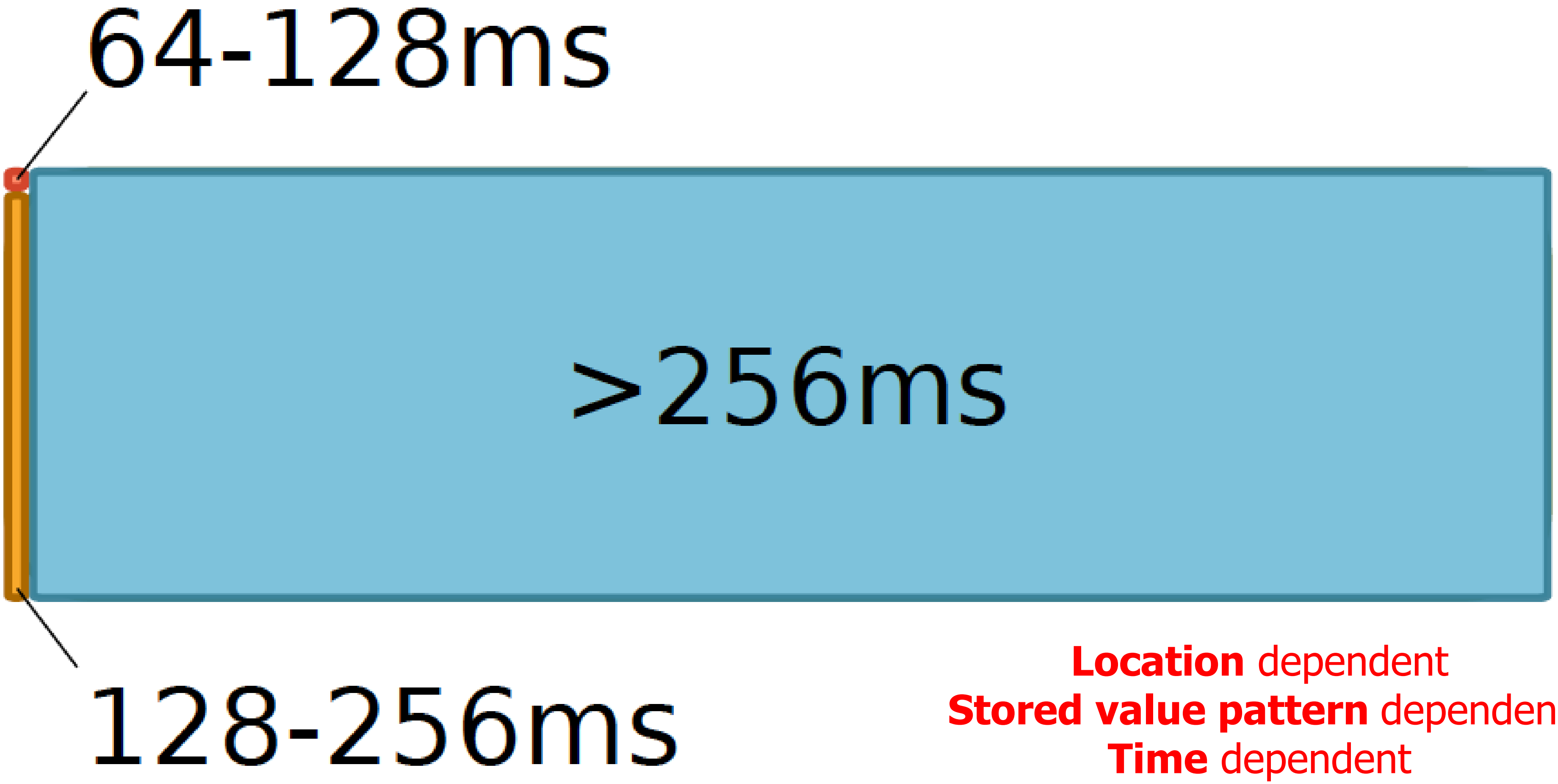
SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies

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

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

Data Retention in Memory **[Liu et al., ISCA 2013]**

- Retention Time Profile of DRAM looks like this:



64-128ms

>256ms

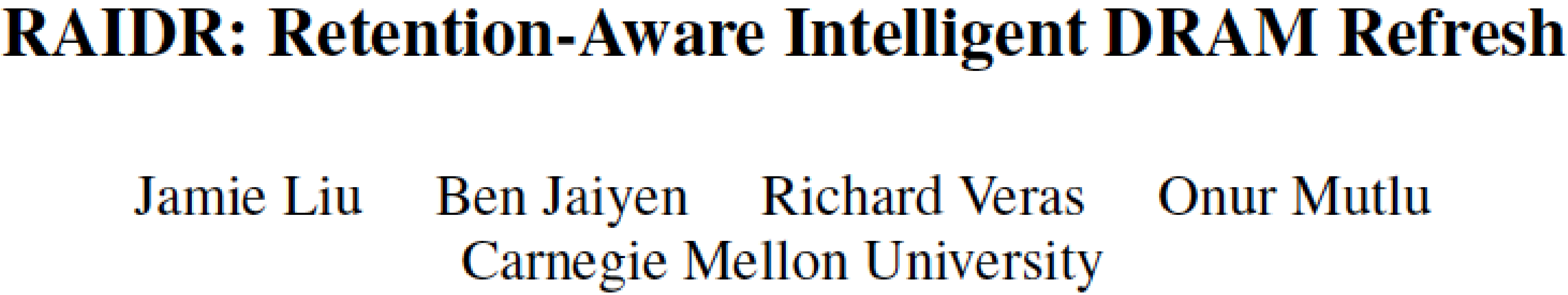
128-256ms

Location dependent
Stored value pattern dependent
Time dependent

SAFARI Liu+, "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

RAIDR: Heterogeneous Refresh [ISCA'12]

- 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)

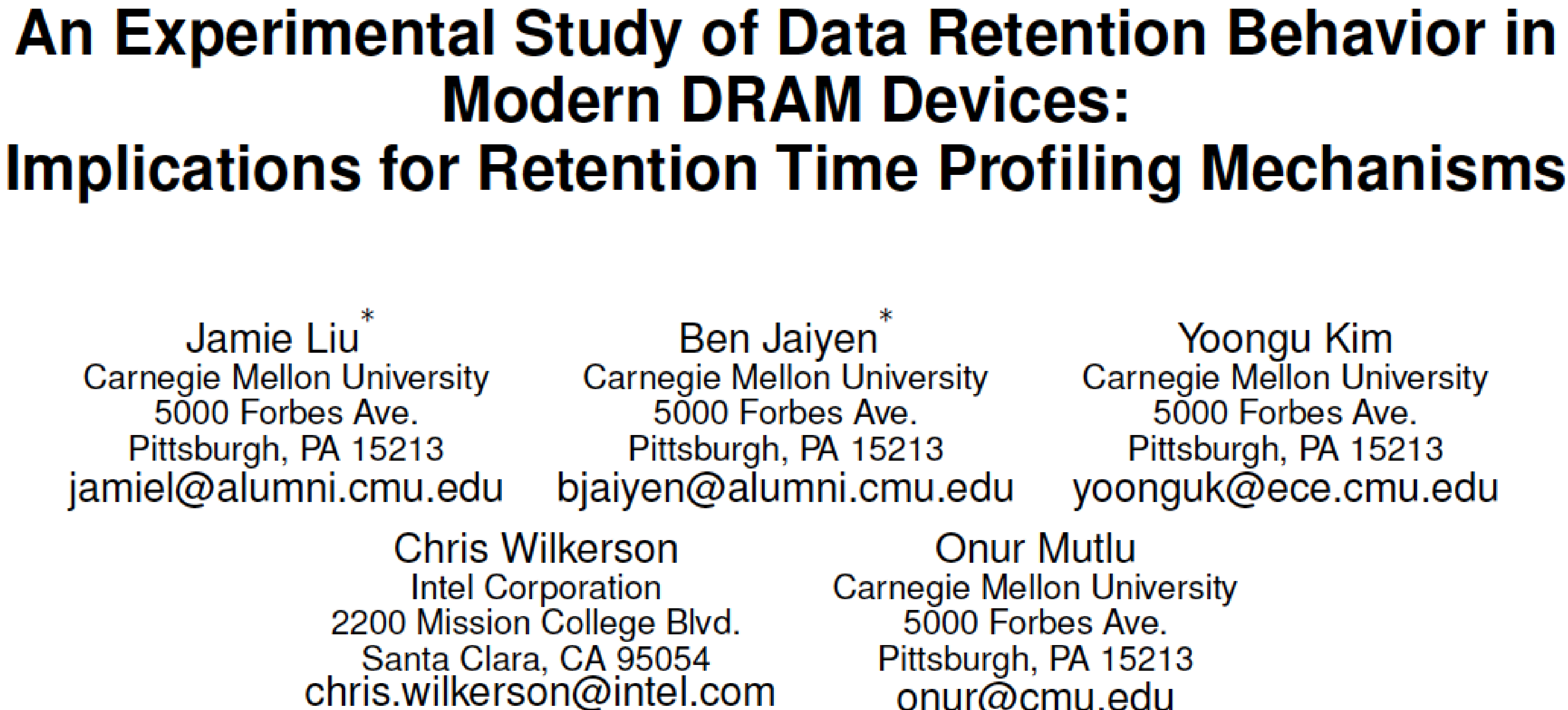


RAIDR: Retention-Aware Intelligent DRAM Refresh

Jamie Liu Ben Jaiyen Richard Veras Onur Mutlu
Carnegie Mellon University

Analysis of Data Retention Failures [ISCA'13]

- Jamie Liu, Ben Jaiyen, Yoongu Kim, Chris Wilkerson, and Onur Mutlu, **"An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms"** *Proceedings of the 40th International Symposium on Computer Architecture (**ISCA**)*, Tel-Aviv, Israel, June 2013. Slides (ppt) Slides (pdf)



An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms

Jamie Liu*
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
jamiel@alumni.cmu.edu

Ben Jaiyen*
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
bjaiyen@alumni.cmu.edu

Yoongu Kim
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
yoonguk@ece.cmu.edu

Chris Wilkerson
Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95054
chris.wilkerson@intel.com

Onur Mutlu
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
onur@cmu.edu

Mitigation of Retention Issues [SIGMETRICS'14]

- Samira Khan, Donghyuk Lee, Yoongu Kim, Alaa Alameldeen, Chris Wilkerson, and Onur Mutlu,
  "The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study"
  *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (**SIGMETRICS**)*, Austin, TX, June 2014. [Slides (pptx) (pdf)] [Poster (pptx) (pdf)] [Full data sets]

The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study

Samira Khan[†][*]
samirakhan@cmu.edu

Donghyuk Lee[†]
donghyuk1@cmu.edu

Yoongu Kim[†]
yoongukim@cmu.edu

Alaa R. Alameldeen[*]
alaa.r.alameldeen@intel.com

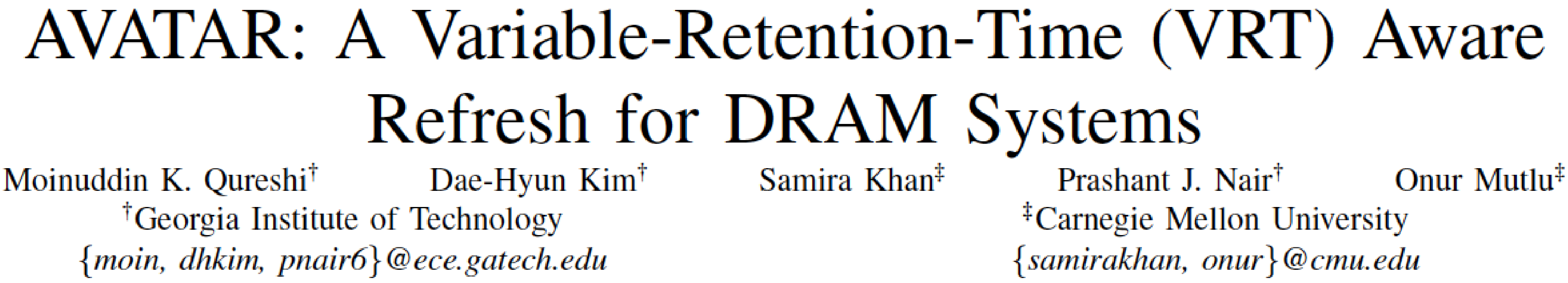
Chris Wilkerson[*]
chris.wilkerson@intel.com

Onur Mutlu[†]
onur@cmu.edu

[†]Carnegie Mellon University [*]Intel Labs

Mitigation of Retention Issues [DSN'15]

- Moinuddin Qureshi, Dae Hyun Kim, Samira Khan, Prashant Nair, and Onur Mutlu,
  "AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems"
  *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**)*, Rio de Janeiro, Brazil, June 2015.
  [Slides (pptx) (pdf)]



AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems

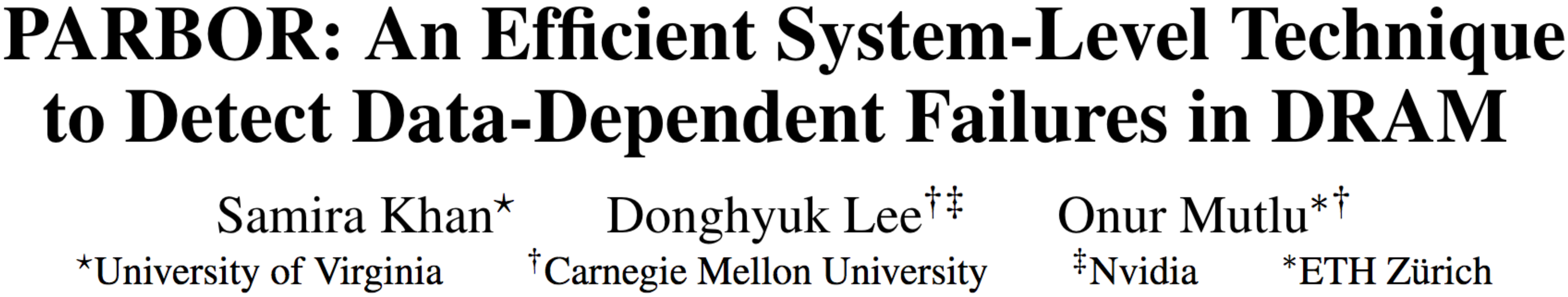
Moinuddin K. Qureshi[†] Dae-Hyun Kim[†] Samira Khan[‡] Prashant J. Nair[†] Onur Mutlu[‡]

[†]Georgia Institute of Technology
{moin, dhkim, pnair6}@ece.gatech.edu

[‡]Carnegie Mellon University
{samirakhan, onur}@cmu.edu

Mitigation of Retention Issues [DSN'16]

- Samira Khan, Donghyuk Lee, and Onur Mutlu,
  "PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM"
  *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**)*, Toulouse, France, June 2016.
  [Slides (pptx) (pdf)]



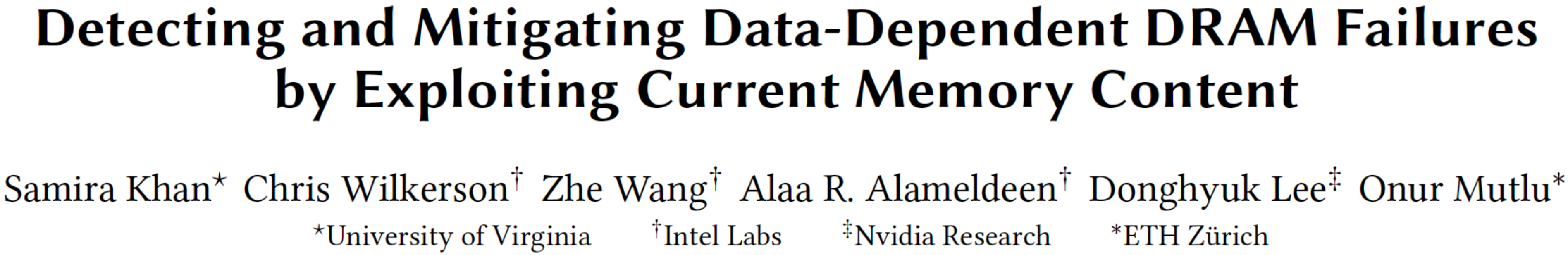
PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM

Samira Khan* Donghyuk Lee†‡ Onur Mutlu*†

*University of Virginia †Carnegie Mellon University ‡Nvidia *ETH Zürich

Mitigation of Retention Issues [MICRO'17]

- Samira Khan, Chris Wilkerson, Zhe Wang, Alaa R. Alameldeen, Donghyuk Lee, and Onur Mutlu,
"Detecting and Mitigating Data-Dependent DRAM Failures by Exploiting Current Memory Content"
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)]



Detecting and Mitigating Data-Dependent DRAM Failures by Exploiting Current Memory Content

Samira Khan[⋆] Chris Wilkerson[†] Zhe Wang[†] Alaa R. Alameldeen[†] Donghyuk Lee[‡] Onur Mutlu[*]
[⋆]University of Virginia [†]Intel Labs [‡]Nvidia Research [*]ETH Zürich

Mitigation of Retention Issues [ISCA'17]

- Minesh Patel, Jeremie S. Kim, and Onur Mutlu,
  "The Reach Profiler (REAPER): Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions"
  *Proceedings of the 44th International Symposium on Computer Architecture (**ISCA**)*, Toronto, Canada, June 2017.
  [Slides (pptx) (pdf)]
  [Lightning Session Slides (pptx) (pdf)]

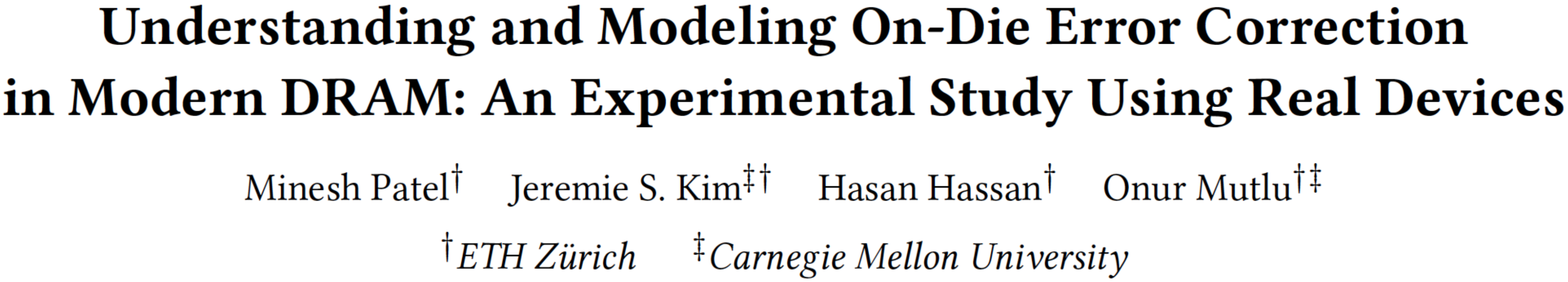
- First experimental analysis of (mobile) LPDDR4 chips
- Analyzes the complex tradeoff space of retention time profiling
- Idea: enable fast and robust profiling at higher refresh intervals & temperatures

**The Reach Profiler (REAPER):
Enabling the Mitigation of DRAM Retention Failures
via Profiling at Aggressive Conditions**

Minesh Patel[§‡] Jeremie S. Kim[‡§] Onur Mutlu[§‡]
[§]ETH Zürich [‡]Carnegie Mellon University

Mitigation of Retention Issues [DSN'19]

- Minesh Patel, Jeremie S. Kim, Hasan Hassan, and Onur Mutlu, **"Understanding and Modeling On-Die Error Correction in Modern DRAM: An Experimental Study Using Real Devices"** *Proceedings of the 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**)*, Portland, OR, USA, June 2019.
  [Source Code for EINSim, the Error Inference Simulator]
  Best paper award.



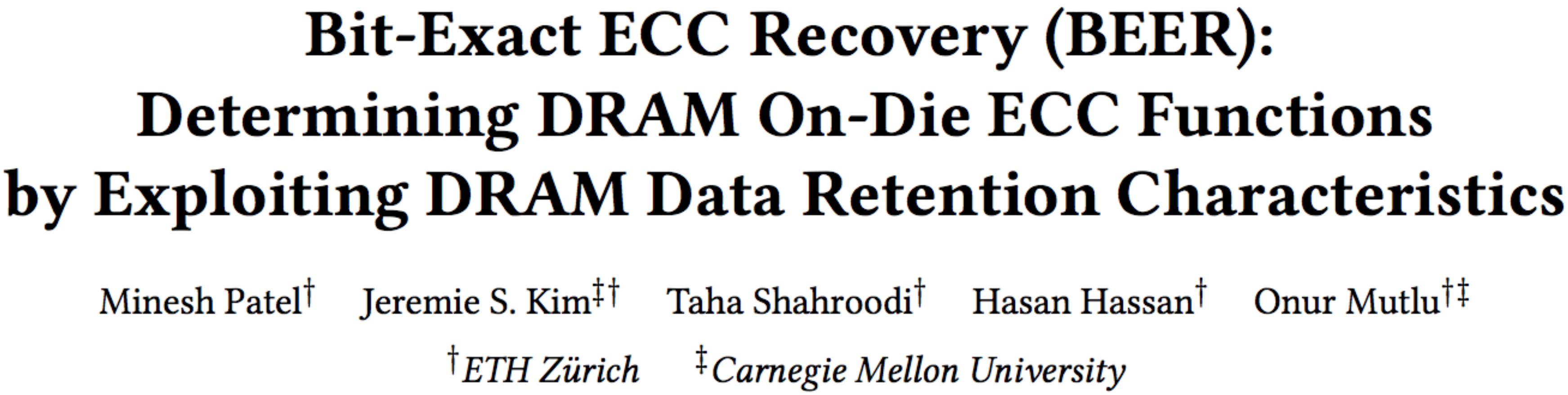
Understanding and Modeling On-Die Error Correction in Modern DRAM: An Experimental Study Using Real Devices

Minesh Patel[†] Jeremie S. Kim[‡†] Hasan Hassan[†] Onur Mutlu[†‡]

[†]*ETH Zürich* [‡]*Carnegie Mellon University*

Mitigation of Retention Issues [MICRO'20]

- Minesh Patel, Jeremie S. Kim, Taha Shahroodi, Hasan Hassan, and Onur Mutlu, **"Bit-Exact ECC Recovery (BEER): Determining DRAM On-Die ECC Functions by Exploiting DRAM Data Retention Characteristics"**
  Proceedings of the 53rd International Symposium on Microarchitecture (***MICRO***), Virtual, October 2020.
  [Slides (pptx) (pdf)]
  [Lightning Talk Slides (pptx) (pdf)]
  [Talk Video (15 minutes)]
  [Lightning Talk Video (1.5 minutes)]
  Best paper award.



Bit-Exact ECC Recovery (BEER): Determining DRAM On-Die ECC Functions by Exploiting DRAM Data Retention Characteristics

Minesh Patel[†] Jeremie S. Kim[‡†] Taha Shahroodi[†] Hasan Hassan[†] Onur Mutlu[†‡]

[†]*ETH Zürich* [‡]*Carnegie Mellon University*

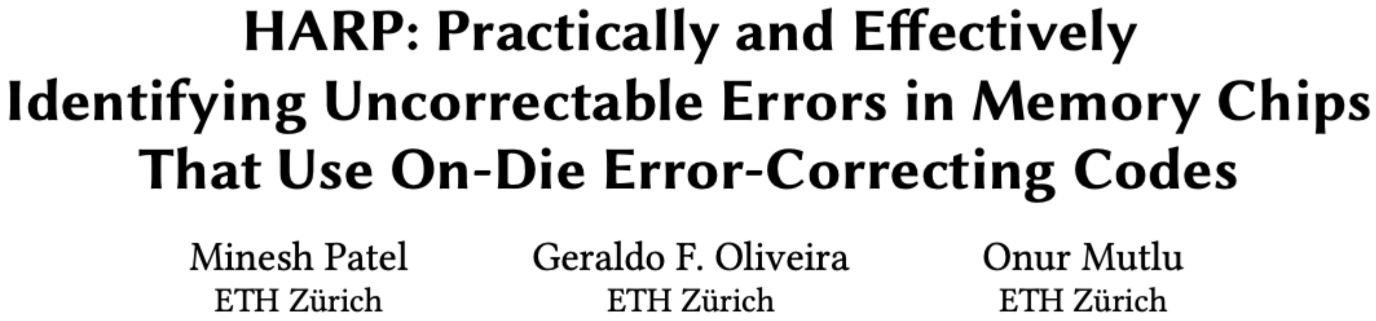
Mitigation of Retention Issues [MICRO'21]

- Minesh Patel, Geraldo F. de Oliveira Jr., and Onur Mutlu,
  "HARP: Practically and Effectively Identifying Uncorrectable Errors in Memory Chips That Use On-Die Error-Correcting Codes"
  *Proceedings of the 54th International Symposium on Microarchitecture (**MICRO**)*,
  Virtual, October 2021.
  [Slides (pptx) (pdf)]
  [Short Talk Slides (pptx) (pdf)]
  [Lightning Talk Slides (pptx) (pdf)]
  [Talk Video (20 minutes)]
  [Lightning Talk Video (1.5 minutes)]
  [HARP Source Code (Officially Artifact Evaluated with All Badges)]

Artifacts Evaluated acm Functional V1.1

Artifacts Available acm V1.1

Results Reproduced acm V1.1



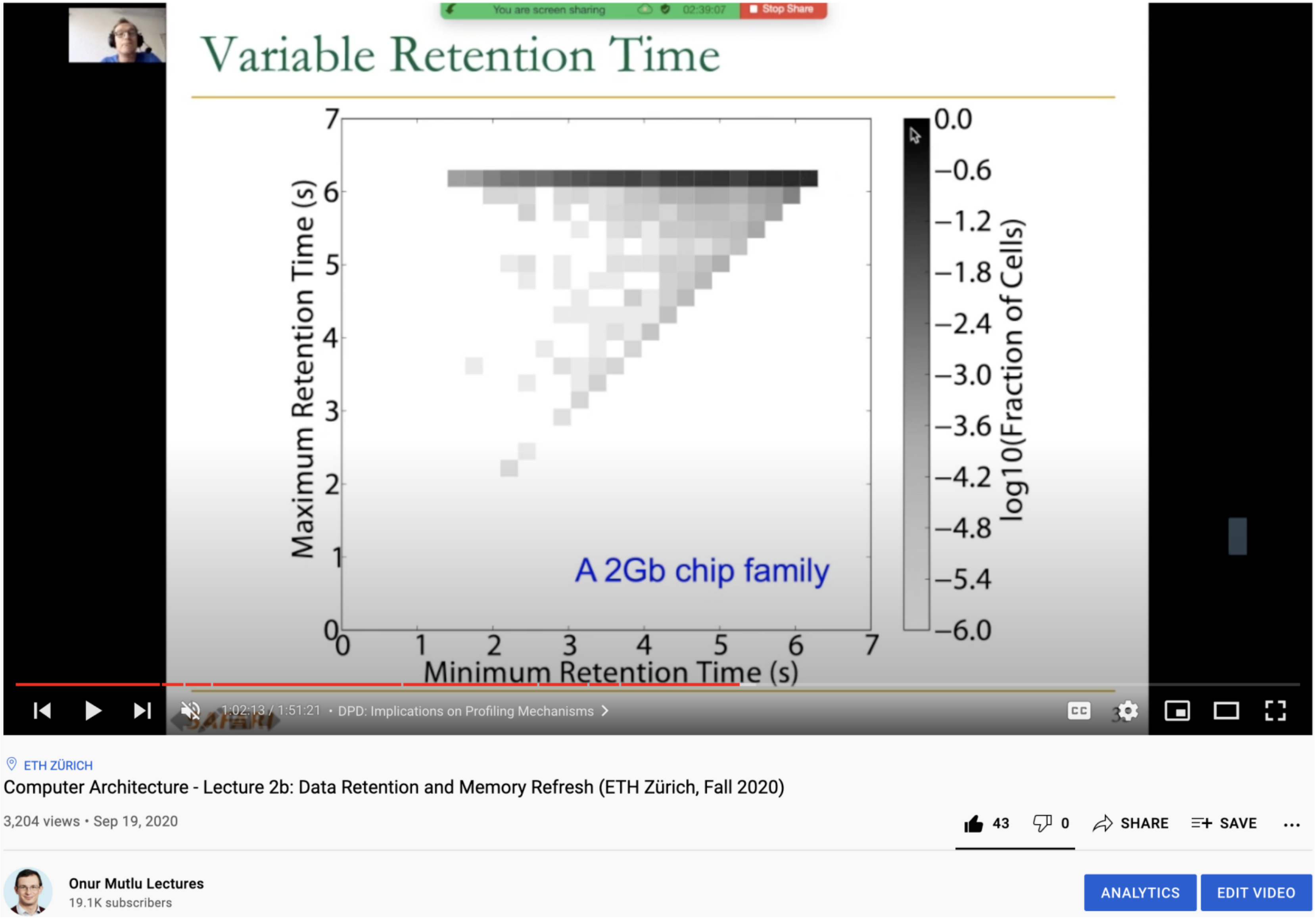
HARP: Practically and Effectively Identifying Uncorrectable Errors in Memory Chips That Use On-Die Error-Correcting Codes

Minesh Patel
ETH Zürich

Geraldo F. Oliveira
ETH Zürich

Onur Mutlu
ETH Zürich

More on DRAM Refresh & Data Retention



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

3,204 views • Sep 19, 2020

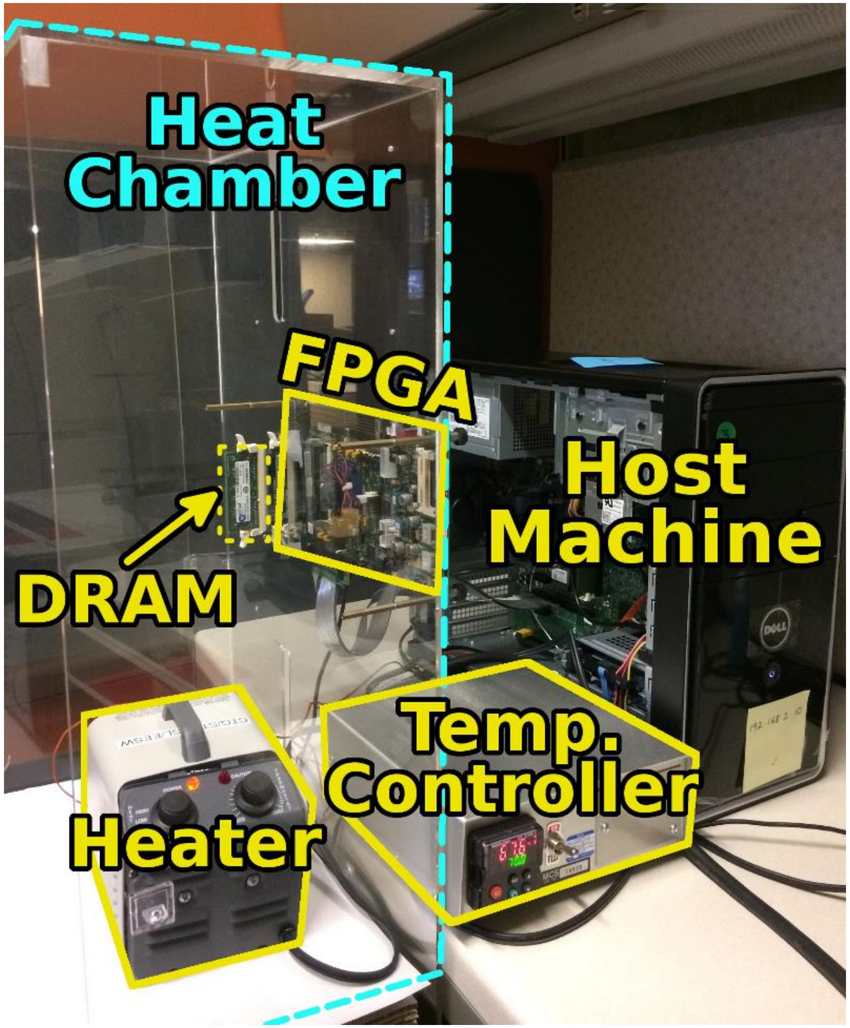
Onur Mutlu Lectures

19.1K subscribers

https://www.youtube.com/watch?v=v702wUnaWGE&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=3

SoftMC: Enabling 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

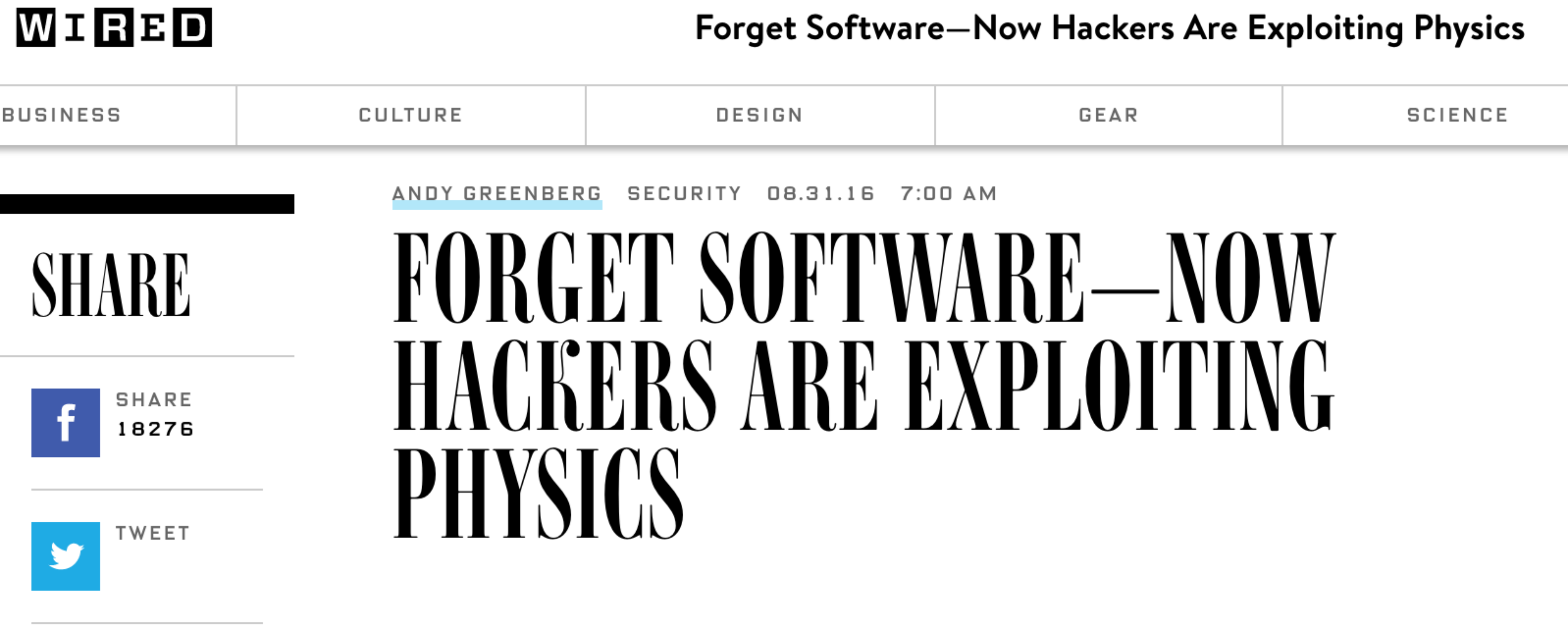
A Curious Phenomenon

A Curious Discovery **[Kim et al., ISCA 2014]**

One can
predictably induce errors
in most DRAM memory chips

DRAM RowHammer

A simple hardware failure mechanism can create a widespread system security vulnerability

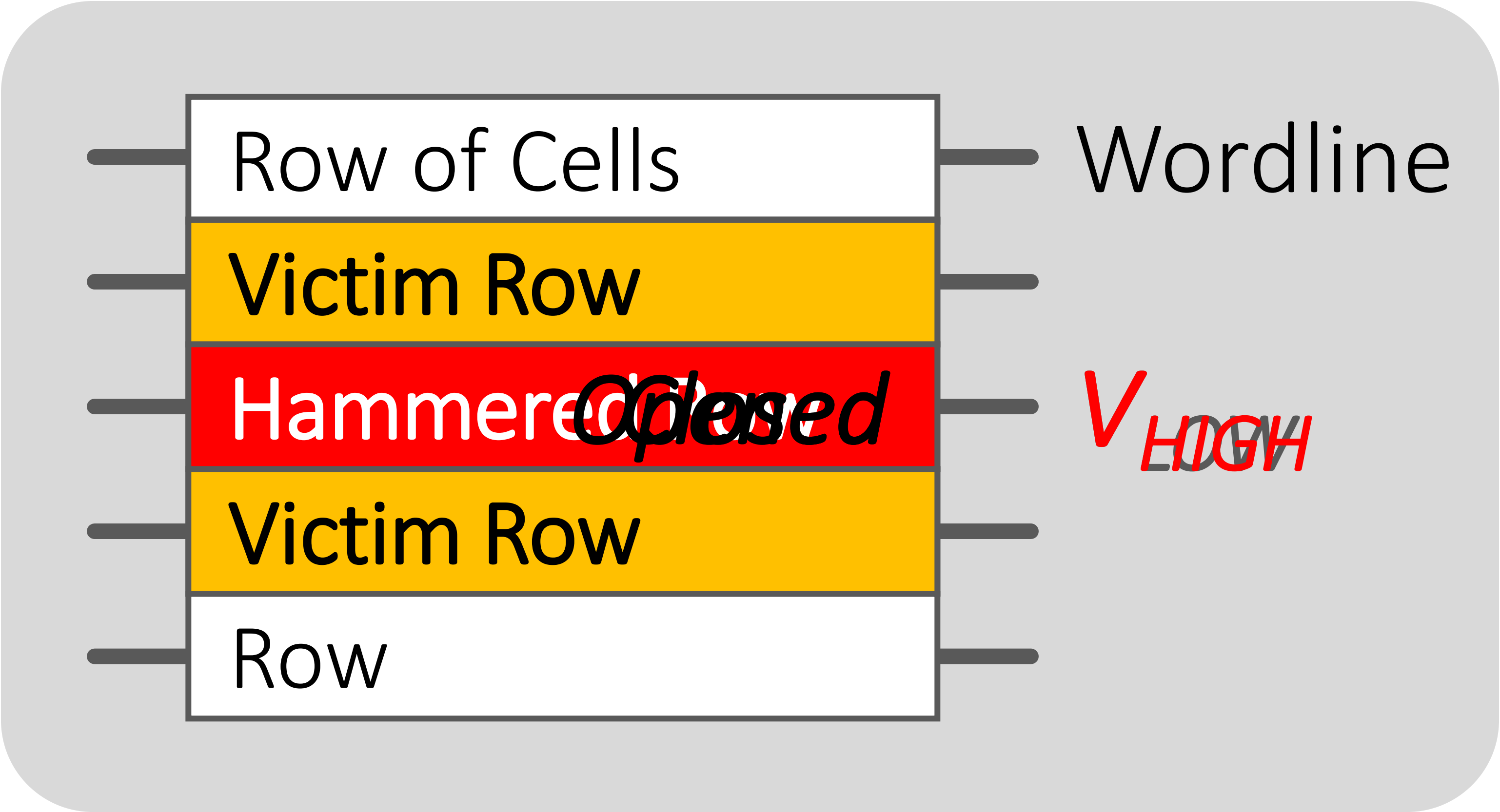


Forget Software—Now Hackers Are Exploiting Physics

ANDY GREENBERG SECURITY 08.31.16 7:00 AM

FORGET SOFTWARE—NOW HACKERS ARE EXPLOITING PHYSICS

Modern DRAM is Prone to Disturbance Errors

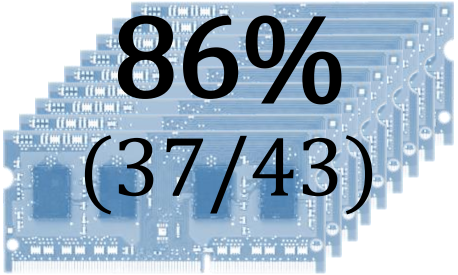


Repeatedly reading a row enough times (before memory gets refreshed) induces disturbance errors in **adjacent rows** in most real DRAM chips you can buy today

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

Most DRAM Modules Are Vulnerable

A company



86%

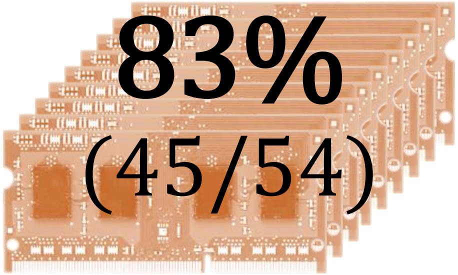
(37/43)

Up to

1.0×10^{7}

errors

B company



83%

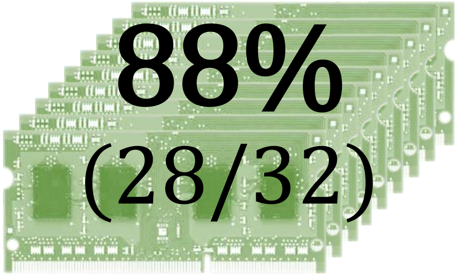
(45/54)

Up to

2.7×10^{6}

errors

C company



88%

(28/32)

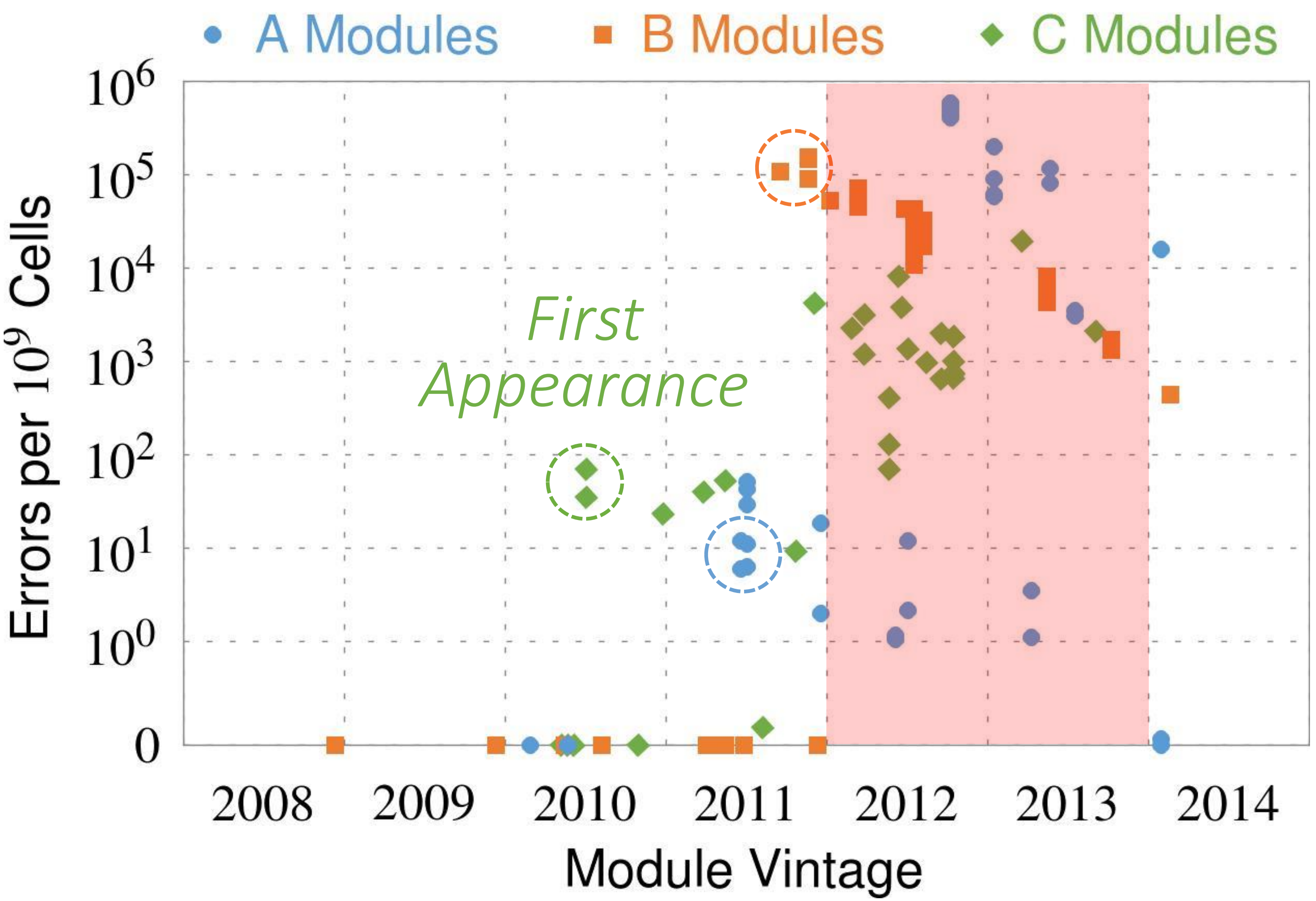
Up to

3.3×10^{5}

errors

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

Recent DRAM Is More Vulnerable



All modules from 2012–2013 are vulnerable

Why Is This Happening?

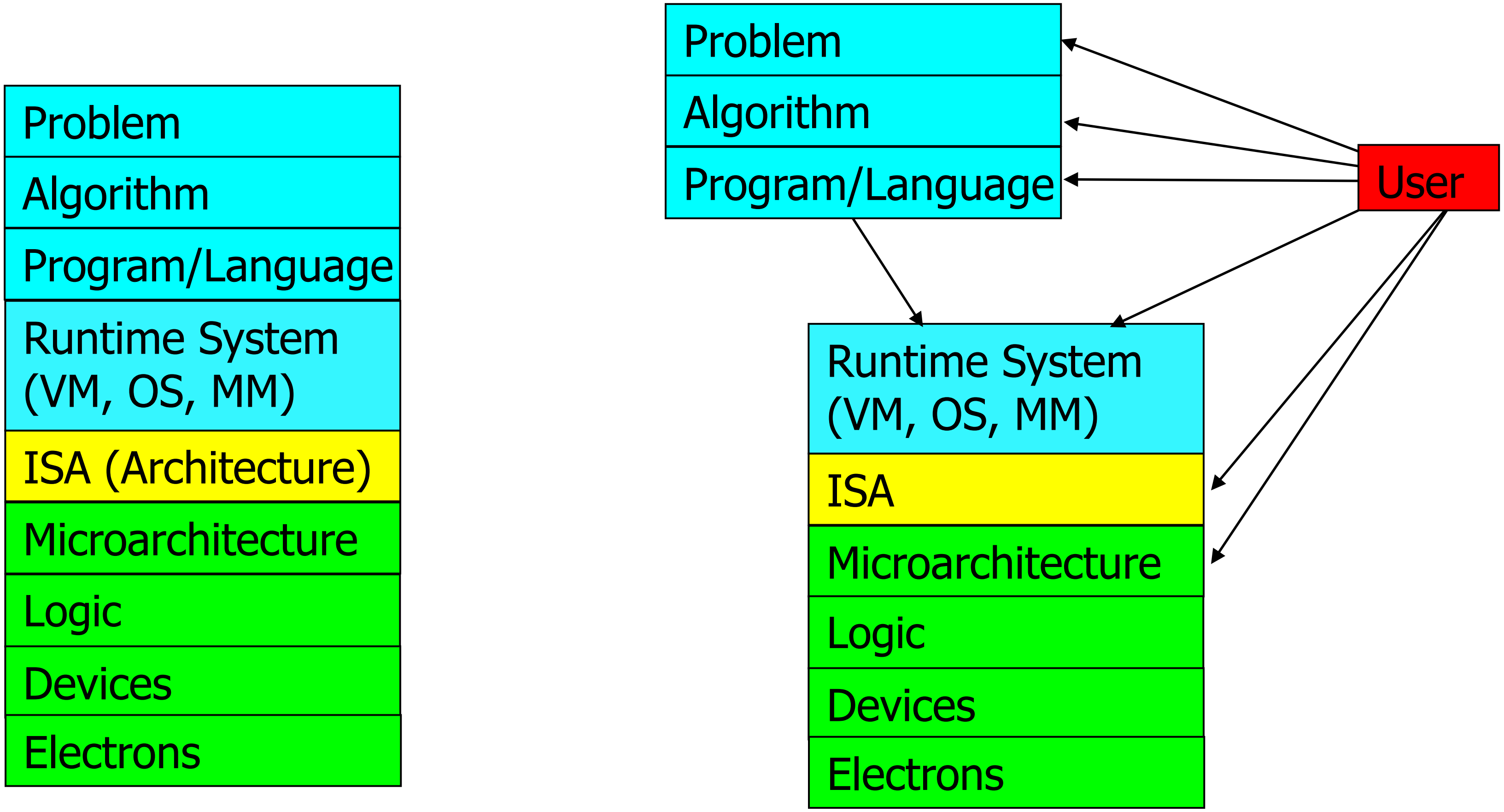
- DRAM cells are too close to each other!
  - They are not electrically isolated from each other

- Access to one cell affects the value in nearby cells
  - due to electrical interference between
    - the cells
    - wires used for accessing the cells
  - Also called cell-to-cell coupling/interference

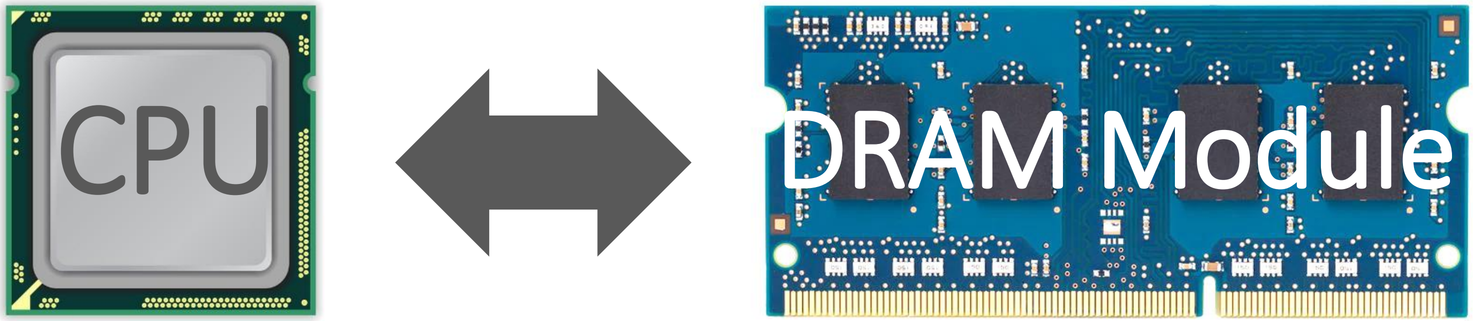
- Example: When we activate (apply high voltage) to a row, an adjacent row gets slightly activated as well
  - Vulnerable cells in that slightly-activated row lose a little bit of charge
  - If RowHammer happens enough times, charge in such cells gets drained

Higher-Level Implications

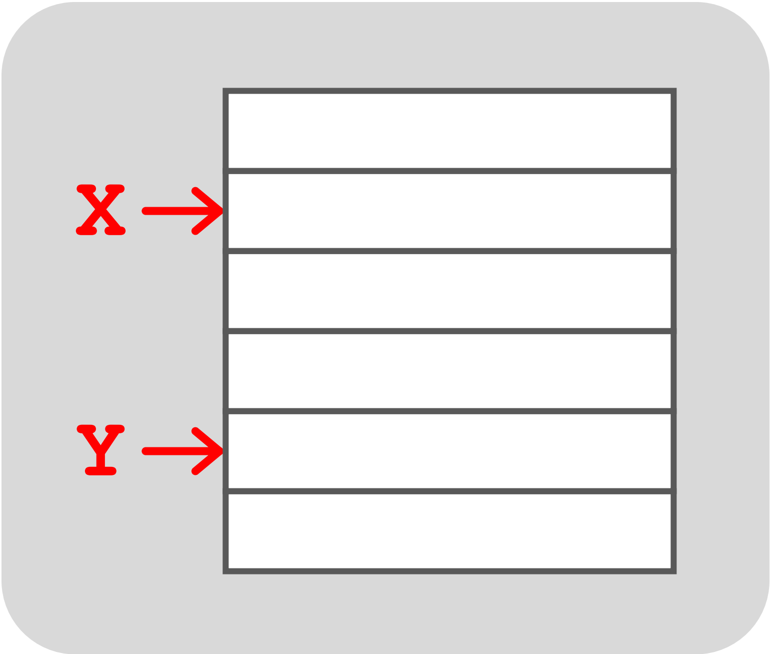
- This simple circuit level failure mechanism has enormous implications on upper layers of the transformation hierarchy



A Simple Program Can Induce Many Errors

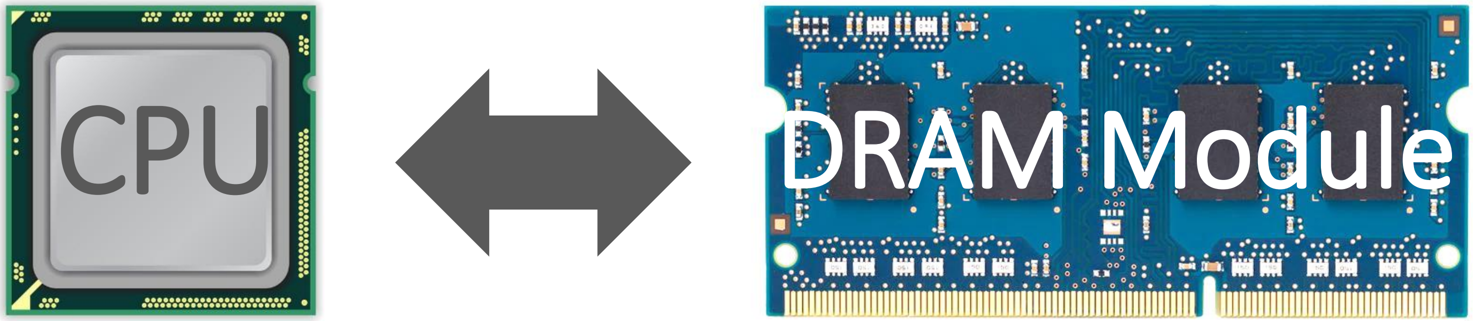


```
loop:
   mov (X), %eax
   mov (Y), %ebx
   clflush (X)
   clflush (Y)
   mfence
   jmp loop
```



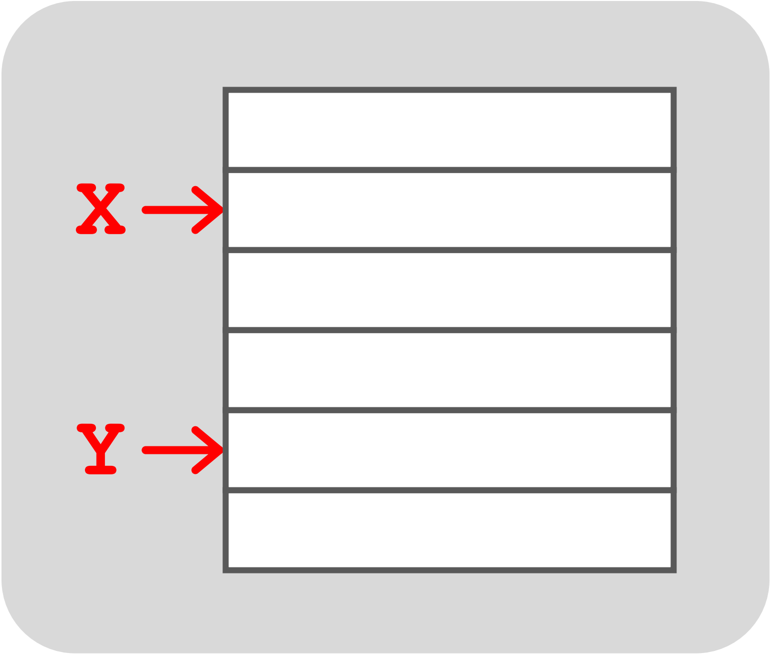
Download from: https://github.com/CMU-SAFARI/rowhammer

A Simple Program Can Induce Many Errors



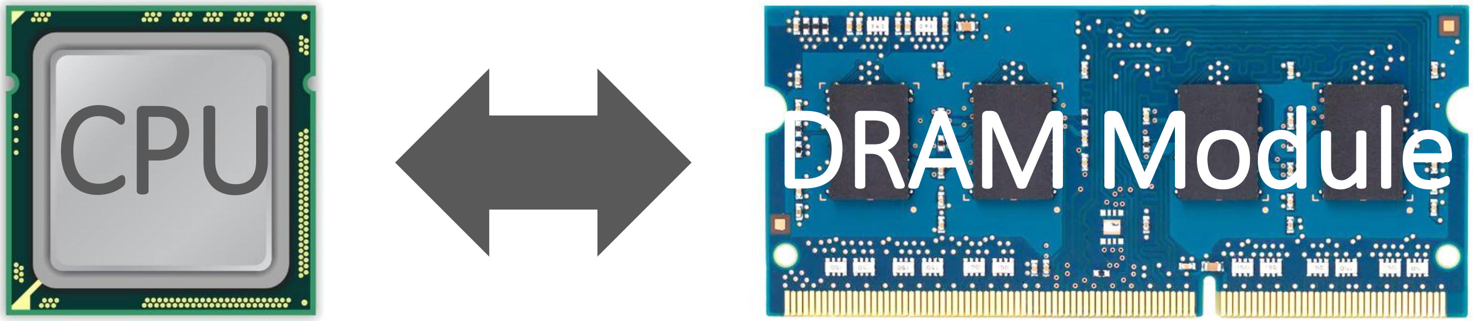
1. Avoid ***cache hits***
   - Flush **X** from cache

2. Avoid ***row hits*** to **X**
   - Read **Y** in another row

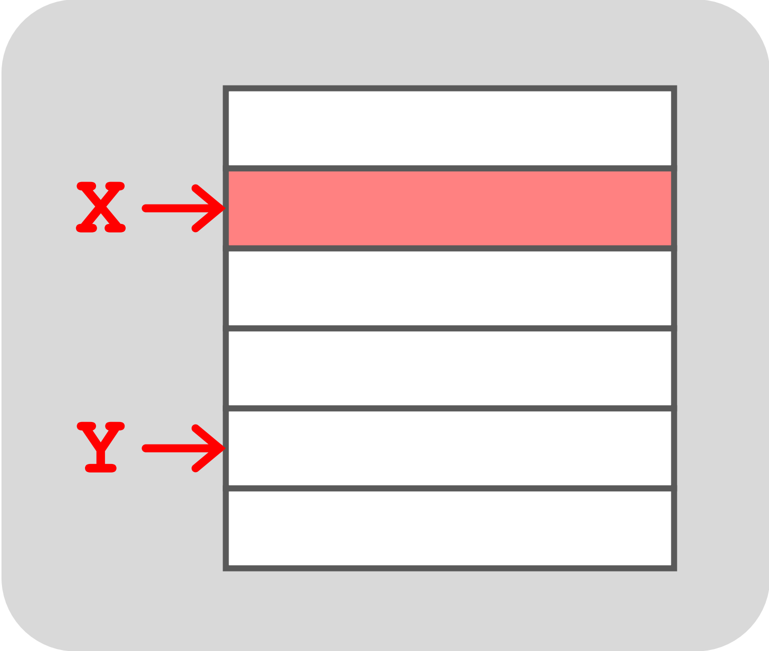


Download from: https://github.com/CMU-SAFARI/rowhammer

A Simple Program Can Induce Many Errors

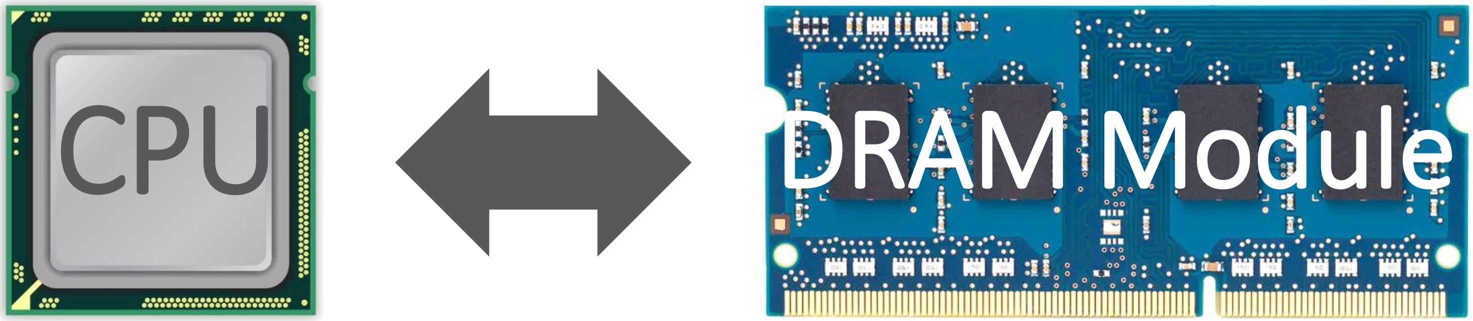


```
loop:
   mov (X), %eax
   mov (Y), %ebx
   clflush (X)
   clflush (Y)
   mfence
   jmp loop
```

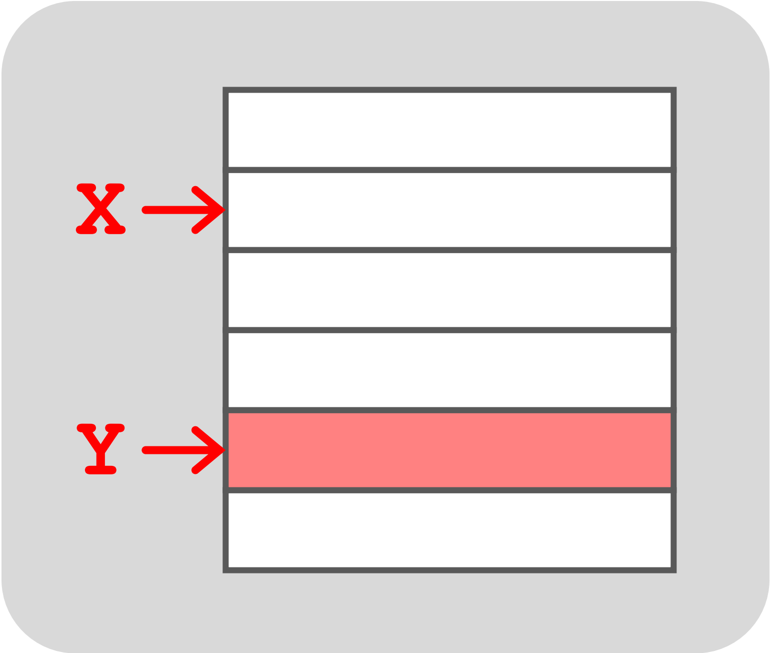


Download from: https://github.com/CMU-SAFARI/rowhammer

A Simple Program Can Induce Many Errors

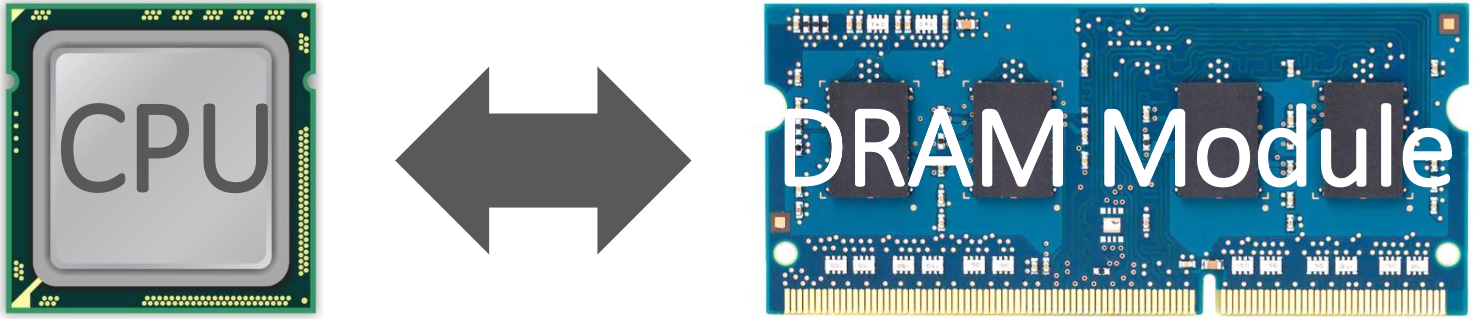


```
loop:
   mov (X), %eax
   mov (Y), %ebx
   clflush (X)
   clflush (Y)
   mfence
   jmp loop
```

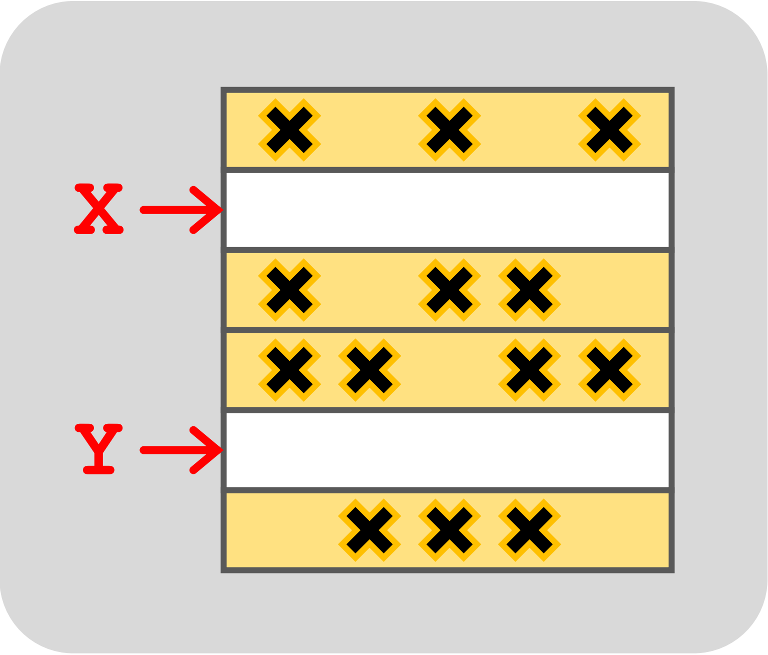


Download from: https://github.com/CMU-SAFARI/rowhammer

A Simple Program Can Induce Many Errors



```
loop:
   mov  (X), %eax
   mov  (Y), %ebx
   clflush (X)
   clflush (Y)
   mfence
   jmp loop
```



Download from: https://github.com/CMU-SAFARI/rowhammer

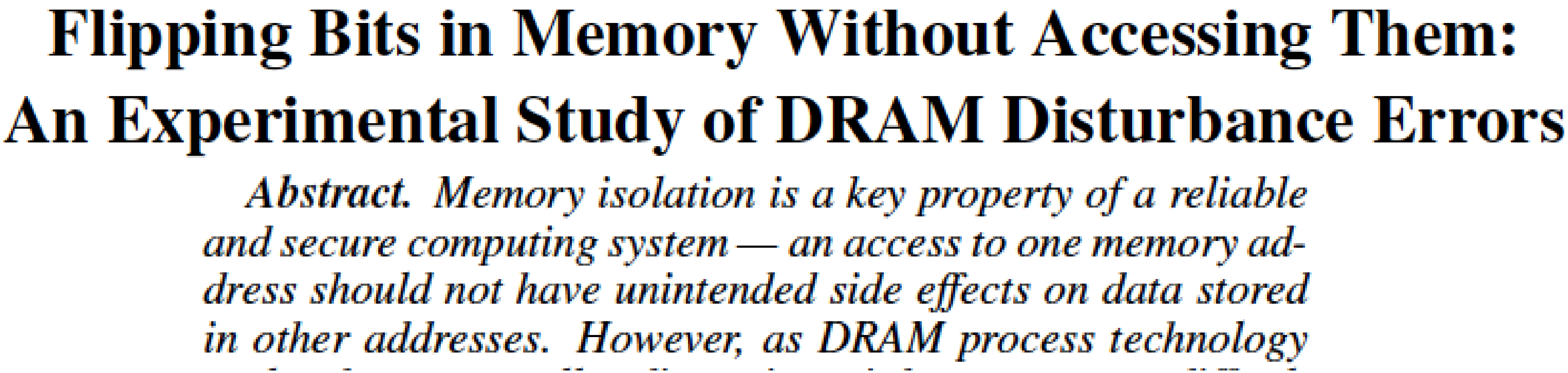
Observed Errors in Real Systems

CPU Architecture	Errors	Access-Rate
Intel Haswell (2013)	22.9K	12.3M/sec
Intel Ivy Bridge (2012)	20.7K	11.7M/sec
Intel Sandy Bridge (2011)	16.1K	11.6M/sec
AMD Piledriver (2012)	59	6.1M/sec

A real reliability & security issue

Kim+, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," ISCA 2014.

One Can Take Over an Otherwise-Secure System



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

Abstract. *Memory isolation is a key property of a reliable and secure computing system — an access to one memory address should not have unintended side effects on data stored in other addresses. However, as DRAM process technology*

Project Zero

News and updates from the Project Zero team at Google

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

Exploiting the DRAM rowhammer bug to gain kernel privileges (Seaborn, 2015)

Monday, March 9, 2015

Exploiting the DRAM rowhammer bug to gain kernel privileges

RowHammer Security Attack Example

- "Rowhammer" is a problem with some recent DRAM devices in which repeatedly accessing a row of memory can cause bit flips in adjacent rows (Kim et al., ISCA 2014).
  - Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors (Kim et al., ISCA 2014)

- We tested a selection of laptops and found that a subset of them exhibited the problem.
- We built two working privilege escalation exploits that use this effect.
  - Exploiting the DRAM rowhammer bug to gain kernel privileges (Seaborn+, 2015)

- One exploit uses rowhammer-induced bit flips to gain kernel privileges on x86-64 Linux when run as an unprivileged userland process.
- When run on a machine vulnerable to the rowhammer problem, the process was able to induce bit flips in page table entries (PTEs).
- It was able to use this to gain write access to its own page table, and hence gain read-write access to all of physical memory.

Exploiting the DRAM rowhammer bug to gain kernel privileges (Seaborn & Dullien, 2015)

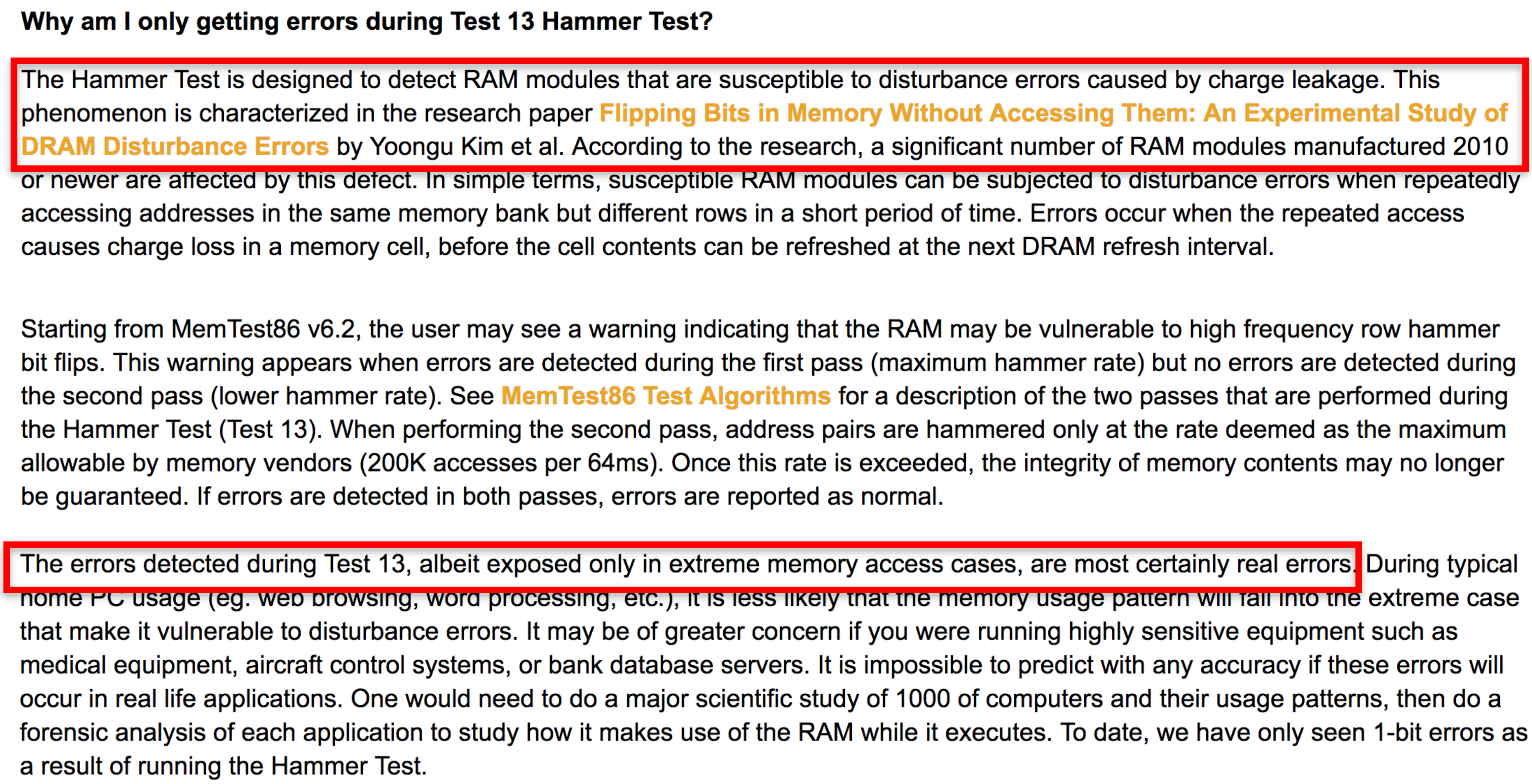
Security Implications

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

Selected Readings on RowHammer (XI.A)

- PassMark Software, memtest86, since 2014
  - https://www.memtest86.com/troubleshooting.htm#hammer



Why am I only getting errors during Test 13 Hammer Test?

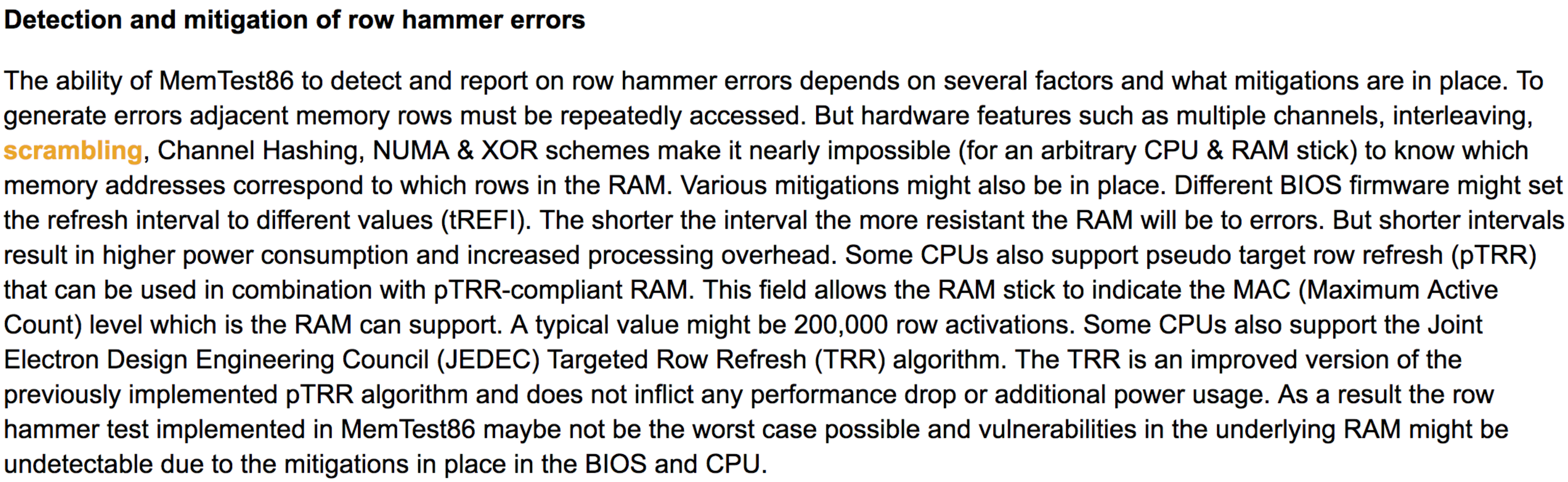
The Hammer Test is designed to detect RAM modules that are susceptible to disturbance errors caused by charge leakage. This phenomenon is characterized in the research paper **Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors** by Yoongu Kim et al. According to the research, a significant number of RAM modules manufactured 2010 or newer are affected by this defect. In simple terms, susceptible RAM modules can be subjected to disturbance errors when repeatedly accessing addresses in the same memory bank but different rows in a short period of time. Errors occur when the repeated access causes charge loss in a memory cell, before the cell contents can be refreshed at the next DRAM refresh interval.

Starting from MemTest86 v6.2, the user may see a warning indicating that the RAM may be vulnerable to high frequency row hammer bit flips. This warning appears when errors are detected during the first pass (maximum hammer rate) but no errors are detected during the second pass (lower hammer rate). See **MemTest86 Test Algorithms** for a description of the two passes that are performed during the Hammer Test (Test 13). When performing the second pass, address pairs are hammered only at the rate deemed as the maximum allowable by memory vendors (200K accesses per 64ms). Once this rate is exceeded, the integrity of memory contents may no longer be guaranteed. If errors are detected in both passes, errors are reported as normal.

The errors detected during Test 13, albeit exposed only in extreme memory access cases, are most certainly real errors. During typical home PC usage (eg. web browsing, word processing, etc.), it is less likely that the memory usage pattern will fall into the extreme case that make it vulnerable to disturbance errors. It may be of greater concern if you were running highly sensitive equipment such as medical equipment, aircraft control systems, or bank database servers. It is impossible to predict with any accuracy if these errors will occur in real life applications. One would need to do a major scientific study of 1000 of computers and their usage patterns, then do a forensic analysis of each application to study how it makes use of the RAM while it executes. To date, we have only seen 1-bit errors as a result of running the Hammer Test.

Selected Readings on RowHammer (XI.B)

- PassMark Software, memtest86, since 2014
  - https://www.memtest86.com/troubleshooting.htm#hammer



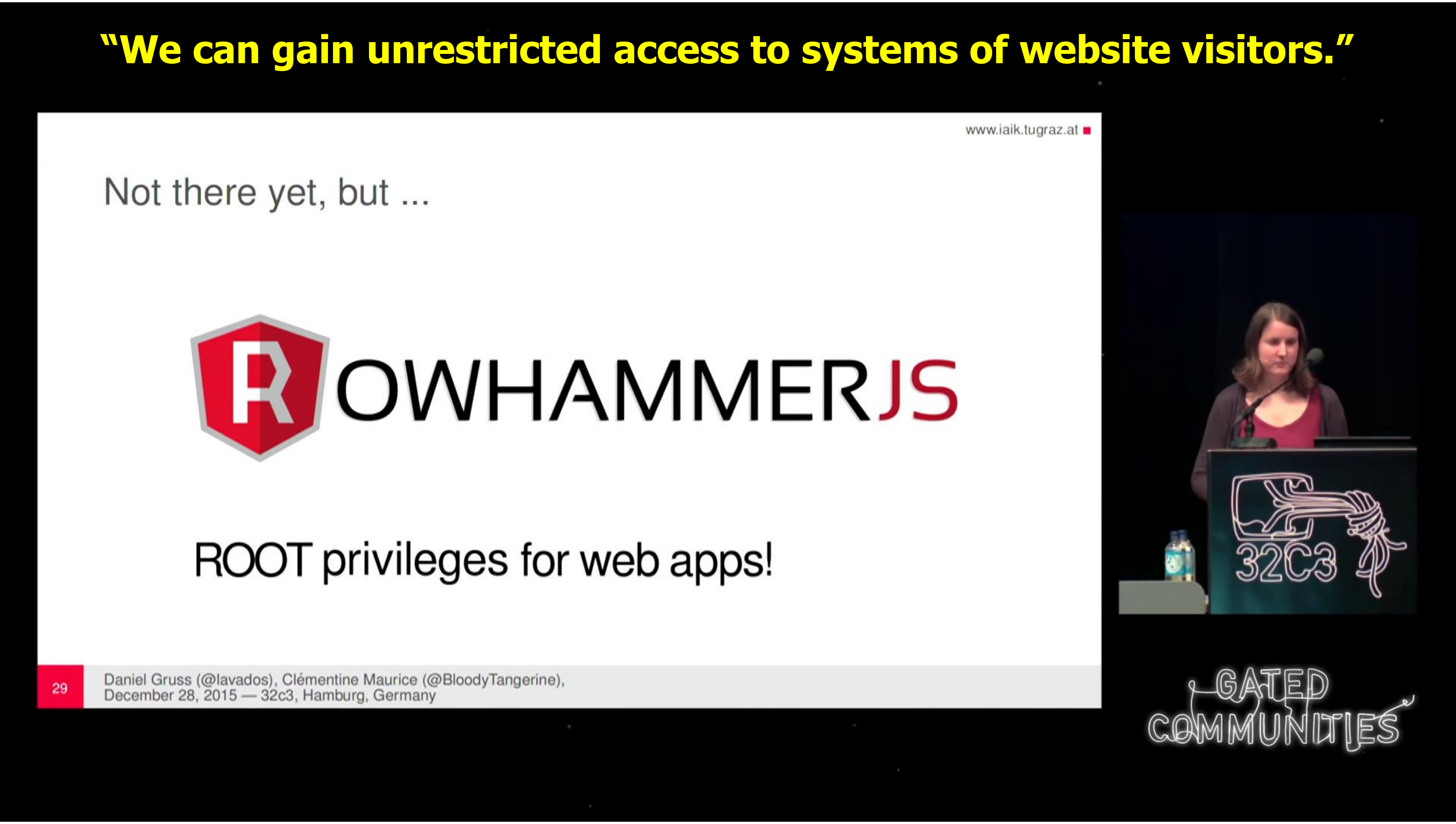
Detection and mitigation of row hammer errors

The ability of MemTest86 to detect and report on row hammer errors depends on several factors and what mitigations are in place. To generate errors adjacent memory rows must be repeatedly accessed. But hardware features such as multiple channels, interleaving, scrambling, Channel Hashing, NUMA & XOR schemes make it nearly impossible (for an arbitrary CPU & RAM stick) to know which memory addresses correspond to which rows in the RAM. Various mitigations might also be in place. Different BIOS firmware might set the refresh interval to different values (tREFI). The shorter the interval the more resistant the RAM will be to errors. But shorter intervals result in higher power consumption and increased processing overhead. Some CPUs also support pseudo target row refresh (pTRR) that can be used in combination with pTRR-compliant RAM. This field allows the RAM stick to indicate the MAC (Maximum Active Count) level which is the RAM can support. A typical value might be 200,000 row activations. Some CPUs also support the Joint Electron Design Engineering Council (JEDEC) Targeted Row Refresh (TRR) algorithm. The TRR is an improved version of the previously implemented pTRR algorithm and does not inflict any performance drop or additional power usage. As a result the row hammer test implemented in MemTest86 maybe not be the worst case possible and vulnerabilities in the underlying RAM might be undetectable due to the mitigations in place in the BIOS and CPU.

Security Implications (ISCA 2014)

- *Breach of memory protection*
  - OS page (4KB) fits inside DRAM row (8KB)
  - Adjacent DRAM row → Different OS page
- *Vulnerability: disturbance attack*
  - By accessing its own page, a program could corrupt pages belonging to another program
- *We constructed a proof-of-concept*
  - Using only user-level instructions

More Security Implications (I)



“We can gain unrestricted access to systems of website visitors.”

www.iaik.tugraz.at

Not there yet, but ...

ROWHAMMERJS

ROOT privileges for web apps!

29

Daniel Gruss (@lavados), Clémentine Maurice (@BloodyTangerine),
December 28, 2015 — 32c3, Hamburg, Germany

Rowhammer.js: A Remote Software-Induced Fault Attack in JavaScript (DIMVA’16)

Source: https://lab.dsst.io/32c3-slides/7197.html

More Security Implications (II)

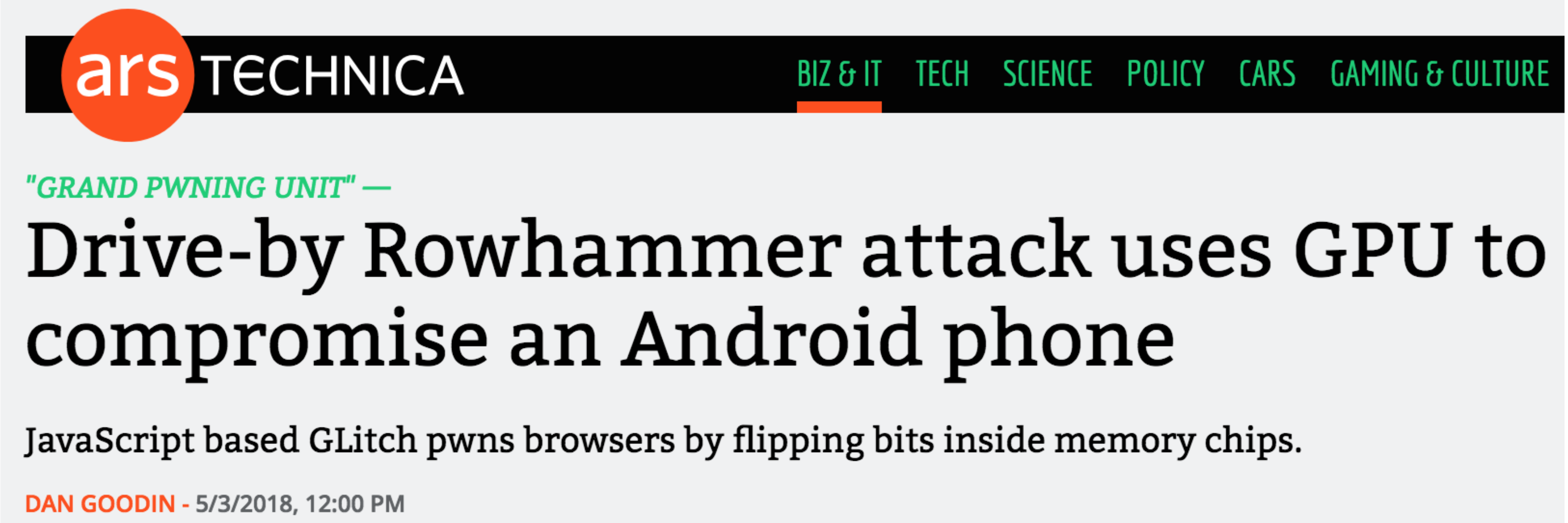
"Can gain control of a smart phone deterministically"

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

Source: https://fossbytes.com/drammer-rowhammer-attack-android-root-devices/

More Security Implications (III)

- Using an integrated GPU in a mobile system to remotely escalate privilege via the WebGL interface. IEEE S&P 2018

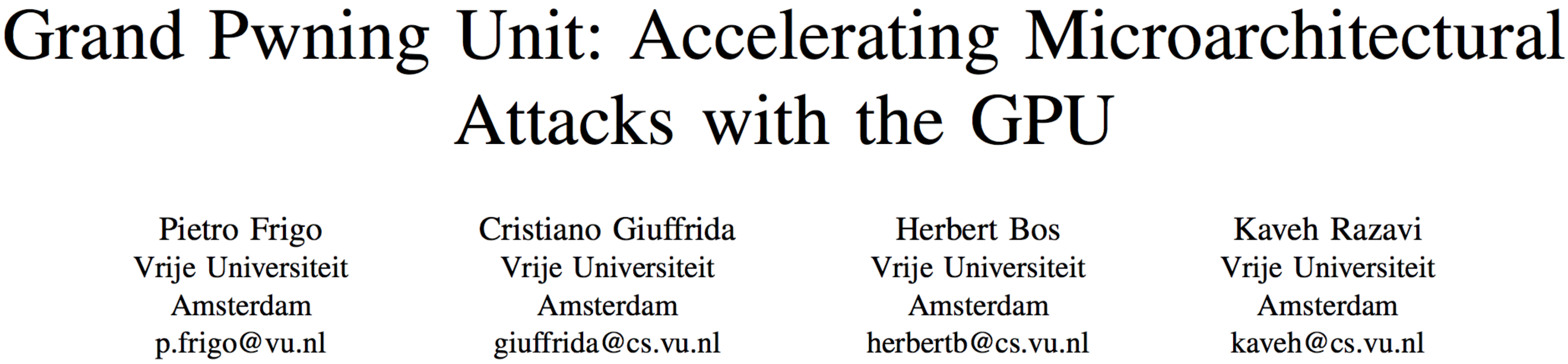


"GRAND PWNING UNIT" —

Drive-by Rowhammer attack uses GPU to compromise an Android phone

JavaScript based GLitch pwns browsers by flipping bits inside memory chips.

DAN GOODIN - 5/3/2018, 12:00 PM

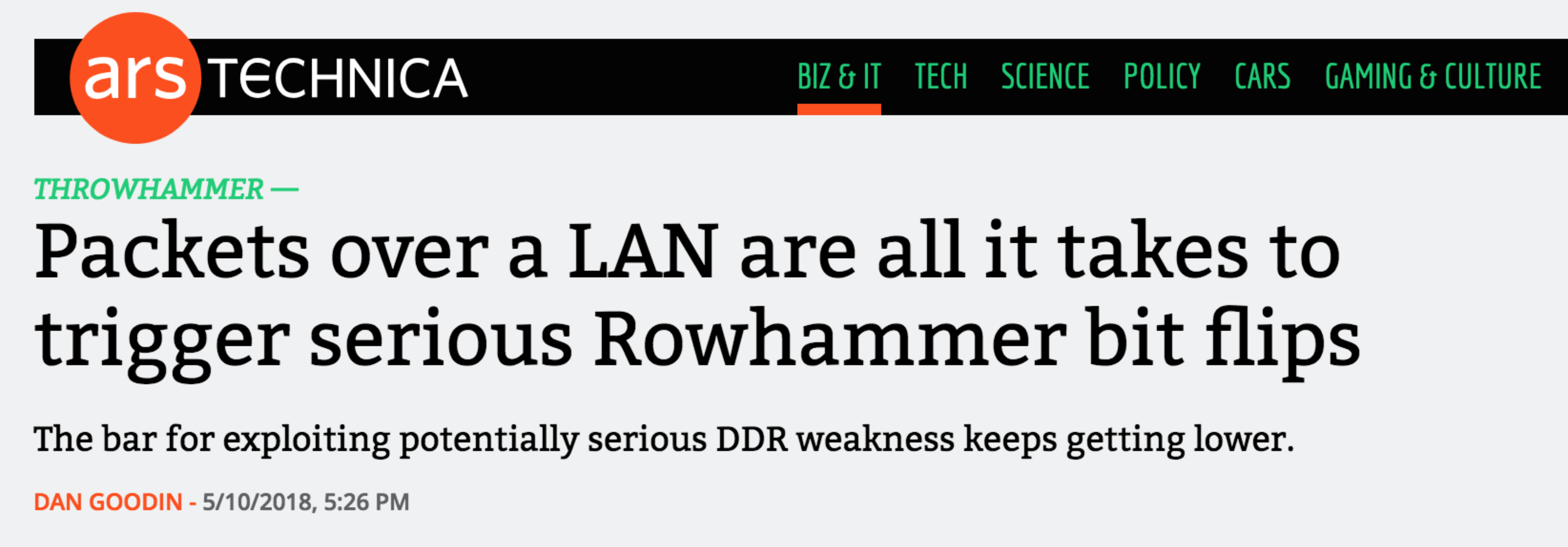


Grand Pwning Unit: Accelerating Microarchitectural Attacks with the GPU

Pietro Frigo	Cristiano Giuffrida	Herbert Bos	Kaveh Razavi
Vrije Universiteit Amsterdam	Vrije Universiteit Amsterdam	Vrije Universiteit Amsterdam	Vrije Universiteit Amsterdam
p.frigo@vu.nl	giuffrida@cs.vu.nl	herbertb@cs.vu.nl	kaveh@cs.vu.nl

More Security Implications (IV)

- Rowhammer over RDMA (I) USENIX ATC 2018

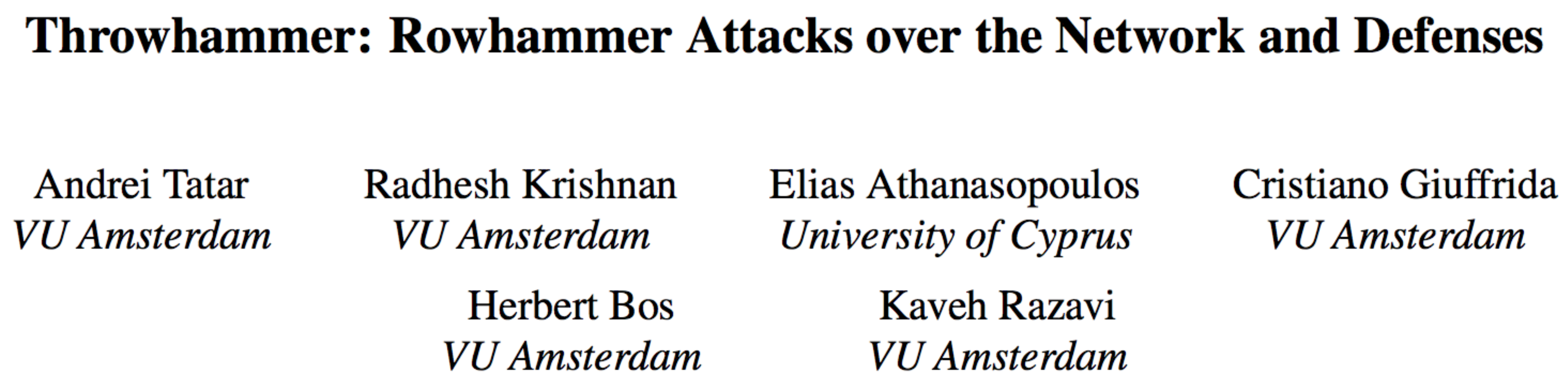


THROWHAMMER —

Packets over a LAN are all it takes to trigger serious Rowhammer bit flips

The bar for exploiting potentially serious DDR weakness keeps getting lower.

DAN GOODIN - 5/10/2018, 5:26 PM



Throwhammer: Rowhammer Attacks over the Network and Defenses

Andrei Tatar
VU Amsterdam

Radhesh Krishnan
VU Amsterdam

Elias Athanasopoulos
University of Cyprus

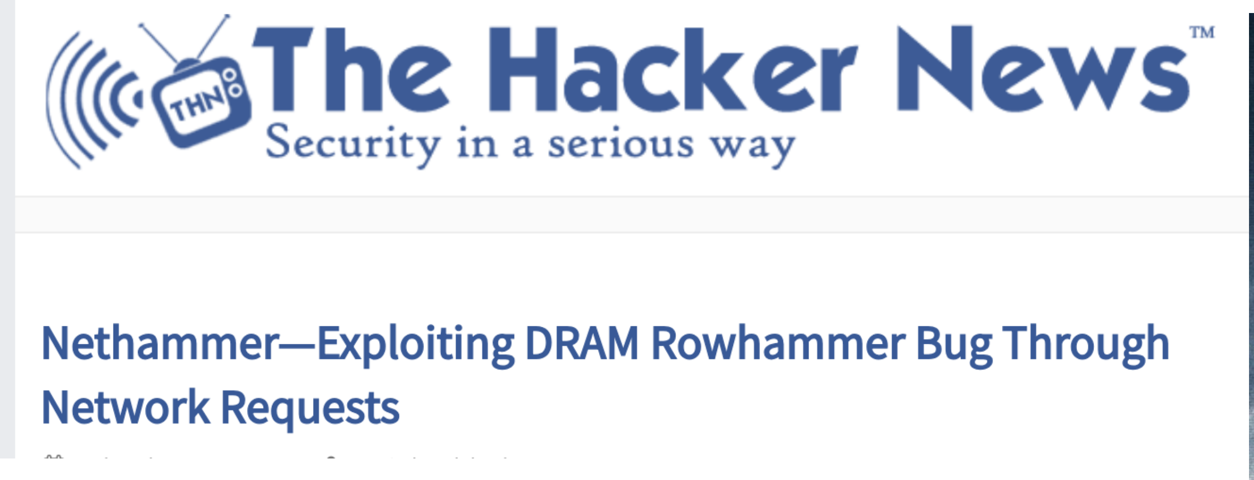
Cristiano Giuffrida
VU Amsterdam

Herbert Bos
VU Amsterdam

Kaveh Razavi
VU Amsterdam

More Security Implications (V)

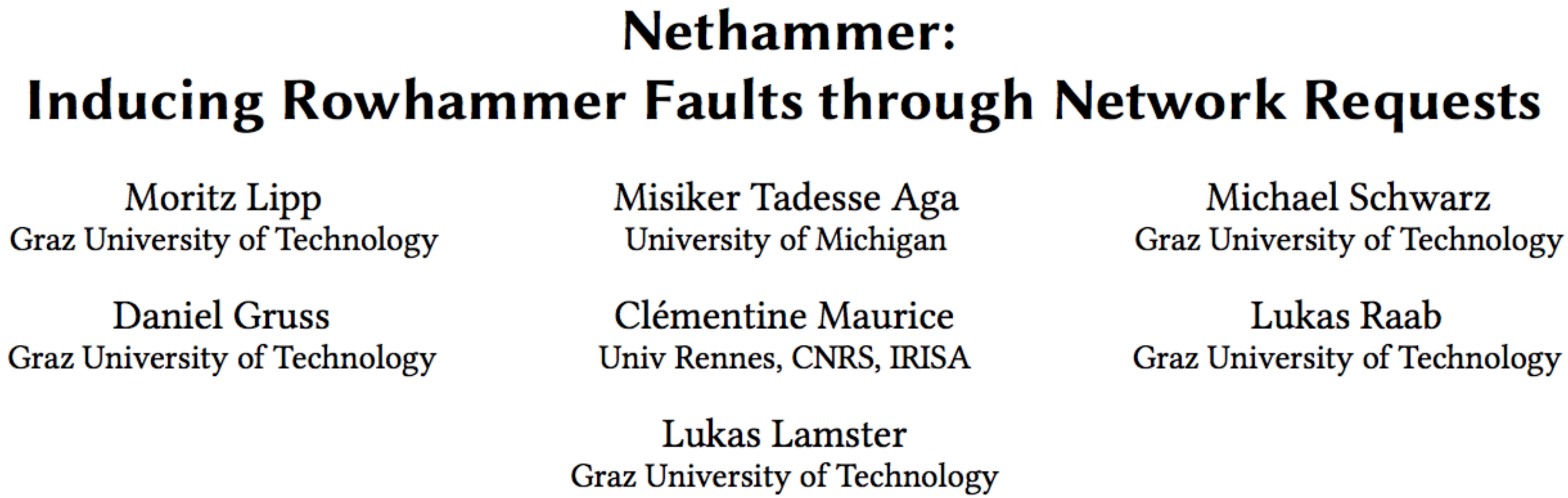
- Rowhammer over RDMA (II)



The Hacker News™

Security in a serious way

Nethammer—Exploiting DRAM Rowhammer Bug Through Network Requests



Nethammer: Inducing Rowhammer Faults through Network Requests

Moritz Lipp
Graz University of Technology

Misiker Tadesse Aga
University of Michigan

Michael Schwarz
Graz University of Technology

Daniel Gruss
Graz University of Technology

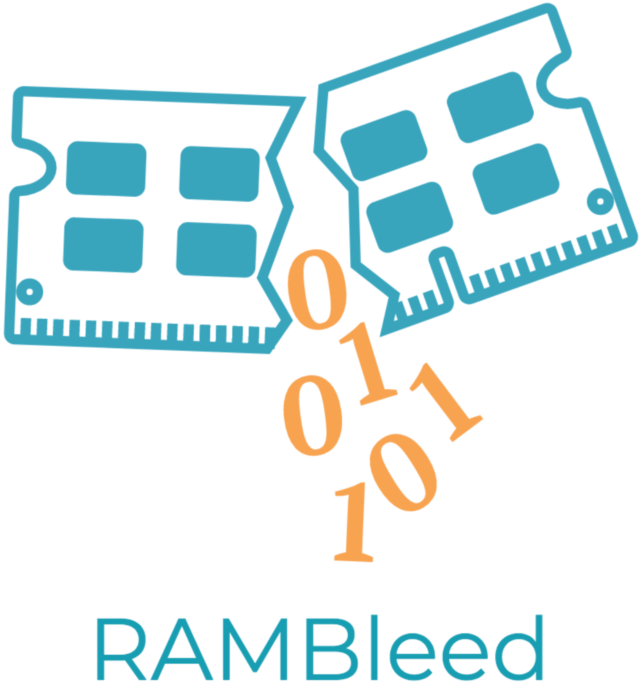
Clémentine Maurice
Univ Rennes, CNRS, IRISA

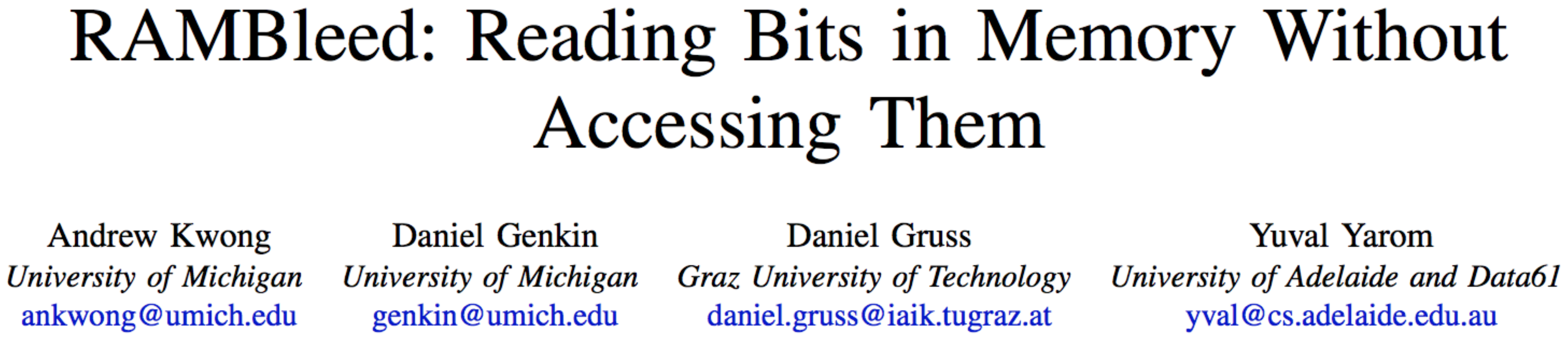
Lukas Raab
Graz University of Technology

Lukas Lamster
Graz University of Technology

More Security Implications (VI)

- IEEE S&P 2020





RAMBleed: Reading Bits in Memory Without Accessing Them

Andrew Kwong
University of Michigan
ankwong@umich.edu

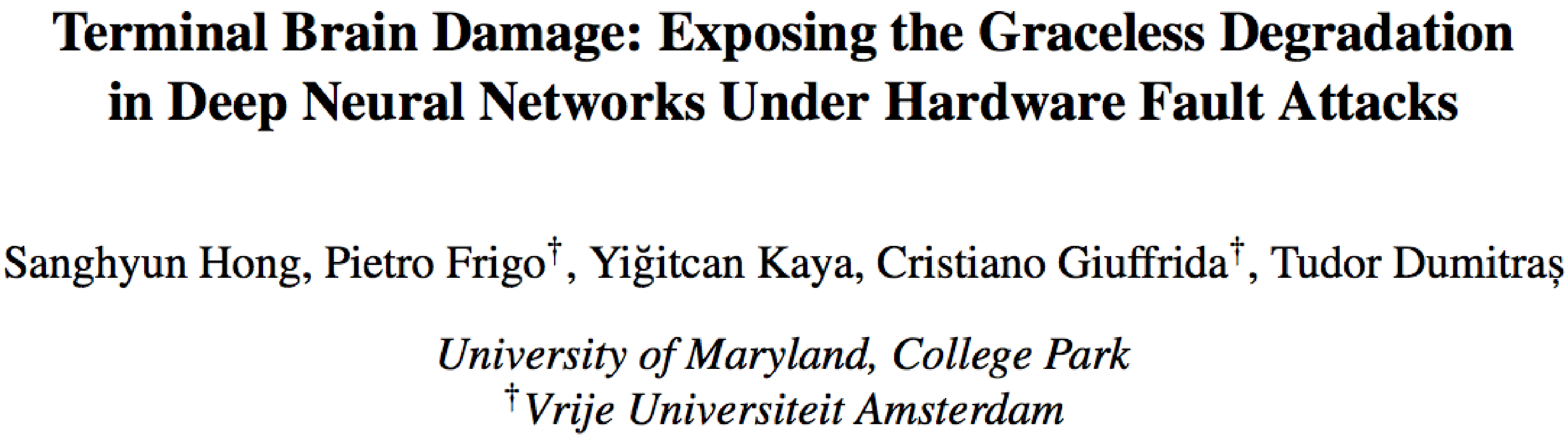
Daniel Genkin
University of Michigan
genkin@umich.edu

Daniel Gruss
Graz University of Technology
daniel.gruss@iaik.tugraz.at

Yuval Yarom
University of Adelaide and Data61
yval@cs.adelaide.edu.au

More Security Implications (VII)

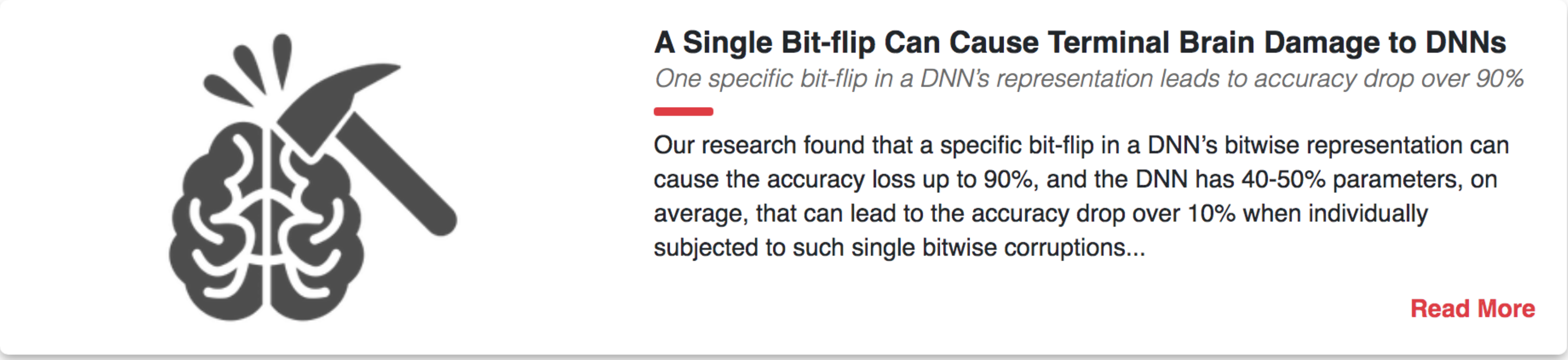
- USENIX Security 2019



Terminal Brain Damage: Exposing the Graceless Degradation in Deep Neural Networks Under Hardware Fault Attacks

Sanghyun Hong, Pietro Frigo[†], Yiğitcan Kaya, Cristiano Giuffrida[†], Tudor Dumitraș

University of Maryland, College Park
[†]Vrije Universiteit Amsterdam



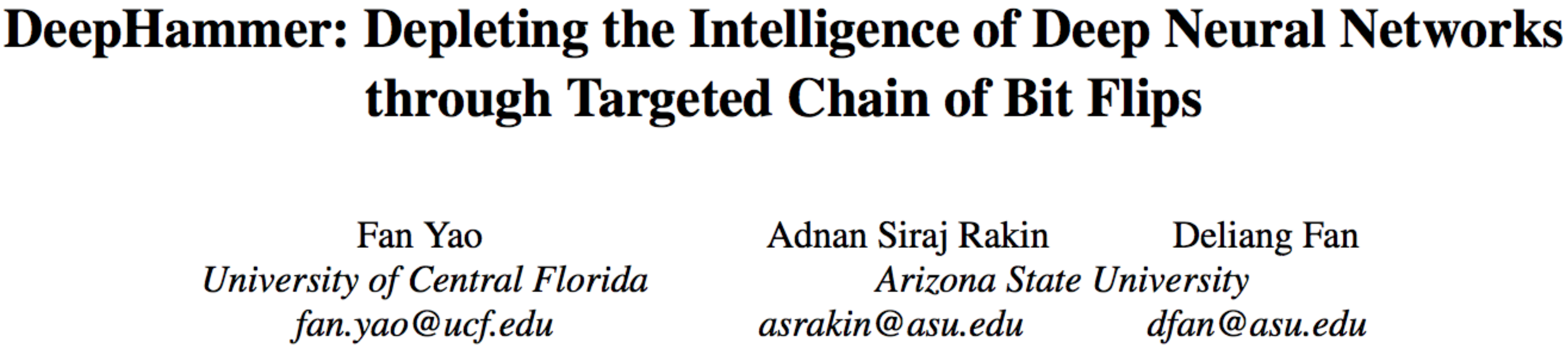
A Single Bit-flip Can Cause Terminal Brain Damage to DNNs

One specific bit-flip in a DNN's representation leads to accuracy drop over 90%

Our research found that a specific bit-flip in a DNN's bitwise representation can cause the accuracy loss up to 90%, and the DNN has 40-50% parameters, on average, that can lead to the accuracy drop over 10% when individually subjected to such single bitwise corruptions...

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
asrakin@asu.edu

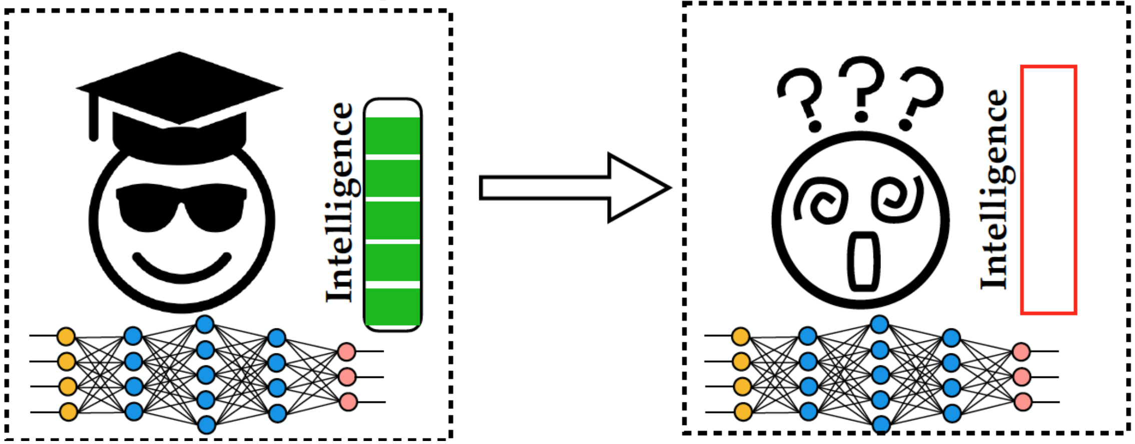
Deliang Fan
dfan@asu.edu

Arizona State University

Degrade the **inference accuracy** to the level of **Random Guess**

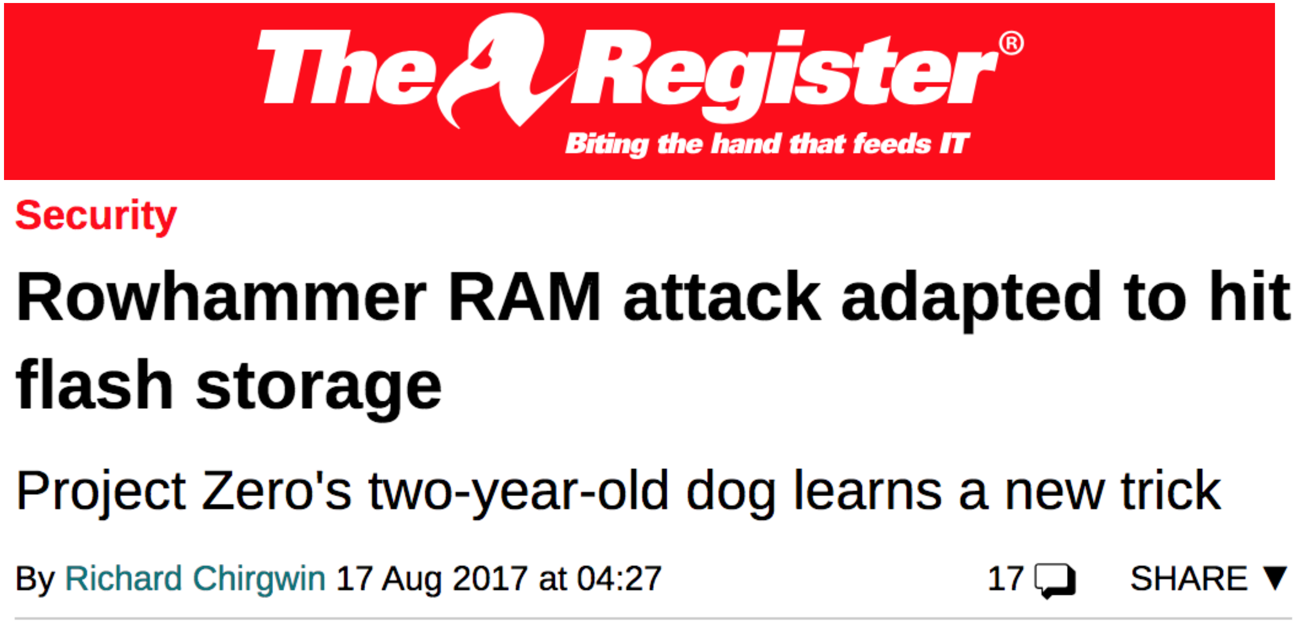
Example: ResNet-20 for CIFAR-10, **10** output classes

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



More Security Implications (IX)

- Rowhammer on MLC NAND Flash (based on [Cai+, HPCA 2017])



The Register®

Biting the hand that feeds IT

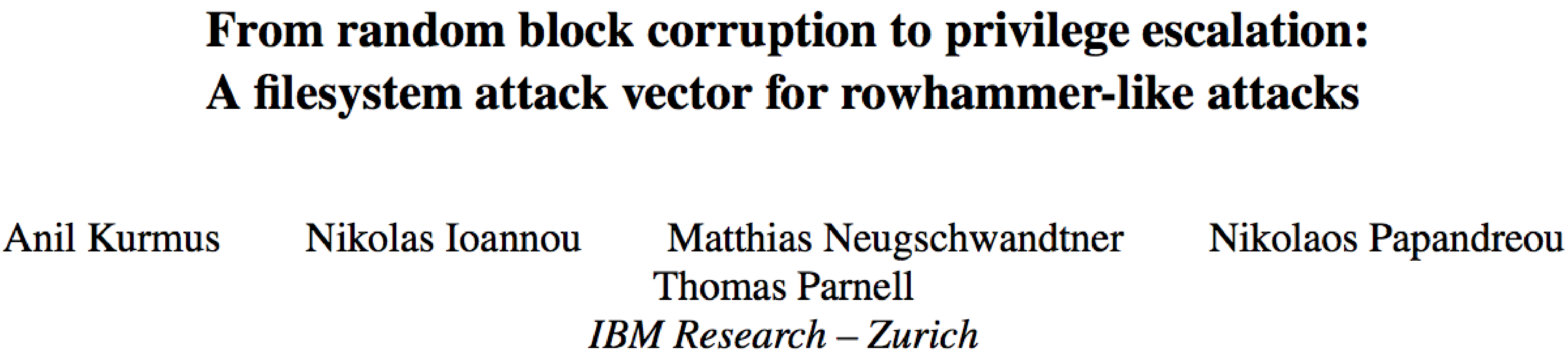
Security

Rowhammer RAM attack adapted to hit flash storage

Project Zero's two-year-old dog learns a new trick

By Richard Chirgwin 17 Aug 2017 at 04:27

17 SHARE ▼



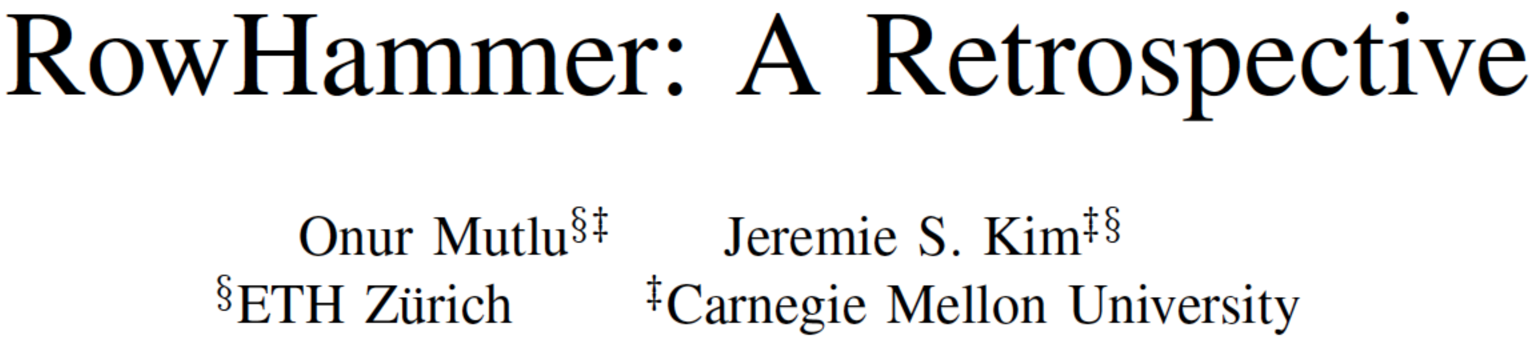
From random block corruption to privilege escalation: A filesystem attack vector for rowhammer-like attacks

Anil Kurmus Nikolas Ioannou Matthias Neugschwandtner Nikolaos Papandreou
Thomas Parnell
IBM Research – Zurich

More Security Implications?

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)]



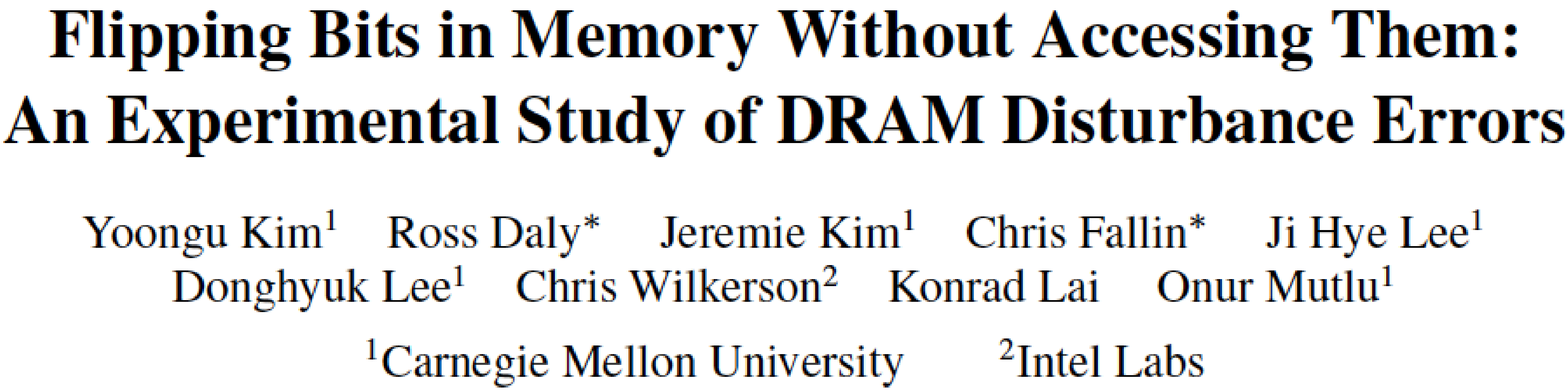
RowHammer: A Retrospective

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

Understanding RowHammer

First RowHammer Analysis

- 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).

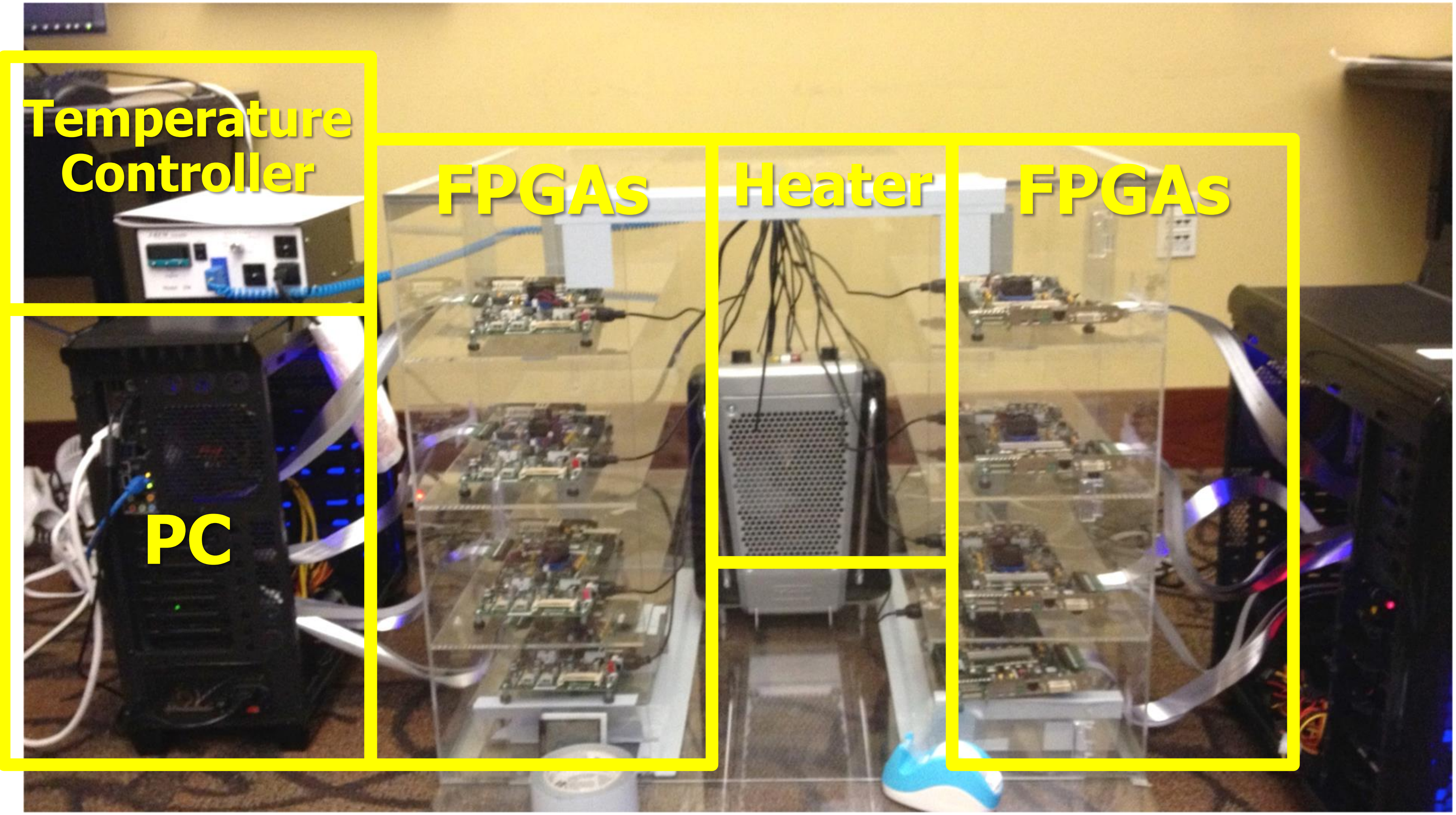


Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors

Yoongu Kim[1] Ross Daly* Jeremie Kim[1] Chris Fallin* Ji Hye Lee[1]
Donghyuk Lee[1] Chris Wilkerson[2] Konrad Lai Onur Mutlu[1]

[1]Carnegie Mellon University [2]Intel Labs

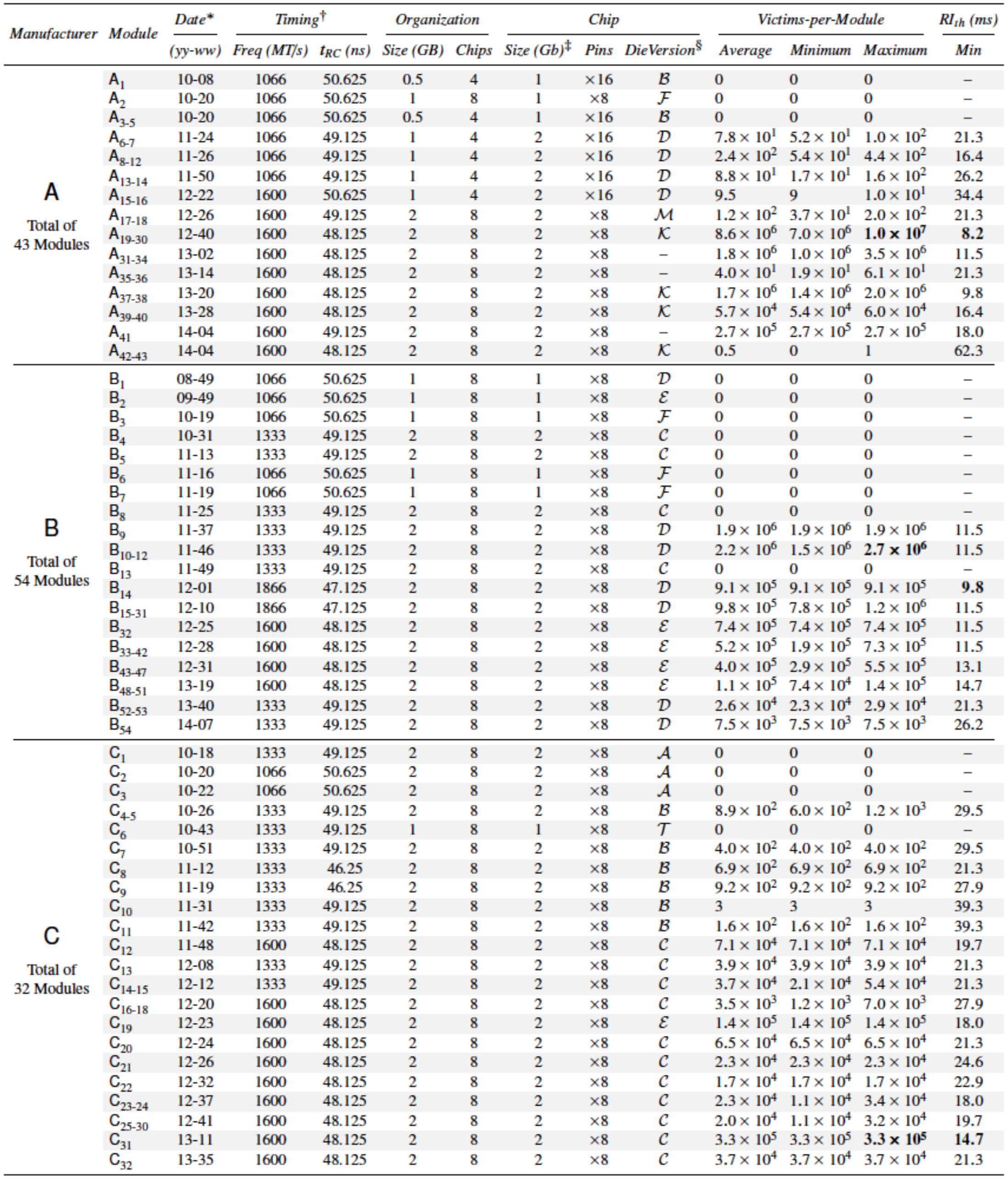
RowHammer Infrastructure (2012-2014)



Kim+, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," ISCA 2014.

Tested DRAM Modules from 2008-2014

(129 total)



Manufacturer	Module	*Date**	*Timing†*		*Organization*		*Chip*			*Victims-per-Module*			*RI_{th} (ms)*
		(yy-ww)	*Freq (MT/s)*	*t_{RC} (ns)*	*Size (GB)*	*Chips*	*Size (Gb)‡*	*Pins*	*DieVersion§*	*Average*	*Minimum*	*Maximum*	*Min*
A Total of 43 Modules	A_{1}	10-08	1066	50.625	0.5	4	1	×16	$\mathcal{B}$	0	0	0	–
	A_{2}	10-20	1066	50.625	1	8	1	×8	$\mathcal{F}$	0	0	0	–
	$A_{3\text{-}5}$	10-20	1066	50.625	0.5	4	1	×16	$\mathcal{B}$	0	0	0	–
	$A_{6\text{-}7}$	11-24	1066	49.125	1	4	2	×16	$\mathcal{D}$	7.8×10^{1}	5.2×10^{1}	1.0×10^{2}	21.3
	$A_{8\text{-}12}$	11-26	1066	49.125	1	4	2	×16	$\mathcal{D}$	2.4×10^{2}	5.4×10^{1}	4.4×10^{2}	16.4
	$A_{13\text{-}14}$	11-50	1066	49.125	1	4	2	×16	$\mathcal{D}$	8.8×10^{1}	1.7×10^{1}	1.6×10^{2}	26.2
	$A_{15\text{-}16}$	12-22	1600	50.625	1	4	2	×16	$\mathcal{D}$	9.5	9	1.0×10^{1}	34.4
	$A_{17\text{-}18}$	12-26	1600	49.125	2	8	2	×8	$\mathcal{M}$	1.2×10^{2}	3.7×10^{1}	2.0×10^{2}	21.3
	$A_{19\text{-}30}$	12-40	1600	48.125	2	8	2	×8	$\mathcal{K}$	8.6×10^{6}	7.0×10^{6}	$\mathbf{1.0\times10^{7}}$	**8.2**
	$A_{31\text{-}34}$	13-02	1600	48.125	2	8	2	×8	–	1.8×10^{6}	1.0×10^{6}	3.5×10^{6}	11.5
	$A_{35\text{-}36}$	13-14	1600	48.125	2	8	2	×8	–	4.0×10^{1}	1.9×10^{1}	6.1×10^{1}	21.3
	$A_{37\text{-}38}$	13-20	1600	48.125	2	8	2	×8	$\mathcal{K}$	1.7×10^{6}	1.4×10^{6}	2.0×10^{6}	9.8
	$A_{39\text{-}40}$	13-28	1600	48.125	2	8	2	×8	$\mathcal{K}$	5.7×10^{4}	5.4×10^{4}	6.0×10^{4}	16.4
	A_{41}	14-04	1600	49.125	2	8	2	×8	–	2.7×10^{5}	2.7×10^{5}	2.7×10^{5}	18.0
	$A_{42\text{-}43}$	14-04	1600	48.125	2	8	2	×8	$\mathcal{K}$	0.5	0	1	62.3
B Total of 54 Modules	B_{1}	08-49	1066	50.625	1	8	1	×8	$\mathcal{D}$	0	0	0	–
	B_{2}	09-49	1066	50.625	1	8	1	×8	$\mathcal{E}$	0	0	0	–
	B_{3}	10-19	1066	50.625	1	8	1	×8	$\mathcal{F}$	0	0	0	–
	B_{4}	10-31	1333	49.125	2	8	2	×8	$\mathcal{C}$	0	0	0	–
	B_{5}	11-13	1333	49.125	2	8	2	×8	$\mathcal{C}$	0	0	0	–
	B_{6}	11-16	1066	50.625	1	8	1	×8	$\mathcal{F}$	0	0	0	–
	B_{7}	11-19	1066	50.625	1	8	1	×8	$\mathcal{F}$	0	0	0	–
	B_{8}	11-25	1333	49.125	2	8	2	×8	$\mathcal{C}$	0	0	0	–
	B_{9}	11-37	1333	49.125	2	8	2	×8	$\mathcal{D}$	1.9×10^{6}	1.9×10^{6}	1.9×10^{6}	11.5
	$B_{10\text{-}12}$	11-46	1333	49.125	2	8	2	×8	$\mathcal{D}$	2.2×10^{6}	1.5×10^{6}	$\mathbf{2.7\times10^{6}}$	11.5
	B_{13}	11-49	1333	49.125	2	8	2	×8	$\mathcal{C}$	0	0	0	–
	B_{14}	12-01	1866	47.125	2	8	2	×8	$\mathcal{D}$	9.1×10^{5}	9.1×10^{5}	9.1×10^{5}	**9.8**
	$B_{15\text{-}31}$	12-10	1866	47.125	2	8	2	×8	$\mathcal{D}$	9.8×10^{5}	7.8×10^{5}	1.2×10^{6}	11.5
	B_{32}	12-25	1600	48.125	2	8	2	×8	$\mathcal{E}$	7.4×10^{5}	7.4×10^{5}	7.4×10^{5}	11.5
	$B_{33\text{-}42}$	12-28	1600	48.125	2	8	2	×8	$\mathcal{E}$	5.2×10^{5}	1.9×10^{5}	7.3×10^{5}	11.5
	$B_{43\text{-}47}$	12-31	1600	48.125	2	8	2	×8	$\mathcal{E}$	4.0×10^{5}	2.9×10^{5}	5.5×10^{5}	13.1
	$B_{48\text{-}51}$	13-19	1600	48.125	2	8	2	×8	$\mathcal{E}$	1.1×10^{5}	7.4×10^{4}	1.4×10^{5}	14.7
	$B_{52\text{-}53}$	13-40	1333	49.125	2	8	2	×8	$\mathcal{D}$	2.6×10^{4}	2.3×10^{4}	2.9×10^{4}	21.3
	B_{54}	14-07	1333	49.125	2	8	2	×8	$\mathcal{D}$	7.5×10^{3}	7.5×10^{3}	7.5×10^{3}	26.2
C Total of 32 Modules	C_{1}	10-18	1333	49.125	2	8	2	×8	$\mathcal{A}$	0	0	0	–
	C_{2}	10-20	1066	50.625	2	8	2	×8	$\mathcal{A}$	0	0	0	–
	C_{3}	10-22	1066	50.625	2	8	2	×8	$\mathcal{A}$	0	0	0	–
	$C_{4\text{-}5}$	10-26	1333	49.125	2	8	2	×8	$\mathcal{B}$	8.9×10^{2}	6.0×10^{2}	1.2×10^{3}	29.5
	C_{6}	10-43	1333	49.125	1	8	1	×8	$\mathcal{T}$	0	0	0	–
	C_{7}	10-51	1333	49.125	2	8	2	×8	$\mathcal{B}$	4.0×10^{2}	4.0×10^{2}	4.0×10^{2}	29.5
	C_{8}	11-12	1333	46.25	2	8	2	×8	$\mathcal{B}$	6.9×10^{2}	6.9×10^{2}	6.9×10^{2}	21.3
	C_{9}	11-19	1333	46.25	2	8	2	×8	$\mathcal{B}$	9.2×10^{2}	9.2×10^{2}	9.2×10^{2}	27.9
	C_{10}	11-31	1333	49.125	2	8	2	×8	$\mathcal{B}$	3	3	3	39.3
	C_{11}	11-42	1333	49.125	2	8	2	×8	$\mathcal{B}$	1.6×10^{2}	1.6×10^{2}	1.6×10^{2}	39.3
	C_{12}	11-48	1600	48.125	2	8	2	×8	$\mathcal{C}$	7.1×10^{4}	7.1×10^{4}	7.1×10^{4}	19.7
	C_{13}	12-08	1333	49.125	2	8	2	×8	$\mathcal{C}$	3.9×10^{4}	3.9×10^{4}	3.9×10^{4}	21.3
	$C_{14\text{-}15}$	12-12	1333	49.125	2	8	2	×8	$\mathcal{C}$	3.7×10^{4}	2.1×10^{4}	5.4×10^{4}	21.3
	$C_{16\text{-}18}$	12-20	1600	48.125	2	8	2	×8	$\mathcal{C}$	3.5×10^{3}	1.2×10^{3}	7.0×10^{3}	27.9
	C_{19}	12-23	1600	48.125	2	8	2	×8	$\mathcal{E}$	1.4×10^{5}	1.4×10^{5}	1.4×10^{5}	18.0
	C_{20}	12-24	1600	48.125	2	8	2	×8	$\mathcal{C}$	6.5×10^{4}	6.5×10^{4}	6.5×10^{4}	21.3
	C_{21}	12-26	1600	48.125	2	8	2	×8	$\mathcal{C}$	2.3×10^{4}	2.3×10^{4}	2.3×10^{4}	24.6
	C_{22}	12-32	1600	48.125	2	8	2	×8	$\mathcal{C}$	1.7×10^{4}	1.7×10^{4}	1.7×10^{4}	22.9
	$C_{23\text{-}24}$	12-37	1600	48.125	2	8	2	×8	$\mathcal{C}$	2.3×10^{4}	1.1×10^{4}	3.4×10^{4}	18.0
	$C_{25\text{-}30}$	12-41	1600	48.125	2	8	2	×8	$\mathcal{C}$	2.0×10^{4}	1.1×10^{4}	3.2×10^{4}	19.7
	C_{31}	13-11	1600	48.125	2	8	2	×8	$\mathcal{C}$	3.3×10^{5}	3.3×10^{5}	$\mathbf{3.3\times10^{5}}$	**14.7**
	C_{32}	13-35	1600	48.125	2	8	2	×8	$\mathcal{C}$	3.7×10^{4}	3.7×10^{4}	3.7×10^{4}	21.3

∗ We report the manufacture date marked on the chip packages, which is more accurate than other dates that can be gleaned from a module.

† We report timing constraints stored in the module's on-board ROM [33], which is read by the system BIOS to calibrate the memory controller.

‡ The maximum DRAM chip size supported by our testing platform is 2Gb.

§ We report DRAM die versions marked on the chip packages, which typically progress in the following manner: $\mathcal{M}\to\mathcal{A}\to\mathcal{B}\to\mathcal{C}\to\cdots$.

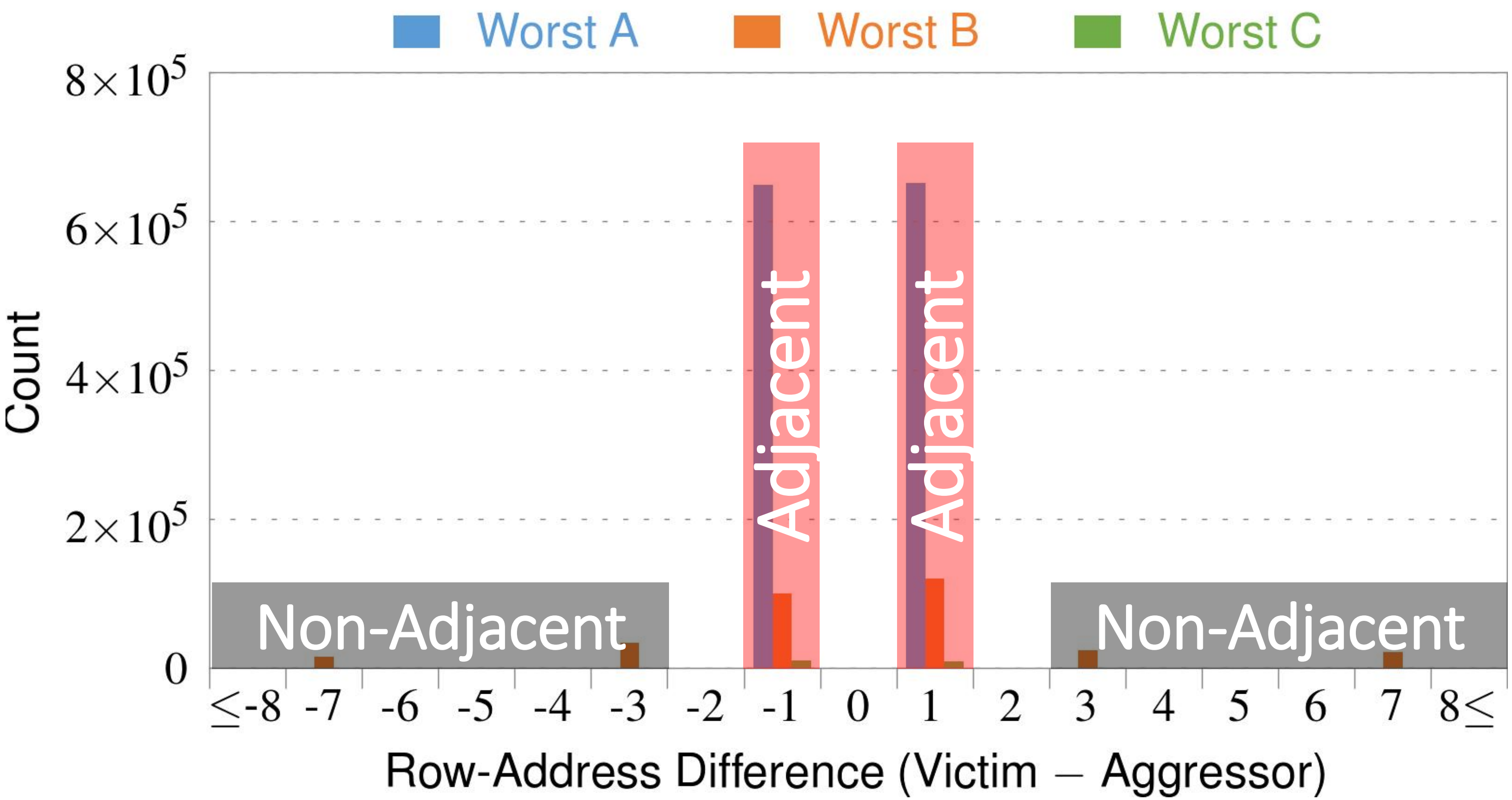
Table 3. Sample population of 129 DDR3 DRAM modules, categorized by manufacturer and sorted by manufacture date

RowHammer Characterization Results

1. Most Modules Are at Risk
2. Errors vs. Vintage
3. Error = Charge Loss
4. Adjacency: Aggressor & Victim
5. Sensitivity Studies
6. Other Results in Paper
7. Solution Space

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

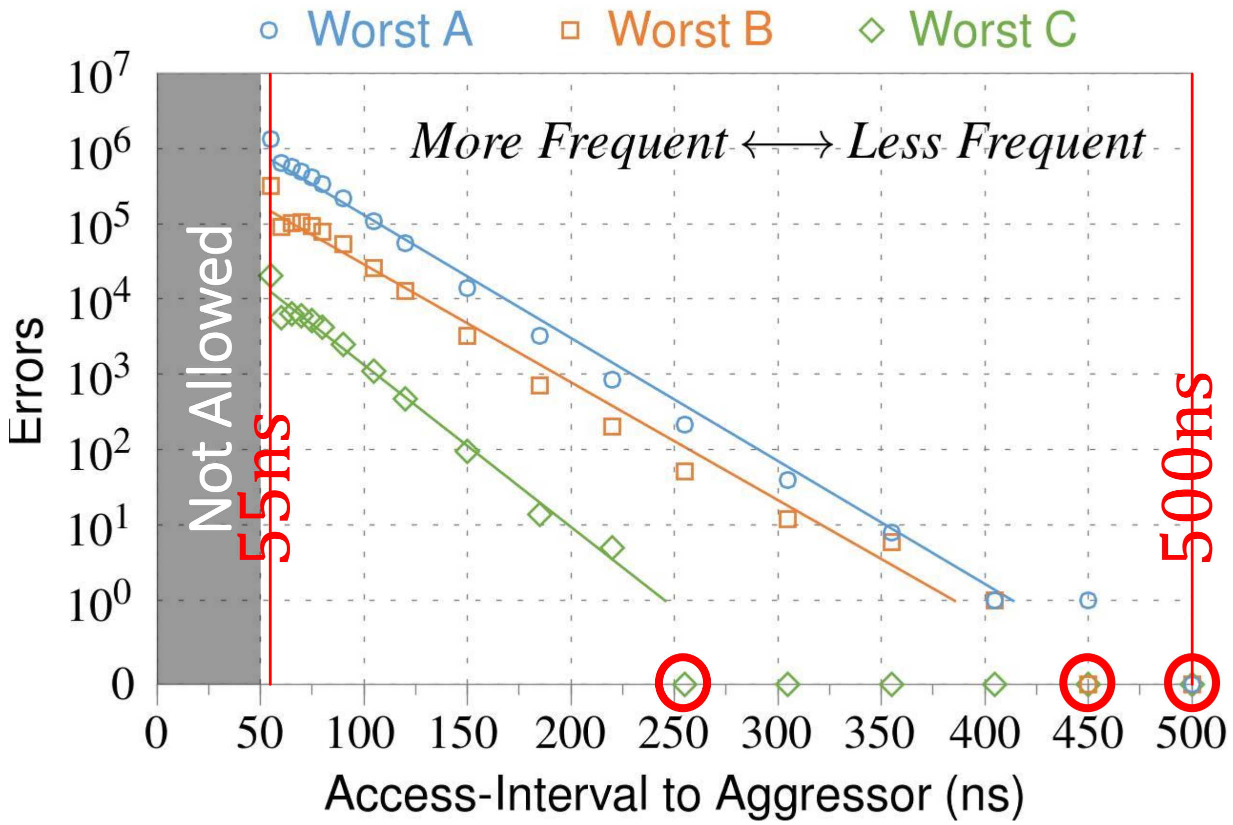
4. Adjacency: Aggressor & Victim



Note: For three modules with the most errors (only first bank)

Most aggressors & victims are adjacent

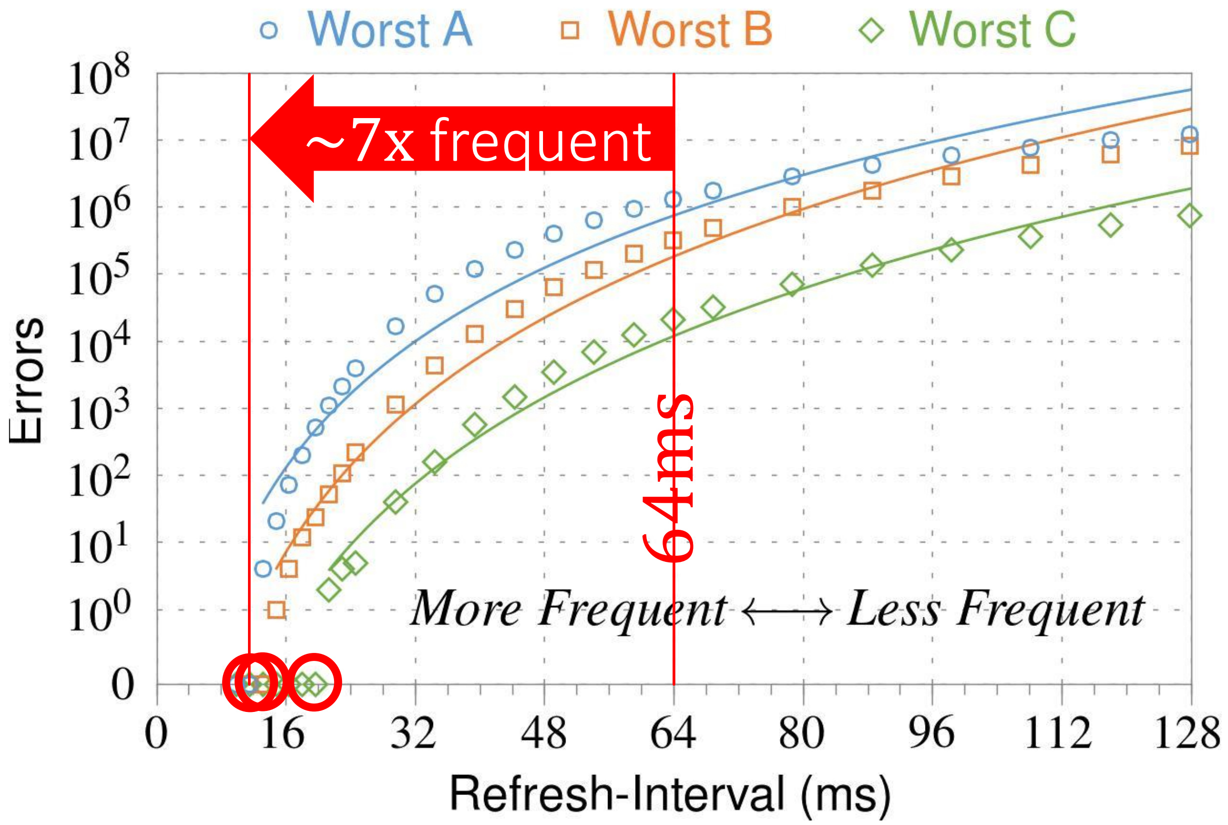
❶ Access Interval (Aggressor)



Note: For three modules with the most errors (only first bank)

Less frequent accesses → Fewer errors

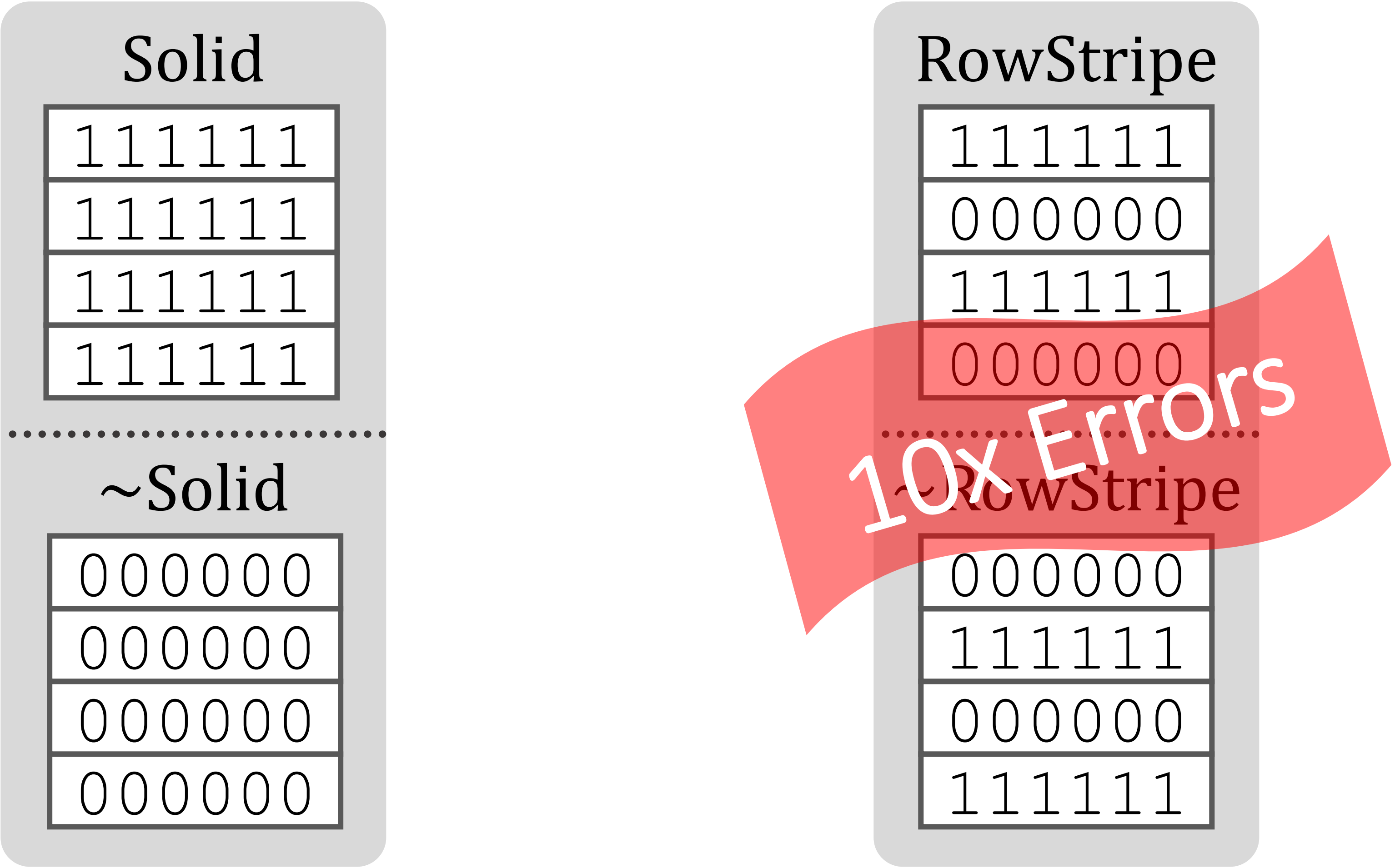
❷ Refresh Interval



Note: Using three modules with the most errors (only first bank)

More frequent refreshes → Fewer errors

❸ Data Pattern



Solid

111111
111111
111111
111111

~Solid

000000
000000
000000
000000

RowStripe

111111
000000
111111
000000

~RowStripe

000000
111111
000000
111111

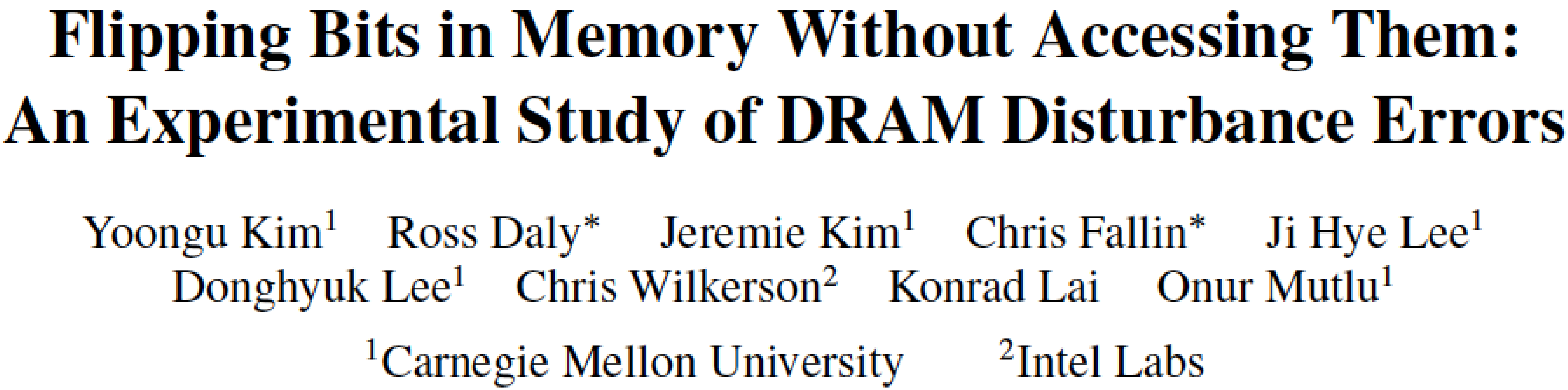
Errors affected by data stored in other cells

6. Other Key Observations [ISCA'14]

- *Victim Cells ≠ Retention-Weak Cells*
  - Almost no overlap between them
- *Errors are repeatable*
  - Across ten iterations of testing, >70% of victim cells had errors in every iteration
- *As many as 4 errors per cache-line*
  - Simple ECC (e.g., SECDED) cannot prevent all errors
- *Cells affected by two aggressors on either side*
  - Double sided hammering

Major RowHammer Characteristics (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).



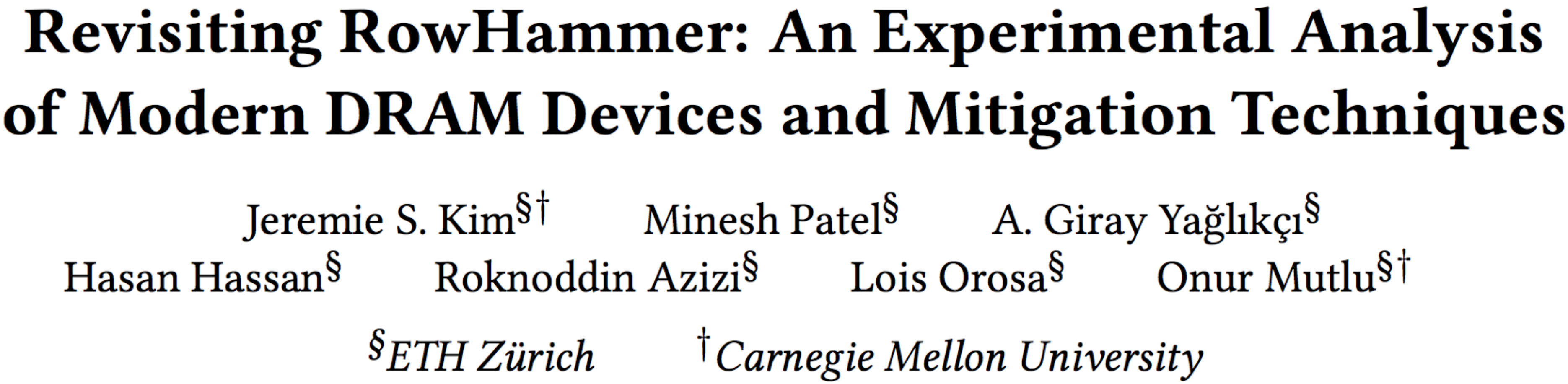
Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors

Yoongu Kim[1] Ross Daly* Jeremie Kim[1] Chris Fallin* Ji Hye Lee[1]
Donghyuk Lee[1] Chris Wilkerson[2] Konrad Lai Onur Mutlu[1]

[1]Carnegie Mellon University [2]Intel Labs

RowHammer is Getting Much Worse (2020)

- 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)]



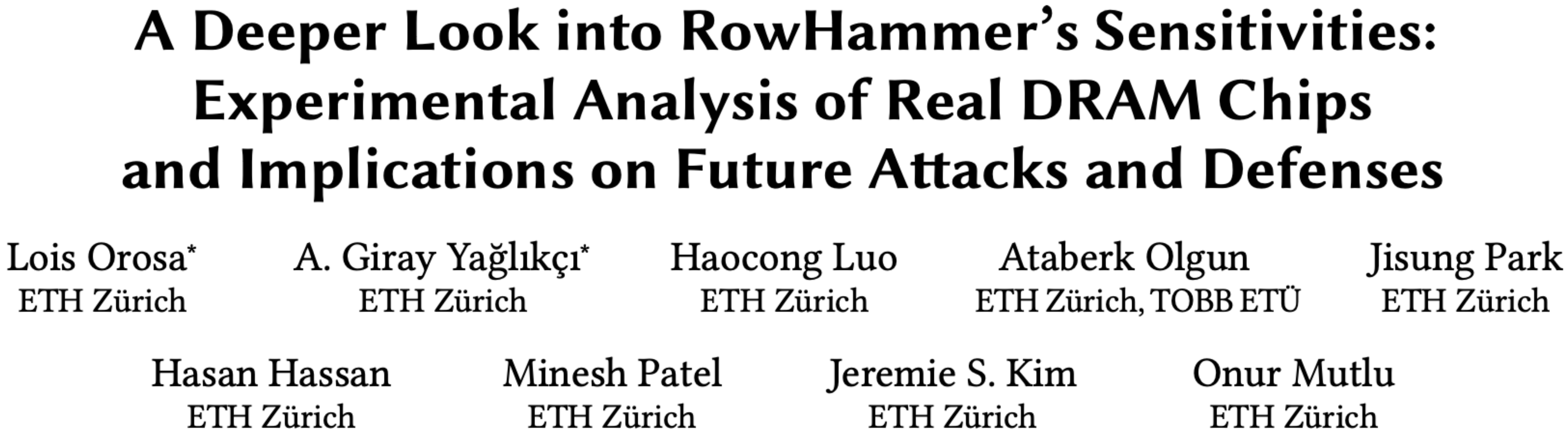
Revisiting RowHammer: An Experimental Analysis of Modern DRAM Devices and Mitigation Techniques

Jeremie S. Kim[§†] Minesh Patel[§] A. Giray Yağlıkçı[§]
Hasan Hassan[§] Roknoddin Azizi[§] Lois Orosa[§] Onur Mutlu[§†]

[§]*ETH Zürich* [†]*Carnegie Mellon University*

New RowHammer Dimensions (2021)

- 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]

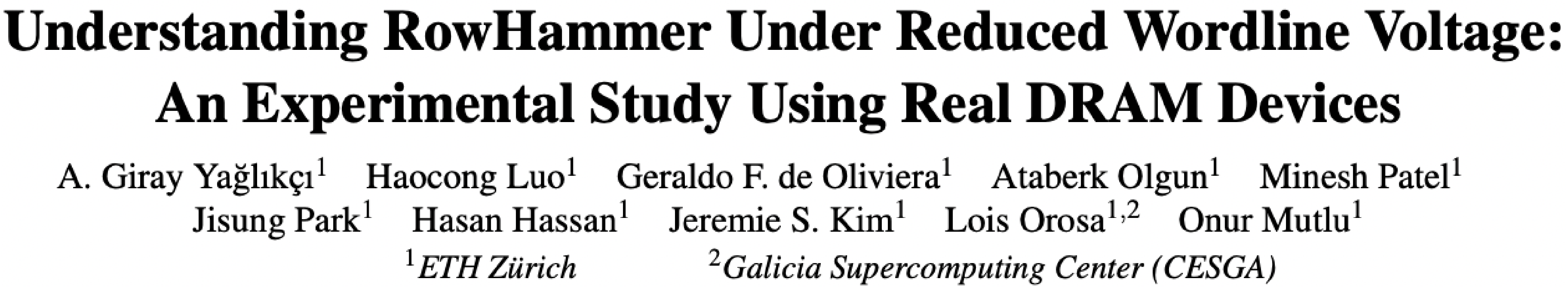


A Deeper Look into RowHammer's Sensitivities: Experimental Analysis of Real DRAM Chips and Implications on Future Attacks and Defenses

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

RowHammer vs. Wordline Voltage (2022)

- A. Giray Yağlıkçı, Haocong Luo, Geraldo F. de Oliviera, Ataberk Olgun, Minesh Patel, Jisung Park, Hasan Hassan, Jeremie S. Kim, Lois Orosa, and Onur Mutlu, **"Understanding RowHammer Under Reduced Wordline Voltage: An Experimental Study Using Real DRAM Devices"**
  *Proceedings of the 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**)*, Baltimore, MD, USA, June 2022.
  [Slides (pptx) (pdf)]
  [Lightning Talk Slides (pptx) (pdf)]
  [arXiv version]
  [Talk Video (34 minutes, including Q&A)]
  [Lightning Talk Video (2 minutes)]



Understanding RowHammer Under Reduced Wordline Voltage: An Experimental Study Using Real DRAM Devices

A. Giray Yağlıkçı[1] Haocong Luo[1] Geraldo F. de Oliviera[1] Ataberk Olgun[1] Minesh Patel[1]
Jisung Park[1] Hasan Hassan[1] Jeremie S. Kim[1] Lois Orosa[1,2] Onur Mutlu[1]
[1]*ETH Zürich* [2]*Galicia Supercomputing Center (CESGA)*

RowHammer Solutions

Two Types of RowHammer Solutions

- Immediate
  - To protect the vulnerable DRAM chips in the field
  - Limited possibilities

- Longer-term
  - To protect future DRAM chips
  - Wider range of protection mechanisms

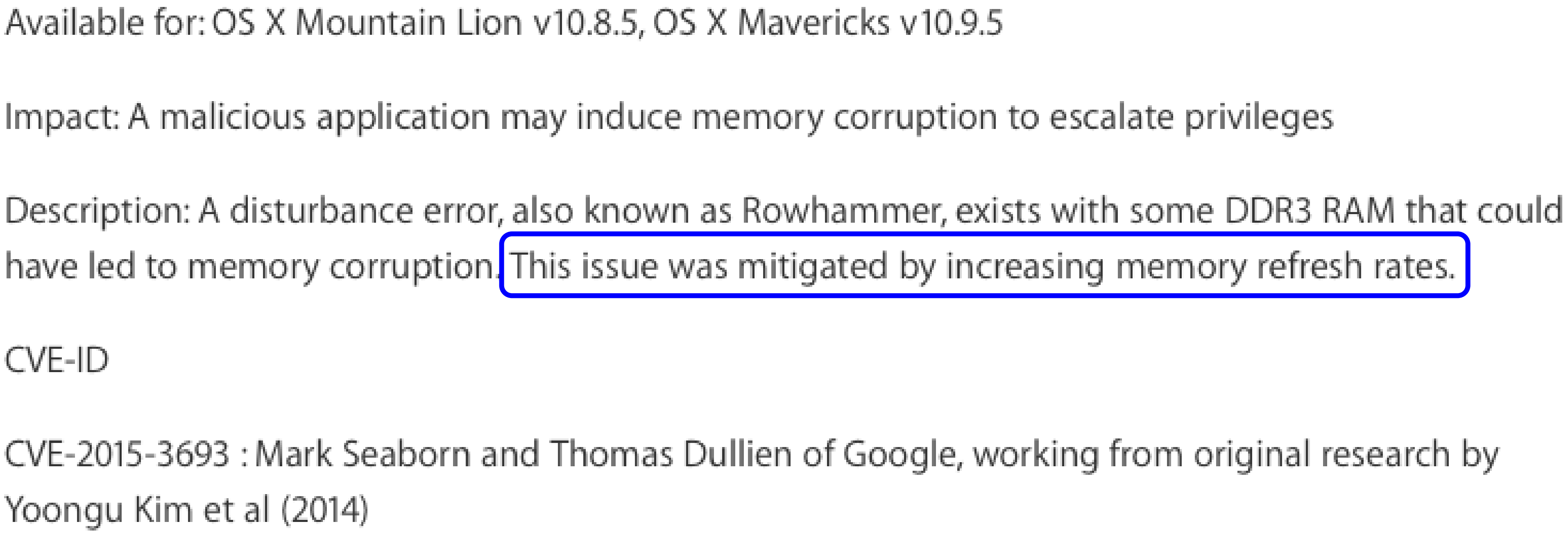
- Our ISCA 2014 paper proposes both types of solutions
  - Seven solutions in total
  - PARA proposed as best solution → already employed in the field

Some Potential Solutions (ISCA 2014)

• Make better DRAM chips	Cost
• Refresh frequently	Power, Performance
• Sophisticated ECC	Cost, Power
• Access counters	Cost, Power, Complexity

Apple's Security Patch for RowHammer

- https://support.apple.com/en-gb/HT204934



Available for: OS X Mountain Lion v10.8.5, OS X Mavericks v10.9.5

Impact: A malicious application may induce memory corruption to escalate privileges

Description: A disturbance error, also known as Rowhammer, exists with some DDR3 RAM that could have led to memory corruption. This issue was mitigated by increasing memory refresh rates.

CVE-ID

CVE-2015-3693 : Mark Seaborn and Thomas Dullien of Google, working from original research by Yoongu Kim et al (2014)

HP, Lenovo, and many other vendors released similar patches

Our Solution to RowHammer

- **PARA:** *Probabilistic Adjacent Row Activation*

- Key Idea
  - After closing a row, we activate (i.e., refresh) one of its neighbors with a low probability: $p = 0.005$

- Reliability Guarantee
  - When $p=0.005$, errors in one year: 9.4×10^{-14}
  - By adjusting the value of p, we can vary the strength of protection against errors

Advantages of PARA

- *PARA refreshes rows infrequently*
  - Low power
  - Low performance-overhead
    - Average slowdown: 0.20% (for 29 benchmarks)
    - Maximum slowdown: 0.75%
- *PARA is stateless*
  - Low cost
  - Low complexity

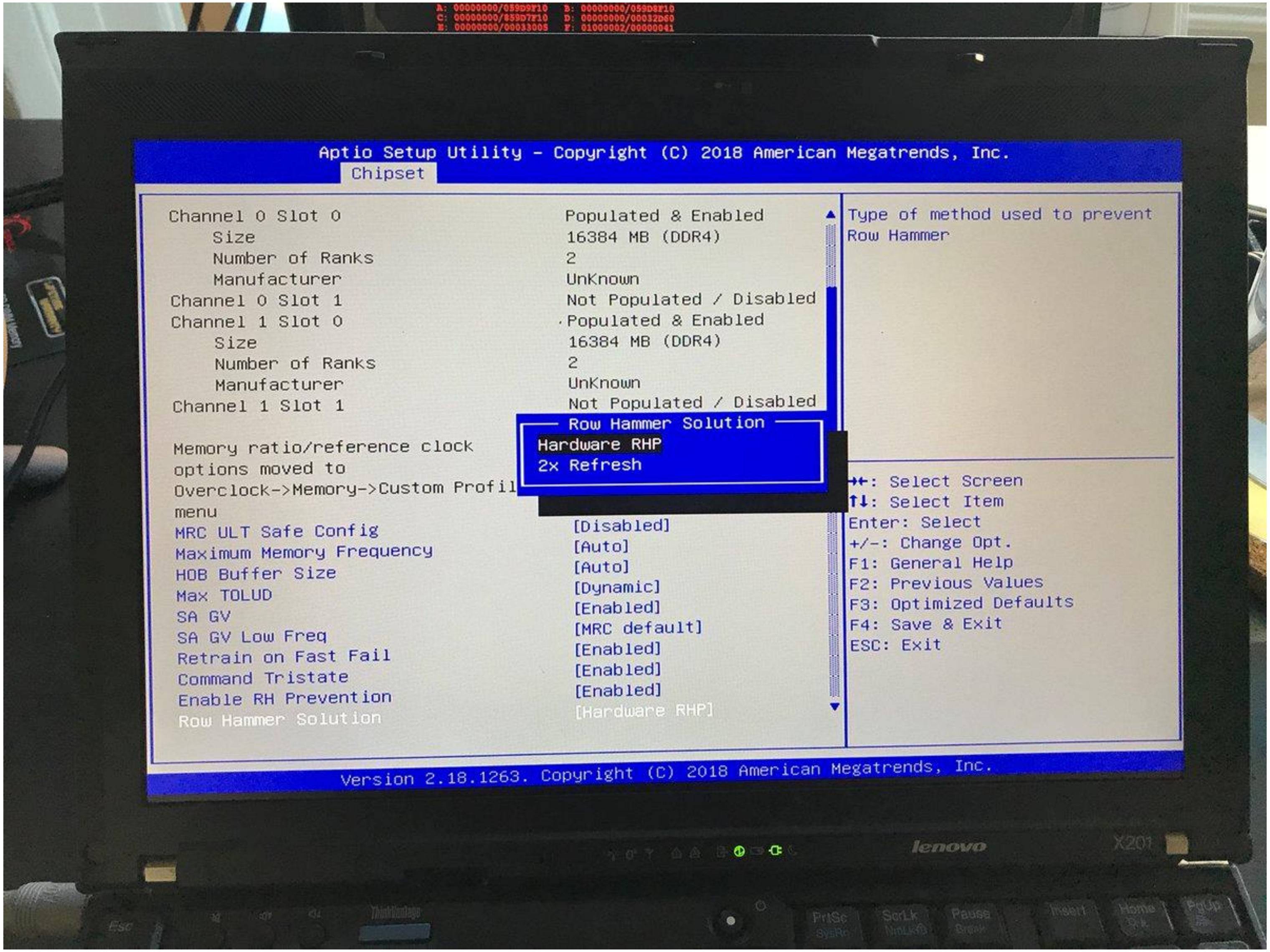
- *PARA is an effective and low-overhead solution to prevent disturbance errors*

Requirements for PARA

- If implemented in DRAM chip (done today)
  - Enough slack in timing and refresh parameters
  - Plenty of slack today:
    - Lee et al., "Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common Case," HPCA 2015.
    - Chang et al., "Understanding Latency Variation in Modern DRAM Chips," SIGMETRICS 2016.
    - Lee et al., "Design-Induced Latency Variation in Modern DRAM Chips," SIGMETRICS 2017.
    - Chang et al., "Understanding Reduced-Voltage Operation in Modern DRAM Devices," SIGMETRICS 2017.
    - Ghose et al., "What Your DRAM Power Models Are Not Telling You: Lessons from a Detailed Experimental Study," SIGMETRICS 2018.
    - Kim et al., "Solar-DRAM: Reducing DRAM Access Latency by Exploiting the Variation in Local Bitlines," ICCD 2018.

- If implemented in memory controller
  - Need coordination between controller and DRAM
  - Memory controller should know which rows are physically adjacent

Probabilistic Activation in Real Life (I)

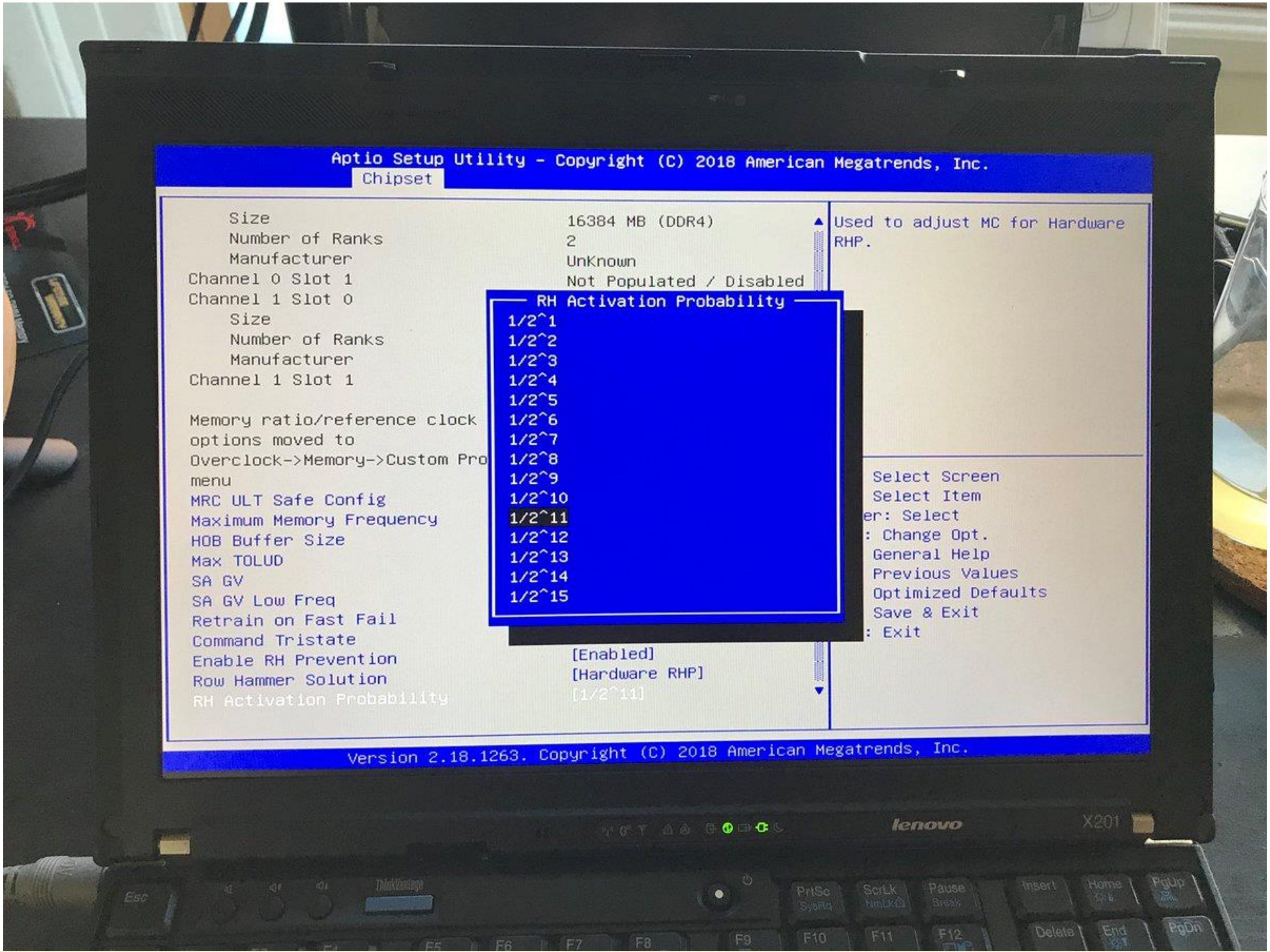


```
A: 00000000/059D9F10   B: 00000000/059D8F10
C: 00000000/859D7F10   D: 00000000/00032D60
E: 00000000/00033005   F: 01000002/00000041
```

```
        Aptio Setup Utility - Copyright (C) 2018 American Megatrends, Inc.
          Chipset

 Channel 0 Slot 0                          Populated & Enabled        Type of method used to prevent
     Size                                  16384 MB (DDR4)            Row Hammer
     Number of Ranks                       2
     Manufacturer                          UnKnown
 Channel 0 Slot 1                          Not Populated / Disabled
 Channel 1 Slot 0                          Populated & Enabled
     Size                                  16384 MB (DDR4)
     Number of Ranks                       2
     Manufacturer                          UnKnown
 Channel 1 Slot 1                          Not Populated / Disabled
                                   ┌── Row Hammer Solution ──┐
 Memory ratio/reference clock      │ Hardware RHP            │
 options moved to                  │ 2x Refresh              │
 Overclock->Memory->Custom Profil  └─────────────────────────┘    →←: Select Screen
 menu                                                             ↑↓: Select Item
 MRC ULT Safe Config                        [Disabled]            Enter: Select
 Maximum Memory Frequency                   [Auto]                +/-: Change Opt.
 HOB Buffer Size                            [Auto]                F1: General Help
 Max TOLUD                                  [Dynamic]             F2: Previous Values
 SA GV                                      [Enabled]             F3: Optimized Defaults
 SA GV Low Freq                             [MRC default]         F4: Save & Exit
 Retrain on Fast Fail                       [Enabled]             ESC: Exit
 Command Tristate                           [Enabled]
 Enable RH Prevention                       [Enabled]
 Row Hammer Solution                        [Hardware RHP]

        Version 2.18.1263. Copyright (C) 2018 American Megatrends, Inc.
```

https://twitter.com/isislovecruft/status/1021939922754723841

Probabilistic Activation in Real Life (II)

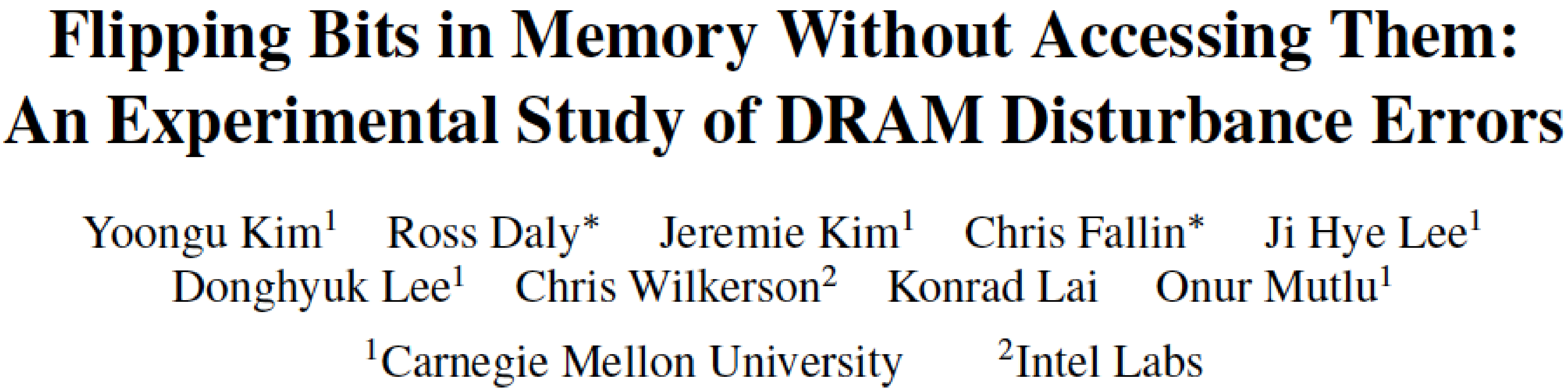


```
        Aptio Setup Utility - Copyright (C) 2018 American Megatrends, Inc.
                 Chipset

    Size                                 16384 MB (DDR4)          Used to adjust MC for Hardware
    Number of Ranks                      2                        RHP.
    Manufacturer                         UnKnown
Channel 0 Slot 1                         Not Populated / Disabled
Channel 1 Slot 0               ┌ RH Activation Probability ┐
    Size                       │ 1/2^1                      │
    Number of Ranks            │ 1/2^2                      │
    Manufacturer               │ 1/2^3                      │
Channel 1 Slot 1               │ 1/2^4                      │
                               │ 1/2^5                      │
Memory ratio/reference clock   │ 1/2^6                      │
options moved to               │ 1/2^7                      │
Overclock->Memory->Custom Pro  │ 1/2^8                      │
menu                           │ 1/2^9                      │   Select Screen
MRC ULT Safe Config            │ 1/2^10                     │   Select Item
Maximum Memory Frequency       │ 1/2^11                     │  er: Select
HOB Buffer Size                │ 1/2^12                     │  : Change Opt.
Max TOLUD                      │ 1/2^13                     │   General Help
SA GV                          │ 1/2^14                     │   Previous Values
SA GV Low Freq                 │ 1/2^15                     │   Optimized Defaults
Retrain on Fast Fail           └────────────────────────────┘   Save & Exit
Command Tristate                                              : Exit
Enable RH Prevention                     [Enabled]
Row Hammer Solution                      [Hardware RHP]
RH Activation Probability                [1/2^11]

        Version 2.18.1263. Copyright (C) 2018 American Megatrends, Inc.
```

https://twitter.com/isislovecruft/status/1021939922754723841

Seven RowHammer Solutions Proposed

- 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 Kim[1] Ross Daly* Jeremie Kim[1] Chris Fallin* Ji Hye Lee[1]
Donghyuk Lee[1] Chris Wilkerson[2] Konrad Lai Onur Mutlu[1]

[1]Carnegie Mellon University [2]Intel Labs

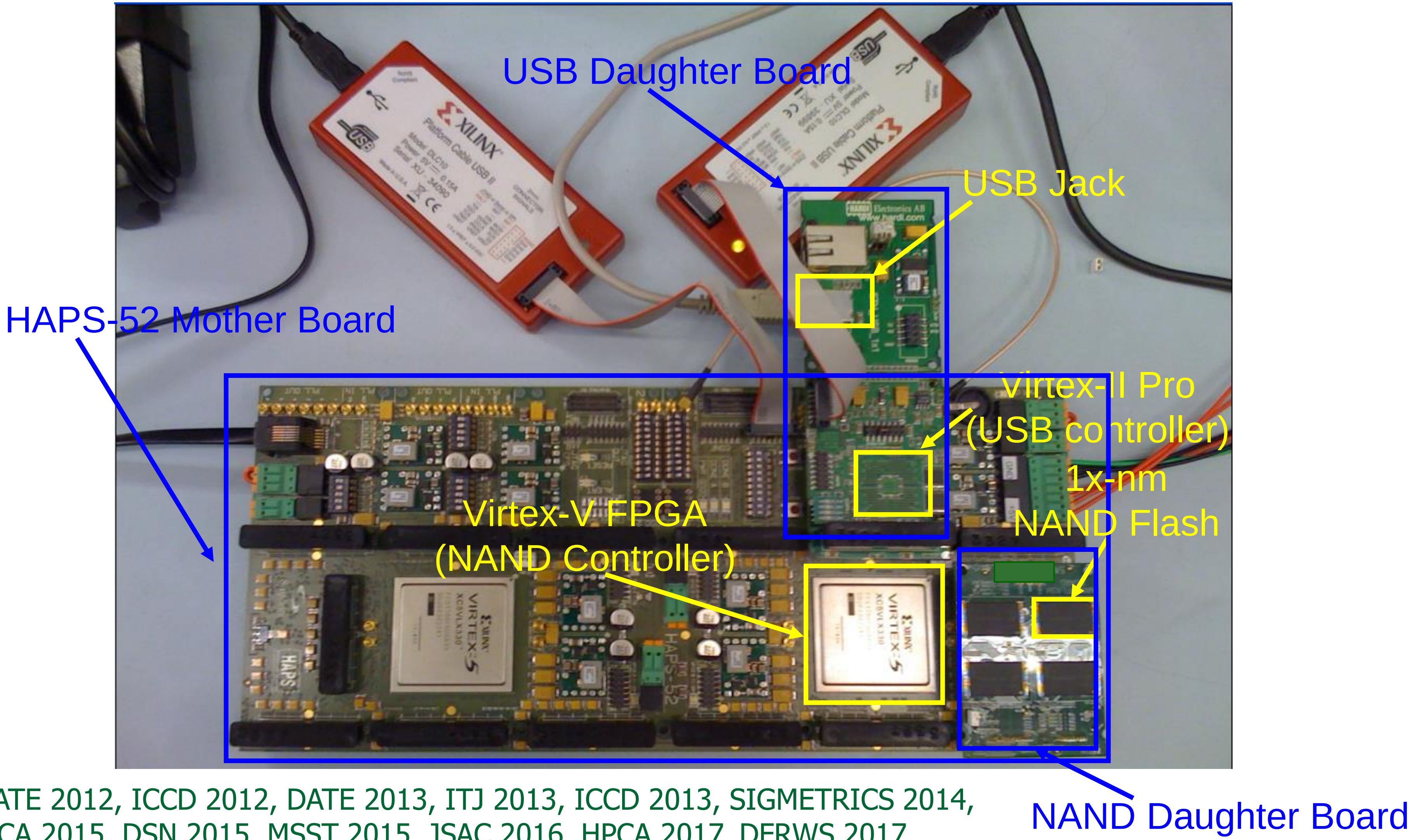
A Takeaway

Main Memory Needs

Intelligent Controllers

for Security, Safety, Reliability, Scaling

Aside: Intelligent Controller for NAND Flash

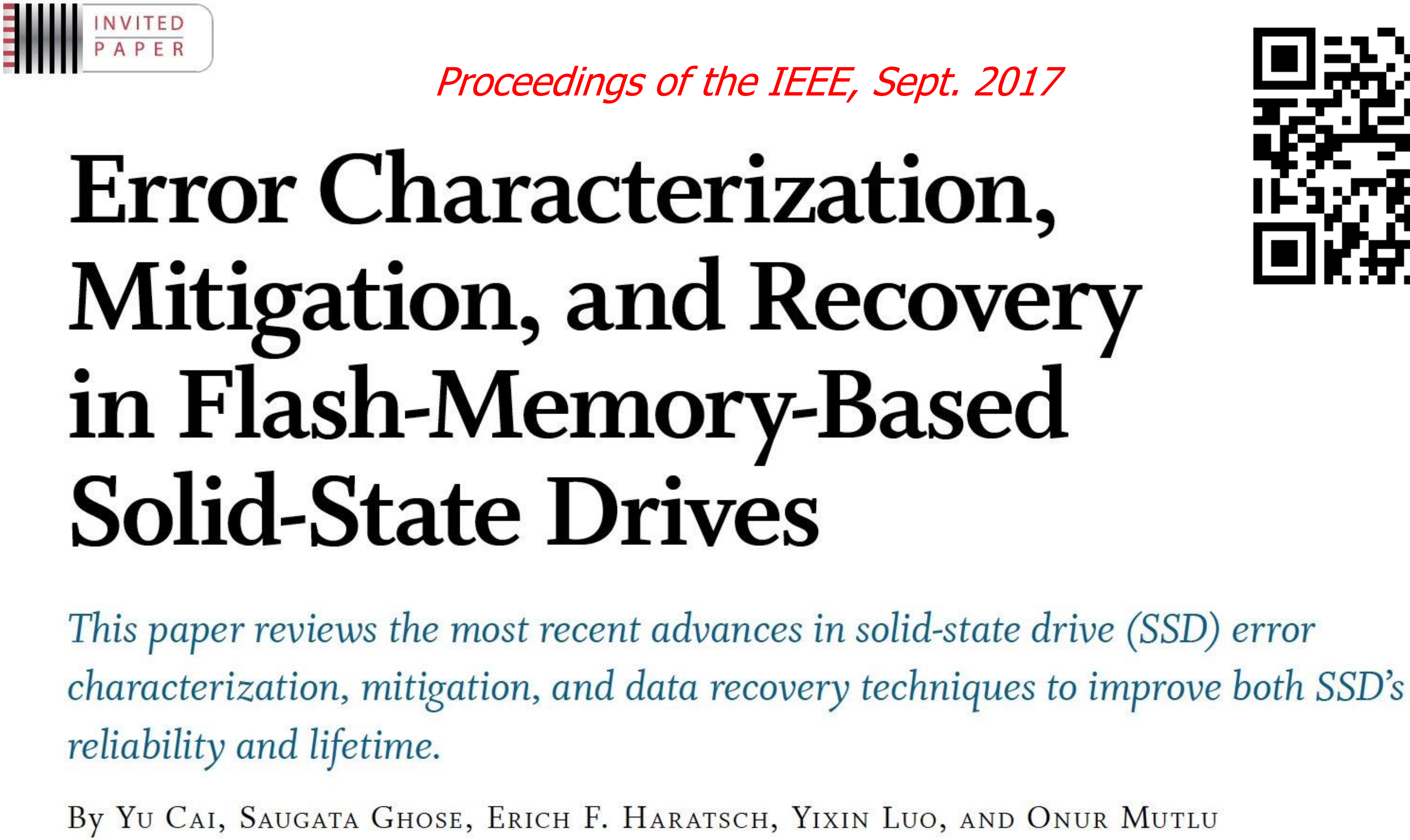


[DATE 2012, ICCD 2012, DATE 2013, ITJ 2013, ICCD 2013, SIGMETRICS 2014, HPCA 2015, DSN 2015, MSST 2015, JSAC 2016, HPCA 2017, DFRWS 2017, PIEEE 2017, HPCA 2018, SIGMETRICS 2018]

Cai+, "Error Characterization, Mitigation, and Recovery in Flash Memory Based Solid State Drives," Proc. IEEE 2017.

Intelligent Flash Controllers **[PIEEE'17]**

Proceedings of the IEEE, Sept. 2017



INVITED PAPER

Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives

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

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

https://arxiv.org/pdf/1706.08642

Detailed Lectures on RowHammer

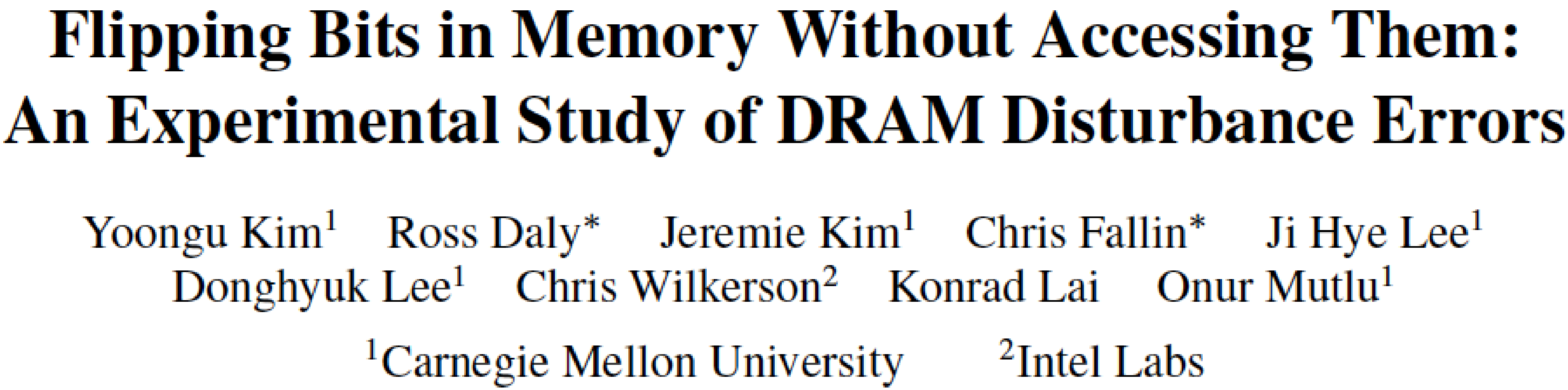
- Computer Architecture, Fall 2021, Lecture 5
  - RowHammer (ETH Zürich, Fall 2021)
  - https://www.youtube.com/watch?v=7wVKnPj3NVw&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAGILkTOF&index=5

- 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

First RowHammer Analysis

- 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).***



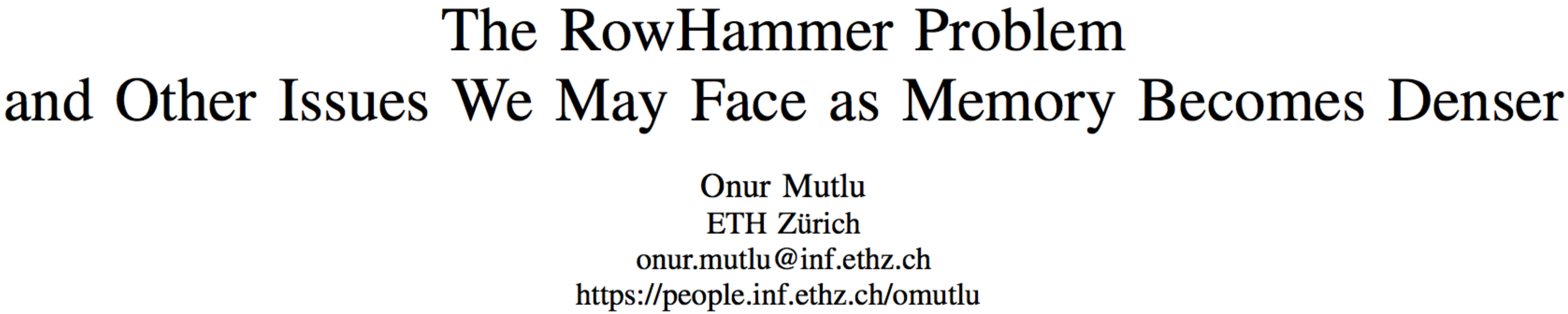
Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors

Yoongu Kim[1] Ross Daly* Jeremie Kim[1] Chris Fallin* Ji Hye Lee[1]
Donghyuk Lee[1] Chris Wilkerson[2] Konrad Lai Onur Mutlu[1]

[1]Carnegie Mellon University [2]Intel Labs

Retrospective on RowHammer & Future

- Onur Mutlu,
  "The RowHammer Problem and Other Issues We May Face as Memory Becomes Denser"
  Invited Paper in Proceedings of the Design, Automation, and Test in Europe Conference (***DATE***), Lausanne, Switzerland, March 2017.
  [Slides (pptx) (pdf)]



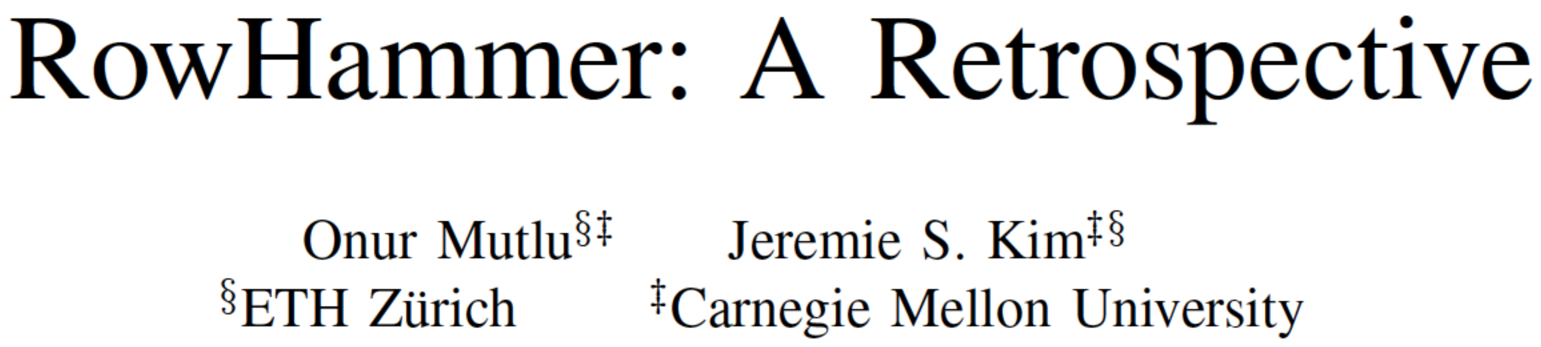
The RowHammer Problem
and Other Issues We May Face as Memory Becomes Denser

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

https://people.inf.ethz.ch/omutlu/pub/rowhammer-and-other-memory-issues_date17.pdf

A More Recent RowHammer Retrospective

- 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

A Key Takeaway

Main Memory Needs

Intelligent Controllers

RowHammer in 2020-2022

Revisiting RowHammer

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*

Key Takeaways from 1580 Chips

- **Newer DRAM chips are much more vulnerable to RowHammer (more bit flips, happening earlier)**

- There are new chips whose weakest cells fail after **only 4800 hammers**

- Chips of newer DRAM technology nodes can exhibit RowHammer bit flips 1) in **more rows** and 2) **farther away** from the victim row.

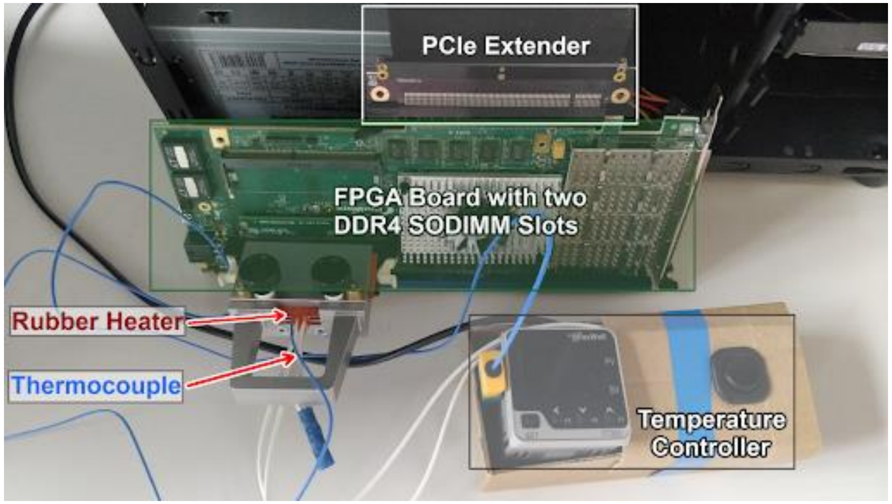
- **Existing mitigation mechanisms are NOT effective at future technology nodes**

DRAM Testing Infrastructures

Three separate testing infrastructures

1. **DDR3:** FPGA-based SoftMC [Hassan+, HPCA'17] (Xilinx ML605)
2. **DDR4:** FPGA-based SoftMC [Hassan+, HPCA'17] (Xilinx Virtex UltraScale 95)
3. **LPDDR4:** In-house testing hardware for LPDDR4 chips

All provide fine-grained control over DRAM commands, timing parameters and temperature



DDR4 DRAM testing infrastructure

1580 DRAM Chips Tested

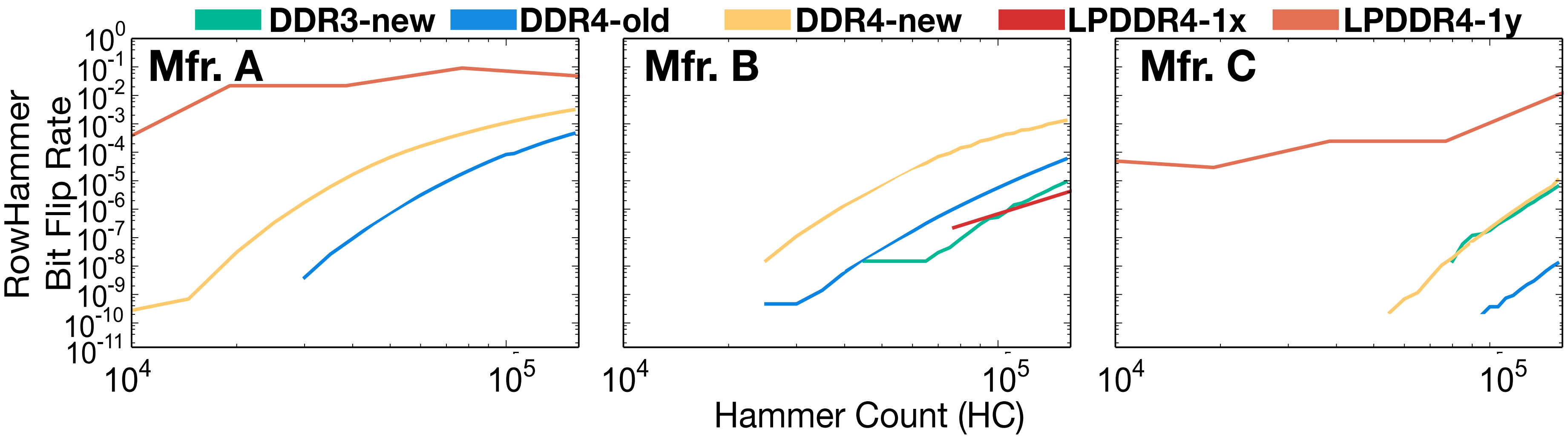
DRAM type-node	Number of Chips (Modules) Tested			
	Mfr. A	*Mfr. B*	*Mfr. C*	*Total*
DDR3-old	56 (10)	88 (11)	28 (7)	**172 (28)**
DDR3-new	80 (10)	52 (9)	104 (13)	**236 (32)**
DDR4-old	112 (16)	24 (3)	128 (18)	**264 (37)**
DDR4-new	264 (43)	16 (2)	108 (28)	**388 (73)**
LPDDR4-1x	12 (3)	180 (45)	N/A	**192 (48)**
LPDDR4-1y	184 (46)	N/A	144 (36)	**328 (82)**

1580 total DRAM chips tested from **300** DRAM modules

- **Three** major DRAM manufacturers {A, B, C}
- **Three** DRAM *types* or *standards* {DDR3, DDR4, LPDDR4}
  - LPDDR4 chips we test implement on-die ECC
- **Two** technology nodes per DRAM type {old/new, 1x/1y}
  - Categorized based on manufacturing date, datasheet publication date, purchase date, and characterization results

Type-node: configuration describing a chip's type and technology node generation: **DDR3-old/new, DDR4-old/new, LPDDR4-1x/1y**

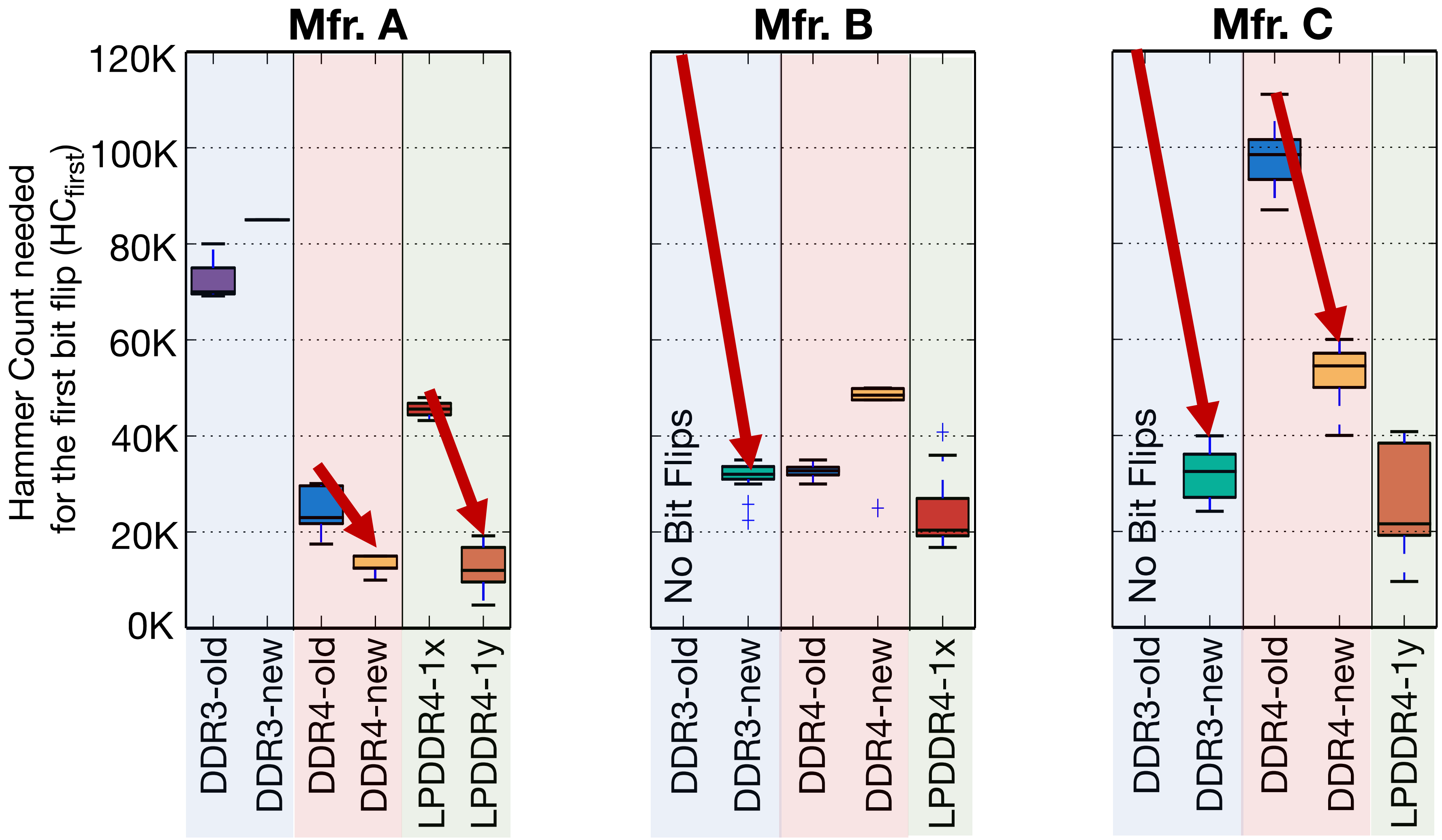
3. Hammer Count (HC) Effects



RowHammer bit flip rates **increase**
when going **from old to new** DDR4 technology node generations

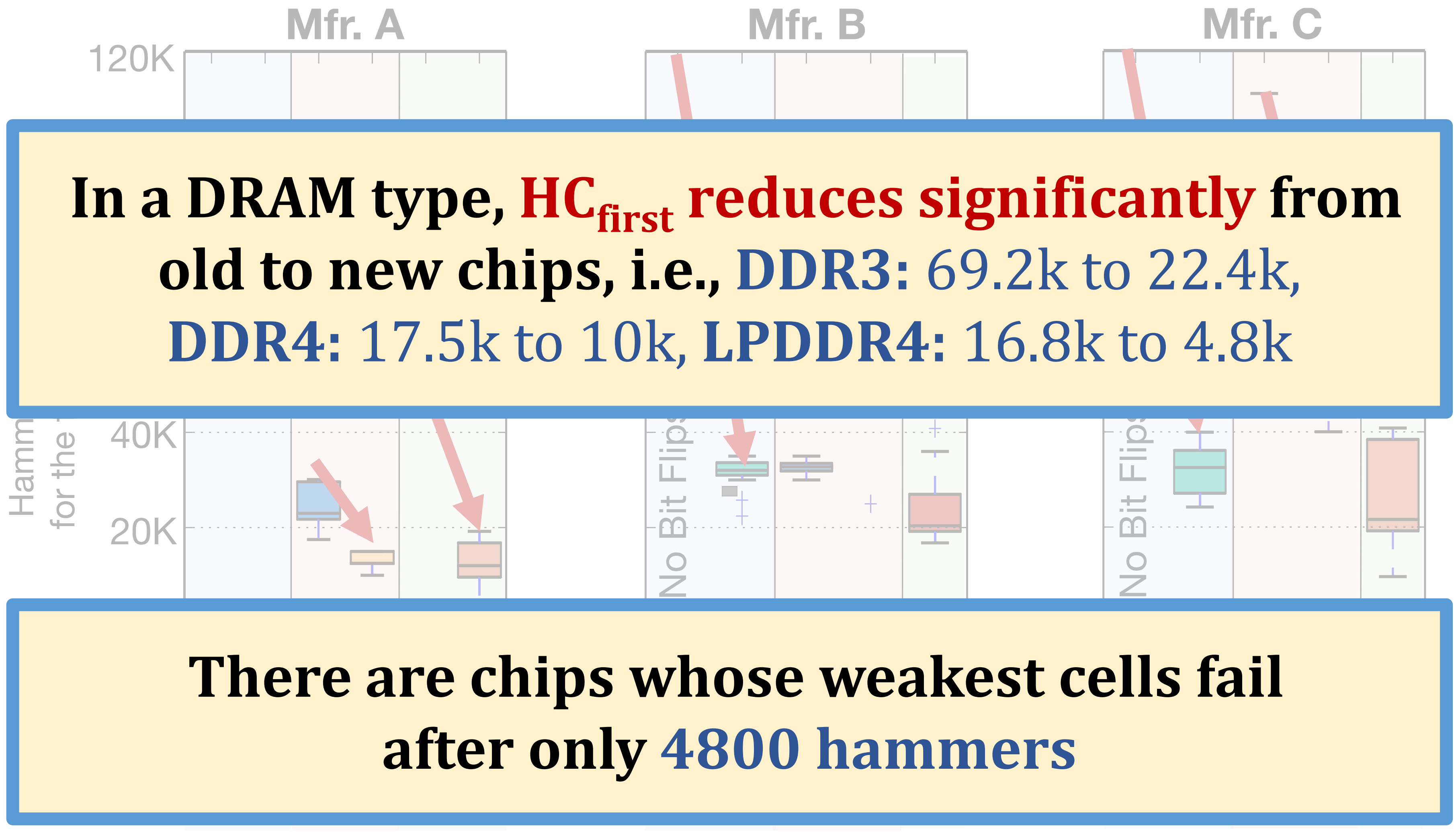
RowHammer bit flip rates (i.e., RowHammer vulnerability) increase with technology node generation

5. First RowHammer Bit Flips per Chip



Newer chips from each DRAM manufacturer are more vulnerable to RowHammer

5. First RowHammer Bit Flips per Chip



In a DRAM type, HC_{first} reduces significantly from old to new chips, i.e., DDR3: 69.2k to 22.4k, DDR4: 17.5k to 10k, LPDDR4: 16.8k to 4.8k

There are chips whose weakest cells fail after only 4800 hammers

Newer chips from a given DRAM manufacturer
more vulnerable to RowHammer

SAFARI

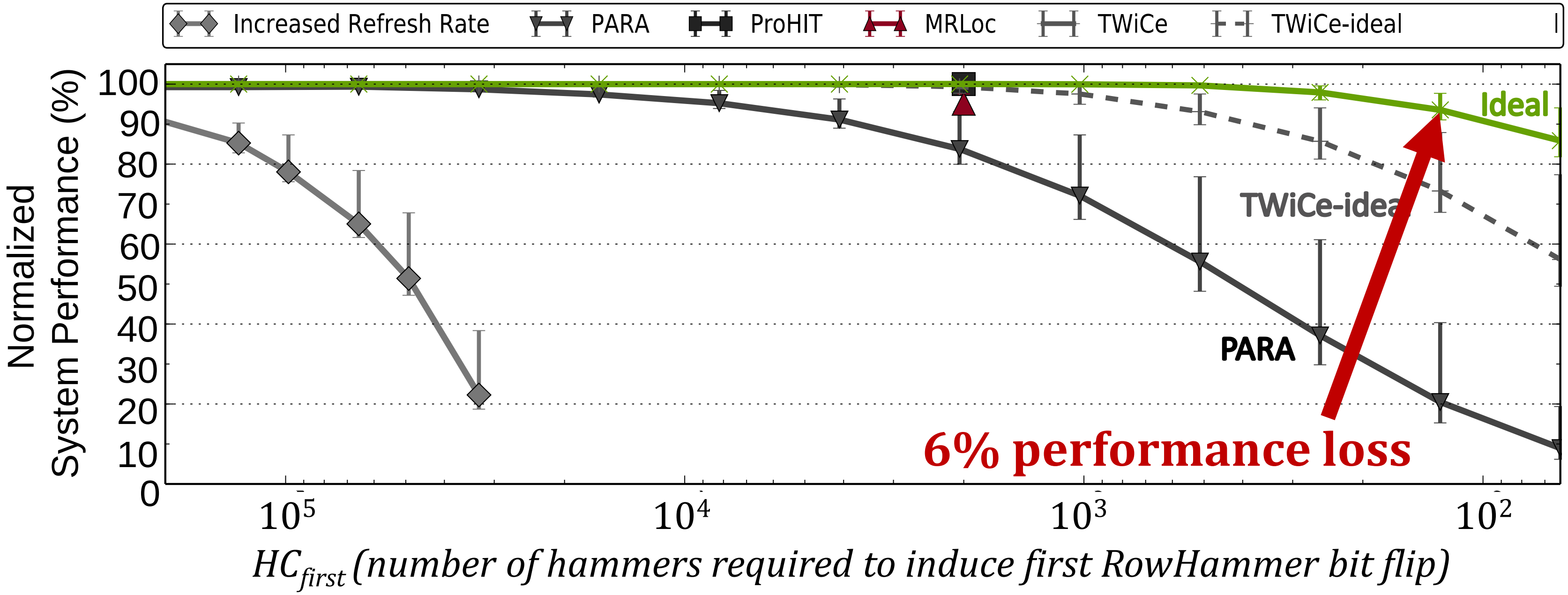
Evaluation of Solutions

- We evaluate **five** state-of-the-art mitigation mechanisms:
  - **Increased Refresh Rate** **[Kim+, ISCA'14]**
  - **PARA** **[Kim+, ISCA'14]**
  - **ProHIT** **[Son+, DAC'17]**
  - **MRLoc** **[You+, DAC'19]**
  - **TWiCe** **[Lee+, ISCA'19]**

- and **one** ideal refresh-based mitigation mechanism:
  - **Ideal**

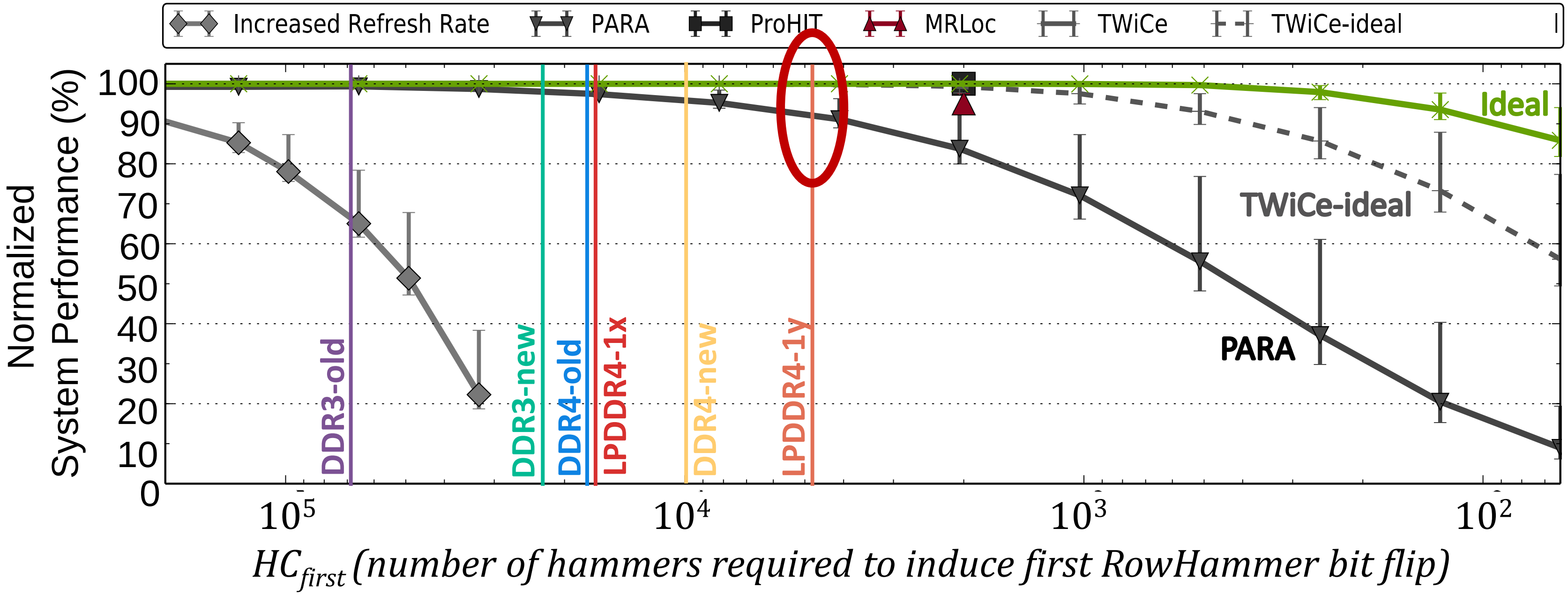
- **More detailed descriptions in the paper on:**
  - Descriptions of mechanisms in our paper and the original publications
  - How we scale each mechanism to more vulnerable DRAM chips (lower $\mathbf{HC_{first}}$)

Mitigation Mechanism Evaluation (Ideal)



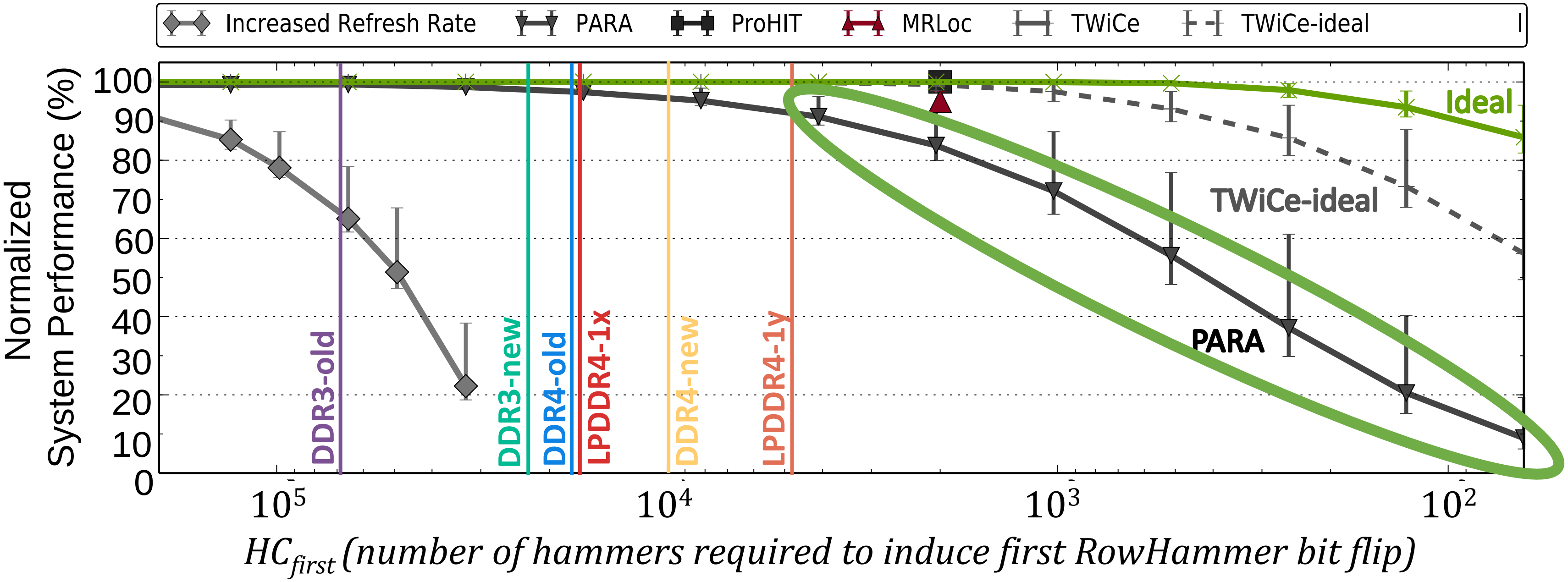
Ideal mechanism issues a refresh command to a row only right before the row can potentially experience a RowHammer bit flip

Mitigation Mechanism Evaluation



PARA, ProHIT, and MRLoc mitigate RowHammer bit flips in worst chips today with reasonable system performance (92%, 100%, 100%)

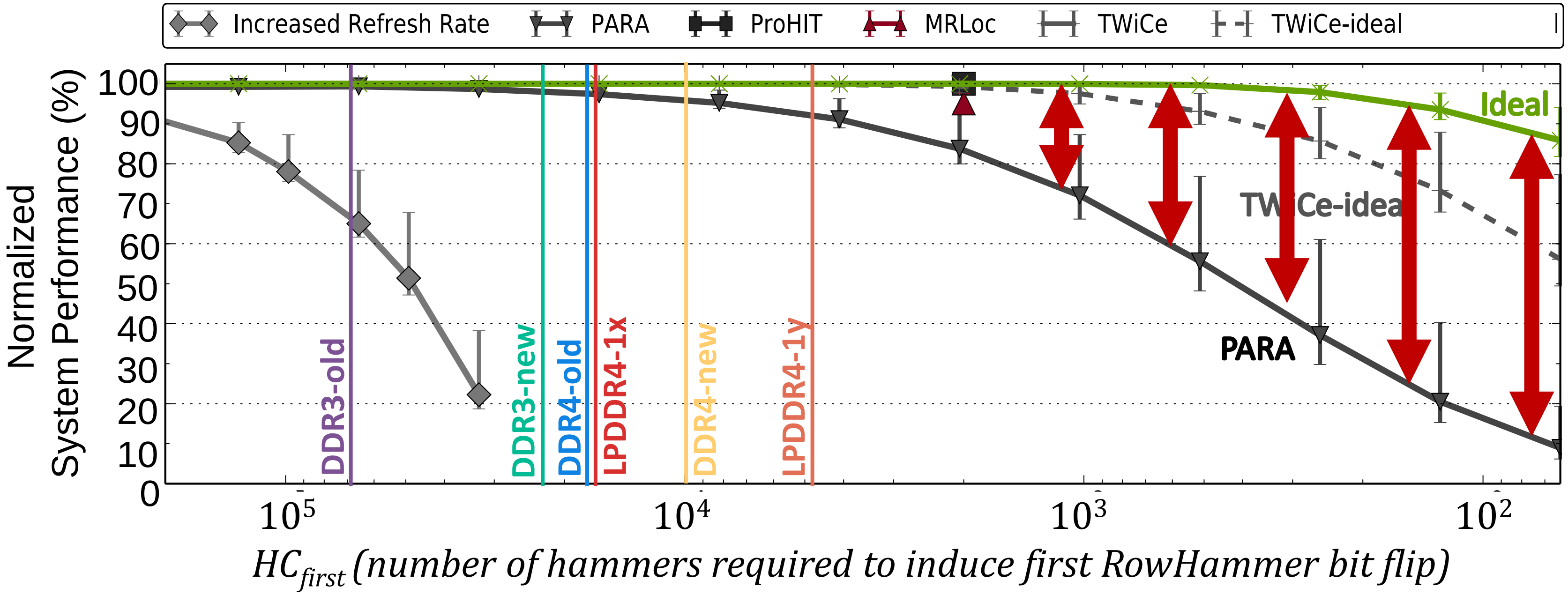
Mitigation Mechanism Evaluation



Only PARA's design scales to low HC_{first} values
but has very low normalized system performance

SAFARI

Mitigation Mechanism Evaluation



Ideal mechanism is significantly better than any existing mechanism for HC_{first} < 1024

Significant opportunity for developing a RowHammer solution with low performance overhead that supports low HC_{first}

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*

Detailed Lecture on Revisiting RowHammer

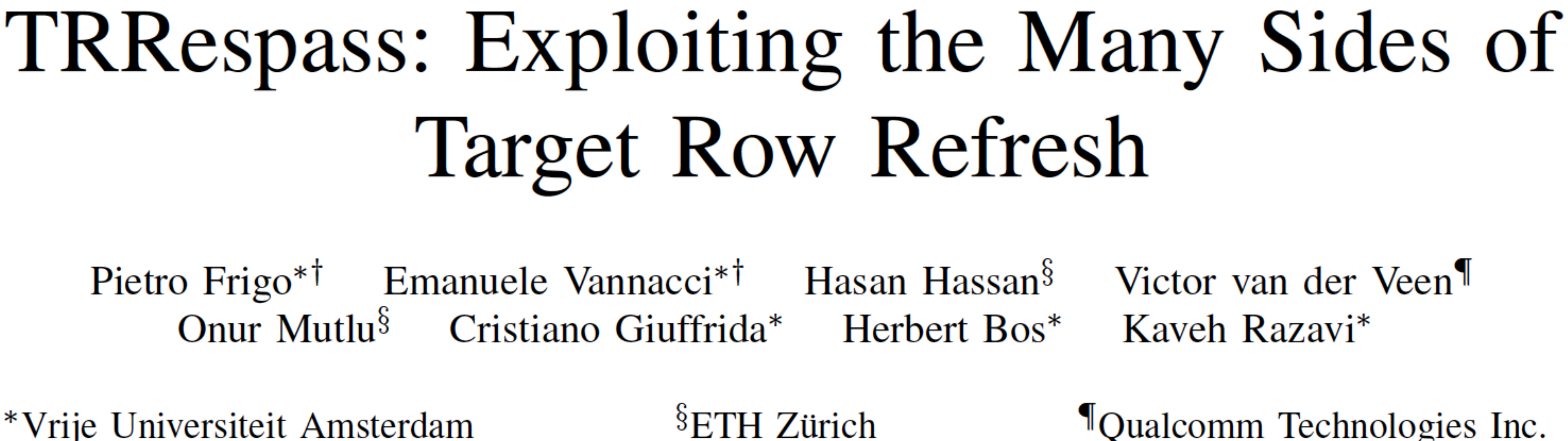
- Computer Architecture, Fall 2020, Lecture 5b
  - RowHammer in 2020: Revisiting RowHammer (ETH Zürich, Fall 2020)
  - https://www.youtube.com/watch?v=gR7XR-Eepcg&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=10

https://www.youtube.com/onurmutlulectures

TRRespass

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*

*Vrije Universiteit Amsterdam §ETH Zürich ¶Qualcomm Technologies Inc.

TRRespass

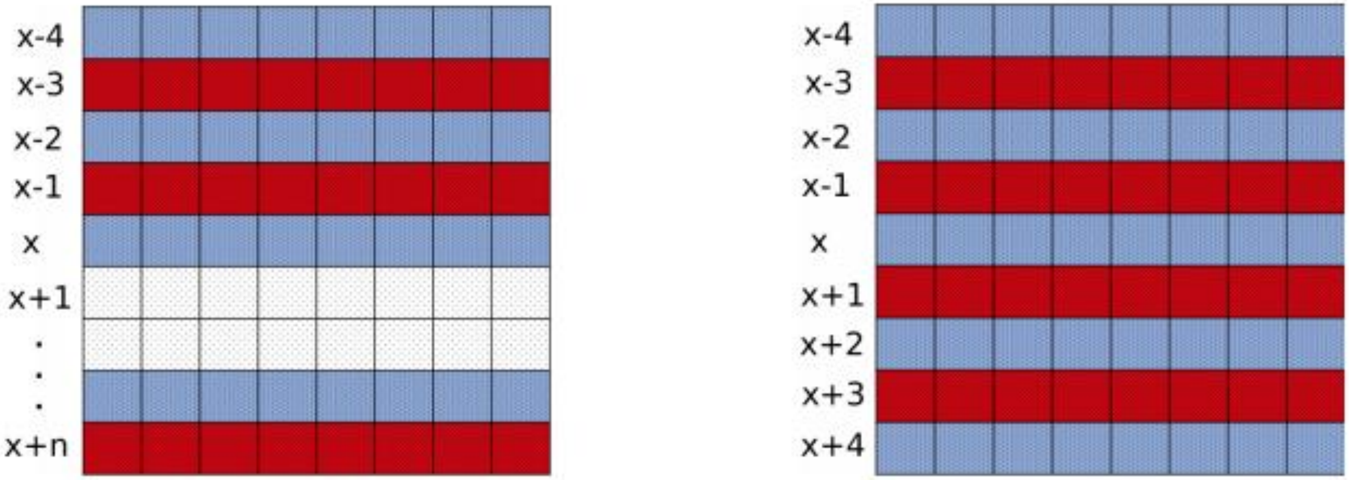
- First work to show that TRR-protected DRAM chips are vulnerable to RowHammer in the field
  - Mitigations advertised as secure are not secure

- Introduces the Many-sided RowHammer attack
  - Idea: Hammer many rows to bypass TRR mitigations (e.g., by overflowing proprietary TRR tables that detect aggressor rows)

- (Partially) reverse-engineers the TRR and pTRR mitigation mechanisms implemented in DRAM chips and memory controllers

- Provides an automatic tool that can effectively create many-sided RowHammer attacks in DDR4 and LPDDR4(X) chips

Example Many-Sided Hammering Patterns



(a) Assisted double-sided **(b)** 4-sided

Fig. 12: Hammering patterns discovered by *TRRespass*. Aggressor rows are in red (🟥) and victim rows are in blue (🟦).

BitFlips vs. Number of Aggressor Rows

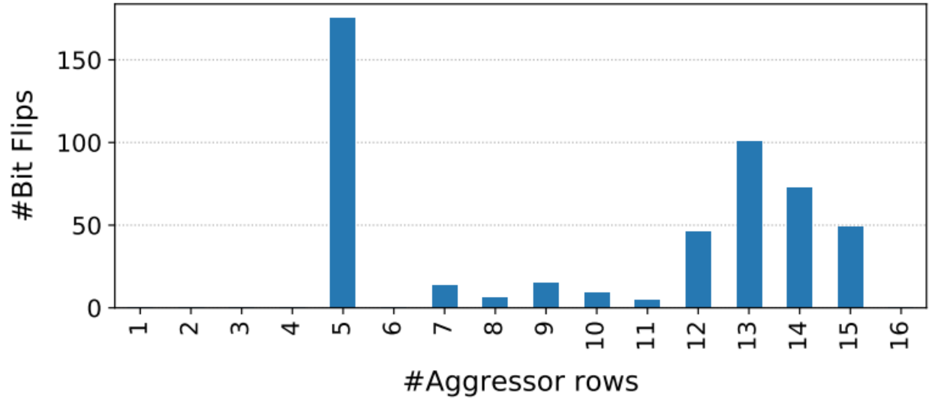


Fig. 10: Bit flips vs. number of aggressor rows. Module $\mathcal{C}_{12}$: Number of bit flips in bank 0 as we vary the number of aggressor rows. Using SoftMC, we refresh DRAM with standard `tREFI` and run the tests until each aggressor rows is hammered 500K times.

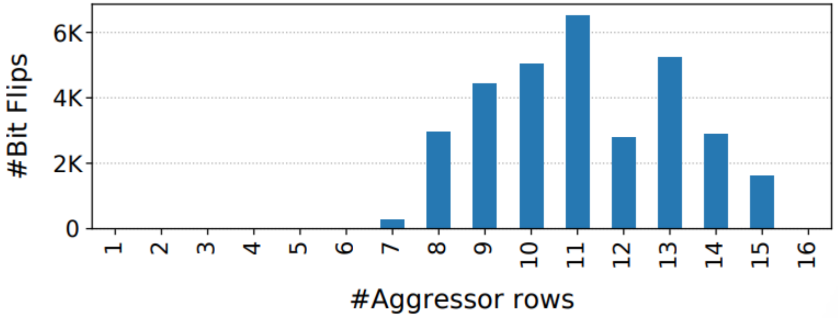


Fig. 11: Bit flips vs. number of aggressor rows. Module $\mathcal{A}_{15}$: Number of bit flips in bank 0 as we vary the number of aggressor rows. Using SoftMC, we refresh DRAM with standard `tREFI` and run the tests until each aggressor rows is hammered 500K times.

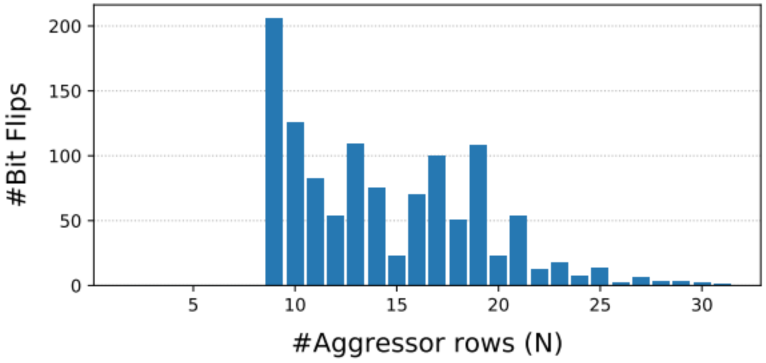
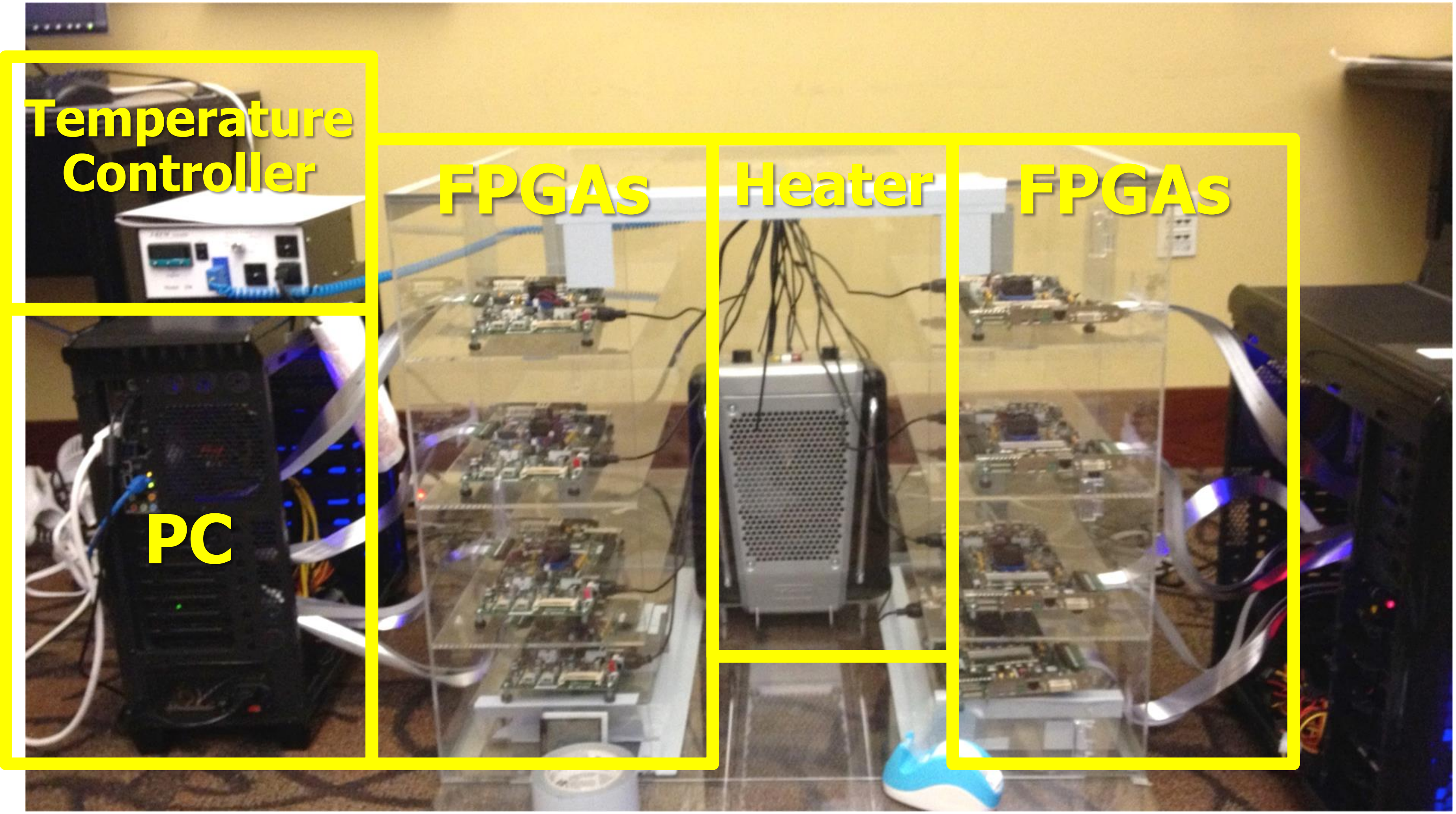


Fig. 13: Bit flips vs. number of aggressor rows. Module $\mathcal{A}_{10}$: Number of bit flips triggered with *N-sided* RowHammer for varying number of *N* on Intel Core i7-7700K. Each aggressor row is one row away from the closest aggressor row (i.e., `VAVAVA...` configuration) and aggressor rows are hammered in a round-robin fashion.

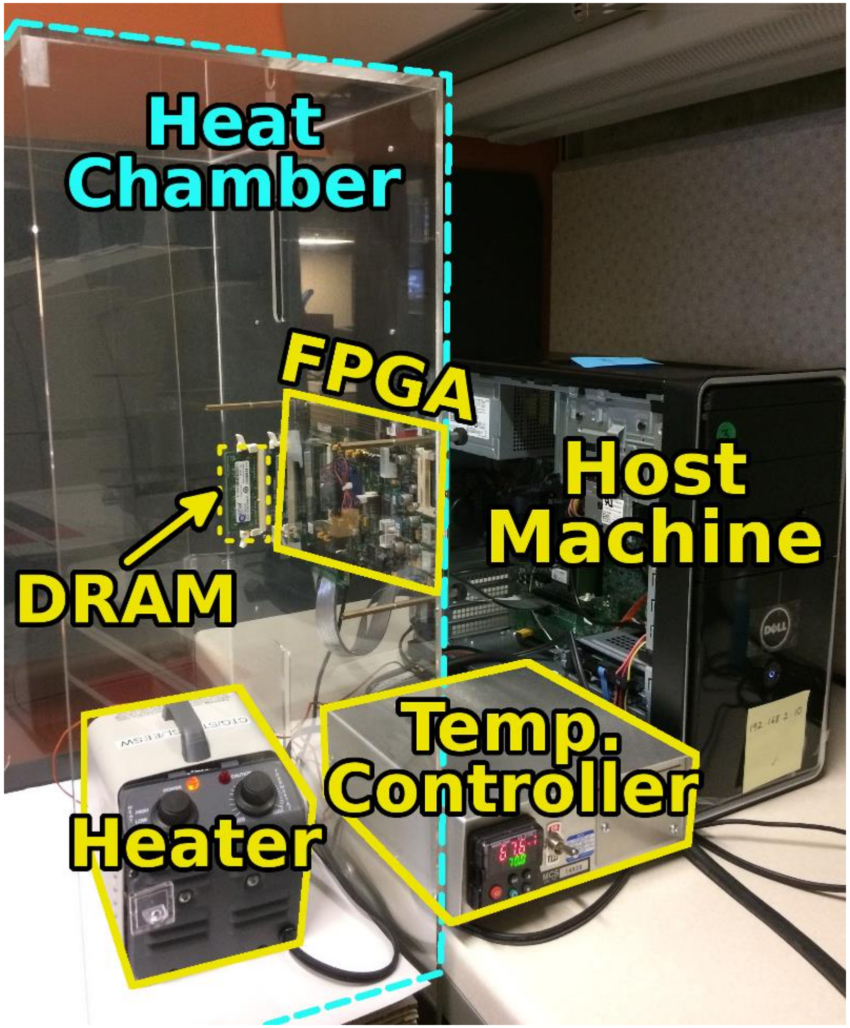
Infrastructures to Understand Such Issues



Kim+, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," ISCA 2014.

SoftMC: Open Source DRAM Infrastructure

- Hasan Hassan et al., "**SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies**," HPCA 2017.

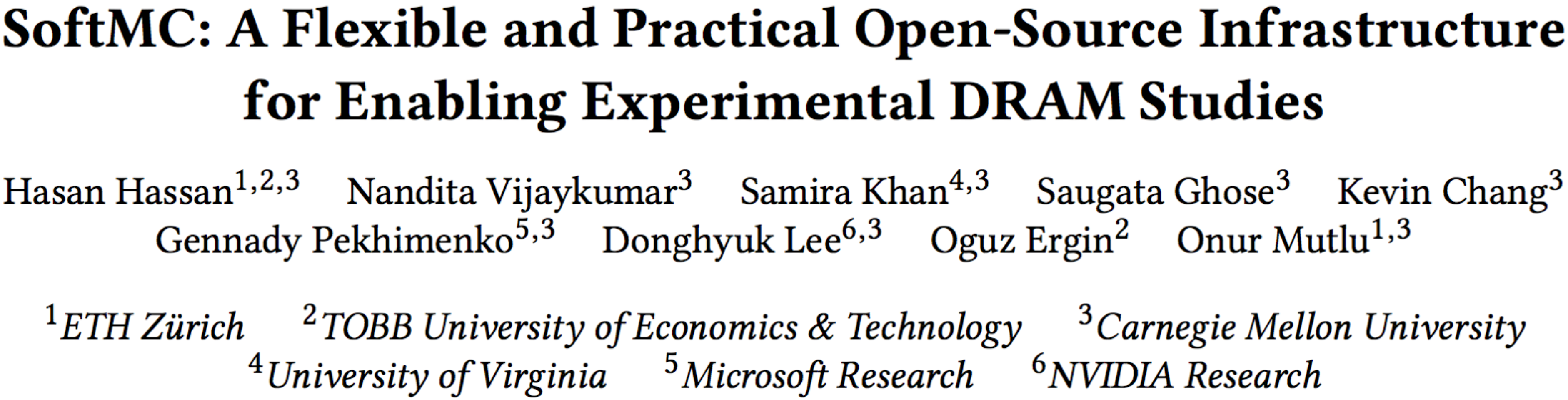


- **Flexible**
- **Easy to Use (C++ API)**
- **Open-source**

github.com/CMU-SAFARI/SoftMC

SoftMC

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



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

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

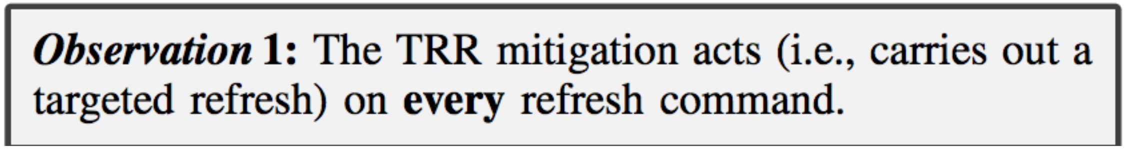
[1]*ETH Zürich* [2]*TOBB University of Economics & Technology* [3]*Carnegie Mellon University*
[4]*University of Virginia* [5]*Microsoft Research* [6]*NVIDIA Research*

Components of In-DRAM TRR

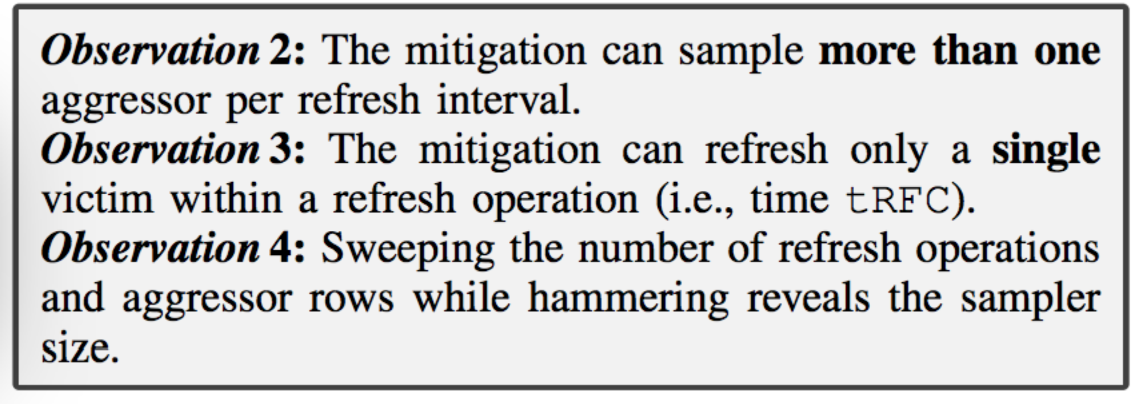
- **Sampler**
  - Tracks aggressor rows activations
  - Design options:
    - Frequency based (record every N^th^ row activation)
    - Time based (record first N row activations)
    - Random seed (record based on a coin flip)
  - Regardless, the sampler has a limited size

- **Inhibitor**
  - Prevents bit flips by refreshing victim rows
    - The latency of performing victim row refreshes is squeezed into slack time available in *tRFC* (i.e., the latency of regular Refresh command)

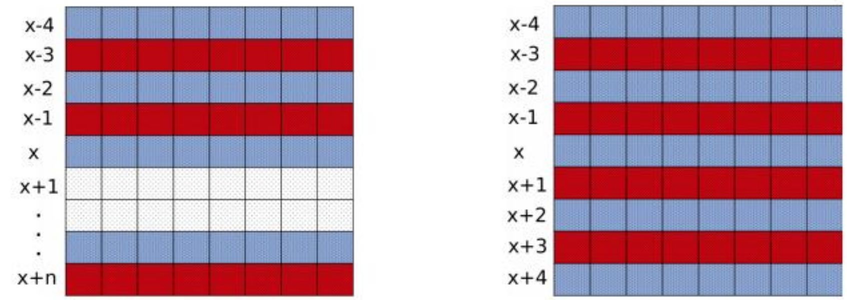
Some Observations



***Observation* 1:** The TRR mitigation acts (i.e., carries out a targeted refresh) on **every** refresh command.



***Observation* 2:** The mitigation can sample **more than one** aggressor per refresh interval.
***Observation* 3:** The mitigation can refresh only a **single** victim within a refresh operation (i.e., time `tRFC`).
***Observation* 4:** Sweeping the number of refresh operations and aggressor rows while hammering reveals the sampler size.



(a) Assisted double-sided **(b)** 4-sided

Fig. 12: Hammering patterns discovered by *TRRespass*. Aggressor rows are in red (🟥) and victim rows are in blue (🟦).

TRRespass Vulnerable DRAM Modules

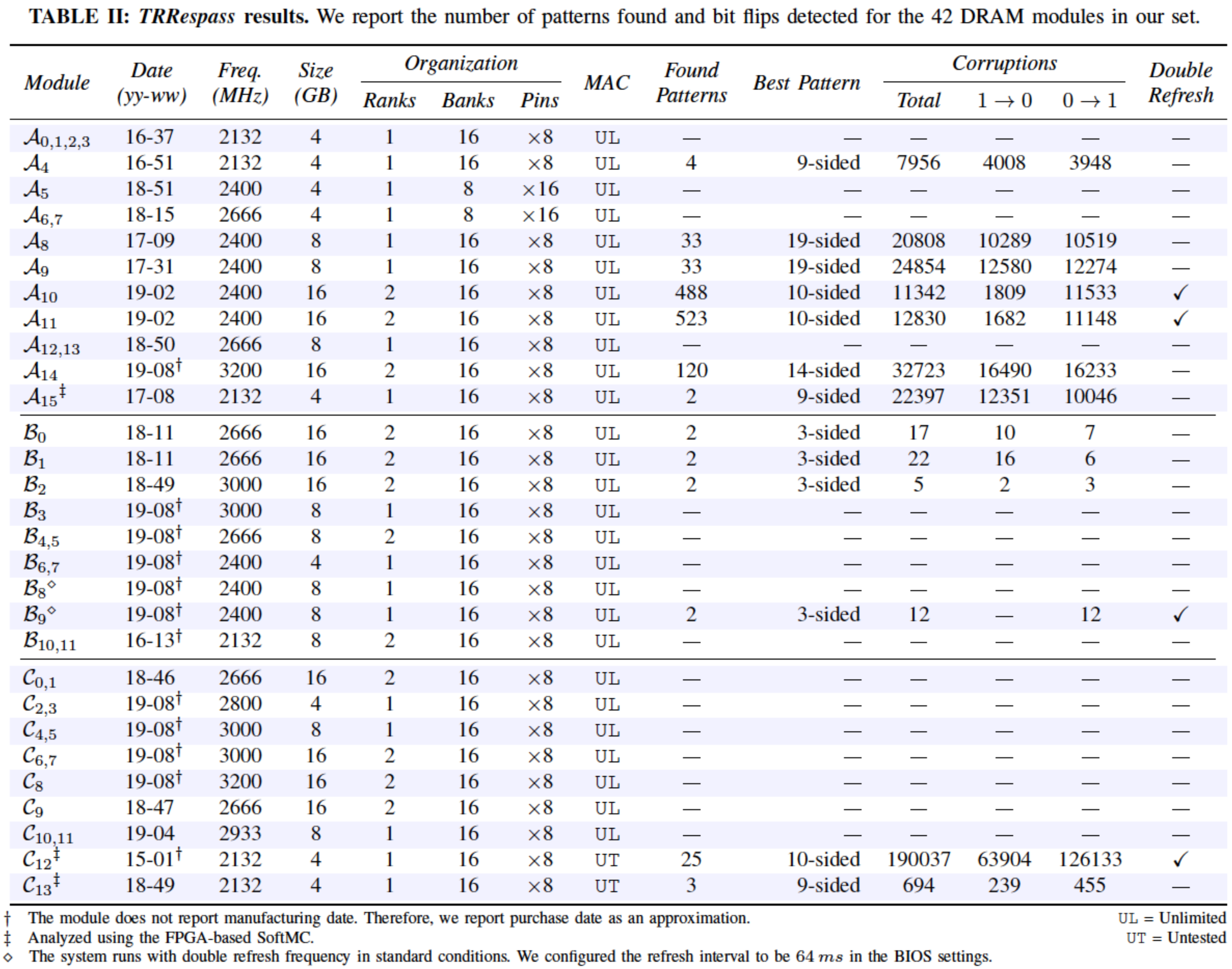


TABLE II: *TRRespass* results. We report the number of patterns found and bit flips detected for the 42 DRAM modules in our set.

Module	*Date*	*Freq.*	*Size*	*Organization*			*MAC*	*Found Patterns*	*Best Pattern*	*Corruptions*			*Double Refresh*
	(yy-ww)	*(MHz)*	*(GB)*	*Ranks*	*Banks*	*Pins*				*Total*	1 → 0	0 → 1	
$\mathcal{A}_{0,1,2,3}$	16-37	2132	4	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{A}_{4}$	16-51	2132	4	1	16	×8	UL	4	9-sided	7956	4008	3948	—
$\mathcal{A}_{5}$	18-51	2400	4	1	8	×16	UL	—	—	—	—	—	—
$\mathcal{A}_{6,7}$	18-15	2666	4	1	8	×16	UL	—	—	—	—	—	—
$\mathcal{A}_{8}$	17-09	2400	8	1	16	×8	UL	33	19-sided	20808	10289	10519	—
$\mathcal{A}_{9}$	17-31	2400	8	1	16	×8	UL	33	19-sided	24854	12580	12274	—
$\mathcal{A}_{10}$	19-02	2400	16	2	16	×8	UL	488	10-sided	11342	1809	11533	✓
$\mathcal{A}_{11}$	19-02	2400	16	2	16	×8	UL	523	10-sided	12830	1682	11148	✓
$\mathcal{A}_{12,13}$	18-50	2666	8	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{A}_{14}$	19-08†	3200	16	2	16	×8	UL	120	14-sided	32723	16490	16233	—
$\mathcal{A}_{15}^{\ddagger}$	17-08	2132	4	1	16	×8	UL	2	9-sided	22397	12351	10046	—
$\mathcal{B}_{0}$	18-11	2666	16	2	16	×8	UL	2	3-sided	17	10	7	—
$\mathcal{B}_{1}$	18-11	2666	16	2	16	×8	UL	2	3-sided	22	16	6	—
$\mathcal{B}_{2}$	18-49	3000	16	2	16	×8	UL	2	3-sided	5	2	3	—
$\mathcal{B}_{3}$	19-08†	3000	8	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{B}_{4,5}$	19-08†	2666	8	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{B}_{6,7}$	19-08†	2400	4	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{B}_{8}^{\diamond}$	19-08†	2400	8	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{B}_{9}^{\diamond}$	19-08†	2400	8	1	16	×8	UL	2	3-sided	12	—	12	✓
$\mathcal{B}_{10,11}$	16-13†	2132	8	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{0,1}$	18-46	2666	16	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{2,3}$	19-08†	2800	4	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{4,5}$	19-08†	3000	8	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{6,7}$	19-08†	3000	16	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{8}$	19-08†	3200	16	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{9}$	18-47	2666	16	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{10,11}$	19-04	2933	8	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{12}^{\ddagger}$	15-01†	2132	4	1	16	×8	UT	25	10-sided	190037	63904	126133	✓
$\mathcal{C}_{13}^{\ddagger}$	18-49	2132	4	1	16	×8	UT	3	9-sided	694	239	455	—

† The module does not report manufacturing date. Therefore, we report purchase date as an approximation.
‡ Analyzed using the FPGA-based SoftMC.
◇ The system runs with double refresh frequency in standard conditions. We configured the refresh interval to be 64 ms in the BIOS settings.

UL = Unlimited
UT = Untested

TRRespass Vulnerable Mobile Phones

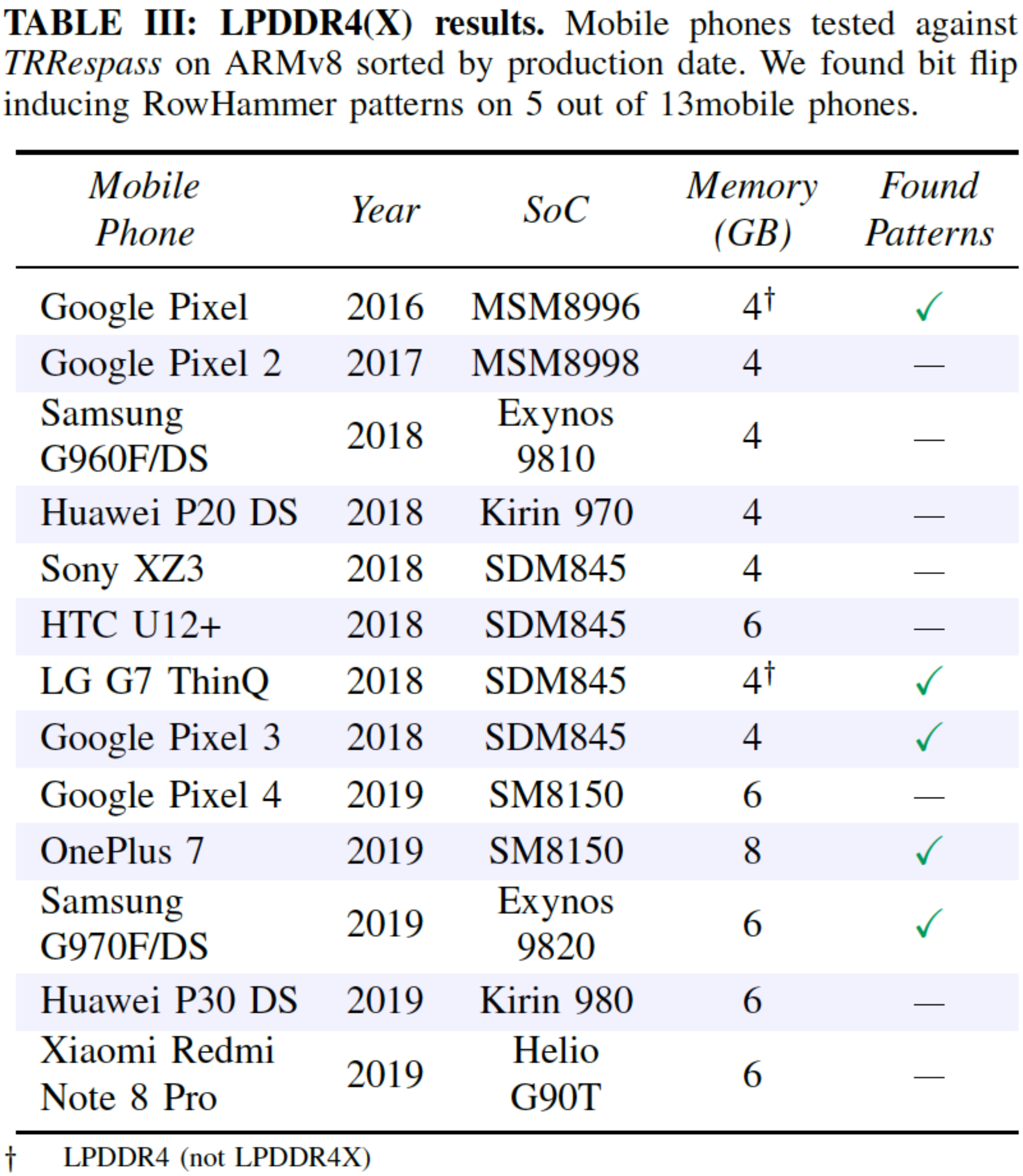


TABLE III: LPDDR4(X) results. Mobile phones tested against *TRRespass* on ARMv8 sorted by production date. We found bit flip inducing RowHammer patterns on 5 out of 13mobile phones.

Mobile Phone	*Year*	*SoC*	*Memory (GB)*	*Found Patterns*
Google Pixel	2016	MSM8996	4†	✓
Google Pixel 2	2017	MSM8998	4	—
Samsung G960F/DS	2018	Exynos 9810	4	—
Huawei P20 DS	2018	Kirin 970	4	—
Sony XZ3	2018	SDM845	4	—
HTC U12+	2018	SDM845	6	—
LG G7 ThinQ	2018	SDM845	4†	✓
Google Pixel 3	2018	SDM845	4	✓
Google Pixel 4	2019	SM8150	6	—
OnePlus 7	2019	SM8150	8	✓
Samsung G970F/DS	2019	Exynos 9820	6	✓
Huawei P30 DS	2019	Kirin 980	6	—
Xiaomi Redmi Note 8 Pro	2019	Helio G90T	6	—

† LPDDR4 (not LPDDR4X)

TRRespass Based RowHammer Attack

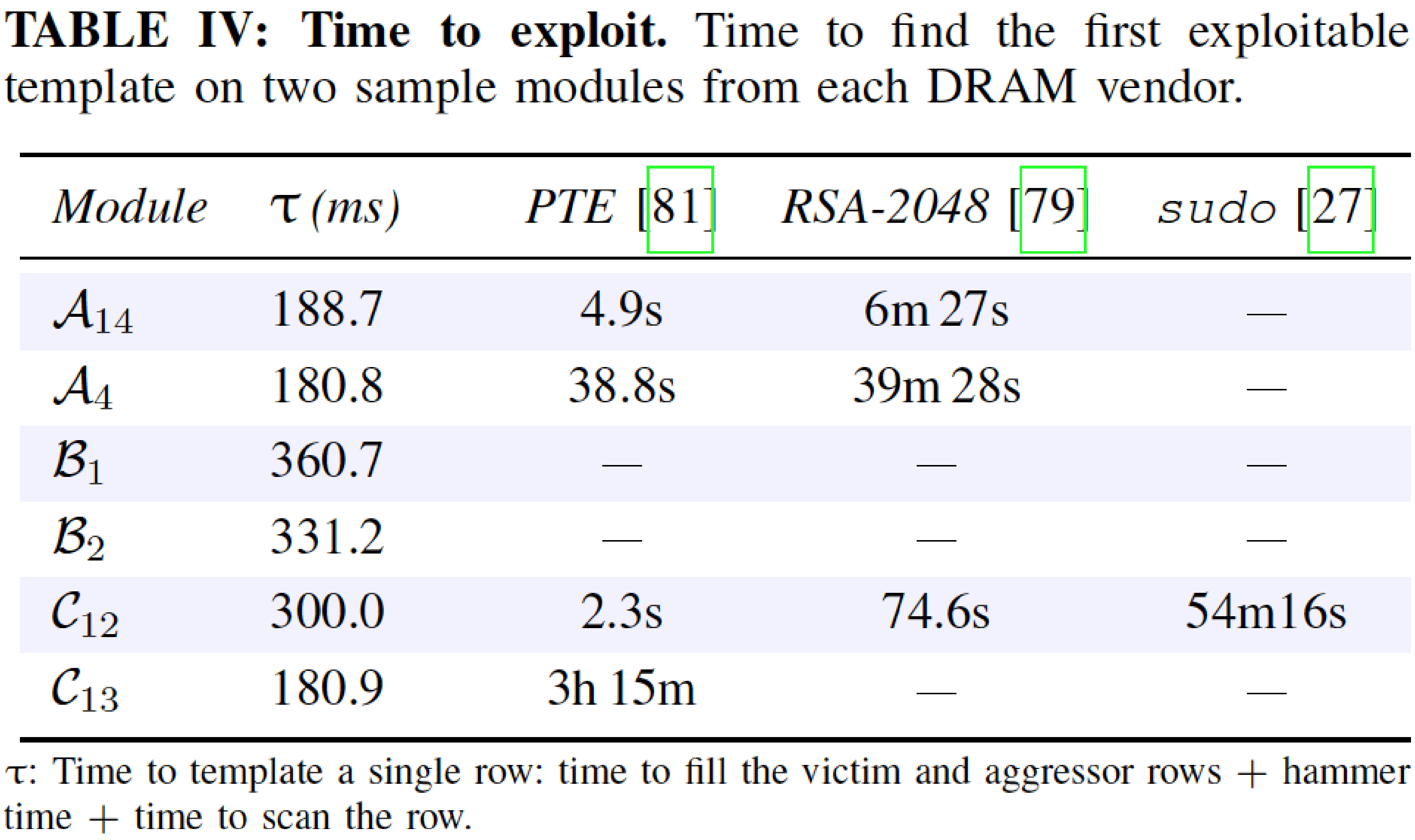


TABLE IV: Time to exploit. Time to find the first exploitable template on two sample modules from each DRAM vendor.

Module	τ *(ms)*	*PTE* [81]	*RSA-2048* [79]	sudo [27]
$\mathcal{A}_{14}$	188.7	4.9s	6m 27s	—
$\mathcal{A}_{4}$	180.8	38.8s	39m 28s	—
$\mathcal{B}_{1}$	360.7	—	—	—
$\mathcal{B}_{2}$	331.2	—	—	—
$\mathcal{C}_{12}$	300.0	2.3s	74.6s	54m16s
$\mathcal{C}_{13}$	180.9	3h 15m	—	—

τ: Time to template a single row: time to fill the victim and aggressor rows + hammer time + time to scan the row.

TRRespass Key Results

- 13 out of 42 tested DDR4 DRAM modules are vulnerable
  - From all 3 major manufacturers
  - 3-, 9-, 10-, 14-, 19-sided hammer attacks needed

- 5 out of 13 mobile phones tested vulnerable
  - From 4 major manufacturers
  - With LPDDR4(X) DRAM chips

- These results are scratching the surface
  - TRRespass tool is not exhaustive
  - There is a lot of room for uncovering more vulnerable chips and phones

TRRespass Key Takeaways

RowHammer is still
an open problem

Security by obscurity
is likely not a good solution

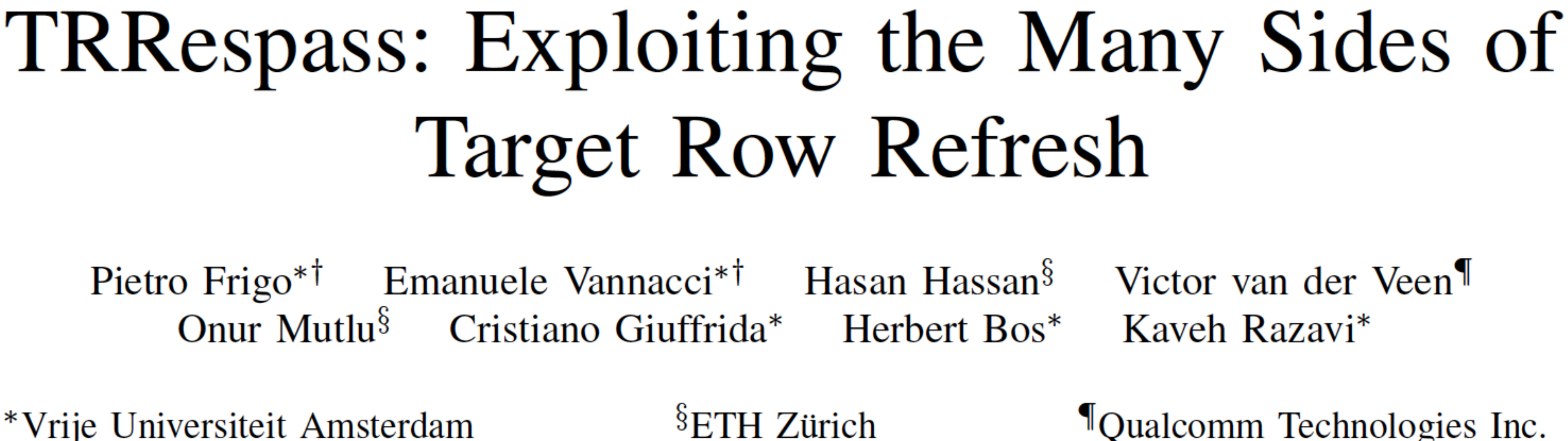
Detailed Lecture on TRRespass

- Computer Architecture, Fall 2020, Lecture 5a
  - RowHammer in 2020: TRRespass (ETH Zürich, Fall 2020)
  - https://www.youtube.com/watch?v=pwRw7QqK_qA&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=9

https://www.youtube.com/onurmutlulectures

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*

*Vrije Universiteit Amsterdam §ETH Zürich ¶Qualcomm Technologies Inc.

How to Guarantee That a Chip is RowHammer-Free?

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)]

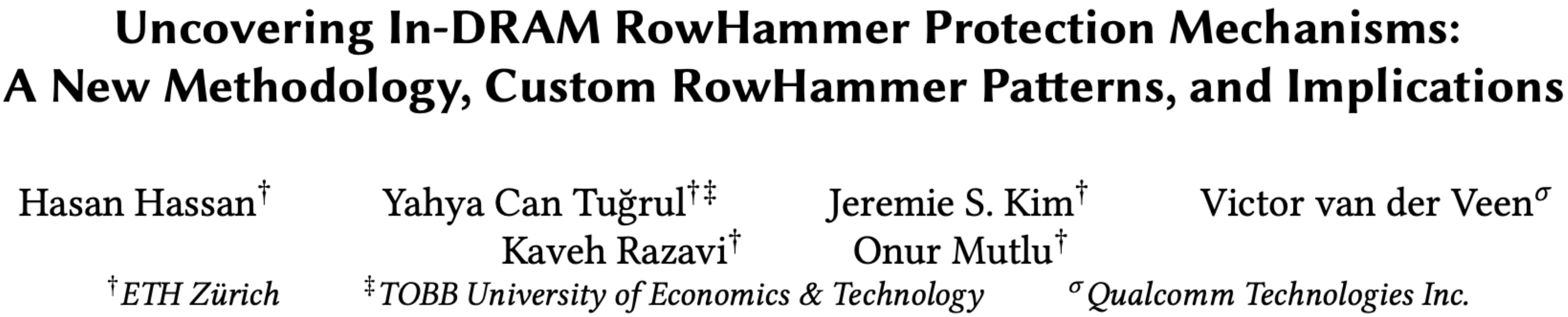
Are We Susceptible to Rowhammer?
An End-to-End Methodology for Cloud Providers

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

Uncovering TRR Almost Completely

Industry-Adopted Solutions Are Very 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[†] Yahya Can Tuğrul[†‡] Jeremie S. Kim[†] Victor van der Veen[σ]
Kaveh Razavi[†] Onur Mutlu[†]
[†]*ETH Zürich* [‡]*TOBB University of Economics & Technology* [σ]*Qualcomm Technologies Inc.*

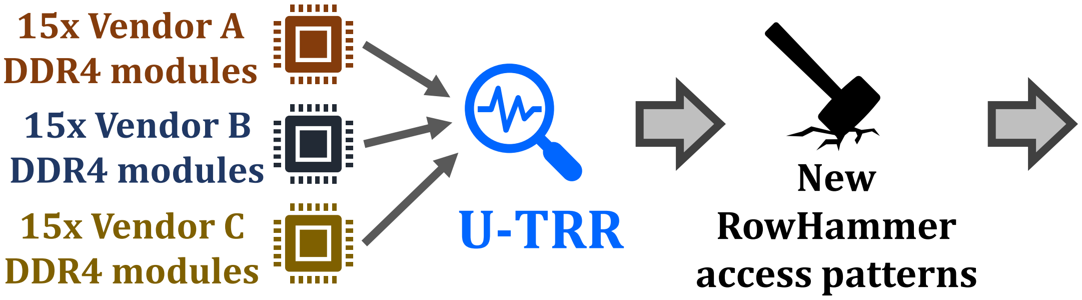
U-TRR Summary & Key Results

Target Row Refresh (TRR):
a set of obscure, undocumented, and proprietary RowHammer mitigation techniques

We cannot easily study the *security properties* of TRR

Is TRR fully secure? How can we validate its security guarantees?

U-TRR | A new methodology that leverages *data retention failures* to uncover the inner workings of TRR and study its security


15x Vendor A DDR4 modules
15x Vendor B DDR4 modules
15x Vendor C DDR4 modules
U-TRR
New RowHammer access patterns

All 45 modules we test are **vulnerable**

99.9% of rows in a DRAM bank experience **at least one RowHammer bit flip**

Up to **7** RowHammer **bit flips** in an 8-byte dataword, **making ECC ineffective**

TRR **does not provide security** against RowHammer

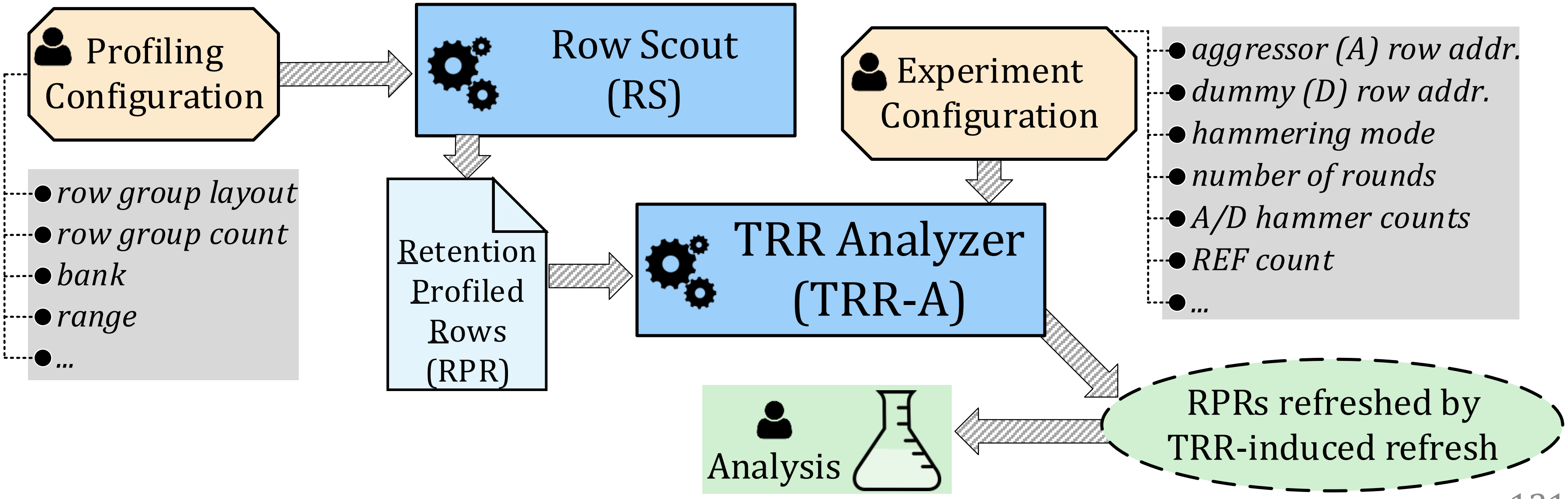
U-TRR can facilitate the development of **new RowHammer attacks** and **more secure RowHammer protection** mechanisms

SAFARI

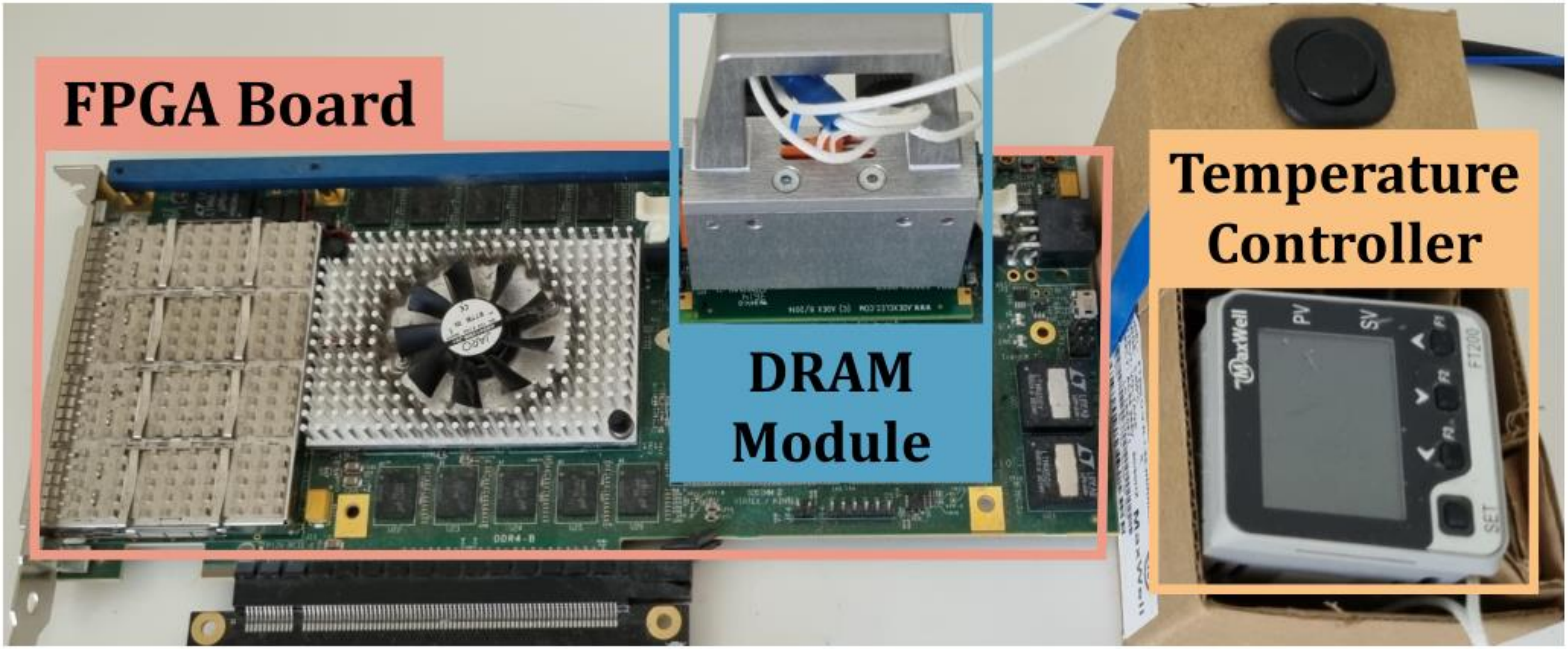
Overview of U-TRR

U-TRR: A new methodology to *uncover* the inner workings of TRR

Key idea: Use data retention failures as a side channel to detect when a row is refreshed by TRR

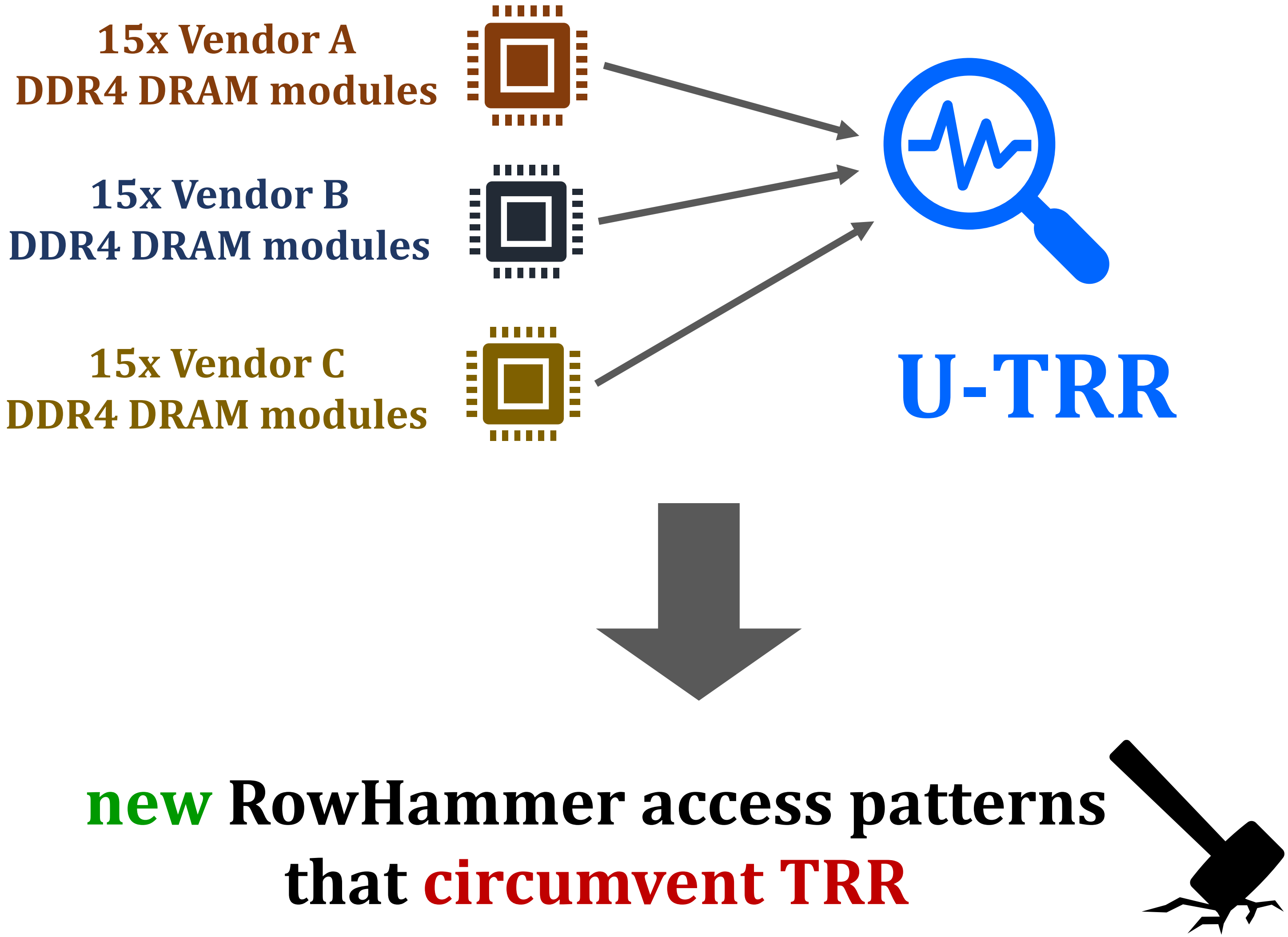


Analyzing TRR-Protected DDR4 Chips



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

U-TRR Analysis Summary



15x Vendor A
DDR4 DRAM modules

15x Vendor B
DDR4 DRAM modules

15x Vendor C
DDR4 DRAM modules

U-TRR

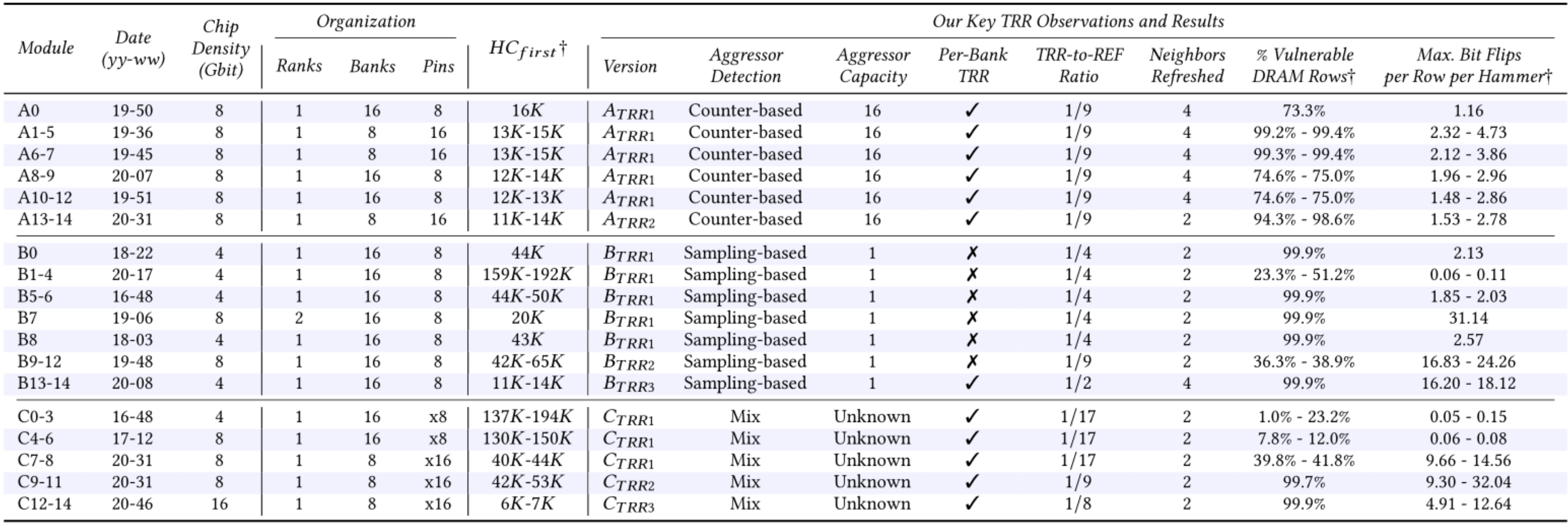
new RowHammer access patterns that circumvent TRR

Key Takeaways

All 45 modules we test are vulnerable

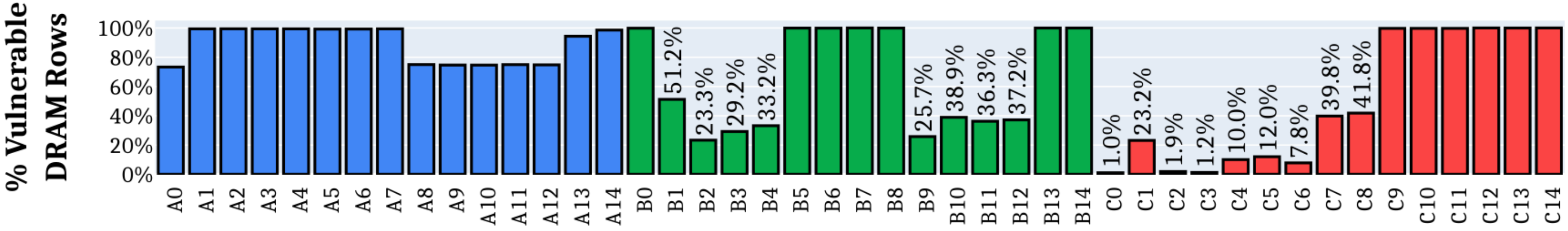
99.9% of rows in a DRAM bank
experience **at least one RowHammer bit flip**

ECC is ineffective: up to 7 RowHammer bit flips
in an 8-byte dataword



Module	Date (yy-ww)	Chip Density (Gbit)	Organization			HC_{first}†	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

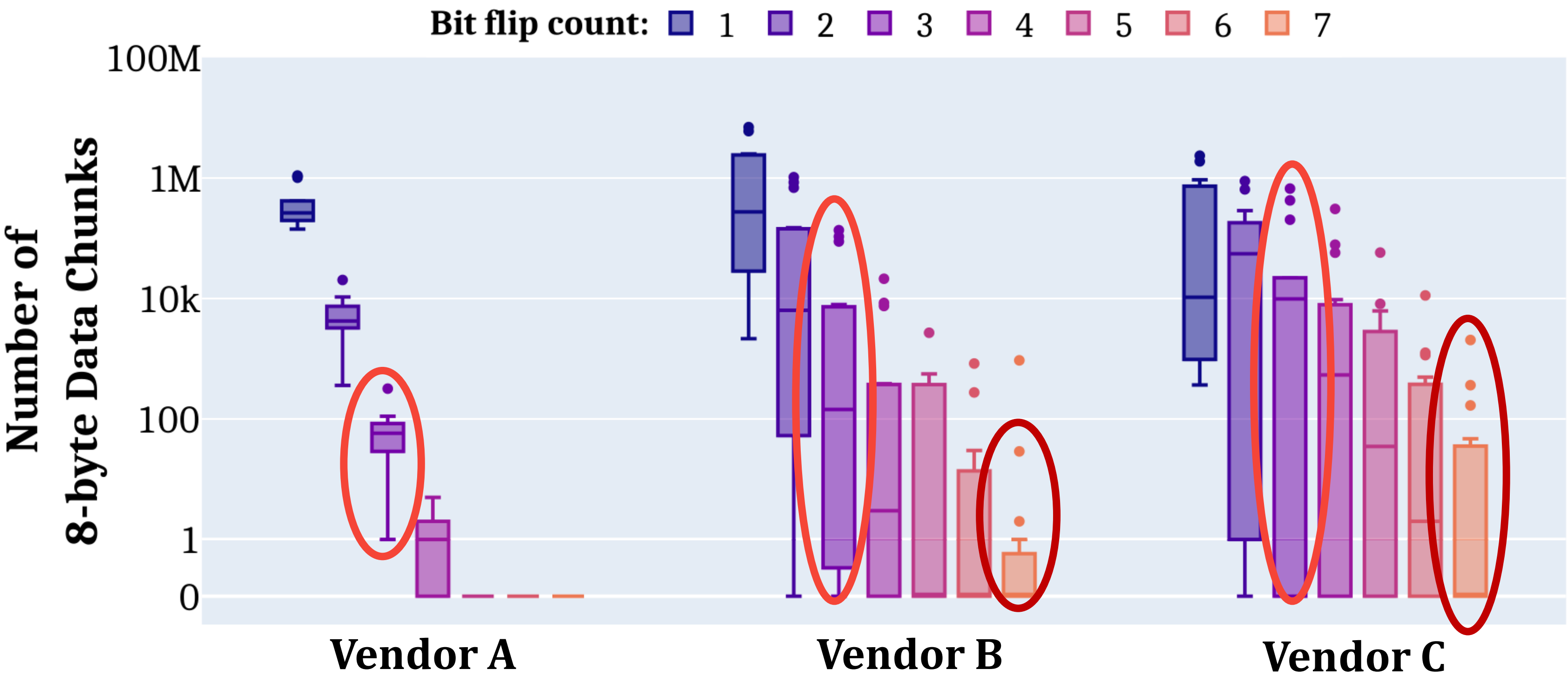
Effect on Individual Rows



All 45 modules we tested are vulnerable
to our new RowHammer access patterns

Our RowHammer access patterns
cause bit flips in more than 99.9% of the rows

Bypassing ECC with New RowHammer Patterns

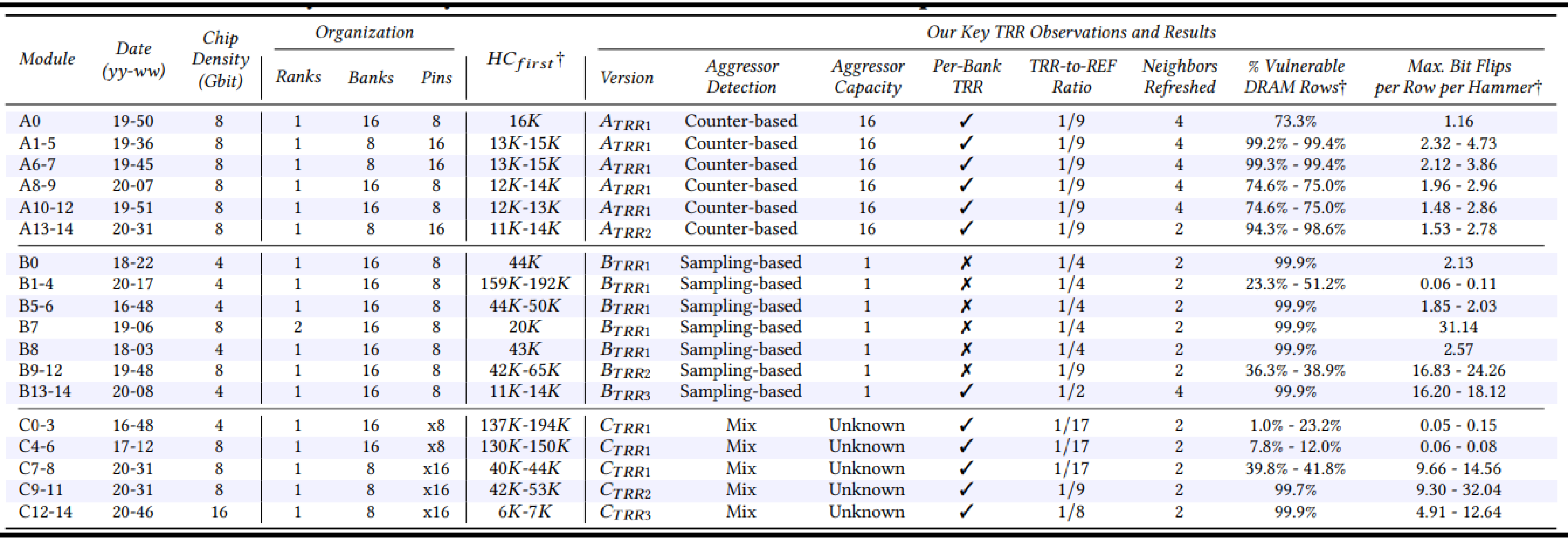


Modules from all three vendors have many **8-byte data chunks** with 3 and more (up to 7) RowHammer bit flips

Conventional DRAM ECC cannot protect against our new RowHammer access patterns

Many Observations & Results in the Paper

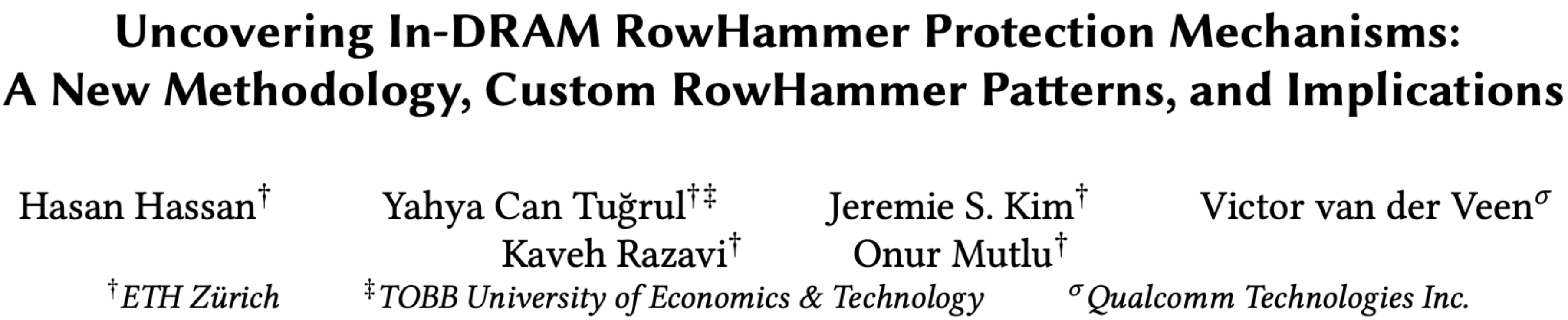
- More observations on the TRRs of the three vendors
- Detailed description of the crafted access patterns
- Hammers per aggressor row sensitivity analysis
- Observations and results for individual modules
- ...



Module	Date (yy-ww)	Chip Density (Gbit)	Organization			HC_{first}†	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

Uncovering TRR Can Help Future Solutions

- 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[†] Yahya Can Tuğrul[†‡] Jeremie S. Kim[†] Victor van der Veen[σ]
Kaveh Razavi[†] Onur Mutlu[†]
[†]*ETH Zürich* [‡]*TOBB University of Economics & Technology* [σ]*Qualcomm Technologies Inc.*

New RowHammer Characteristics

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

Our Goal

Provide insights into **three fundamental properties**

Temperature

Aggressor Row Active Time

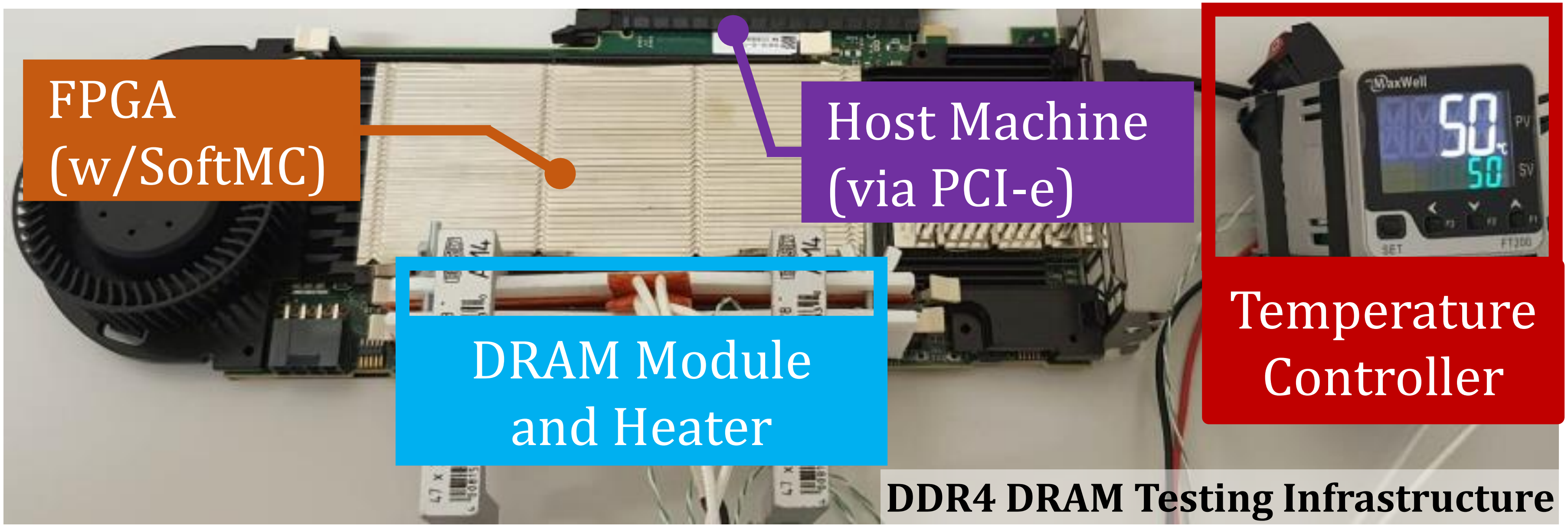
Victim DRAM Cell's Physical Location

To find **effective and efficient** attacks and defenses

DRAM Testing Infrastructures

Two separate testing infrastructures

1. **DDR3:** FPGA-based SoftMC (Xilinx ML605)
2. **DDR4:** FPGA-based SoftMC (Xilinx Virtex UltraScale+ XCU200)



DDR4 DRAM Testing Infrastructure

Fine-grained control over **DRAM commands**, **timing parameters** and **temperature (±0.1°C)**

DRAM Chips Tested

Two DRAM standards

Mfr.	DDR4 DIMMs	DDR3 SODIMMs	# Chips	Density	Die	Org.
A (Micron)	9	1	144 (8)	8Gb (4Gb)	B (P)	x4 (x8)
B (Samsung)	4	1	32 (8)	4Gb (4Gb)	F (Q)	x8 (x8)
C (SK Hynix)	5	1	40 (8)	4Gb (4Gb)	B (B)	x8 (x8)
D (Nanya)	4	-	32 (-)	8Gb (-)	C (-)	x8 (-)

4 Major Manufacturers

272 DRAM Chips in total

Summary of The Study & Key Results

- **272 DRAM chips** from **four major manufacturers**

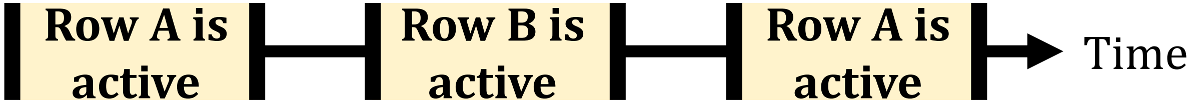
- **6 major takeaways** from **16 novel observations**

- A RowHammer bit flip is **more likely to occur**
  1) in **a bounded range of temperature**
  2) if the aggressor row is **active for longer time**
  3) in **certain physical regions** of the DRAM module under attack

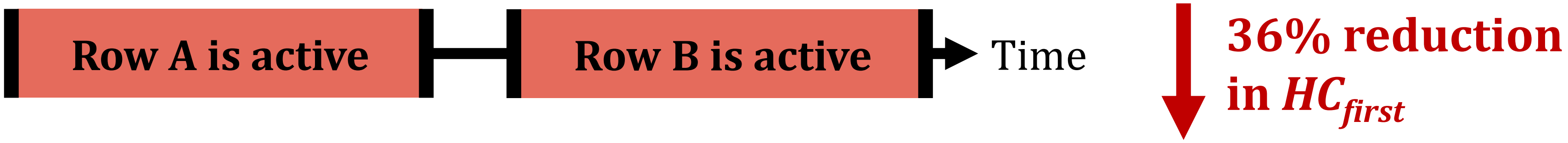
- Our novel observations can inspire and aid future work
  - Craft **more effective attacks**
  - Design **more effective and efficient defenses**

Example Attack Improvement 3:
Bypassing Defenses with Aggressor Row Active Time

Activating aggressor rows as frequently as possible:



Keeping aggressor rows active for a longer time:



Reduces the minimum activation count to induce a bit flip **by 36%**

Bypasses defenses that do not account for this reduction

Key Takeaways from Spatial Variation Analysis

Key Takeaway 5

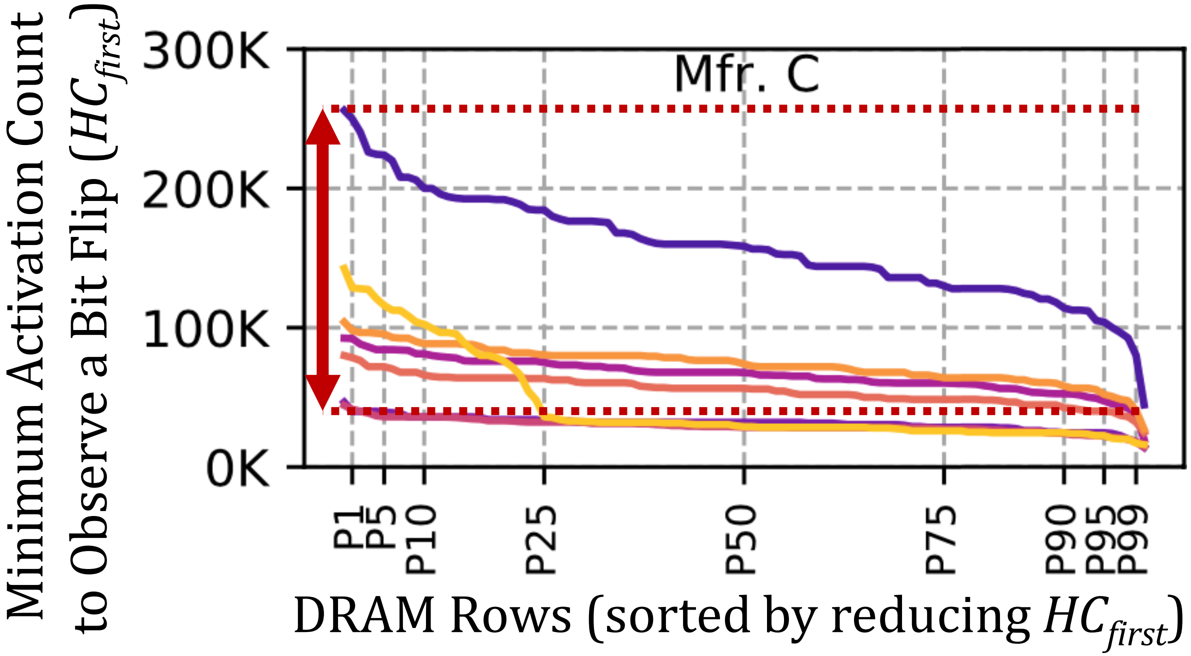
RowHammer vulnerability **significantly varies** across DRAM rows and columns due to **design-induced** and **manufacturing-process-induced** variation

Key Takeaway 6

The distribution of **the minimum activation count to observe bit flips (*HC_{first}*)** exhibits **a diverse set of values in a subarray** but **similar values across subarrays** in the same DRAM module

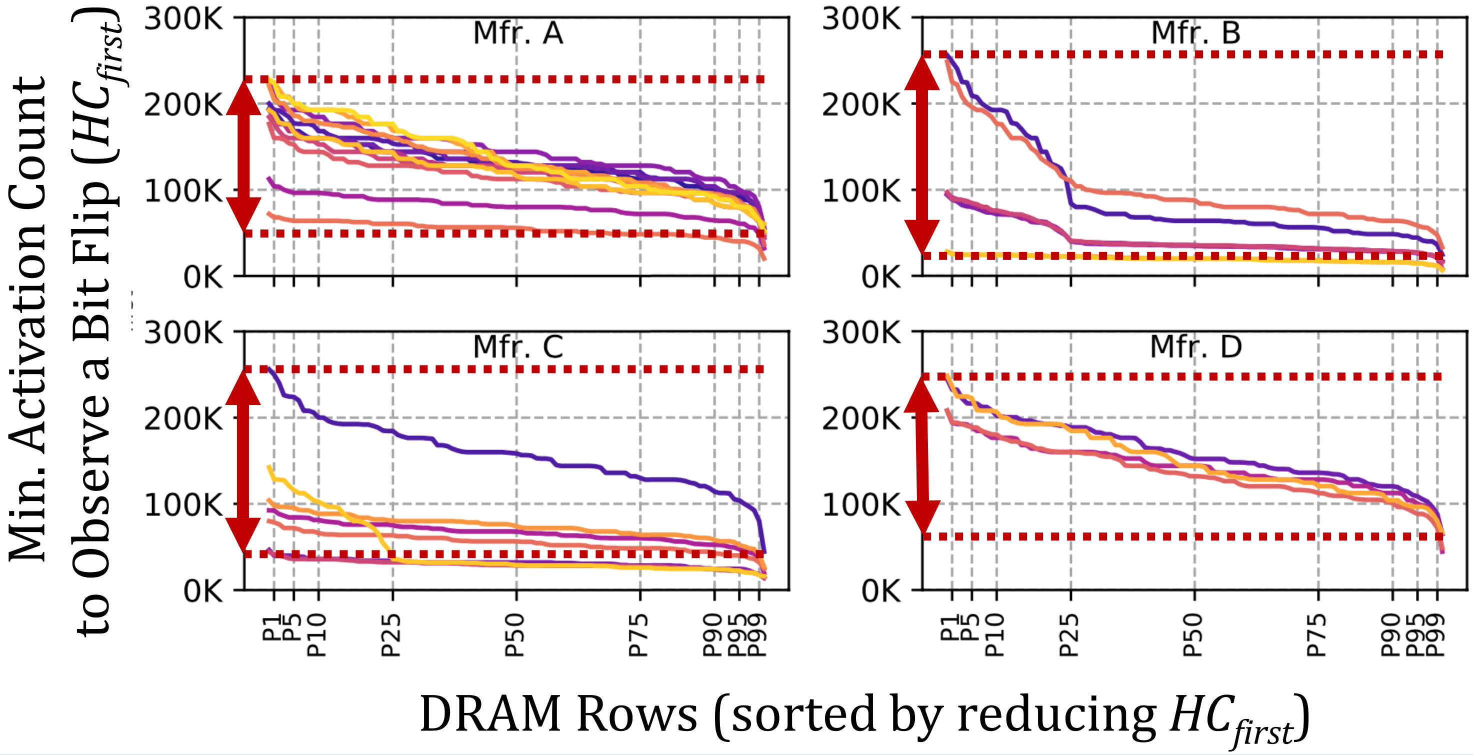
Spatial Variation across Rows

The **minimum activation count** to observe bit flips **(*HC_{first}*)** across **DRAM rows:**



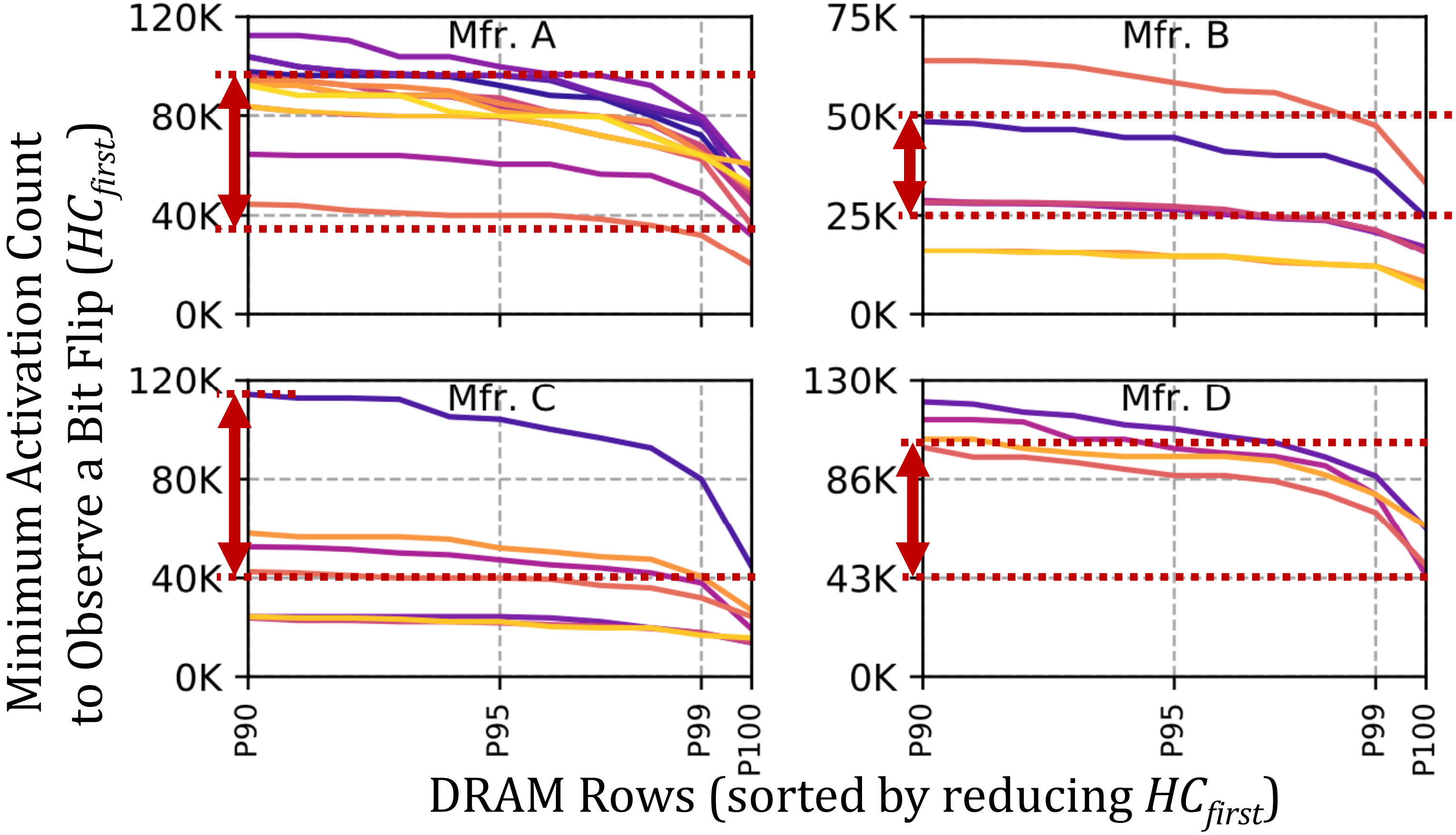
The RowHammer vulnerability **significantly varies** across DRAM rows

Spatial Variation across Rows



The RowHammer vulnerability **significantly varies** across DRAM rows

Spatial Variation across Rows

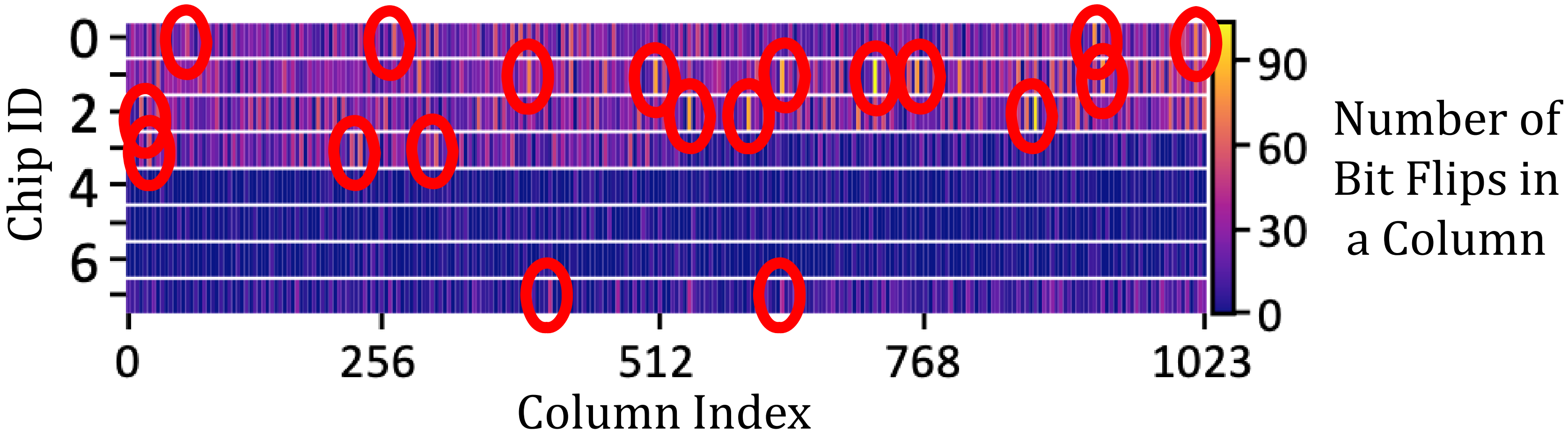


OBSERVATION 12

A small fraction of DRAM rows are **significantly more vulnerable** to RowHammer than **the vast majority** of the rows

SAFARI

Spatial Variation across Columns

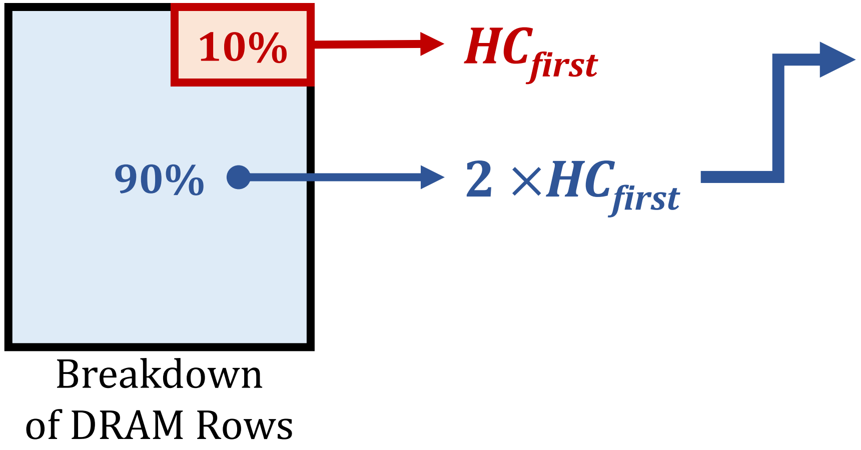


OBSERVATION 13

Certain columns are **significantly more vulnerable** to RowHammer than other columns

Example Defense Improvements

- **Example 1: Leveraging variation across DRAM rows**



10%

HC_{first}

90%

$2 \times HC_{first}$

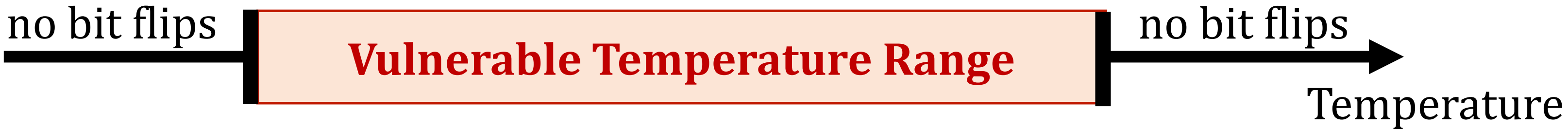
Breakdown
of DRAM Rows

Aggressiveness can be reduced:

33% area reduction
for BlockHammer [Yağlıkçı+, HPCA'21]

80% area reduction
for Graphene [Park+, MICRO'20]

- **Example 2: Leveraging variation with temperature**
  - A DRAM cell experiences **bit flips** within **a bounded temperature range**



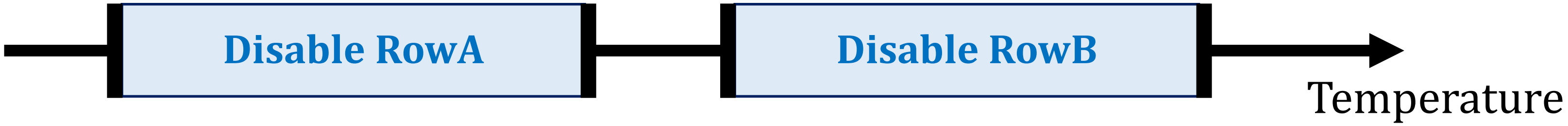
no bit flips

Vulnerable Temperature Range

no bit flips

Temperature

  - A row can be **disabled** within the row's **vulnerable temperature range**



Disable RowA

Disable RowB

Temperature

Many More Analyses In The Paper

- 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]

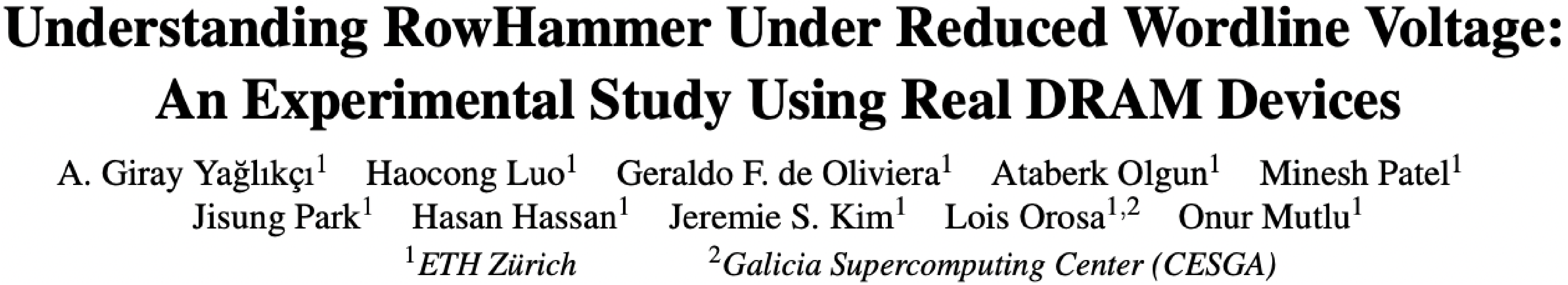
A Deeper Look into RowHammer's Sensitivities: Experimental Analysis of Real DRAM Chips and Implications on Future Attacks and Defenses

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

More RowHammer Analysis

RowHammer vs. Wordline Voltage (2022)

- A. Giray Yağlıkçı, Haocong Luo, Geraldo F. de Oliviera, Ataberk Olgun, Minesh Patel, Jisung Park, Hasan Hassan, Jeremie S. Kim, Lois Orosa, and Onur Mutlu, **"Understanding RowHammer Under Reduced Wordline Voltage: An Experimental Study Using Real DRAM Devices"**
  Proceedings of the 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (***DSN***), Baltimore, MD, USA, June 2022.
  [Slides (pptx) (pdf)]
  [Lightning Talk Slides (pptx) (pdf)]
  [arXiv version]
  [Talk Video (34 minutes, including Q&A)]
  [Lightning Talk Video (2 minutes)]

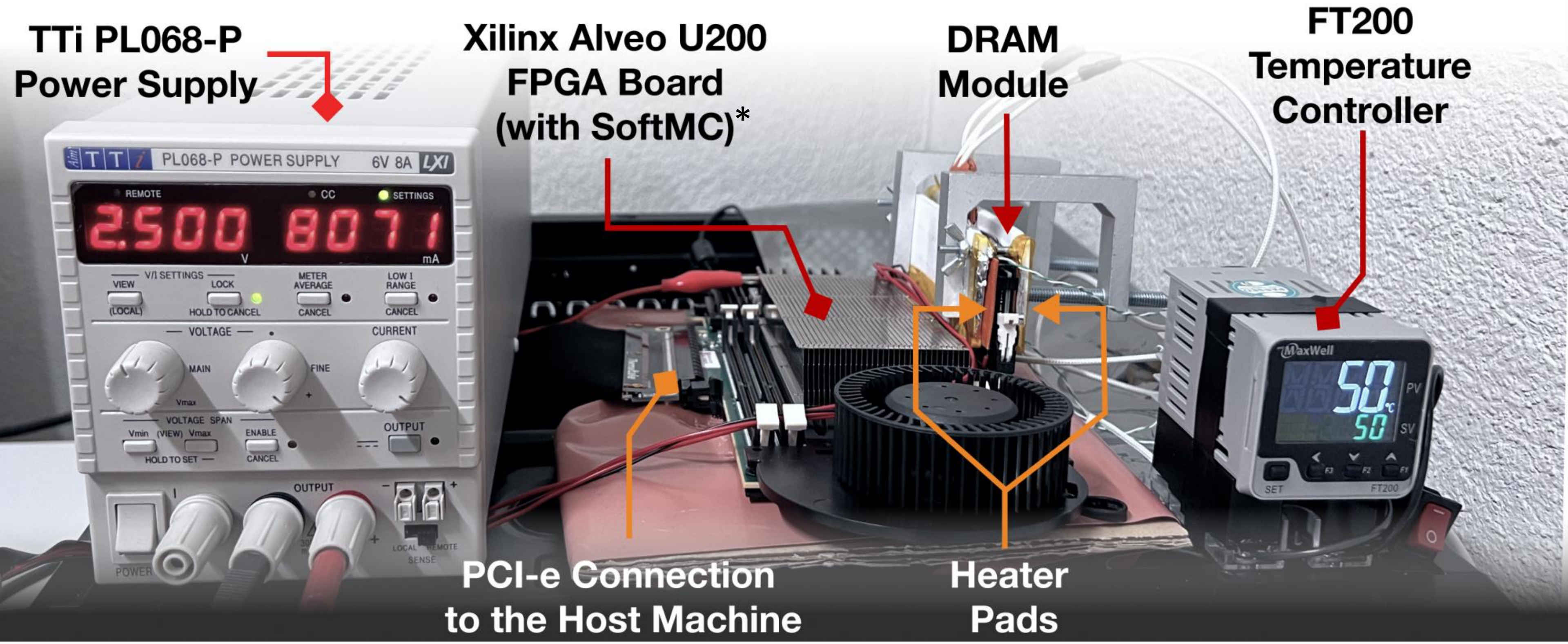


Understanding RowHammer Under Reduced Wordline Voltage: An Experimental Study Using Real DRAM Devices

A. Giray Yağlıkçı[1] Haocong Luo[1] Geraldo F. de Oliviera[1] Ataberk Olgun[1] Minesh Patel[1]
Jisung Park[1] Hasan Hassan[1] Jeremie S. Kim[1] Lois Orosa[1,2] Onur Mutlu[1]
[1]*ETH Zürich* [2]*Galicia Supercomputing Center (CESGA)*

Updated DRAM Testing Infrastructure

FPGA-based SoftMC (Xilinx Virtex UltraScale+ XCU200)


TTi PL068-P Power Supply
Xilinx Alveo U200 FPGA Board (with SoftMC)*
DRAM Module
FT200 Temperature Controller
PCI-e Connection to the Host Machine
Heater Pads

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

*Hassan et al., "SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies," in HPCA, 2017. [Available on GitHub: https://github.com/CMU-SAFARI/SoftMC]

Summary

We provide *the first* RowHammer characterization **under reduced wordline voltage**

Experimental results with *272 real DRAM chips* show that **reducing wordline voltage:**

1. **Reduces RowHammer vulnerability**
   - **Bit error rate** caused by a RowHammer attack reduces by **15.2% (66.9% max)**
   - A row needs to be activated **7.4% more times (85.8% max)** to induce *the first* bit flip

2. **Increases row activation latency**
   - More than **76%** of the tested DRAM chips **reliably operate** using **nominal** timing parameters
   - Remaining **24%** **reliably operate** with **increased** (up to 24ns) row activation latency

3. **Reduces data retention time**
   - **80%** of the tested DRAM chips **reliably operate using nominal refresh rate**
   - Remaining **20%** **reliably operate** by
     - Using **single error correcting codes**
     - **Doubling the refresh rate** for **a small fraction (16.4%) of DRAM rows**

Reducing wordline voltage can **reduce RowHammer vulnerability**
without significantly affecting **reliable DRAM operation**

We Covered Until Here in Lecture. To Be Continued…

Computer Architecture

Lecture 6: The Story of RowHammer

Memory Security & Reliability

Prof. Onur Mutlu

ETH Zürich

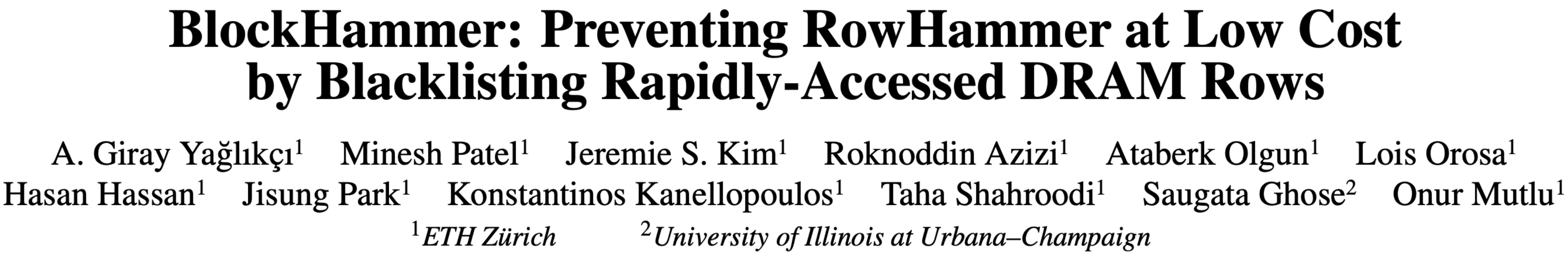
Fall 2022

14 October 2022

New RowHammer Solutions

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)]
  [Intel Hardware Security Academic Awards Short Talk Slides (pptx) (pdf)]
  [Talk Video (22 minutes)]
  [Short Talk Video (7 minutes)]
  [Intel Hardware Security Academic Awards Short Talk Video (2 minutes)]
  [BlockHammer Source Code]
  Intel Hardware Security Academic Award Finalist (one of 4 finalists out of 34 nominations)

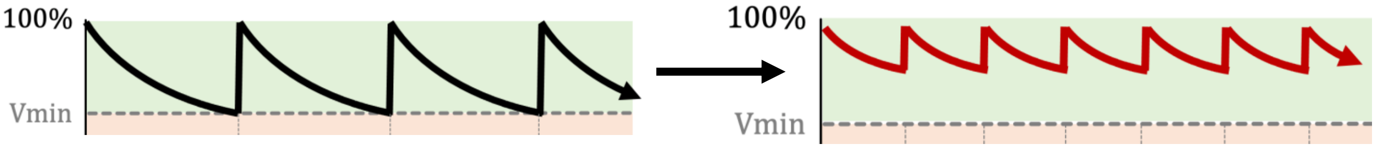


BlockHammer: Preventing RowHammer at Low Cost by Blacklisting Rapidly-Accessed DRAM Rows

A. Giray Yağlıkçı[1] Minesh Patel[1] Jeremie S. Kim[1] Roknoddin Azizi[1] Ataberk Olgun[1] Lois Orosa[1]
Hasan Hassan[1] Jisung Park[1] Konstantinos Kanellopoulos[1] Taha Shahroodi[1] Saugata Ghose[2] Onur Mutlu[1]
[1]*ETH Zürich* [2]*University of Illinois at Urbana–Champaign*

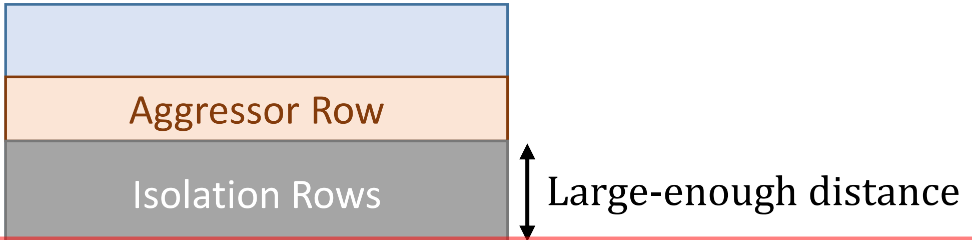
RowHammer Solution Approaches

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



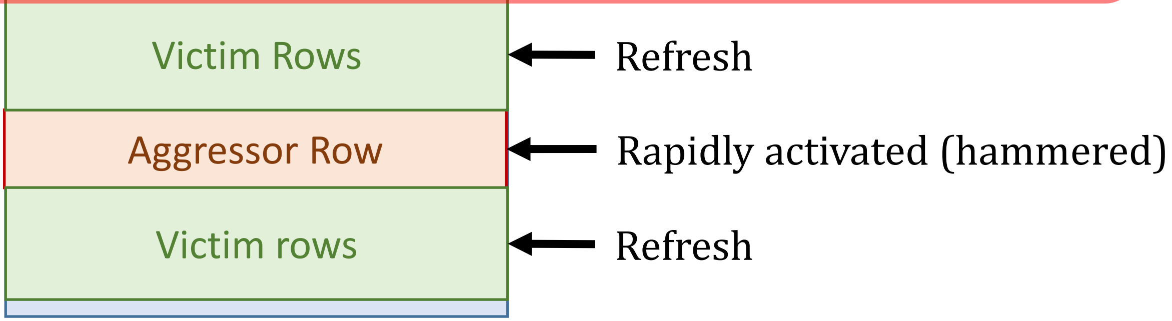
Fewer activations possible in a refresh interval

- Physical isolation

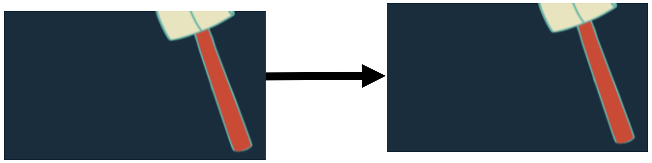


Cost, Power, Performance, Complexity

- Reactive refresh



- Proactive throttling



Fewer activations allowed for aggressive applications

Two Key Challenges

1 **Scalability**
with worsening RowHammer vulnerability

2 **Compatibility**
with commodity DRAM chips

Our Goal

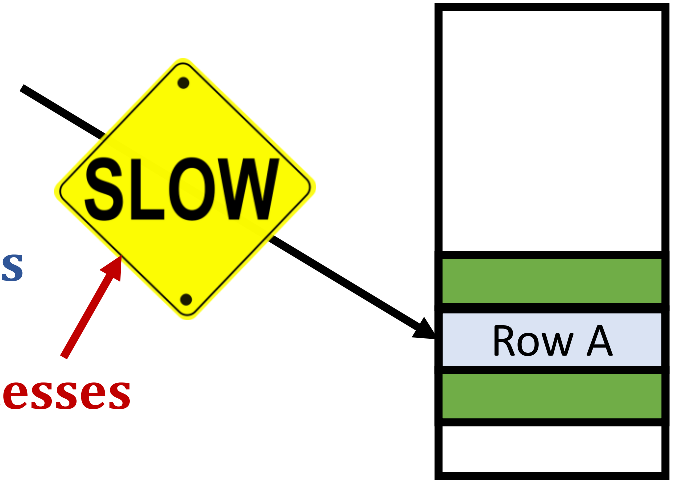
To prevent RowHammer efficiently and scalably *without* knowledge of or modifications to DRAM internals

BlockHammer

Key Idea

Selectively throttle memory accesses that may cause RowHammer bit-flips

BlockHammer: Practical Throttling-based Mechanism



Physical
Row Layout

- A RowHammer attack hammers Row A
- BlockHammer detects a RowHammer attack using **area-efficient Bloom filters**
- BlockHammer **selectively throttles accesses** from within **the memory controller**
- Bit flips **do not** occur
- BlockHammer can *optionally* **inform the system software** about the attack

BlockHammer is compatible with commodity DRAM chips
No need for proprietary info of or modifications to DRAM chips

BlockHammer

Overview of Approach

RowBlocker

Tracks row activation rates using area-efficient Bloom filters

Blacklists rows that are activated at a high rate

Throttles activations targeting a blacklisted row

No row can be activated at a high enough rate to induce bit-flips

AttackThrottler

Identifies threads that perform a RowHammer attack

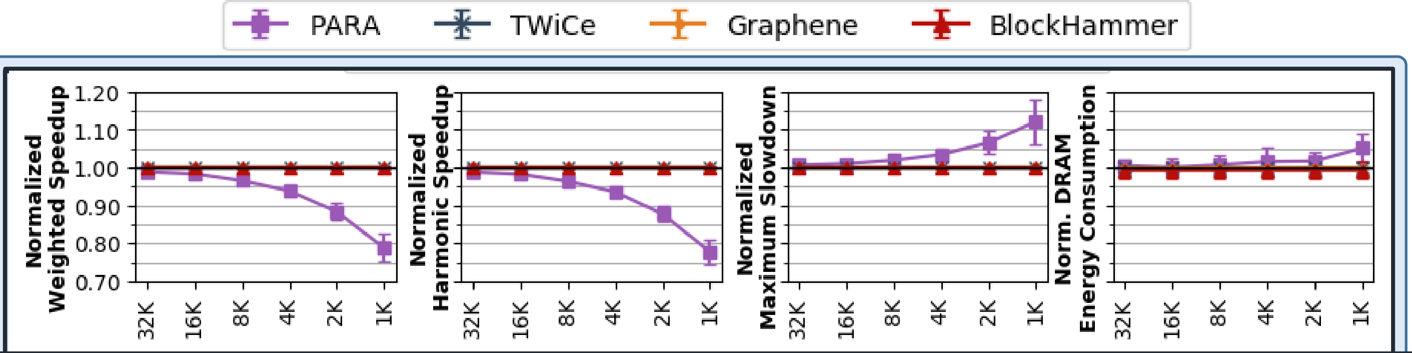
Reduces memory bandwidth usage of identified threads

Greatly reduces the **performance degradation** and **energy wastage** a RowHammer attack inflicts on a system

Evaluation: BlockHammer
Scaling with RowHammer Vulnerability

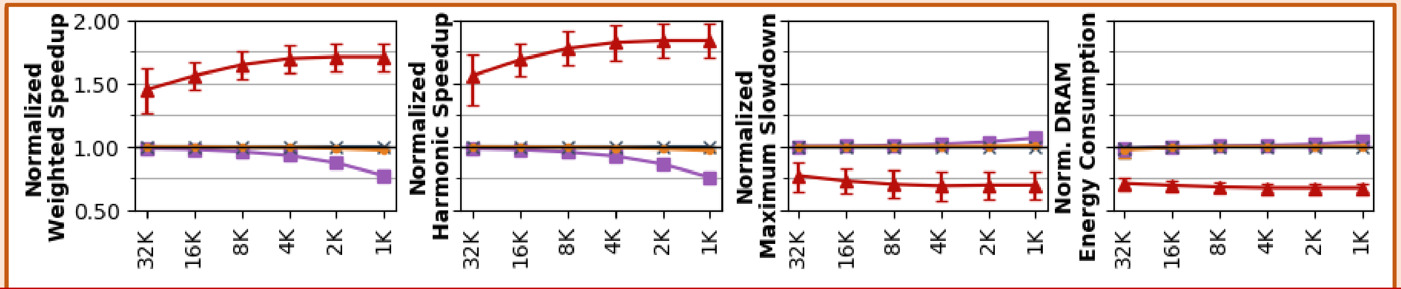
- System throughput (weighted speedup)
- Job turnaround time (harmonic speedup)
- Unfairness (maximum slowdown)
- DRAM energy consumption

No RowHammer Attack



BlockHammer’s performance and energy overheads remain **negligible (<0.6%)**

RowHammer Attack Present

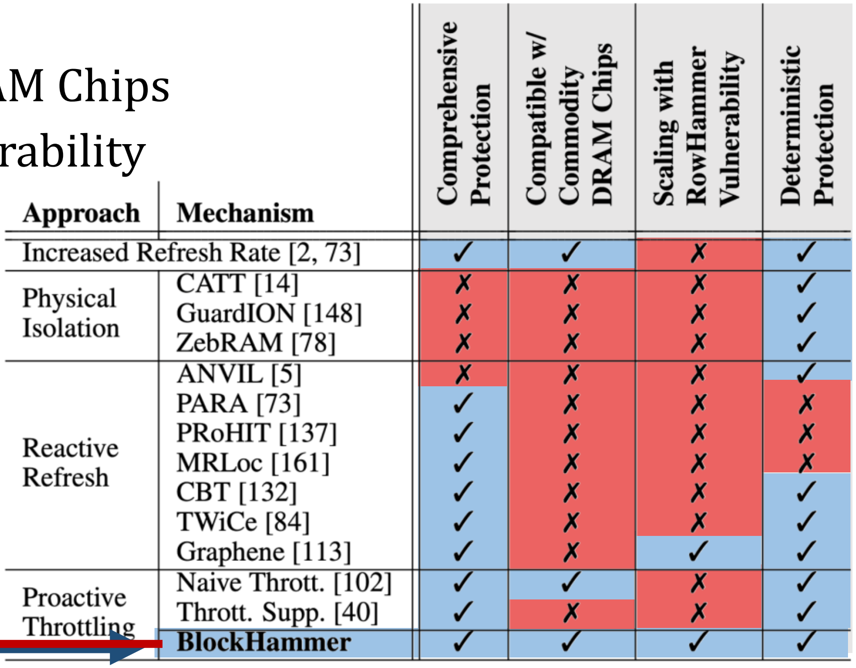


BlockHammer scalably provides **much higher performance** (71% on average) and **lower energy consumption** (32% on average) than state-of-the-art mechanisms

Key Results: BlockHammer

- **Competitive** with state-of-the-art mechanisms **when there is no attack**
- **Superior** performance and DRAM energy **when RowHammer attack present**
- **Better hardware area scaling with RowHammer vulnerability**
- **Security Proof**
- Addresses **Many-Sided Attacks**
- Evaluation of **14 mechanisms** across four desirable properties
  - Comprehensive Protection
  - Compatibility with Commodity DRAM Chips
  - Scalability with RowHammer Vulnerability
  - Deterministic Protection

BlockHammer is the only solution that satisfies all four desirable properties



Approach	Mechanism	Comprehensive Protection	Compatible w/ Commodity DRAM Chips	Scaling with RowHammer Vulnerability	Deterministic Protection
Increased Refresh Rate [2, 73]		✓	✓	✗	✓
Physical Isolation	CATT [14]	✗	✗	✗	✓
	GuardION [148]	✗	✗	✗	✓
	ZebRAM [78]	✗	✗	✗	✓
Reactive Refresh	ANVIL [5]	✗	✗	✗	✓
	PARA [73]	✓	✗	✗	✗
	PRoHIT [137]	✓	✗	✗	✗
	MRLoc [161]	✓	✗	✗	✗
	CBT [132]	✓	✗	✗	✓
	TWiCe [84]	✓	✗	✗	✓
	Graphene [113]	✓	✗	✓	✓
Proactive Throttling	Naive Thrott. [102]	✓	✓	✗	✓
	Thrott. Supp. [40]	✓	✗	✗	✓
	BlockHammer	✓	✓	✓	✓

More in the Paper: BlockHammer

Full Paper

- Using **area-efficient Bloom filters** for RowHammer detection
- Security Proof
  - Mathematically represent **all possible** access patterns
  - **No row can be activated high-enough times** to induce bit-flips
- BlockHammer prevents **many-sided attacks**
  - TRRespass [Frigo+, S&P'20]
  - U-TRR [Hassan+, MICRO'21]
  - BlackSmith [Jattke+, S&P'22]
  - Half-Double [Kogler+, USENIX Security'22]
- System Integration
  - **BlockHammer** can detect **RowHammer attacks** with **high accuracy** and **inform system software**
  - Measures **RowHammer likelihood of each thread**
- **Hardware complexity** analysis

Summary: BlockHammer

- BlockHammer is **the first work to practically enable throttling-based RowHammer mitigation**

- BlockHammer is implemented in **the memory controller** (*no proprietary information of / no modifications* to DRAM chips)

- BlockHammer is *both* **scalable with worsening RowHammer** and **compatible with commodity DRAM chips**

- BlockHammer is **open-source** along with **six state-of-the-art mechanisms**: https://github.com/CMU-SAFARI/BlockHammer

Source

A Takeaway

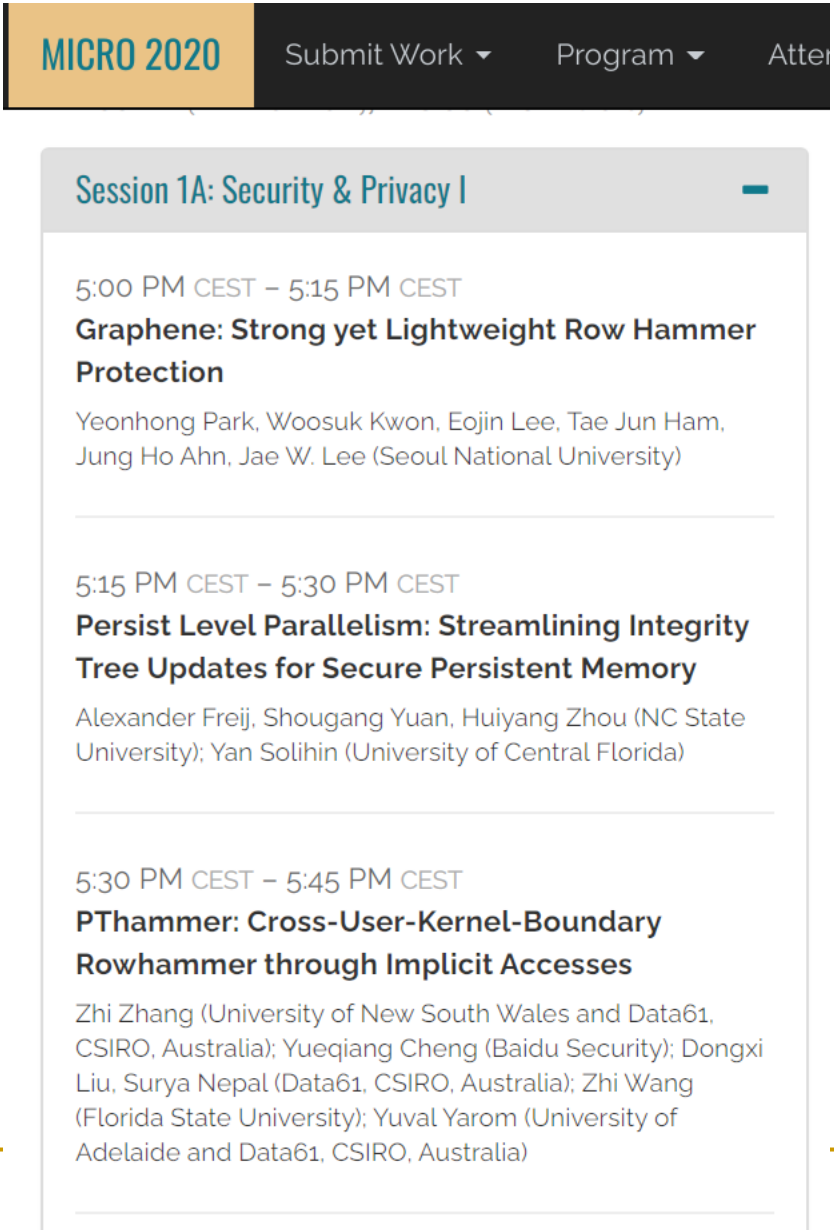
Main Memory Needs

Intelligent Controllers

for Security, Safety, Reliability, Scaling

More RowHammer in 2020-2022

RowHammer in 2020 (I)



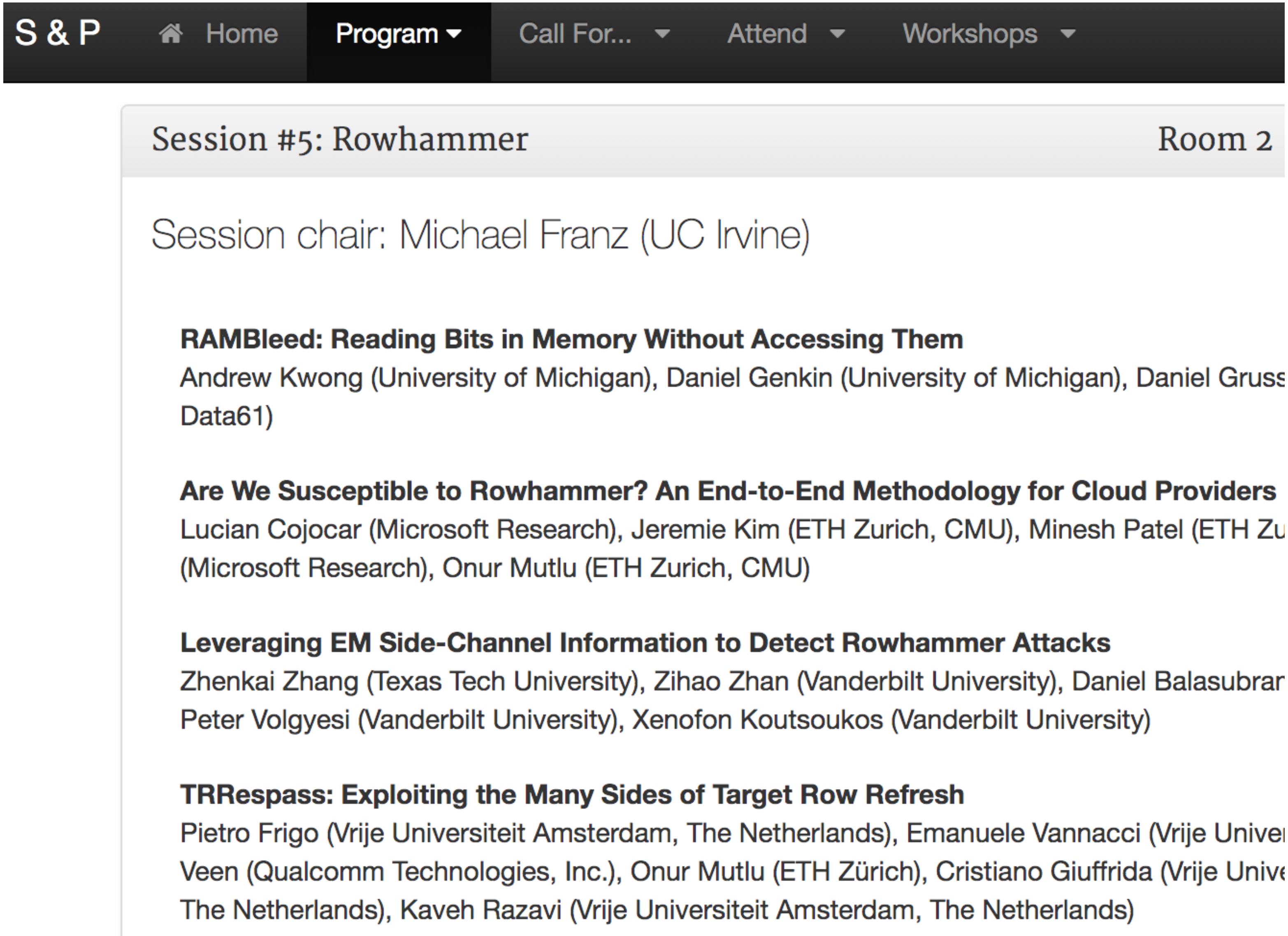
Session 1A: Security & Privacy I

5:00 PM CEST – 5:15 PM CEST
Graphene: Strong yet Lightweight Row Hammer Protection
Yeonhong Park, Woosuk Kwon, Eojin Lee, Tae Jun Ham, Jung Ho Ahn, Jae W. Lee (Seoul National University)

5:15 PM CEST – 5:30 PM CEST
Persist Level Parallelism: Streamlining Integrity Tree Updates for Secure Persistent Memory
Alexander Freij, Shougang Yuan, Huiyang Zhou (NC State University); Yan Solihin (University of Central Florida)

5:30 PM CEST – 5:45 PM CEST
PThammer: Cross-User-Kernel-Boundary Rowhammer through Implicit Accesses
Zhi Zhang (University of New South Wales and Data61, CSIRO, Australia); Yueqiang Cheng (Baidu Security); Dongxi Liu, Surya Nepal (Data61, CSIRO, Australia); Zhi Wang (Florida State University); Yuval Yarom (University of Adelaide and Data61, CSIRO, Australia)

RowHammer in 2020 (II)



Session #5: Rowhammer Room 2

Session chair: Michael Franz (UC Irvine)

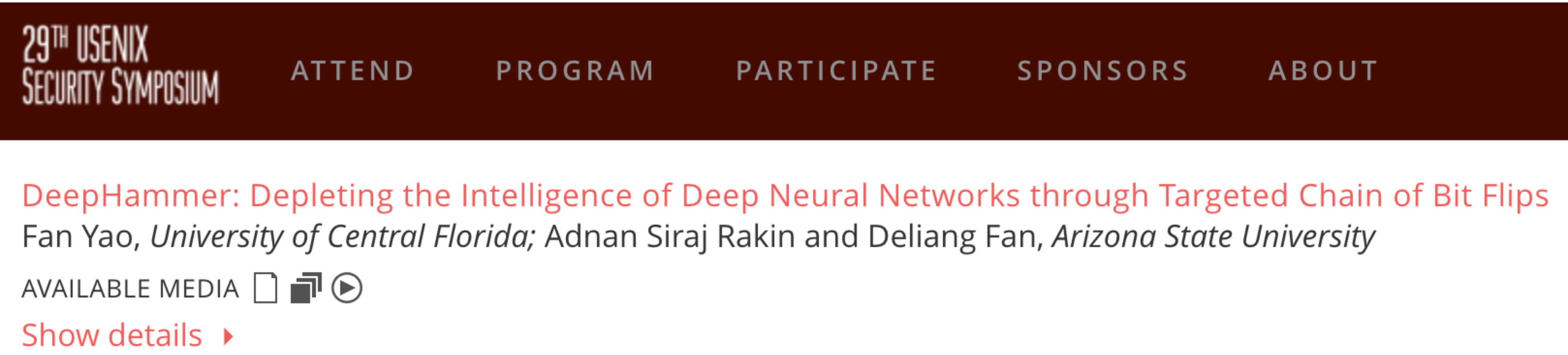
RAMBleed: Reading Bits in Memory Without Accessing Them
Andrew Kwong (University of Michigan), Daniel Genkin (University of Michigan), Daniel Gruss
Data61)

Are We Susceptible to Rowhammer? An End-to-End Methodology for Cloud Providers
Lucian Cojocar (Microsoft Research), Jeremie Kim (ETH Zurich, CMU), Minesh Patel (ETH Zu
(Microsoft Research), Onur Mutlu (ETH Zurich, CMU)

Leveraging EM Side-Channel Information to Detect Rowhammer Attacks
Zhenkai Zhang (Texas Tech University), Zihao Zhan (Vanderbilt University), Daniel Balasubrar
Peter Volgyesi (Vanderbilt University), Xenofon Koutsoukos (Vanderbilt University)

TRRespass: Exploiting the Many Sides of Target Row Refresh
Pietro Frigo (Vrije Universiteit Amsterdam, The Netherlands), Emanuele Vannacci (Vrije Univer
Veen (Qualcomm Technologies, Inc.), Onur Mutlu (ETH Zürich), Cristiano Giuffrida (Vrije Unive
The Netherlands), Kaveh Razavi (Vrije Universiteit Amsterdam, The Netherlands)

RowHammer in 2020 (III)



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

Fan Yao, *University of Central Florida;* Adnan Siraj Rakin and Deliang Fan, *Arizona State University*

RowHammer in 2021 (I)

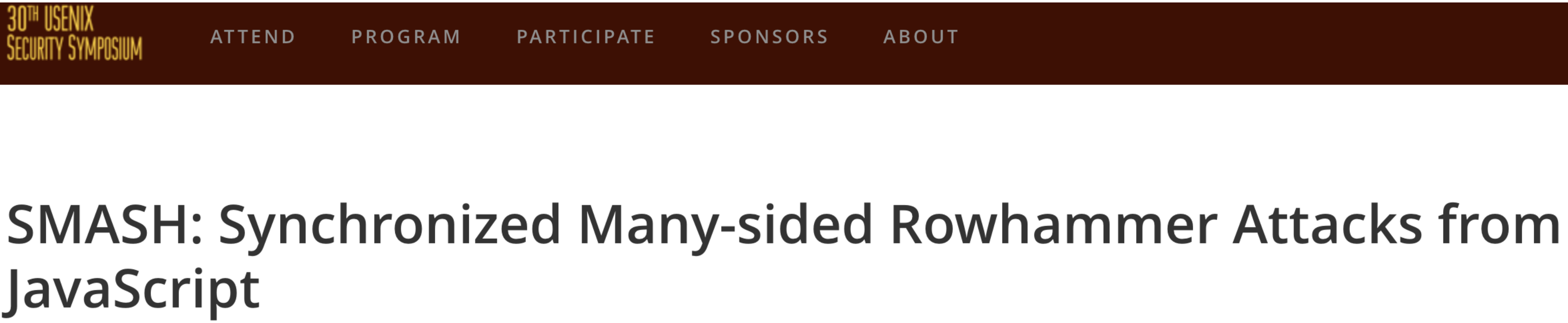
HotOS XVIII

The 18th Workshop on Hot Topics in Operating Systems

~~31 May~~ 1 June–3 June 2021, Cyberspace, People's Couches, and Zoom

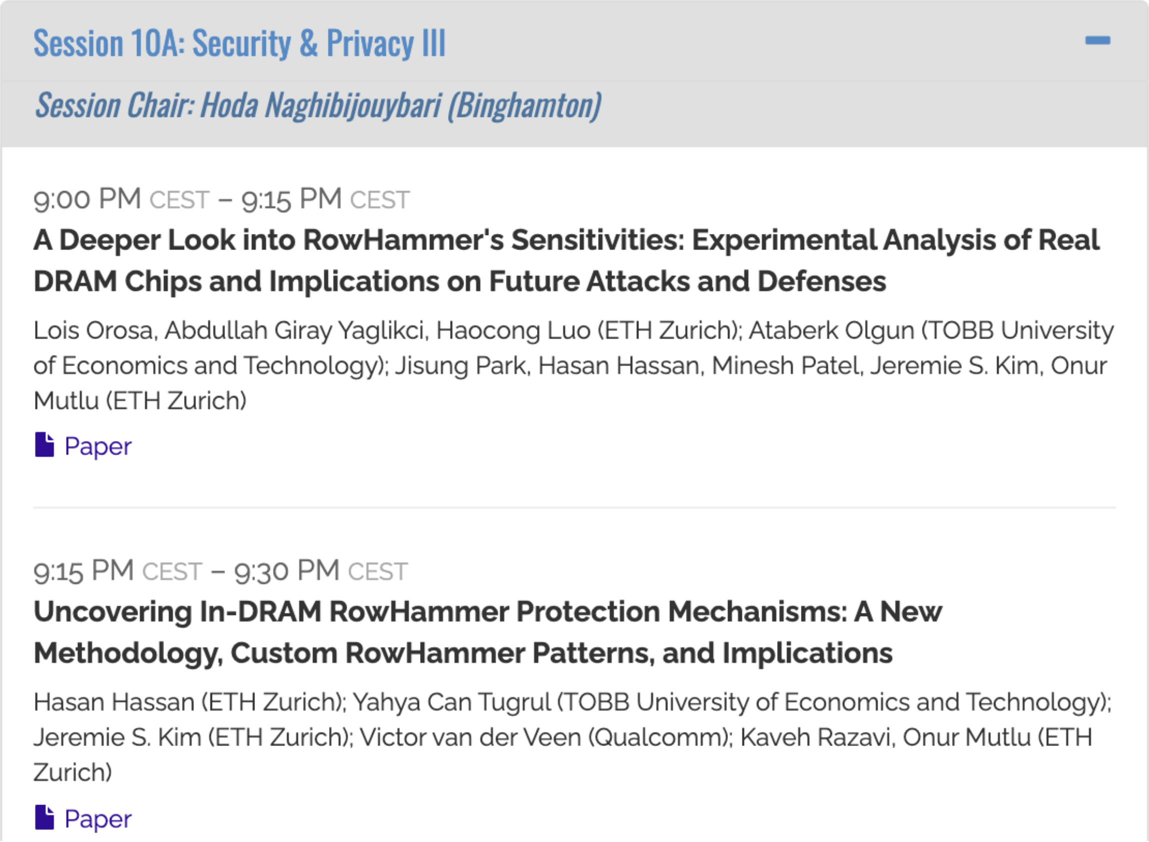
Stop! Hammer Time: Rethinking Our Approach to Rowhammer Mitigations

RowHammer in 2021 (II)



SMASH: Synchronized Many-sided Rowhammer Attacks from JavaScript

RowHammer in 2021 (III)



Session 10A: Security & Privacy III

Session Chair: Hoda Naghibijouybari (Binghamton)

9:00 PM CEST – 9:15 PM CEST

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

Lois Orosa, Abdullah Giray Yaglikci, Haocong Luo (ETH Zurich); Ataberk Olgun (TOBB University of Economics and Technology); Jisung Park, Hasan Hassan, Minesh Patel, Jeremie S. Kim, Onur Mutlu (ETH Zurich)

Paper

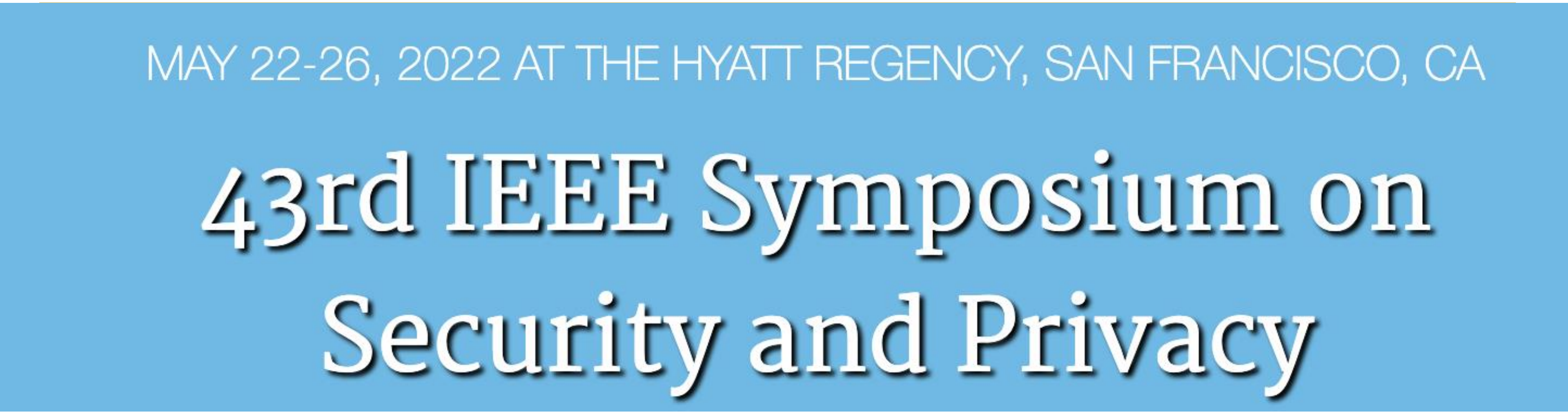
9:15 PM CEST – 9:30 PM CEST

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

Hasan Hassan (ETH Zurich); Yahya Can Tugrul (TOBB University of Economics and Technology); Jeremie S. Kim (ETH Zurich); Victor van der Veen (Qualcomm); Kaveh Razavi, Onur Mutlu (ETH Zurich)

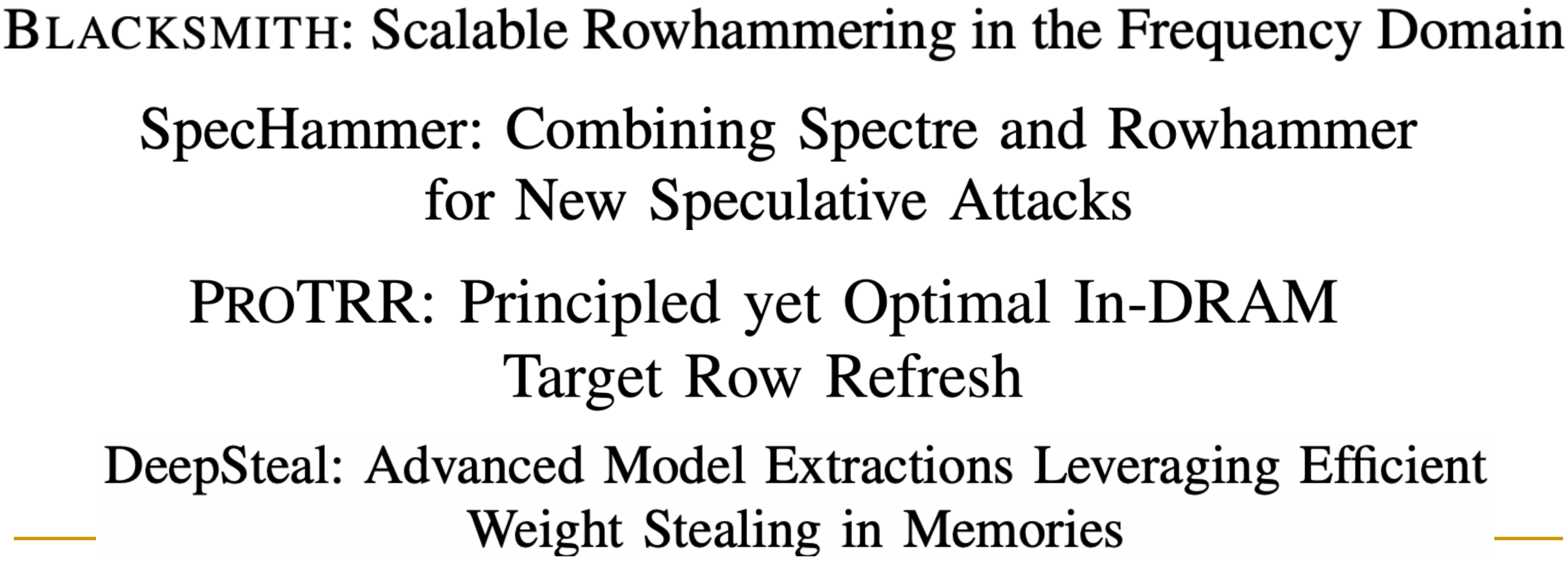
Paper

RowHammer in 2022 (I)



MAY 22-26, 2022 AT THE HYATT REGENCY, SAN FRANCISCO, CA

43rd IEEE Symposium on Security and Privacy



BLACKSMITH: Scalable Rowhammering in the Frequency Domain

SpecHammer: Combining Spectre and Rowhammer for New Speculative Attacks

PROTRR: Principled yet Optimal In-DRAM Target Row Refresh

DeepSteal: Advanced Model Extractions Leveraging Efficient Weight Stealing in Memories

RowHammer in 2022 (II)

Randomized Row-Swap: Mitigating Row Hammer by Breaking Spatial Correlation between Aggressor and Victim Rows

RowHammer in 2022 (III)

HPCA 2022

The 28th IEEE International Symposium on High-Performance Computer Architecture (HPCA-28), Seoul, South Korea

SafeGuard: Reducing the Security Risk from Row-Hammer via Low-Cost Integrity Protection

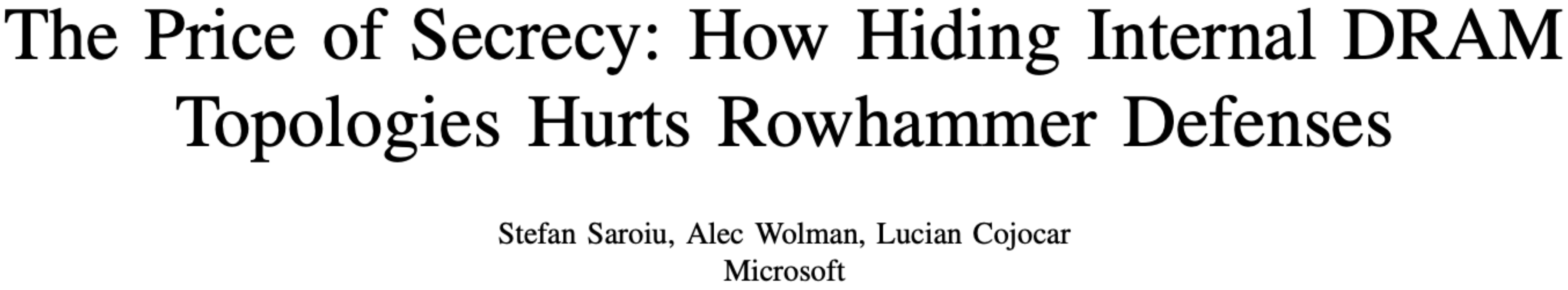
Mithril: Cooperative Row Hammer Protection on Commodity DRAM Leveraging Managed Refresh

RowHammer in 2022 (IV)

IRPS

International Reliability Physics Symposium

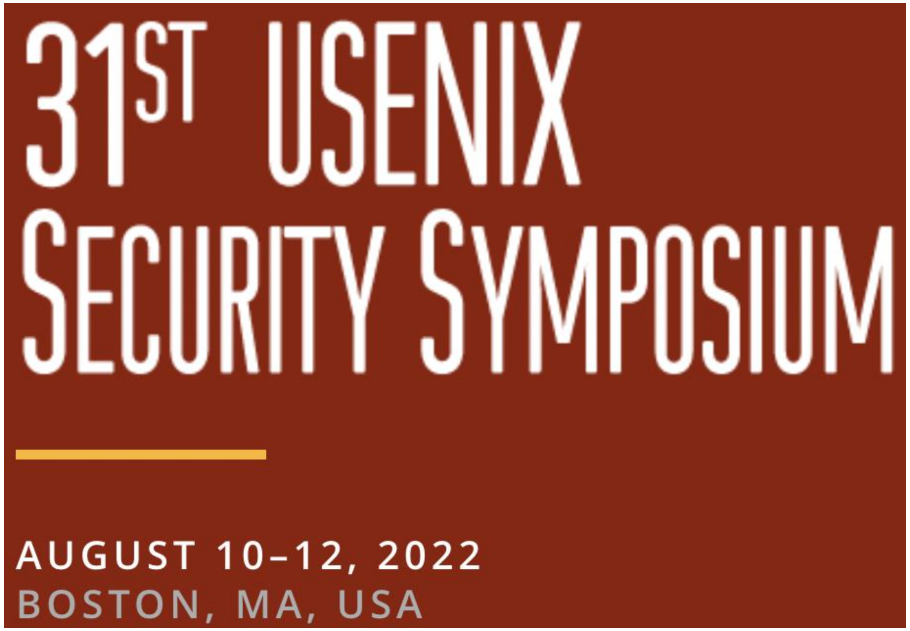
IRPS 2022



The Price of Secrecy: How Hiding Internal DRAM Topologies Hurts Rowhammer Defenses

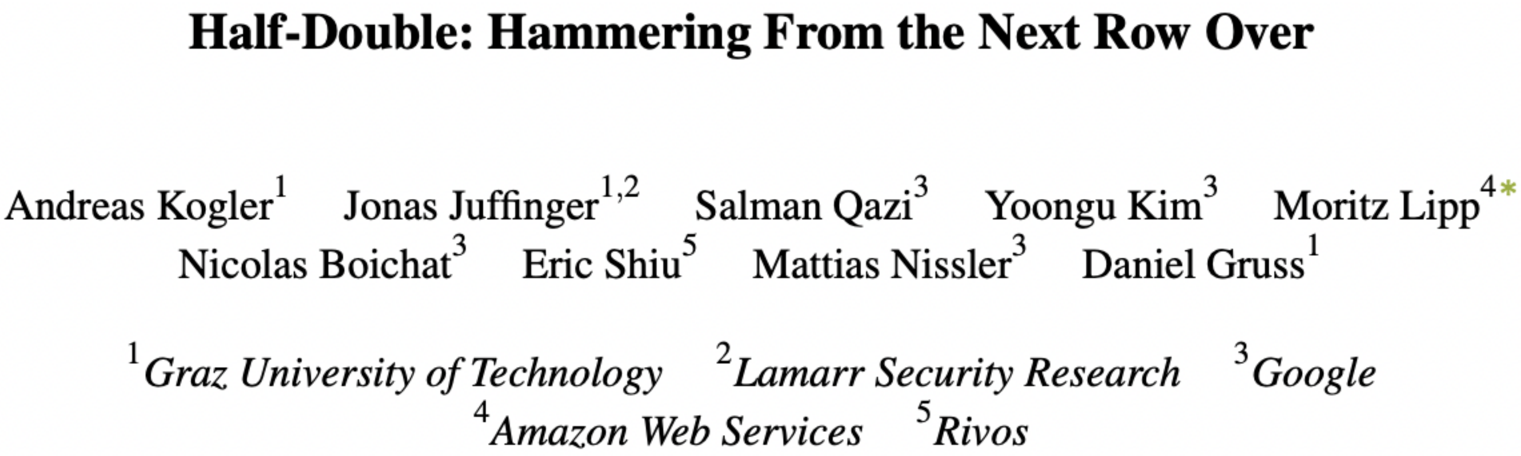
Stefan Saroiu, Alec Wolman, Lucian Cojocar
Microsoft

RowHammer in 2022 (V)



31ST USENIX SECURITY SYMPOSIUM

AUGUST 10–12, 2022
BOSTON, MA, USA



Half-Double: Hammering From the Next Row Over

Andreas Kogler[1] Jonas Juffinger[1,2] Salman Qazi[3] Yoongu Kim[3] Moritz Lipp[4*]
Nicolas Boichat[3] Eric Shiu[5] Mattias Nissler[3] Daniel Gruss[1]

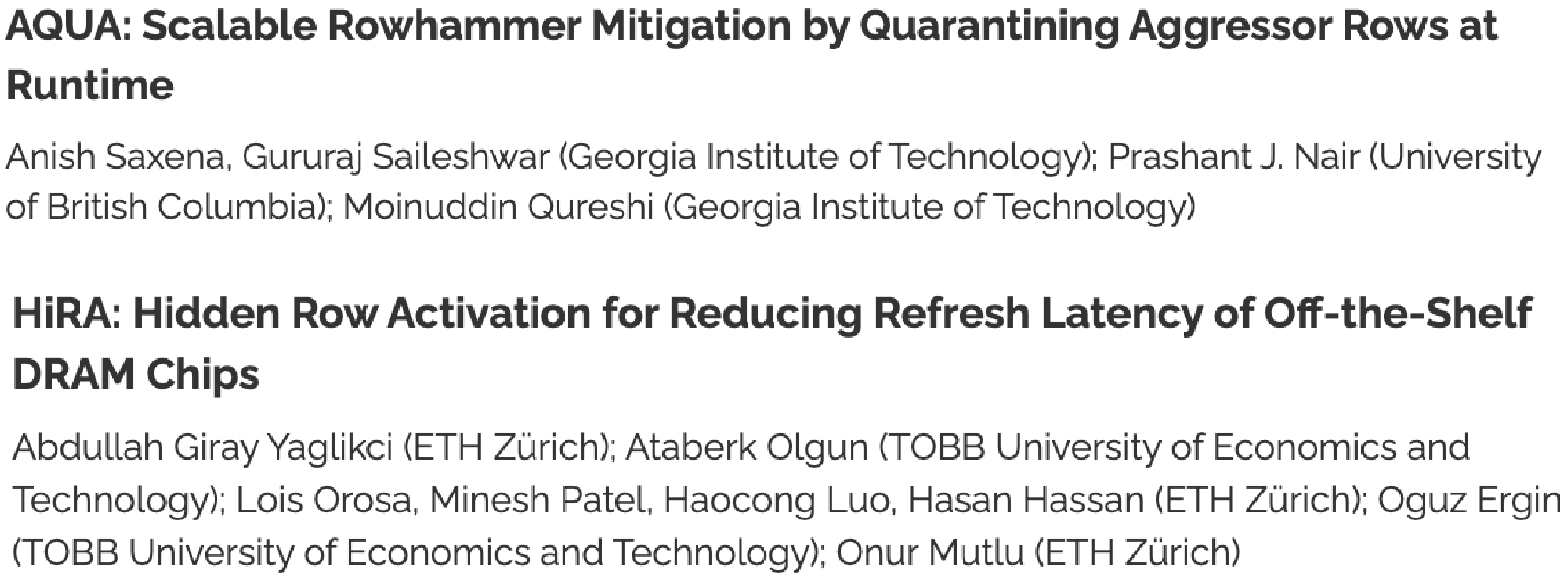
[1] *Graz University of Technology* [2] *Lamarr Security Research* [3] *Google*
[4] *Amazon Web Services* [5] *Rivos*

RowHammer in 2022 (VI)

HammerScope: Observing DRAM Power Consumption Using Rowhammer

**When Frodo Flips:
End-to-End Key Recovery on FrodoKEM via Rowhammer**

RowHammer in 2022 (VII)



AQUA: Scalable Rowhammer Mitigation by Quarantining Aggressor Rows at Runtime

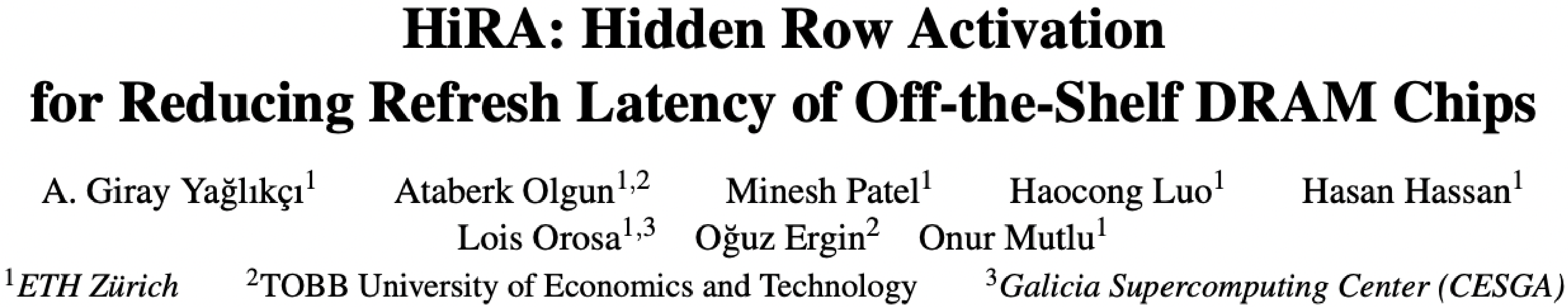
Anish Saxena, Gururaj Saileshwar (Georgia Institute of Technology); Prashant J. Nair (University of British Columbia); Moinuddin Qureshi (Georgia Institute of Technology)

HiRA: Hidden Row Activation for Reducing Refresh Latency of Off-the-Shelf DRAM Chips

Abdullah Giray Yaglikci (ETH Zürich); Ataberk Olgun (TOBB University of Economics and Technology); Lois Orosa, Minesh Patel, Haocong Luo, Hasan Hassan (ETH Zürich); Oguz Ergin (TOBB University of Economics and Technology); Onur Mutlu (ETH Zürich)

RowHammer in 2022 (VII)

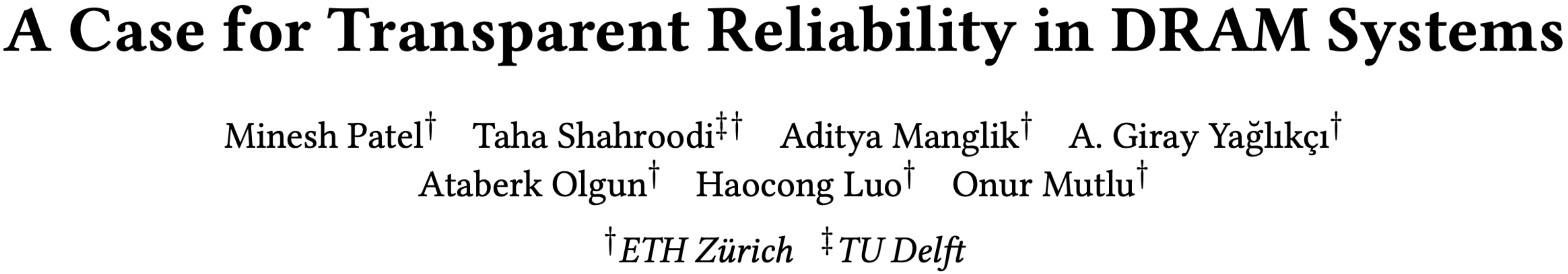
- **To appear at MICRO 2022**



HiRA: Hidden Row Activation for Reducing Refresh Latency of Off-the-Shelf DRAM Chips

A. Giray Yağlıkçı[1] Ataberk Olgun[1,2] Minesh Patel[1] Haocong Luo[1] Hasan Hassan[1]
Lois Orosa[1,3] Oğuz Ergin[2] Onur Mutlu[1]
[1] *ETH Zürich* [2] TOBB University of Economics and Technology [3] *Galicia Supercomputing Center (CESGA)*

RowHammer in 2022 (VIII)



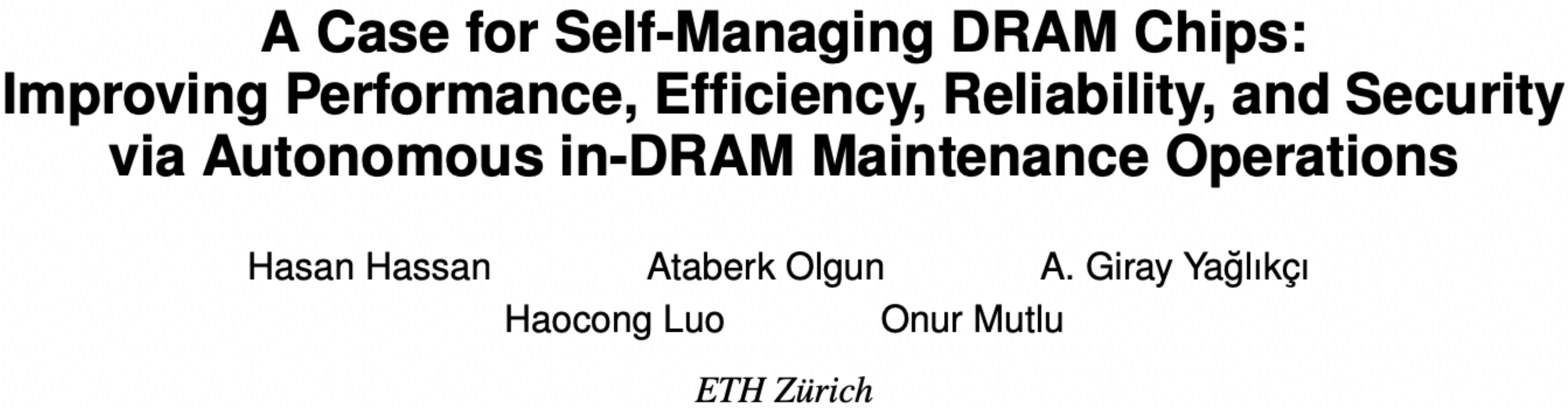
A Case for Transparent Reliability in DRAM Systems

Minesh Patel[†] Taha Shahroodi[‡†] Aditya Manglik[†] A. Giray Yağlıkçı[†]
Ataberk Olgun[†] Haocong Luo[†] Onur Mutlu[†]

[†]*ETH Zürich* [‡]*TU Delft*

https://arxiv.org/pdf/2204.10378.pdf

RowHammer in 2022 (IX)



A Case for Self-Managing DRAM Chips:
Improving Performance, Efficiency, Reliability, and Security
via Autonomous in-DRAM Maintenance Operations

Hasan Hassan Ataberk Olgun A. Giray Yağlıkçı
Haocong Luo Onur Mutlu

ETH Zürich

https://arxiv.org/pdf/2207.13358.pdf

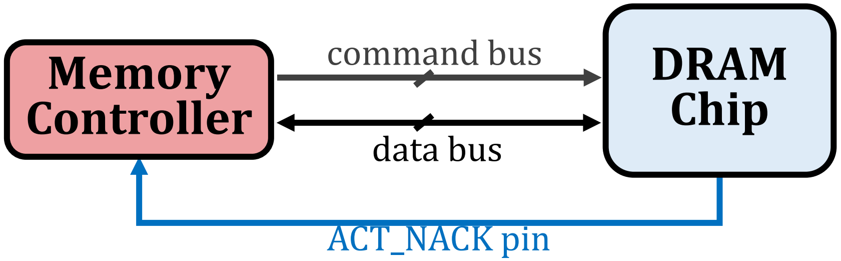
Self-Managing DRAM: Overview

Self-Managing DRAM (SMD)

enables autonomous in-DRAM maintenance operations

Key Idea:

Prevent the memory controller from accessing DRAM regions that are *under maintenance* by rejecting row activation (ACT) commands



Leveraging the ability to *reject an ACT*, a maintenance operation can be implemented *completely* within a 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*	

Self-Managing DRAM: Summary

The three major DRAM maintenance operations:

- ❖Refresh
- ❖RowHammer Protection
- ❖Memory Scrubbing

Implementing new **maintenance mechanisms** often requires difficult-to-realize changes

Our Goal

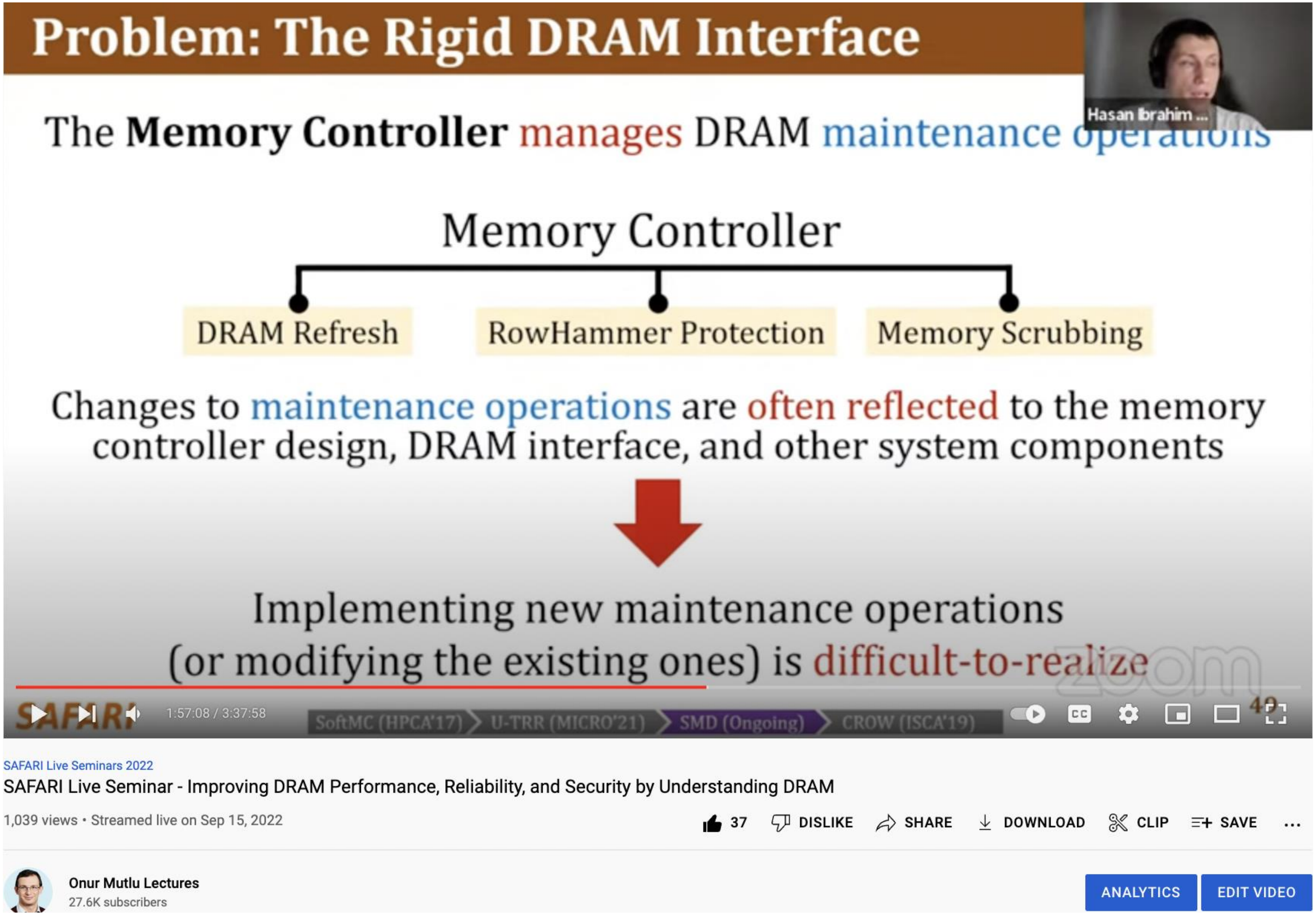
1. Ease the process of enabling new DRAM maintenance operations
2. Enable more efficient in-DRAM maintenance operations

Self-Managing DRAM (SMD)

Enables implementing new **in-DRAM** maintenance mechanisms with **no further changes** in the *DRAM interface* and *memory controller*

SMD-based *refresh*, *RowHammer protection*, and *scrubbing* achieve **9.2% speedup** and **6.2% lower DRAM energy** vs. conventional DRAM

Talk on Self-Managing DRAM



Problem: The Rigid DRAM Interface

The **Memory Controller** manages DRAM maintenance operations

Memory Controller

DRAM Refresh | RowHammer Protection | Memory Scrubbing

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

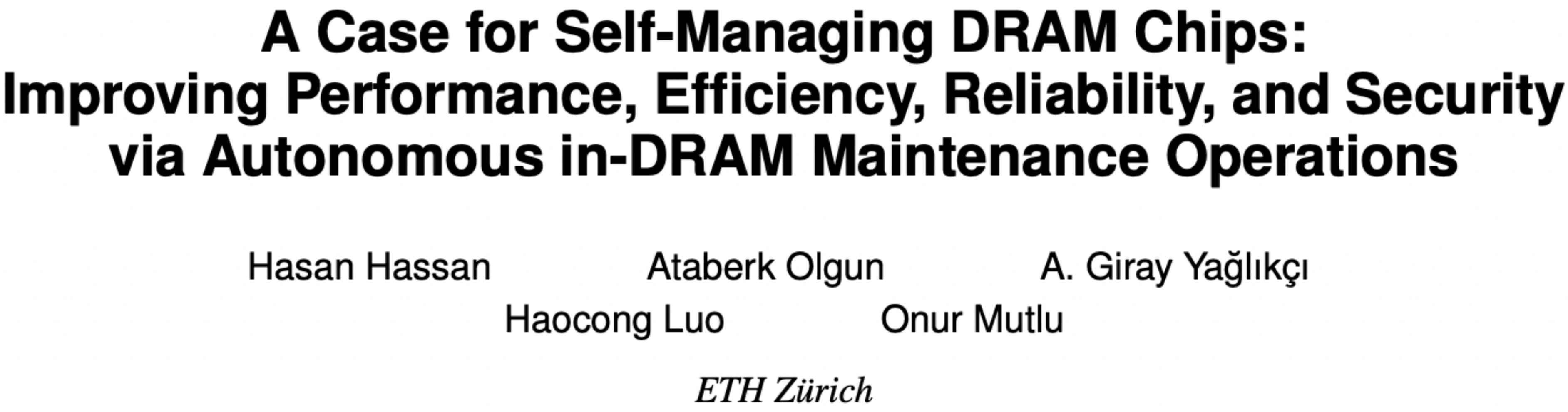
1,039 views • Streamed live on Sep 15, 2022

Onur Mutlu Lectures

27.6K subscribers

https://www.youtube.com/watch?v=mGa6-vpExbE

Much More in Our Preprint…



A Case for Self-Managing DRAM Chips: Improving Performance, Efficiency, Reliability, and Security via Autonomous in-DRAM Maintenance Operations

Hasan Hassan Ataberk Olgun A. Giray Yağlıkçı
Haocong Luo Onur Mutlu

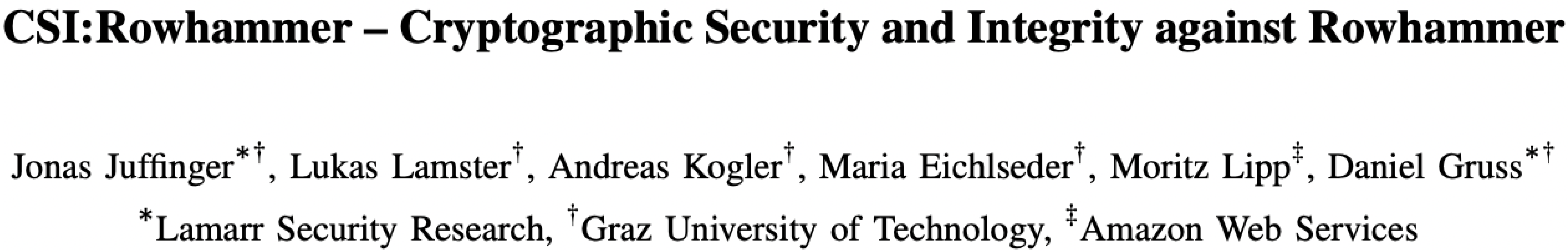
ETH Zürich

https://arxiv.org/pdf/2207.13358.pdf

RowHammer in 2023

MAY 22-26, 2023 AT THE HYATT REGENCY, SAN FRANCISCO, CA

44th IEEE Symposium on Security and Privacy



CSI:Rowhammer – Cryptographic Security and Integrity against Rowhammer

Jonas Juffinger[*†], Lukas Lamster[†], Andreas Kogler[†], Maria Eichlseder[†], Moritz Lipp[‡], Daniel Gruss[*†]
[*]Lamarr Security Research, [†]Graz University of Technology, [‡]Amazon Web Services

More to Come…

Future Memory Reliability/Security Challenges

Future of Main Memory Security/Reliability

- DRAM is becoming less reliable → more vulnerable

- Due to difficulties in DRAM scaling, other problems may also appear (or they may be going unnoticed)

- Some errors may already be slipping into the field
  - Read disturb errors (Rowhammer)
  - Retention errors
  - Read errors, write errors
  - …

- These errors can also pose security vulnerabilities

Future of Main Memory Security/Reliability

- DRAM

- Flash memory

- Emerging Technologies
  - Phase Change Memory
  - STT-MRAM
  - RRAM, memristors
  - …

Many Errors and Their Mitigation [PIEEE'17]

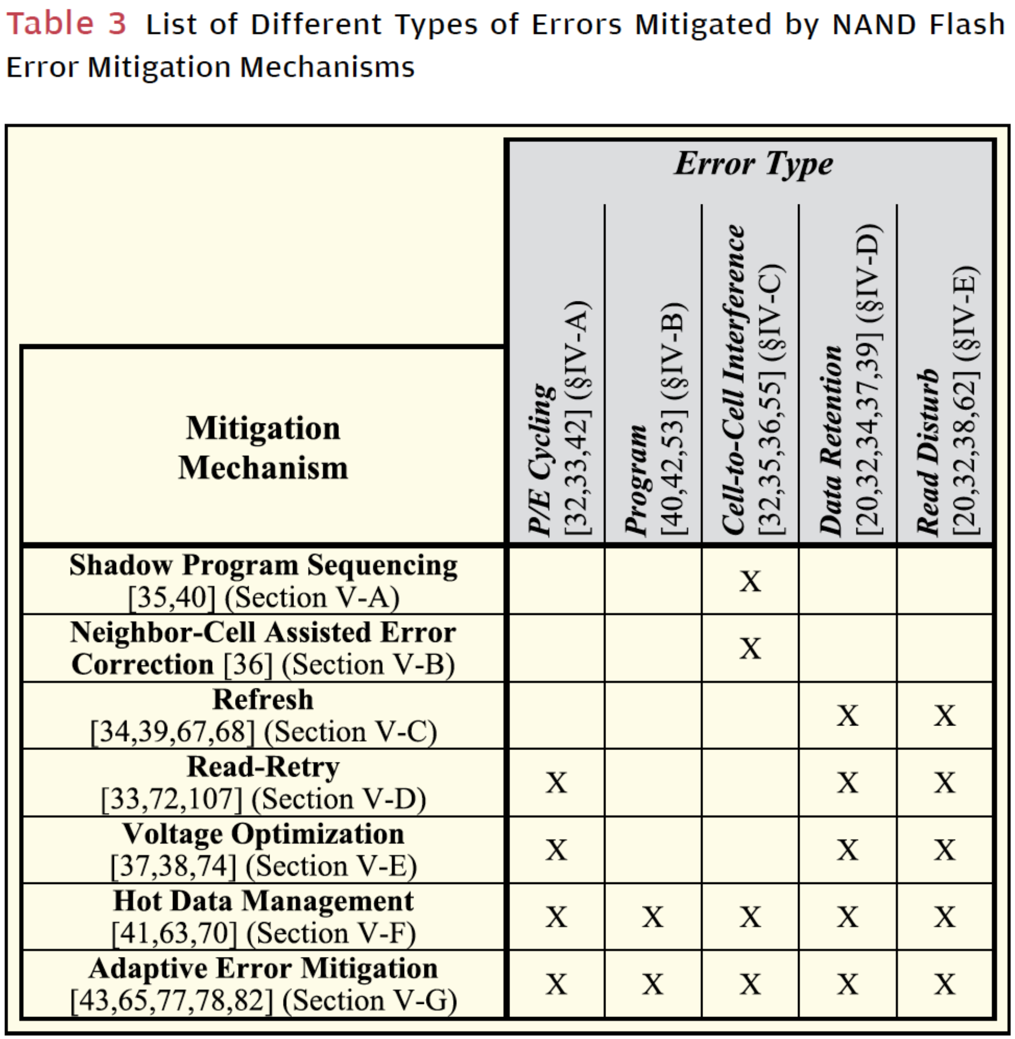


Table 3 List of Different Types of Errors Mitigated by NAND Flash Error Mitigation Mechanisms

Mitigation Mechanism	***Error Type***				
	P/E Cycling [32,33,42] (§IV-A)	***Program*** [40,42,53] (§IV-B)	***Cell-to-Cell Interference*** [32,35,36,55] (§IV-C)	***Data Retention*** [20,32,34,37,39] (§IV-D)	***Read Disturb*** [20,32,38,62] (§IV-E)
Shadow Program Sequencing [35,40] (Section V-A)			X		
Neighbor-Cell Assisted Error Correction [36] (Section V-B)			X		
Refresh [34,39,67,68] (Section V-C)				X	X
Read-Retry [33,72,107] (Section V-D)	X			X	X
Voltage Optimization [37,38,74] (Section V-E)	X			X	X
Hot Data Management [41,63,70] (Section V-F)	X	X	X	X	X
Adaptive Error Mitigation [43,65,77,78,82] (Section V-G)	X	X	X	X	X

Cai+, "Error Characterization, Mitigation, and Recovery in Flash Memory Based Solid State Drives," Proc. IEEE 2017.

A Takeaway

Main Memory Needs

Intelligent Controllers

for Security, Safety, Reliability, Scaling

The Takeaway

Intelligent

Memory Controllers

Can Avoid Many Failures

& Enable Better Scaling

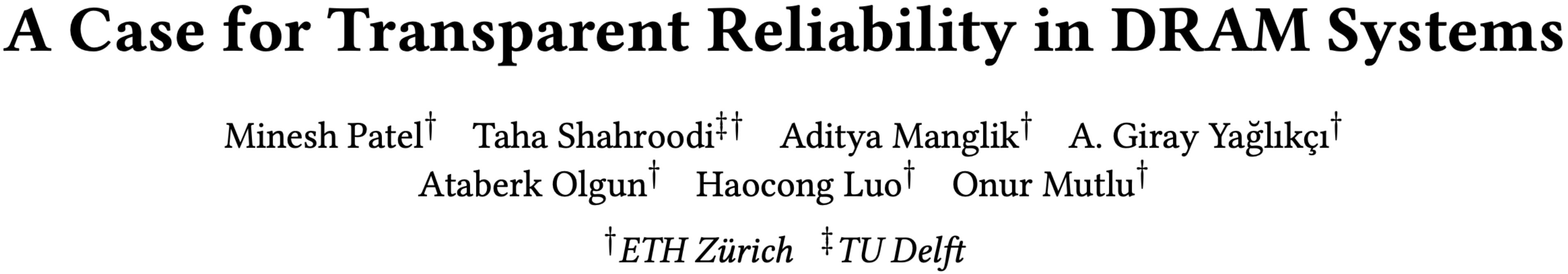
Architecting Future Memory for Security

- **Understand**: Methods for vulnerability modeling & discovery
  - Modeling and prediction based on real (device) data and analysis
  - Understanding vulnerabilities
  - Developing reliable metrics

- **Architect**: Principled architectures with security as key concern
  - Good partitioning of duties across the stack
  - Cannot give up performance and efficiency
  - Patch-ability in the field

- **Design & Test**: Principled design, automation, (online) testing
  - Design for security
  - High coverage and good interaction with system reliability methods

Better Communication Between DRAM & Controller



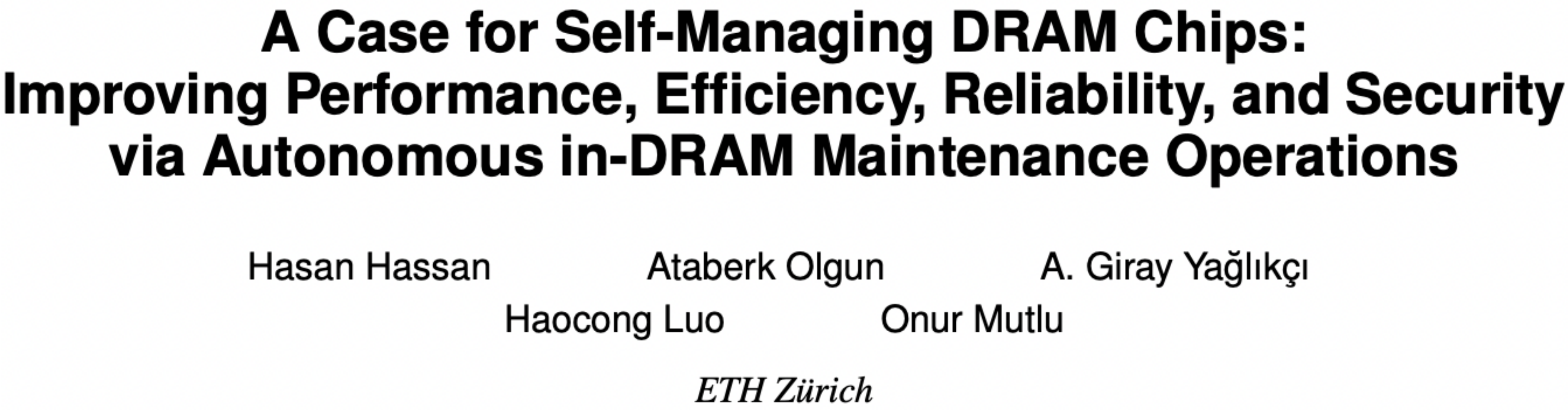
A Case for Transparent Reliability in DRAM Systems

Minesh Patel[†] Taha Shahroodi[‡†] Aditya Manglik[†] A. Giray Yağlıkçı[†]
Ataberk Olgun[†] Haocong Luo[†] Onur Mutlu[†]

[†]*ETH Zürich* [‡]*TU Delft*

https://arxiv.org/pdf/2204.10378.pdf

Better Coordination of DRAM & Controller



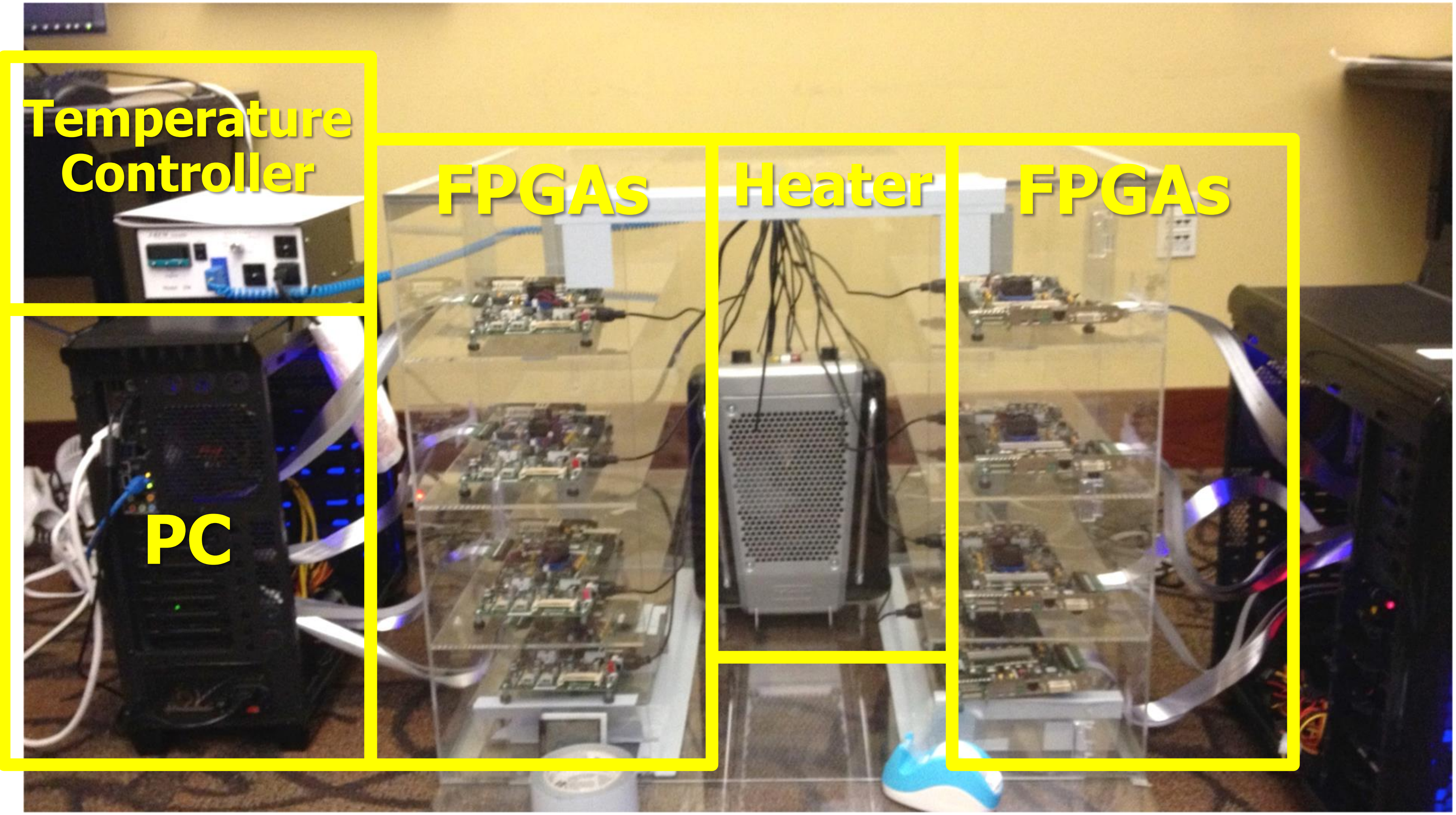
A Case for Self-Managing DRAM Chips: Improving Performance, Efficiency, Reliability, and Security via Autonomous in-DRAM Maintenance Operations

Hasan Hassan Ataberk Olgun A. Giray Yağlıkçı
Haocong Luo Onur Mutlu

ETH Zürich

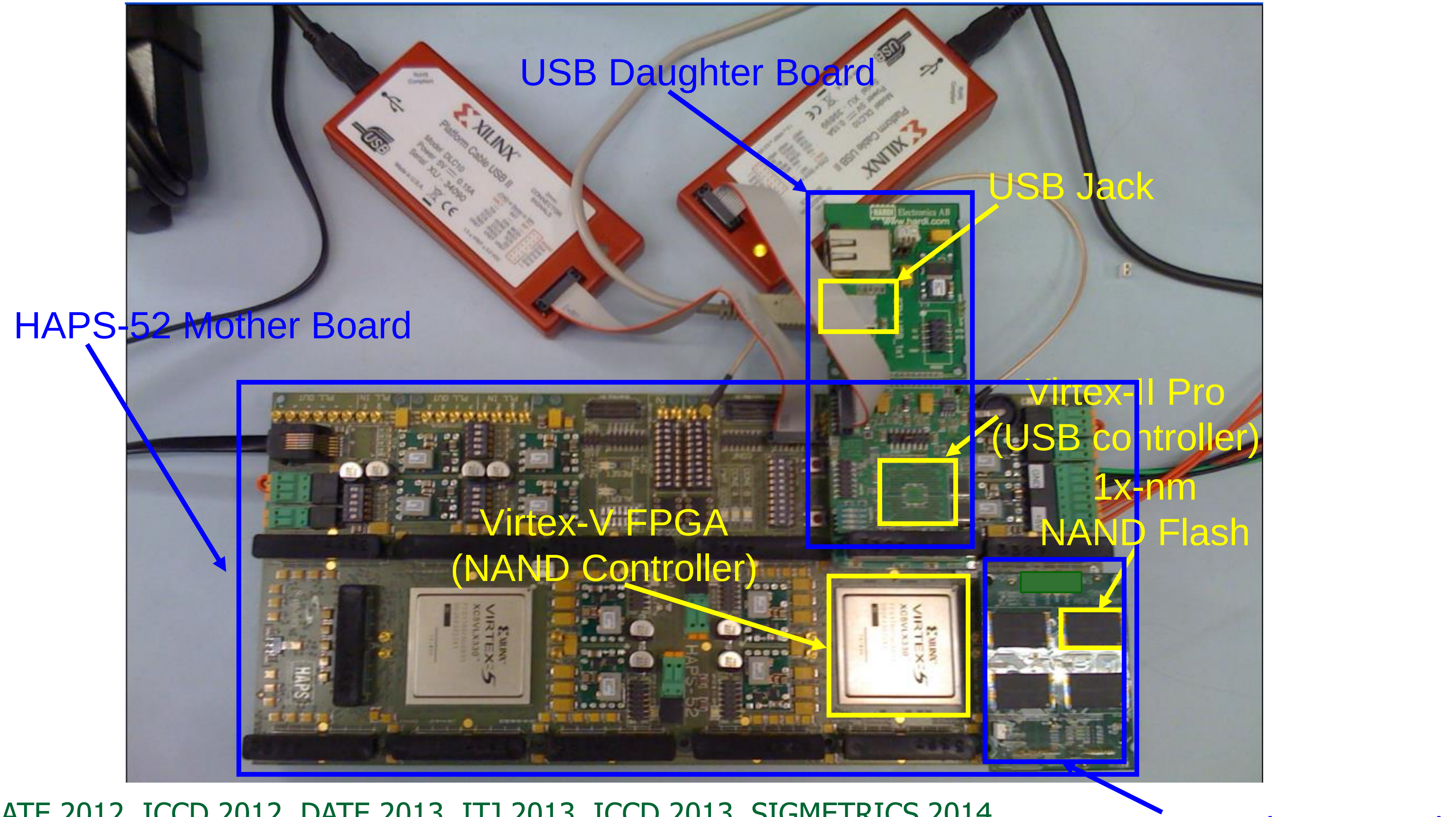
https://arxiv.org/pdf/2207.13358.pdf

Understand and Model with Experiments (DRAM)



Kim+, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," ISCA 2014.

Understand and Model with Experiments (Flash)

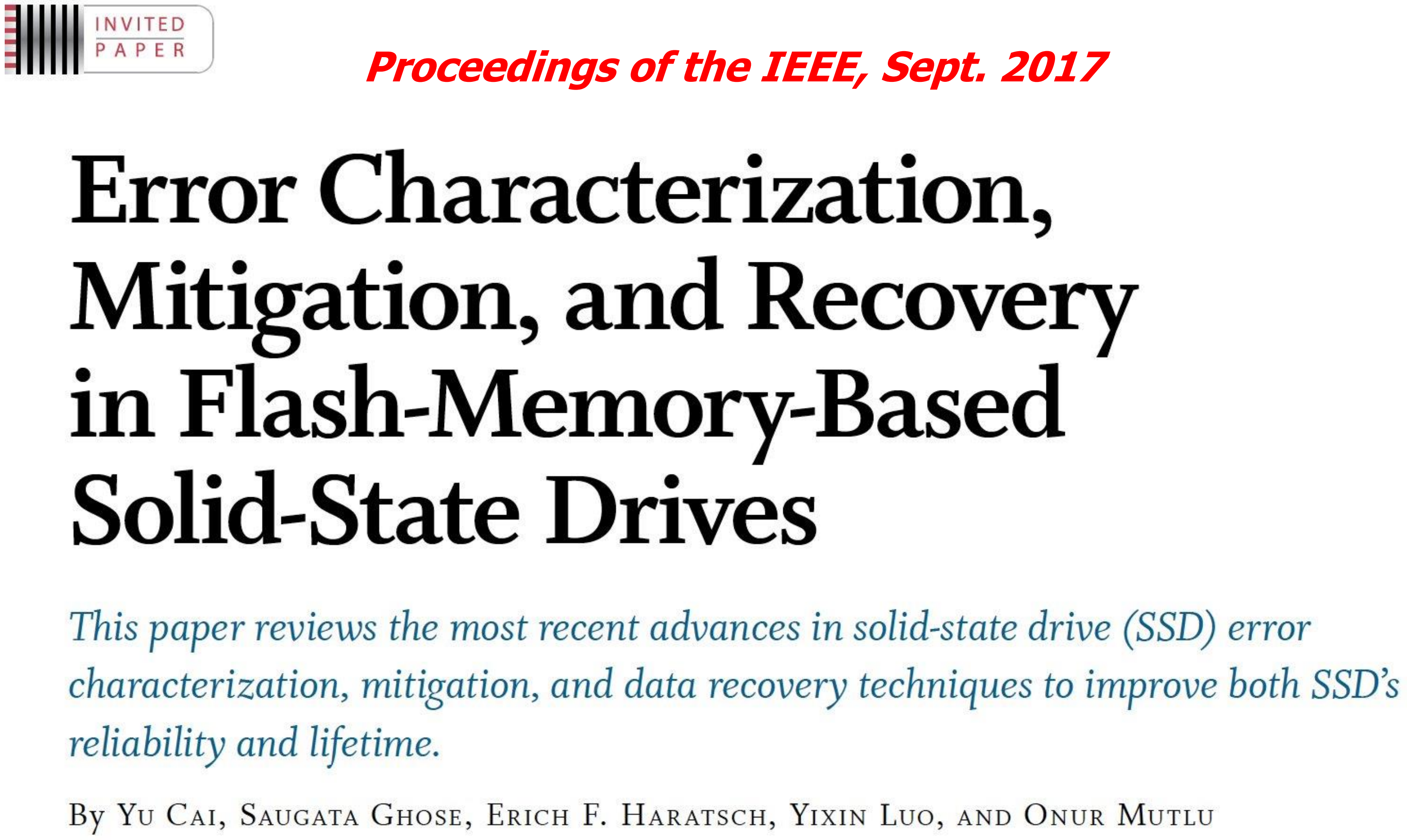


[DATE 2012, ICCD 2012, DATE 2013, ITJ 2013, ICCD 2013, SIGMETRICS 2014, HPCA 2015, DSN 2015, MSST 2015, JSAC 2016, HPCA 2017, DFRWS 2017, PIEEE 2017, HPCA 2018, SIGMETRICS 2018]

Cai+, "Error Characterization, Mitigation, and Recovery in Flash Memory Based Solid State Drives," Proc. IEEE 2017.

An Example Intelligent Controller

Proceedings of the IEEE, Sept. 2017



INVITED PAPER

Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives

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

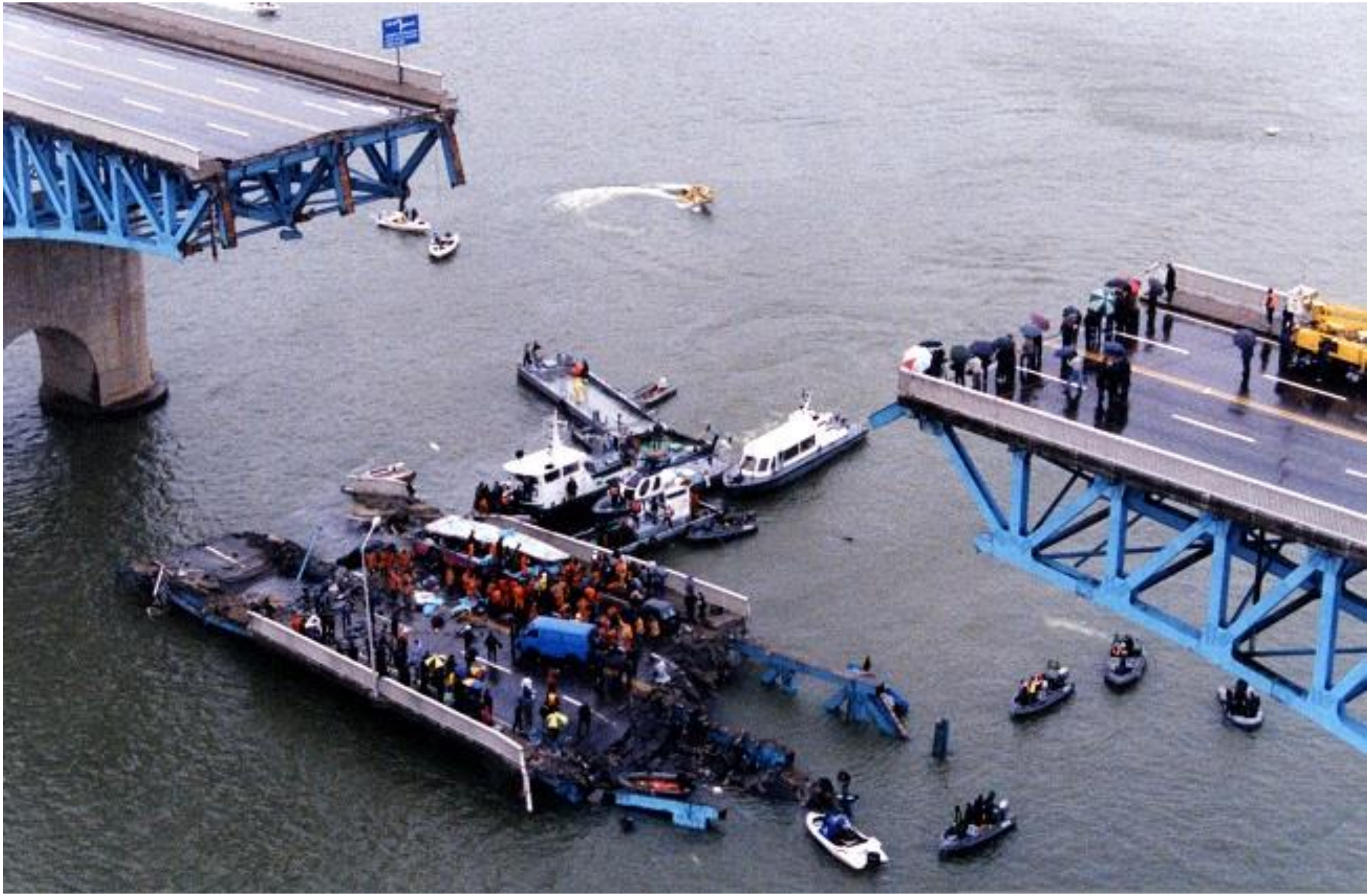
By Yu Cai, Saugata Ghose, Erich F. Haratsch, Yixin Luo, and Onur Mutlu

https://arxiv.org/pdf/1706.08642

Collapse of the "Galloping Gertie" (1940)

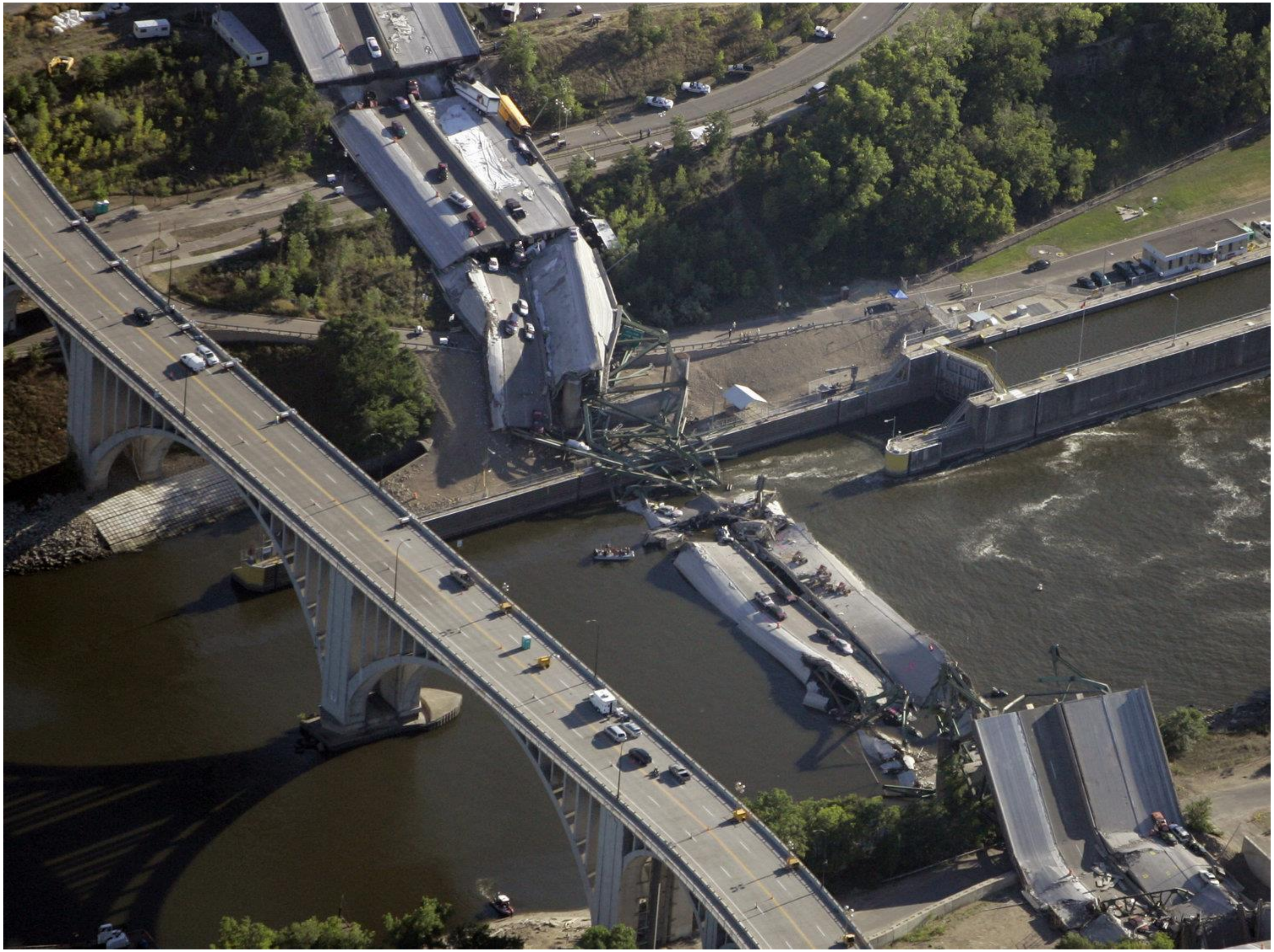
Source: AP
http://www.wsdot.wa.gov/tnbhistory/connections/connections3.htm

Another Example (1994)



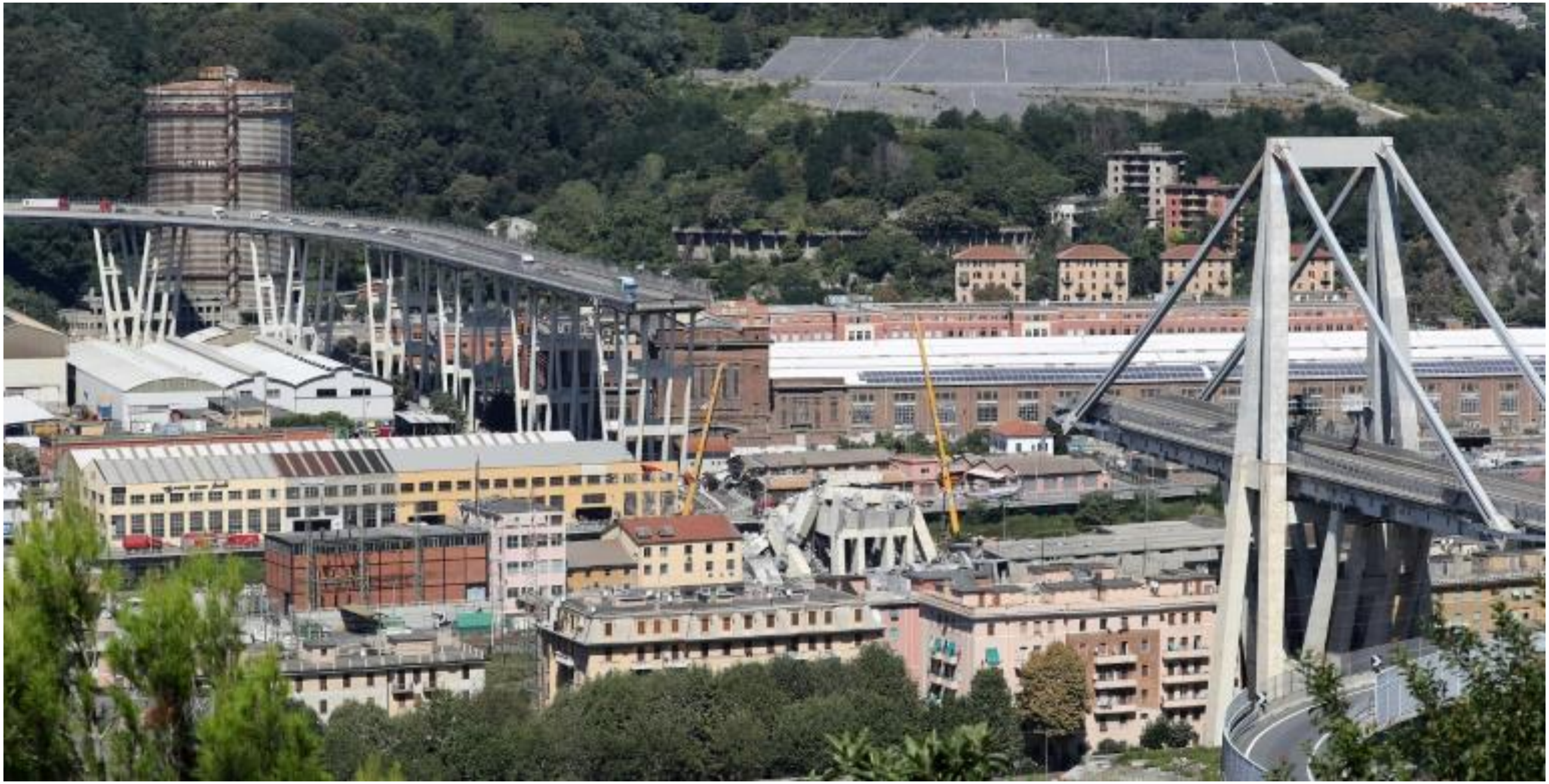
Source: By 최광모 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=35197984

Yet Another Example (2007)



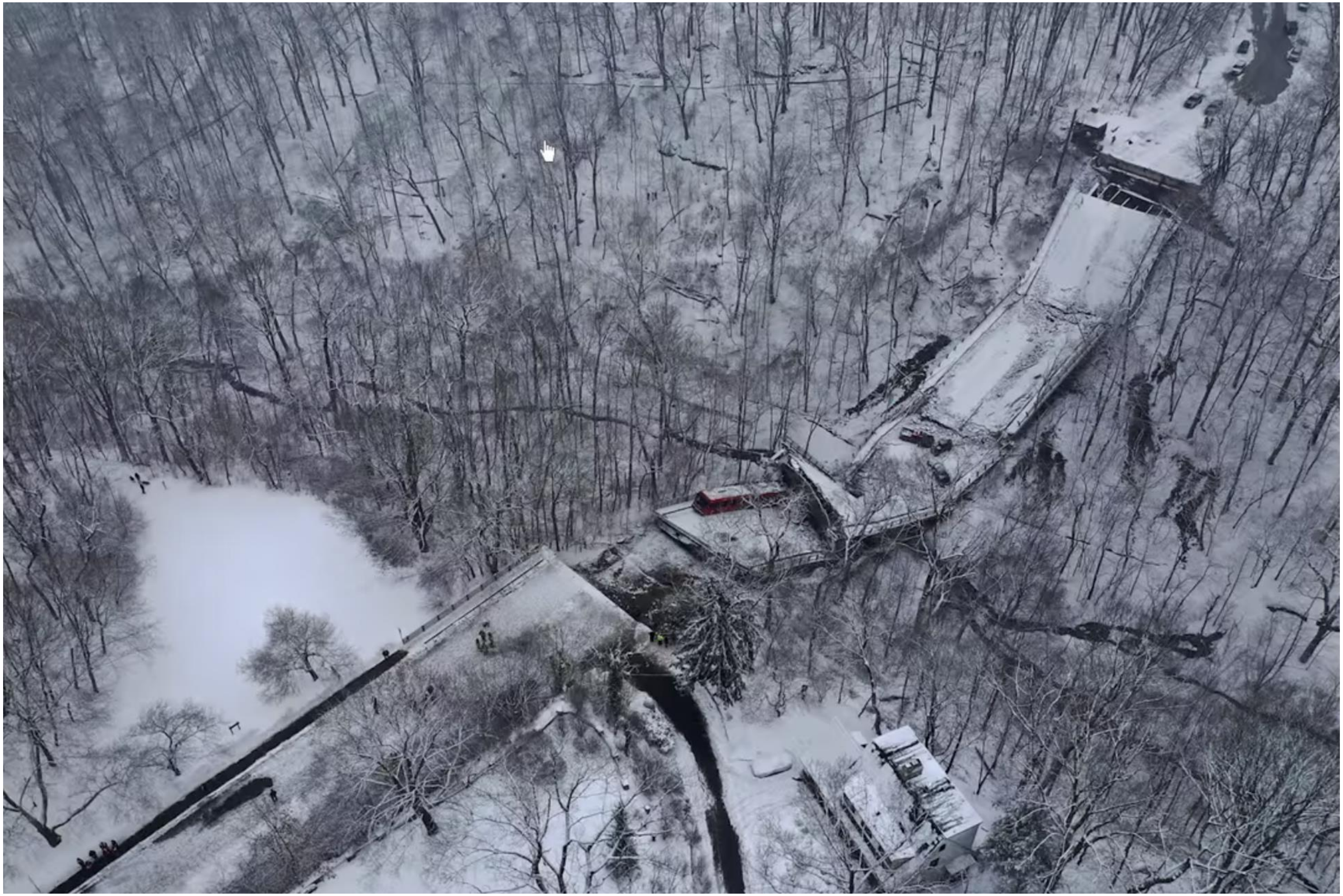
Source: Morry Gash/AP,
https://www.npr.org/2017/08/01/540669701/10-years-after-bridge-collapse-america-is-still-crumbling?t=1535427165809

A More Recent Example (2018)



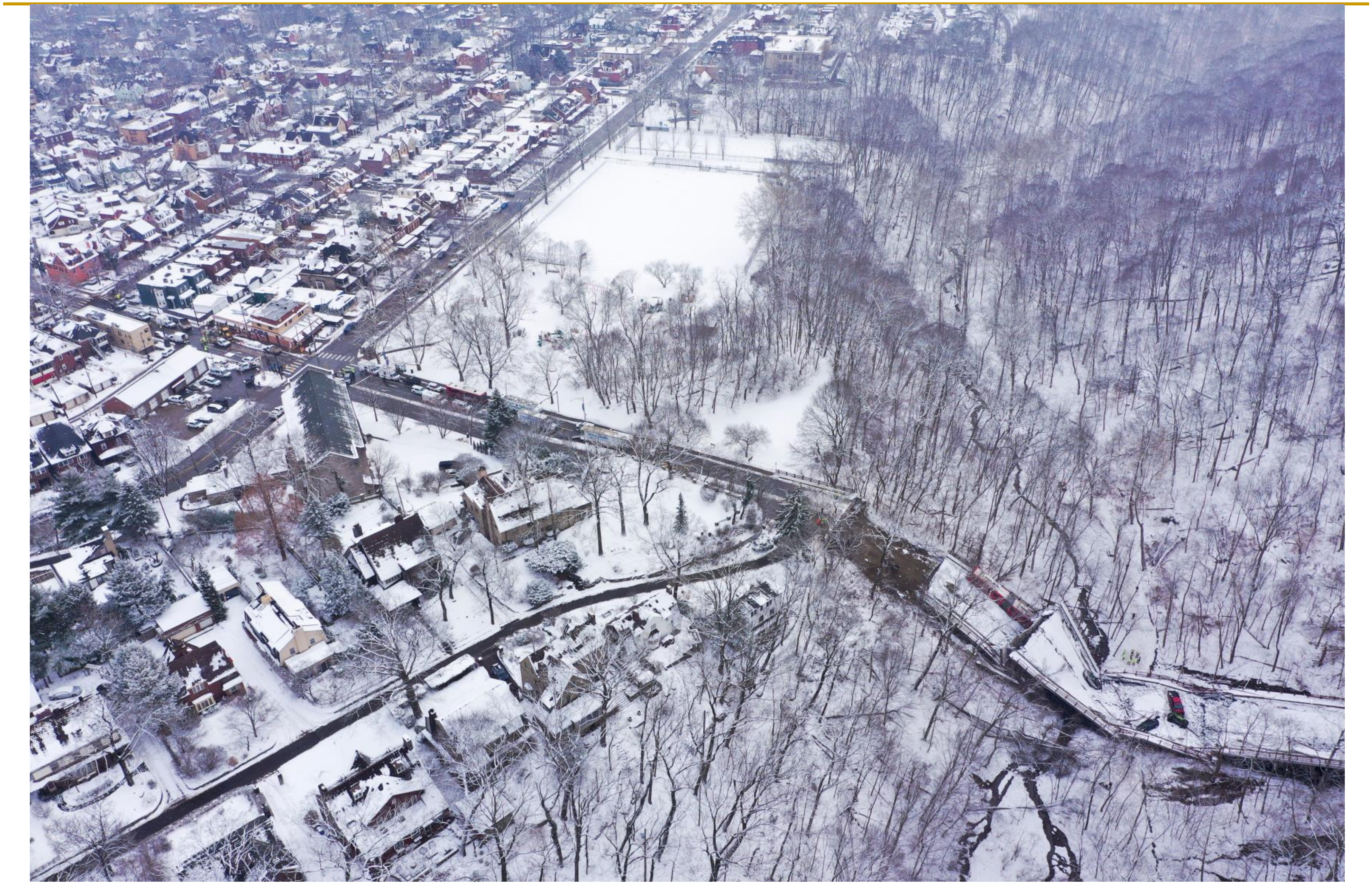
Source: AFP / Valery HACHE, https://www.capitalfm.co.ke/news/2018/08/genoa-bridge-collapse-what-we-know/

A Most Recent Example (2022)



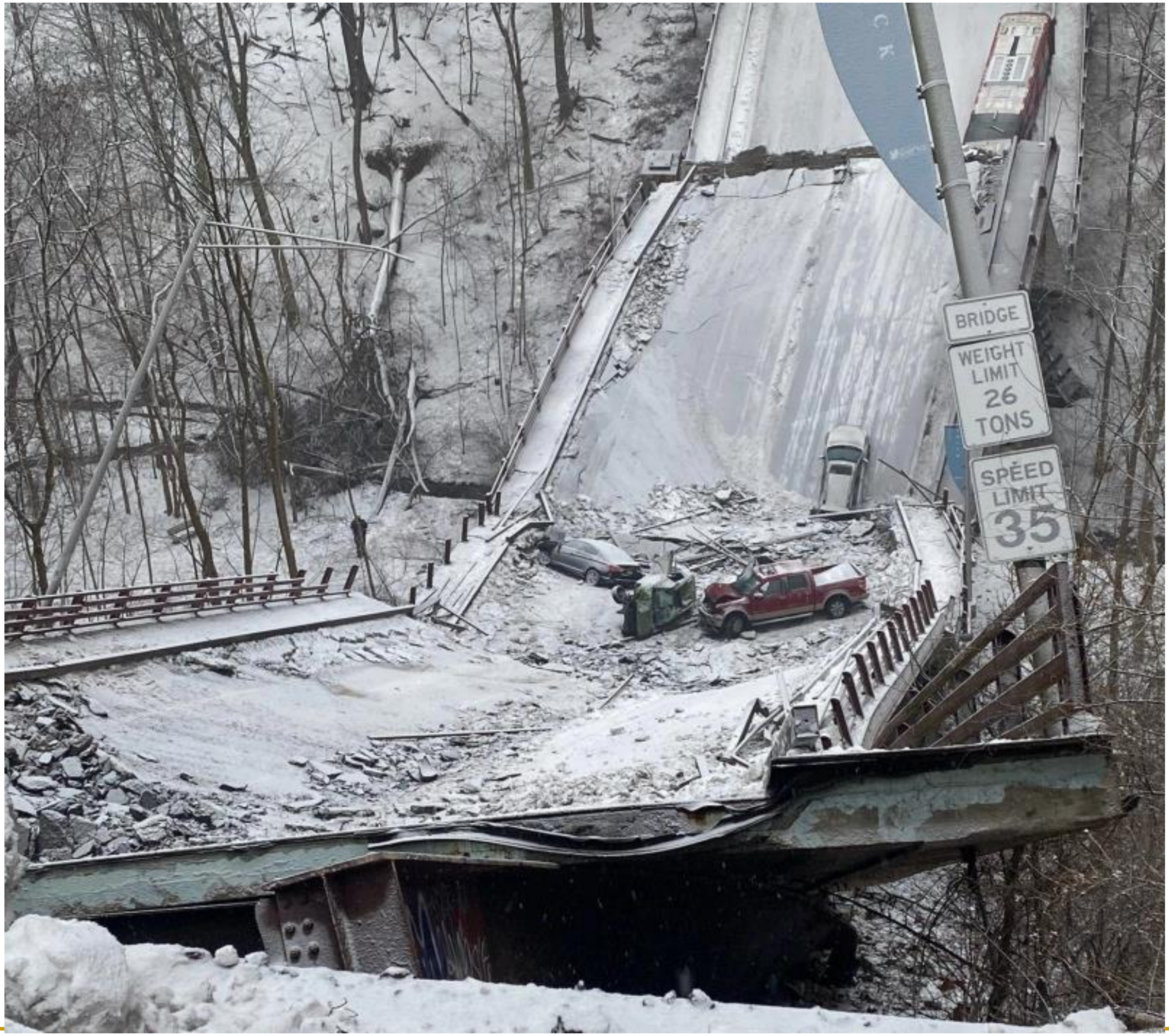
https://www.post-gazette.com/local/city/2022/01/28/pittsburgh-bridge-collapse-forbes-braddock-avenue-point-breeze-squirrel-hill/stories/202201280075

A Most Recent Example (2022)



https://www.reuters.com/world/us/biden-highlight-manufacturing-jobs-gdp-growth-pittsburgh-2022-01-28/

A Most Recent Example (2022)



https://usa.streetsblog.org/2022/01/28/pittsburgh-bridge-collapse-underscores-urgent-need-for-fix-it-first-policy/

A Most Recent Example (2022)

https://www.npr.org/2022/01/28/1076343656/pittsburgh-bridge-collapse-biden-visit

The Takeaway, Again

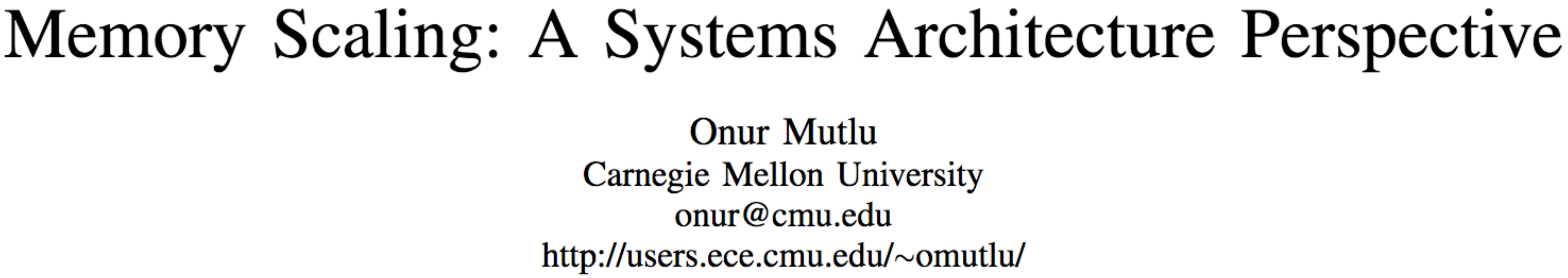
In-Field Patch-ability

(Intelligent Memory)

Can Avoid Such Failures

An Early Proposal for Intelligent Controllers **[IMW'13]**

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

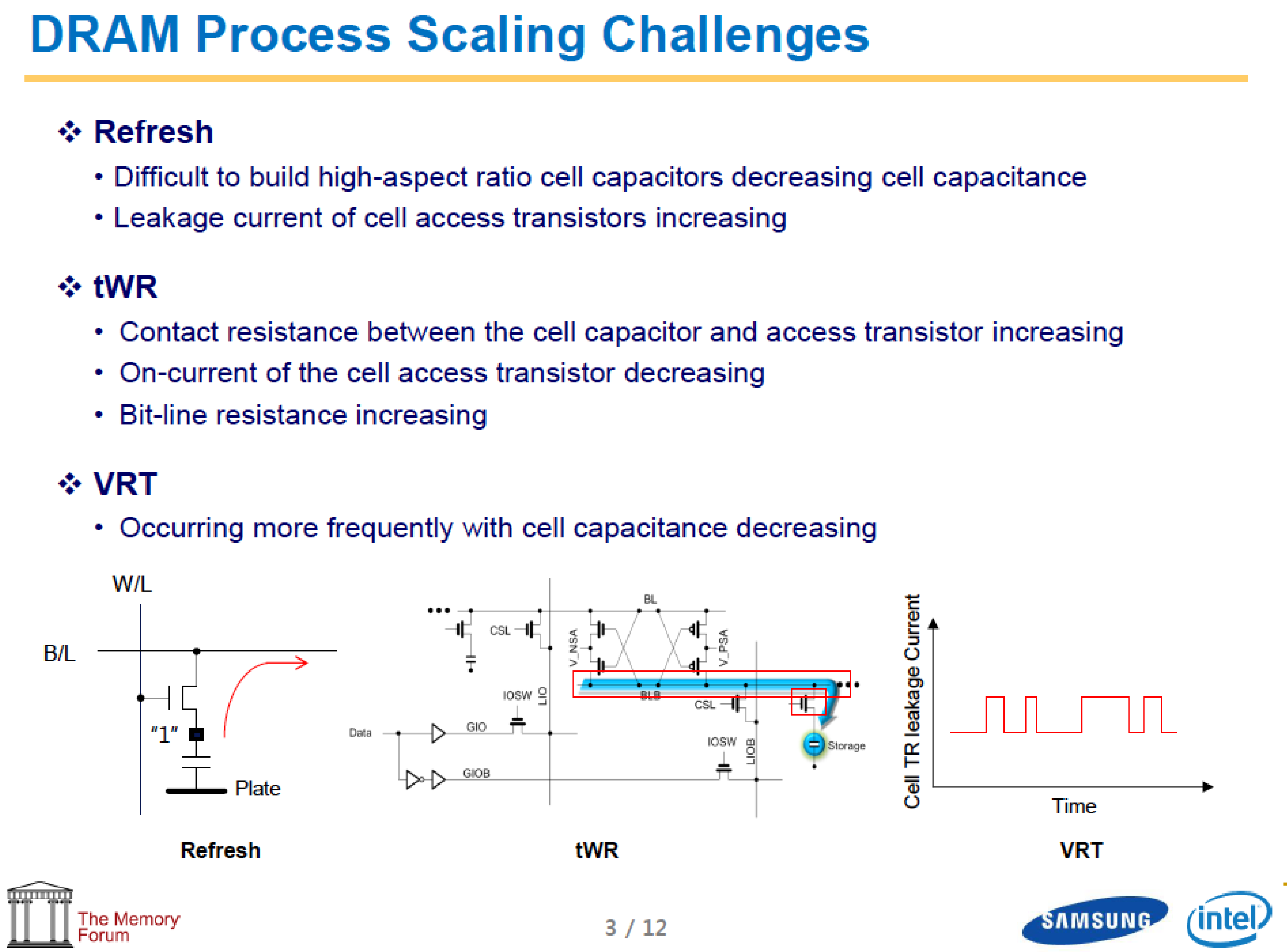


Memory Scaling: A Systems Architecture Perspective

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

https://people.inf.ethz.ch/omutlu/pub/memory-scaling_memcon13.pdf

Industry Is Writing Papers About It, Too

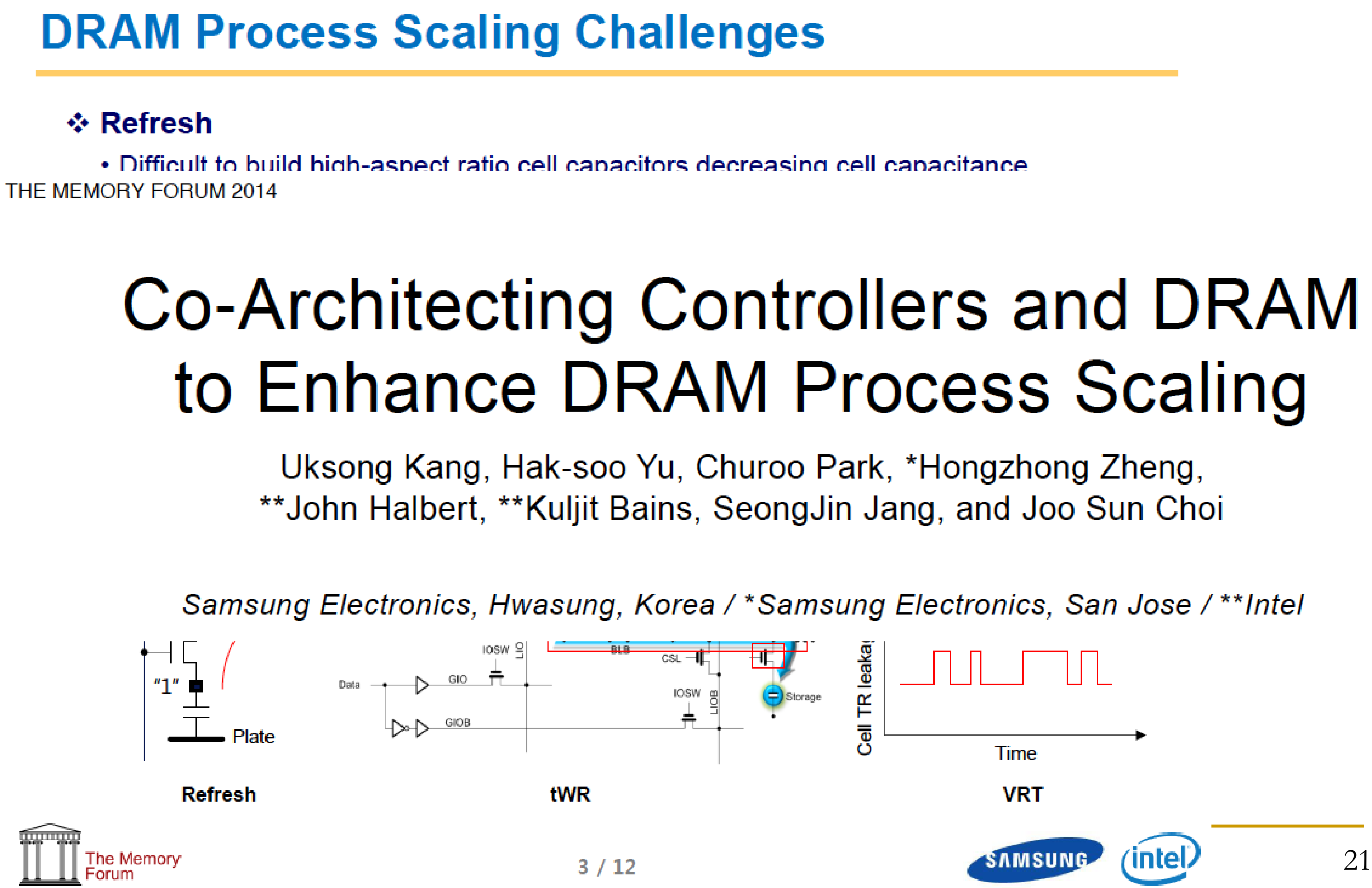


DRAM Process Scaling Challenges

- **Refresh**
  - Difficult to build high-aspect ratio cell capacitors decreasing cell capacitance
  - Leakage current of cell access transistors increasing
- **tWR**
  - Contact resistance between the cell capacitor and access transistor increasing
  - On-current of the cell access transistor decreasing
  - Bit-line resistance increasing
- **VRT**
  - Occurring more frequently with cell capacitance decreasing

Refresh **tWR** **VRT**

Industry Is Writing Papers About It, Too



DRAM Process Scaling Challenges

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

THE MEMORY FORUM 2014

Co-Architecting Controllers and DRAM to Enhance DRAM Process Scaling

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

*Samsung Electronics, Hwasung, Korea / *Samsung Electronics, San Jose / **Intel*

Refresh **tWR** **VRT**

Final Thoughts on RowHammer

Aside: Byzantine Failures

- This class of failures is known as Byzantine failures

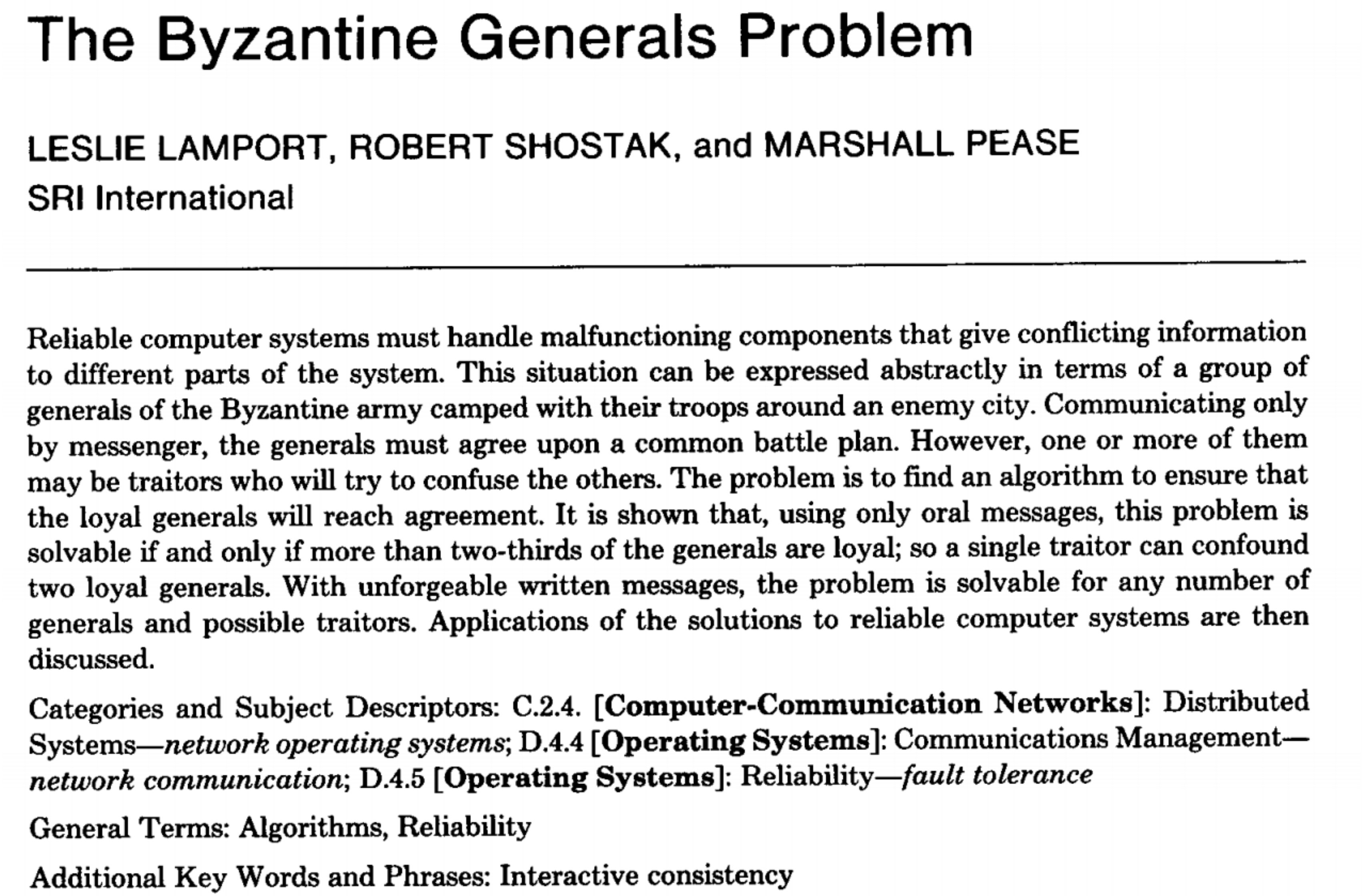
- Characterized by
  - Undetected erroneous computation
  - Opposite of "fail fast (with an error or no result)"

- "erroneous" can be "malicious" (intent is the only distinction)
- Very difficult to detect and confine Byzantine failures
- Do all you can to avoid them

- Lamport et al., "The Byzantine Generals Problem," ACM TOPLAS 1982.

Slide credit: Mahadev Satyanarayanan, CMU, 15-440, Spring 2015

Aside: Byzantine Generals Problem



The Byzantine Generals Problem

LESLIE LAMPORT, ROBERT SHOSTAK, and MARSHALL PEASE
SRI International

Reliable computer systems must handle malfunctioning components that give conflicting information to different parts of the system. This situation can be expressed abstractly in terms of a group of generals of the Byzantine army camped with their troops around an enemy city. Communicating only by messenger, the generals must agree upon a common battle plan. However, one or more of them may be traitors who will try to confuse the others. The problem is to find an algorithm to ensure that the loyal generals will reach agreement. It is shown that, using only oral messages, this problem is solvable if and only if more than two-thirds of the generals are loyal; so a single traitor can confound two loyal generals. With unforgeable written messages, the problem is solvable for any number of generals and possible traitors. Applications of the solutions to reliable computer systems are then discussed.

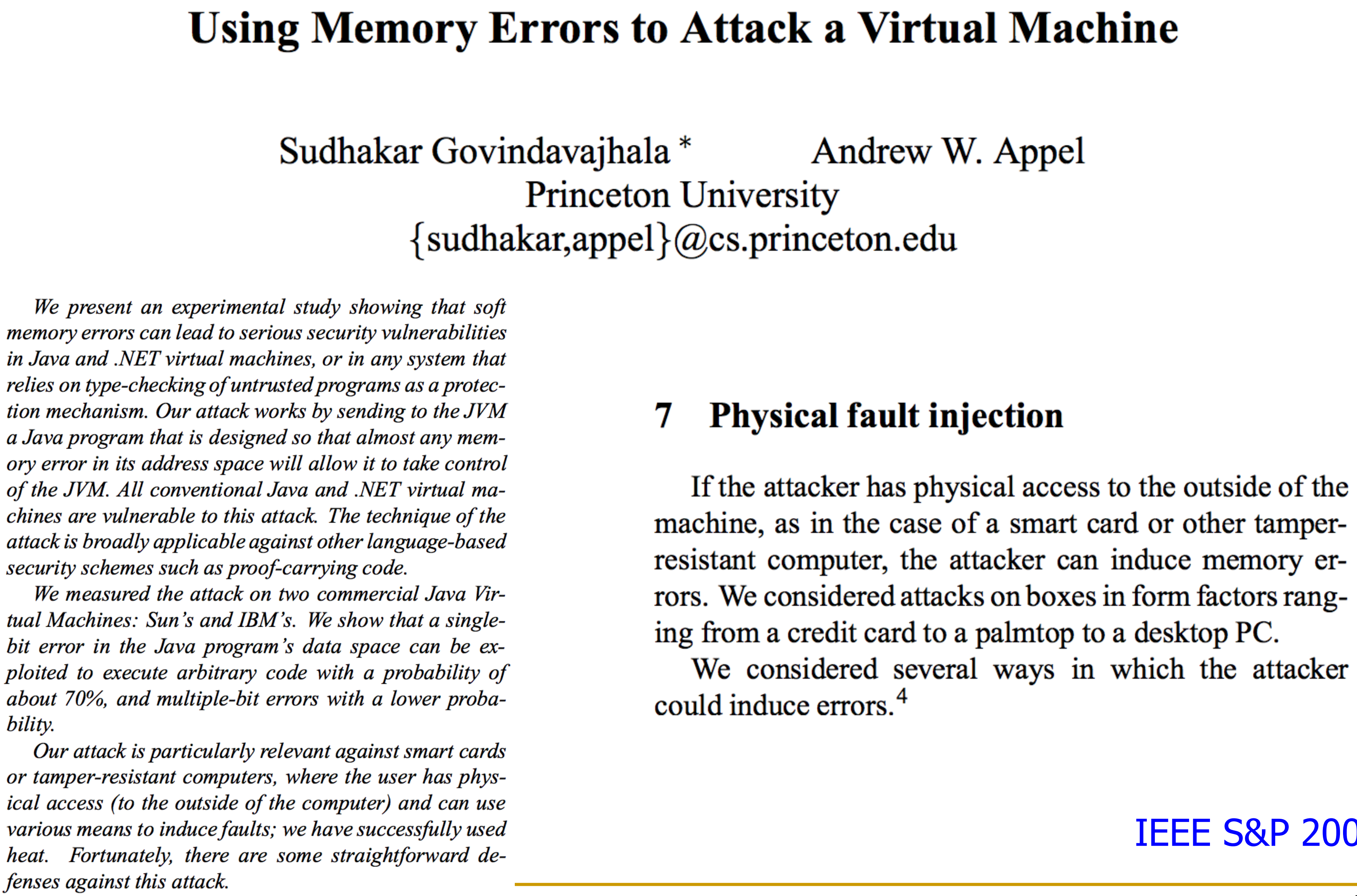
Categories and Subject Descriptors: C.2.4. [**Computer-Communication Networks**]: Distributed Systems—*network operating systems*; D.4.4 [**Operating Systems**]: Communications Management—*network communication*; D.4.5 [**Operating Systems**]: Reliability—*fault tolerance*

General Terms: Algorithms, Reliability

Additional Key Words and Phrases: Interactive consistency

https://dl.acm.org/citation.cfm?id=357176

Before RowHammer (I)



Using Memory Errors to Attack a Virtual Machine

Sudhakar Govindavajhala * Andrew W. Appel
Princeton University
{sudhakar,appel}@cs.princeton.edu

We present an experimental study showing that soft memory errors can lead to serious security vulnerabilities in Java and .NET virtual machines, or in any system that relies on type-checking of untrusted programs as a protection mechanism. Our attack works by sending to the JVM a Java program that is designed so that almost any memory error in its address space will allow it to take control of the JVM. All conventional Java and .NET virtual machines are vulnerable to this attack. The technique of the attack is broadly applicable against other language-based security schemes such as proof-carrying code.

We measured the attack on two commercial Java Virtual Machines: Sun's and IBM's. We show that a single-bit error in the Java program's data space can be exploited to execute arbitrary code with a probability of about 70%, and multiple-bit errors with a lower probability.

Our attack is particularly relevant against smart cards or tamper-resistant computers, where the user has physical access (to the outside of the computer) and can use various means to induce faults; we have successfully used heat. Fortunately, there are some straightforward defenses against this attack.

7 Physical fault injection

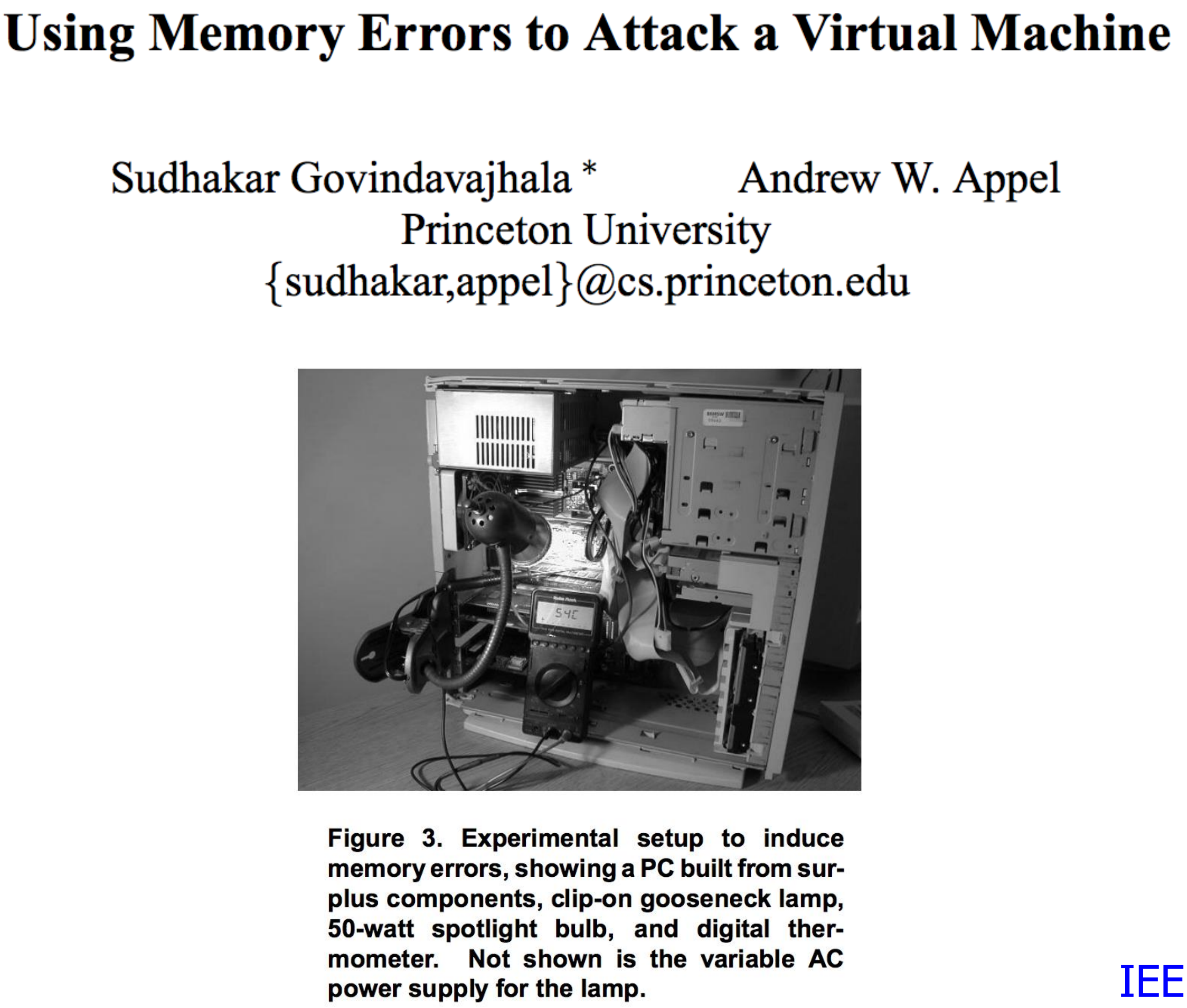
If the attacker has physical access to the outside of the machine, as in the case of a smart card or other tamper-resistant computer, the attacker can induce memory errors. We considered attacks on boxes in form factors ranging from a credit card to a palmtop to a desktop PC.

We considered several ways in which the attacker could induce errors.[4]

IEEE S&P 2003

https://www.cs.princeton.edu/~appel/papers/memerr.pdf

Before RowHammer (II)



Using Memory Errors to Attack a Virtual Machine

Sudhakar Govindavajhala * Andrew W. Appel
Princeton University
{sudhakar,appel}@cs.princeton.edu

Figure 3. Experimental setup to induce memory errors, showing a PC built from surplus components, clip-on gooseneck lamp, 50-watt spotlight bulb, and digital thermometer. Not shown is the variable AC power supply for the lamp.

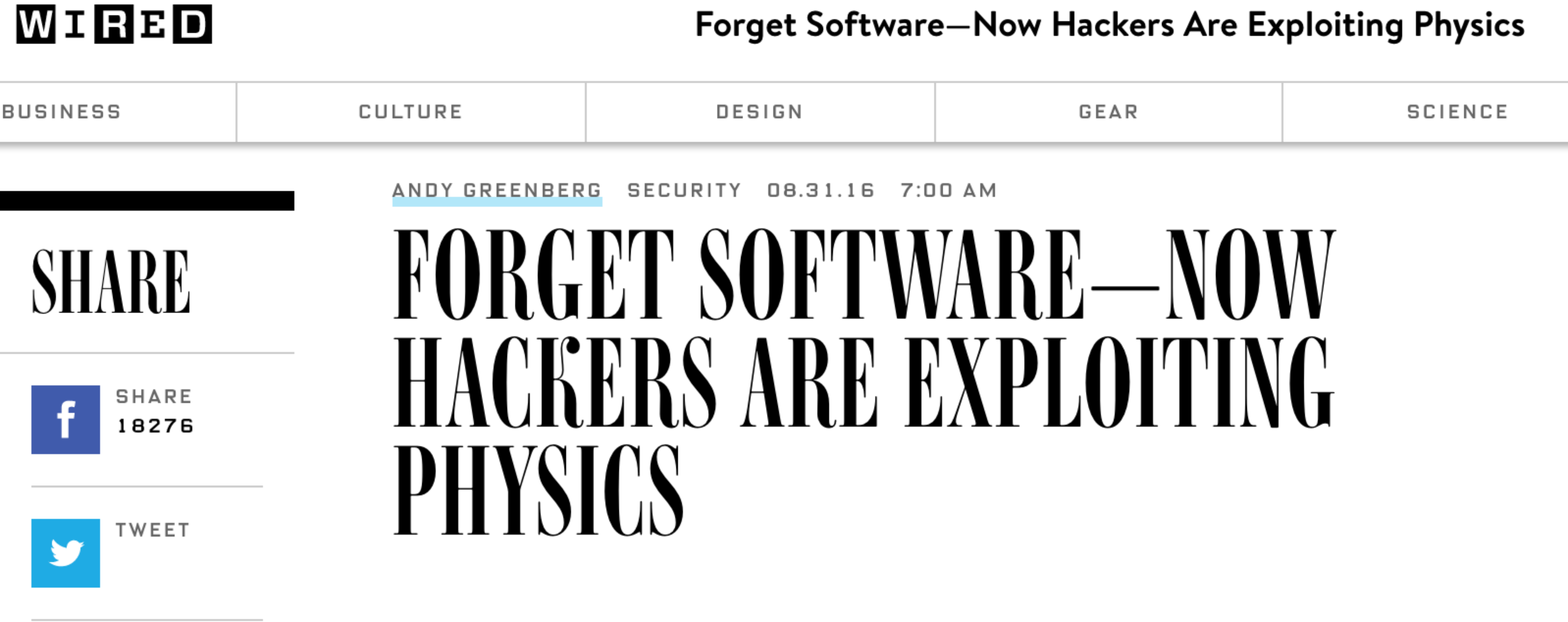
IEEE S&P 2003

https://www.cs.princeton.edu/~appel/papers/memerr.pdf

After RowHammer

A simple memory error

can be induced by software



WIRED

Forget Software—Now Hackers Are Exploiting Physics

BUSINESS CULTURE DESIGN GEAR SCIENCE

ANDY GREENBERG SECURITY 08.31.16 7:00 AM

FORGET SOFTWARE—NOW HACKERS ARE EXPLOITING PHYSICS

SHARE

SHARE 18276

TWEET

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 in ASPLOS 2022)
  - 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…**

Conclusion

Summary: RowHammer

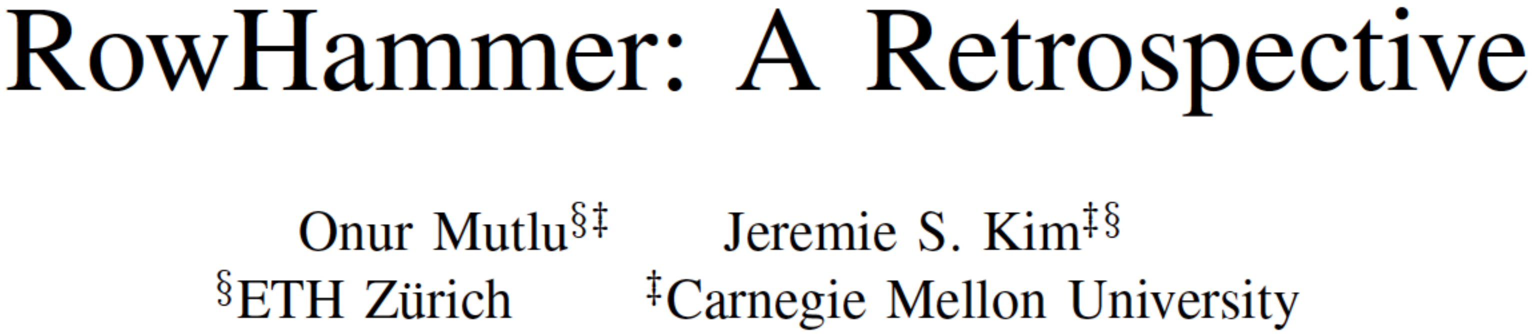
- Memory reliability is reducing
- Reliability issues open up security vulnerabilities
  - Very hard to defend against
- **Rowhammer is a prime example**
  - First example of how a simple hardware failure mechanism can create a widespread system security vulnerability
  - Its implications on system security research are tremendous & exciting

- Bad news: RowHammer is getting worse

- **Good news: We have a lot more to do**
  - We are now fully aware hardware is easily fallible
  - We are developing both attacks and solutions
  - We are developing principled models, methodologies, solutions

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

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

Rowhammer

Funding Acknowledgments

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

Thank you!

Acknowledgments

SAFARI

SAFARI Research Group

safari.ethz.ch

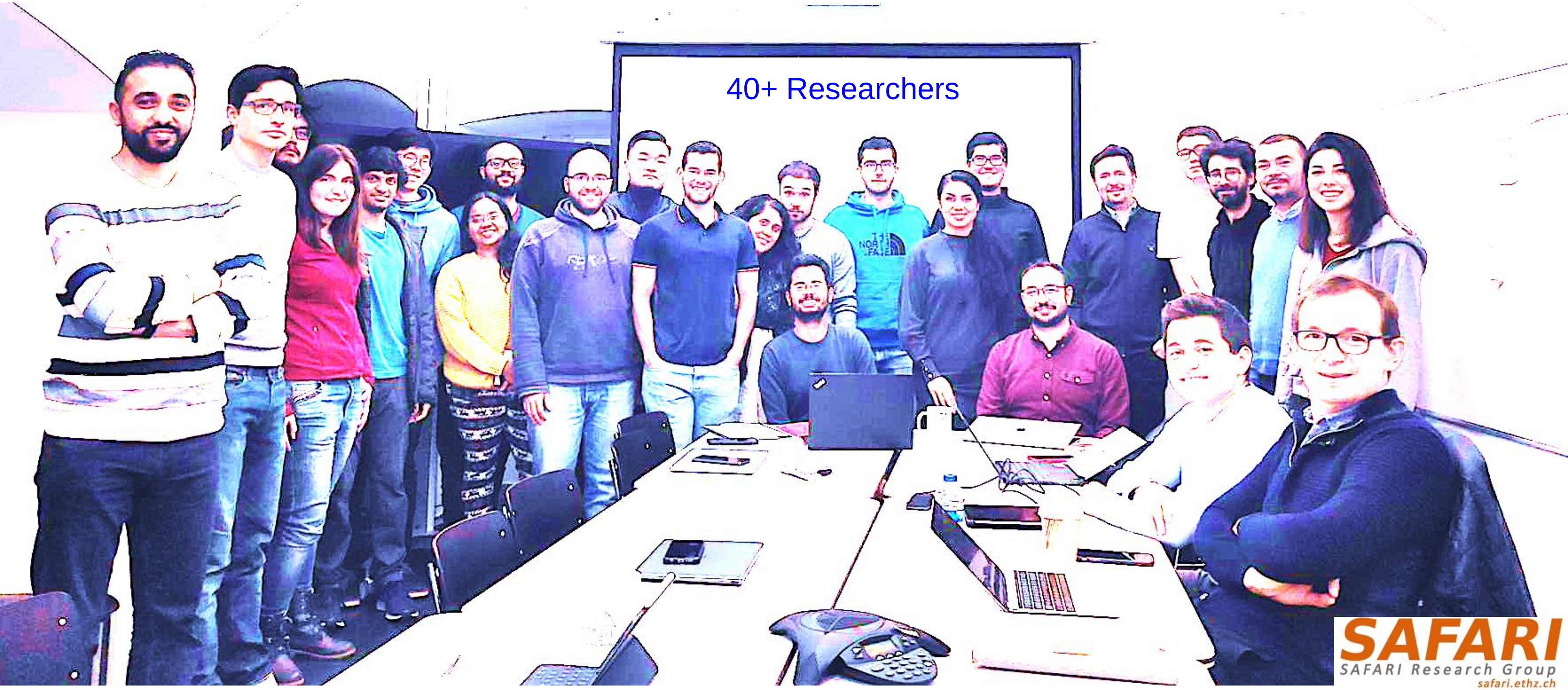
Think BIG, Aim HIGH!

https://safari.ethz.ch

SAFARI Research Group

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

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



Think BIG, Aim HIGH!

SAFARI

https://safari.ethz.ch

SAFARI Research Group

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

SAFARI

SAFARI Research Group

Think Big, Aim High

ETH zürich

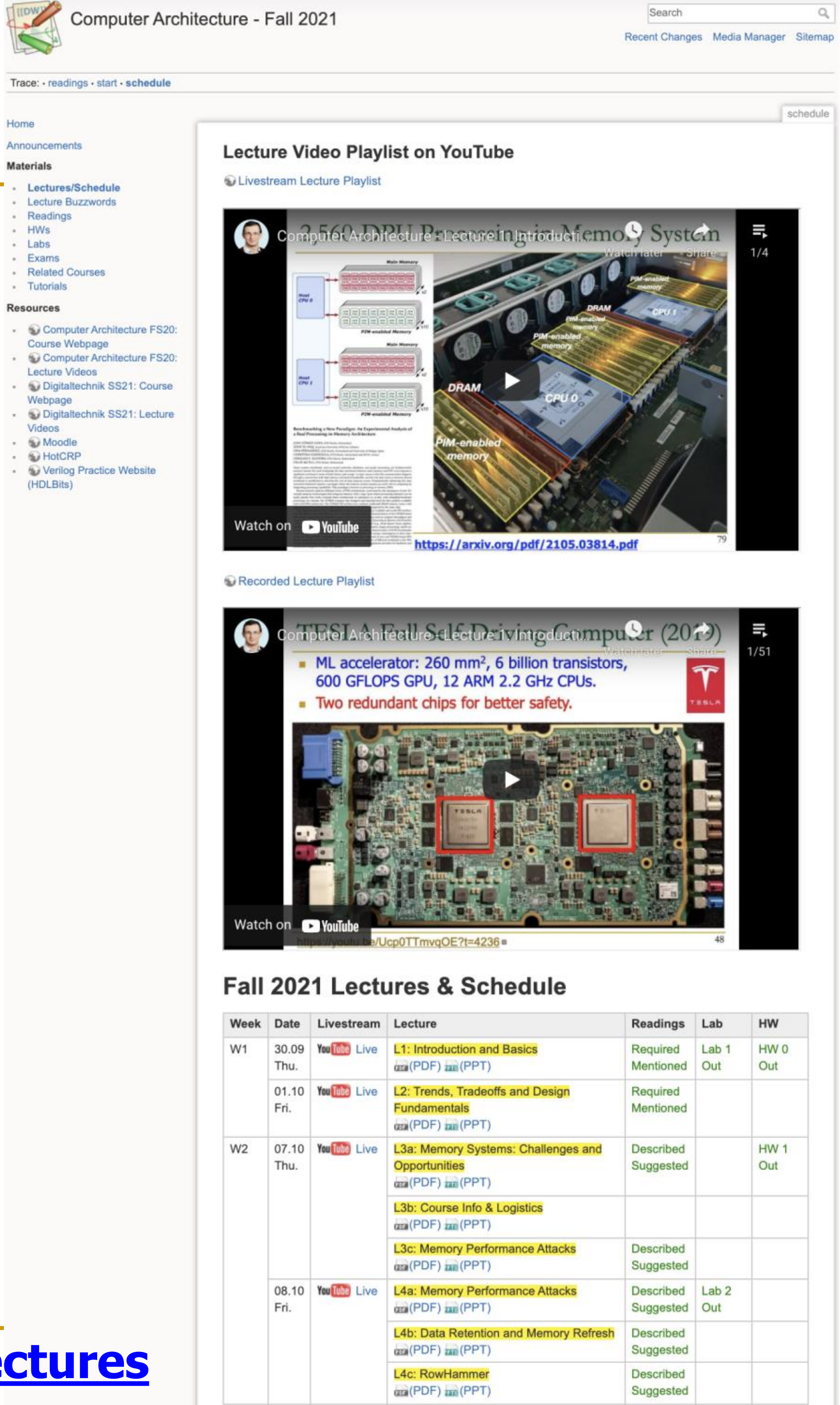
View in your browser

December 2021

Comp Arch (Fall 2021)

- **Fall 2021 Edition:**
  - https://safari.ethz.ch/architecture/fall2021/doku.php?id=schedule
- **Fall 2020 Edition:**
  - https://safari.ethz.ch/architecture/fall2020/doku.php?id=schedule
- **Youtube Livestream (2021):**
  - https://www.youtube.com/watch?v=4yfkM_5EFgo&list=PL5Q2soXY2Zi-Mnk1PxjEIG32HAGILkTOF
- **Youtube Livestream (2020):**
  - https://www.youtube.com/watch?v=c3mPdZA-Fmc&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN

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



Computer Architecture - Fall 2021

Trace: · readings · start · schedule

Home

Announcements

Materials

- Lectures/Schedule
- Lecture Buzzwords
- Readings
- HWs
- Labs
- Exams
- Related Courses
- Tutorials

Resources

- Computer Architecture FS20: Course Webpage
- Computer Architecture FS20: Lecture Videos
- Digitaltechnik SS21: Course Webpage
- Digitaltechnik SS21: Lecture Videos
- Moodle
- HotCRP
- Verilog Practice Website (HDLBits)

Lecture Video Playlist on YouTube

Livestream Lecture Playlist

Computer Architecture - Lecture 1: Introducti...
2,560-DPU Processing-in-Memory System
Watch later
Share
1/4
DRAM
CPU 1
PIM-enabled memory
CPU 0
Watch on YouTube
https://arxiv.org/pdf/2105.03814.pdf
79

Recorded Lecture Playlist

Computer Architecture - Lecture 1: Introducti...
TESLA Full Self-Driving Computer (2019)
Watch later
Share
1/51
ML accelerator: 260 mm², 6 billion transistors, 600 GFLOPS GPU, 12 ARM 2.2 GHz CPUs.
Two redundant chips for better safety.
TESLA
Watch on YouTube
https://youtu.be/Ucp0TTmvqOE?t=4236
48

Fall 2021 Lectures & Schedule

Week	Date	Livestream	Lecture	Readings	Lab	HW
W1	30.09 Thu.	YouTube Live	L1: Introduction and Basics (PDF) (PPT)	Required Mentioned	Lab 1 Out	HW 0 Out
	01.10 Fri.	YouTube Live	L2: Trends, Tradeoffs and Design Fundamentals (PDF) (PPT)	Required Mentioned		
W2	07.10 Thu.	YouTube Live	L3a: Memory Systems: Challenges and Opportunities (PDF) (PPT)	Described Suggested		HW 1 Out
			L3b: Course Info & Logistics (PDF) (PPT)			
			L3c: Memory Performance Attacks (PDF) (PPT)	Described Suggested		
	08.10 Fri.	YouTube Live	L4a: Memory Performance Attacks (PDF) (PPT)	Described Suggested	Lab 2 Out	
			L4b: Data Retention and Memory Refresh (PDF) (PPT)	Described Suggested		
			L4c: RowHammer (PDF) (PPT)	Described Suggested		

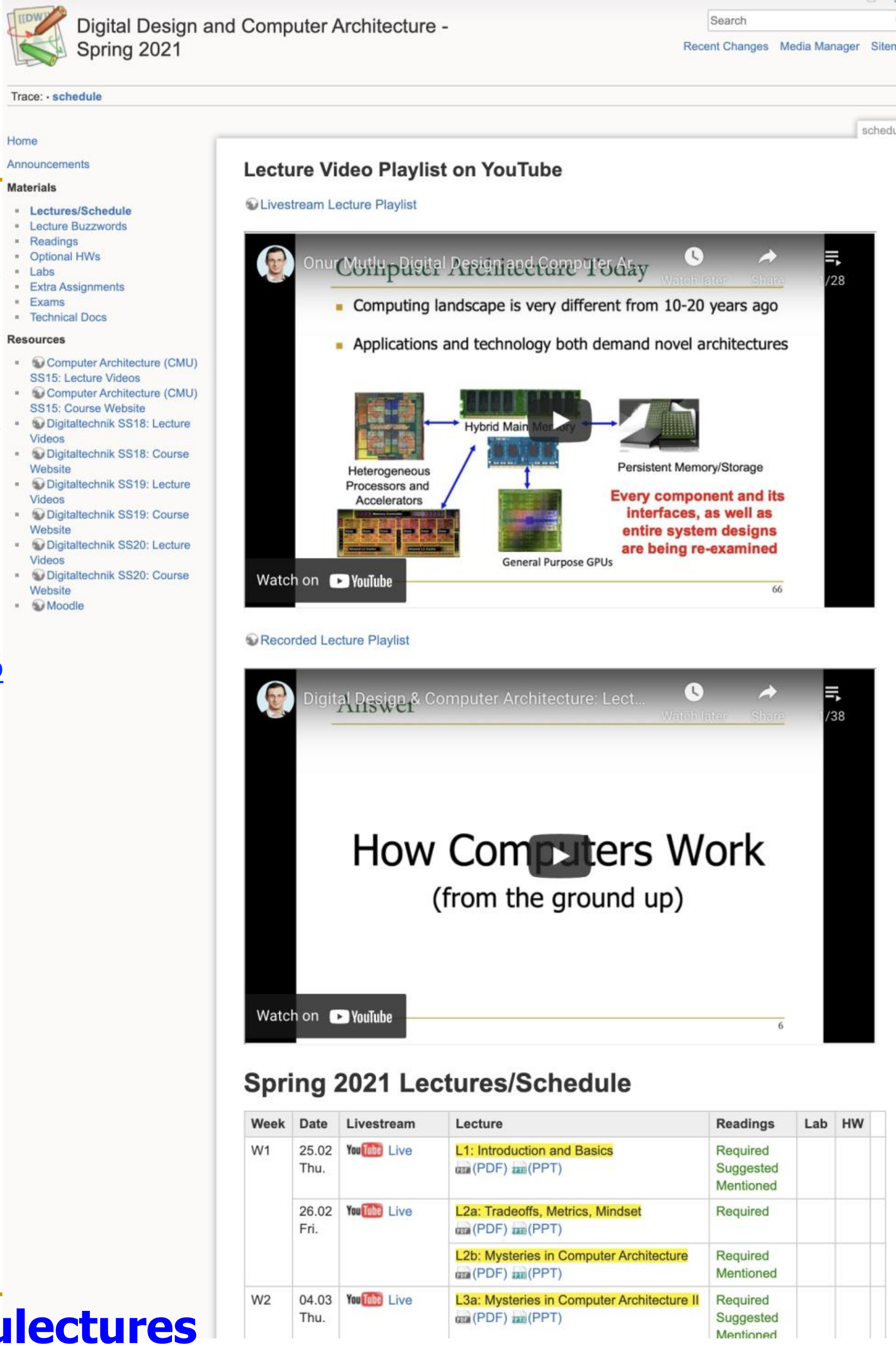
https://www.youtube.com/onurmutlulectures

DDCA (Spring 2022)

- **Spring 2022 Edition:**
  - https://safari.ethz.ch/digitaltechnik/spring2022/doku.php?id=schedule
- **Spring 2021 Edition:**
  - https://safari.ethz.ch/digitaltechnik/spring2021/doku.php?id=schedule

- **Youtube Livestream (Spring 2022):**
  - https://www.youtube.com/watch?v=cpXdE3HwvK0&list=PL5Q2soXY2Zi97Ya5DEUpMpO2bbAoaG7c6
- **Youtube Livestream (Spring 2021):**
  - https://www.youtube.com/watch?v=LbC0EZY8yw4&list=PL5Q2soXY2Zi_uej3aY39YB5pfW4SJ7LlN

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



Digital Design and Computer Architecture - Spring 2021

Search

Recent Changes Media Manager Siten

Trace: · **schedule**

Home

Announcements

Materials

- **Lectures/Schedule**
- Lecture Buzzwords
- Readings
- Optional HWs
- Labs
- Extra Assignments
- Exams
- Technical Docs

Resources

- Computer Architecture (CMU) SS15: Lecture Videos
- Computer Architecture (CMU) SS15: Course Website
- Digitaltechnik SS18: Lecture Videos
- Digitaltechnik SS18: Course Website
- Digitaltechnik SS19: Lecture Videos
- Digitaltechnik SS19: Course Website
- Digitaltechnik SS20: Lecture Videos
- Digitaltechnik SS20: Course Website
- Moodle

Lecture Video Playlist on YouTube

Livestream Lecture Playlist

Onur Mutlu - Digital Design and Computer Ar...
Watch later
Share
1/28
Computer Architecture Today
Computing landscape is very different from 10-20 years ago
Applications and technology both demand novel architectures
Hybrid Main Memory
Heterogeneous Processors and Accelerators
Persistent Memory/Storage
Every component and its interfaces, as well as entire system designs are being re-examined
General Purpose GPUs
Watch on YouTube
66

Recorded Lecture Playlist

Spring 2021 Lectures/Schedule

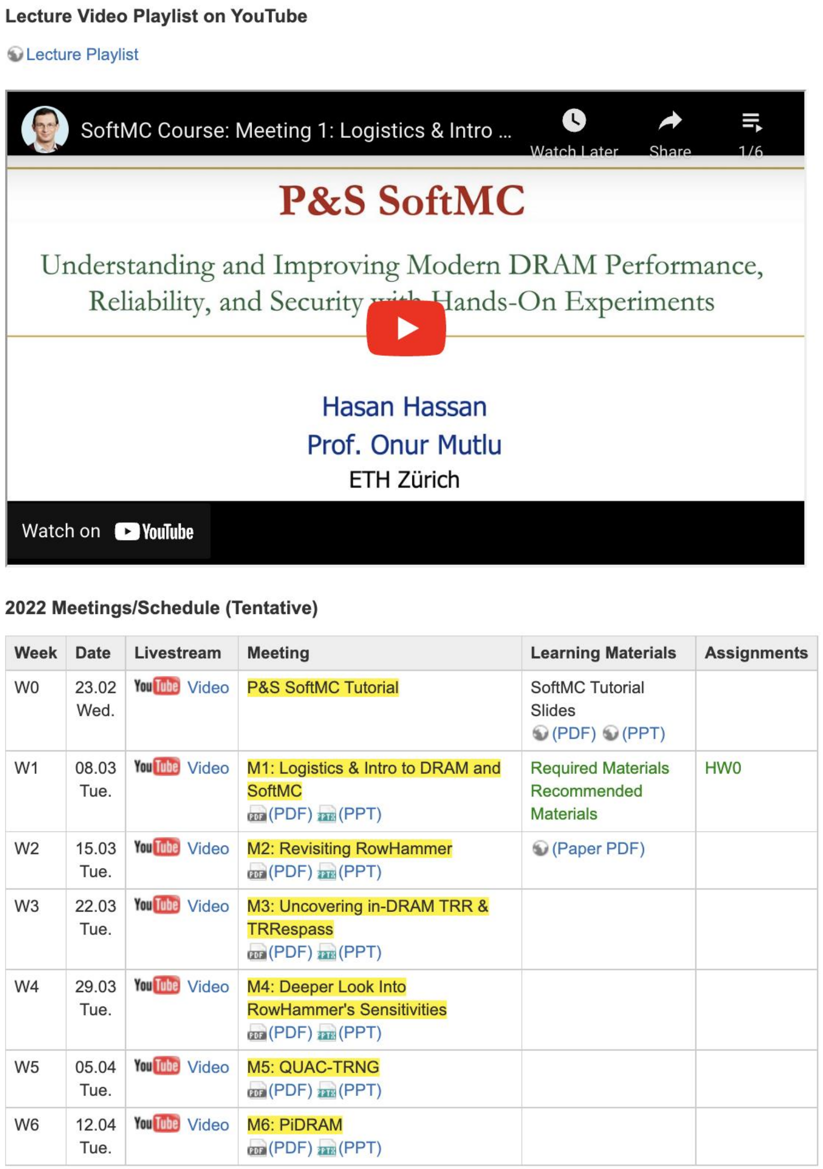
Week	Date	Livestream	Lecture	Readings	Lab	HW
W1	25.02 Thu.	YouTube Live	L1: Introduction and Basics (PDF) (PPT)	Required Suggested Mentioned		
	26.02 Fri.	YouTube Live	L2a: Tradeoffs, Metrics, Mindset (PDF) (PPT)	Required		
			L2b: Mysteries in Computer Architecture (PDF) (PPT)	Required Mentioned		
W2	04.03 Thu.	YouTube Live	L3a: Mysteries in Computer Architecture II (PDF) (PPT)	Required Suggested Mentioned		

https://www.youtube.com/onurmutlulectures

Projects & Seminars: SoftMC

FPGA-Based Exploration of DRAM and RowHammer (Fall 2022)

- **Fall 2022 Edition:**
  - https://safari.ethz.ch/projects_and_seminars/fall2022/doku.php?id=softmc
- **Spring 2022 Edition:**
  - https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=softmc
- **Youtube Livestream (Spring 2022):**
  - https://www.youtube.com/watch?v=r5QxuoJWttg&list=PL5Q2soXY2Zi_1trfCckr6PTN8WR72icUO
- Bachelor's course
  - Elective at ETH Zurich
  - Introduction to DRAM organization & operation
  - Tutorial on using FPGA-based infrastructure
  - Verilog & C++
  - Potential research exploration



Lecture Video Playlist on YouTube

Lecture Playlist

SoftMC Course: Meeting 1: Logistics & Intro ...

P&S SoftMC

Understanding and Improving Modern DRAM Performance, Reliability, and Security with Hands-On Experiments

Hasan Hassan

Prof. Onur Mutlu

ETH Zürich

2022 Meetings/Schedule (Tentative)

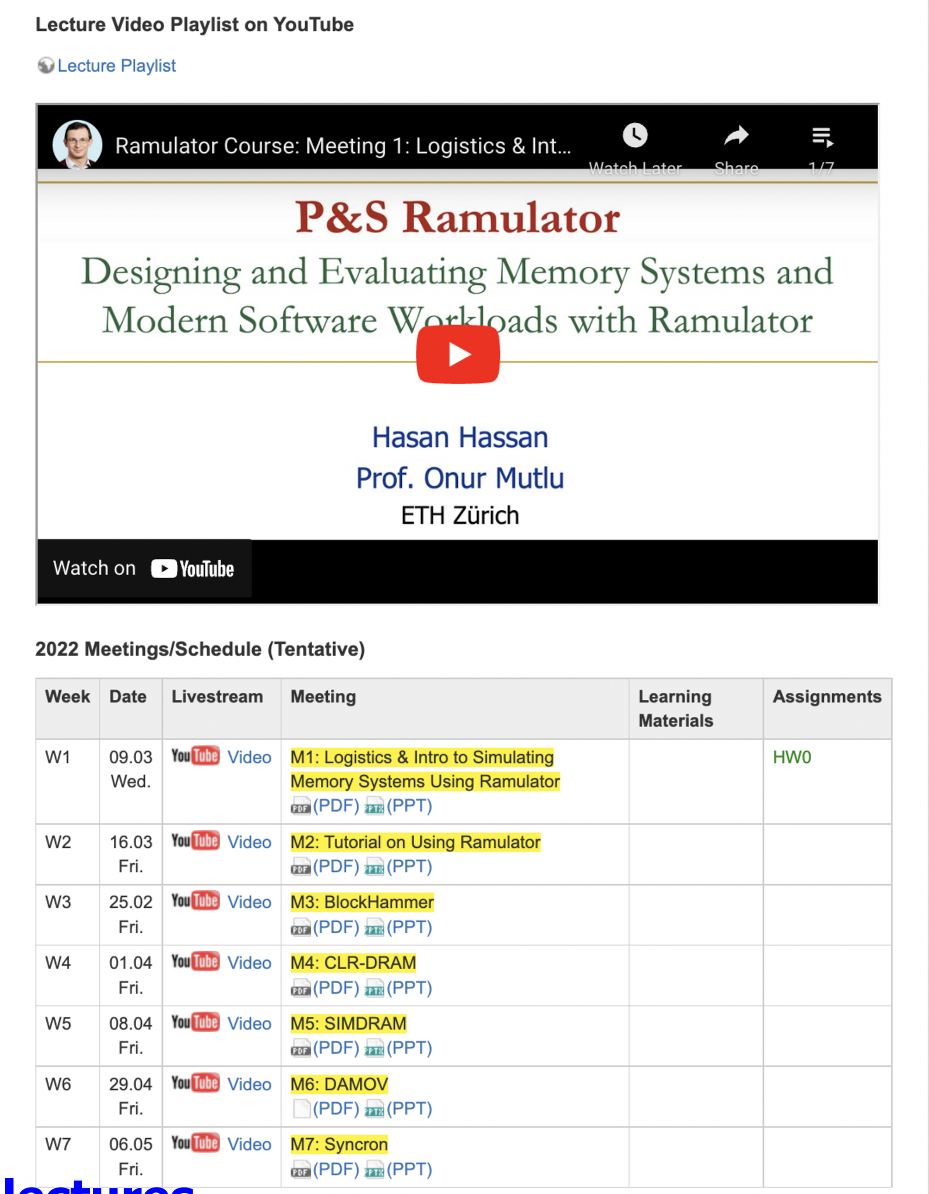
Week	Date	Livestream	Meeting	Learning Materials	Assignments
W0	23.02 Wed.	YouTube Video	P&S SoftMC Tutorial	SoftMC Tutorial Slides (PDF) (PPT)	
W1	08.03 Tue.	YouTube Video	M1: Logistics & Intro to DRAM and SoftMC (PDF) (PPT)	Required Materials Recommended Materials	HW0
W2	15.03 Tue.	YouTube Video	M2: Revisiting RowHammer (PDF) (PPT)	(Paper PDF)	
W3	22.03 Tue.	YouTube Video	M3: Uncovering in-DRAM TRR & TRRespass (PDF) (PPT)		
W4	29.03 Tue.	YouTube Video	M4: Deeper Look Into RowHammer's Sensitivities (PDF) (PPT)		
W5	05.04 Tue.	YouTube Video	M5: QUAC-TRNG (PDF) (PPT)		
W6	12.04 Tue.	YouTube Video	M6: PiDRAM (PDF) (PPT)		

https://www.youtube.com/onurmutlulectures

Projects & Seminars: Ramulator

Exploration of Emerging Memory Systems (Fall 2022)

- **Fall 2022 Edition:**
  - https://safari.ethz.ch/projects_and_seminars/fall2022/doku.php?id=ramulator
- **Spring 2022 Edition:**
  - https://safari.ethz.ch/projects_and_seminars/spring2022/doku.php?id=ramulator
- **Youtube Livestream (Spring 2022):**
  - https://www.youtube.com/watch?v=aM-IlXRQd3s&list=PL5Q2soXY2Zi_TlmLGw_Z8hBo2925ZApqV
- Bachelor's course
  - Elective at ETH Zurich
  - Introduction to memory system simulation
  - Tutorial on using Ramulator
  - C++
  - Potential research exploration



Lecture Video Playlist on YouTube

Lecture Playlist

Ramulator Course: Meeting 1: Logistics & Int...
Watch Later
Share
1/7
P&S Ramulator
Designing and Evaluating Memory Systems and Modern Software Workloads with Ramulator
Hasan Hassan
Prof. Onur Mutlu
ETH Zürich
Watch on YouTube

2022 Meetings/Schedule (Tentative)

Week	Date	Livestream	Meeting	Learning Materials	Assignments
W1	09.03 Wed.	YouTube Video	M1: Logistics & Intro to Simulating Memory Systems Using Ramulator (PDF) (PPT)		HW0
W2	16.03 Fri.	YouTube Video	M2: Tutorial on Using Ramulator (PDF) (PPT)		
W3	25.02 Fri.	YouTube Video	M3: BlockHammer (PDF) (PPT)		
W4	01.04 Fri.	YouTube Video	M4: CLR-DRAM (PDF) (PPT)		
W5	08.04 Fri.	YouTube Video	M5: SIMDRAM (PDF) (PPT)		
W6	29.04 Fri.	YouTube Video	M6: DAMOV (PDF) (PPT)		
W7	06.05 Fri.	YouTube Video	M7: Syncron (PDF) (PPT)		

https://www.youtube.com/onurmutlulectures

Computer Architecture

Lecture 6: The Story of RowHammer

Memory Security & Reliability

Prof. Onur Mutlu

ETH Zürich

Fall 2022

14 October 2022

Backup Slides

RowHammer Review History

Some More Historical Perspective

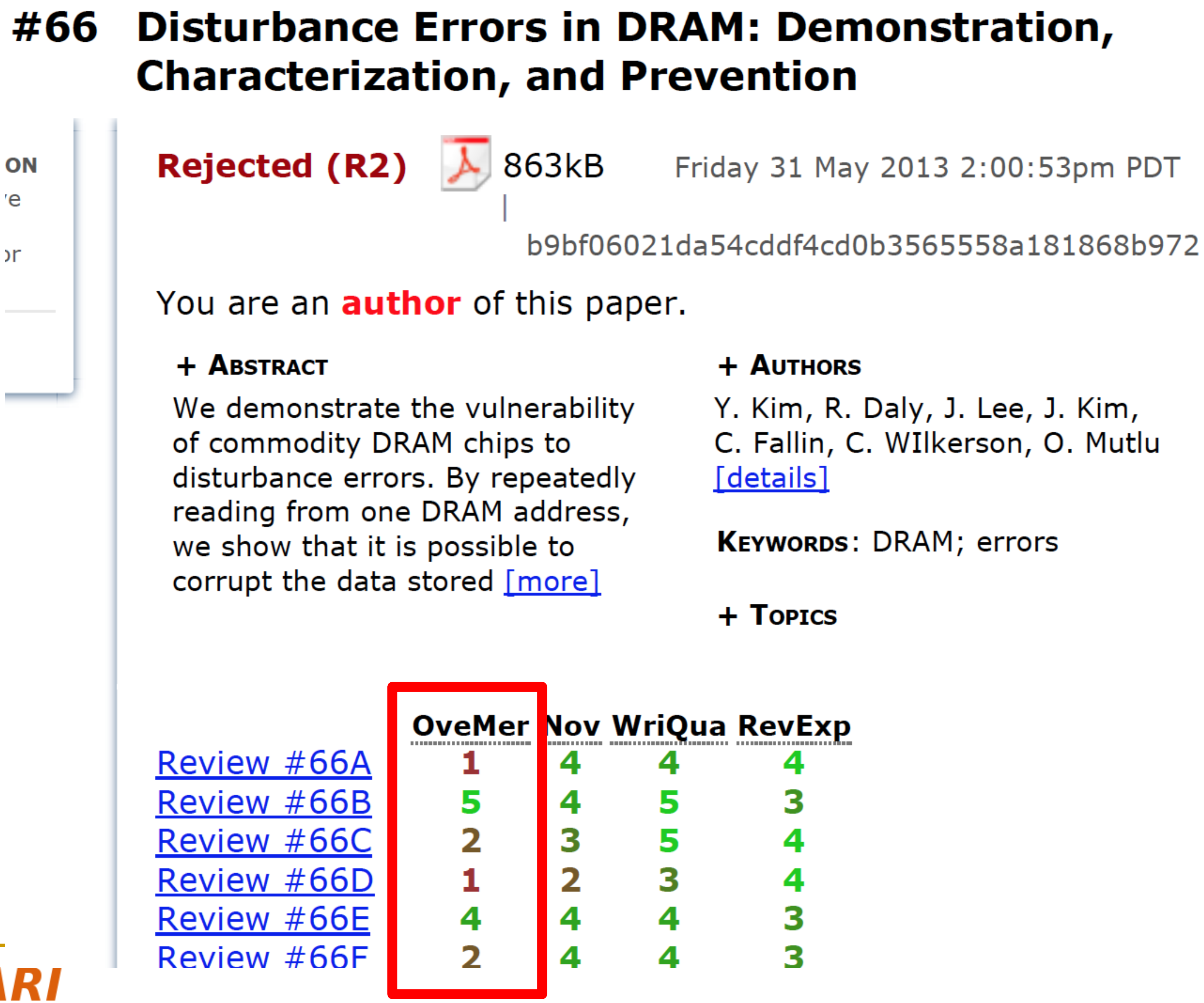
- RowHammer is the first example of a circuit-level failure mechanism causing a widespread system security vulnerability

- It led to a large body of work in security attacks, mitigations, architectural solutions, analyses, …

- Work building on RowHammer still continues
  - See MICRO 2021, and many top venues in 2020/2021

- Initially, it was dismissed by some reviewers
  - Rejected from MICRO 2013 conference

Initial RowHammer Reviews (MICRO 2013)



#66 Disturbance Errors in DRAM: Demonstration, Characterization, and Prevention

ON
'e
or

Rejected (R2) 863kB Friday 31 May 2013 2:00:53pm PDT | b9bf06021da54cddf4cd0b3565558a181868b972

You are an **author** of this paper.

+ **Abstract**

We demonstrate the vulnerability of commodity DRAM chips to disturbance errors. By repeatedly reading from one DRAM address, we show that it is possible to corrupt the data stored [more]

+ **Authors**

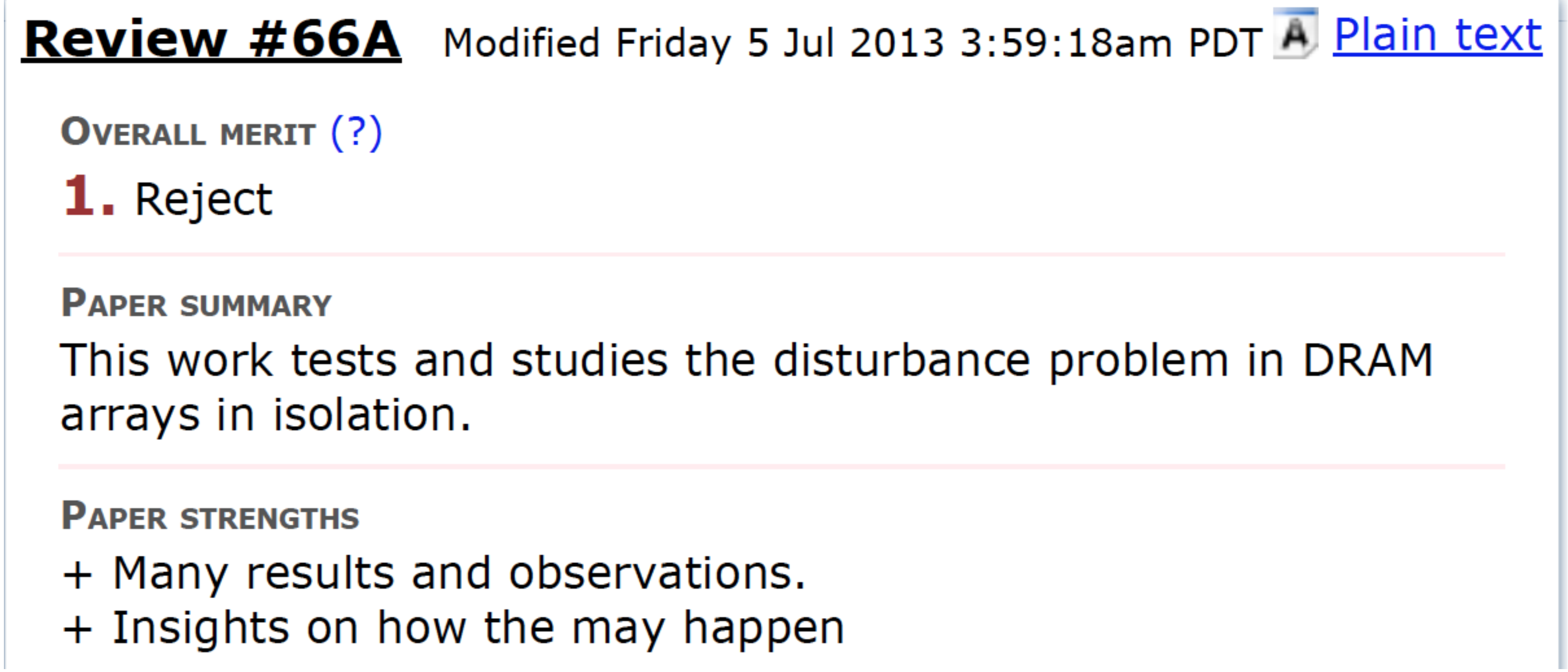
Y. Kim, R. Daly, J. Lee, J. Kim, C. Fallin, C. WIlkerson, O. Mutlu [details]

Keywords: DRAM; errors

+ **Topics**

	OveMer	Nov	WriQua	RevExp
Review #66A	1	4	4	4
Review #66B	5	4	5	3
Review #66C	2	3	5	4
Review #66D	1	2	3	4
Review #66E	4	4	4	3
Review #66F	2	4	4	3

Reviewer A



Review #66A Modified Friday 5 Jul 2013 3:59:18am PDT Plain text

OVERALL MERIT (?)

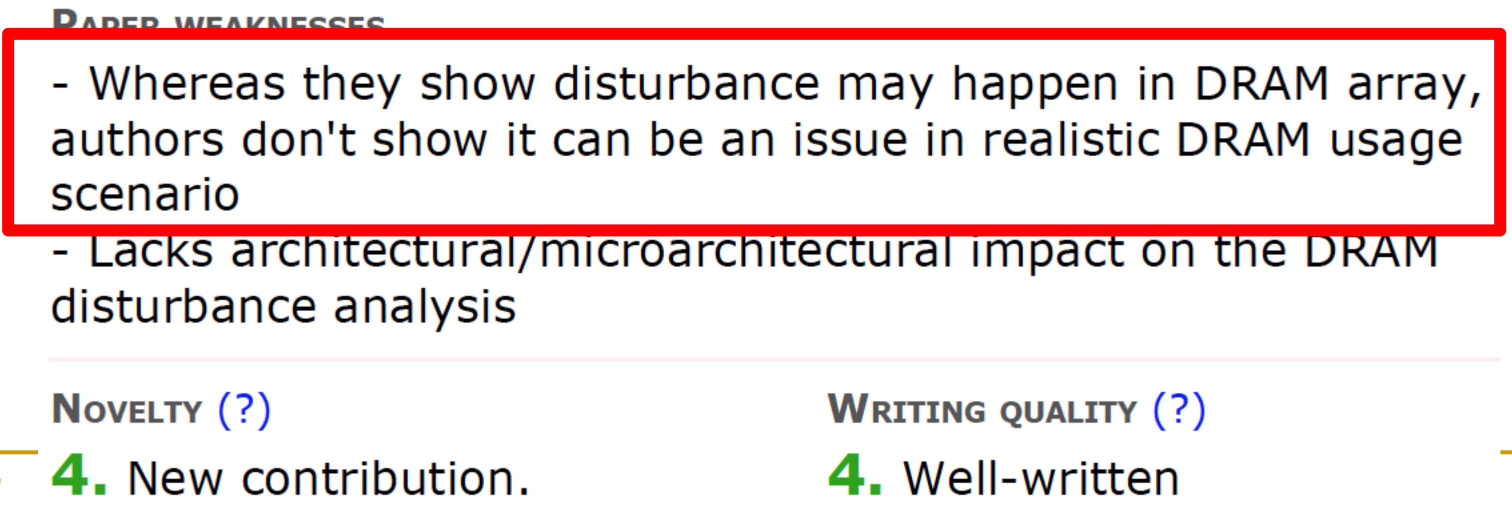
1. Reject

PAPER SUMMARY

This work tests and studies the disturbance problem in DRAM arrays in isolation.

PAPER STRENGTHS

+ Many results and observations.
+ Insights on how the may happen



PAPER WEAKNESSES

- Whereas they show disturbance may happen in DRAM array, authors don't show it can be an issue in realistic DRAM usage scenario
- Lacks architectural/microarchitectural impact on the DRAM disturbance analysis

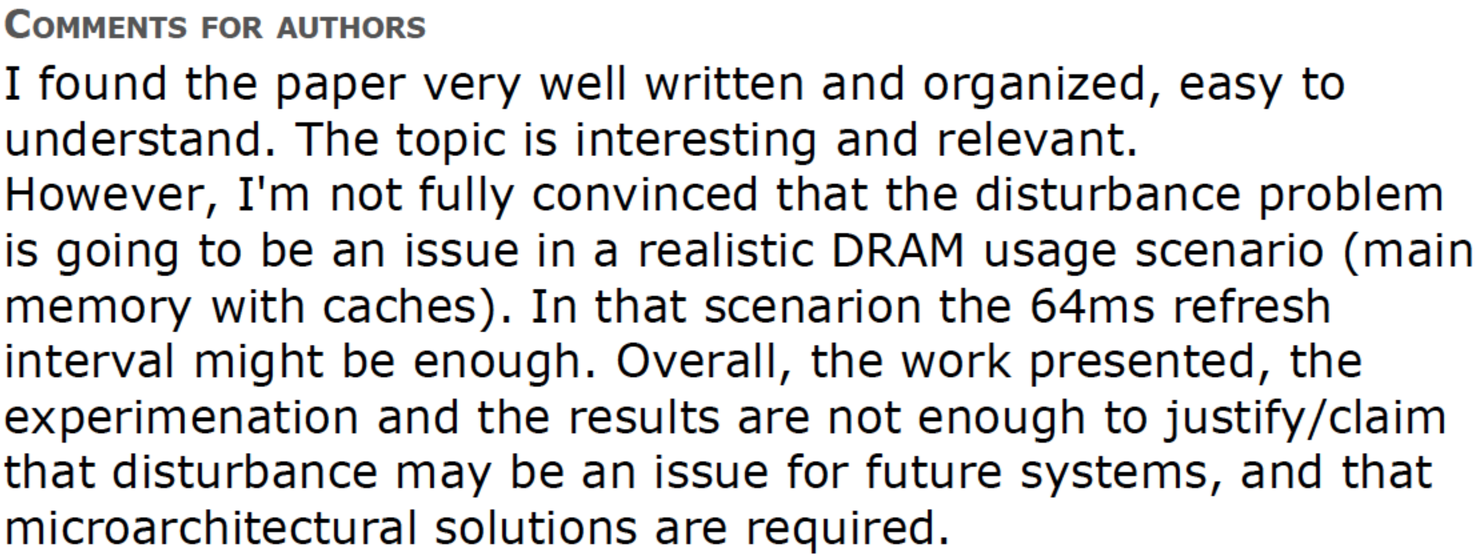
NOVELTY (?)

4. New contribution.

WRITING QUALITY (?)

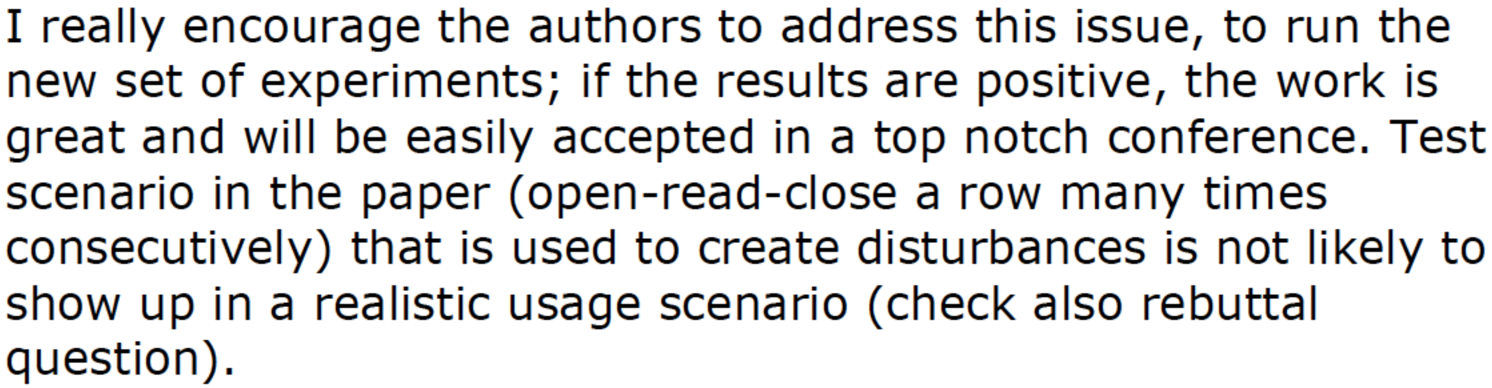
4. Well-written

Reviewer A -- Security is Not "Realistic"



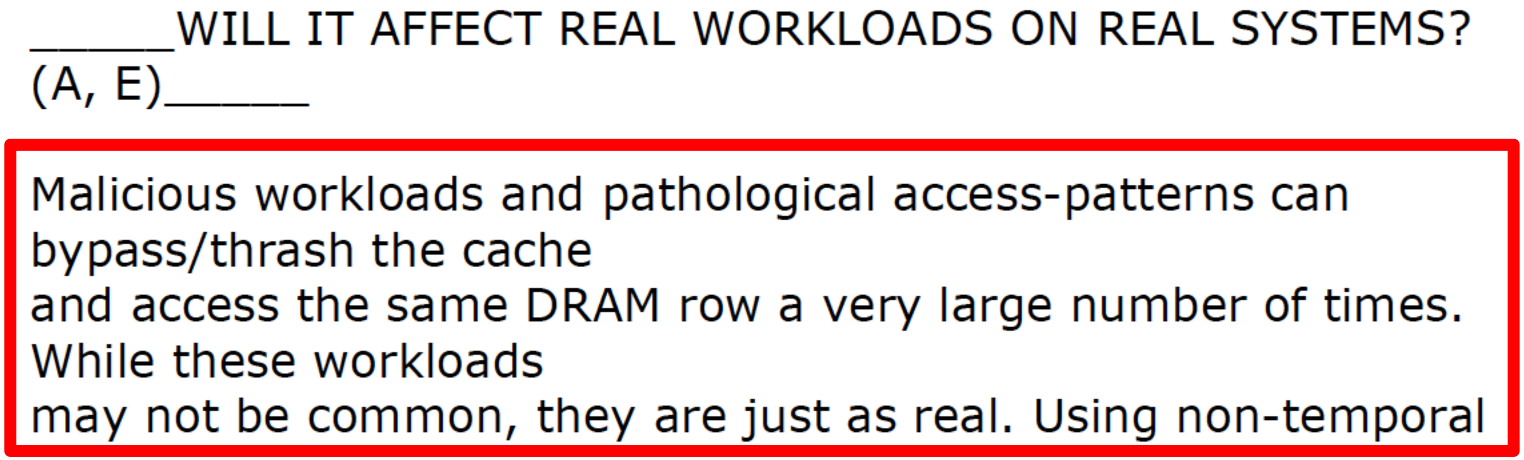
COMMENTS FOR AUTHORS

I found the paper very well written and organized, easy to understand. The topic is interesting and relevant.
However, I'm not fully convinced that the disturbance problem is going to be an issue in a realistic DRAM usage scenario (main memory with caches). In that scenarion the 64ms refresh interval might be enough. Overall, the work presented, the experimenation and the results are not enough to justify/claim that disturbance may be an issue for future systems, and that microarchitectural solutions are required.



I really encourage the authors to address this issue, to run the new set of experiments; if the results are positive, the work is great and will be easily accepted in a top notch conference. Test scenario in the paper (open-read-close a row many times consecutively) that is used to create disturbances is not likely to show up in a realistic usage scenario (check also rebuttal question).

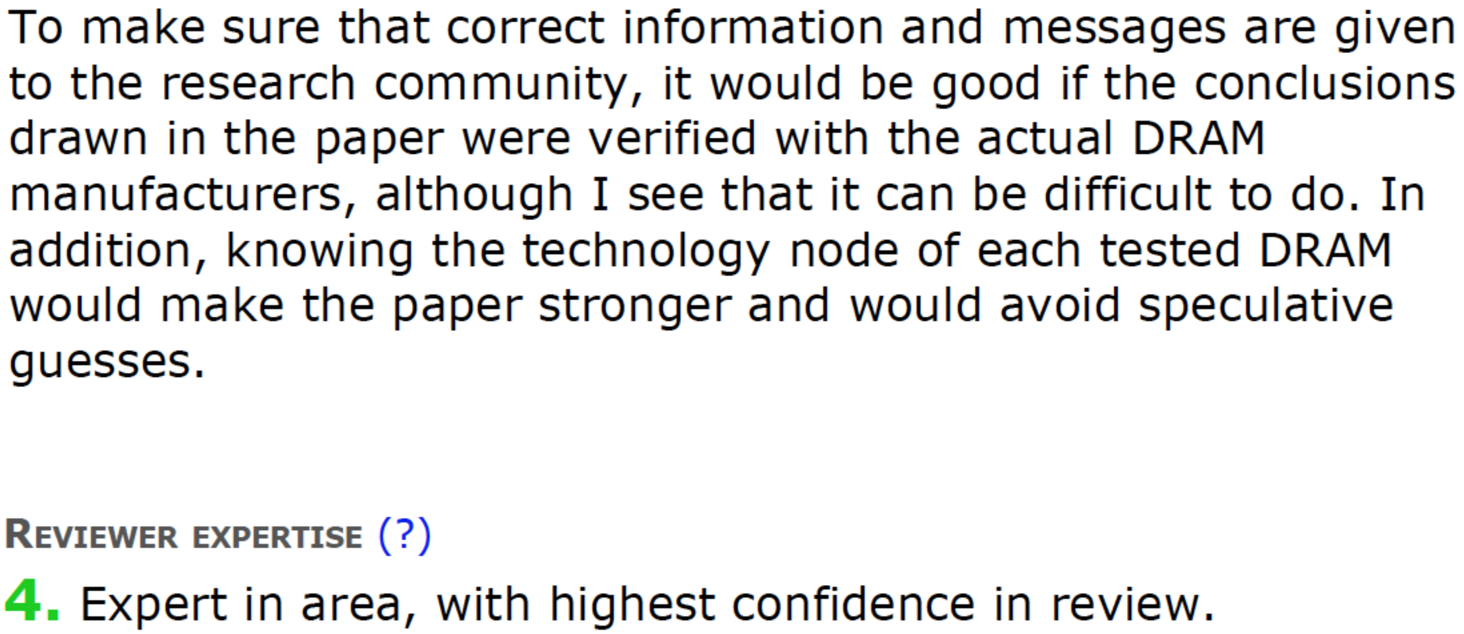
Rebuttal to Reviewer A



______WILL IT AFFECT REAL WORKLOADS ON REAL SYSTEMS?
(A, E)______

Malicious workloads and pathological access-patterns can
bypass/thrash the cache
and access the same DRAM row a very large number of times.
While these workloads
may not be common, they are just as real. Using non-temporal

Reviewer A -- Demands

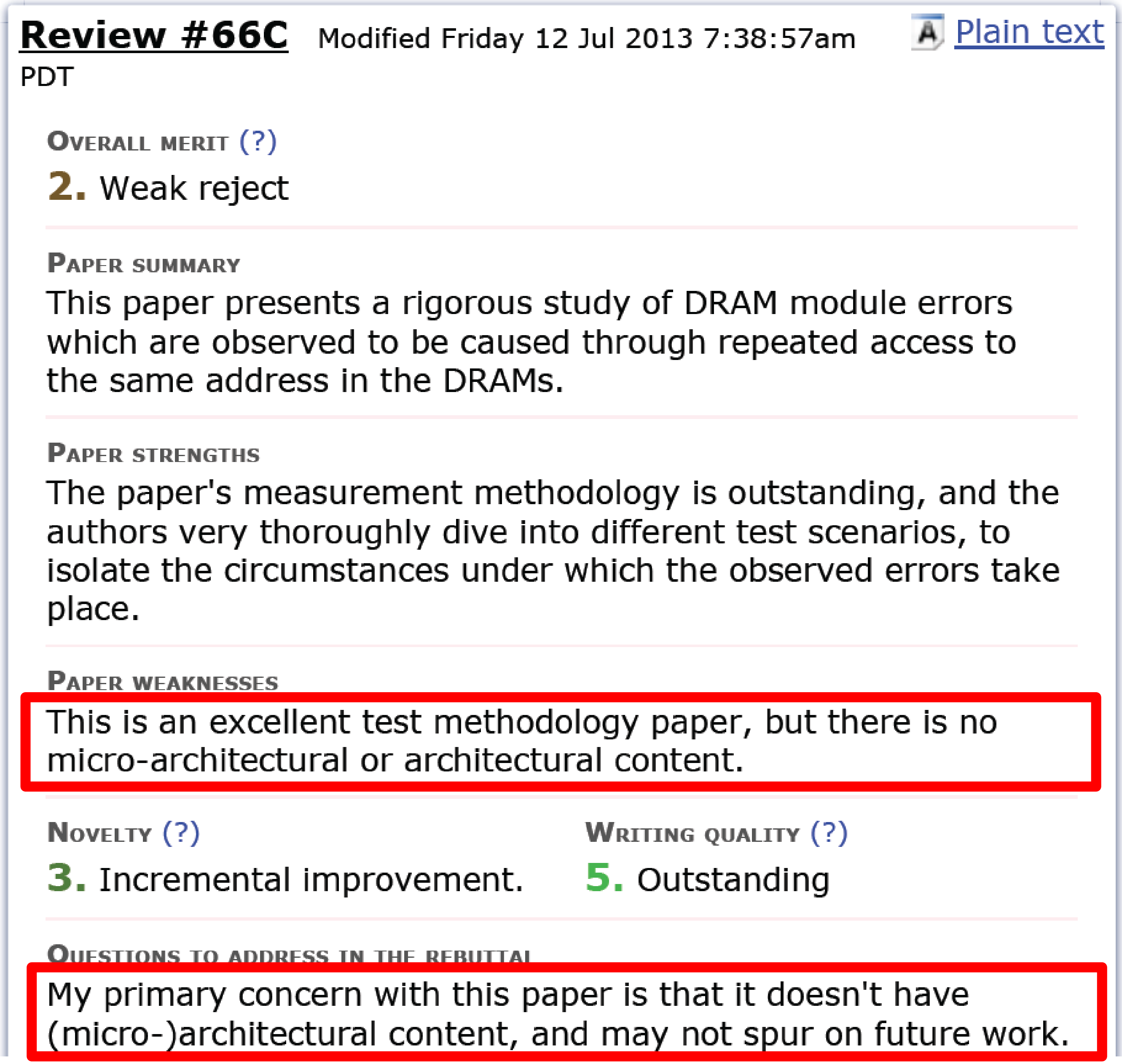


To make sure that correct information and messages are given to the research community, it would be good if the conclusions drawn in the paper were verified with the actual DRAM manufacturers, although I see that it can be difficult to do. In addition, knowing the technology node of each tested DRAM would make the paper stronger and would avoid speculative guesses.

REVIEWER EXPERTISE (?)

4. Expert in area, with highest confidence in review.

Reviewer C



Review #66C Modified Friday 12 Jul 2013 7:38:57am PDT

OVERALL MERIT (?)

2. Weak reject

PAPER SUMMARY

This paper presents a rigorous study of DRAM module errors which are observed to be caused through repeated access to the same address in the DRAMs.

PAPER STRENGTHS

The paper's measurement methodology is outstanding, and the authors very thoroughly dive into different test scenarios, to isolate the circumstances under which the observed errors take place.

PAPER WEAKNESSES

This is an excellent test methodology paper, but there is no micro-architectural or architectural content.

NOVELTY (?)

3. Incremental improvement.

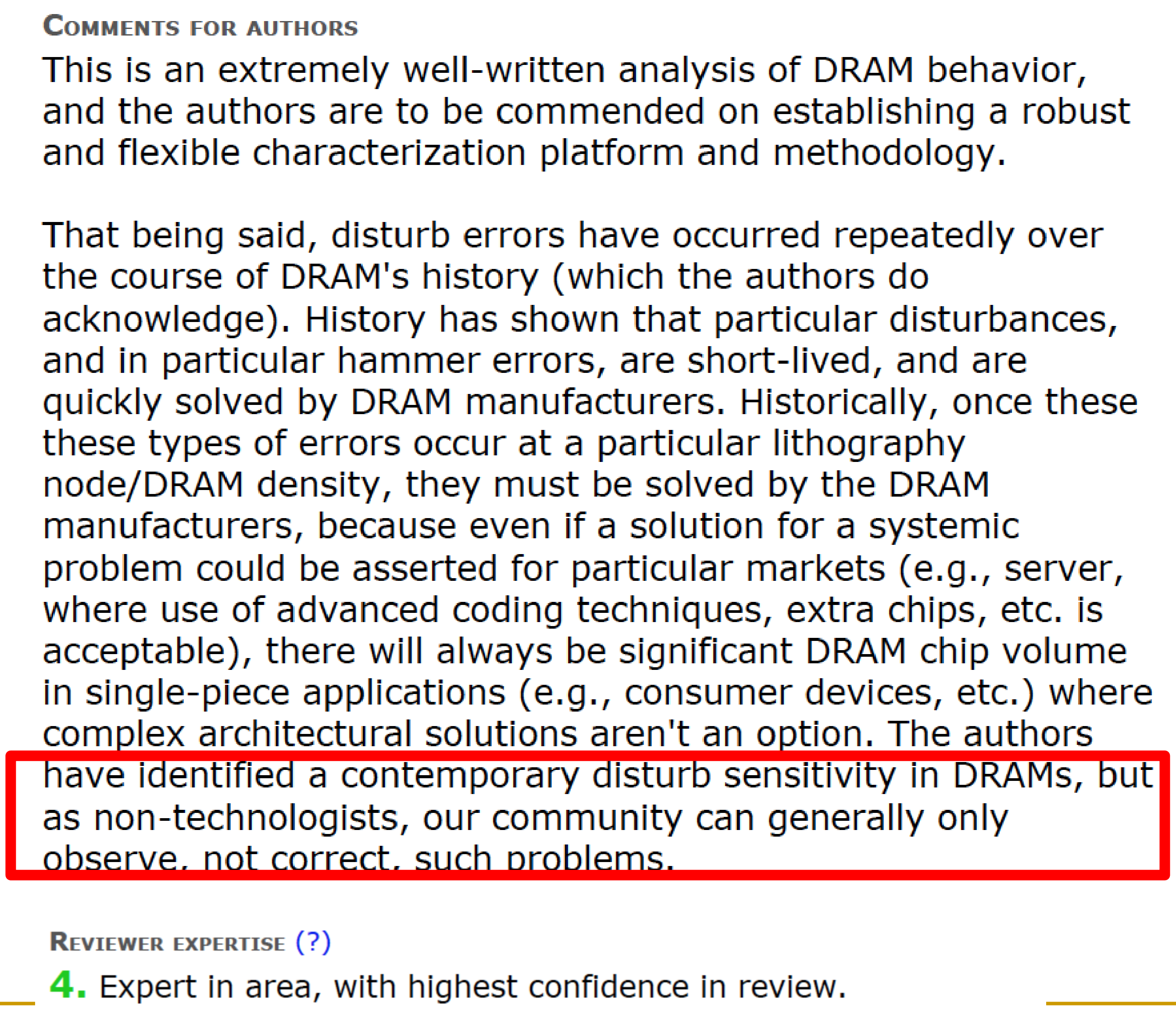
WRITING QUALITY (?)

5. Outstanding

QUESTIONS TO ADDRESS IN THE REBUTTAL

My primary concern with this paper is that it doesn't have (micro-)architectural content, and may not spur on future work.

Reviewer C -- Leave It to DRAM Vendors



Comments for authors

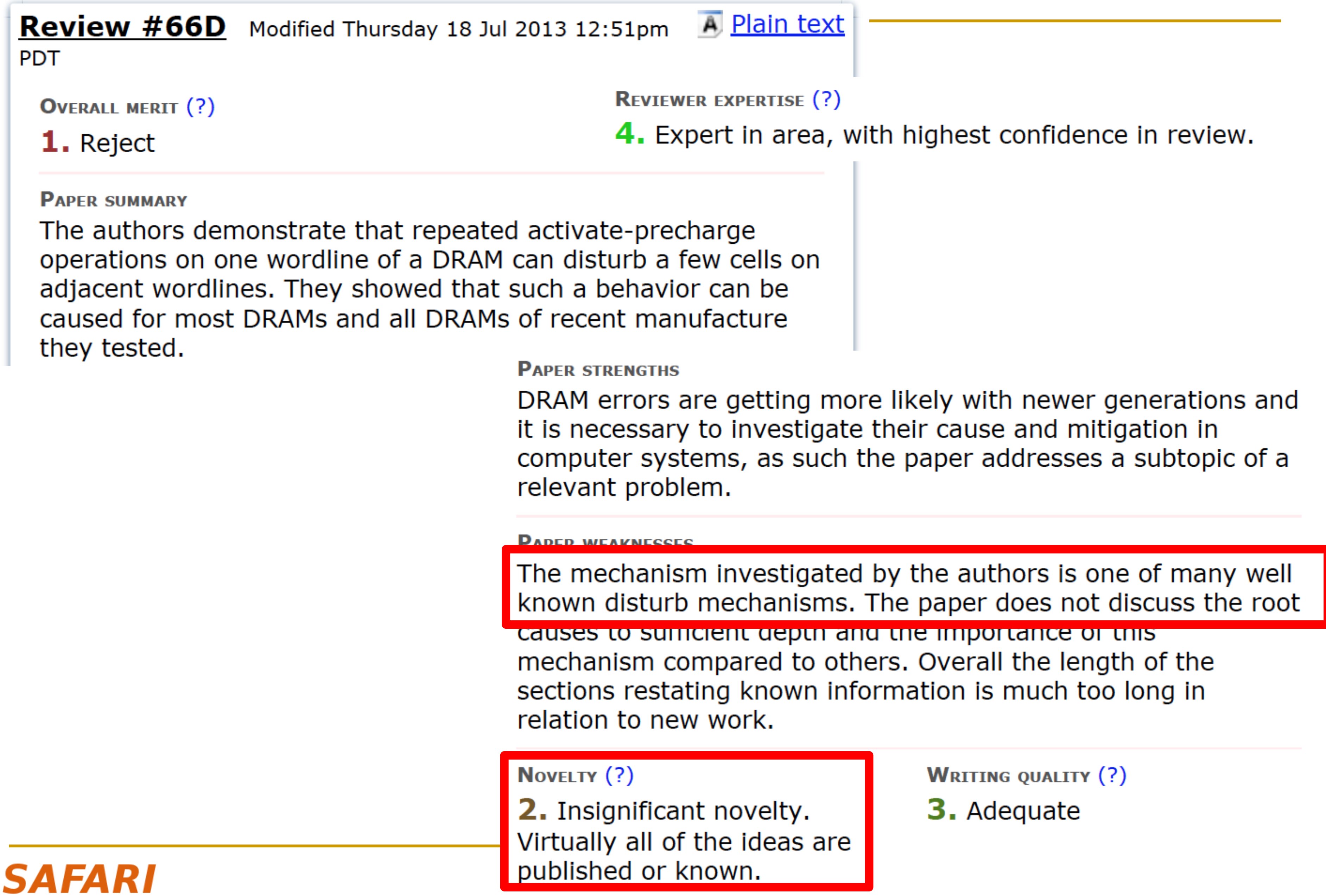
This is an extremely well-written analysis of DRAM behavior, and the authors are to be commended on establishing a robust and flexible characterization platform and methodology.

That being said, disturb errors have occurred repeatedly over the course of DRAM's history (which the authors do acknowledge). History has shown that particular disturbances, and in particular hammer errors, are short-lived, and are quickly solved by DRAM manufacturers. Historically, once these these types of errors occur at a particular lithography node/DRAM density, they must be solved by the DRAM manufacturers, because even if a solution for a systemic problem could be asserted for particular markets (e.g., server, where use of advanced coding techniques, extra chips, etc. is acceptable), there will always be significant DRAM chip volume in single-piece applications (e.g., consumer devices, etc.) where complex architectural solutions aren't an option. The authors have identified a contemporary disturb sensitivity in DRAMs, but as non-technologists, our community can generally only observe, not correct, such problems.

Reviewer expertise (?)

4. Expert in area, with highest confidence in review.

Reviewer D -- Nothing New in RowHammer



Review #66D Modified Thursday 18 Jul 2013 12:51pm PDT

OVERALL MERIT (?)

1. Reject

REVIEWER EXPERTISE (?)

4. Expert in area, with highest confidence in review.

PAPER SUMMARY

The authors demonstrate that repeated activate-precharge operations on one wordline of a DRAM can disturb a few cells on adjacent wordlines. They showed that such a behavior can be caused for most DRAMs and all DRAMs of recent manufacture they tested.

PAPER STRENGTHS

DRAM errors are getting more likely with newer generations and it is necessary to investigate their cause and mitigation in computer systems, as such the paper addresses a subtopic of a relevant problem.

PAPER WEAKNESSES

The mechanism investigated by the authors is one of many well known disturb mechanisms. The paper does not discuss the root causes to sufficient depth and the importance of this mechanism compared to others. Overall the length of the sections restating known information is much too long in relation to new work.

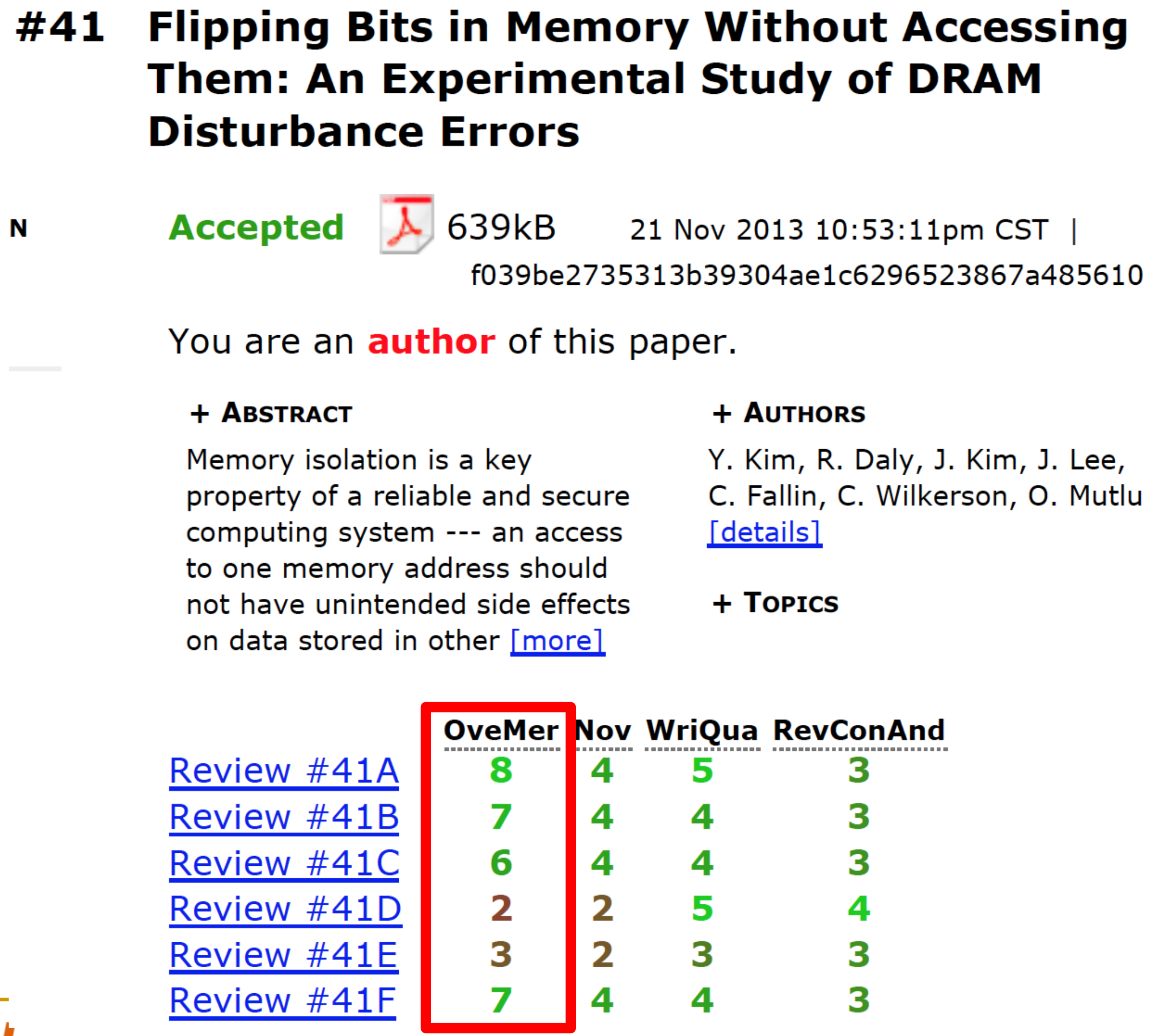
NOVELTY (?)

2. Insignificant novelty. Virtually all of the ideas are published or known.

WRITING QUALITY (?)

3. Adequate

ISCA 2014 Submission



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

N

Accepted 639kB 21 Nov 2013 10:53:11pm CST | f039be2735313b39304ae1c6296523867a485610

You are an **author** of this paper.

+ ABSTRACT

Memory isolation is a key property of a reliable and secure computing system --- an access to one memory address should not have unintended side effects on data stored in other [more]

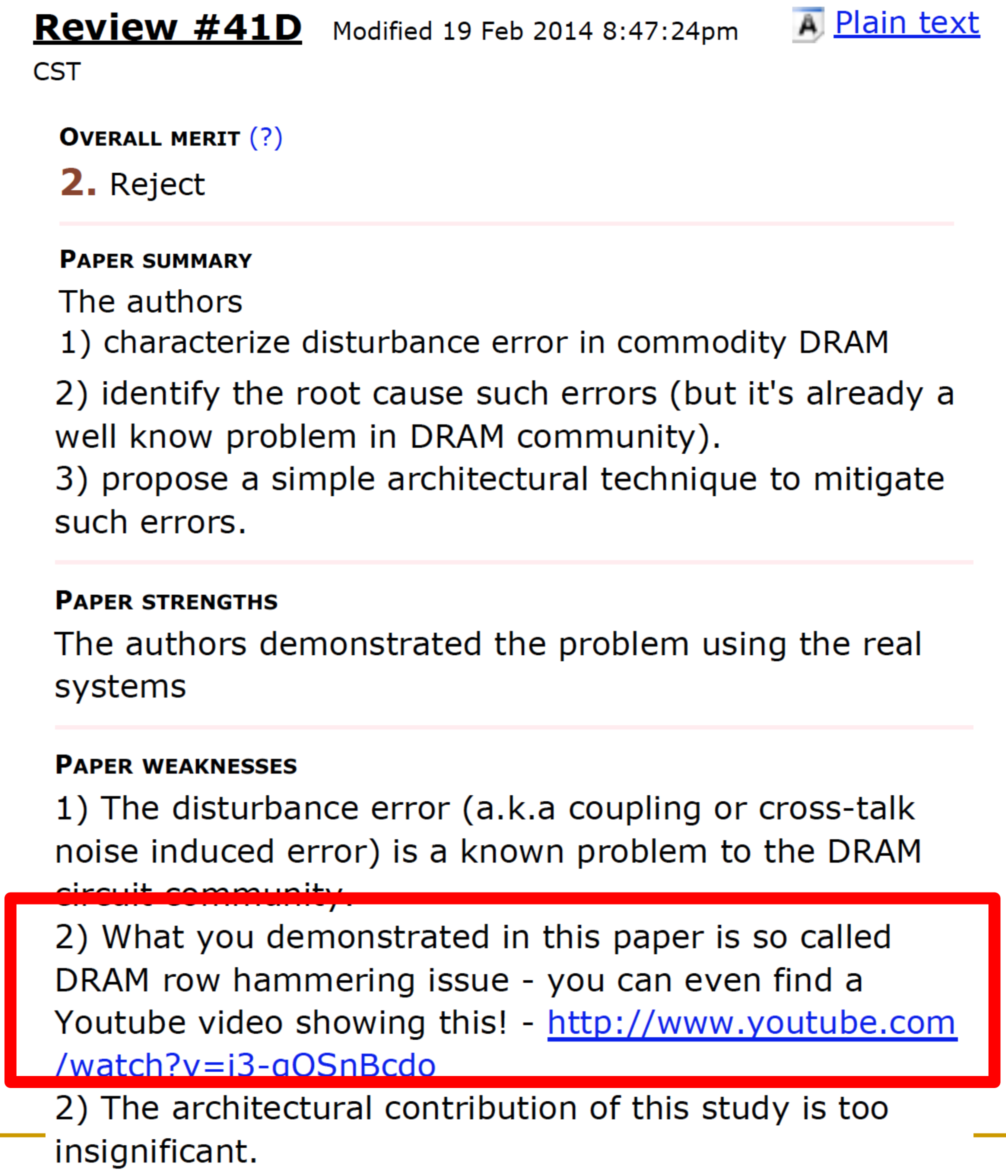
+ AUTHORS

Y. Kim, R. Daly, J. Kim, J. Lee, C. Fallin, C. Wilkerson, O. Mutlu [details]

+ TOPICS

	OveMer	Nov	WriQua	RevConAnd
Review #41A	8	4	5	3
Review #41B	7	4	4	3
Review #41C	6	4	4	3
Review #41D	2	2	5	4
Review #41E	3	2	3	3
Review #41F	7	4	4	3

Reviewer D



Review #41D Modified 19 Feb 2014 8:47:24pm CST

Plain text

Overall merit (?)

2. Reject

Paper summary

The authors

1) characterize disturbance error in commodity DRAM

2) identify the root cause such errors (but it's already a well know problem in DRAM community).

3) propose a simple architectural technique to mitigate such errors.

Paper strengths

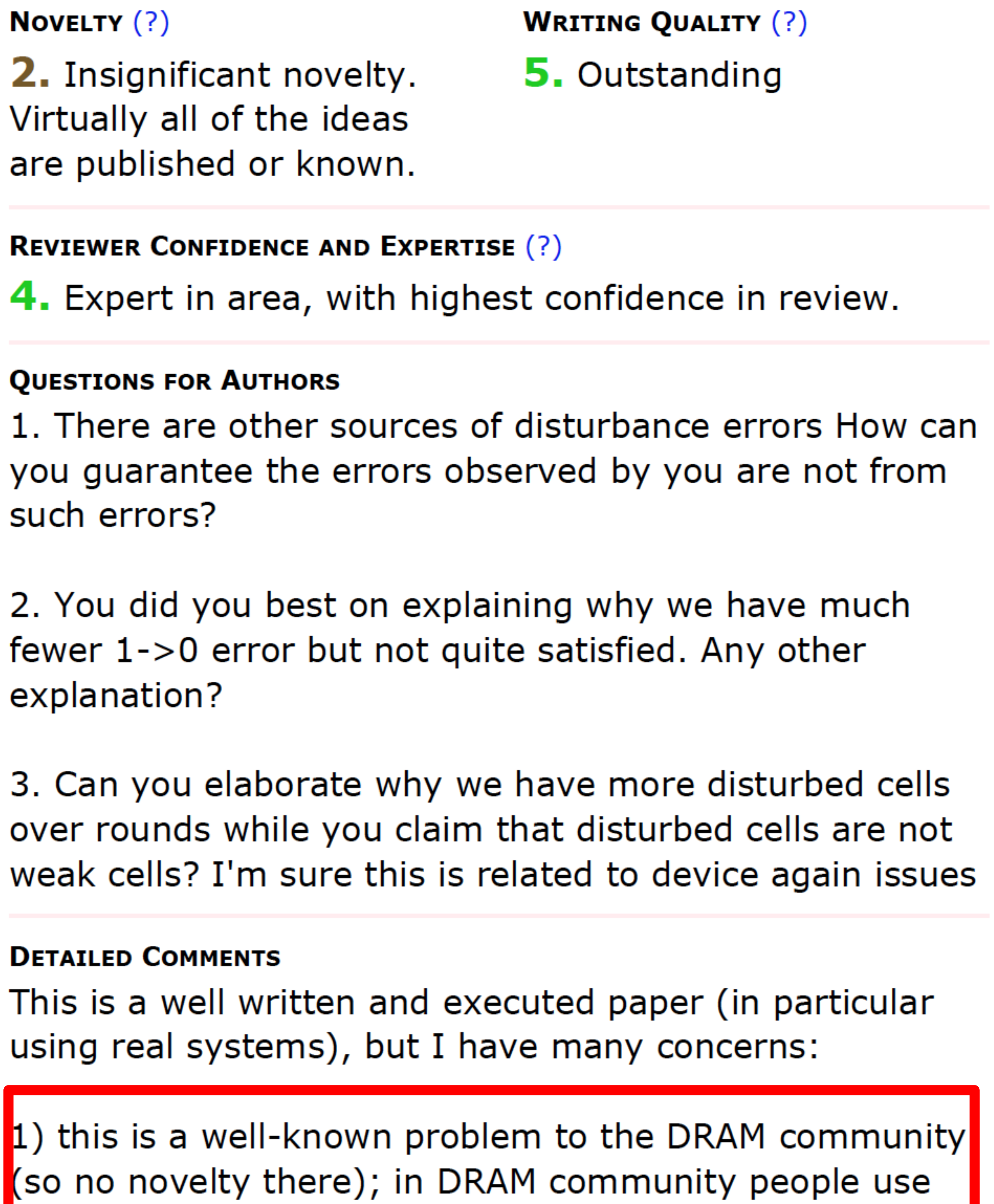
The authors demonstrated the problem using the real systems

Paper weaknesses

1) The disturbance error (a.k.a coupling or cross-talk noise induced error) is a known problem to the DRAM circuit community.

2) What you demonstrated in this paper is so called DRAM row hammering issue - you can even find a Youtube video showing this! - http://www.youtube.com/watch?v=i3-gOSnBcdo

2) The architectural contribution of this study is too insignificant.



Novelty (?)

2. Insignificant novelty. Virtually all of the ideas are published or known.

Writing Quality (?)

5. Outstanding

Reviewer Confidence and Expertise (?)

4. Expert in area, with highest confidence in review.

Questions for Authors

1. There are other sources of disturbance errors How can you guarantee the errors observed by you are not from such errors?

2. You did you best on explaining why we have much fewer 1->0 error but not quite satisfied. Any other explanation?

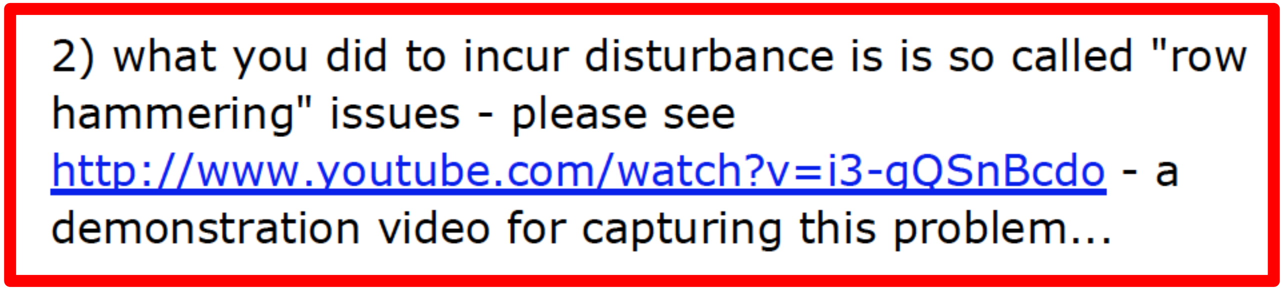
3. Can you elaborate why we have more disturbed cells over rounds while you claim that disturbed cells are not weak cells? I'm sure this is related to device again issues

Detailed Comments

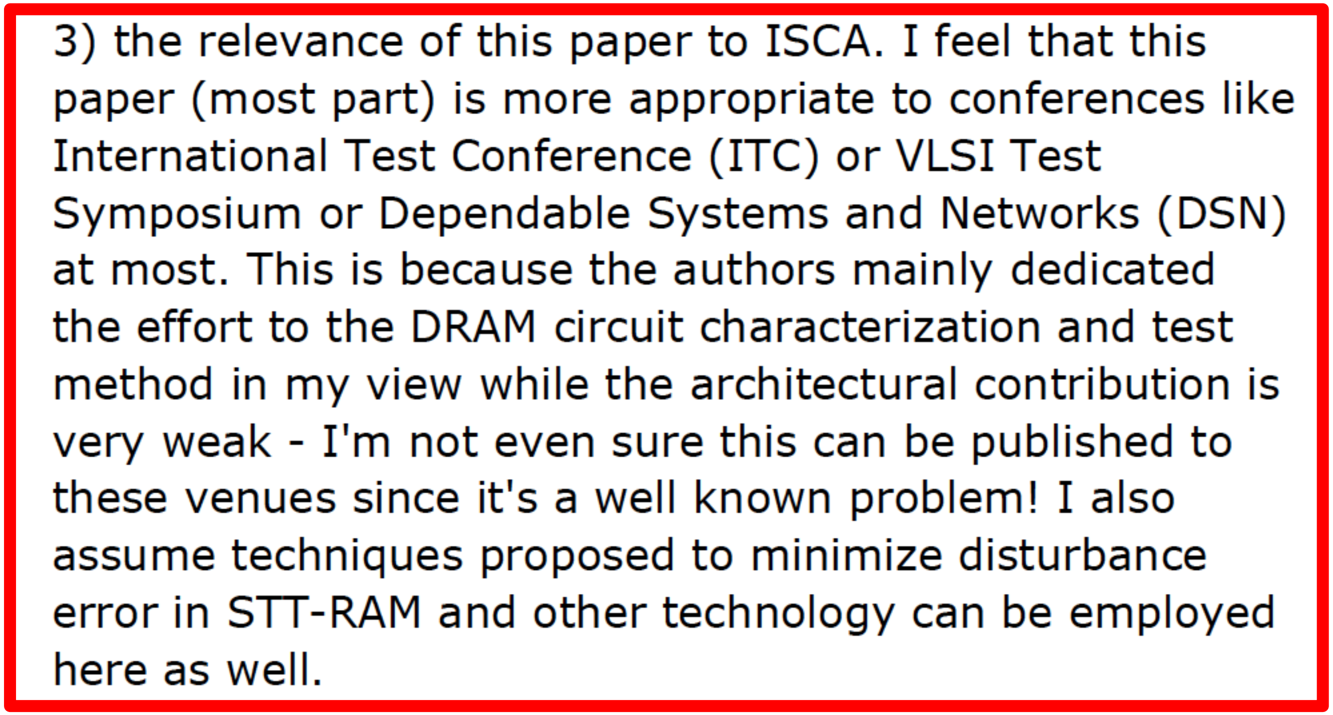
This is a well written and executed paper (in particular using real systems), but I have many concerns:

1) this is a well-known problem to the DRAM community (so no novelty there); in DRAM community people use

Reviewer D Continued…



2) what you did to incur disturbance is is so called "row hammering" issues - please see http://www.youtube.com/watch?v=i3-gQSnBcdo - a demonstration video for capturing this problem...



3) the relevance of this paper to ISCA. I feel that this paper (most part) is more appropriate to conferences like International Test Conference (ITC) or VLSI Test Symposium or Dependable Systems and Networks (DSN) at most. This is because the authors mainly dedicated the effort to the DRAM circuit characterization and test method in my view while the architectural contribution is very weak - I'm not even sure this can be published to these venues since it's a well known problem! I also assume techniques proposed to minimize disturbance error in STT-RAM and other technology can be employed here as well.

Rebuttal to Reviewer D

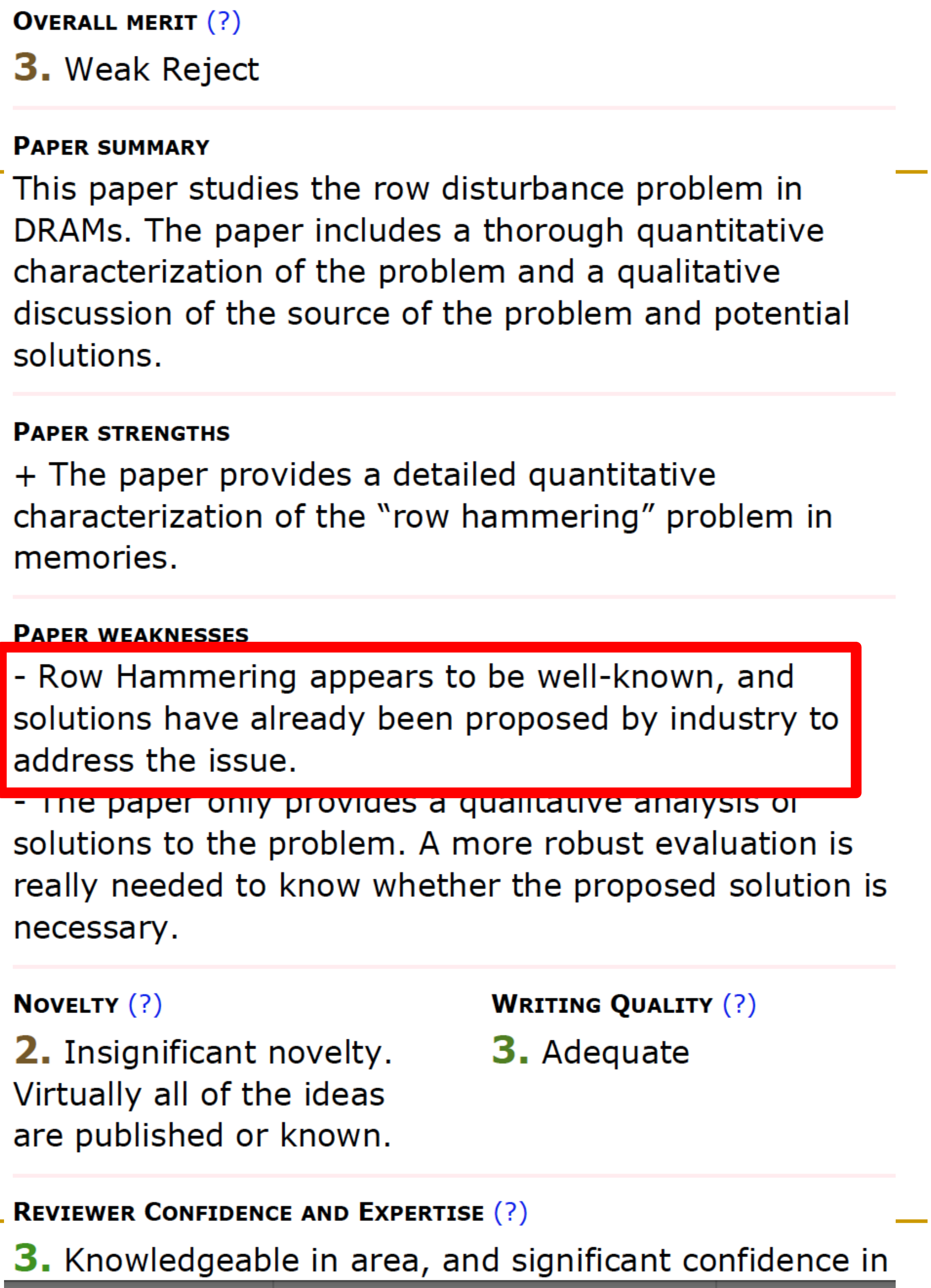
______Reviewer D (Comments)______

- 1. As we acknowledge in the paper, it is true that different types of DRAM coupling phenomena have been known to the DRAM circuits/testing community. However, there is a clear distinction between circuits/testing techniques confined to the *foundry* versus characterization/solution of a problem out in the *field*. The three citations (from 10+ years ago) do *not* demonstrate that disturbance errors exist in DIMMs sold then or now. They do *not* provide any real data (only simulated ones), let alone a large-scale characterization across many DIMMs from multiple manufacturers. They do *not* construct an attack on real systems, and they do *not* provide any solutions. Finally, our paper *already* references all three citations, or their more relevant equivalents. (The second/third citations provided by the reviewer are on bitline-coupling, whereas we cite works from the same authors on wordline-coupling [2, 3, 37].)

- 2. We were aware of the video from Teledyne (a test equipment company) and have *already* referenced slides from the same company [36]. In terms of their content regarding "row hammer", the video and the slides are identical: all they mention is that "aggressive row activations can corrupt adjacent rows". (They then advertise how their test equipment is able to capture a timestamped DRAM access trace, which can then be post-processed to identify when the number of activations exceeds a user-set threshold.) Both the video and slides do *not* say that this is a real problem affecting DIMMs on the market now. They do *not* provide any quantitative data, *nor* real-system demonstration, *nor* solution.

Reviewer E

Review #41E Modified 7 Feb 2014 11:08:04pm CST Plain text



Overall merit (?)

3. Weak Reject

Paper summary

This paper studies the row disturbance problem in DRAMs. The paper includes a thorough quantitative characterization of the problem and a qualitative discussion of the source of the problem and potential solutions.

Paper strengths

+ The paper provides a detailed quantitative characterization of the "row hammering" problem in memories.

Paper weaknesses

- Row Hammering appears to be well-known, and solutions have already been proposed by industry to address the issue.
- The paper only provides a qualitative analysis of solutions to the problem. A more robust evaluation is really needed to know whether the proposed solution is necessary.

Novelty (?)

2. Insignificant novelty. Virtually all of the ideas are published or known.

Writing Quality (?)

3. Adequate

Reviewer Confidence and Expertise (?)

3. Knowledgeable in area, and significant confidence in but there are numerous mentions of hammering in the literature, and clearly industry has studied this problem for many years. In particular, Intel has a patent application on a memory controller technique that addresses this exact problem, with priority date June 2012:

http://www.google.com/patents/WO2014004748A1?cl=en

The patent application details sound very similar to solution 6 in this paper, so a more thorough comparison with solution 7 seems mandatory.

My overall feeling is that while the reliability characterization is important and interesting, a better target audience for the characterization work would be in a testing/reliability venue. The most interesting part of this paper from the ISCA point of view are the proposed solutions, but all of these are discussed in a very qualitative manner. My preference would be to see a much shorter characterization section with a much stronger and quantitative evaluation and comparison of the proposed solutions.

Rebuttal to Reviewer

______Reviewer E (Comments)______

After our paper was submitted, two patents that had been filed by Intel were made public (one is mentioned by the reviewer [R1]). Together, the two patents describe what we posed as the *sixth* potential solution in our paper (Section 8). Essentially, the memory controller maintains a table of counters to track the number of activations to recently activated rows [R2]. And if one of the counters exceeds a certain threshold, the memory controller notifies the DRAM chips using a special command [R1]. The DRAM chips would then refresh an entire "region" of rows that includes both the aggressor and its victim(s) [R1]. For the patent [R1] to work, DRAM manufacturers must cooperate and implement this special command. (It is a convenient way of circumventing the opacity in the logical-physical mapping. If implemented, the same command can also be used for our *seventh* solution.) The limitation of this *sixth* solution is the storage overhead of the counters and the extra power required to associatively search through them on every activation (Section 8). That is why we believe our *seventh* solution to be more attractive. We will cite the patents and include a more concrete comparison between the two solutions.

Nevertheless, we were able to induce a large number of DRAM disturbance errors on all the latest Intel/AMD platforms that we tested: Haswell, Ivy Bridge, Sandy Bridge, and Piledriver. (At the time of submission, we had tested only Sandy Bridge.) Importantly, the patents do *not* provide quantitative characterization

nor real-system demonstration.

[R1] "Row Hammer Refresh Command." US20140006703 A1
[R2] "Row Hammer Condition Monitoring." US20140006704 A1

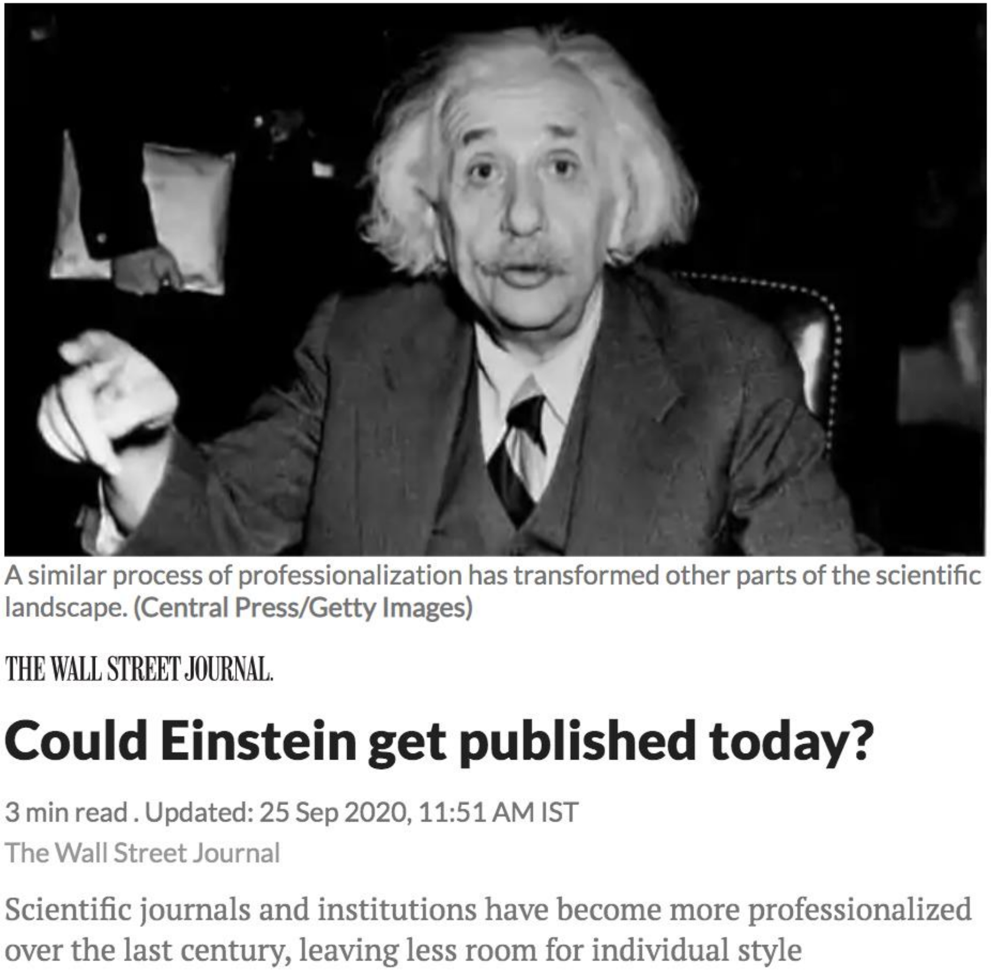
Suggestions to Reviewers

- Be fair; you do not know it all
- Be open-minded; you do not know it all
- Be accepting of diverse research methods: there is no single way of doing research
- Be constructive, not destructive
- Do not have double standards…

Do not block or delay scientific progress for non-reasons

A Fun Reading: Food for Thought

- https://www.livemint.com/science/news/could-einstein-get-published-today-11601014633853.html



A similar process of professionalization has transformed other parts of the scientific landscape. (Central Press/Getty Images)

THE WALL STREET JOURNAL.

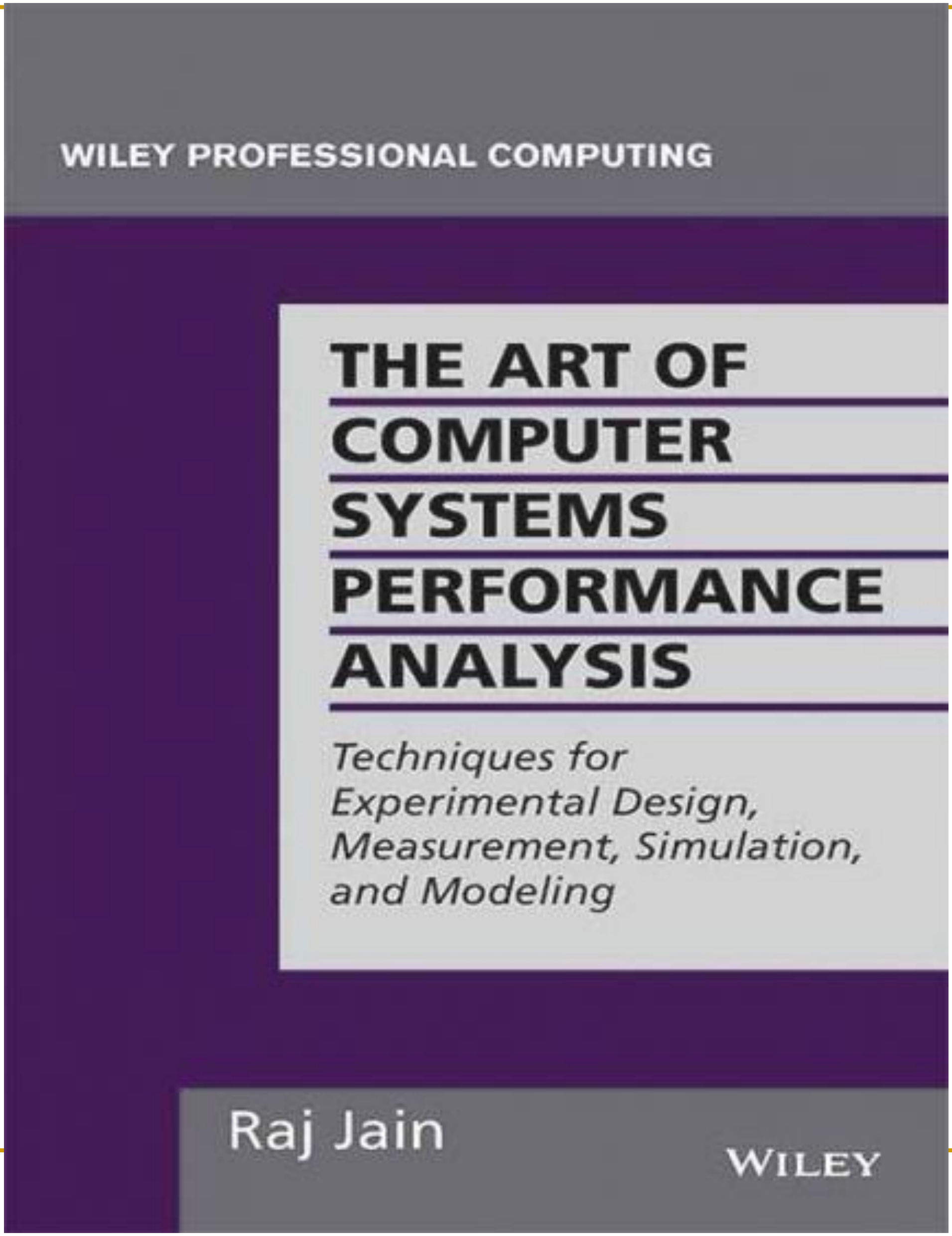
Could Einstein get published today?

3 min read . Updated: 25 Sep 2020, 11:51 AM IST

The Wall Street Journal

Scientific journals and institutions have become more professionalized over the last century, leaving less room for individual style

Aside: A Recommended Book



WILEY PROFESSIONAL COMPUTING

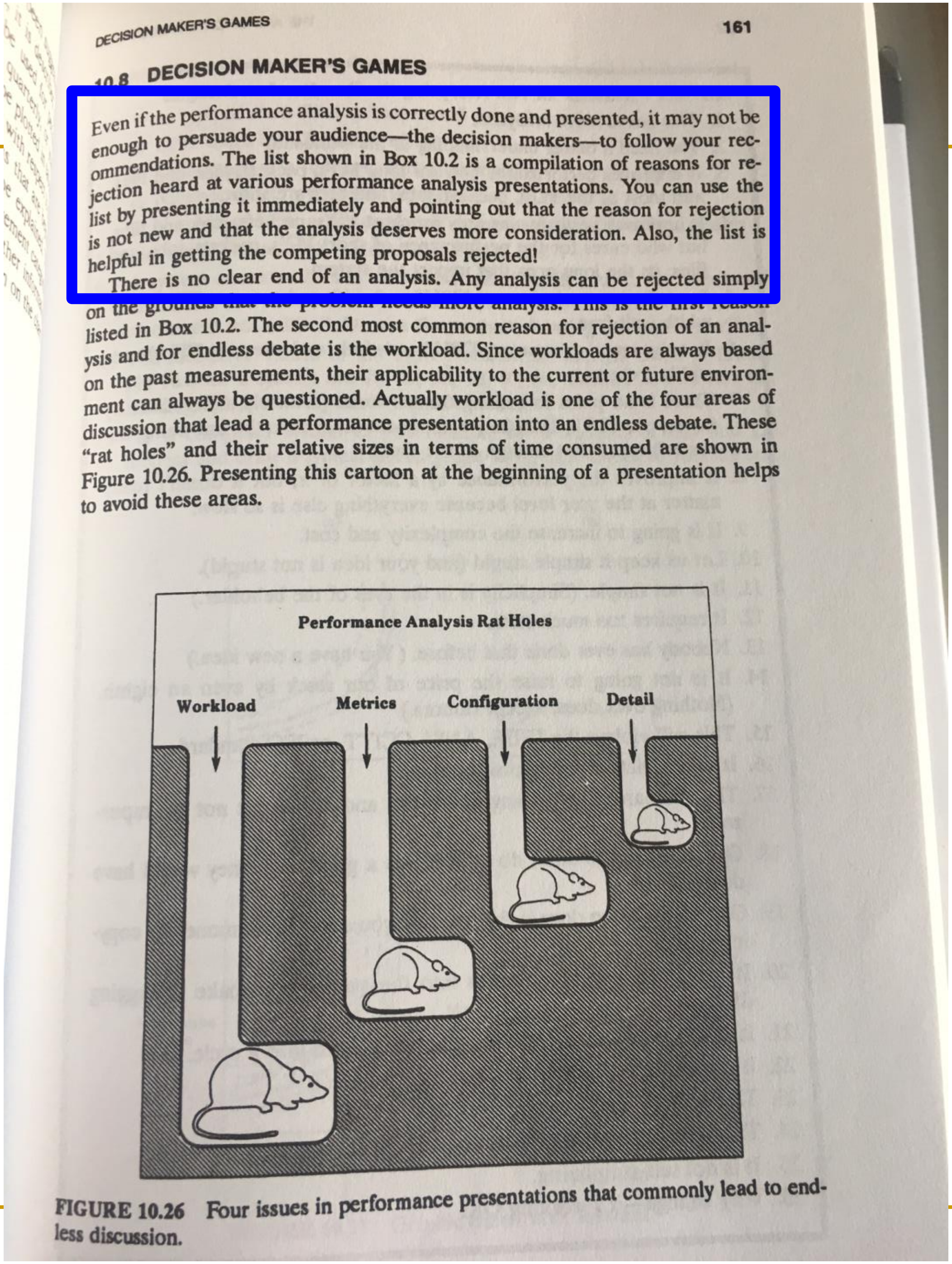
THE ART OF COMPUTER SYSTEMS PERFORMANCE ANALYSIS

Techniques for Experimental Design, Measurement, Simulation, and Modeling

Raj Jain

WILEY

Raj Jain, "The Art of Computer Systems Performance Analysis," Wiley, 1991.



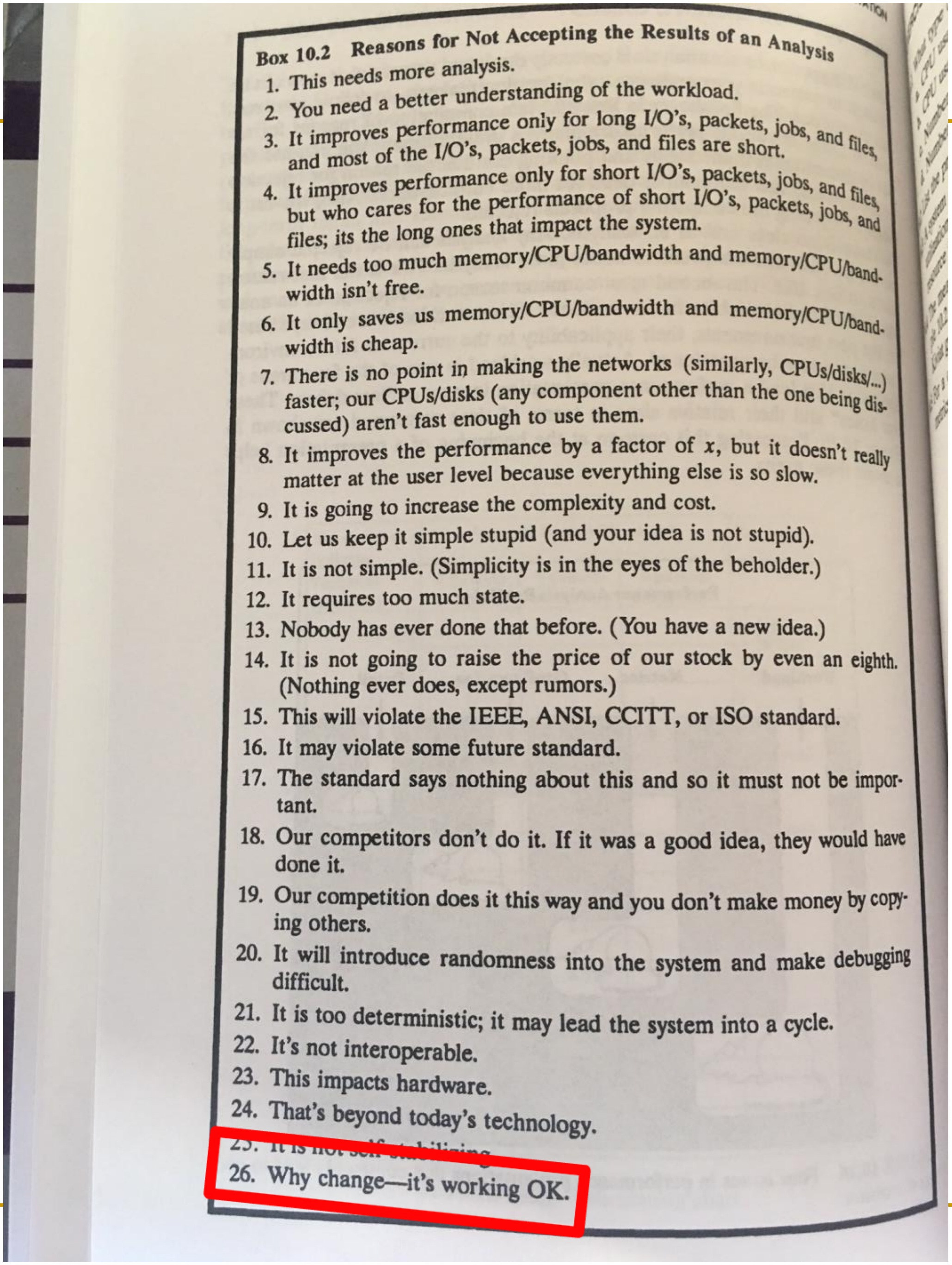
10.8 DECISION MAKER'S GAMES

Even if the performance analysis is correctly done and presented, it may not be enough to persuade your audience—the decision makers—to follow your recommendations. The list shown in Box 10.2 is a compilation of reasons for rejection heard at various performance analysis presentations. You can use the list by presenting it immediately and pointing out that the reason for rejection is not new and that the analysis deserves more consideration. Also, the list is helpful in getting the competing proposals rejected!

There is no clear end of an analysis. Any analysis can be rejected simply on the grounds that the problem needs more analysis. This is the first reason listed in Box 10.2. The second most common reason for rejection of an analysis and for endless debate is the workload. Since workloads are always based on the past measurements, their applicability to the current or future environment can always be questioned. Actually workload is one of the four areas of discussion that lead a performance presentation into an endless debate. These "rat holes" and their relative sizes in terms of time consumed are shown in Figure 10.26. Presenting this cartoon at the beginning of a presentation helps to avoid these areas.

FIGURE 10.26 Four issues in performance presentations that commonly lead to endless discussion.

Raj Jain, "The Art of Computer Systems Performance Analysis," Wiley, 1991.



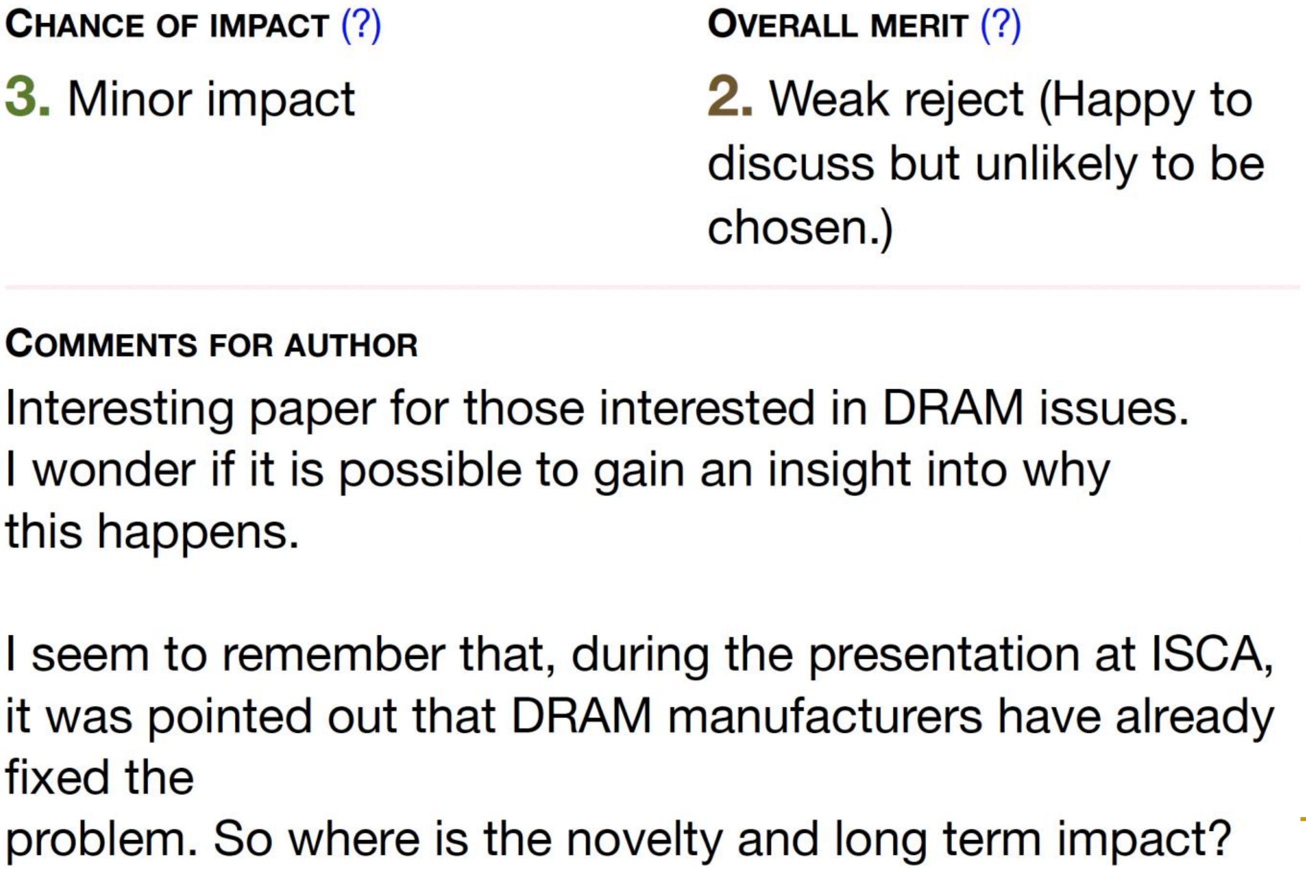
Box 10.2 Reasons for Not Accepting the Results of an Analysis

1. This needs more analysis.
2. You need a better understanding of the workload.
3. It improves performance only for long I/O's, packets, jobs, and files, and most of the I/O's, packets, jobs, and files are short.
4. It improves performance only for short I/O's, packets, jobs, and files, but who cares for the performance of short I/O's, packets, jobs, and files; its the long ones that impact the system.
5. It needs too much memory/CPU/bandwidth and memory/CPU/bandwidth isn't free.
6. It only saves us memory/CPU/bandwidth and memory/CPU/bandwidth is cheap.
7. There is no point in making the networks (similarly, CPUs/disks/...) faster; our CPUs/disks (any component other than the one being discussed) aren't fast enough to use them.
8. It improves the performance by a factor of *x*, but it doesn't really matter at the user level because everything else is so slow.
9. It is going to increase the complexity and cost.
10. Let us keep it simple stupid (and your idea is not stupid).
11. It is not simple. (Simplicity is in the eyes of the beholder.)
12. It requires too much state.
13. Nobody has ever done that before. (You have a new idea.)
14. It is not going to raise the price of our stock by even an eighth. (Nothing ever does, except rumors.)
15. This will violate the IEEE, ANSI, CCITT, or ISO standard.
16. It may violate some future standard.
17. The standard says nothing about this and so it must not be important.
18. Our competitors don't do it. If it was a good idea, they would have done it.
19. Our competition does it this way and you don't make money by copying others.
20. It will introduce randomness into the system and make debugging difficult.
21. It is too deterministic; it may lead the system into a cycle.
22. It's not interoperable.
23. This impacts hardware.
24. That's beyond today's technology.
25. It is not self-stabilizing.
26. Why change—it's working OK.

Raj Jain, "The Art of Computer Systems Performance Analysis," Wiley, 1991.

Reviews **After** the Paper Was Published

I poked around a bit and DRAM vendors have already solved this problem. DRAM row hammering appears to be a known problem.



Chance of impact (?)

3. Minor impact

Overall merit (?)

2. Weak reject (Happy to discuss but unlikely to be chosen.)

Comments for author

Interesting paper for those interested in DRAM issues.
I wonder if it is possible to gain an insight into why
this happens.

I seem to remember that, during the presentation at ISCA,
it was pointed out that DRAM manufacturers have already
fixed the
problem. So where is the novelty and long term impact?

Suggestions to Reviewers

- Be fair; you do not know it all
- Be open-minded; you do not know it all
- Be accepting of diverse research methods: there is no single way of doing research or writing papers
- Be constructive, not destructive
- Enable heterogeneity, but do **not** have double standards…

Do not block or delay scientific progress for non-reasons

Suggestion to Community

We Need to Fix the Reviewer Accountability Problem

Takeaway

Main Memory Needs

Intelligent Controllers

Takeaway

Research Community Needs

Accountable Reviewers

An Interview on Research and Education

- Computing Research and Education (@ ISCA 2019)
  - https://www.youtube.com/watch?v=8ffSEKZhmvo&list=PL5Q2soXY2Zi_4oP9LdL3cc8G6NIjD2Ydz

- Maurice Wilkes Award Speech (10 minutes)
  - https://www.youtube.com/watch?v=tcQ3zZ3JpuA&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=15

More Thoughts and Suggestions

- Onur Mutlu,
  "Some Reflections (on DRAM)"
  Award Speech for ACM SIGARCH Maurice Wilkes Award, at the ***ISCA*** *Awards Ceremony*, Phoenix, AZ, USA, 25 June 2019.
  [Slides (pptx) (pdf)]
  [Video of Award Acceptance Speech (Youtube; 10 minutes) (Youku; 13 minutes)]
  [Video of Interview after Award Acceptance (Youtube; 1 hour 6 minutes) (Youku; 1 hour 6 minutes)]
  [News Article on "ACM SIGARCH Maurice Wilkes Award goes to Prof. Onur Mutlu"]

- Onur Mutlu,
  "How to Build an Impactful Research Group"
  *57th Design Automation Conference Early Career Workshop (**DAC**)*, Virtual, 19 July 2020.
  [Slides (pptx) (pdf)]

https://www.youtube.com/onurmutlulectures

Suggestion to Researchers: Principle: Passion

Follow Your Passion

(Do not get derailed by naysayers)

Suggestion to Researchers: Principle: Resilience

Be Resilient

Principle: Learning and Scholarship

Focus on

learning and scholarship

Principle: Learning and Scholarship

The quality of your work defines your impact

Principle: Work Hard

Work Hard to

Enable Your Passion

Principle: Good Mindset, Goals & Focus

You can make a good impact on the world

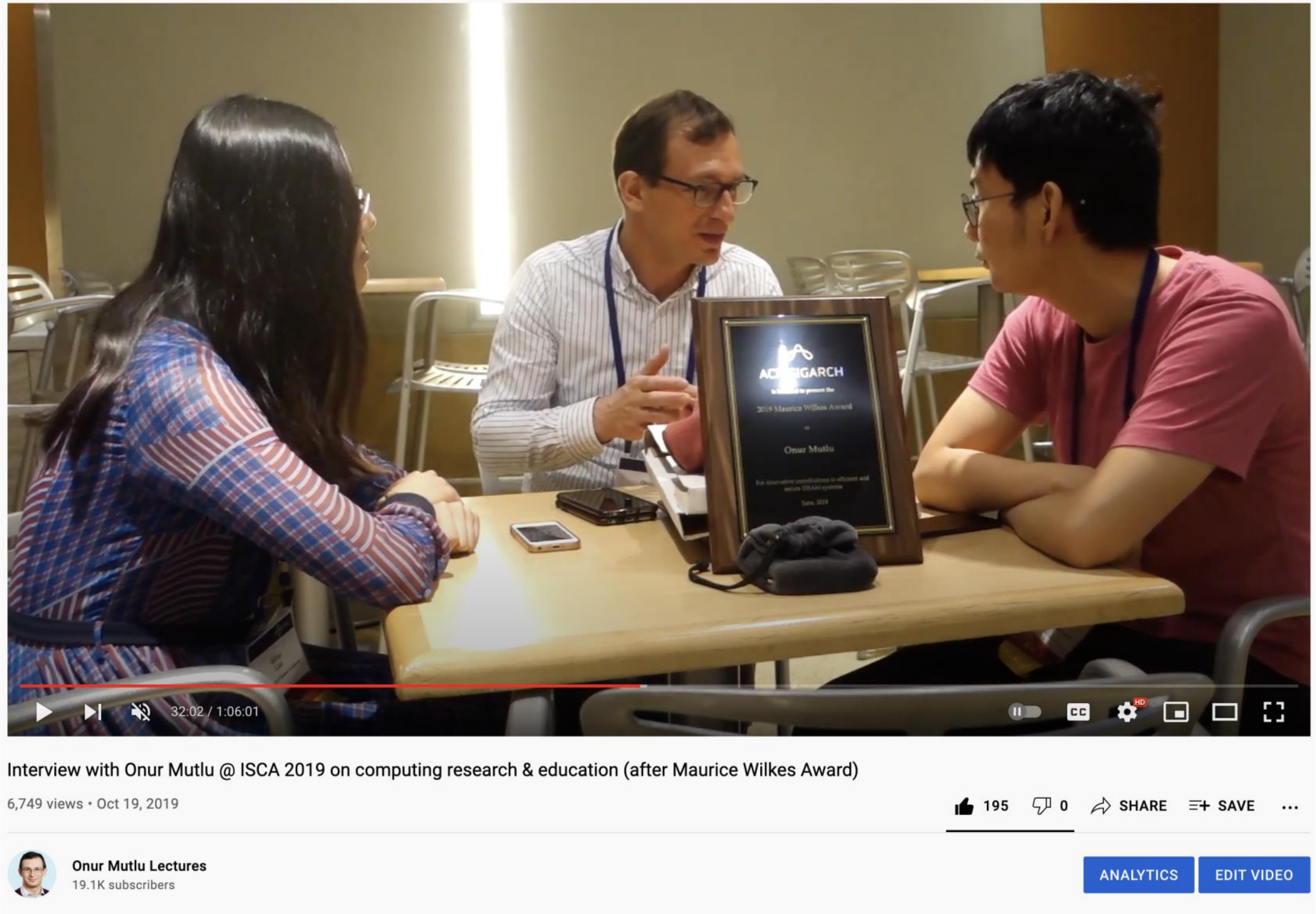
Recommended Interview on Research & Education

- **Computing Research and Education** (@ ISCA 2019)
  - https://www.youtube.com/watch?v=8ffSEKZhmvo&list=PL5Q2soXY2Zi_4oP9LdL3cc8G6NIjD2Ydz

- **Maurice Wilkes Award Speech** (10 minutes)
  - https://www.youtube.com/watch?v=tcQ3zZ3JpuA&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=15

- Onur Mutlu,
  "Some Reflections (on DRAM)"
  Award Speech for ACM SIGARCH Maurice Wilkes Award, at the ***ISCA*** *Awards Ceremony*,
  Phoenix, AZ, USA, 25 June 2019.
  [Slides (pptx) (pdf)]
  [Video of Award Acceptance Speech (Youtube; 10 minutes) (Youku; 13 minutes)]
  [Video of Interview after Award Acceptance (Youtube; 1 hour 6 minutes) (Youku; 1 hour 6 minutes)]
  [News Article on "ACM SIGARCH Maurice Wilkes Award goes to Prof. Onur Mutlu"]

Recommended Interview



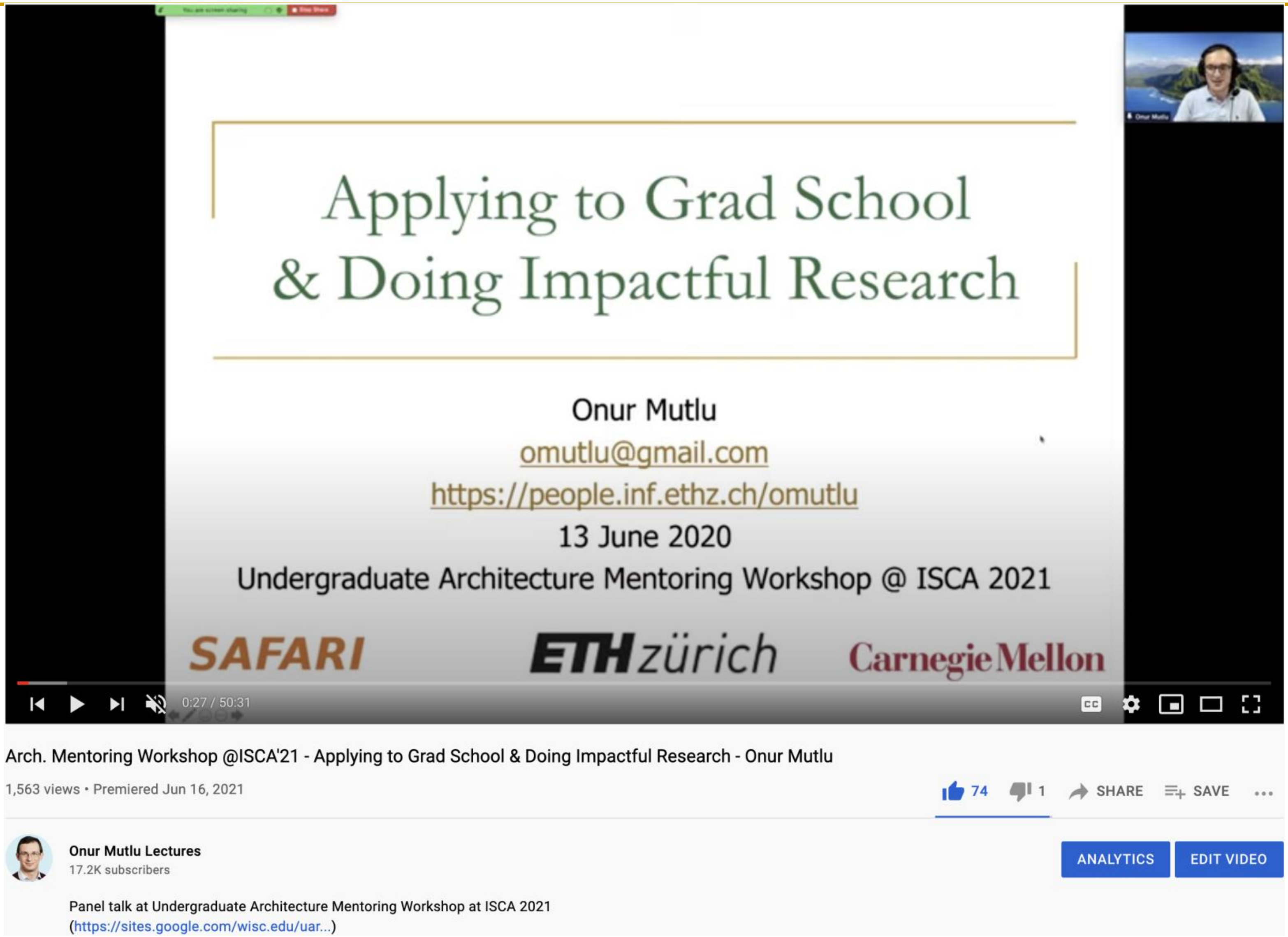
Interview with Onur Mutlu @ ISCA 2019 on computing research & education (after Maurice Wilkes Award)

6,749 views • Oct 19, 2019

Onur Mutlu Lectures
19.1K subscribers

https://www.youtube.com/watch?v=8ffSEKZhmvo

A Talk on Impactful Research & Education



Applying to Grad School & Doing Impactful Research

Onur Mutlu

omutlu@gmail.com

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

13 June 2020

Undergraduate Architecture Mentoring Workshop @ ISCA 2021

Arch. Mentoring Workshop @ISCA'21 - Applying to Grad School & Doing Impactful Research - Onur Mutlu

1,563 views • Premiered Jun 16, 2021

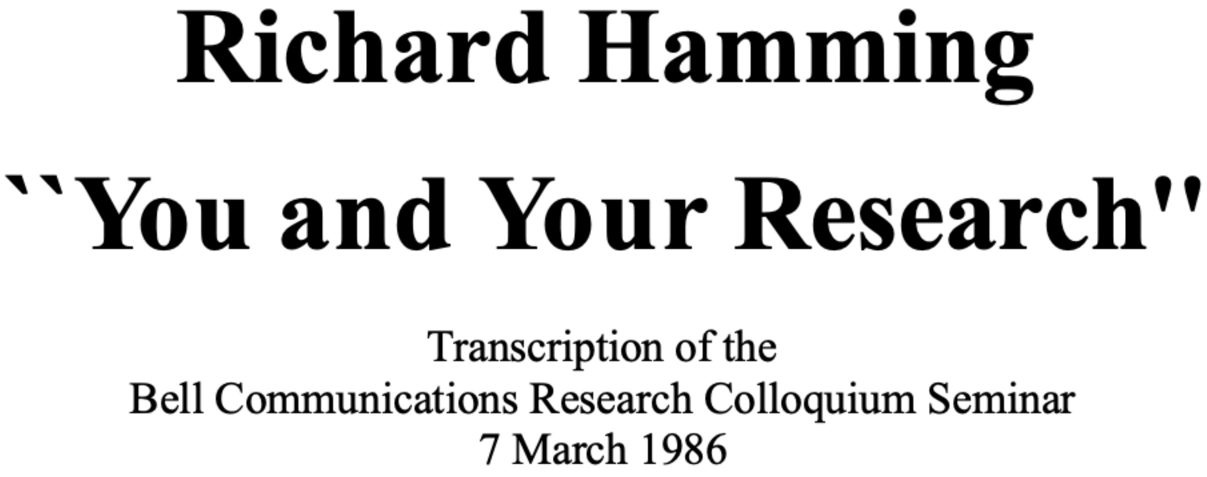
Onur Mutlu Lectures

17.2K subscribers

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

https://www.youtube.com/watch?v=83tlorht7Mc&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=54

Suggested Reading



Richard Hamming

``You and Your Research''

Transcription of the
Bell Communications Research Colloquium Seminar
7 March 1986

https://safari.ethz.ch/architecture/fall2021/lib/exe/fetch.php?media=youandyourresearch.pdf

Revisiting RowHammer

RowHammer in 2020 (I)

- 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*

Revisiting RowHammer

An Experimental Analysis of Modern Devices and Mitigation Techniques

Jeremie S. Kim Minesh Patel
A. Giray Yağlıkçı Hasan Hassan
Roknoddin Azizi Lois Orosa Onur Mutlu

SAFARI

ETH Zürich

Carnegie Mellon

Executive Summary

- **Motivation**: Denser DRAM chips are more vulnerable to RowHammer but no characterization-based study demonstrates how vulnerability scales
- **Problem**: Unclear if existing mitigation mechanisms will remain viable for future DRAM chips that are likely to be more vulnerable to RowHammer
- **Goal**:
  1. Experimentally demonstrate how vulnerable modern DRAM chips are to RowHammer and study how this vulnerability will scale going forward
  2. Study viability of existing mitigation mechanisms on more vulnerable chips
- **Experimental Study:** First rigorous RowHammer characterization study across a broad range of DRAM chips
  - 1580 chips of different DRAM {types, technology node generations, manufacturers}
  - We find that RowHammer vulnerability worsens in newer chips
- **RowHammer Mitigation Mechanism Study:** How five state-of-the-art mechanisms are affected by worsening RowHammer vulnerability
  - Reasonable performance loss (8% on average) on modern DRAM chips
  - Scale poorly to more vulnerable DRAM chips (e.g., 80% performance loss)
- **Conclusion:** it is critical to research more effective solutions to RowHammer for future DRAM chips that will likely be even more vulnerable to RowHammer

Motivation

- Denser DRAM chips are **more vulnerable** to RowHammer

- Three prior works **[Kim+, ISCA'14], [Park+, MR'16], [Park+, MR'16]**, **over the last six years** provide RowHammer characterization data on real DRAM

- However, there is **no comprehensive experimental study** that demonstrates **how vulnerability scales** across DRAM types and technology node generations

- It is **unclear whether current mitigation mechanisms will remain viable** for future DRAM chips that are likely to be more vulnerable to RowHammer

Goal

1. **Experimentally demonstrate** how vulnerable modern DRAM chips are to RowHammer and **predict how this vulnerability will scale** going forward

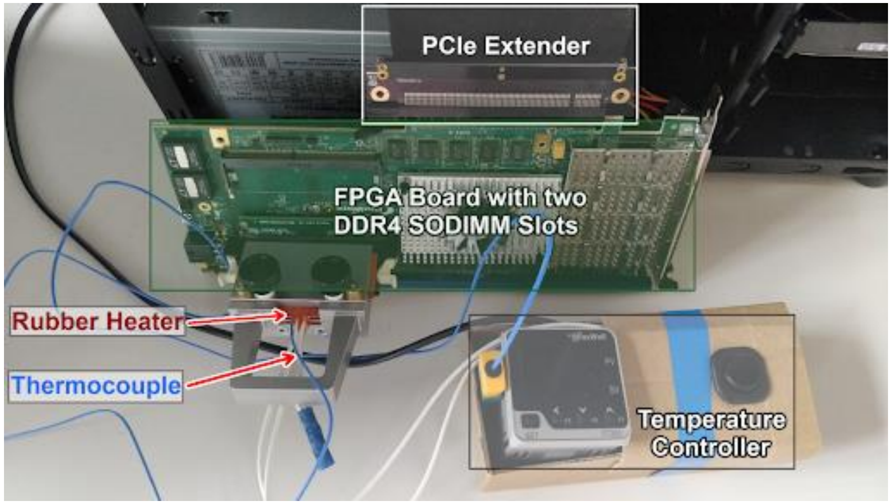
2. Examine the viability of current mitigation mechanisms on **more vulnerable chips**

DRAM Testing Infrastructures

Three separate testing infrastructures

1. **DDR3:** FPGA-based SoftMC [Hassan+, HPCA'17] (Xilinx ML605)
2. **DDR4:** FPGA-based SoftMC [Hassan+, HPCA'17] (Xilinx Virtex UltraScale 95)
3. **LPDDR4:** In-house testing hardware for LPDDR4 chips

All provide fine-grained control over DRAM commands, timing parameters and temperature



DDR4 DRAM testing infrastructure

DRAM Chips Tested

DRAM type-node	Number of Chips (Modules) Tested			
	Mfr. A	*Mfr. B*	*Mfr. C*	*Total*
DDR3-old	56 (10)	88 (11)	28 (7)	**172 (28)**
DDR3-new	80 (10)	52 (9)	104 (13)	**236 (32)**
DDR4-old	112 (16)	24 (3)	128 (18)	**264 (37)**
DDR4-new	264 (43)	16 (2)	108 (28)	**388 (73)**
LPDDR4-1x	12 (3)	180 (45)	N/A	**192 (48)**
LPDDR4-1y	184 (46)	N/A	144 (36)	**328 (82)**

1580 total DRAM chips tested from **300** DRAM modules

- **Three** major DRAM manufacturers {A, B, C}
- **Three** DRAM *types* or *standards* {DDR3, DDR4, LPDDR4}
  - LPDDR4 chips we test implement on-die ECC
- **Two** technology nodes per DRAM type {old/new, 1x/1y}
  - Categorized based on manufacturing date, datasheet publication date, purchase date, and characterization results

Type-node: configuration describing a chip's type and technology node generation: **DDR3-old/new, DDR4-old/new, LPDDR4-1x/1y**

Effective RowHammer Characterization

To characterize our DRAM chips at **worst-case** conditions, we:

1. Prevent sources of interference during core test loop

- We disable:
  - **DRAM refresh**: to avoid refreshing victim row
  - **DRAM calibration events**: to minimize variation in test timing
  - **RowHammer mitigation mechanisms**: to observe circuit-level effects
- Test for **less than refresh window (32ms)** to avoid retention failures

2. Worst-case access sequence

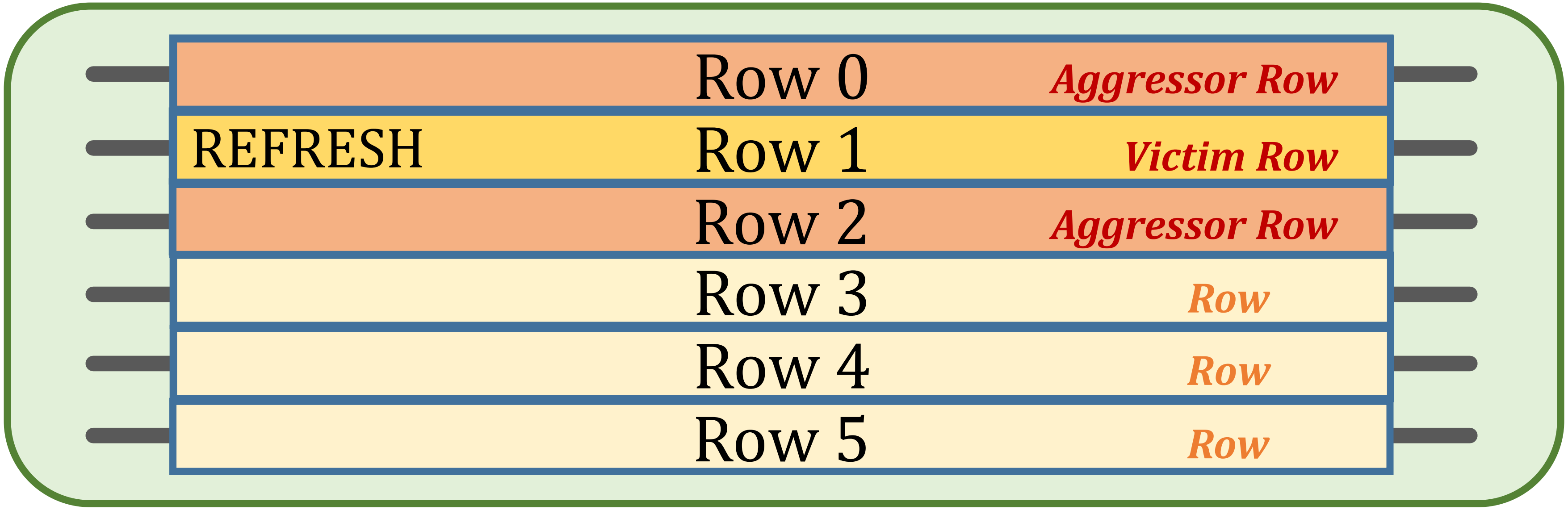
- We use **worst-case** access sequence based on prior works' observations

- For each row, **repeatedly access the two directly physically-adjacent rows as fast as possible**

SAFARI

[More details in the paper]

Testing Methodology



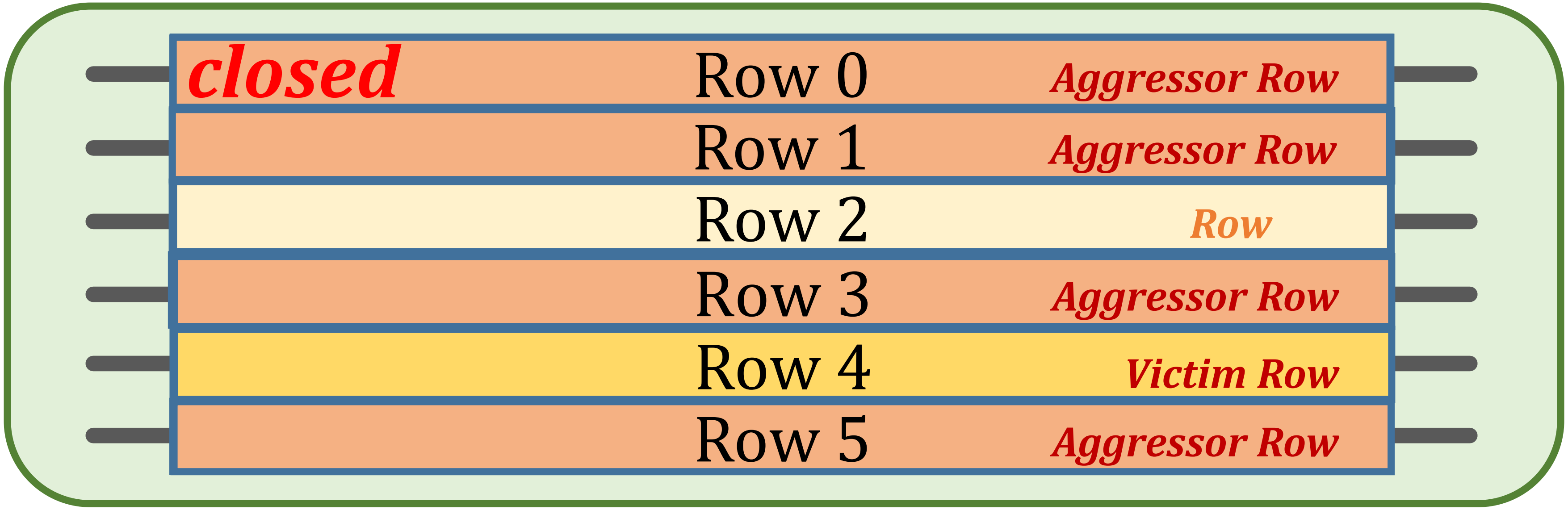
```
DRAM_RowHammer_Characterization():
    foreach row in DRAM:
        set victim_row to row
        set aggressor_row1 to victim_row − 1
        set aggressor_row2 to victim_row + 1
        Disable DRAM refresh
        Refresh victim_row
        for n = 1 → HC: // core test loop
            activate aggressor_row1
            activate aggressor_row2
        Enable DRAM refresh
        Record RowHammer bit flips to storage
        Restore bit flips to original values
```

Disable refresh to **prevent interruptions** in the core loop of our test **from refresh operations**

Induce RowHammer bit flips on a **fully charged row**

SAFARI

Testing Methodology



```
DRAM_RowHammer_Characterization():
    foreach row in DRAM:
        set victim_row to row
        set aggressor_row1 to victim_row – 1
        set aggressor_row2 to victim_row + 1
        Disable DRAM refresh
        Refresh victim_row
        for $n = 1 \rightarrow HC$: // core test loop
            activate aggressor_row1
            activate aggressor_row2
        Enable DRAM refresh
        Record RowHammer bit flips to storage
        Restore bit flips to original values
```

Disable refresh to **prevent interruptions** in the core loop of our test **from refresh operations**

Induce RowHammer bit flips on a **fully charged row**

Core test loop where we alternate accesses to adjacent rows

1 Hammer (HC) = two accesses

Prevent further retention failures

Record bit flips for analysis

Key Takeaways from 1580 Chips

- Chips of newer DRAM technology nodes are **more vulnerable** to RowHammer

- There are chips today whose weakest cells fail after **only 4800 hammers**

- Chips of newer DRAM technology nodes can exhibit RowHammer bit flips 1) in **more rows** and 2) **farther away** from the victim row.

1. RowHammer Vulnerability

Q. Can we induce RowHammer bit flips in all of our DRAM chips?

All chips are vulnerable, except many DDR3 chips

- A total of 1320 out of all 1580 chips **(84%)** are vulnerable
- Within **DDR3-old** chips, **only 12%** of chips (24/204) are vulnerable
- Within **DDR3-new** chips, **65%** of chips (148/228) are vulnerable

Newer DRAM chips are more vulnerable to RowHammer

2. Data Pattern Dependence

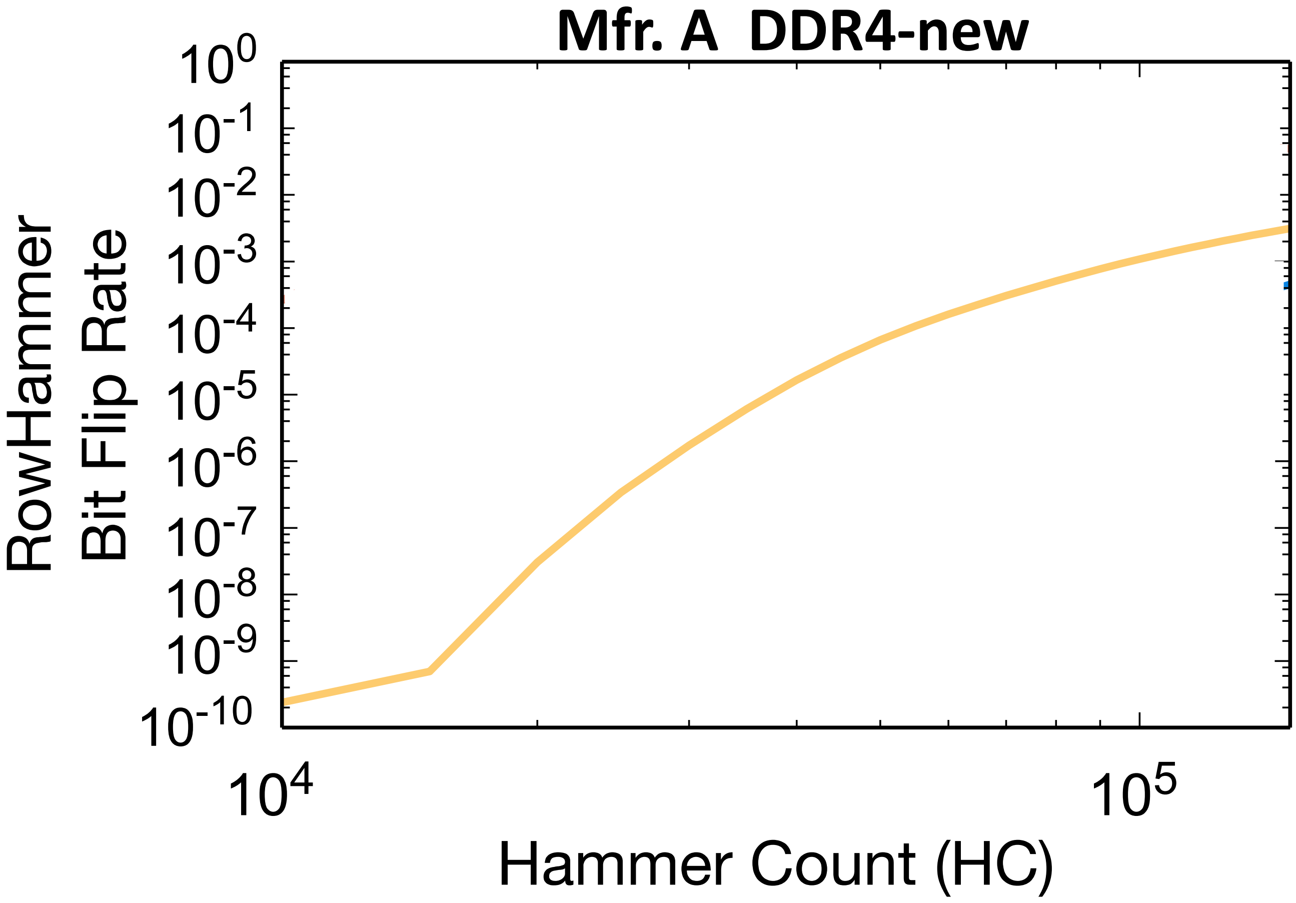
Q. Are some data patterns more effective in inducing RowHammer bit flips?

- We test **several data patterns** typically examined in prior work to identify the worst-case data pattern
- The worst-case data pattern is **consistent across chips** of the same manufacturer and DRAM type-node configuration
- We use the **worst-case data pattern** per DRAM chip to characterize each chip at **worst-case conditions** and **minimize the extensive testing time**

[More detail and figures in paper]

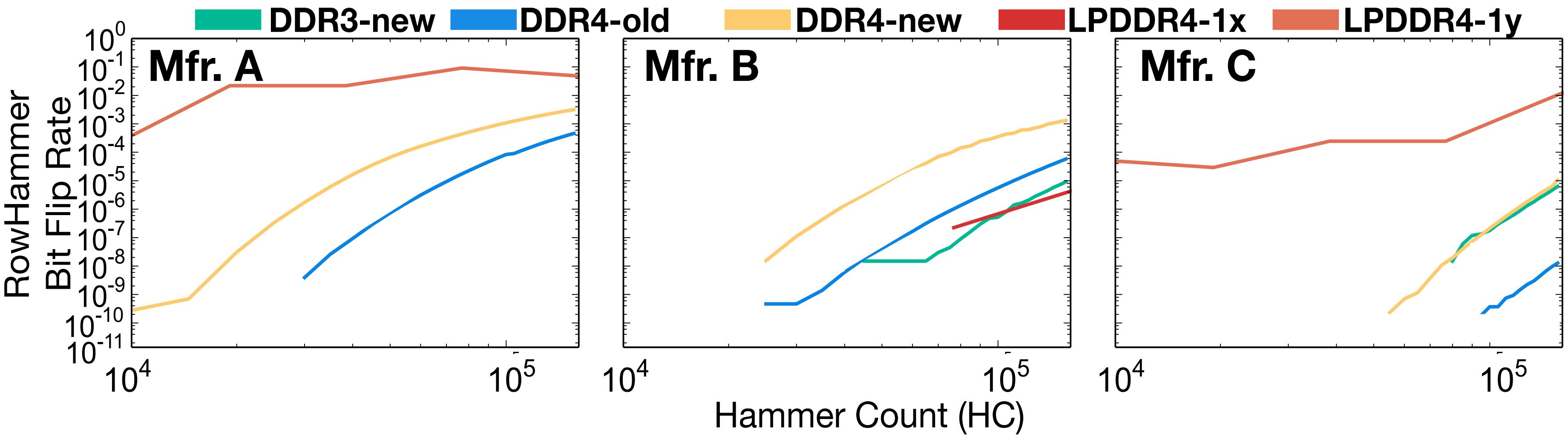
3. Hammer Count (HC) Effects

Q. How does the Hammer Count affect the number of bit flips induced?



Hammer Count = 2 Accesses, one to each adjacent row of victim

3. Hammer Count (HC) Effects

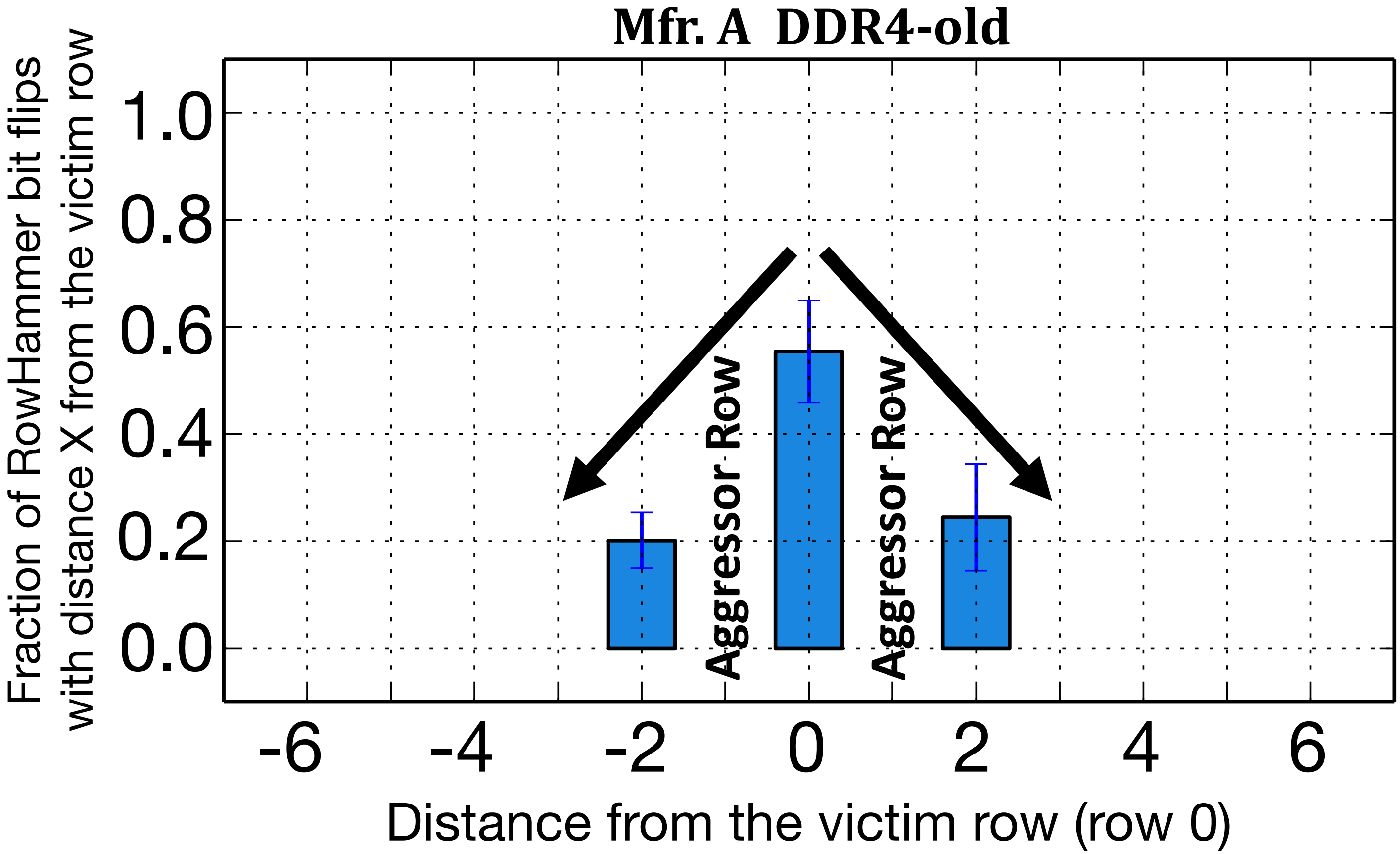


RowHammer bit flip rates **increase**
when going **from old to new** DDR4 technology node generations

RowHammer bit flip rates (i.e., RowHammer vulnerability) increase with technology node generation

4. Spatial Effects: Row Distance

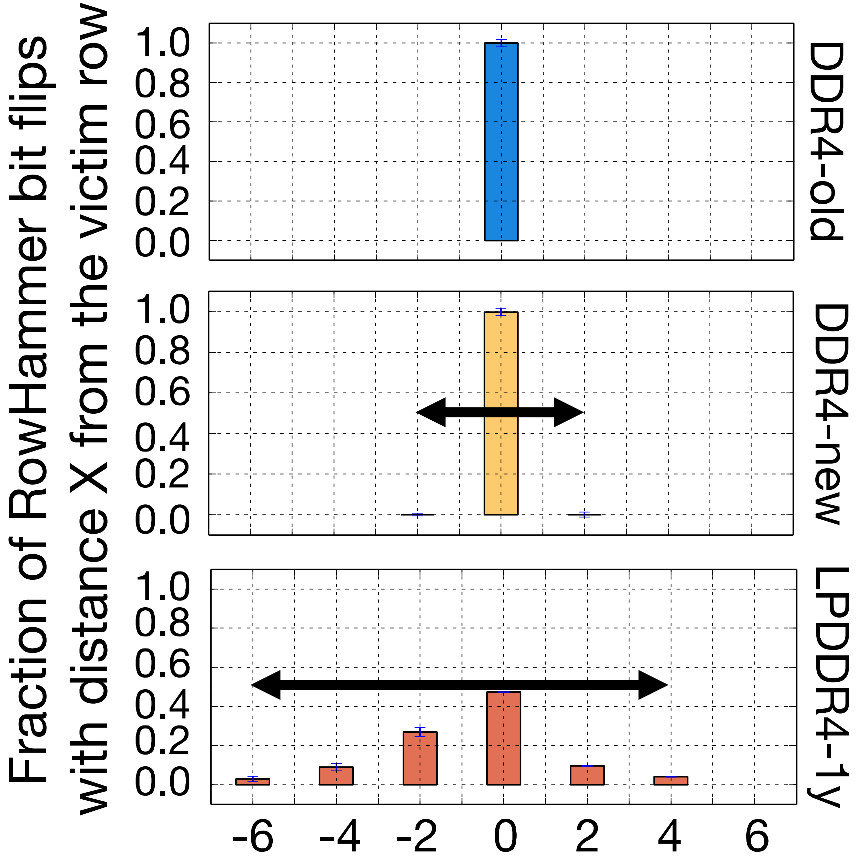
Q. Where do RowHammer bit flips occur relative to aggressor rows?



The number of RowHammer bit flips that occur in a given row decreases as the distance from the **victim row (row 0)** increases.

4. Spatial Effects: Row Distance

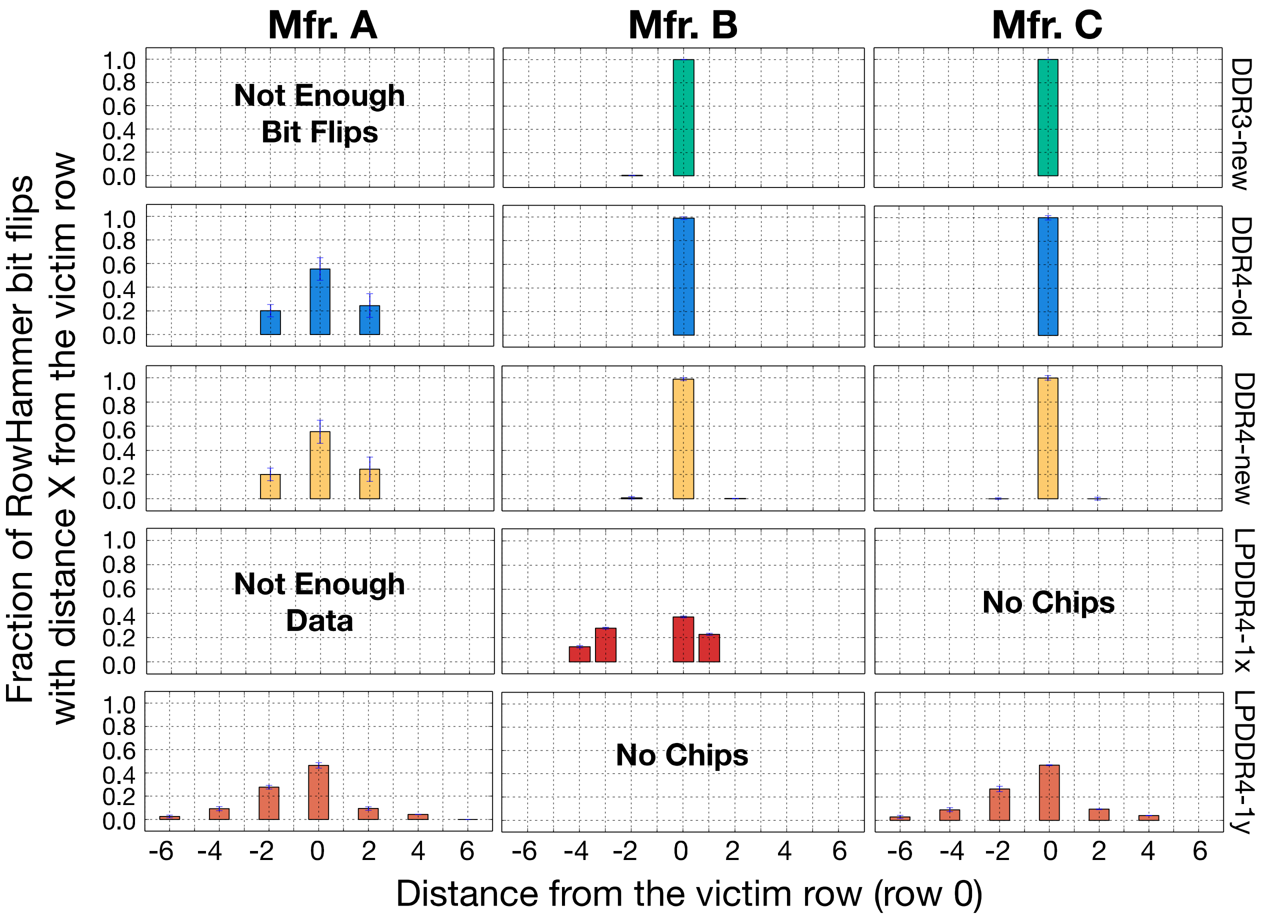
We normalize data by inducing a bit flip rate of 10^{-6} in each chip



Chips of newer DRAM technology nodes can exhibit RowHammer bit flips 1) in **more rows** and 2) **farther away** from the victim row.

4. Spatial Effects: Row Distance

We plot this data for each DRAM type-node configuration per manufacturer



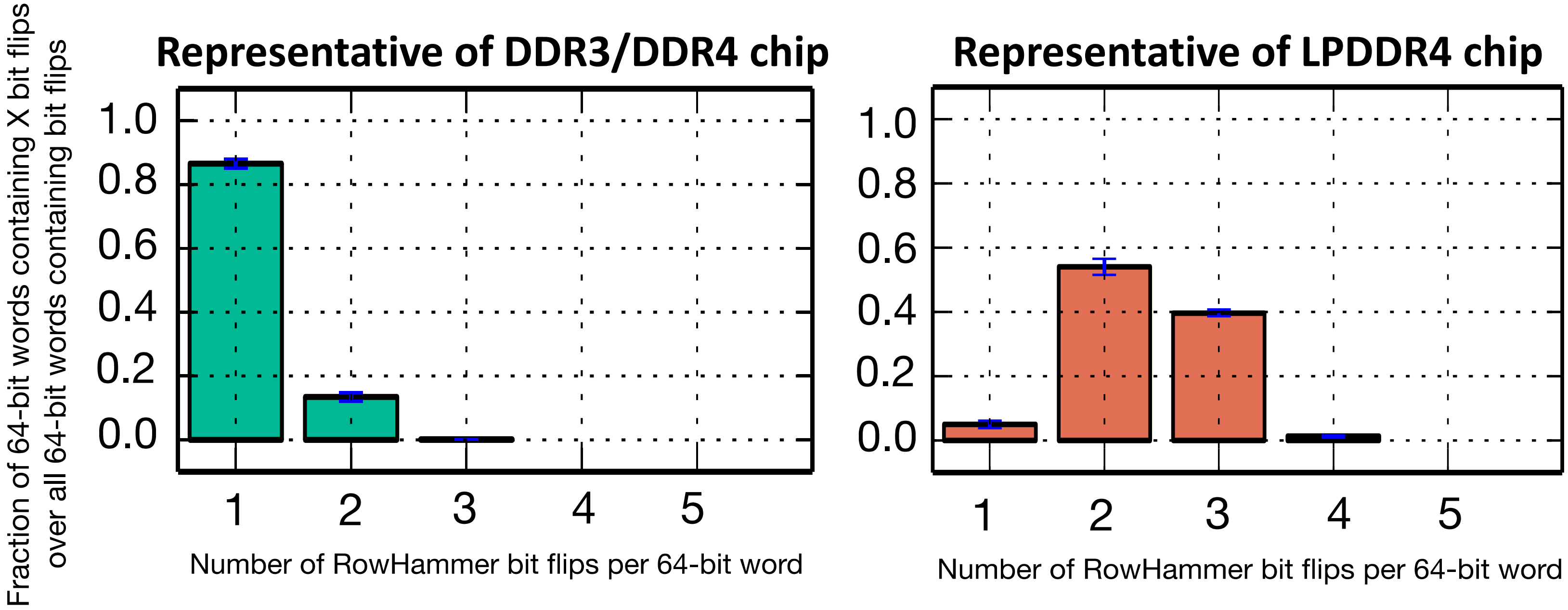
[More analysis in the paper]

SAFARI

4. Spatial Distribution of Bit Flips

Q. How are RowHammer bit flips spatially distributed across a chip?

We normalize data by inducing a bit flip rate of **10^{-6}** in each chip



Representative of DDR3/DDR4 chip

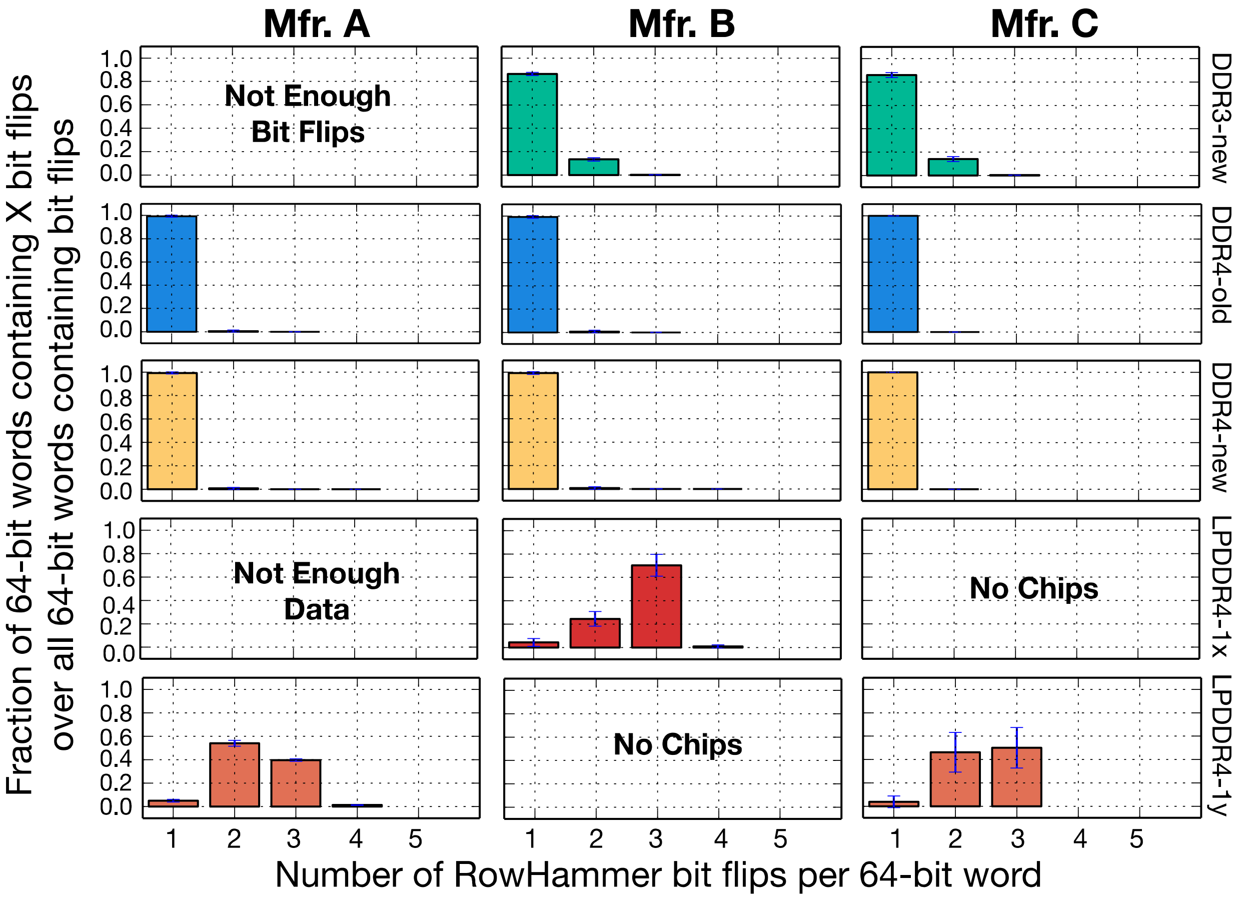
Representative of LPDDR4 chip

The distribution of RowHammer bit flip density per word
changes significantly in LPDDR4 chips from other DRAM types

At a bit flip rate of 10^{-6}, a 64-bit word can contain up to **4 bit flips**.
Even at this very low bit flip rate, a **very strong ECC** is required

4. Spatial Distribution of Bit Flips

We plot this data for each DRAM type-node configuration per manufacturer

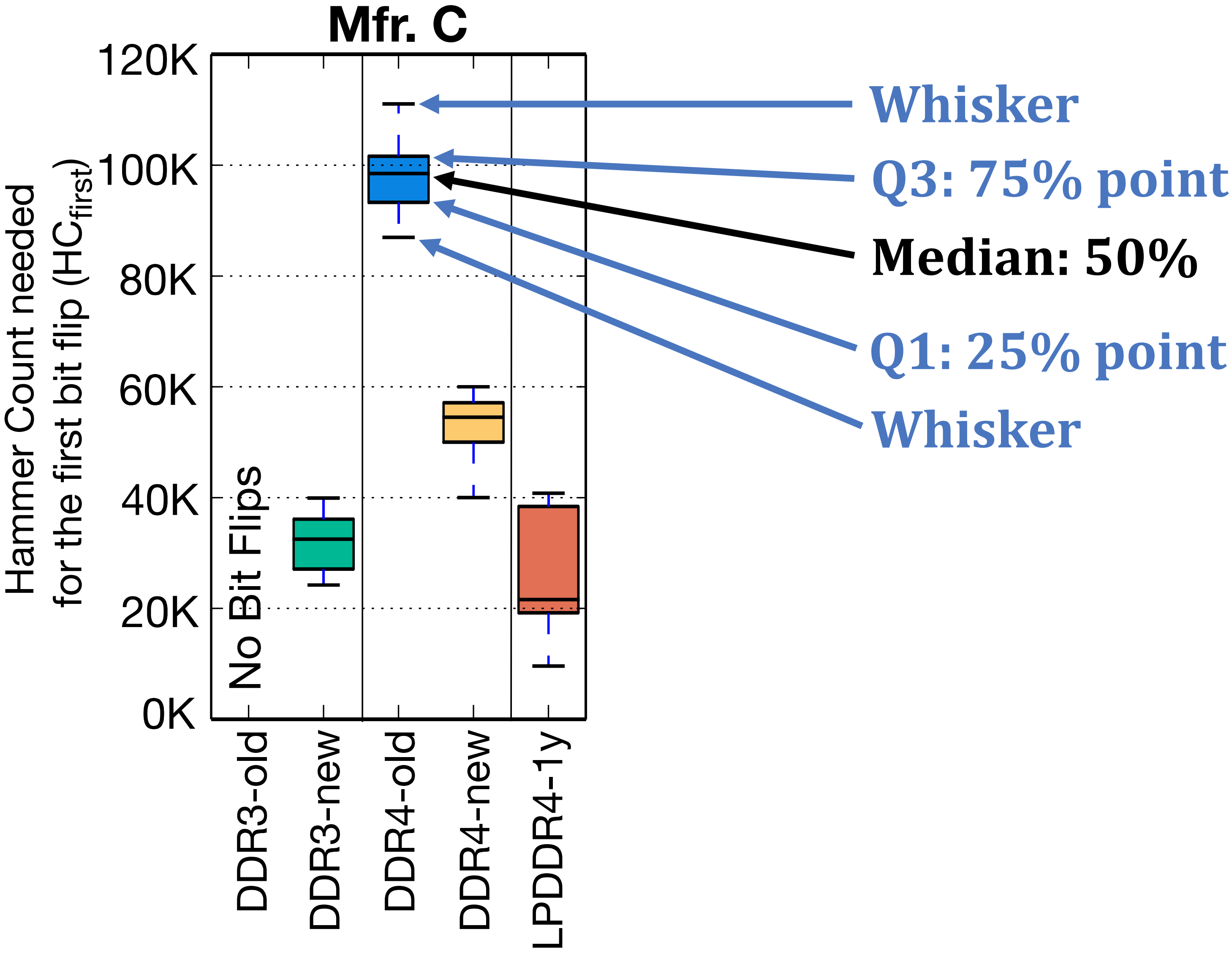


[More analysis in the paper]

SAFARI

5. First RowHammer Bit Flips per Chip

What is the minimum Hammer Count required to cause bit flips (HC_{first})?



Whisker

Q3: 75% point

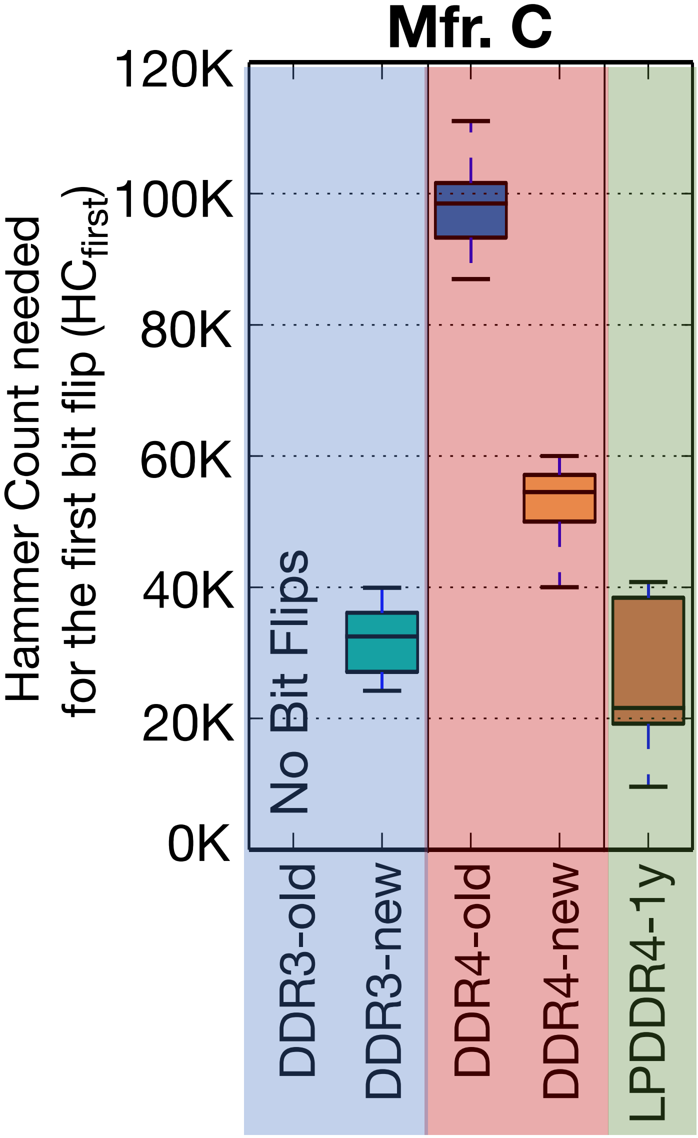
Median: 50%

Q1: 25% point

Whisker

5. First RowHammer Bit Flips per Chip

What is the minimum Hammer Count required to cause bit flips (HC_{first})?

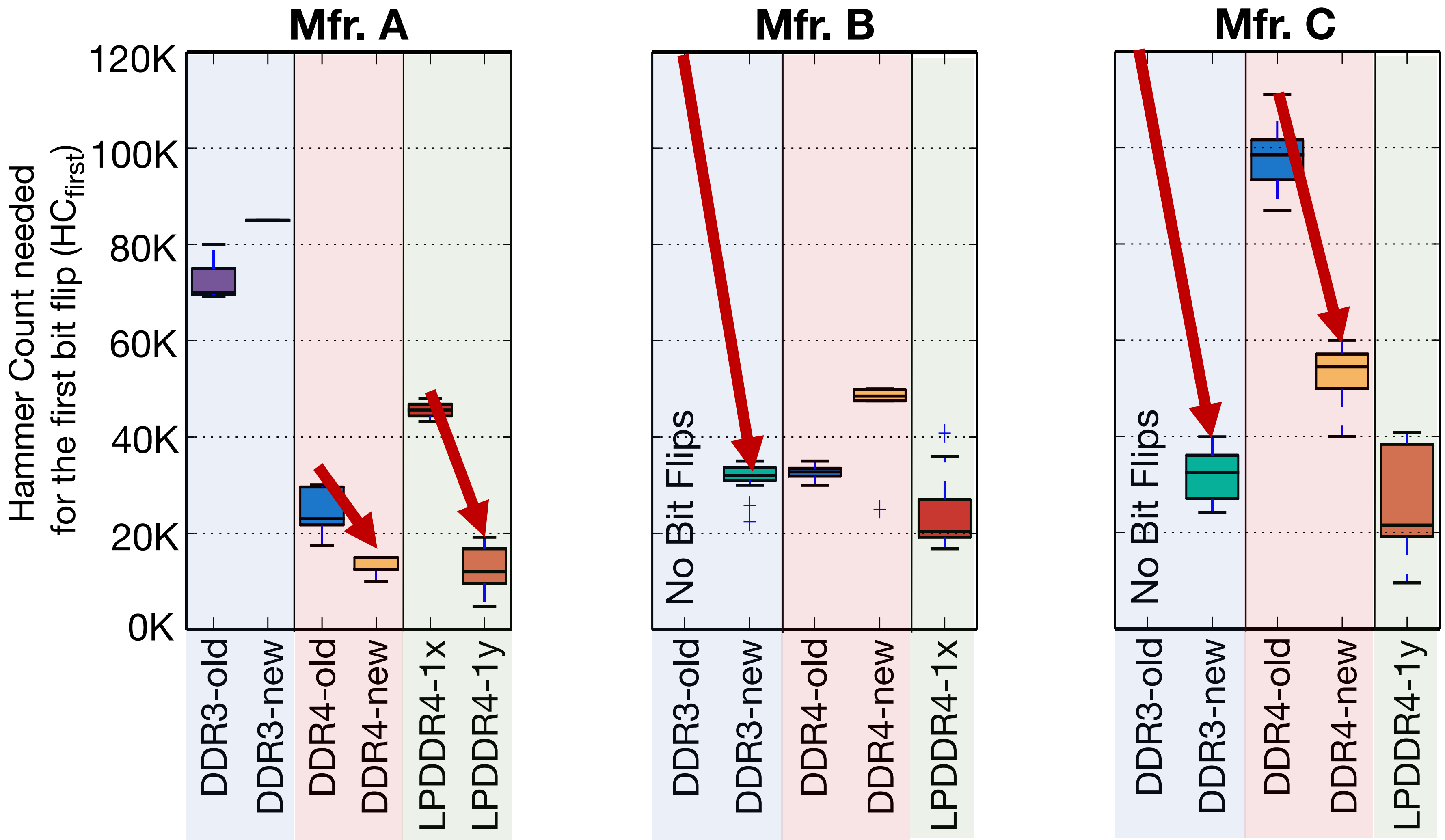


We note the different DRAM types on the x-axis: **DDR3**, **DDR4**, **LPDDR4.**

We focus on trends across chips of the same DRAM type to draw conclusions

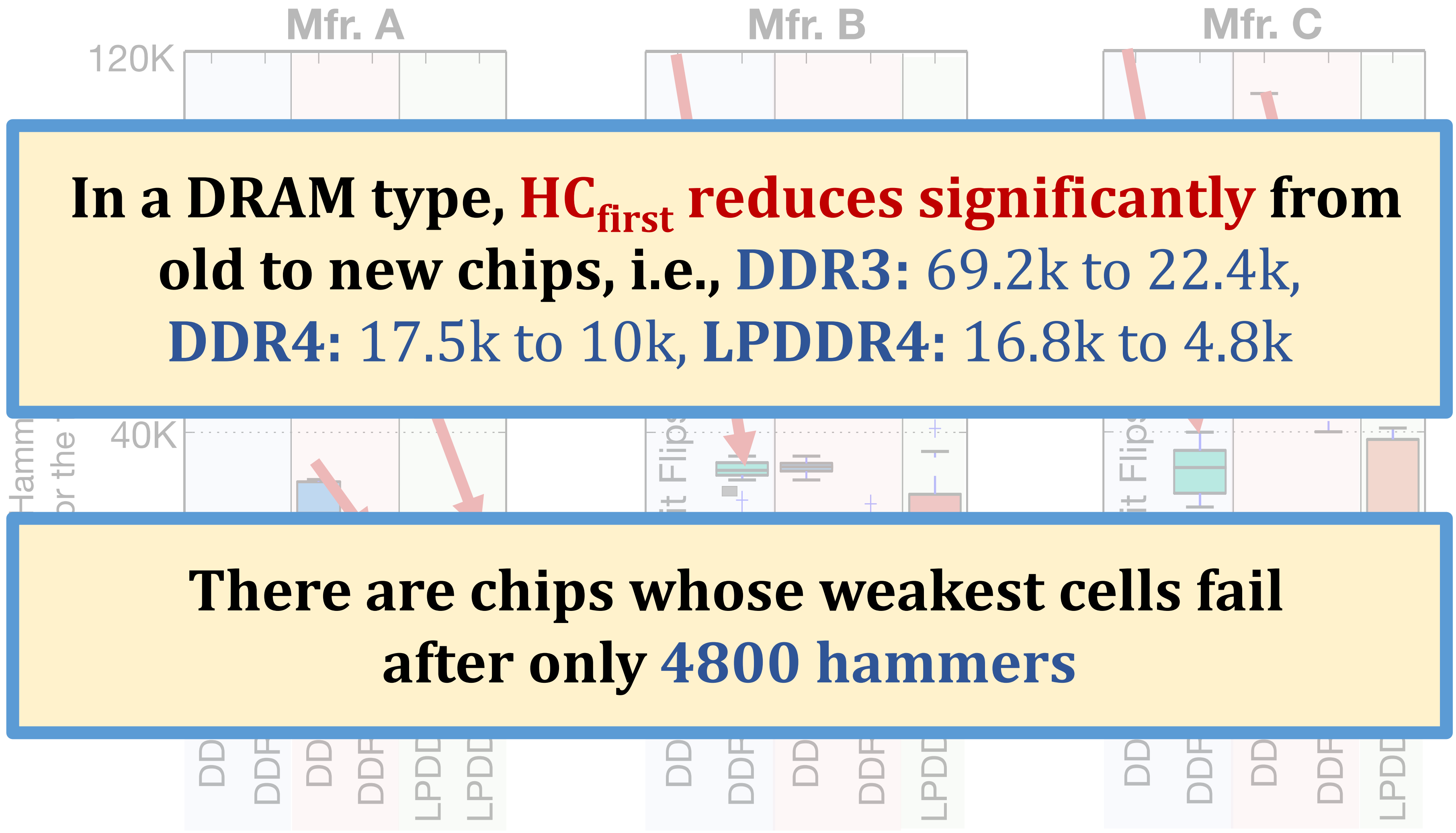
SAFARI

5. First RowHammer Bit Flips per Chip



Newer chips from a given DRAM manufacturer **more** vulnerable to RowHammer

5. First RowHammer Bit Flips per Chip



In a DRAM type, **HC$_{first}$ reduces significantly** from old to new chips, i.e., **DDR3:** 69.2k to 22.4k, **DDR4:** 17.5k to 10k, **LPDDR4:** 16.8k to 4.8k

There are chips whose weakest cells fail after only **4800 hammers**

Newer chips from a given DRAM manufacturer **more** vulnerable to RowHammer

Key Takeaways from 1580 Chips

- Chips of newer DRAM technology nodes are **more vulnerable** to RowHammer

- There are chips today whose weakest cells fail after **only 4800 hammers**

- Chips of newer DRAM technology nodes can exhibit RowHammer bit flips 1) in **more rows** and 2) **farther away** from the victim row.

Evaluation Methodology

- **Cycle-level simulator:** Ramulator [Kim+, CAL'15]
  https://github.com/CMU-SAFARI/ramulator
  - 4GHz, 4-wide, 128 entry instruction window
  - 48 8-core workload mixes randomly drawn from SPEC CPU2006 **(10 < MPKI < 740)**

- **Metrics to evaluate mitigation mechanisms**
  1. ***DRAM Bandwidth Overhead:*** fraction of total system DRAM bandwidth consumption from mitigation mechanism
  2. ***Normalized System Performance:*** normalized weighted speedup to a 100% baseline

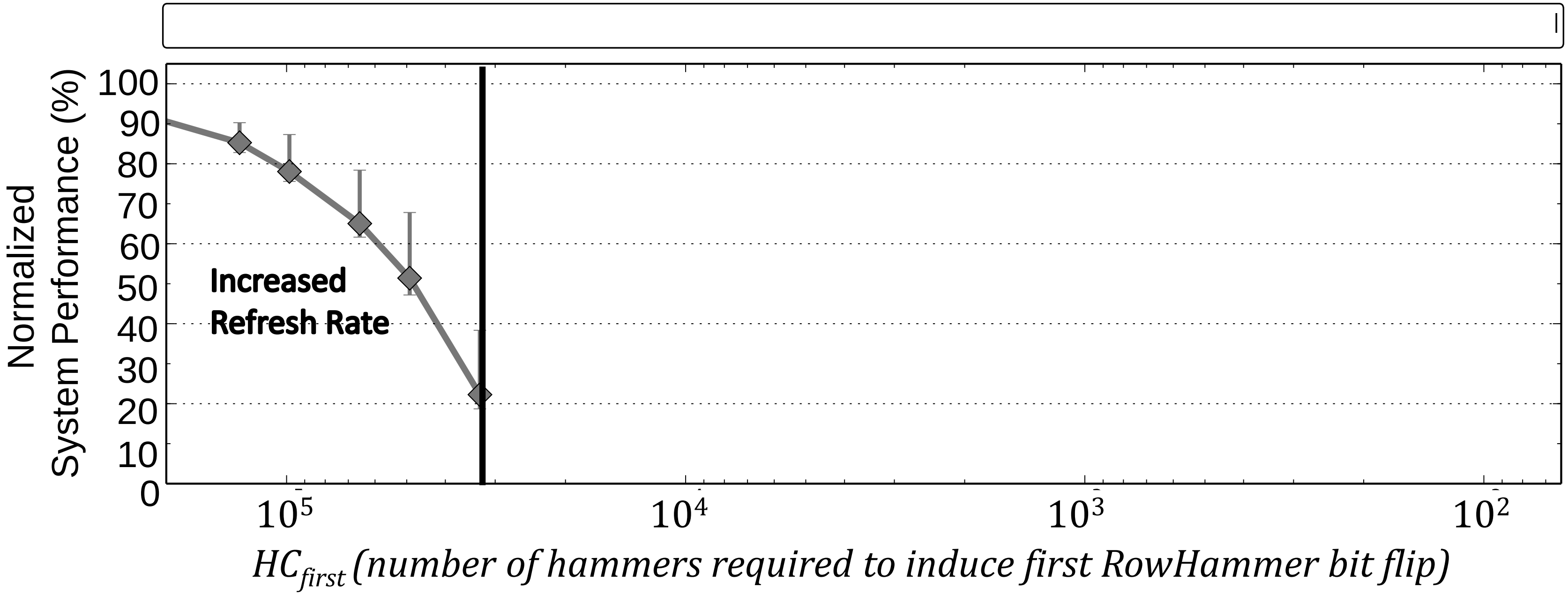
Evaluation Methodology

- We evaluate **five** state-of-the-art mitigation mechanisms:
  - **Increased Refresh Rate** **[Kim+, ISCA'14]**
  - **PARA** **[Kim+, ISCA'14]**
  - **ProHIT** **[Son+, DAC'17]**
  - **MRLoc** **[You+, DAC'19]**
  - **TWiCe** **[Lee+, ISCA'19]**

- and **one** ideal refresh-based mitigation mechanism:
  - **Ideal**

- **More detailed descriptions in the paper on:**
  - Descriptions of mechanisms in our paper and the original publications
  - How we scale each mechanism to more vulnerable DRAM chips (lower $\mathbf{HC_{first}}$)

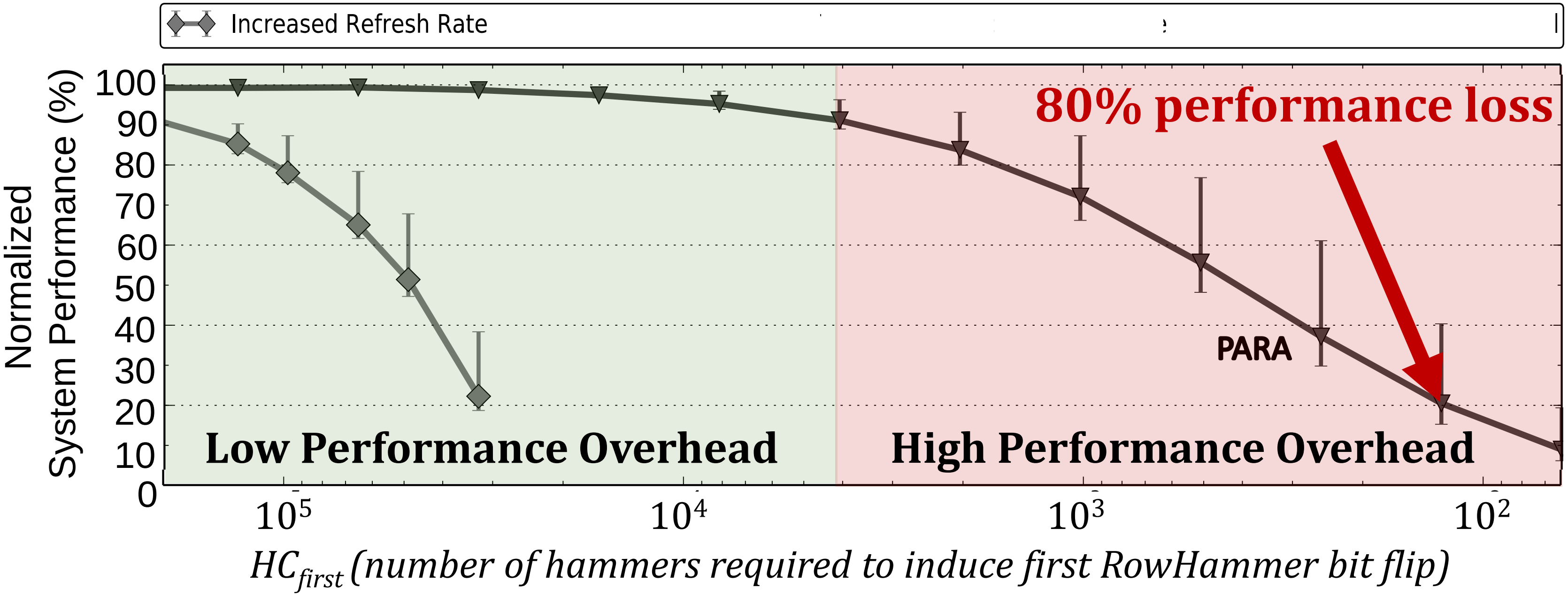
Mitigation Mech. Eval. (Increased Refresh)



Substantial overhead for high HC_{first} values.

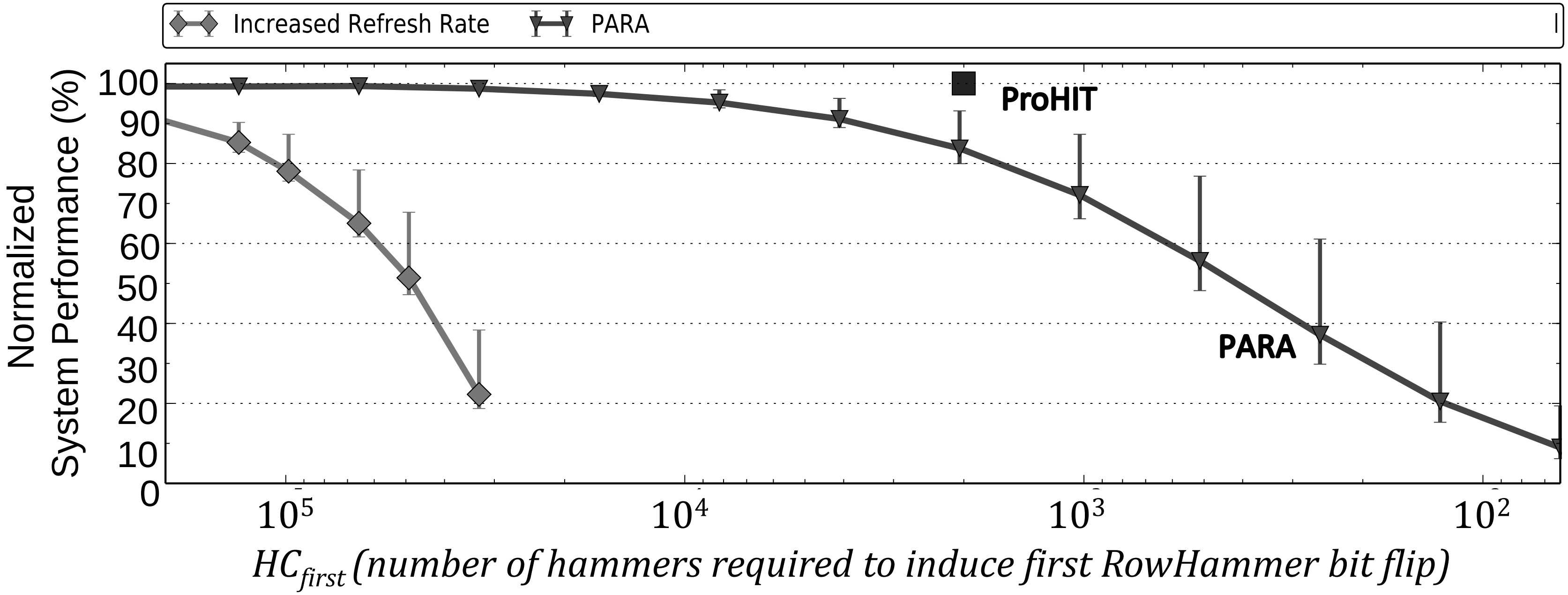
This mechanism does not support $HC_{first} < 32k$ due to the prohibitively high refresh rates required

Mitigation Mechanism Evaluation (PARA)

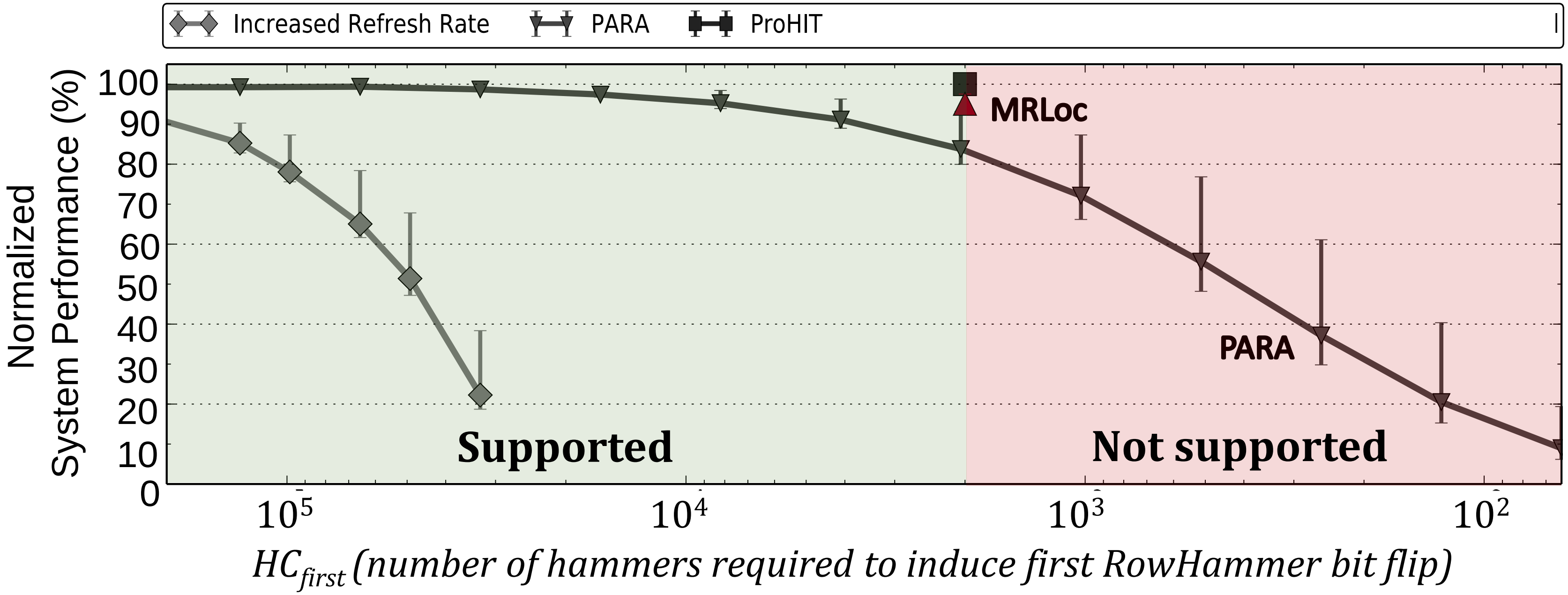


SAFARI

Mitigation Mechanism Evaluation (ProHIT)

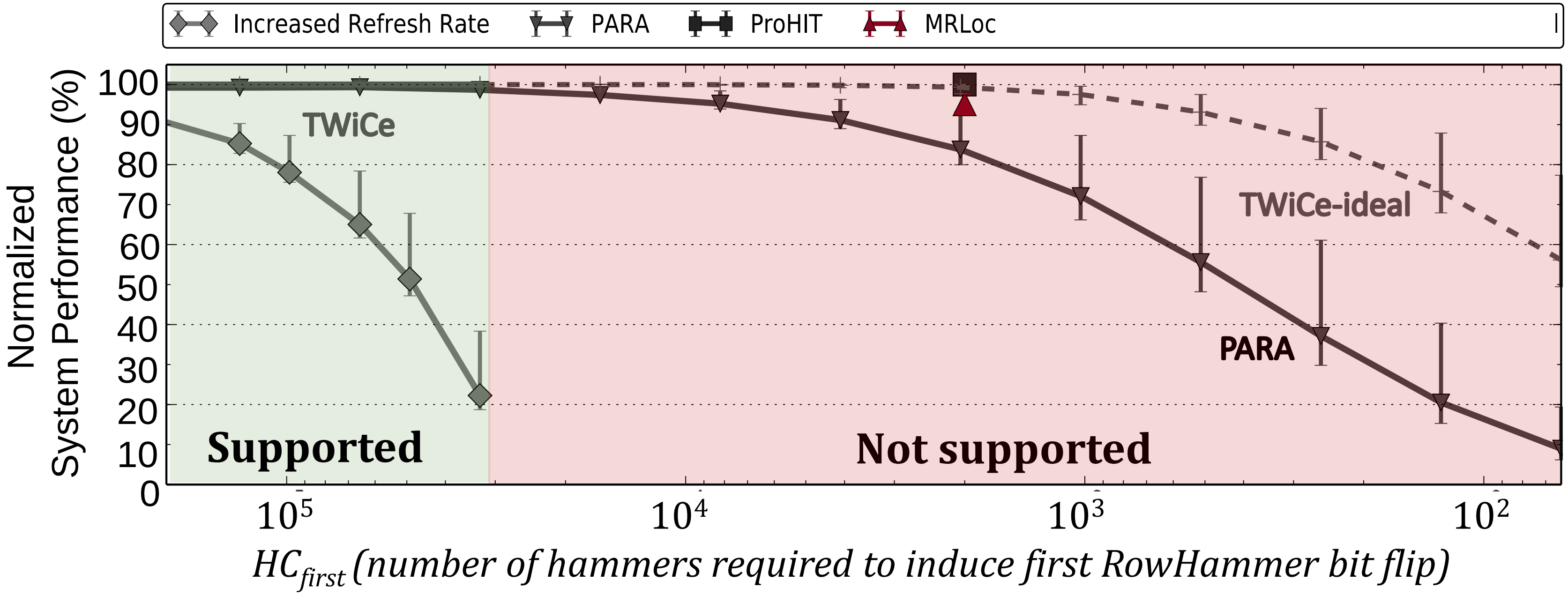


Mitigation Mechanism Evaluation (MRLoc)



Models for scaling ProHIT and MRLoc for HC_{first} < 2k are not provided and how to do so is not intuitive

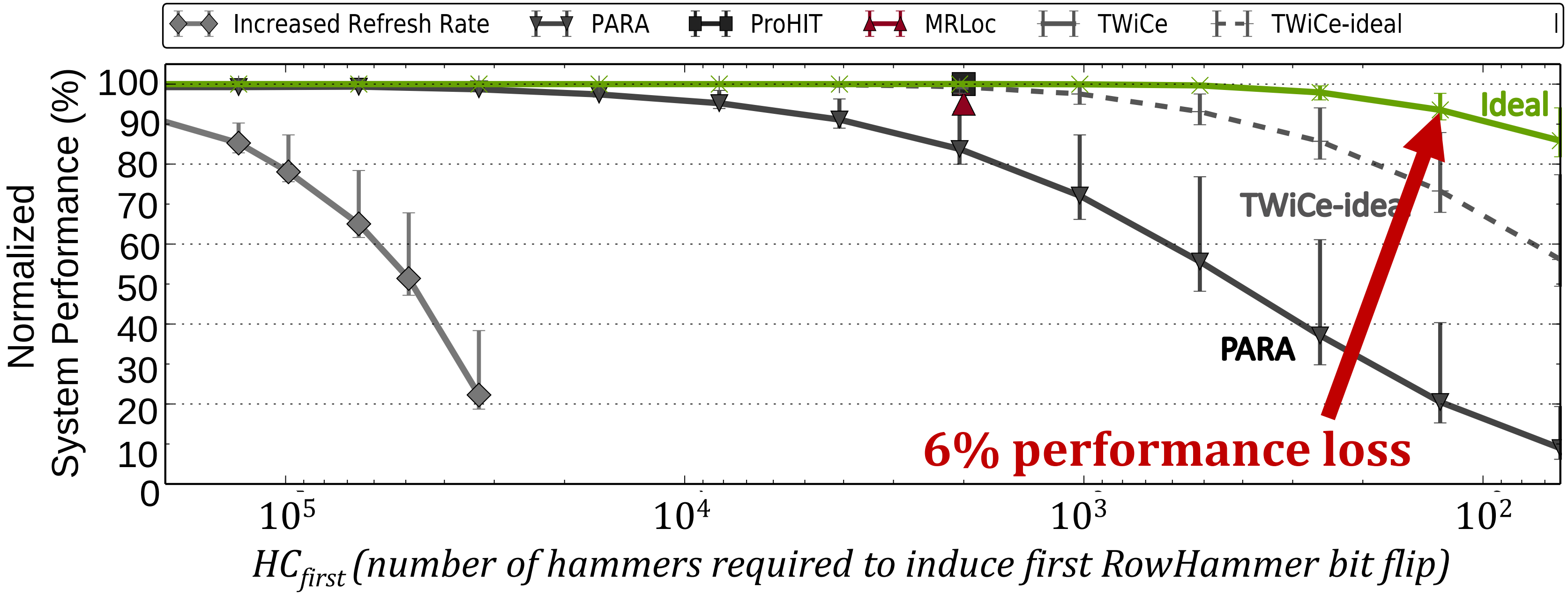
Mitigation Mechanism Evaluation (TWiCe)



TWiCe does not support HC_{first} < 32k.

We evaluate an ideal scalable version (TWiCe-ideal) assuming it solves two critical design issues

Mitigation Mechanism Evaluation (Ideal)

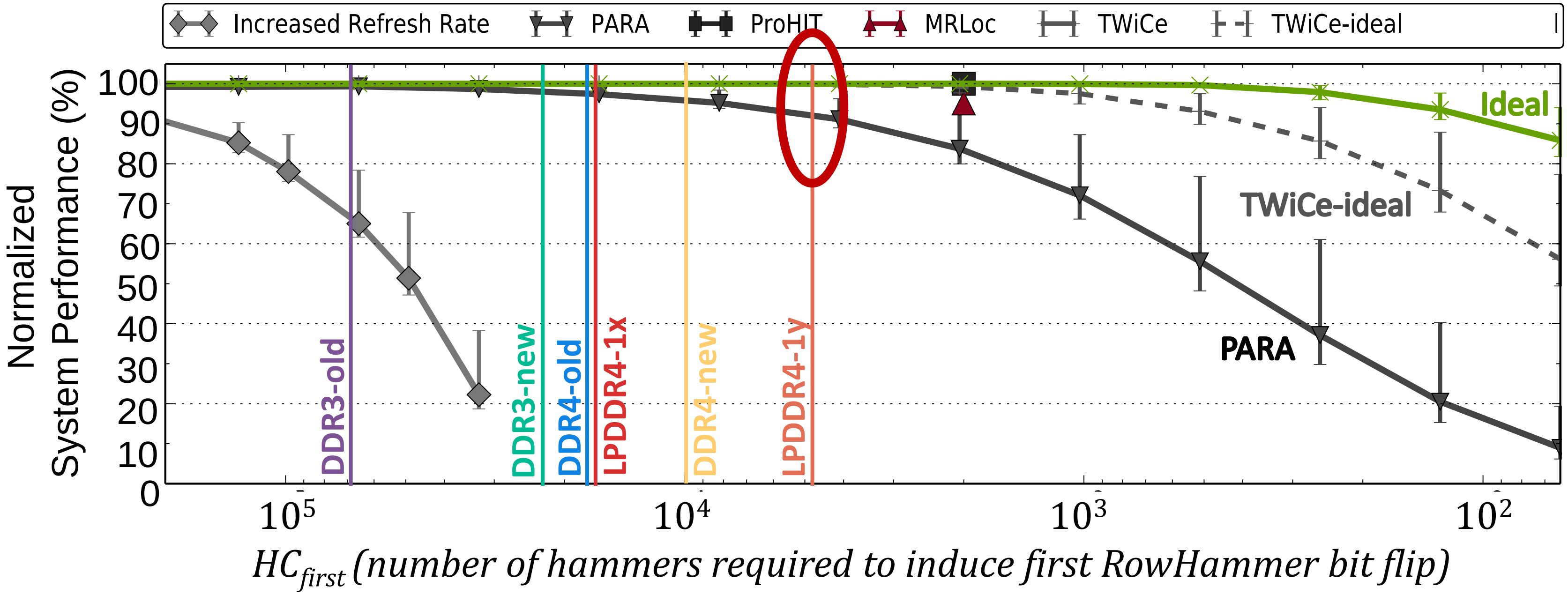


6% performance loss

Ideal mechanism issues a refresh command to a row only right before the row can potentially experience a RowHammer bit flip

SAFARI

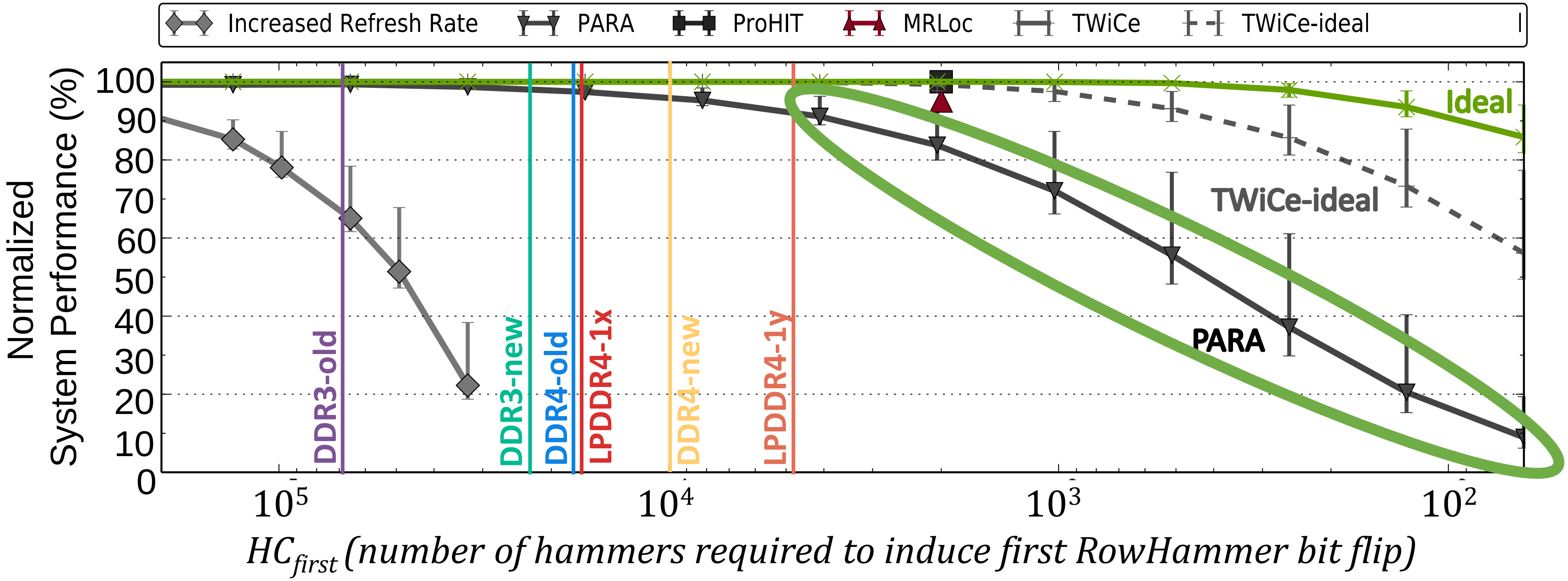
Mitigation Mechanism Evaluation



PARA, ProHIT, and MRLoc mitigate RowHammer bit flips in worst chips today with reasonable system performance (92%, 100%, 100%)

SAFARI

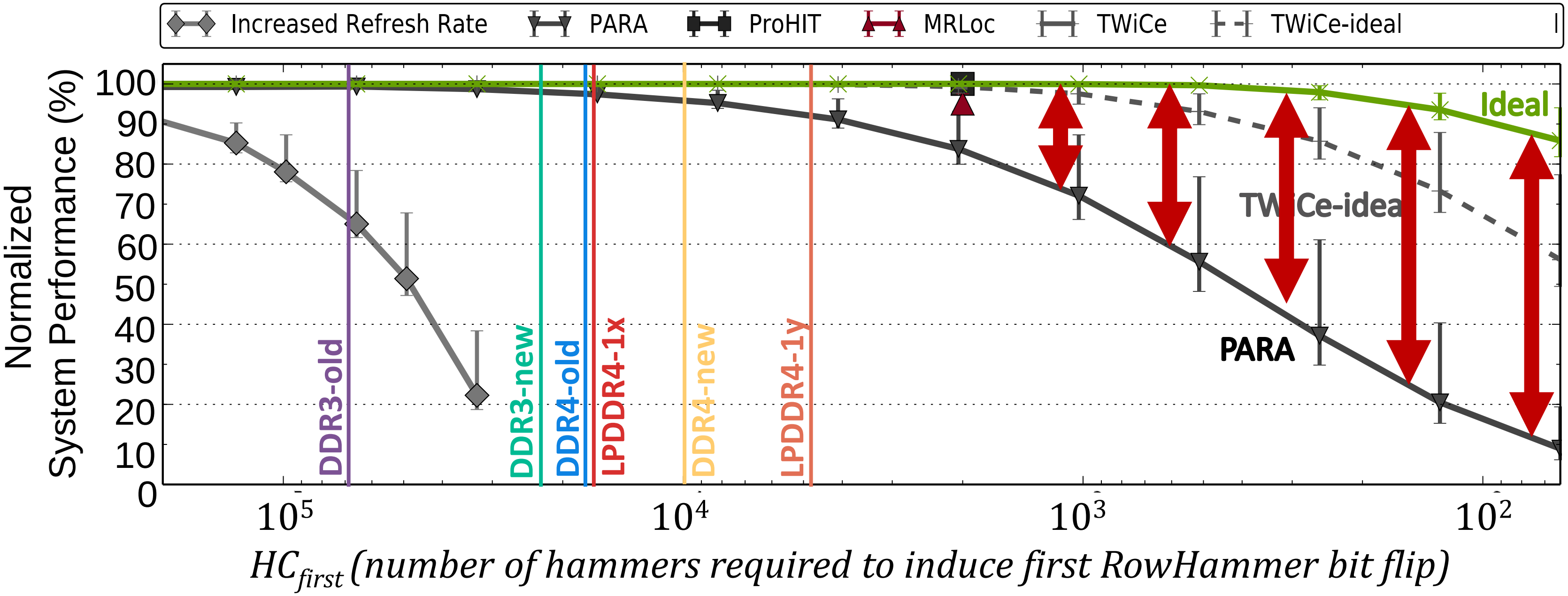
Mitigation Mechanism Evaluation



Only PARA's design scales to low HC_{first} values
but has very low normalized system performance

SAFARI

Mitigation Mechanism Evaluation



Ideal mechanism is significantly better than any existing mechanism for HC_{first} < 1024

Significant opportunity for developing a RowHammer solution with low performance overhead that supports low HC_{first}

Key Takeaways from Mitigation Mechanisms

- Existing RowHammer mitigation mechanisms can prevent RowHammer attacks with **reasonable system performance overhead** in DRAM chips today

- Existing RowHammer mitigation mechanisms **do not scale well** to DRAM chips more vulnerable to RowHammer

- There is still **significant opportunity** for developing a mechanism that is **scalable with low overhead**

Additional Details in the Paper

- **Single-cell RowHammer bit flip probability**
- More details on our **data pattern dependence** study
- Analysis of **Error Correcting Codes (ECC)** in mitigating RowHammer bit flips
- Additional **observations** on our data
- **Methodology details** for characterizing DRAM
- Further discussion on comparing data across different infrastructures
- **Discussion on scaling** each mitigation mechanism

RowHammer Solutions Going Forward

Two promising directions for new RowHammer solutions:

1. DRAM-system cooperation

- We believe the DRAM and system should cooperate more to provide a **holistic** solution can prevent RowHammer at **low cost**

2. Profile-guided

- Accurate **profile of RowHammer-susceptible cells** in DRAM provides a powerful substrate for building **targeted** RowHammer solutions, e.g.:
    - Only increase the refresh rate for rows containing RowHammer-susceptible cells

- A **fast and accurate** profiling mechanism is a key research challenge for developing low-overhead and scalable RowHammer solutions

Conclusion

- We characterized **1580 DRAM** chips of different DRAM types, technology nodes, and manufacturers.

- We studied **five** state-of-the-art RowHammer mitigation mechanisms and an ideal refresh-based mechanism

- We made **two key observations**
  1. **RowHammer is getting much worse**. It takes much fewer hammers to induce RowHammer bit flips in newer chips
     - e.g., **DDR3:** 69.2k to 22.4k, **DDR4:** 17.5k to 10k, **LPDDR4:** 16.8k to 4.8k
  2. **Existing mitigation mechanisms do not scale** to DRAM chips that are more vulnerable to RowHammer
     - e.g., 80% performance loss when the hammer count to induce the first bit flip is 128

- We **conclude** that it is **critical** to do more research on RowHammer and develop scalable mitigation mechanisms to prevent RowHammer in future systems

Revisiting RowHammer

An Experimental Analysis of Modern Devices and Mitigation Techniques

Jeremie S. Kim Minesh Patel
A. Giray Yağlıkçı Hasan Hassan
Roknoddin Azizi Lois Orosa Onur Mutlu

SAFARI

ETH zürich

Carnegie Mellon

Revisiting RowHammer in 2020 (I)

- 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

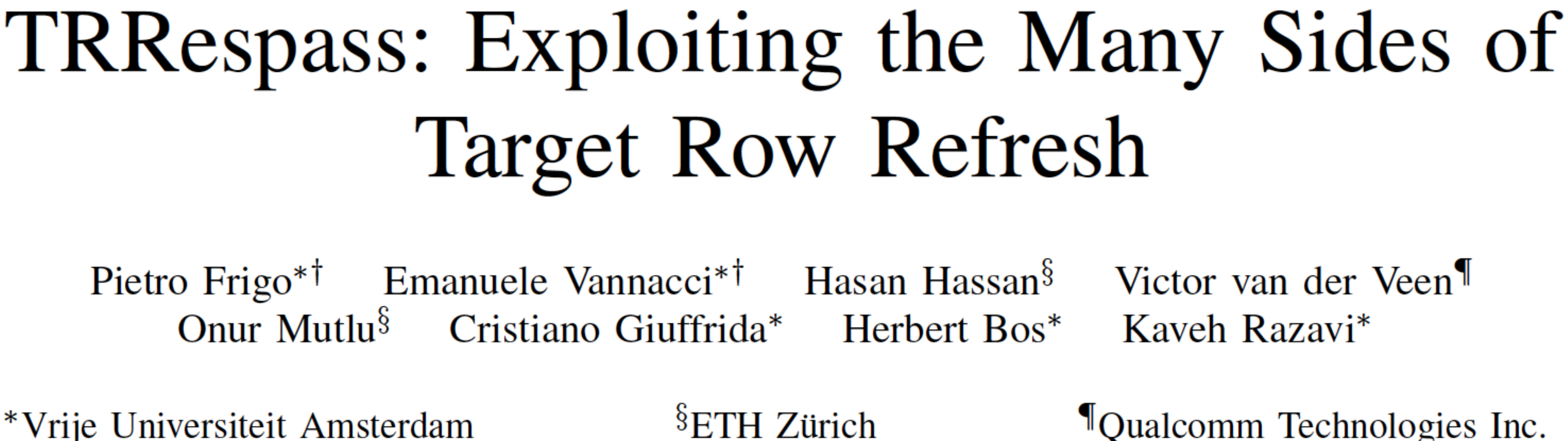
Jeremie S. Kim[§†] Minesh Patel[§] A. Giray Yağlıkçı[§]
Hasan Hassan[§] Roknoddin Azizi[§] Lois Orosa[§] Onur Mutlu[§†]

[§]*ETH Zürich* [†]*Carnegie Mellon University*

TRRespass

RowHammer in 2020 (II)

- 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*

*Vrije Universiteit Amsterdam §ETH Zürich ¶Qualcomm Technologies Inc.

TRRespass

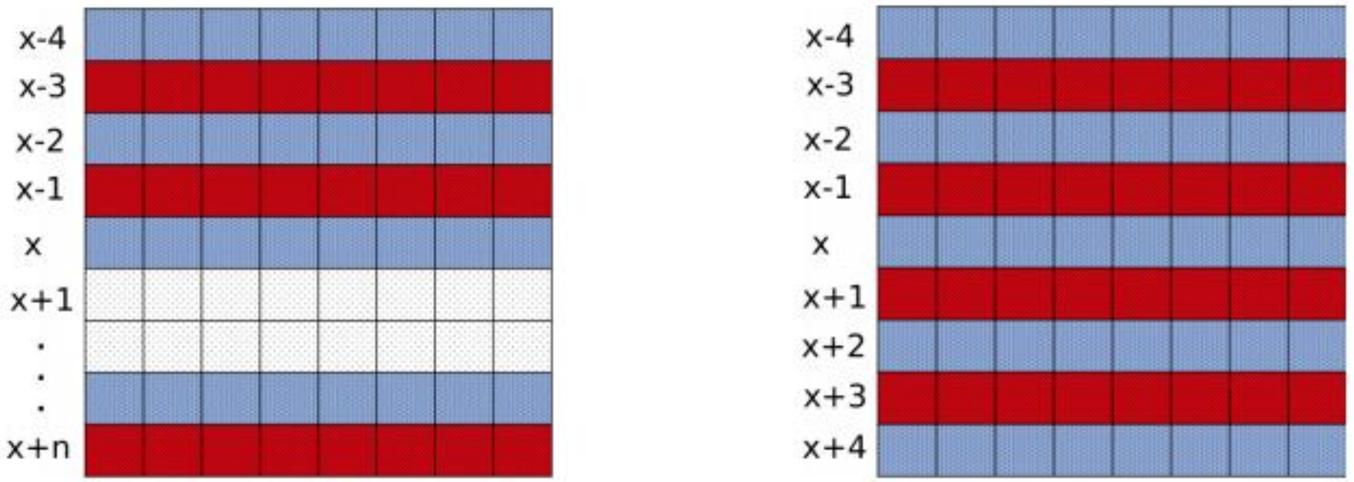
- First work to show that TRR-protected DRAM chips are vulnerable to RowHammer in the field
  - Mitigations advertised as secure are not secure

- Introduces the Many-sided RowHammer attack
  - Idea: Hammer many rows to bypass TRR mitigations (e.g., by overflowing proprietary TRR tables that detect aggressor rows)

- (Partially) reverse-engineers the TRR and pTRR mitigation mechanisms implemented in DRAM chips and memory controllers

- Provides an automatic tool that can effectively create many-sided RowHammer attacks in DDR4 and LPDDR4(X) chips

Example Many-Sided Hammering Patterns



(a) Assisted double-sided

(b) 4-sided

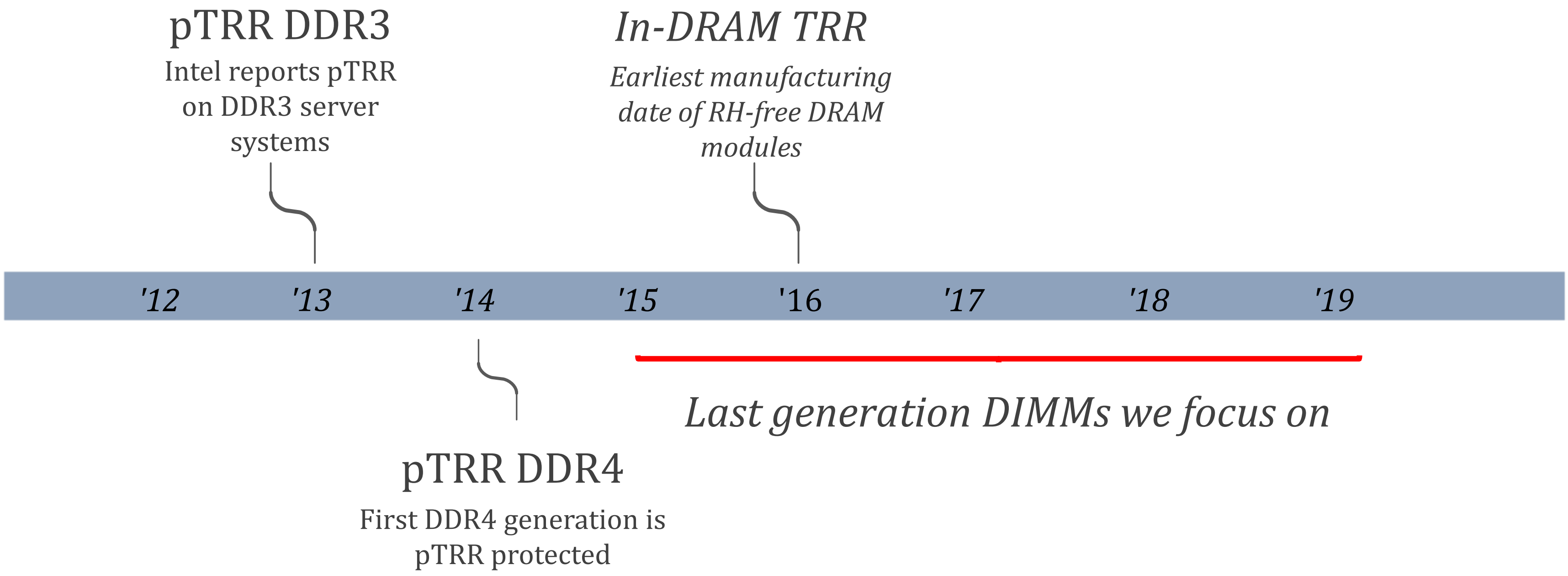
Fig. 12: Hammering patterns discovered by *TRRespass*. Aggressor rows are in red (🟥) and victim rows are in blue (🟦).

Target Row Refresh (TRR)

- How does it work?
  1. *Track activation count of each DRAM row*
  2. *Refresh neighbor rows if row activation count exceeds a threshold*
  - Many possible implementations in practice
  - Security through obscurity

- In-DRAM TRR
  - Embedded in the DRAM circuitry, i.e., not exposed to the memory controller

Timeline of TRR Implementations



pTRR DDR3

Intel reports pTRR on DDR3 server systems

In-DRAM TRR

Earliest manufacturing date of RH-free DRAM modules

'12 '13 '14 '15 '16 '17 '18 '19

pTRR DDR4

First DDR4 generation is pTRR protected

Last generation DIMMs we focus on

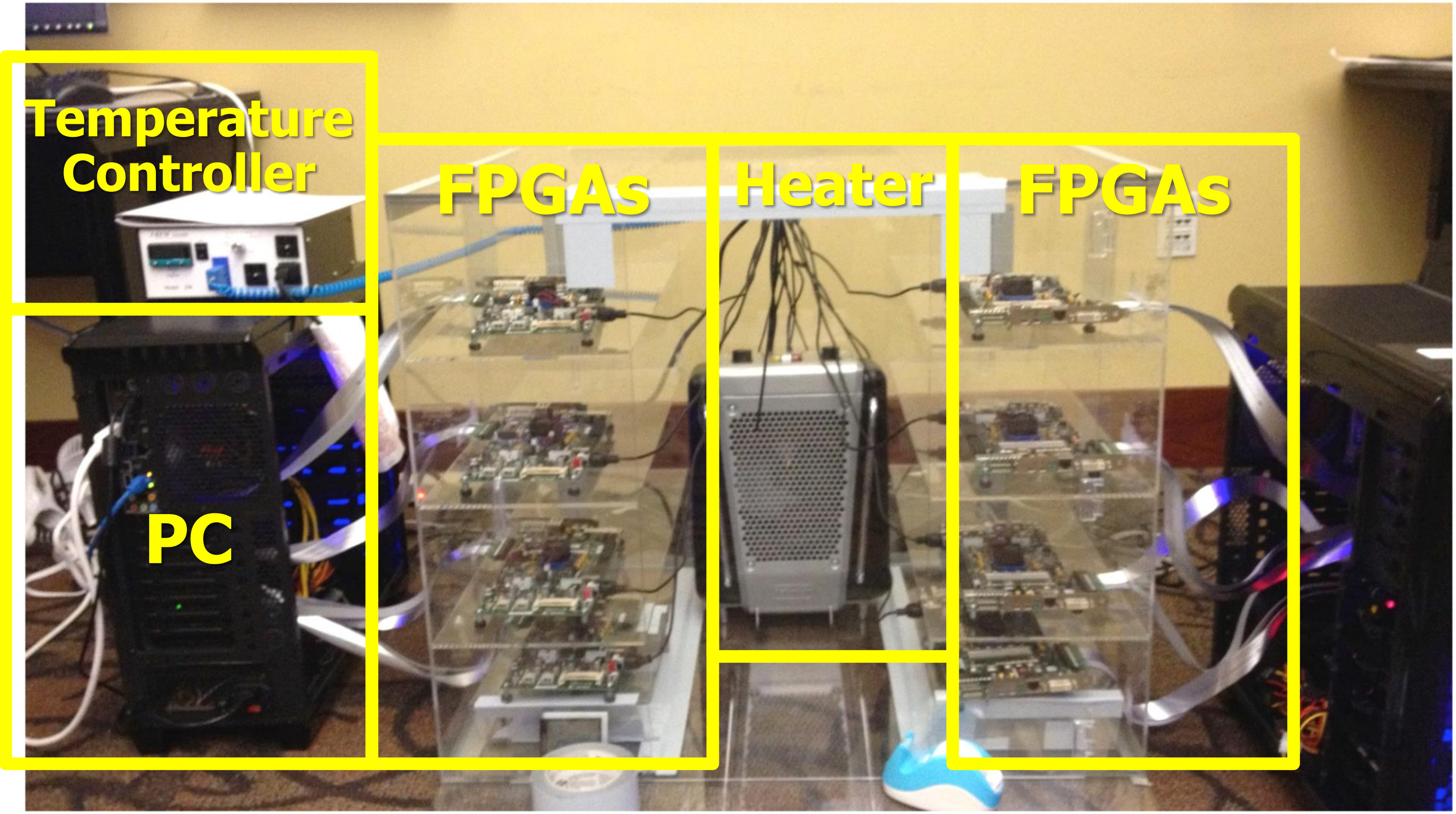
Our Goals

- Reverse engineer in-DRAM TRR to demystify how it works

- Bypass TRR protection
  - A Novel hammering pattern: **The Many-sided RowHammer**
  - Hammering up to **20 aggressor rows** allows bypassing TRR

- Automatically test memory devices: **TRRespass**
  - Automate hammering pattern generation

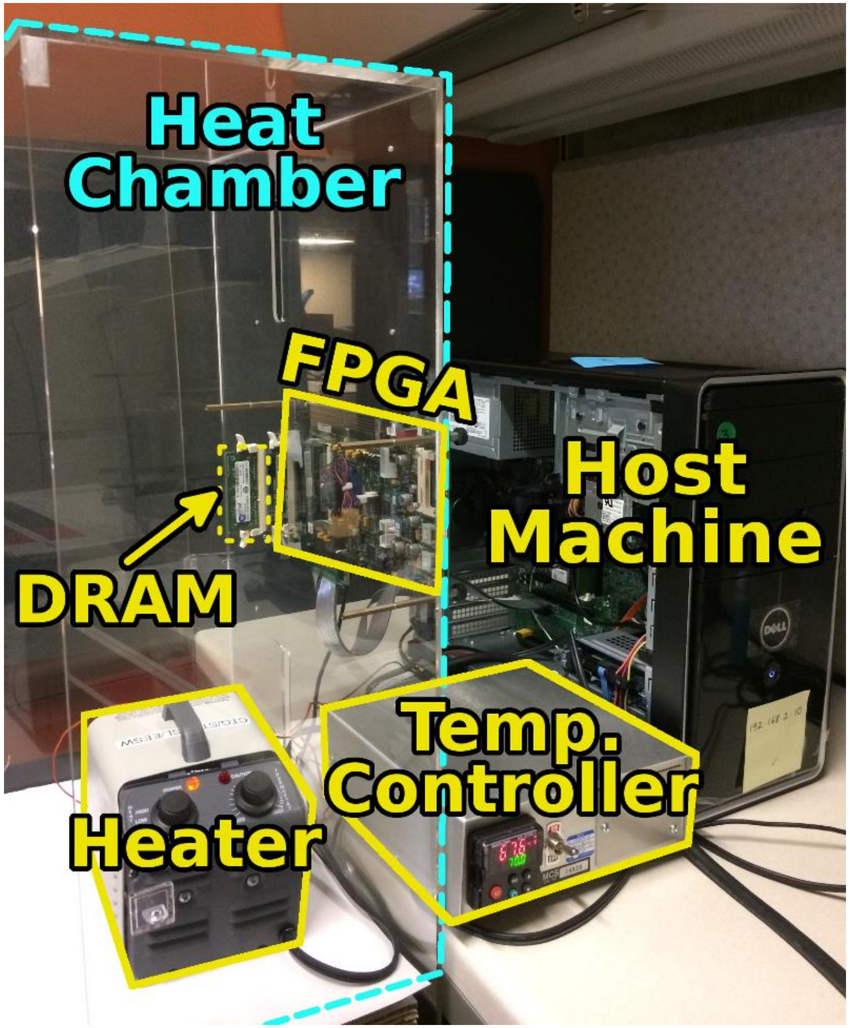
Infrastructures to Understand Such Issues



Kim+, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," ISCA 2014.

SoftMC: Open Source DRAM Infrastructure

- Hasan Hassan et al., "**SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies**," HPCA 2017.

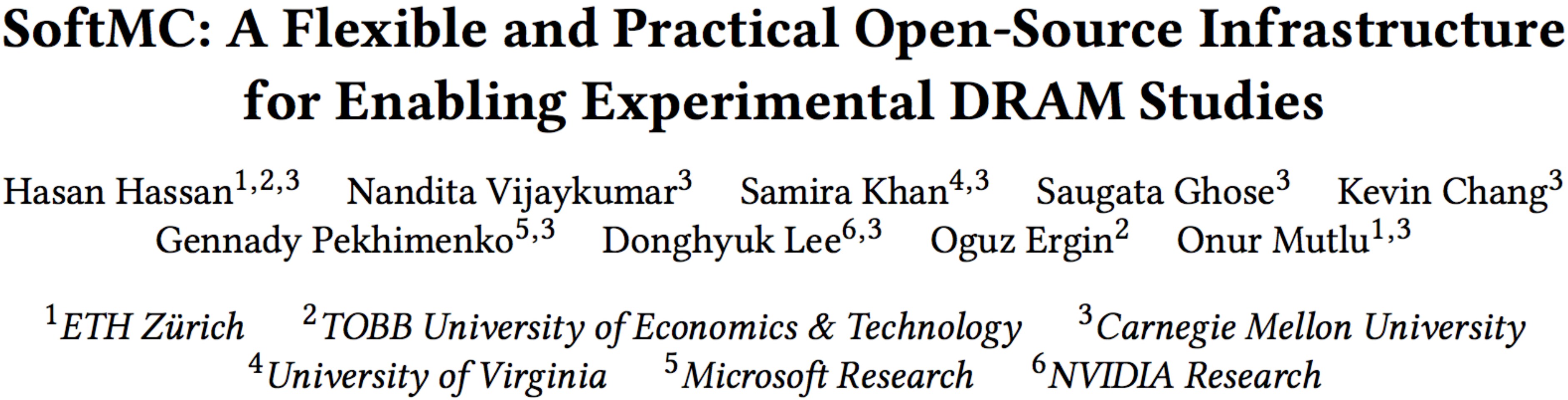


- **Flexible**
- **Easy to Use (C++ API)**
- **Open-source**

  github.com/CMU-SAFARI/SoftMC

SoftMC

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



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

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

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

Components of In-DRAM TRR

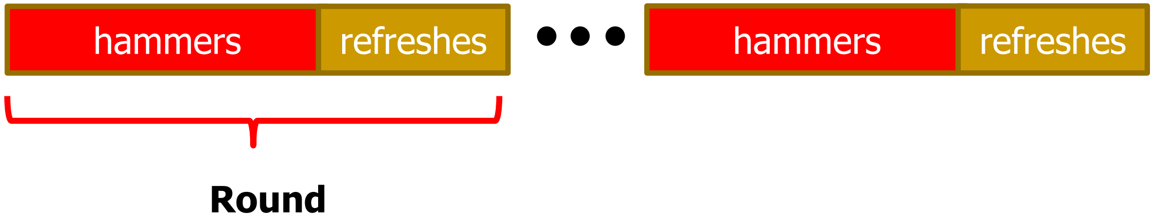
- **Sampler**
  - Tracks aggressor rows activations
  - Design options:
    - Frequency based (record every Nth row activation)
    - Time based (record first N row activations)
    - Random seed (record based on a coin flip)
  - Regardless, the sampler has a limited size

- **Inhibitor**
  - Prevents bit flips by refreshing victim rows
    - The latency of performing victim row refreshes is squeezed into slack time available in *tRFC* (i.e., the latency of regular Refresh command)

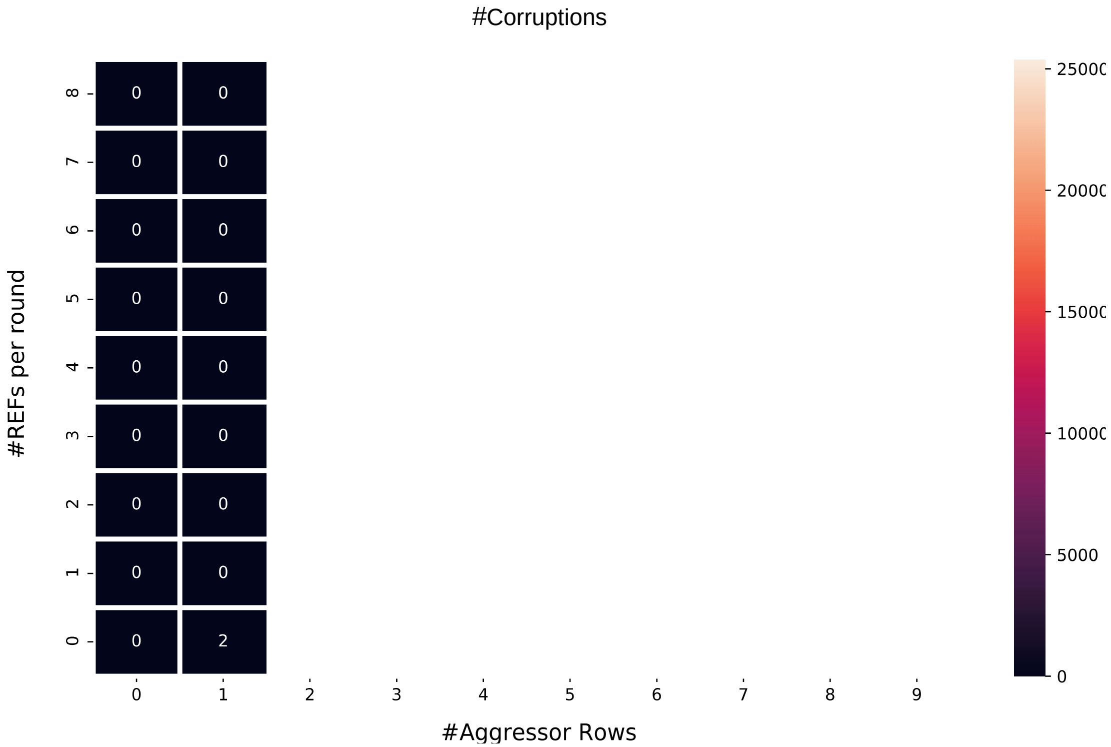
Case Study: Vendor C

How big is the sampler?

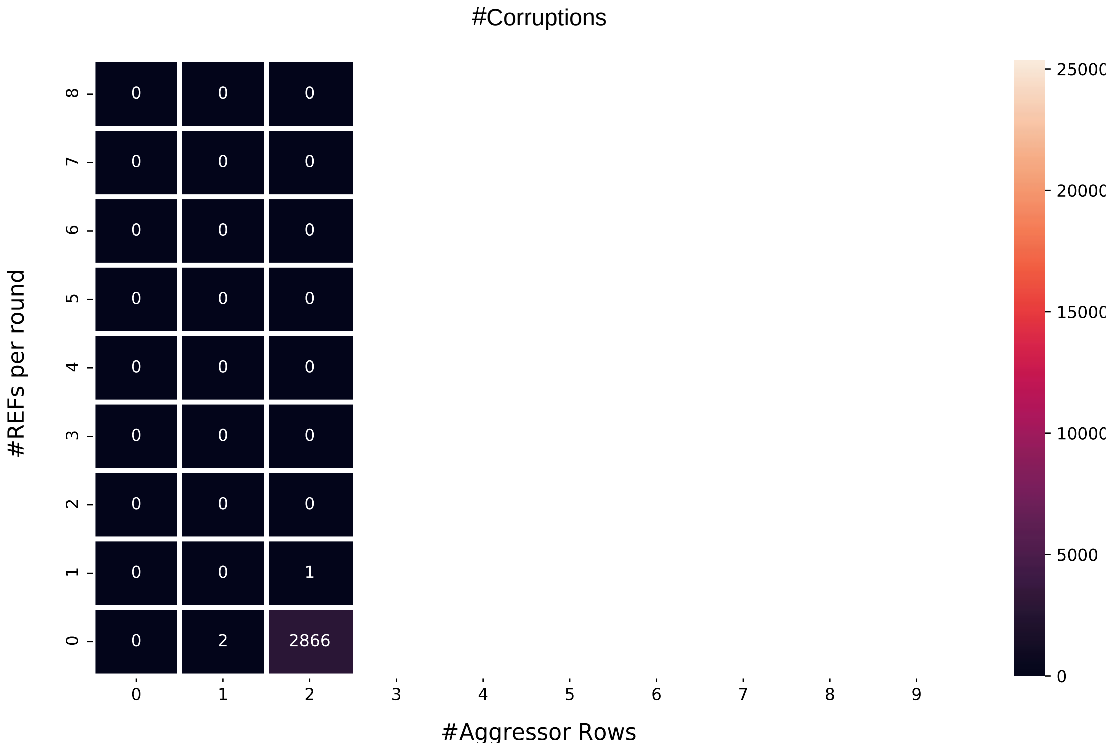
- Pick **N** aggressor rows
- Perform a series of hammers (i.e., activations of aggressors)
  - **8K activations**
- After each series of hammers, issue **R refreshes**
- **10 Rounds**



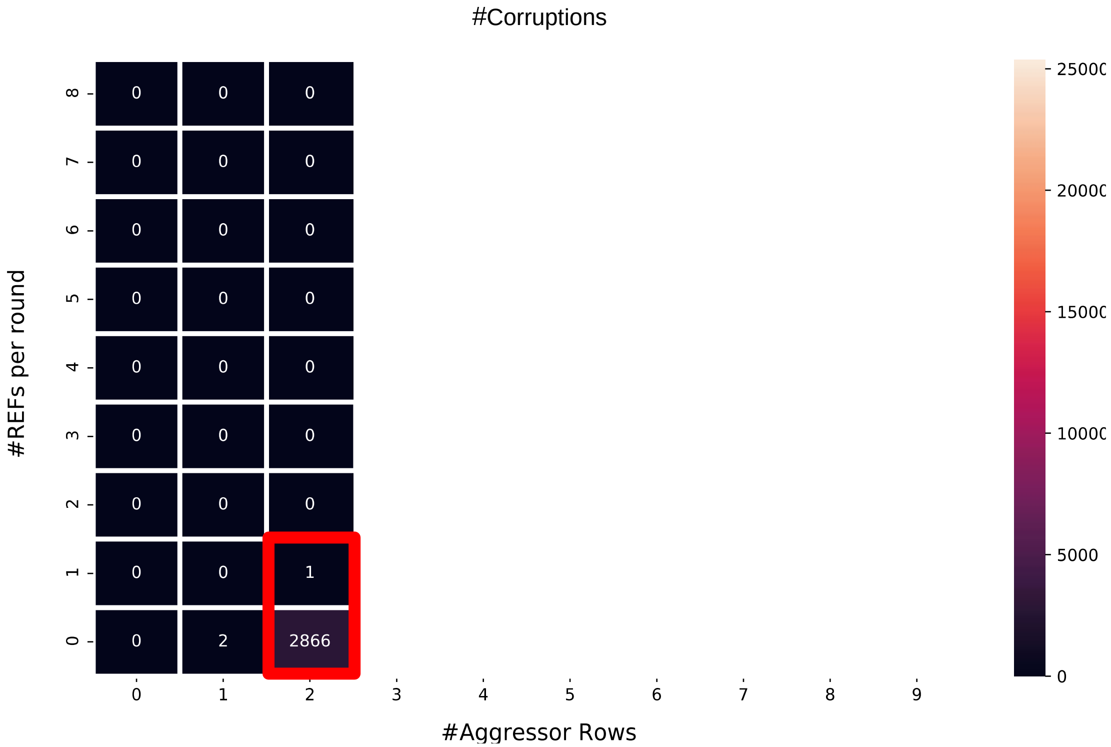
Case Study: Vendor C



Case Study: Vendor C

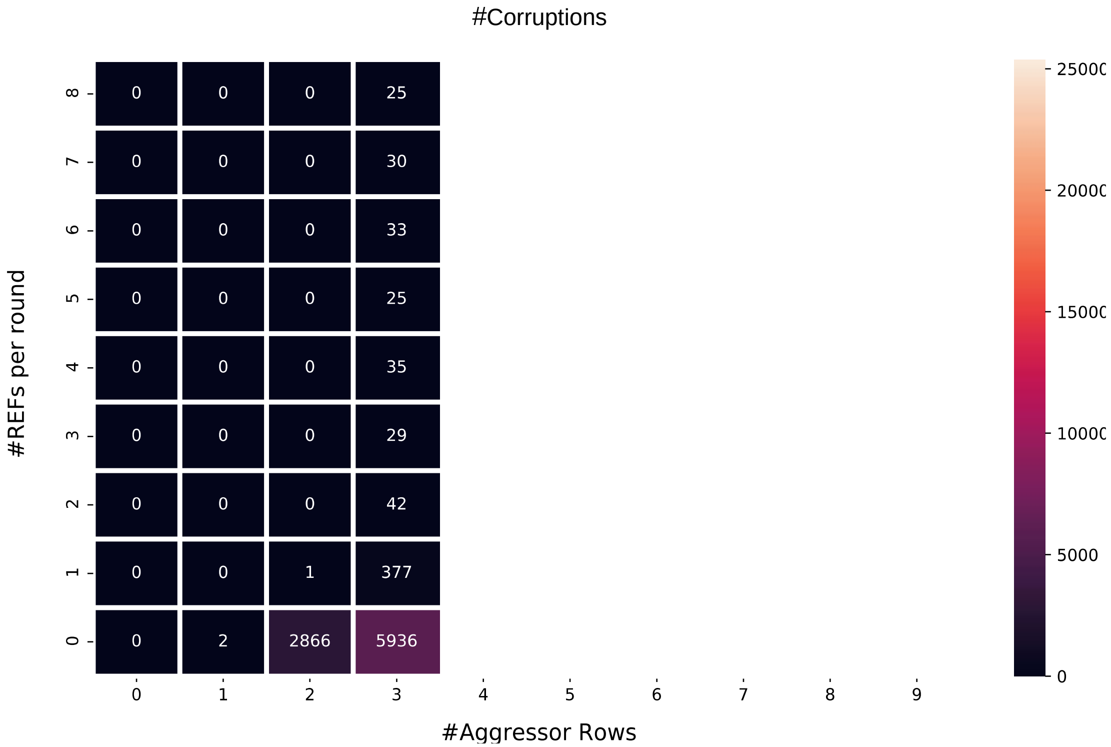


Case Study: Vendor C

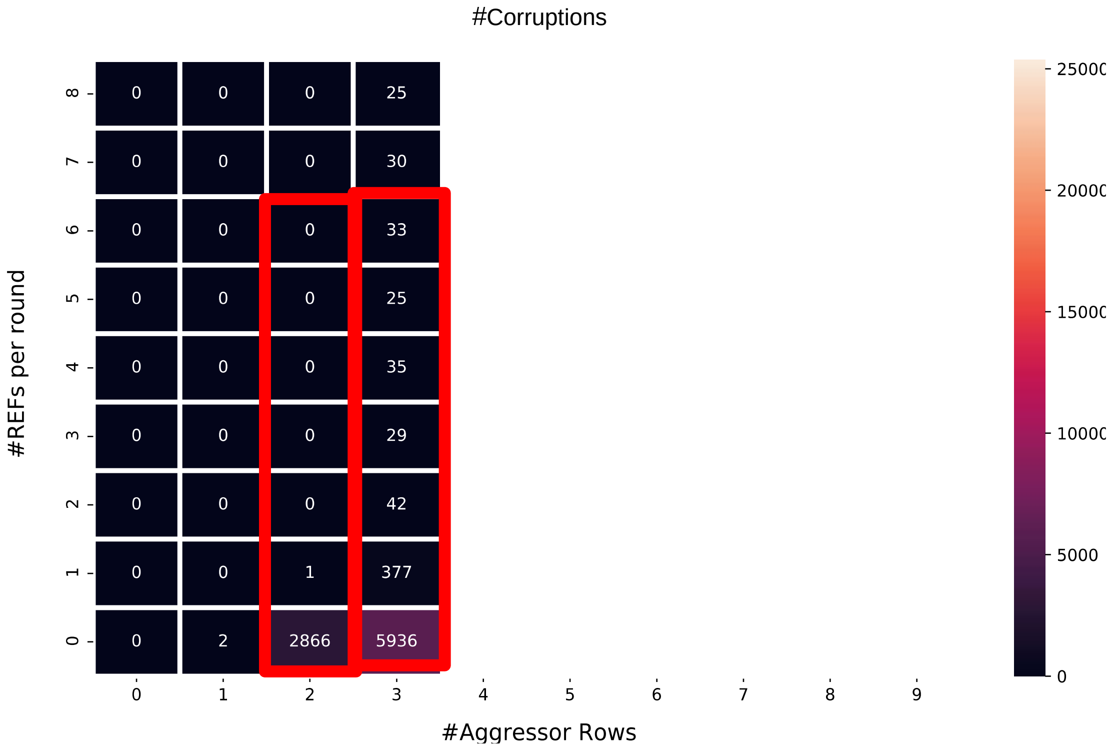


1. The TRR mitigation **acts on a refresh command**

Case Study: Vendor C

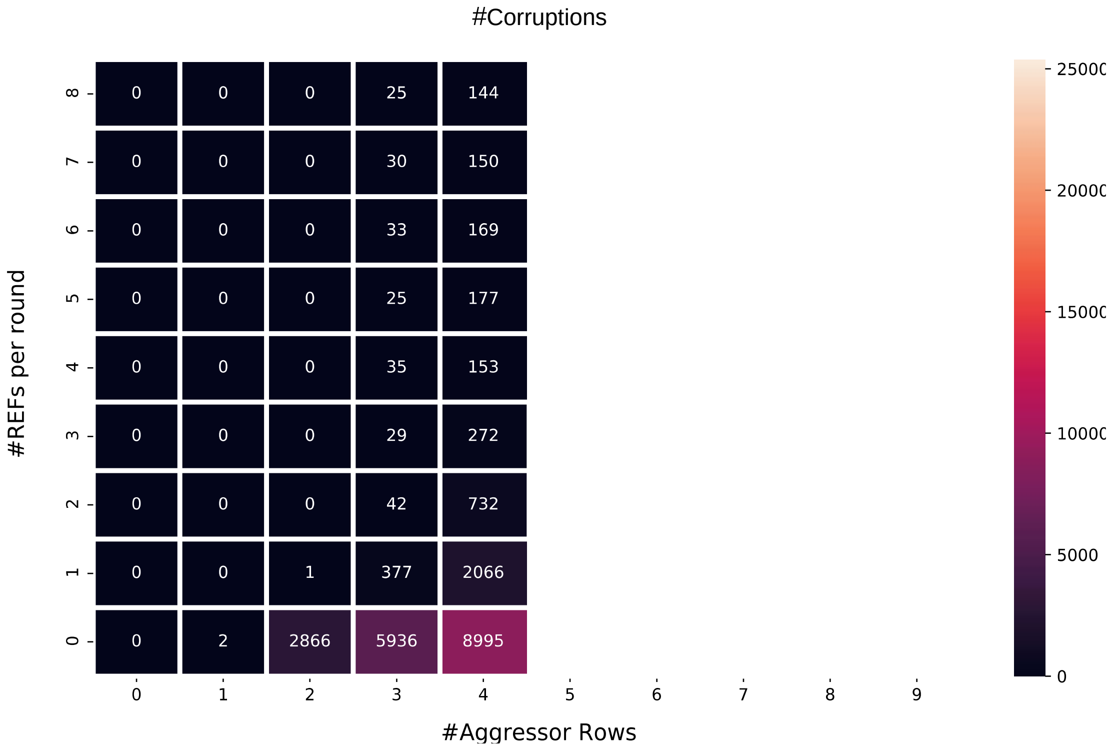


Case Study: Vendor C



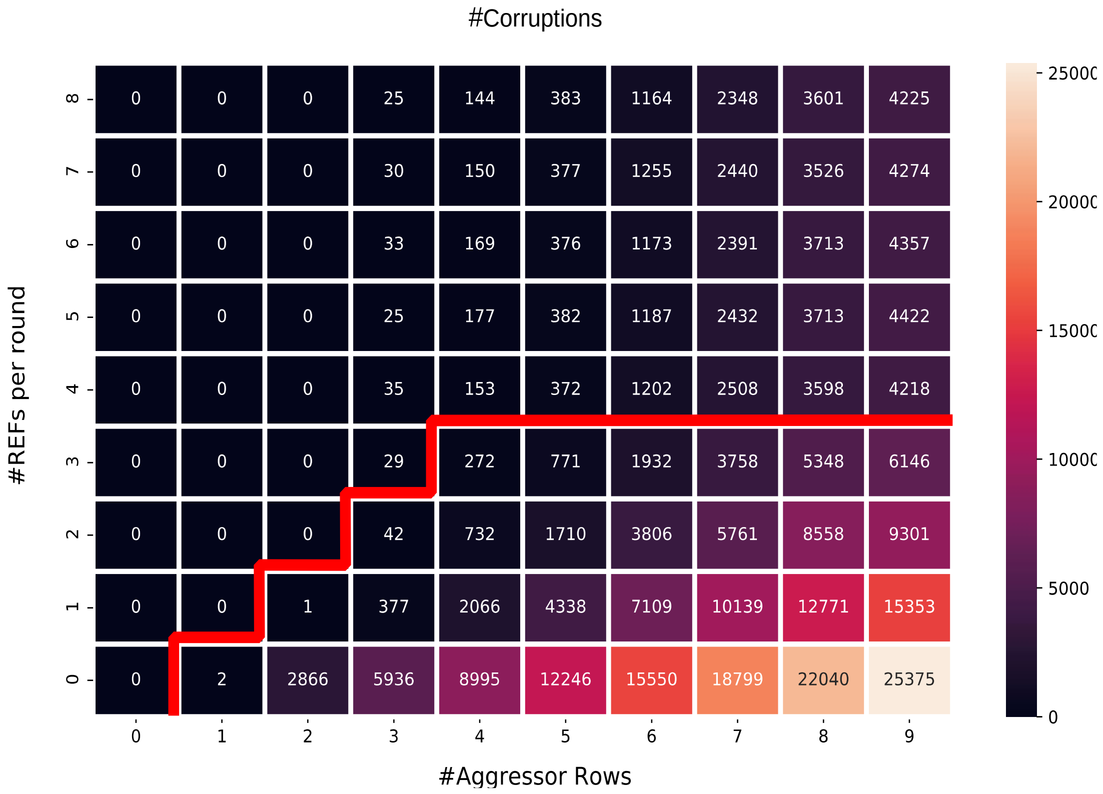
2. The mitigation **can sample more than one aggressor** per refresh interval
3. The mitigation **can refresh only a single victim** within a refresh operation

Case Study: Vendor C



#REFs per round	0	1	2	3	4
8	0	0	0	25	144
7	0	0	0	30	150
6	0	0	0	33	169
5	0	0	0	25	177
4	0	0	0	35	153
3	0	0	0	29	272
2	0	0	0	42	732
1	0	0	1	377	2066
0	0	2	2866	5936	8995

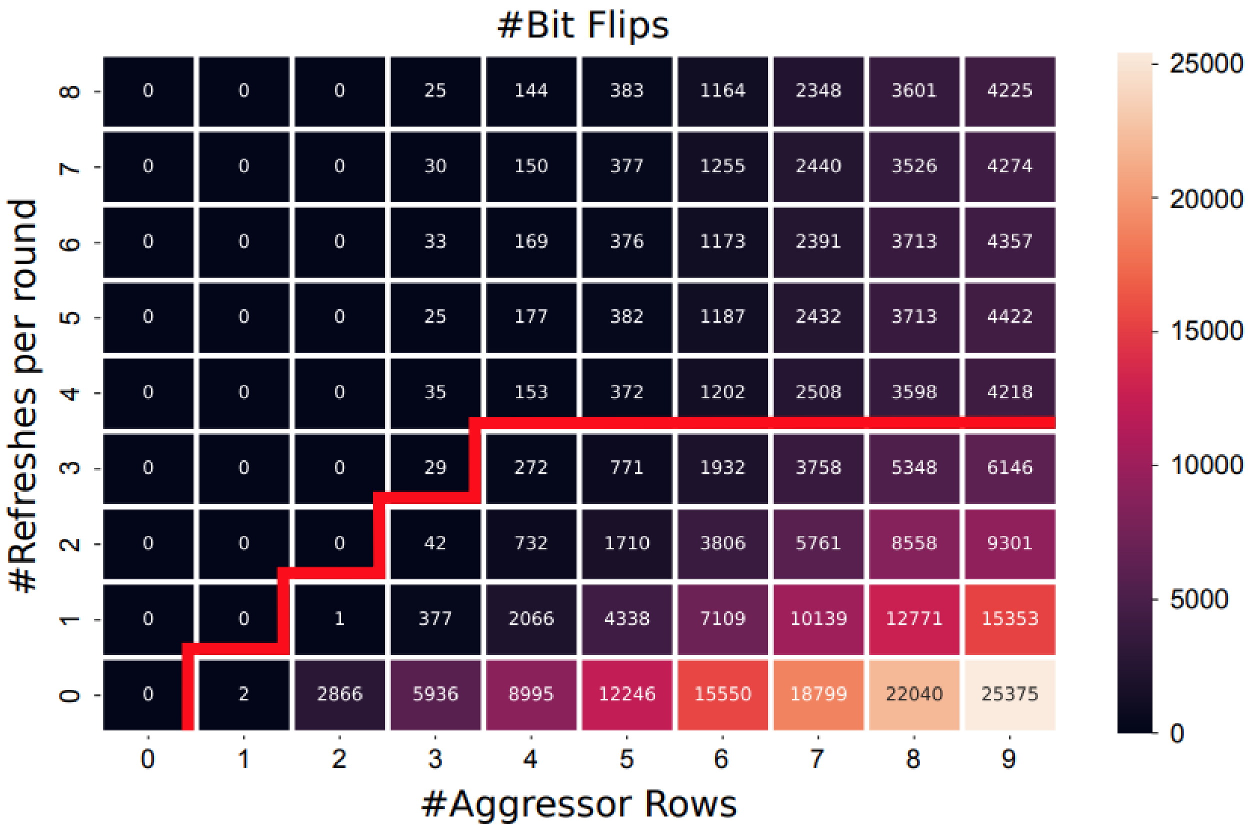
Case Study: Vendor C



	0	1	2	3	4	5	6	7	8	9
8	0	0	0	25	144	383	1164	2348	3601	4225
7	0	0	0	30	150	377	1255	2440	3526	4274
6	0	0	0	33	169	376	1173	2391	3713	4357
5	0	0	0	25	177	382	1187	2432	3713	4422
4	0	0	0	35	153	372	1202	2508	3598	4218
3	0	0	0	29	272	771	1932	3758	5348	6146
2	0	0	0	42	732	1710	3806	5761	8558	9301
1	0	0	1	377	2066	4338	7109	10139	12771	15353
0	0	2	2866	5936	8995	12246	15550	18799	22040	25375

4. Sweeping the number of refresh operations and aggressor rows while hammering reveals the sampler size

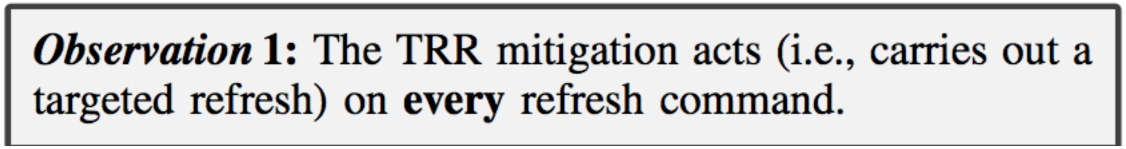
Many-Sided Hammering



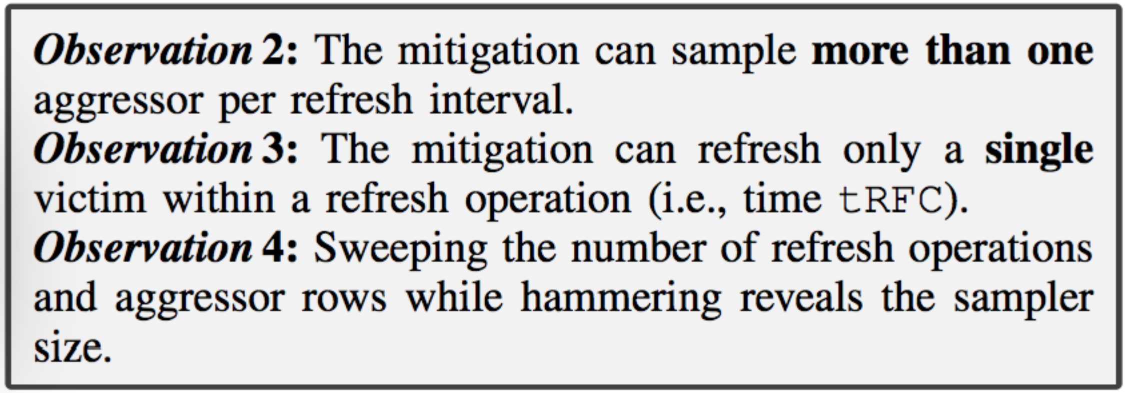
#Refreshes per round	0	1	2	3	4	5	6	7	8	9
8	0	0	0	25	144	383	1164	2348	3601	4225
7	0	0	0	30	150	377	1255	2440	3526	4274
6	0	0	0	33	169	376	1173	2391	3713	4357
5	0	0	0	25	177	382	1187	2432	3713	4422
4	0	0	0	35	153	372	1202	2508	3598	4218
3	0	0	0	29	272	771	1932	3758	5348	6146
2	0	0	0	42	732	1710	3806	5761	8558	9301
1	0	0	1	377	2066	4338	7109	10139	12771	15353
0	0	2	2866	5936	8995	12246	15550	18799	22040	25375

Fig. 9: Refreshes vs. Bit Flips. Module $\mathcal{C}_{12}$: Number of bit flips detected when sending r refresh commands to the module. We report this for different number of aggressor rows (n). For example, when hammering 5 rows, followed by sending 2 refreshes, we find 1,710 bit flips. This figure shows that the number of bit flips stabilizes for $r \geq 4$, implying that the size of the sampler may be 4.

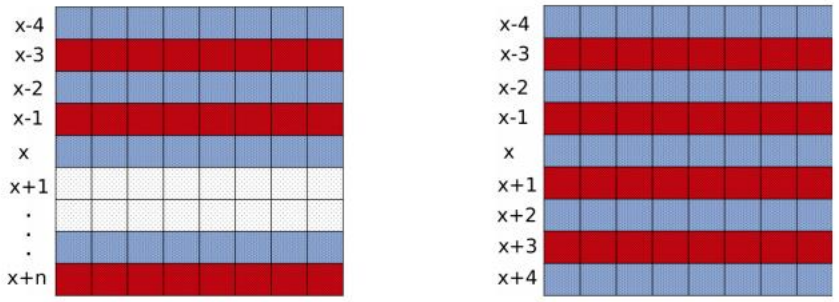
Some Observations



***Observation* 1:** The TRR mitigation acts (i.e., carries out a targeted refresh) on **every** refresh command.



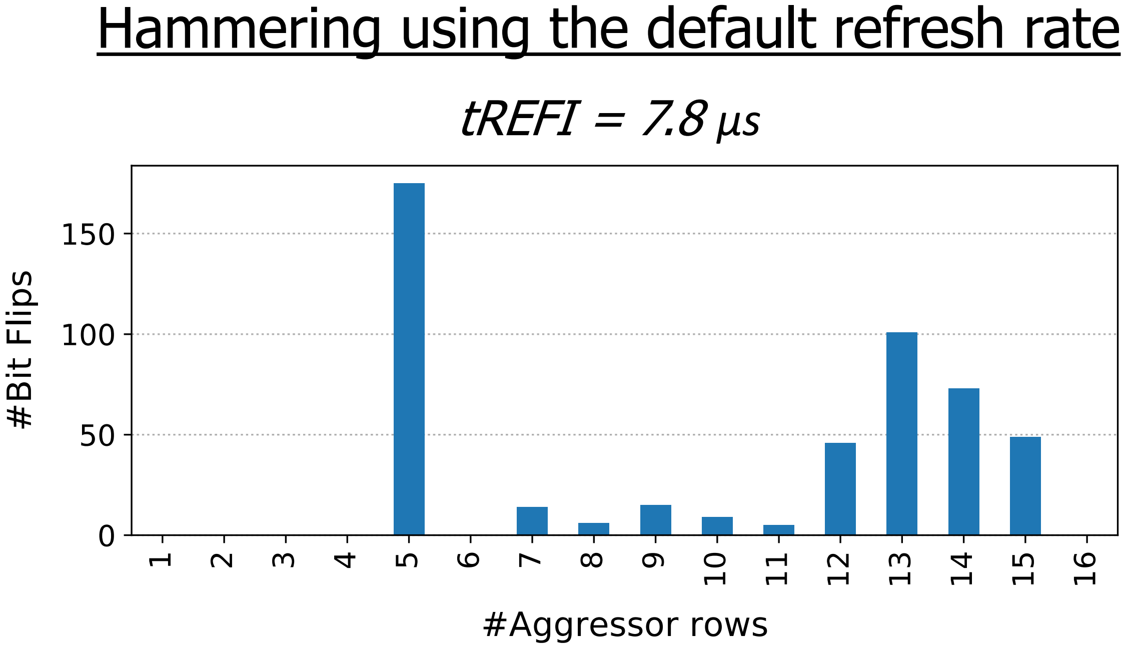
***Observation* 2:** The mitigation can sample **more than one** aggressor per refresh interval.
***Observation* 3:** The mitigation can refresh only a **single** victim within a refresh operation (i.e., time `tRFC`).
***Observation* 4:** Sweeping the number of refresh operations and aggressor rows while hammering reveals the sampler size.



(a) Assisted double-sided **(b)** 4-sided

Fig. 12: Hammering patterns discovered by *TRRespass*. Aggressor rows are in red (🟥) and victim rows are in blue (🟦).

Case Study: Vendor C



BitFlips vs. Number of Aggressor Rows

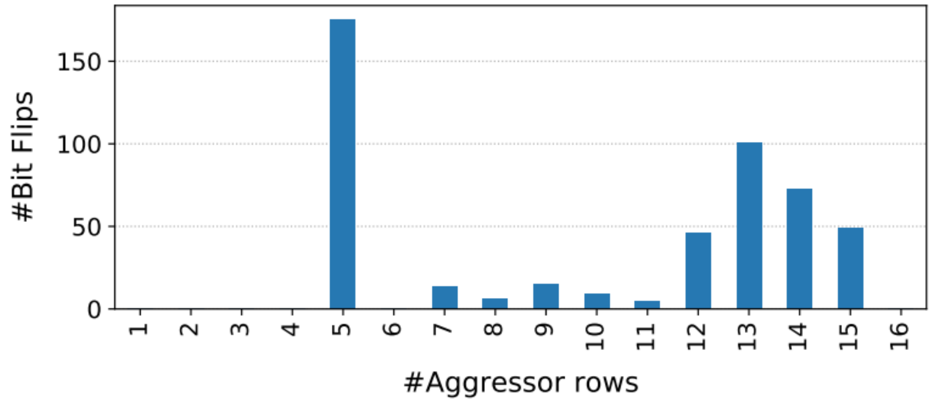


Fig. 10: Bit flips vs. number of aggressor rows. Module $\mathcal{C}_{12}$: Number of bit flips in bank 0 as we vary the number of aggressor rows. Using SoftMC, we refresh DRAM with standard `tREFI` and run the tests until each aggressor rows is hammered 500K times.

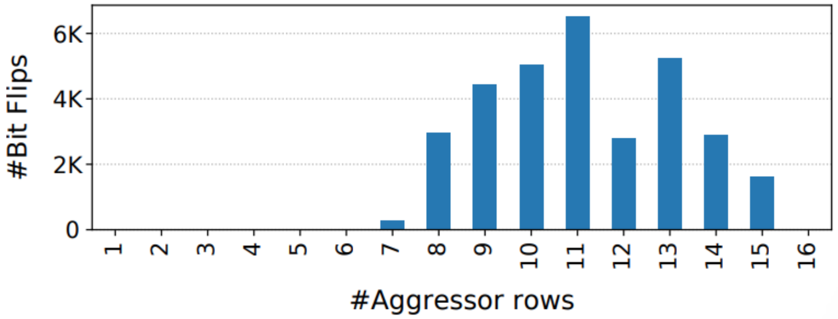


Fig. 11: Bit flips vs. number of aggressor rows. Module $\mathcal{A}_{15}$: Number of bit flips in bank 0 as we vary the number of aggressor rows. Using SoftMC, we refresh DRAM with standard `tREFI` and run the tests until each aggressor rows is hammered 500K times.

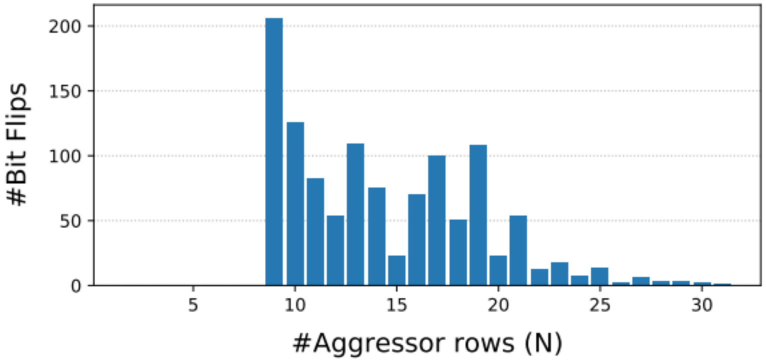


Fig. 13: Bit flips vs. number of aggressor rows. Module $\mathcal{A}_{10}$: Number of bit flips triggered with *N-sided* RowHammer for varying number of *N* on Intel Core i7-7700K. Each aggressor row is one row away from the closest aggressor row (i.e., VAVAVA... configuration) and aggressor rows are hammered in a round-robin fashion.

TRRespass Vulnerable DRAM Modules

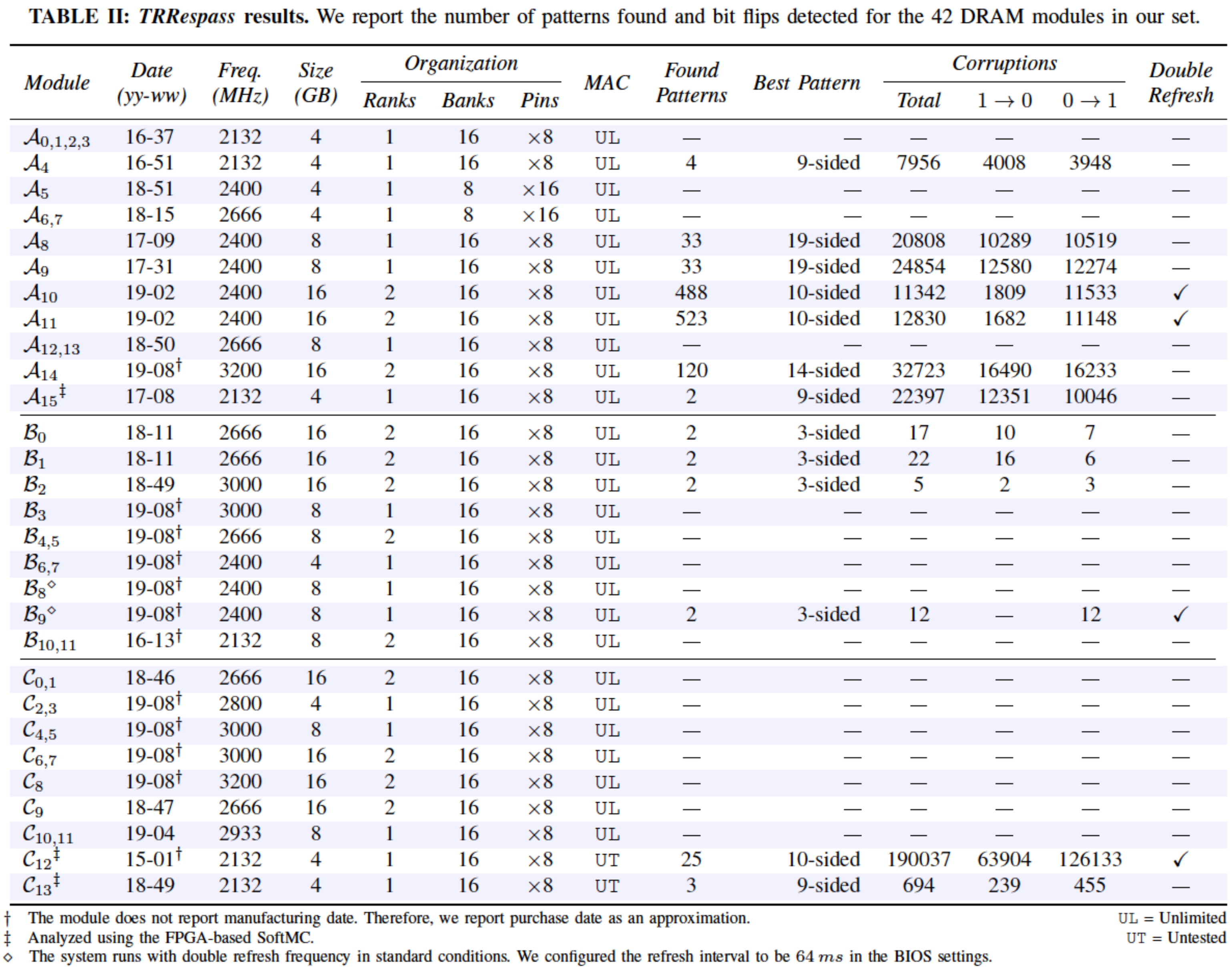


TABLE II: *TRRespass* results. We report the number of patterns found and bit flips detected for the 42 DRAM modules in our set.

Module	*Date* (yy-ww)	*Freq.* (MHz)	*Size* (GB)	*Organization*			*MAC*	*Found Patterns*	*Best Pattern*	*Corruptions*			*Double Refresh*
				Ranks	*Banks*	*Pins*				*Total*	1 → 0	0 → 1	
$\mathcal{A}_{0,1,2,3}$	16-37	2132	4	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{A}_{4}$	16-51	2132	4	1	16	×8	UL	4	9-sided	7956	4008	3948	—
$\mathcal{A}_{5}$	18-51	2400	4	1	8	×16	UL	—	—	—	—	—	—
$\mathcal{A}_{6,7}$	18-15	2666	4	1	8	×16	UL	—	—	—	—	—	—
$\mathcal{A}_{8}$	17-09	2400	8	1	16	×8	UL	33	19-sided	20808	10289	10519	—
$\mathcal{A}_{9}$	17-31	2400	8	1	16	×8	UL	33	19-sided	24854	12580	12274	—
$\mathcal{A}_{10}$	19-02	2400	16	2	16	×8	UL	488	10-sided	11342	1809	11533	✓
$\mathcal{A}_{11}$	19-02	2400	16	2	16	×8	UL	523	10-sided	12830	1682	11148	✓
$\mathcal{A}_{12,13}$	18-50	2666	8	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{A}_{14}$	19-08†	3200	16	2	16	×8	UL	120	14-sided	32723	16490	16233	—
$\mathcal{A}_{15}$‡	17-08	2132	4	1	16	×8	UL	2	9-sided	22397	12351	10046	—
$\mathcal{B}_{0}$	18-11	2666	16	2	16	×8	UL	2	3-sided	17	10	7	—
$\mathcal{B}_{1}$	18-11	2666	16	2	16	×8	UL	2	3-sided	22	16	6	—
$\mathcal{B}_{2}$	18-49	3000	16	2	16	×8	UL	2	3-sided	5	2	3	—
$\mathcal{B}_{3}$	19-08†	3000	8	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{B}_{4,5}$	19-08†	2666	8	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{B}_{6,7}$	19-08†	2400	4	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{B}_{8}$⋄	19-08†	2400	8	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{B}_{9}$⋄	19-08†	2400	8	1	16	×8	UL	2	3-sided	12	—	12	✓
$\mathcal{B}_{10,11}$	16-13†	2132	8	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{0,1}$	18-46	2666	16	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{2,3}$	19-08†	2800	4	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{4,5}$	19-08†	3000	8	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{6,7}$	19-08†	3000	16	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{8}$	19-08†	3200	16	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{9}$	18-47	2666	16	2	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{10,11}$	19-04	2933	8	1	16	×8	UL	—	—	—	—	—	—
$\mathcal{C}_{12}$‡	15-01†	2132	4	1	16	×8	UT	25	10-sided	190037	63904	126133	✓
$\mathcal{C}_{13}$‡	18-49	2132	4	1	16	×8	UT	3	9-sided	694	239	455	—

† The module does not report manufacturing date. Therefore, we report purchase date as an approximation.
‡ Analyzed using the FPGA-based SoftMC.
⋄ The system runs with double refresh frequency in standard conditions. We configured the refresh interval to be 64 ms in the BIOS settings.

UL = Unlimited
UT = Untested

TRRespass Vulnerable Mobile Phones

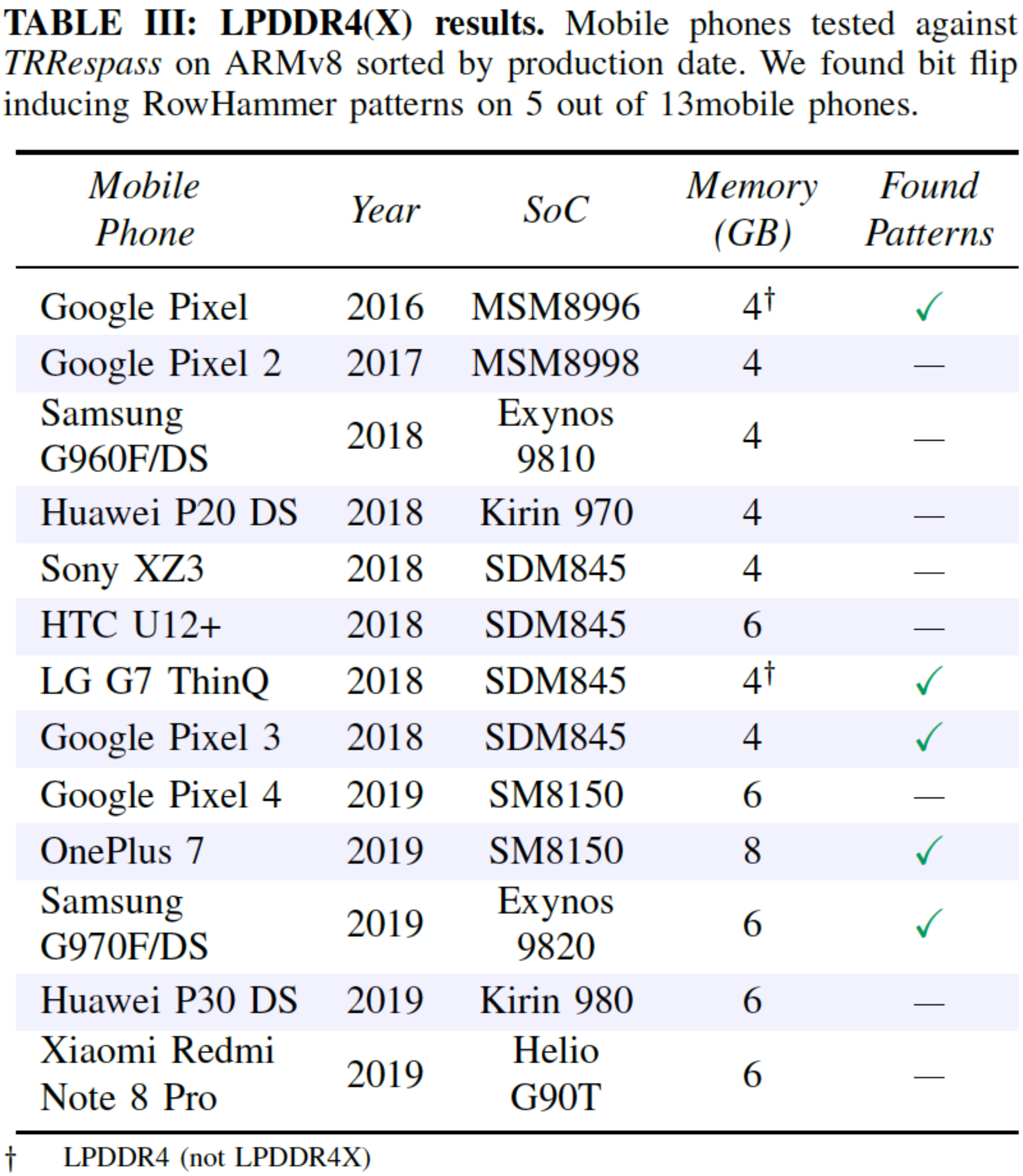


TABLE III: LPDDR4(X) results. Mobile phones tested against *TRRespass* on ARMv8 sorted by production date. We found bit flip inducing RowHammer patterns on 5 out of 13mobile phones.

Mobile Phone	*Year*	*SoC*	*Memory (GB)*	*Found Patterns*
Google Pixel	2016	MSM8996	4[†]	✓
Google Pixel 2	2017	MSM8998	4	—
Samsung G960F/DS	2018	Exynos 9810	4	—
Huawei P20 DS	2018	Kirin 970	4	—
Sony XZ3	2018	SDM845	4	—
HTC U12+	2018	SDM845	6	—
LG G7 ThinQ	2018	SDM845	4[†]	✓
Google Pixel 3	2018	SDM845	4	✓
Google Pixel 4	2019	SM8150	6	—
OnePlus 7	2019	SM8150	8	✓
Samsung G970F/DS	2019	Exynos 9820	6	✓
Huawei P30 DS	2019	Kirin 980	6	—
Xiaomi Redmi Note 8 Pro	2019	Helio G90T	6	—

[†] LPDDR4 (not LPDDR4X)

TRRespass Based RowHammer Attack

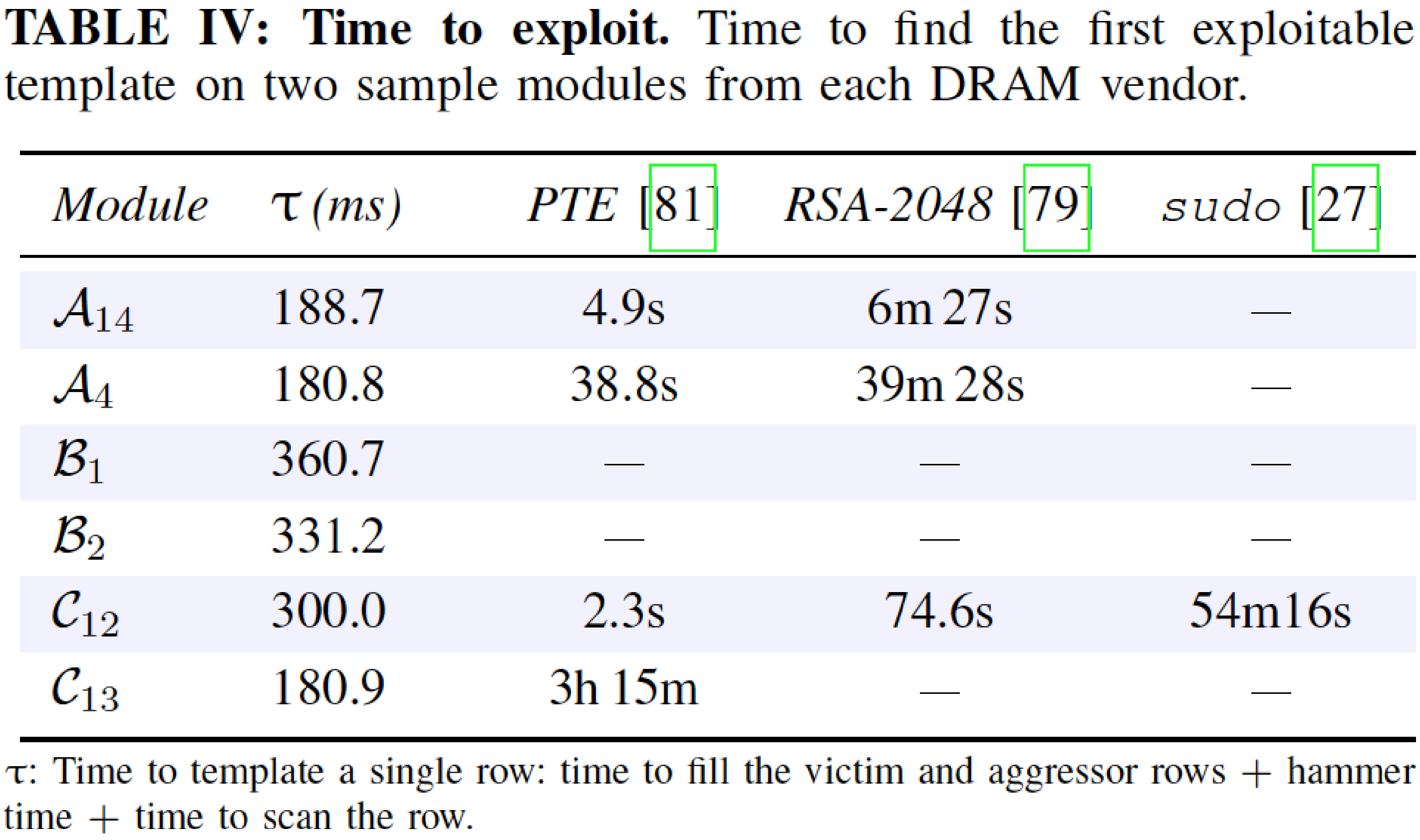


TABLE IV: Time to exploit. Time to find the first exploitable template on two sample modules from each DRAM vendor.

Module	τ *(ms)*	*PTE* [81]	*RSA-2048* [79]	sudo [27]
$\mathcal{A}_{14}$	188.7	4.9s	6m 27s	—
$\mathcal{A}_{4}$	180.8	38.8s	39m 28s	—
$\mathcal{B}_{1}$	360.7	—	—	—
$\mathcal{B}_{2}$	331.2	—	—	—
$\mathcal{C}_{12}$	300.0	2.3s	74.6s	54m16s
$\mathcal{C}_{13}$	180.9	3h 15m	—	—

τ: Time to template a single row: time to fill the victim and aggressor rows + hammer time + time to scan the row.

TRRespass Key Results

- 13 out of 42 tested DDR4 DRAM modules are vulnerable
  - From all 3 major manufacturers
  - 3-, 9-, 10-, 14-, 19-sided hammer attacks needed

- 5 out of 13 mobile phones tested vulnerable
  - From 4 major manufacturers
  - With LPDDR4(X) DRAM chips

- These results are scratching the surface
  - TRRespass tool is not exhaustive
  - There is a lot of room for uncovering more vulnerable chips and phones

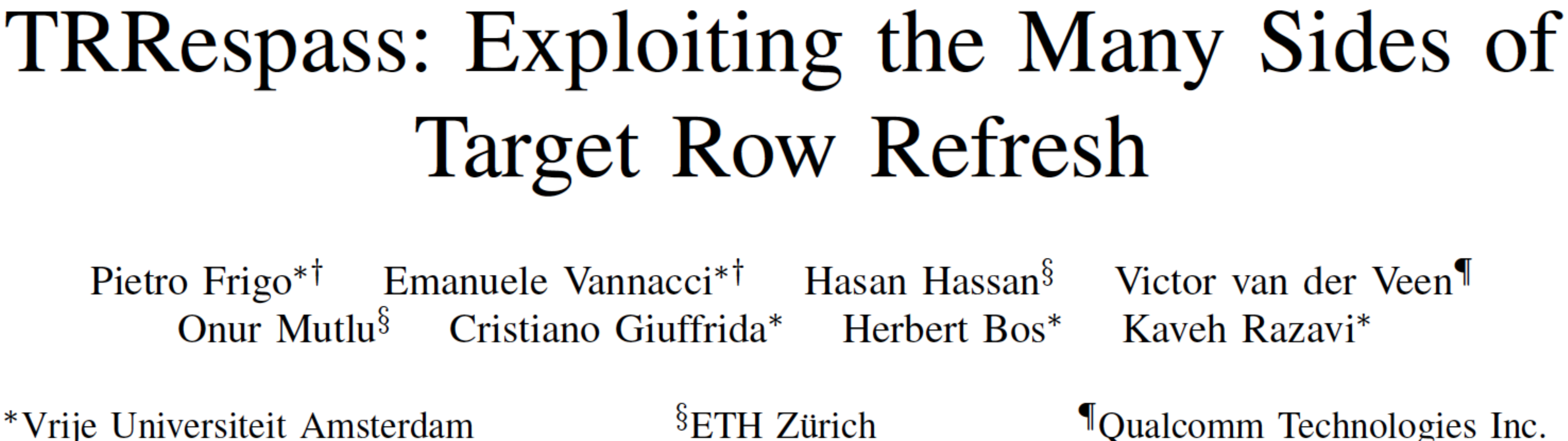
TRRespass Key Takeaways

RowHammer is still
an open problem

Security by obscurity
is likely not a good solution

More on TRRespass

- 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

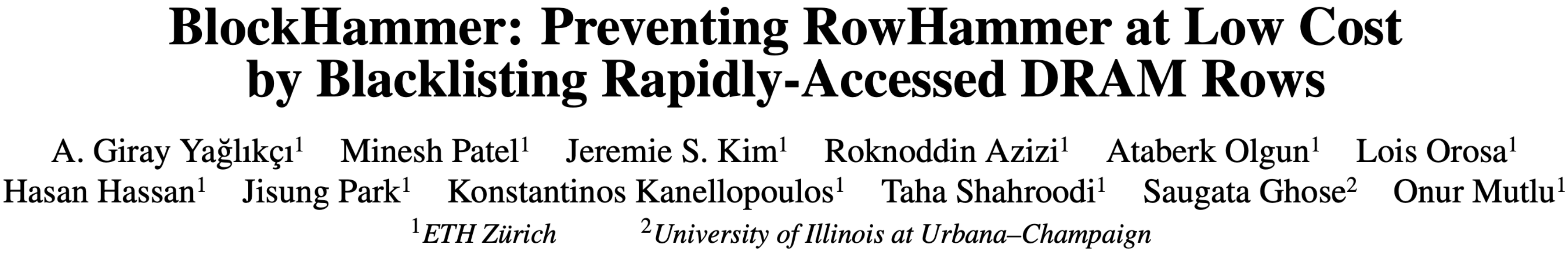
Pietro Frigo*† Emanuele Vannacci*† Hasan Hassan§ Victor van der Veen¶
Onur Mutlu§ Cristiano Giuffrida* Herbert Bos* Kaveh Razavi*

*Vrije Universiteit Amsterdam §ETH Zürich ¶Qualcomm Technologies Inc.

BlockHammer Solution

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)]
  [Intel Hardware Security Academic Awards Short Talk Slides (pptx) (pdf)]
  [Talk Video (22 minutes)]
  [Short Talk Video (7 minutes)]
  [Intel Hardware Security Academic Awards Short Talk Video (2 minutes)]
  [BlockHammer Source Code]
  Intel Hardware Security Academic Award Finalist (one of 4 finalists out of 34 nominations)



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

A. Giray Yağlıkçı[1] Minesh Patel[1] Jeremie S. Kim[1] Roknoddin Azizi[1] Ataberk Olgun[1] Lois Orosa[1]
Hasan Hassan[1] Jisung Park[1] Konstantinos Kanellopoulos[1] Taha Shahroodi[1] Saugata Ghose[2] Onur Mutlu[1]
[1]*ETH Zürich* [2]*University of Illinois at Urbana–Champaign*

BlockHammer

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

Abdullah Giray Yağlıkçı

Minesh Patel Jeremie S. Kim Roknoddin Azizi

Ataberk Olgun Lois Orosa Hasan Hassan Jisung Park

Konstantinos Kanellopoulos Taha Shahroodi

Saugata Ghose* Onur Mutlu

SAFARI

ETH zürich

*UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Executive Summary

- **Motivation**: RowHammer is a worsening DRAM reliability and security problem
- **Problem**: Mitigation mechanisms have limited support for current/future chips
  - **Scalability** with worsening RowHammer vulnerability
  - **Compatibility** with commodity DRAM chips
- **Goal**: **Efficiently** and **scalably** prevent RowHammer bit-flips **without** knowledge of or modifications to DRAM internals
- **Key Idea:** Selectively throttle memory accesses that may cause RowHammer bit-flips
- **Mechanism**: BlockHammer
  - **Tracks** activation rates of all rows by using area-efficient Bloom filters
  - **Throttles** row activations that could cause RowHammer bit flips
  - **Identifies and throttles** threads that perform RowHammer attacks
- **Scalability with Worsening RowHammer Vulnerability:**
  - **Competitive** with state-of-the-art mechanisms **when there is no attack**
  - **Superior** performance and DRAM energy **when a RowHammer attack is present**
- **Compatibility with Commodity DRAM Chips:**
  - **No proprietary information** of DRAM internals
  - **No modifications** to DRAM circuitry

Outline

DRAM and RowHammer Background

Motivation and Goal

BlockHammer

- RowBlocker
- AttackThrottler

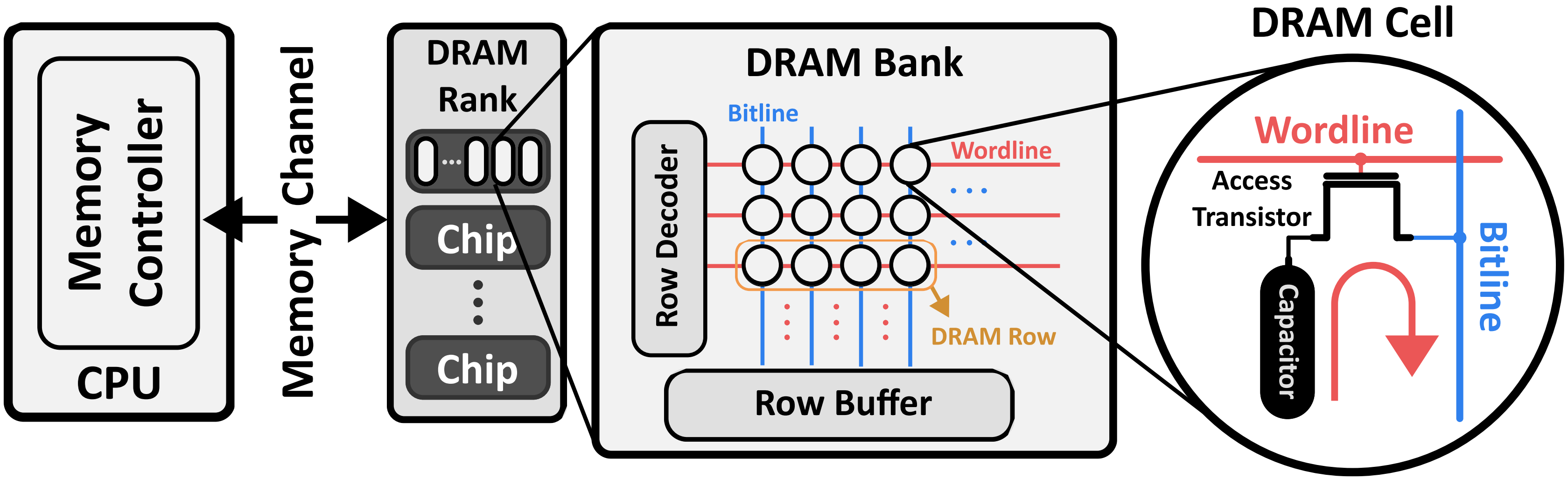
Evaluation

Conclusion

Outline

- DRAM and RowHammer Background
- Motivation and Goal
- BlockHammer
  - RowBlocker
  - AttackThrottler
- Evaluation
- Conclusion

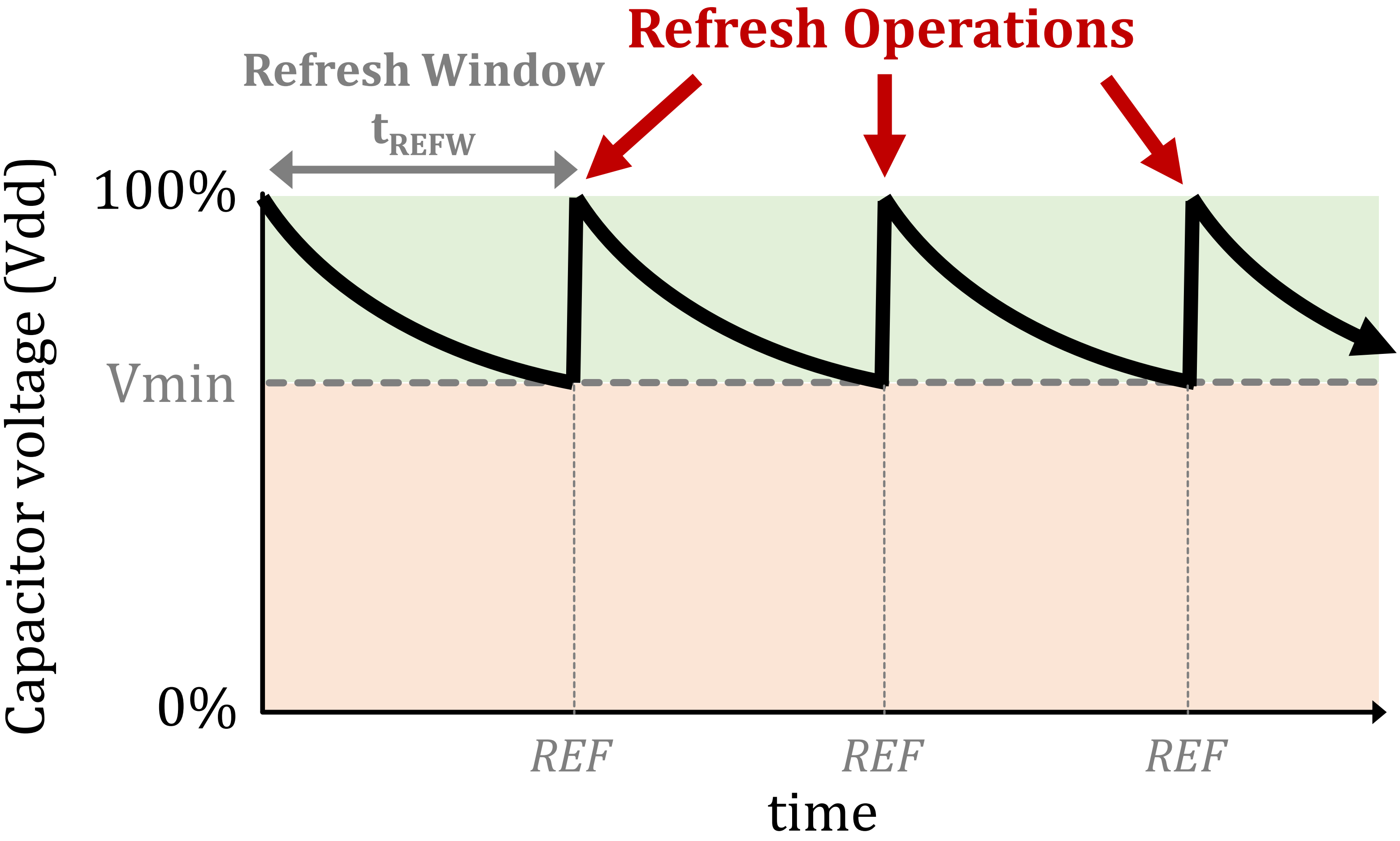
Organizing and Accessing DRAM Cells



A DRAM cell consists of a **capacitor** and an **access transistor**

A row needs to be **activated** to access its content

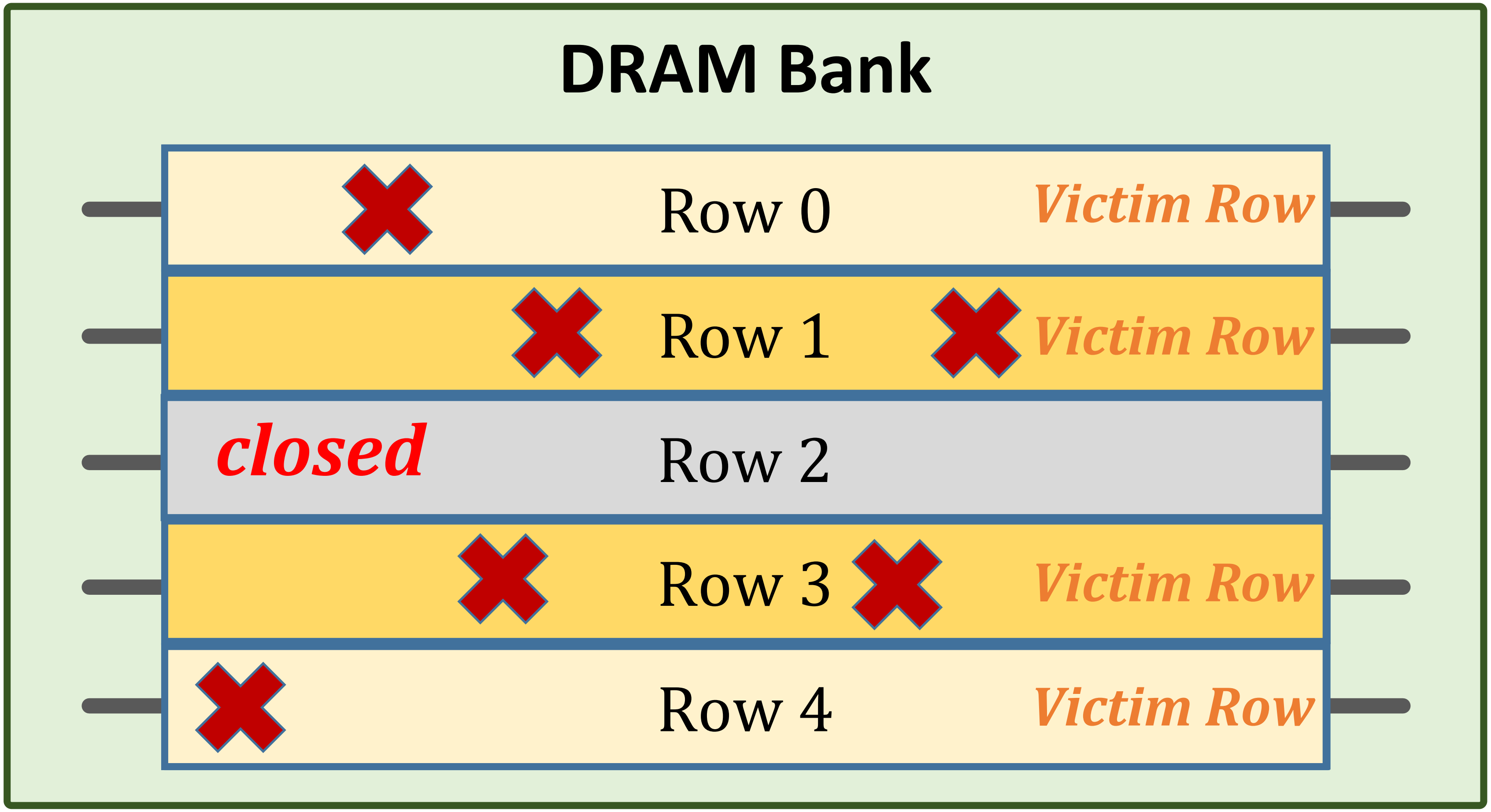
DRAM Refresh



Periodic **refresh operations** preserve stored data

[Patel+ ISCA'17, Kim+ ISCA'20]

The RowHammer Phenomenon



Repeatedly **opening** (activating) and **closing** (precharging) a DRAM row causes **RowHammer bit flips** in nearby cells

SAFARI

[Kim+ ISCA'20]

Outline

DRAM and RowHammer Background

Motivation and Goal

BlockHammer

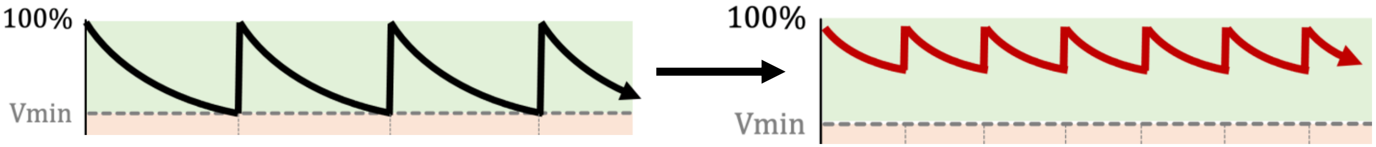
- RowBlocker
- AttackThrottler

Evaluation

Conclusion

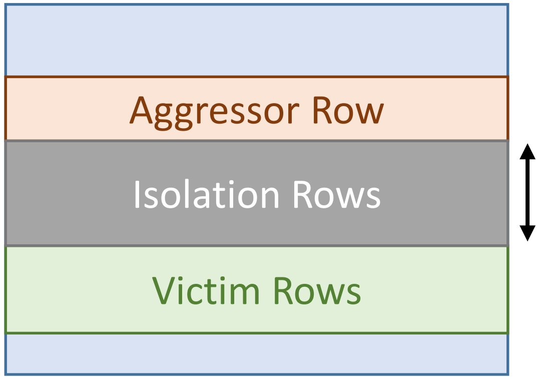
RowHammer Mitigation Approaches

- Increased refresh rate



REF-to-REF time reduces
Fewer activations can fit

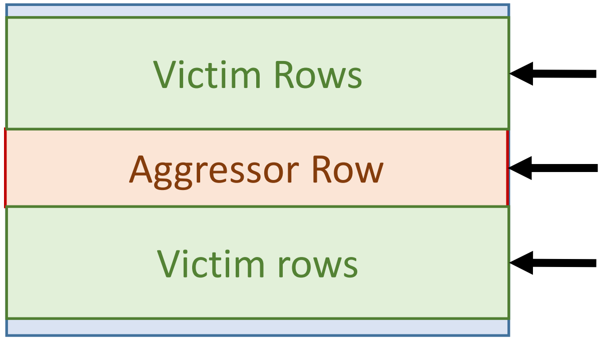
- Physical isolation



Aggressor Row
Isolation Rows
Victim Rows

Large-enough distance

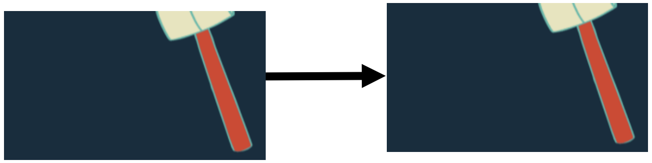
- Reactive refresh



Victim Rows
Aggressor Row
Victim rows

Refresh
Rapidly activated (hammered)
Refresh

- Proactive throttling



Fewer activations can be performed

Two Key Challenges

1 **Scalability**
with worsening RowHammer vulnerability

2 **Compatibility**
with commodity DRAM chips

Scalability

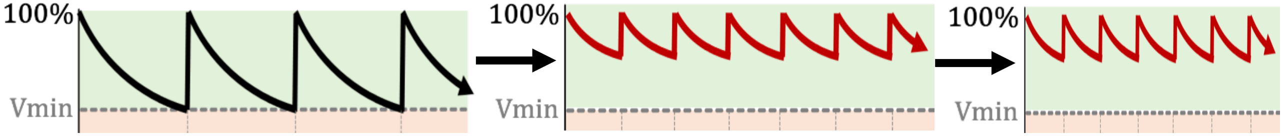
with Worsening RowHammer Vulnerability

- DRAM chips are more vulnerable to RowHammer today
- RowHammer bit-flips occur at much lower activation counts (more than an order of magnitude decrease):
    - 139.2K [Y. Kim+, ISCA 2014]
    - 9.6K [J. S. Kim+, ISCA 2020]
- RowHammer blast radius has increased by 33%:
    - 9 rows [Y. Kim+, ISCA 2014]
    - 12 rows [J. S. Kim+, ISCA 2020]
- In-DRAM mitigation mechanisms are ineffective [Frigo+, S&P 2020]

RowHammer is a more serious problem than ever

Mitigation Approaches
with Worsening RowHammer Vulnerability

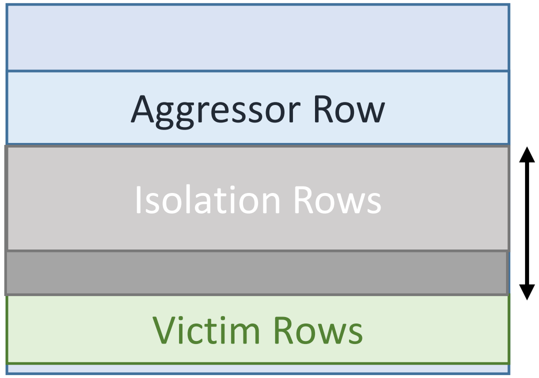
- Increased refresh rate



REF-to-REF time further reduces

Even fewer activations can fit

- Physical isolation



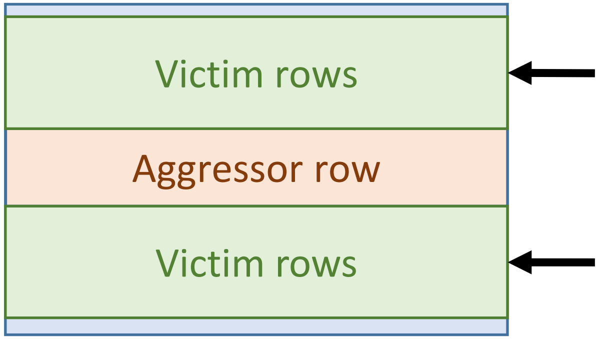
Aggressor Row

Isolation Rows

Victim Rows

Larger distance
more isolation rows

- Reactive refresh



Victim rows

Aggressor row

Victim rows

Refresh more frequently
Refresh more rows

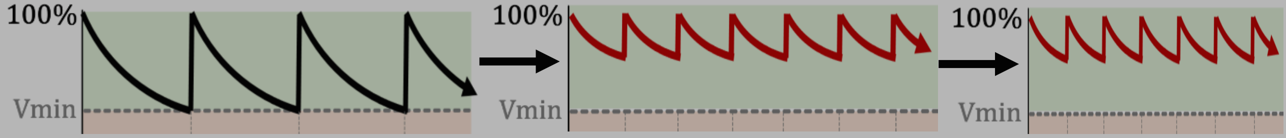
Refresh more frequently
Refresh more rows

- Proactive throttling

More aggressively throttles row activations

Mitigation Approaches
with Worsening RowHammer Vulnerability

- Increased refresh rate



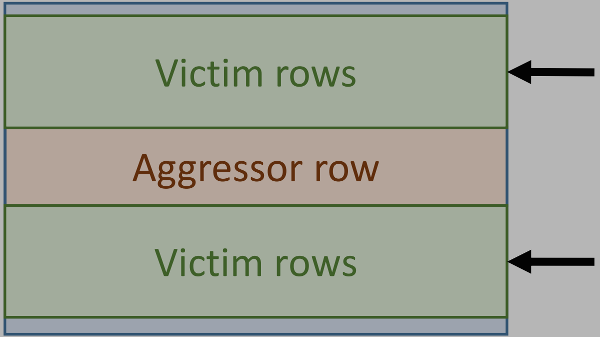
REF-to-REF time further reduces

Even fewer activations can fit

- Physical isolation

Mitigation mechanisms face the challenge of scalability with worsening RowHammer

- Reactive refresh



Refresh more frequently
Refresh more rows

Refresh more frequently
Refresh more rows

- Proactive throttling

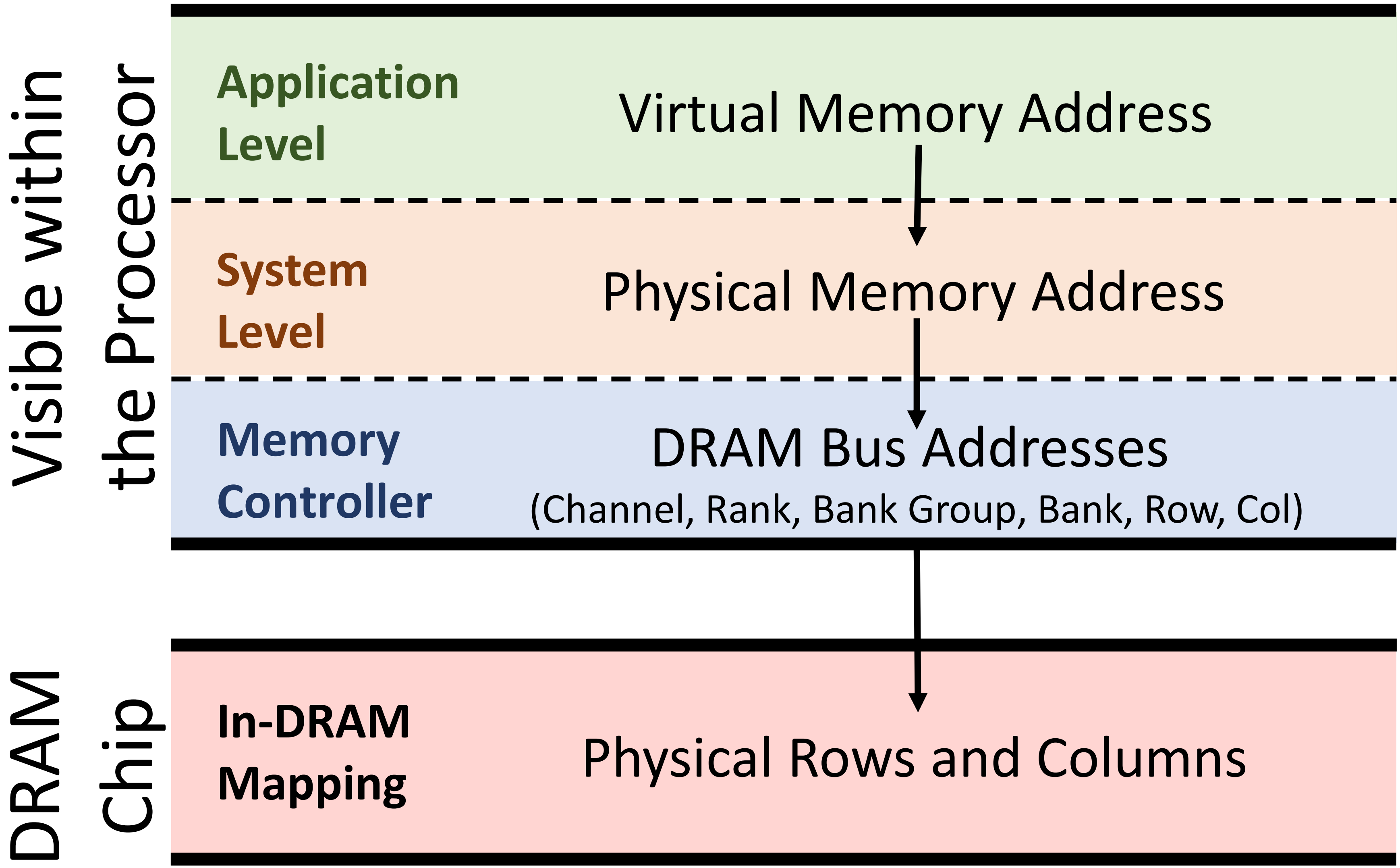
More aggressively throttles row activations

Two Key Challenges

1 **Scalability**
with worsening RowHammer vulnerability

2 **Compatibility**
with commodity DRAM chips

Compatibility
with Commodity DRAM Chips



Compatibility
with Commodity DRAM Chips

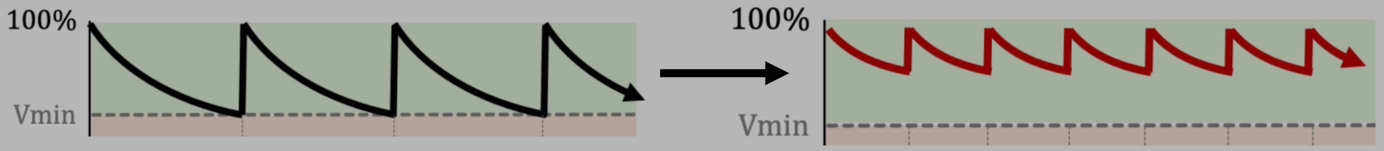
Vendors apply in-DRAM mapping for two reasons:

- **Design Optimizations:** By simplifying DRAM circuitry to provide better density, performance, and power
- **Yield Improvement:** By mapping faulty rows and columns to redundant ones
- In-DRAM mapping scheme includes insights into **chip design** and **manufacturing quality**

In-DRAM mapping is proprietary information

RowHammer Mitigation Approaches

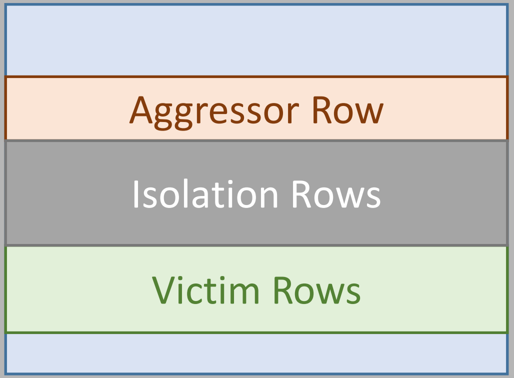
- Increased refresh rate



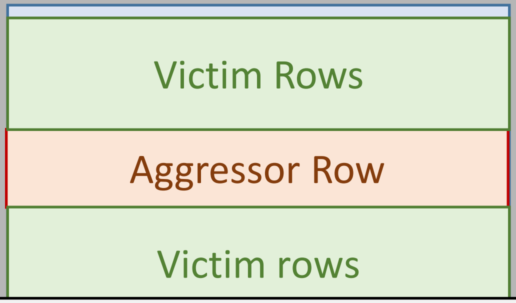
REF-to-REF time reduces

Fewer activations can fit

- Physical isolation



- Reactive refresh



Identifying ***victim*** and ***isolation*** rows requires ***proprietary*** knowledge of ***in-DRAM mapping***

Our Goal

To prevent RowHammer efficiently and scalably *without* knowledge of or modifications to DRAM internals

Outline

- DRAM and RowHammer Background
- Motivation and Goal
- BlockHammer
  - RowBlocker
  - AttackThrottler
- Evaluation
- Conclusion

BlockHammer

Key Idea

Selectively throttle memory accesses that may cause RowHammer bit-flips

BlockHammer

Overview of Approach

RowBlocker

Tracks row activation rates using area-efficient Bloom filters

Blacklists rows that are activated at a high rate

Throttles activations targeting a blacklisted row

No row can be activated at a high enough rate to induce bit-flips

AttackThrottler

Identifies threads that perform a RowHammer attack

Reduces memory bandwidth usage of identified threads

Greatly reduces the **performance degradation** and **energy wastage** a RowHammer attack inflicts on a system

Outline

DRAM and RowHammer Background

Motivation and Goal

BlockHammer

RowBlocker

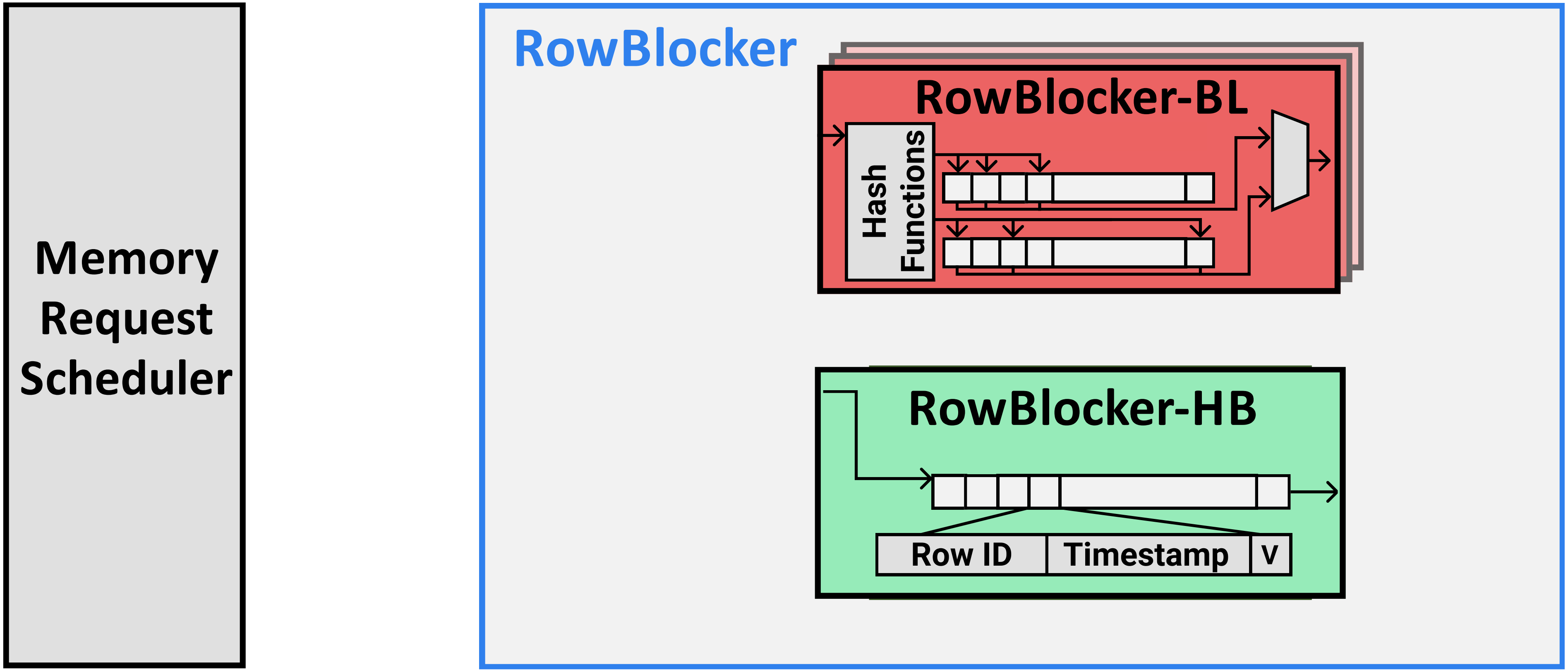
AttackThrottler

Evaluation

Conclusion

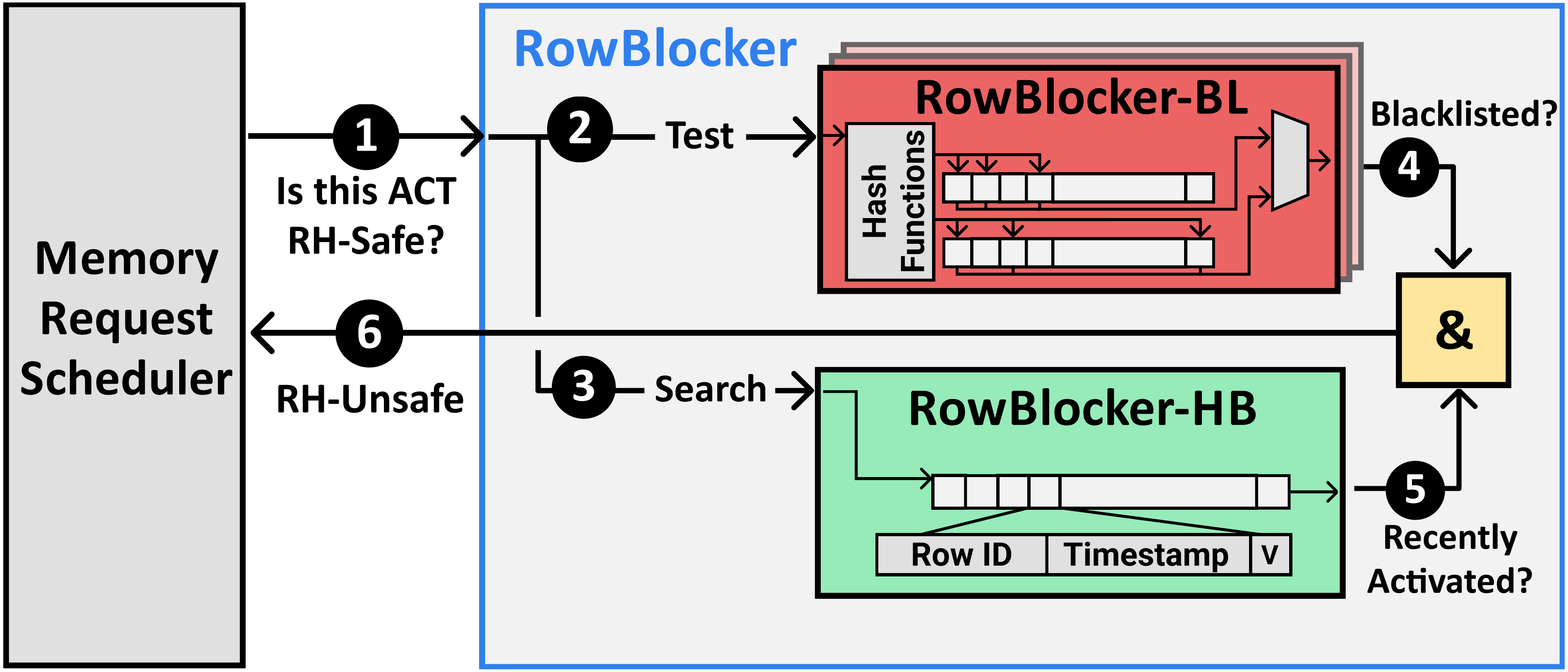
RowBlocker

- Modifies the memory request scheduler to throttle row activations
- **Blacklists** rows with a high activation rate and **delays** subsequent activations targeting blacklisted rows



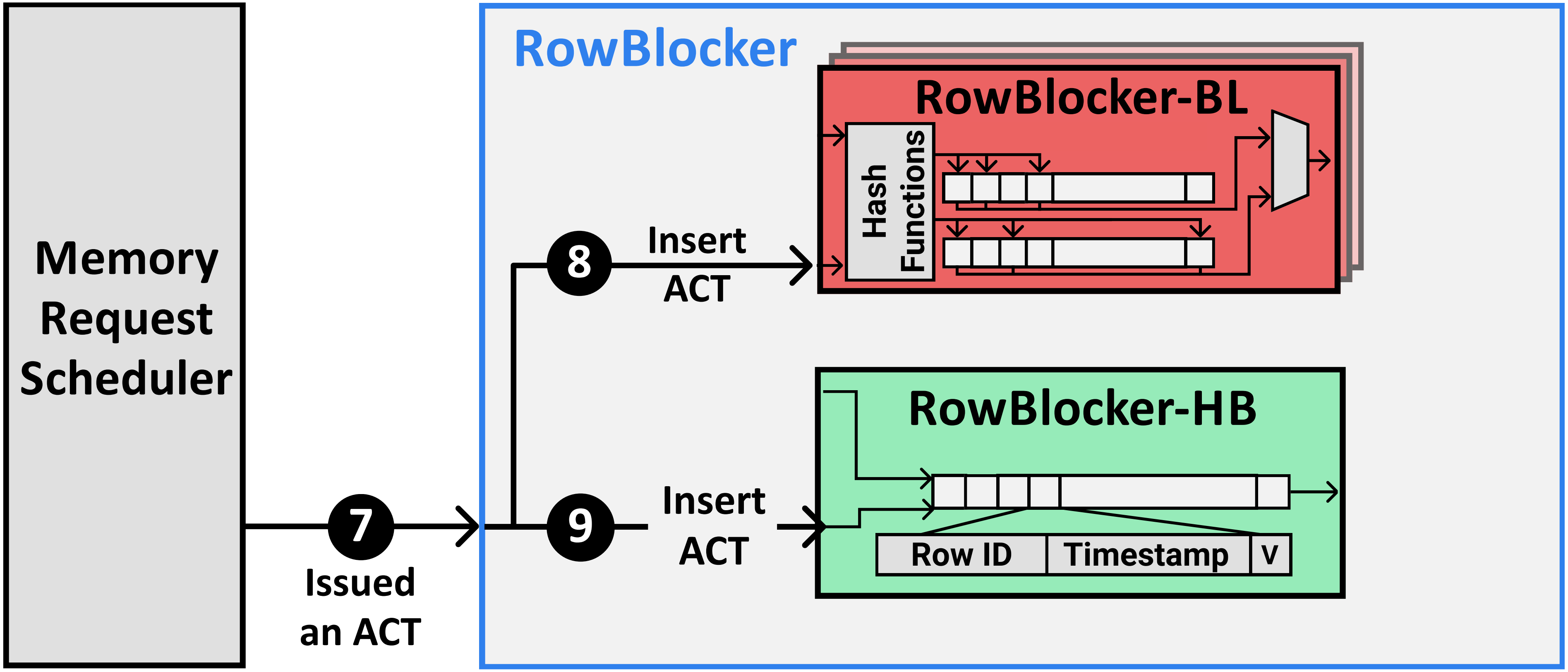
RowBlocker

- Blocks a row activation if the row is **both** blacklisted **and** recently activated



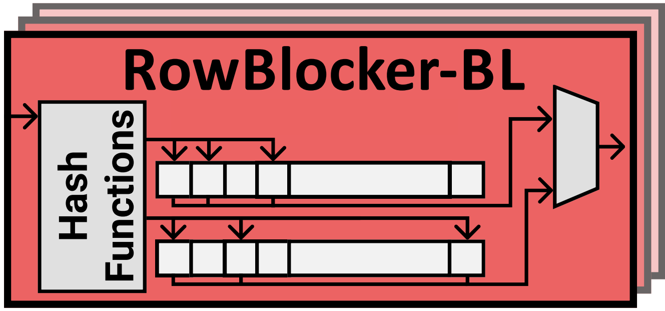
RowBlocker

- When a row activation is performed, both **RowBlocker-BL** and **RowBlocker-HB** are updated with the row activation information



RowBlocker-BL

Blacklisting Logic

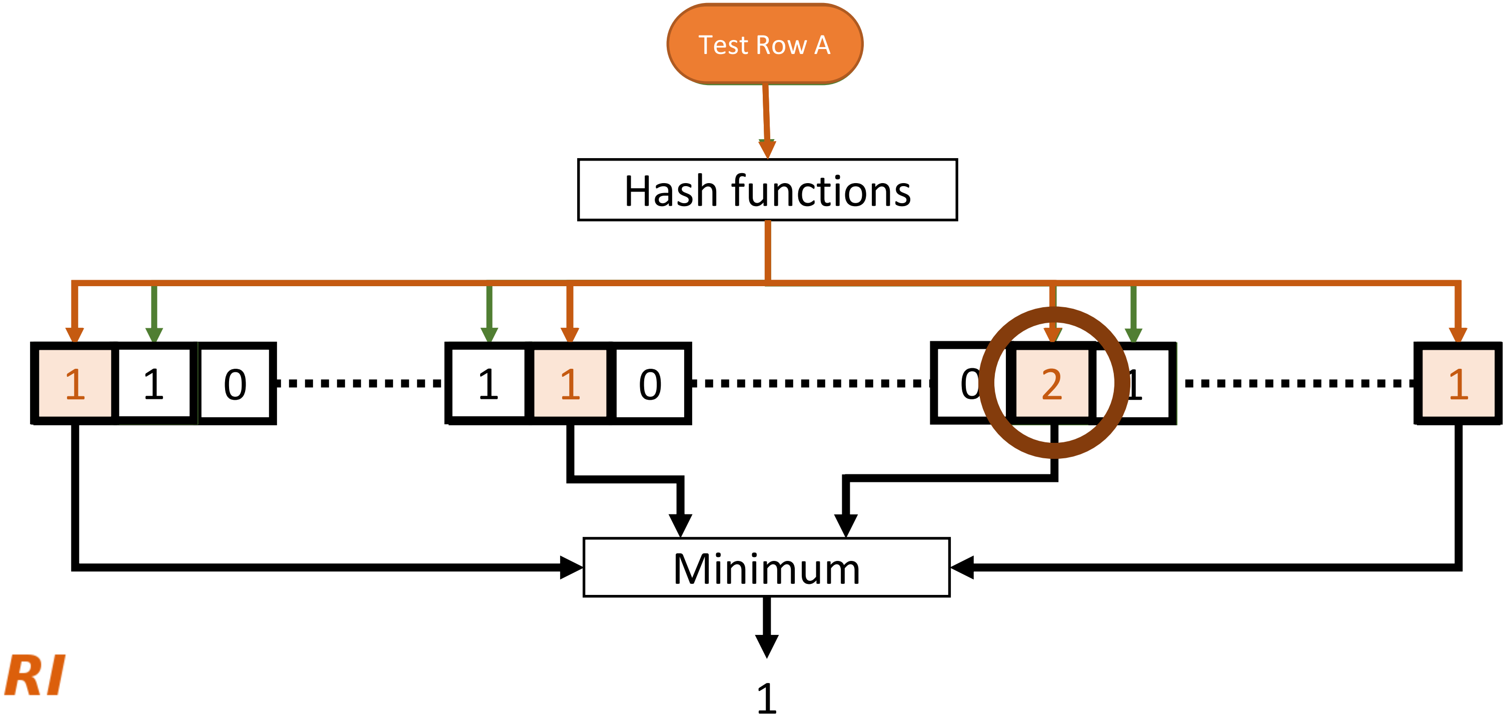


- Blacklists a row when the row's activation count in a time window exceeds a threshold

- Employs two counting Bloom filters for area-efficient activation rate tracking

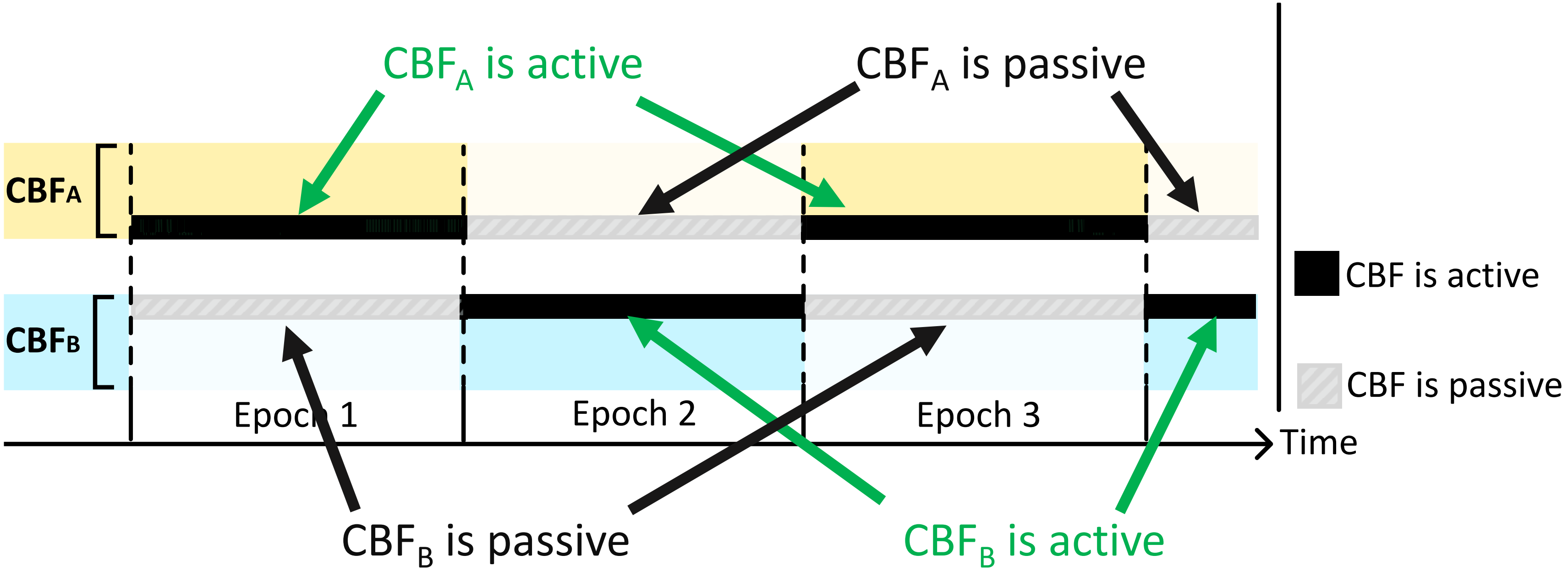
Counting Bloom Filters

- Blacklisting logic counts activations using counting Bloom filters
- A row's activation count
  - can be observed more than it is (false positive)
  - cannot be observed less than it is (no false negative)
- To avoid saturating counters, we use a time-interleaving approach



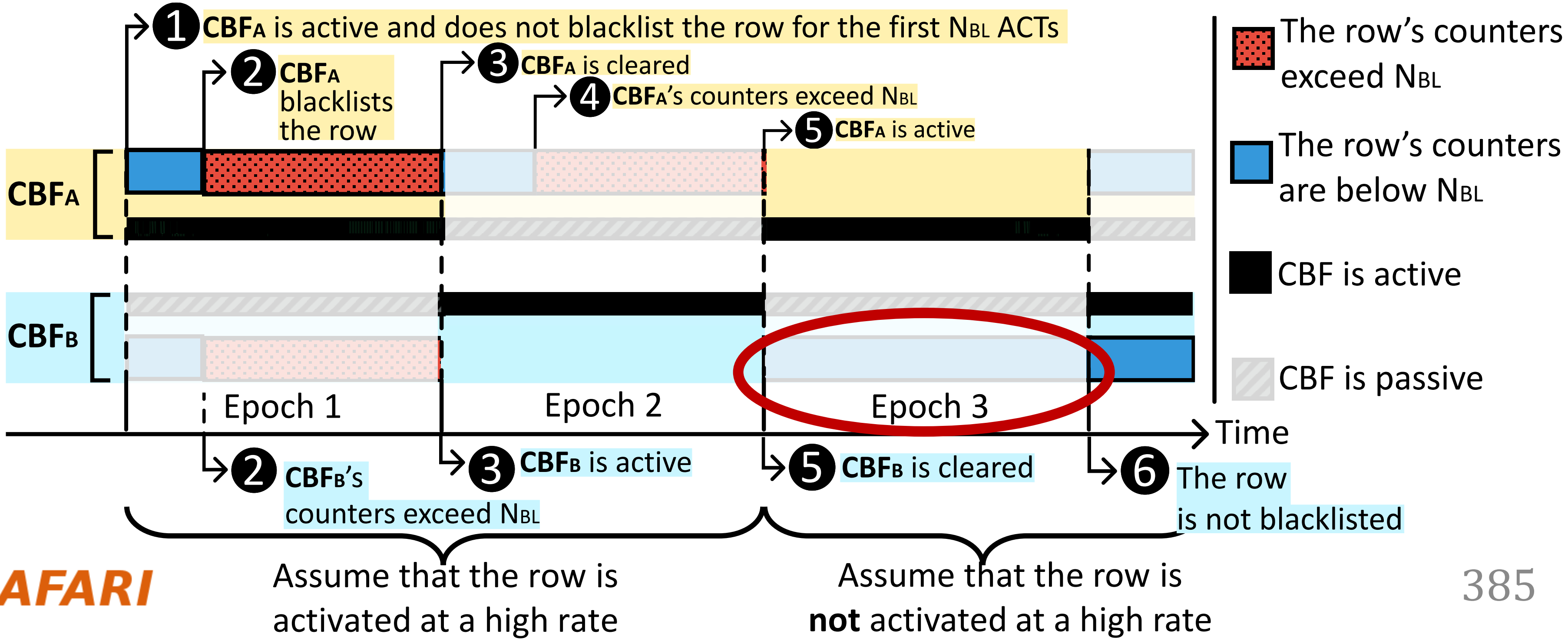
RowBlocker-BL
Blacklisting Logic

- Blacklisting logic employs two counting Bloom filters
- A new row activation is inserted in both filters
- Only one filter (active filter) responds to test queries
- The active filter changes at every epoch



RowBlocker-BL
Blacklisting Logic

- Blacklisting logic employs two counting Bloom filters
- A new row activation is inserted in both filters
- Only one filter (active filter) responds to test queries
- The active filter changes at every epoch
- Blacklists a row if its activation count reaches the blacklisting threshold (N_{BL})



1 CBF_A is active and does not blacklist the row for the first N_{BL} ACTs

2 CBF_A blacklists the row

3 CBF_A is cleared

4 CBF_A's counters exceed N_{BL}

5 CBF_A is active

2 CBF_B's counters exceed N_{BL}

3 CBF_B is active

5 CBF_B is cleared

6 The row is not blacklisted

🟥 The row's counters exceed N_{BL}

🟦 The row's counters are below N_{BL}

■ CBF is active

▨ CBF is passive

Assume that the row is activated at a high rate

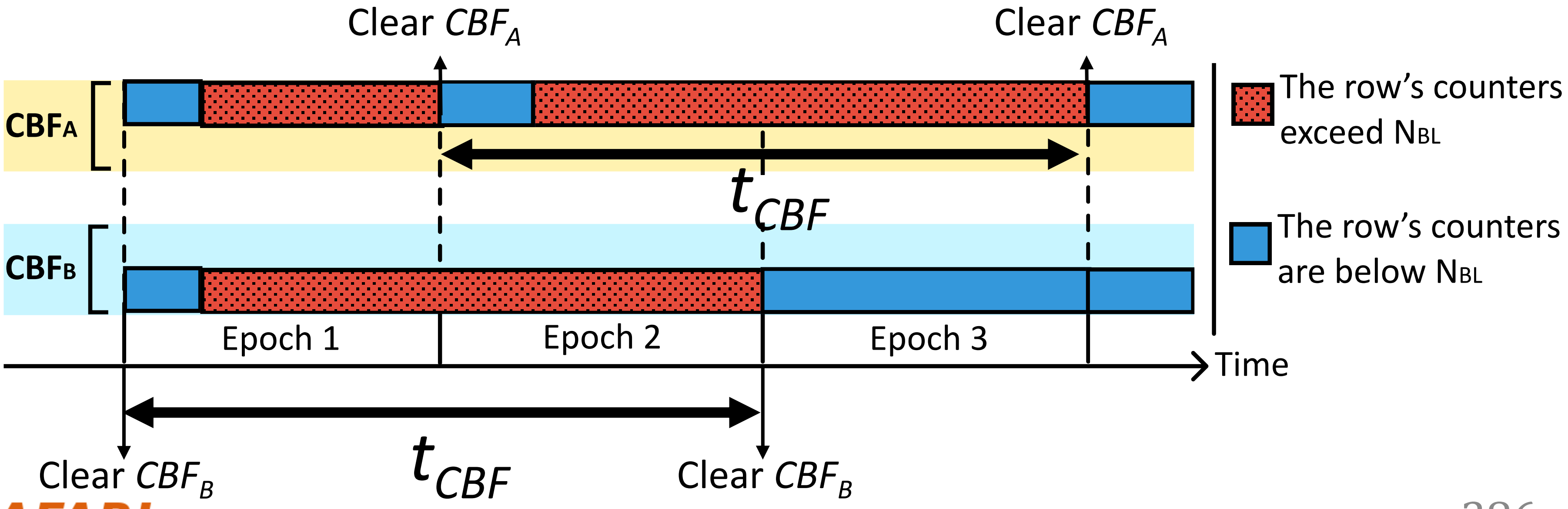
Assume that the row is **not** activated at a high rate

SAFARI

Limiting the Row Activation Rate

- The activation rate is **RowHammer-safe** if it is smaller than or equal to **RowHammer threshold (N_{RH})** activations in a **refresh window (t_{REFW})**
- RowBlocker limits the **activation count (N_{CBF})** in a **CBF's lifetime (t_{CBF})**

$$Activation\ Rate\ in\ a\ t_{CBF} \leq\ N_{RH}\ activations\ in\ a\ refresh\ window\ (t_{REFW})$$



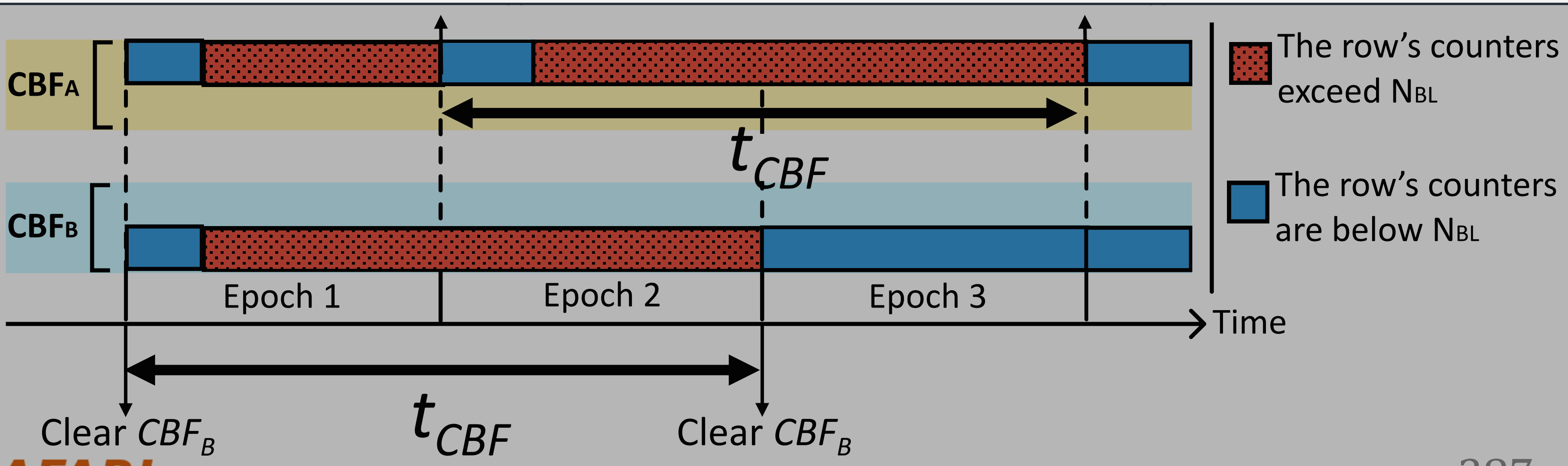
Limiting the Row Activation Rate

- The activation rate is **RowHammer-safe** if it is smaller than or equal to **RowHammer threshold (N_{RH})** activations in a **refresh window (t_{REFW})**
- RowBlocker limits the **activation count (N_{CBF})** in a **CBF's lifetime (t_{CBF})**

$$Activation\ Rate\ in\ a\ t_{CBF} \leq\ N_{RH}\ activations\ in\ a\ refresh\ window\ (t_{REFW})$$

RowHammer Safety Constraint

$$N_{CBF}/t_{CBF} \leq N_{RH}/t_{REFW}$$

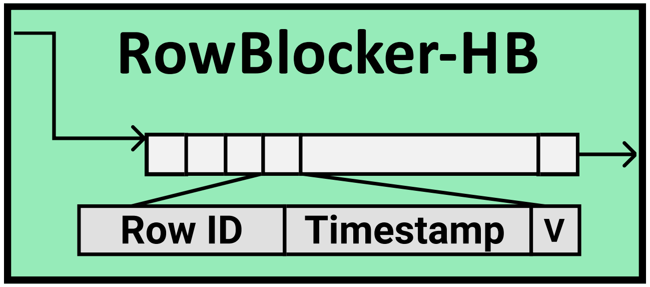


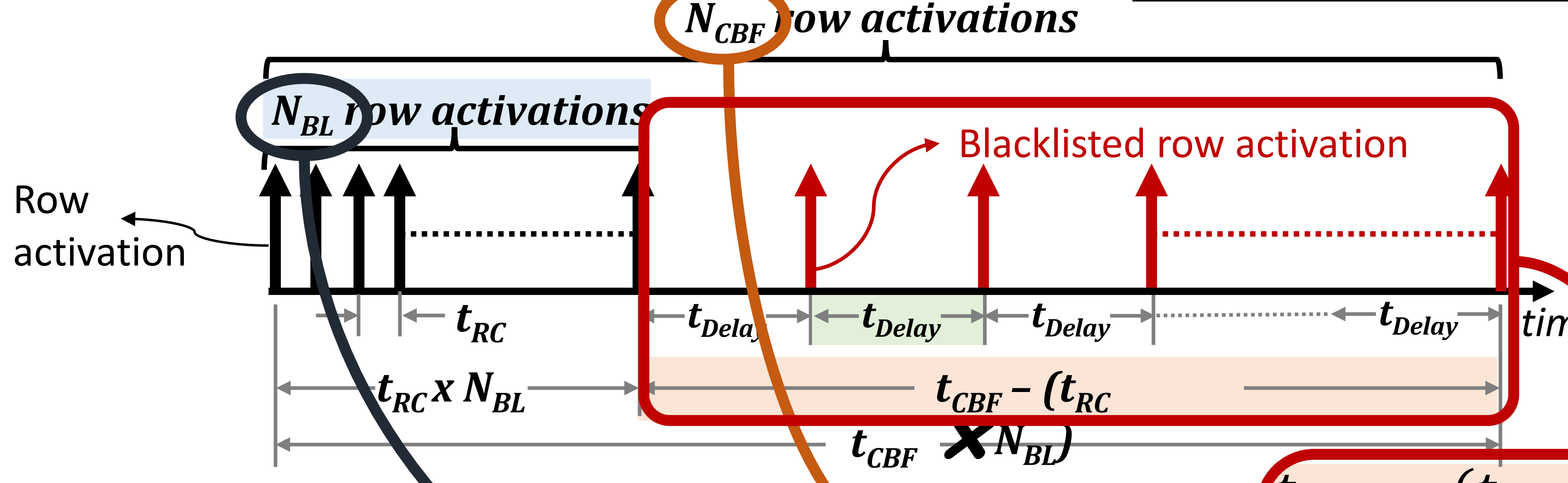
RowBlocker-HB

Limiting the Row Activation Rate

- Ensures that all rows experience a RowHammer-safe activation rate

$$N_{CBF}/t_{CBF} \leq N_{RH}/t_{REFW}$$





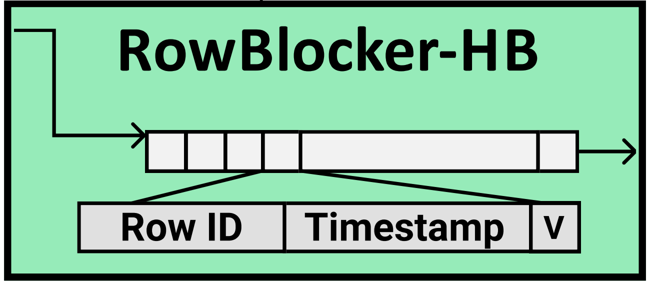
- We limit N_{CBF} by configuring t_{Delay}: $N_{CBF} \leq N_{BL} + \frac{t_{CBF} - (t_{RC} \times N_{BL}}{t_{Delay}}$

SAFARI

RowBlocker-HB

Delaying Row Activations

- RowBlocker-HB ensures no subsequent blacklisted row activation is performed sooner than t_{Delay}



- RowBlocker-HB implements a history buffer for row activations that can fit in a t_{Delay} time window
- A blacklisted row activation is blocked as long as a valid activation record of the row exists in the history buffer

No row can be activated **at a high enough rate** to induce bit-flips

Outline

DRAM and RowHammer Background

Motivation and Goal

BlockHammer

RowBlocker

AttackThrottler

Evaluation

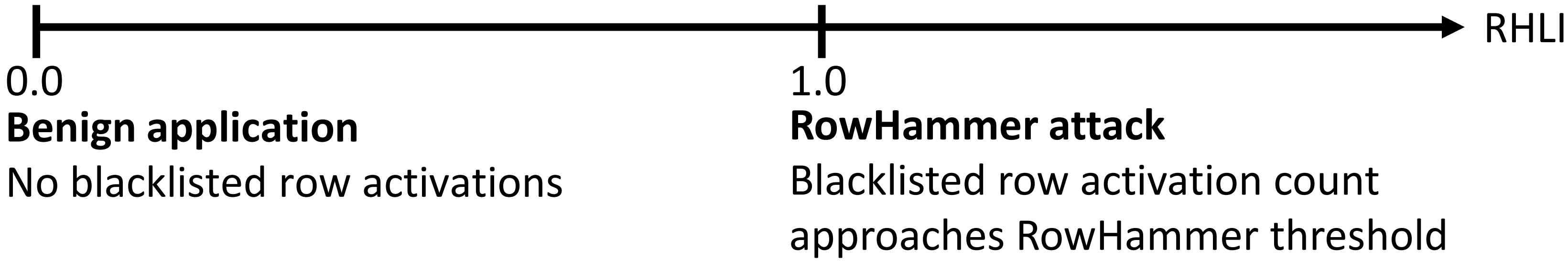
Conclusion

AttackThrottler

- Tackles a RowHammer attack’s **performance degradation** and **energy wastage** on a system

- A RowHammer attack intrinsically keeps activating blacklisted rows

- **RowHammer Likelihood Index (RHLI):** Number of activations that target blacklisted rows (normalized to maximum possible activation count)



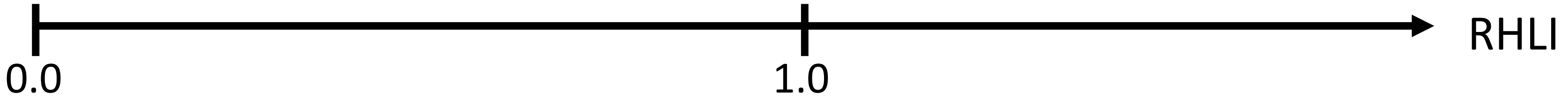
Benign application
No blacklisted row activations

RowHammer attack
Blacklisted row activation count approaches RowHammer threshold

RHLI is larger when the thread’s access pattern is more **similar to a RowHammer attack**

AttackThrottler

- Applies a smaller quota to a thread's in-flight request count as RHLI increases



Benign application
No blacklisted row activations
No quota applied

RowHammer attack
Blacklisted row activation count approaches RowHammer threshold
No request is allowed

- Reduces a RowHammer attack's memory bandwidth consumption, enabling a larger memory bandwidth for concurrent benign applications

Greatly reduces the **perfomance degradation** and **energy wastage** a RowHammer attack inflicts on a system

- RHLI can also be used as a RowHammer attack indicator by the system software

Outline

DRAM and RowHammer Background

Motivation and Goal

BlockHammer

- RowBlocker
- AttackThrottler

Evaluation

Conclusion

Evaluation

BlockHammer’s Hardware Complexity

- We analyze **six state-of-the-art mechanisms** and **BlockHammer**
- We calculate **area**, **access energy**, and **static power** consumption*

	Mitigation	SRAM	CAM	Area		Access Energy	Static Power
	Mechanism	KB	KB	mm^2	%CPU	pJ	mW
	BlockHammer	51.48	1.73	0.14	0.06	20.30	22.27
$N_{RH}=32K$	PARA [73]	-	-	<0.01	-	-	-
	ProHIT [137]	-	0.22	<0.01	<0.01	3.67	0.14
	MRLoc [161]	-	0.47	<0.01	<0.01	4.44	0.21
	CBT [132]	16.00	8.50	0.20	0.08	9.13	35.55
	TWiCe [84]	23.10	14.02	0.15	0.06	7.99	21.28
	Graphene [113]	-	5.22	0.04	0.02	40.67	3.11

BlockHammer is **low cost** and **competitive**
with state-of-the-art mechanisms

*Assuming a high-end 28-core Intel Xeon processor system with 4-channel single-rank DDR4 DIMMs with a RowHammer threshold (NRH) of 32K

Evaluation

BlockHammer’s Hardware Complexity

	Mitigation Mechanism	SRAM KB	CAM KB	Area mm^2	Area %CPU	Access Energy pJ	Static Power mW
N_{RH}=32K	BlockHammer	51.48	1.73	0.14	0.06	20.30	22.27
	PARA [73]	-	-	<0.01	-	-	-
	ProHIT [137]	-	0.22	<0.01	<0.01 10x	3.67	0. 10x
	MRLoc [161]	-	0.47	<0.01	<0.01	4.4 5x	0.2
	CBT [132]	16.00	8.50	0.20	0.08	9.13	35.55
	TWiCe [84]	23.10	14.02	0.15	0.06	7.99	21.28
	Graphene [113]	-	5.22	0.04	0.02	40.67	3.11
N_{RH}=1K	BlockHammer	441.33	55.58	1.57	0.64	99.64	220.99
	PARA [73]	-	-	<0.01	-	-	-
	ProHIT [137]	x	x	x	x	23x x	x
	MRLoc [161]	x	x	x	x	x	x
	CBT [132]	512.00	272.00	3.95	20x 1.60	127.93	15x 535.50
	TWiCe [84]	738.32	448.27	5.17	35x 2.10	124.79	30x 631.98
	Graphene [113]	-	166.03	1.14	23x 0.46	917.55	30x 93.96

BlockHammer’s hardware complexity **scales more efficiently** than state-of-the-art mechanisms

Evaluation
Performance and DRAM Energy

- Cycle-level simulations using **Ramulator** and **DRAMPower**
- System Configuration:

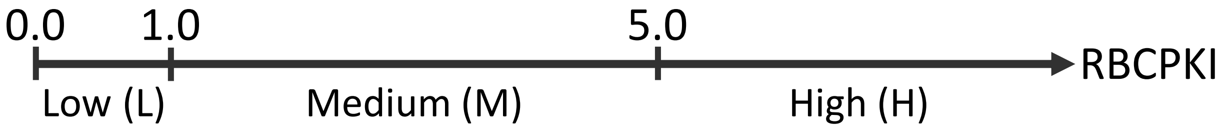
Processor	3.2 GHz, {1,8} core, 4-wide issue, 128-entry instr. window
LLC	64-byte cacheline, 8-way set-associative, {2,16} MB
Memory scheduler	FR-FCFS
Address mapping	Minimalistic Open Pages
DRAM	DDR4 1 channel, 1 rank, 4 bank group, 4 banks per bank group
RowHammer Threshold	32K

- Single-Core Benign Workloads:
  - 22 SPEC CPU 2006
  - 4 YCSB Disk I/O
  - 2 Network Accelerator Traces
  - 2 Bulk Data Copy with Non-Temporal Hint (movnti)
- Randomly Chosen Multiprogrammed Workloads:
  - 125 workloads containing **8 benign applications**
  - 125 workloads containing **7 benign applications** and **1 RowHammer attack thread**

Evaluation

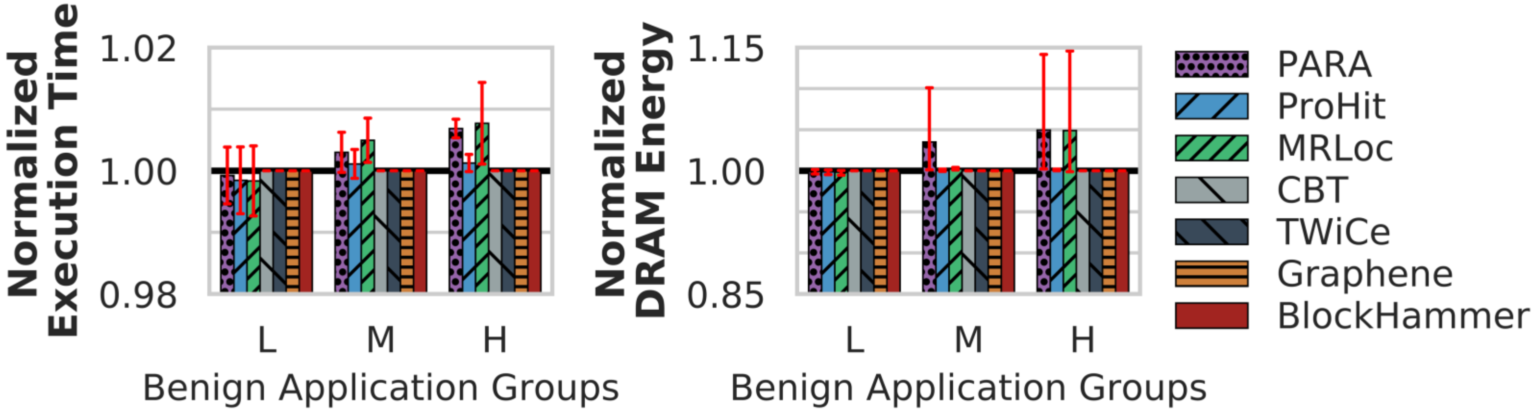
Performance and DRAM Energy

- We classify single-core workloads into three categories based on row buffer conflicts per thousand instructions



0.0 1.0 5.0 RBCPKI

Low (L) Medium (M) High (H)



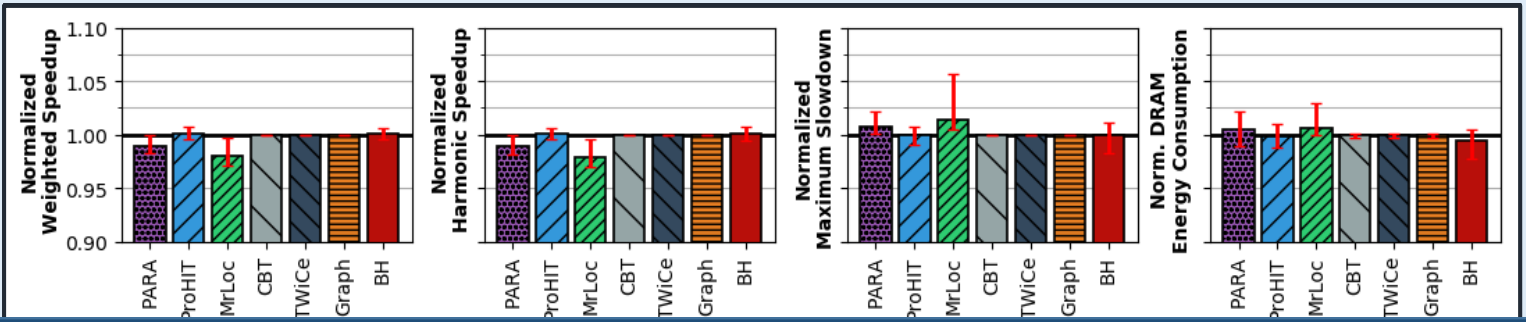
- No application's row activation count exceeds BlockHammer's blacklisting threshold (N_{BL})

BlockHammer does not incur **performance** or **DRAM energy** overheads for single-core benign applications

Evaluation
Performance and DRAM Energy

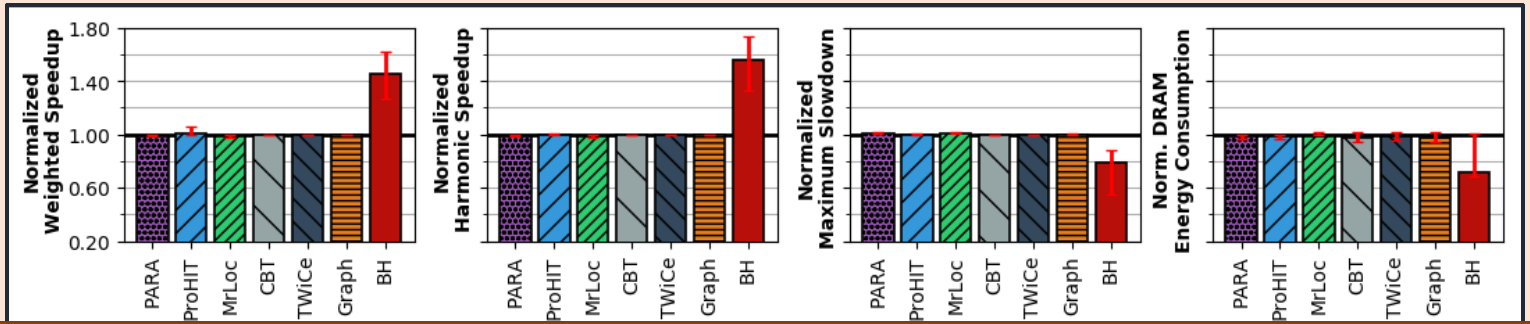
- System throughput (weighted speedup)
- Job turnaround time (harmonic speedup)
- Unfairness (maximum slowdown)
- DRAM energy consumption

No RowHammer Attack



BlockHammer introduces **very low** performance (<0.5%) and DRAM energy (<0.4%) overheads

RowHammer Attack Present



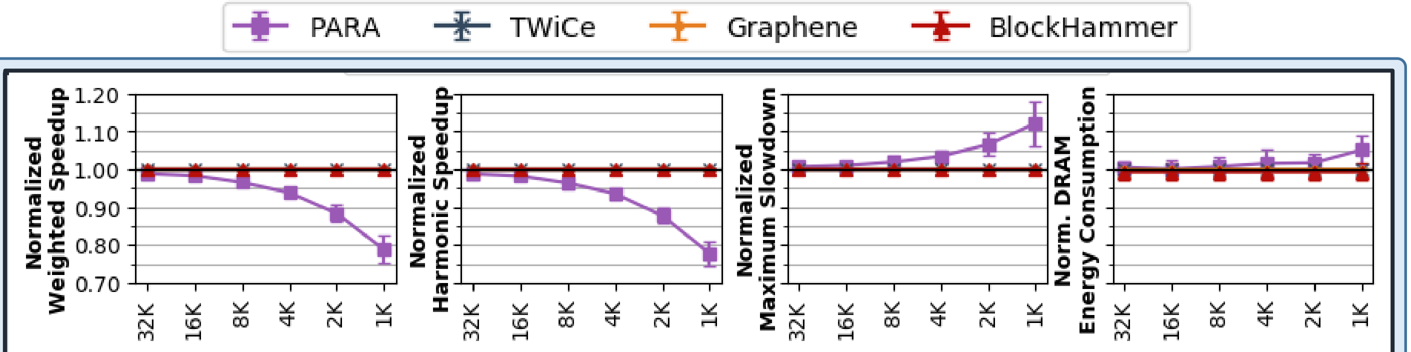
BlockHammer **significantly increases** benign application performance (by 45% on average) and **reduces** DRAM energy consumption (by 29% on average)

Evaluation

Scaling with RowHammer Vulnerability

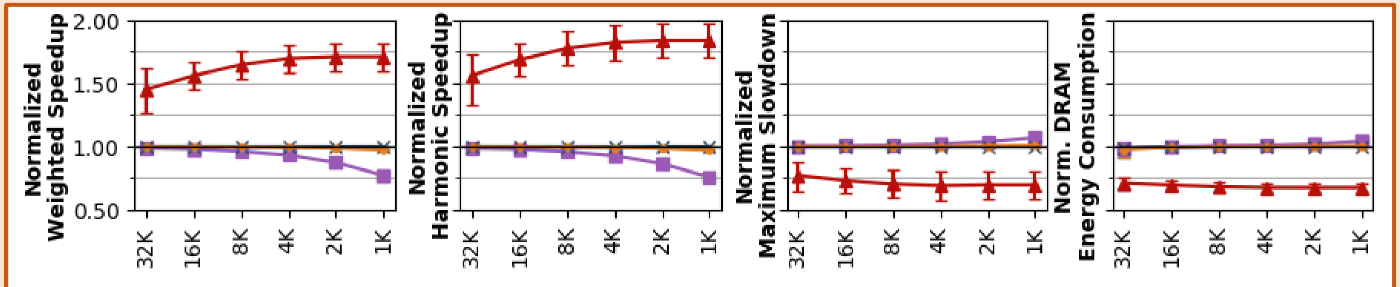
- System throughput (weighted speedup)
- Job turnaround time (harmonic speedup)
- Unfairness (maximum slowdown)
- DRAM energy consumption

No RowHammer Attack



BlockHammer's performance and energy overheads remain **negligible (<0.6%)**

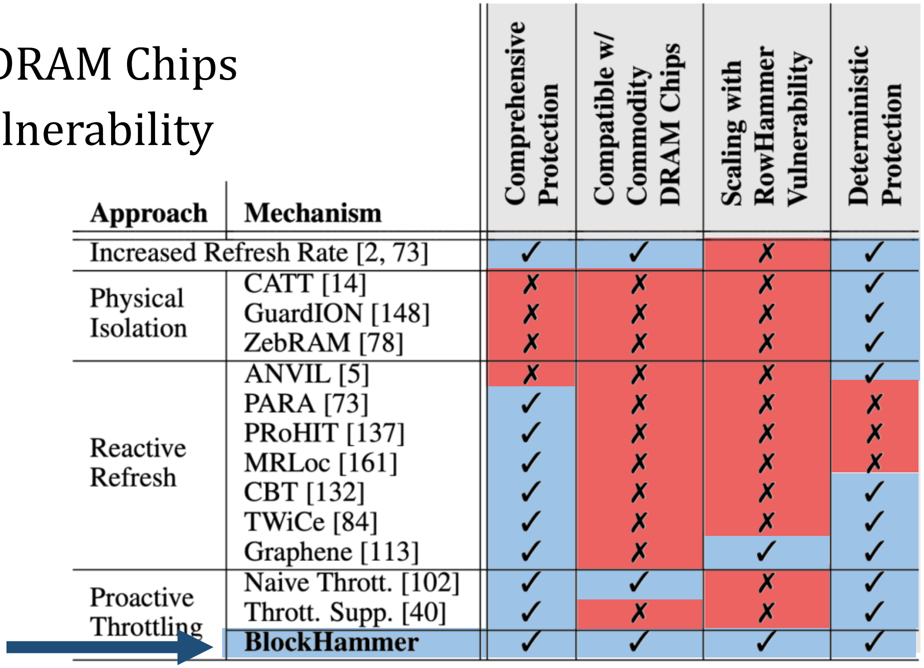
RowHammer Attack Present



BlockHammer scalably provides **much higher performance** (71% on average) and **lower energy consumption** (32% on average) than state-of-the-art mechanisms

More in the Paper

- Security Proof
  - Mathematically represent **all possible** access patterns
  - We show that **no row can be activated high-enough times** to induce bit-flips when BlockHammer is configured correctly
- Addressing **Many-Sided Attacks**
- Evaluation of **14 mechanisms** representing **four mitigation approaches**
  - Comprehensive Protection
  - Compatibility with Commodity DRAM Chips
  - Scalability with RowHammer Vulnerability
  - Deterministic Protection



Approach	Mechanism	Comprehensive Protection	Compatible w/ Commodity DRAM Chips	Scaling with RowHammer Vulnerability	Deterministic Protection
Increased Refresh Rate [2, 73]		✓	✓	✗	✓
Physical Isolation	CATT [14]	✗	✗	✗	✓
	GuardION [148]	✗	✗	✗	✓
	ZebRAM [78]	✗	✗	✗	✓
Reactive Refresh	ANVIL [5]	✗	✗	✗	✓
	PARA [73]	✓	✗	✗	✗
	PRoHIT [137]	✓	✗	✗	✗
	MRLoc [161]	✓	✗	✗	✗
	CBT [132]	✓	✗	✗	✓
	TWiCe [84]	✓	✗	✗	✓
	Graphene [113]	✓	✗	✓	✓
Proactive Throttling	Naive Thrott. [102]	✓	✓	✗	✓
	Thrott. Supp. [40]	✓	✗	✗	✓
	BlockHammer	✓	✓	✓	✓

Outline

- DRAM and RowHammer Background
- Motivation and Goal
- BlockHammer
    - RowBlocker
    - AttackThrottler
- Evaluation
- Conclusion

Conclusion

- **Motivation**: RowHammer is a worsening DRAM reliability and security problem
- **Problem**: Mitigation mechanisms have limited support for current/future chips
  - **Scalability** with worsening RowHammer vulnerability
  - **Compatibility** with commodity DRAM chips
- **Goal**: **Efficiently** and **scalably** prevent RowHammer bit-flips
  without knowledge of or modifications to DRAM internals
- **Key Idea:** Selectively throttle memory accesses that may cause RowHammer bit-flips
- **Mechanism**: BlockHammer
  - **Tracks** activation rates of all rows by using area-efficient Bloom filters
  - **Throttles** row activations that could cause RowHammer bit flips
  - **Identifies and throttles** threads that perform RowHammer attacks
- **Scalability with Worsening RowHammer Vulnerability:**
  - **Competitive** with state-of-the-art mechanisms **when there is no attack**
  - **Superior** performance and DRAM energy **when a RowHammer attack is present**
- **Compatibility with Commodity DRAM Chips:**
  - **No proprietary information** of DRAM internals
  - **No modifications** to DRAM circuitry

BlockHammer

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

Abdullah Giray Yağlıkçı

Minesh Patel Jeremie S. Kim Roknoddin Azizi

Ataberk Olgun Lois Orosa Hasan Hassan Jisung Park

Konstantinos Kanellopoulos Taha Shahroodi

Saugata Ghose* Onur Mutlu

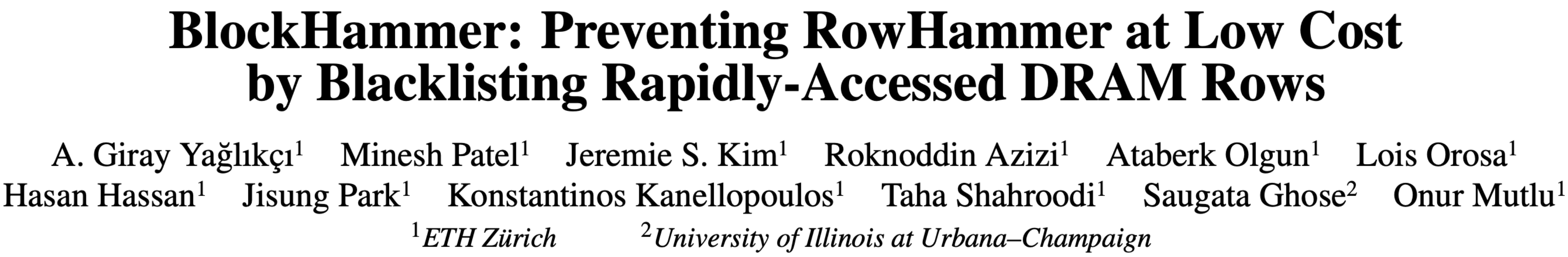
SAFARI

ETH zürich

* UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

More on BlockHammer

- 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)]
  [Intel Hardware Security Academic Awards Short Talk Slides (pptx) (pdf)]
  [Talk Video (22 minutes)]
  [Short Talk Video (7 minutes)]
  [Intel Hardware Security Academic Awards Short Talk Video (2 minutes)]
  [BlockHammer Source Code]
  Intel Hardware Security Academic Award Finalist (one of 4 finalists out of 34 nominations)

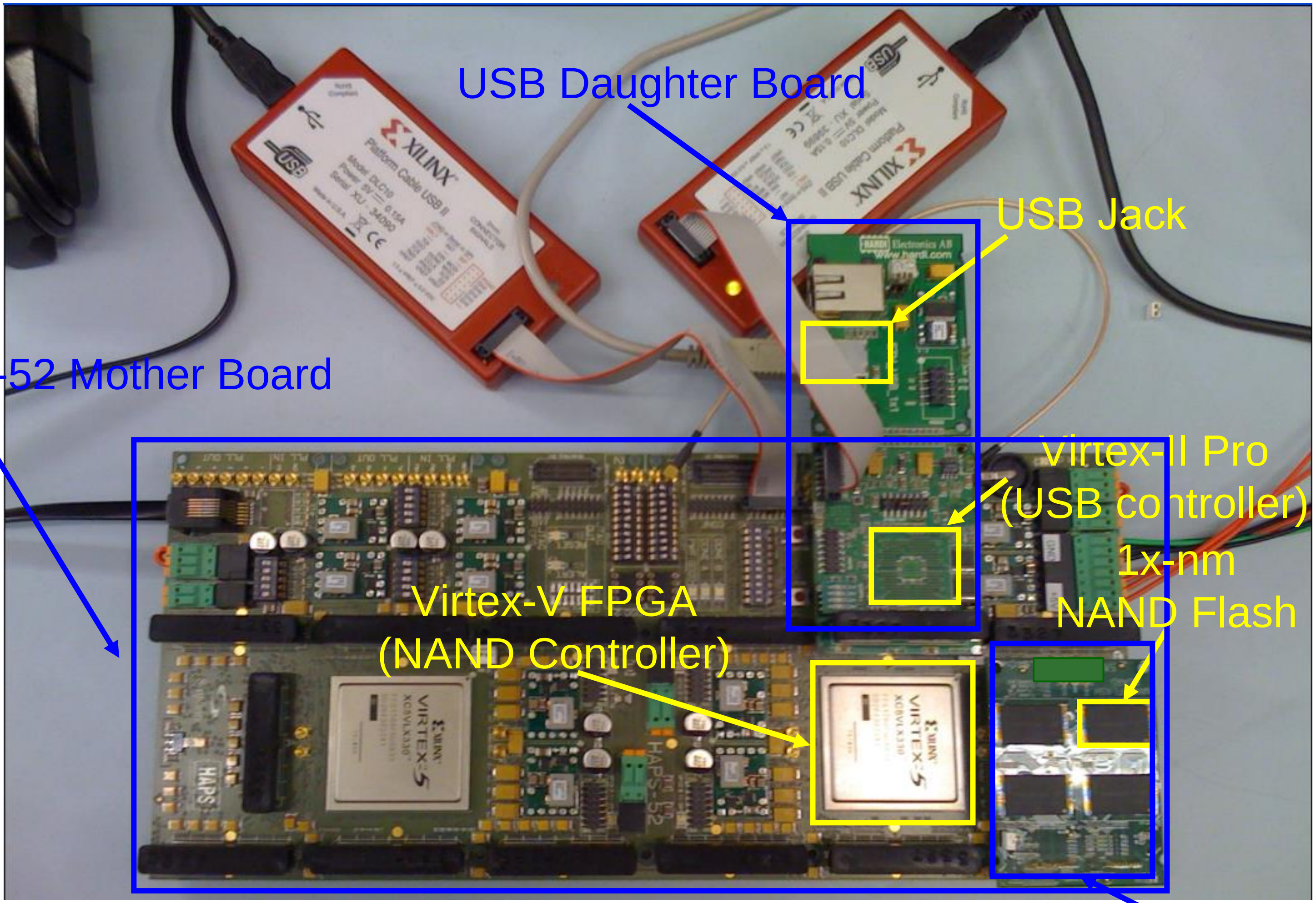


BlockHammer: Preventing RowHammer at Low Cost by Blacklisting Rapidly-Accessed DRAM Rows

A. Giray Yağlıkçı[1] Minesh Patel[1] Jeremie S. Kim[1] Roknoddin Azizi[1] Ataberk Olgun[1] Lois Orosa[1]
Hasan Hassan[1] Jisung Park[1] Konstantinos Kanellopoulos[1] Taha Shahroodi[1] Saugata Ghose[2] Onur Mutlu[1]
[1] *ETH Zürich* [2] *University of Illinois at Urbana–Champaign*

Read Disturb in Flash Memory

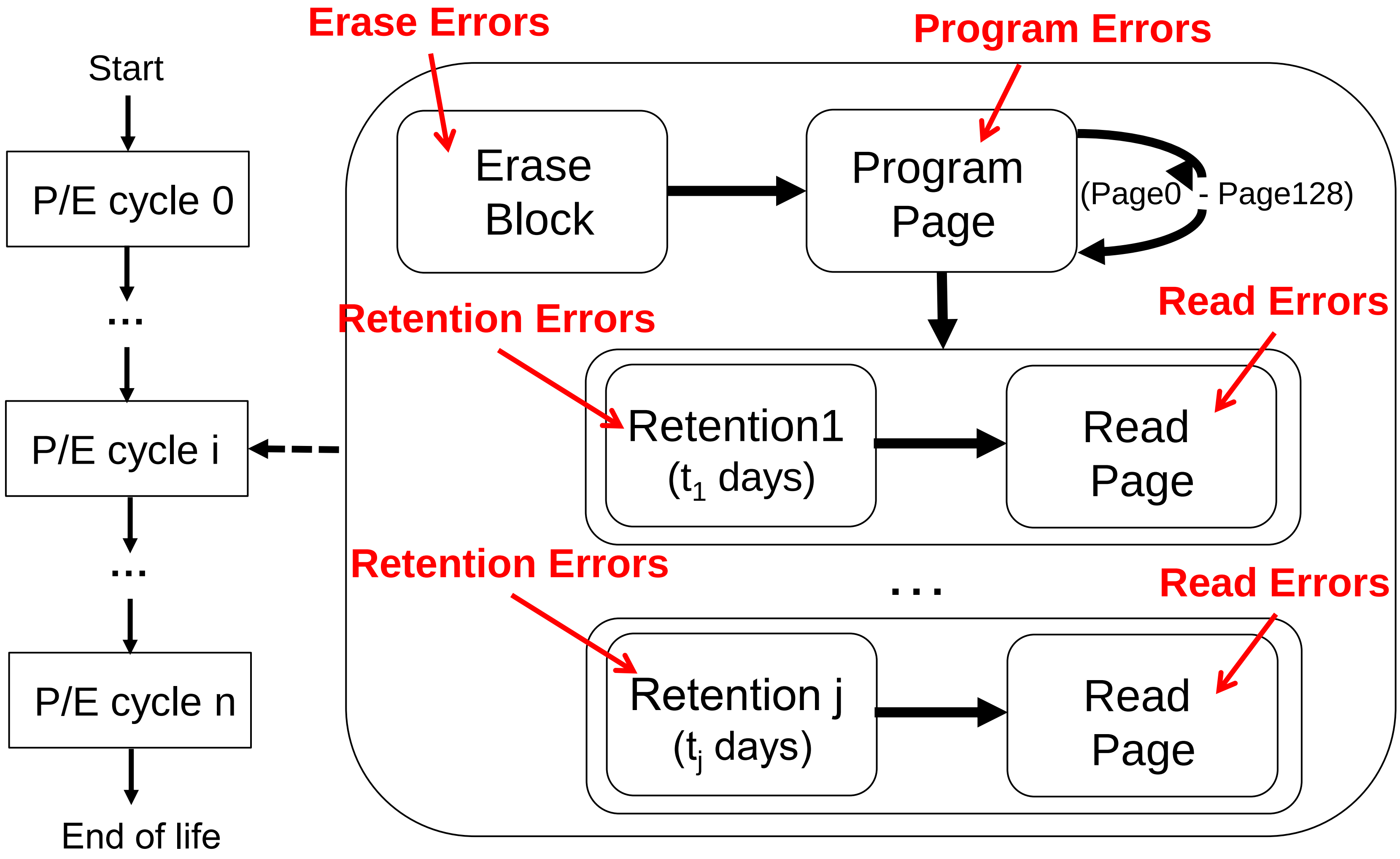
Experimental Testing Platform



[DATE 2012, ICCD 2012, DATE 2013, ITJ 2013, ICCD 2013, SIGMETRICS 2014, HPCA 2015, DSN 2015, MSST 2015, JSAC 2016, HPCA 2017, DFRWS 2017, PIEEE 2017, HPCA 2018, SIGMETRICS 2018]

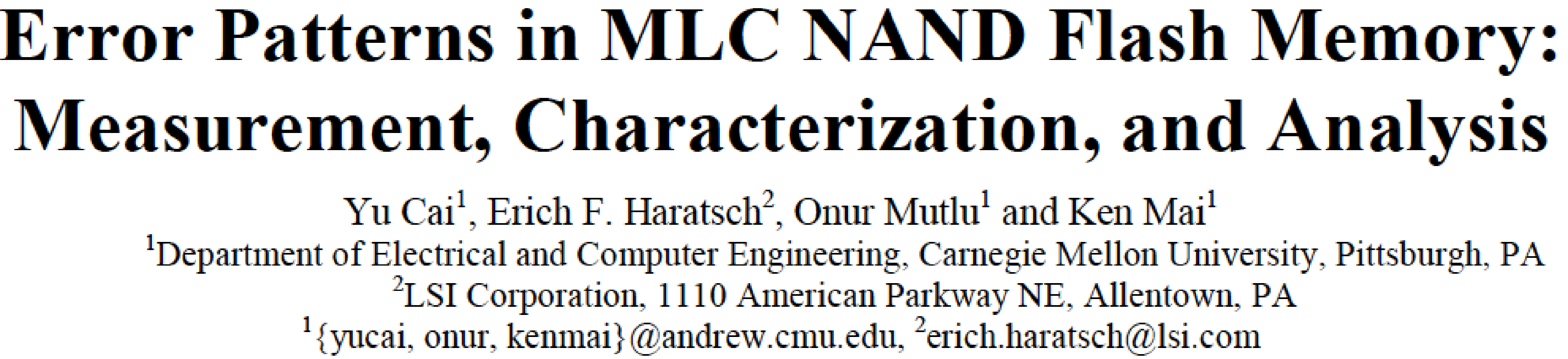
Cai+, "Error Characterization, Mitigation, and Recovery in Flash Memory Based Solid State Drives," Proc. IEEE 2017

NAND Flash Usage and Error Model



More on Flash Error Analysis

- Yu Cai, Erich F. Haratsch, Onur Mutlu, and Ken Mai, **"Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis"** *Proceedings of the Design, Automation, and Test in Europe Conference* (***DATE***), Dresden, Germany, March 2012. Slides (ppt)



Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis

Yu Cai[1], Erich F. Haratsch[2], Onur Mutlu[1] and Ken Mai[1]
[1]Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA
[2]LSI Corporation, 1110 American Parkway NE, Allentown, PA
[1]{yucai, onur, kenmai}@andrew.cmu.edu, [2]erich.haratsch@lsi.com

Many Errors and Their Mitigation [PIEEE'17]

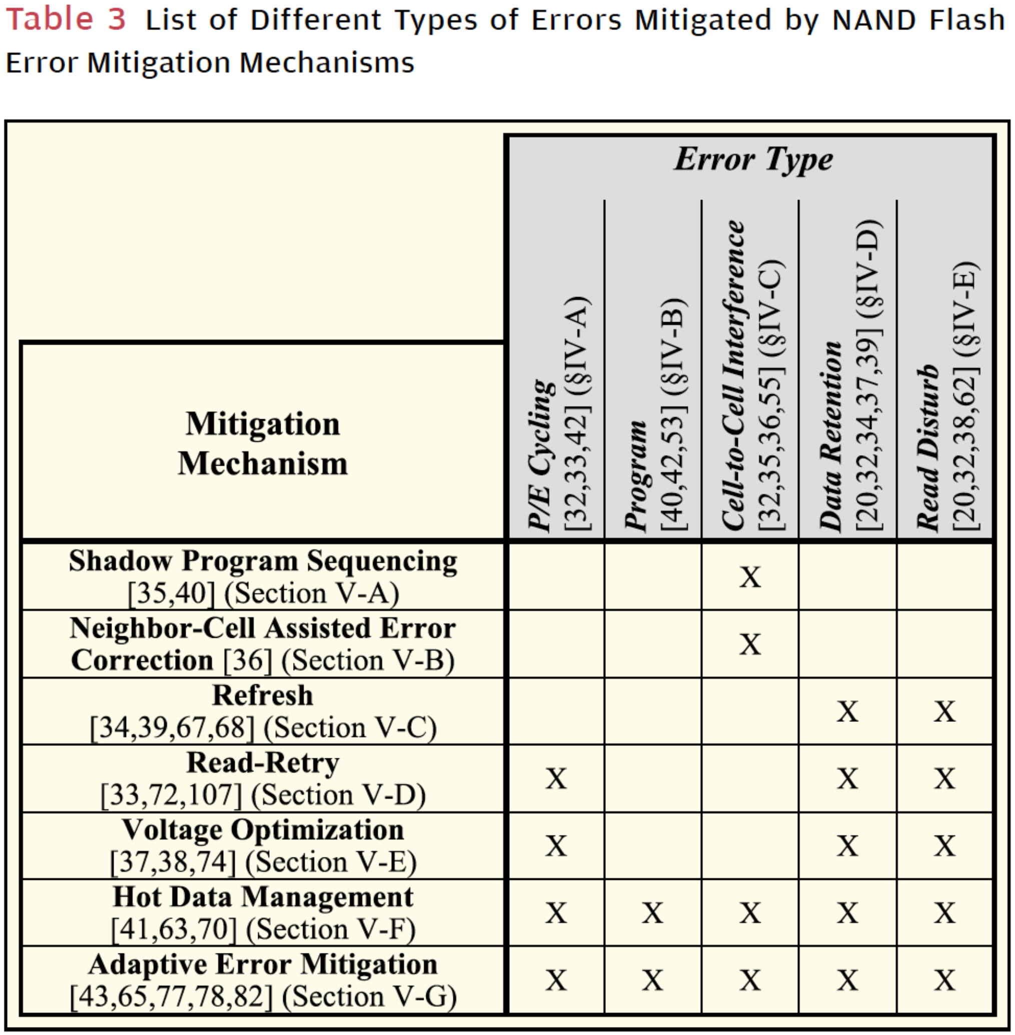
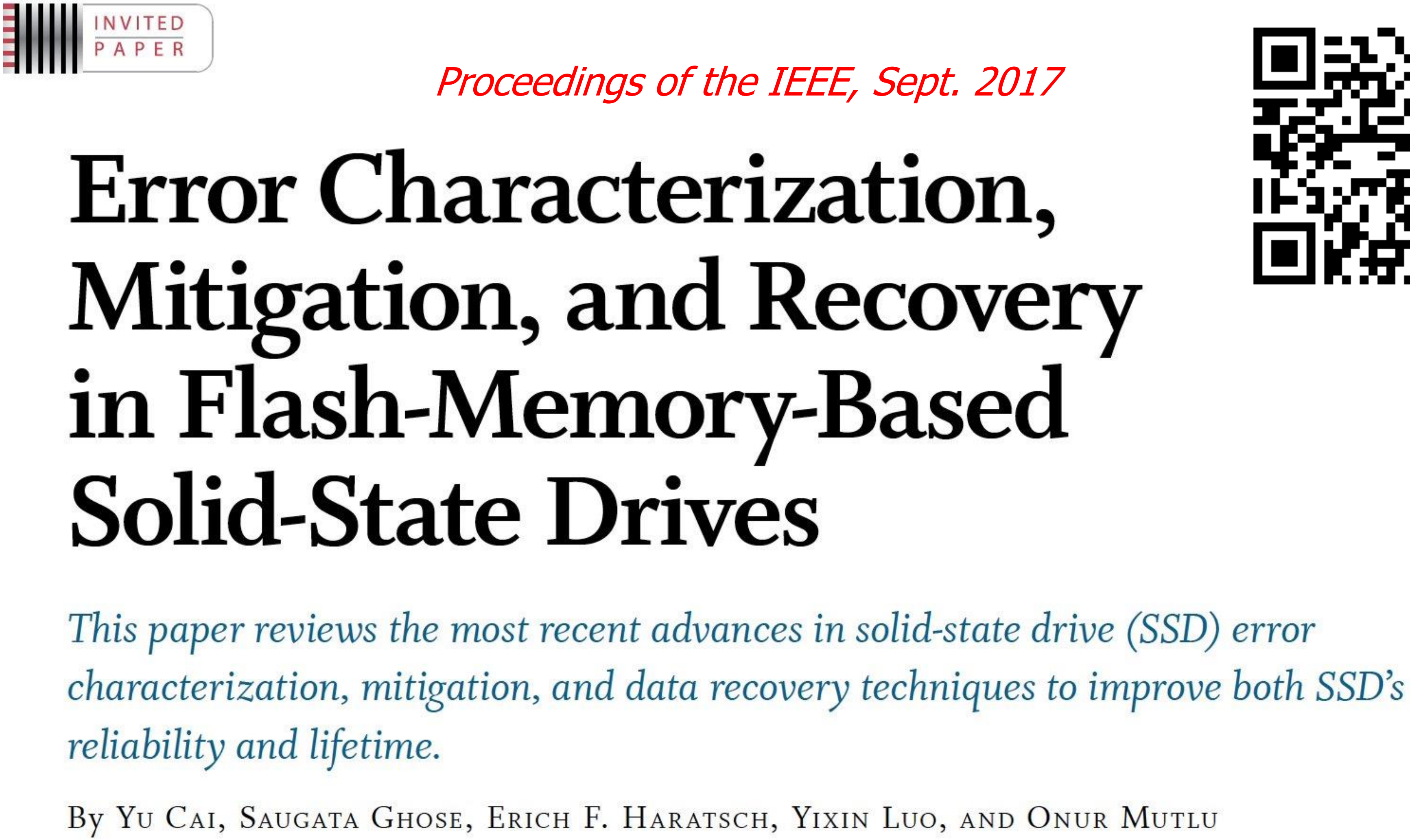


Table 3 List of Different Types of Errors Mitigated by NAND Flash Error Mitigation Mechanisms

	Error Type				
Mitigation Mechanism	***P/E Cycling*** [32,33,42] (§IV-A)	***Program*** [40,42,53] (§IV-B)	***Cell-to-Cell Interference*** [32,35,36,55] (§IV-C)	***Data Retention*** [20,32,34,37,39] (§IV-D)	***Read Disturb*** [20,32,38,62] (§IV-E)
Shadow Program Sequencing [35,40] (Section V-A)			X		
Neighbor-Cell Assisted Error Correction [36] (Section V-B)			X		
Refresh [34,39,67,68] (Section V-C)				X	X
Read-Retry [33,72,107] (Section V-D)	X			X	X
Voltage Optimization [37,38,74] (Section V-E)	X			X	X
Hot Data Management [41,63,70] (Section V-F)	X	X	X	X	X
Adaptive Error Mitigation [43,65,77,78,82] (Section V-G)	X	X	X	X	X

Cai+, "Error Characterization, Mitigation, and Recovery in Flash Memory Based Solid State Drives," Proc. IEEE 2017.

Many Errors and Their Mitigation [PIEEE'17]



Proceedings of the IEEE, Sept. 2017

INVITED PAPER

Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives

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

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

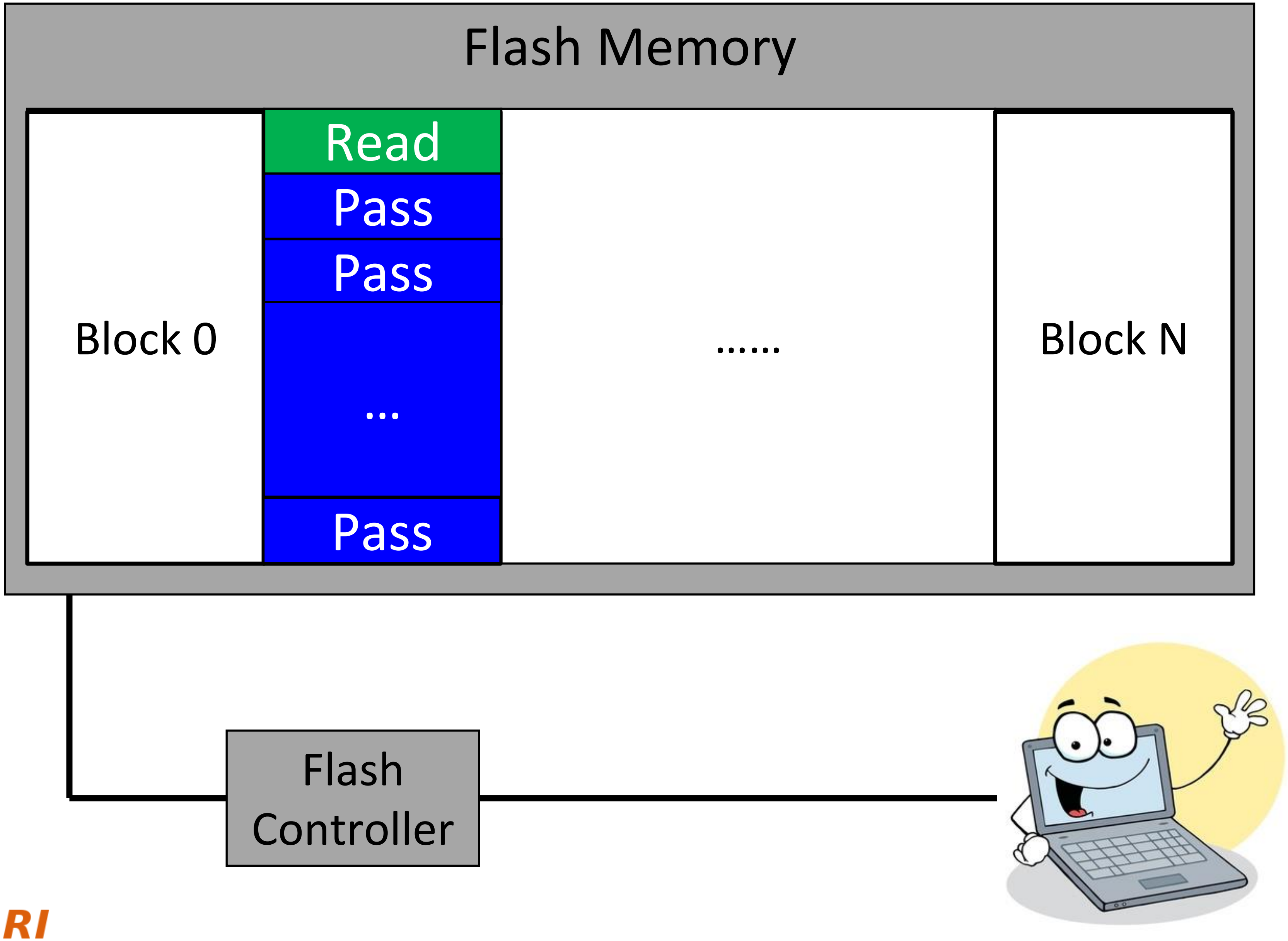
https://arxiv.org/pdf/1706.08642

One Issue: Read Disturb in Flash Memory

- All scaled memories are prone to read disturb errors

- DRAM
- SRAM
- Hard Disks: Adjacent Track Interference
- NAND Flash

NAND Flash Memory Background



Flash Memory

Block 0

Read

Pass

Pass

…

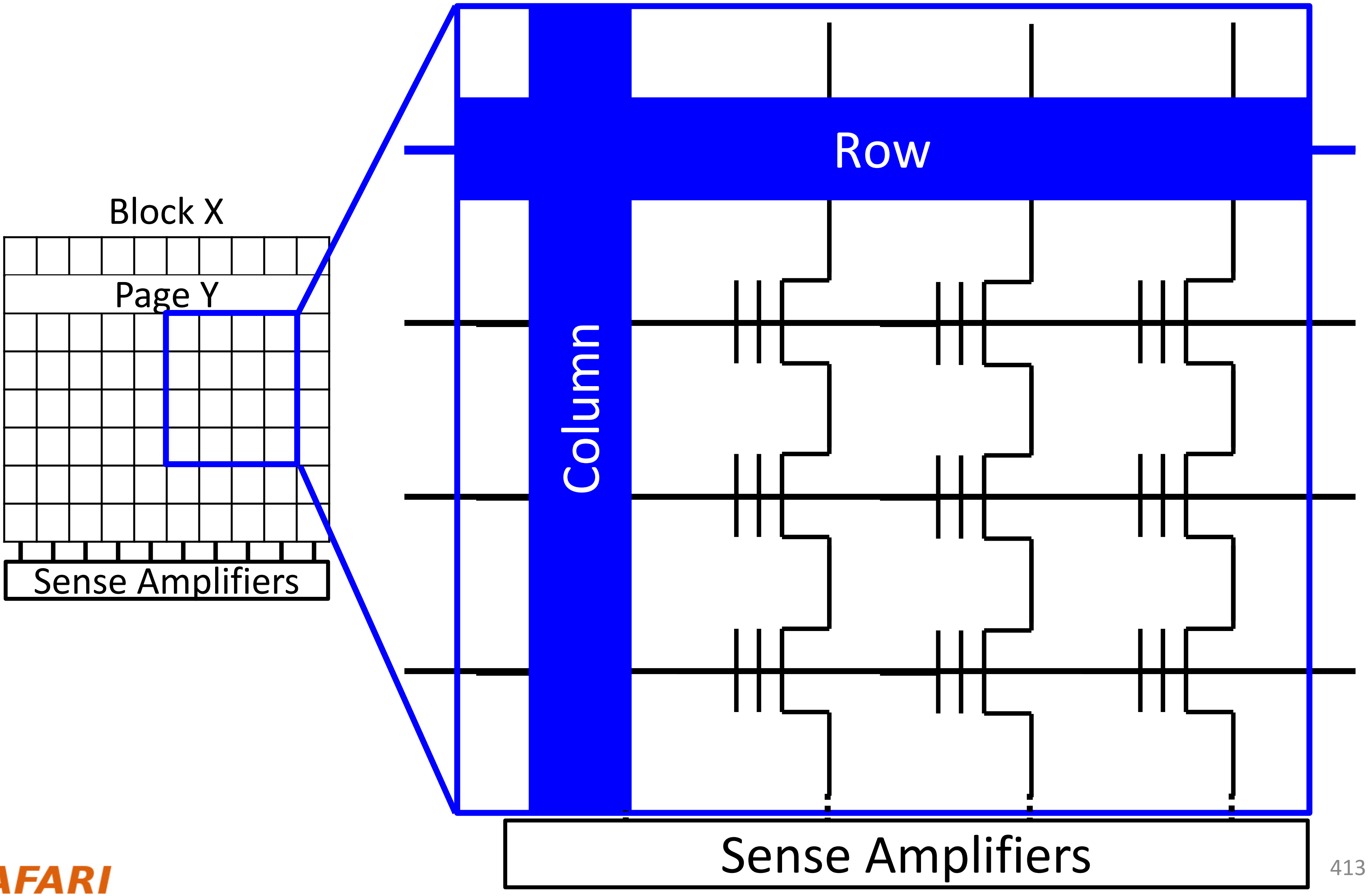
Pass

……

Block N

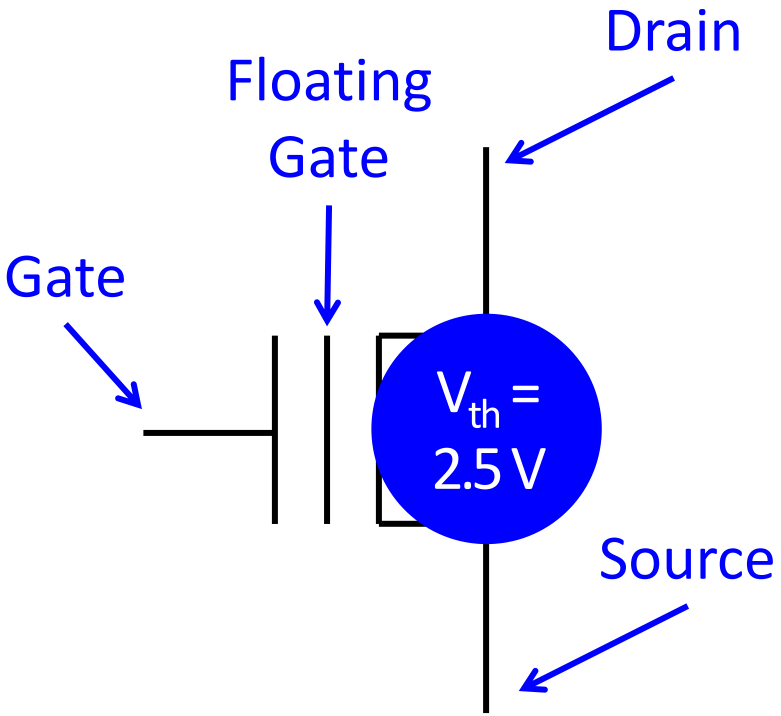
Flash Controller

Flash Cell Array



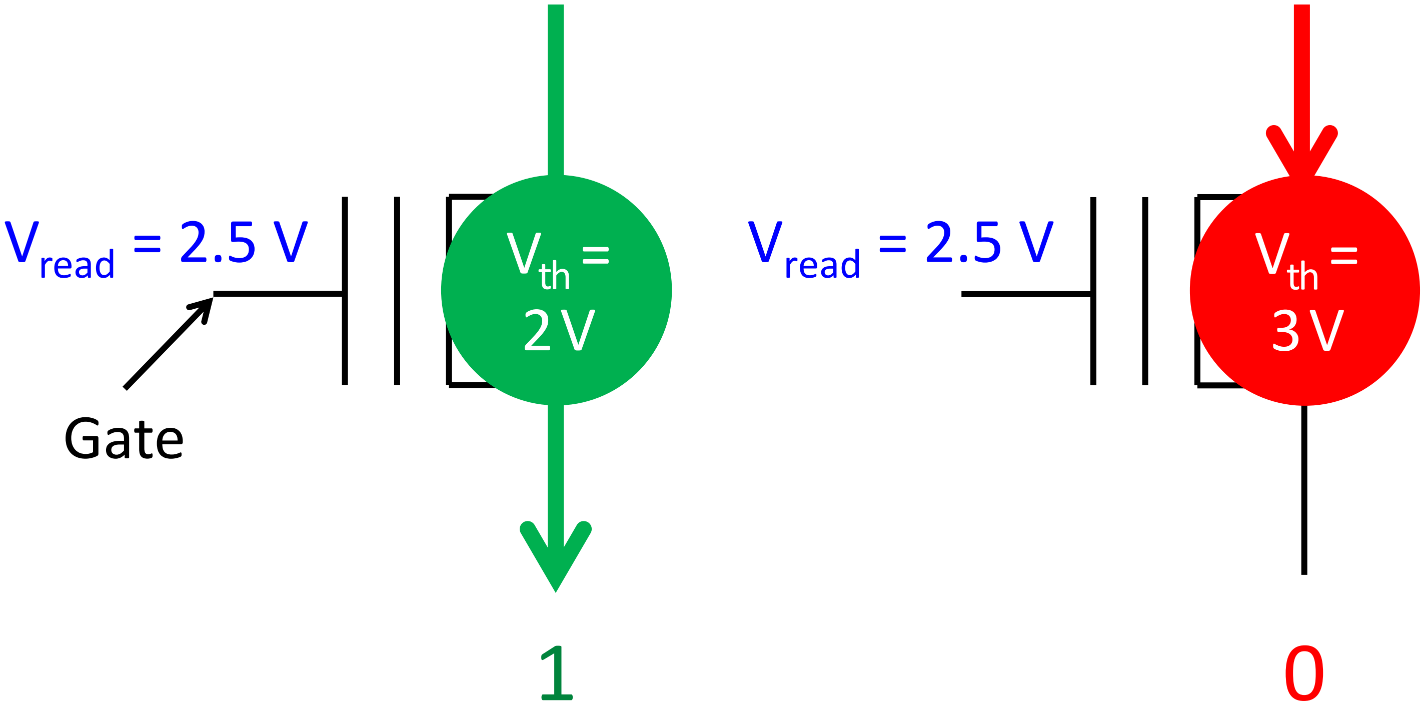
SAFARI

Flash Cell

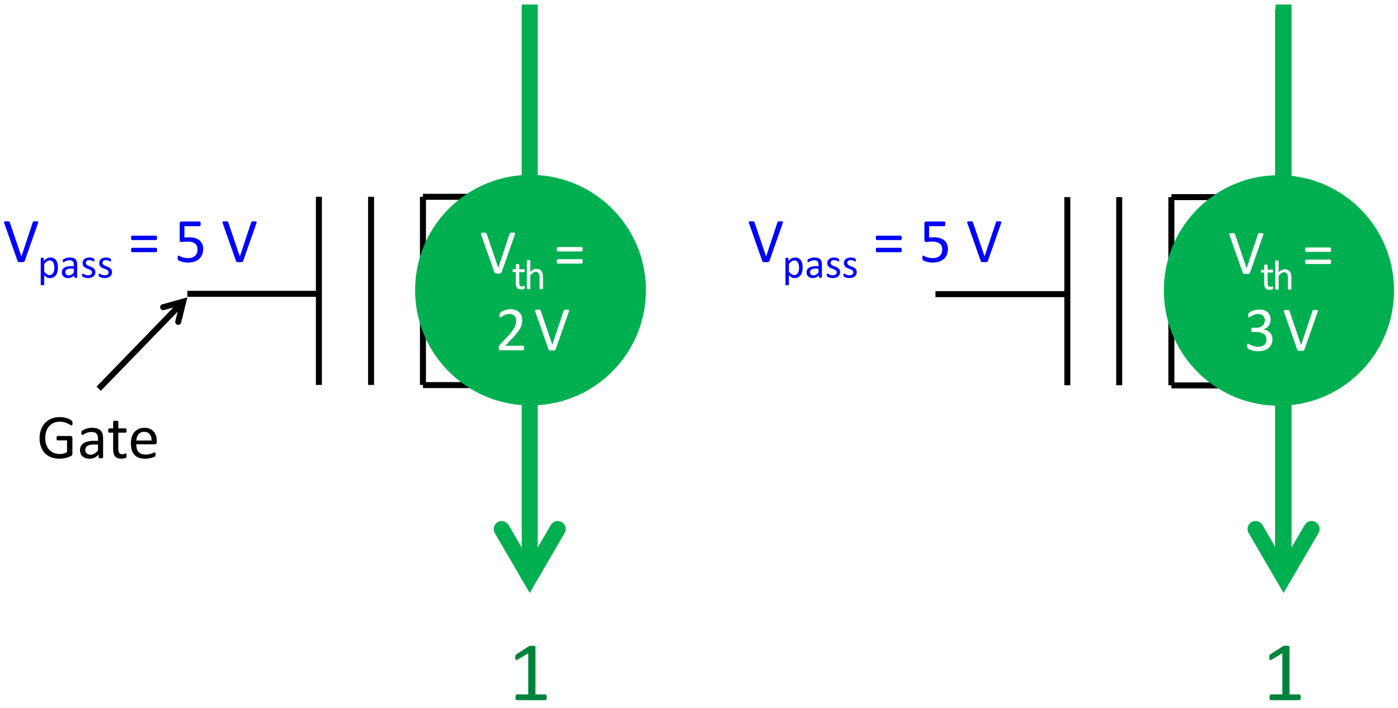


Floating Gate Transistor
(Flash Cell)

Flash Read



Flash Pass-Through



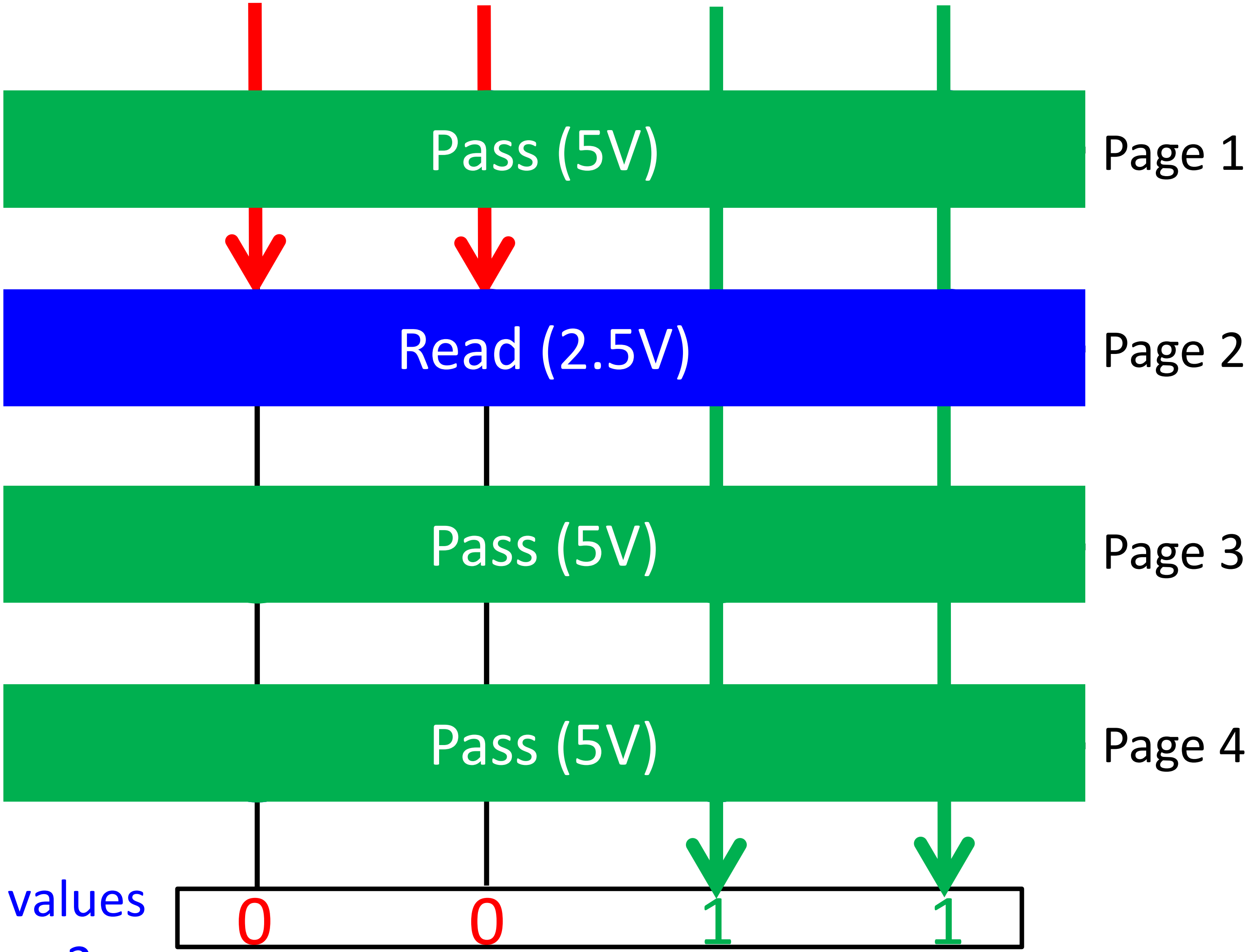
More on Flash Read Disturb Errors **[DSN'15]**

- Yu Cai, Yixin Luo, Saugata Ghose, Erich F. Haratsch, Ken Mai, and Onur Mutlu,
  "Read Disturb Errors in MLC NAND Flash Memory: Characterization and Mitigation"
  Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (***DSN***), Rio de Janeiro, Brazil, June 2015.

Read Disturb Errors in MLC NAND Flash Memory: Characterization, Mitigation, and Recovery

Yu Cai, Yixin Luo, Saugata Ghose, Erich F. Haratsch*, Ken Mai, Onur Mutlu
Carnegie Mellon University, *Seagate Technology
yucaicai@gmail.com, {yixinluo, ghose, kenmai, onur}@cmu.edu

Read from Flash Cell Array



V_{pass} = 5.0

Pass (5V)

Page 1

V_{read} = 2.5

Read (2.5V)

Page 2

V_{pass} = 5.0

Pass (5V)

Page 3

V_{pass} = 5.0

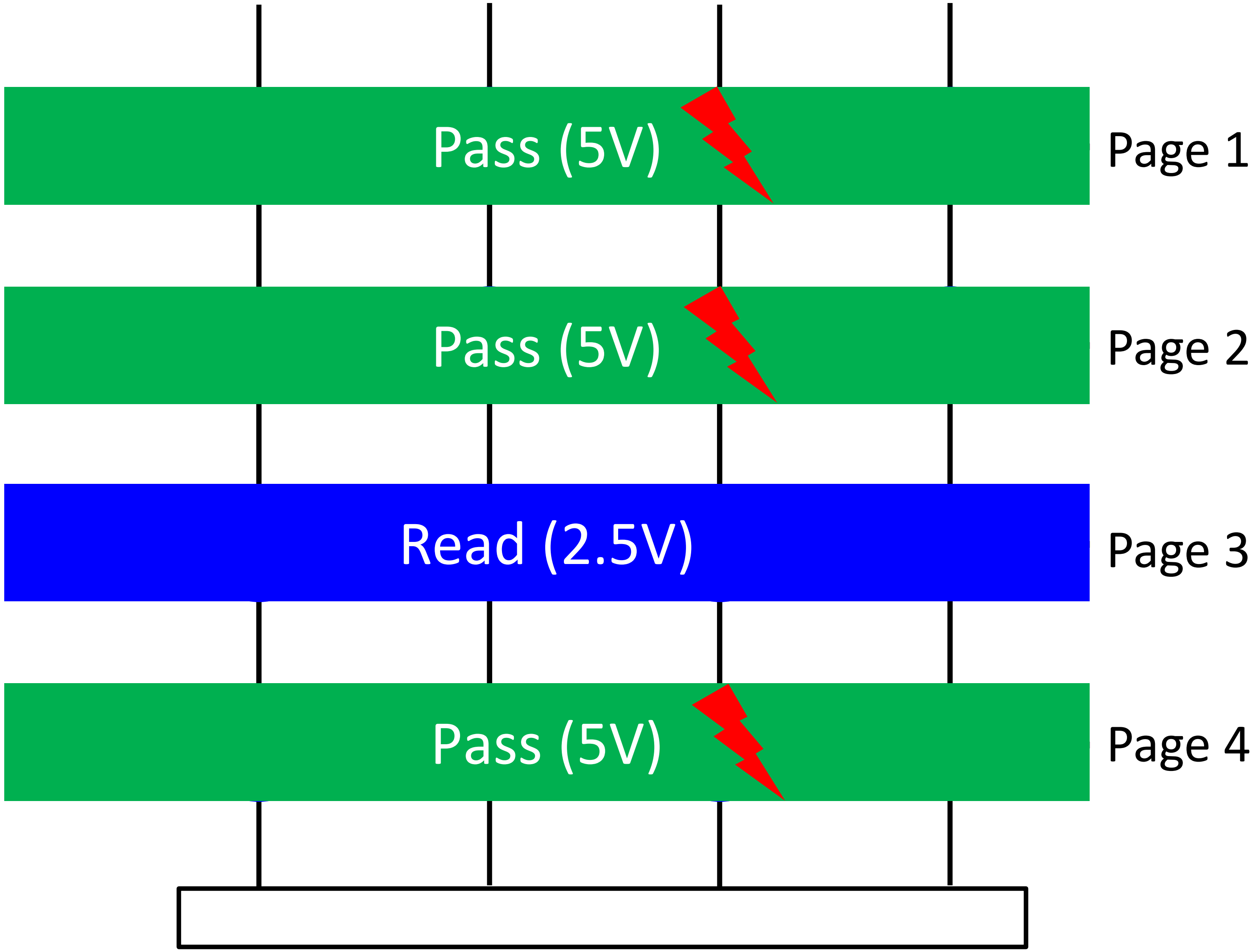
Pass (5V)

Page 4

Correct values for page 2:

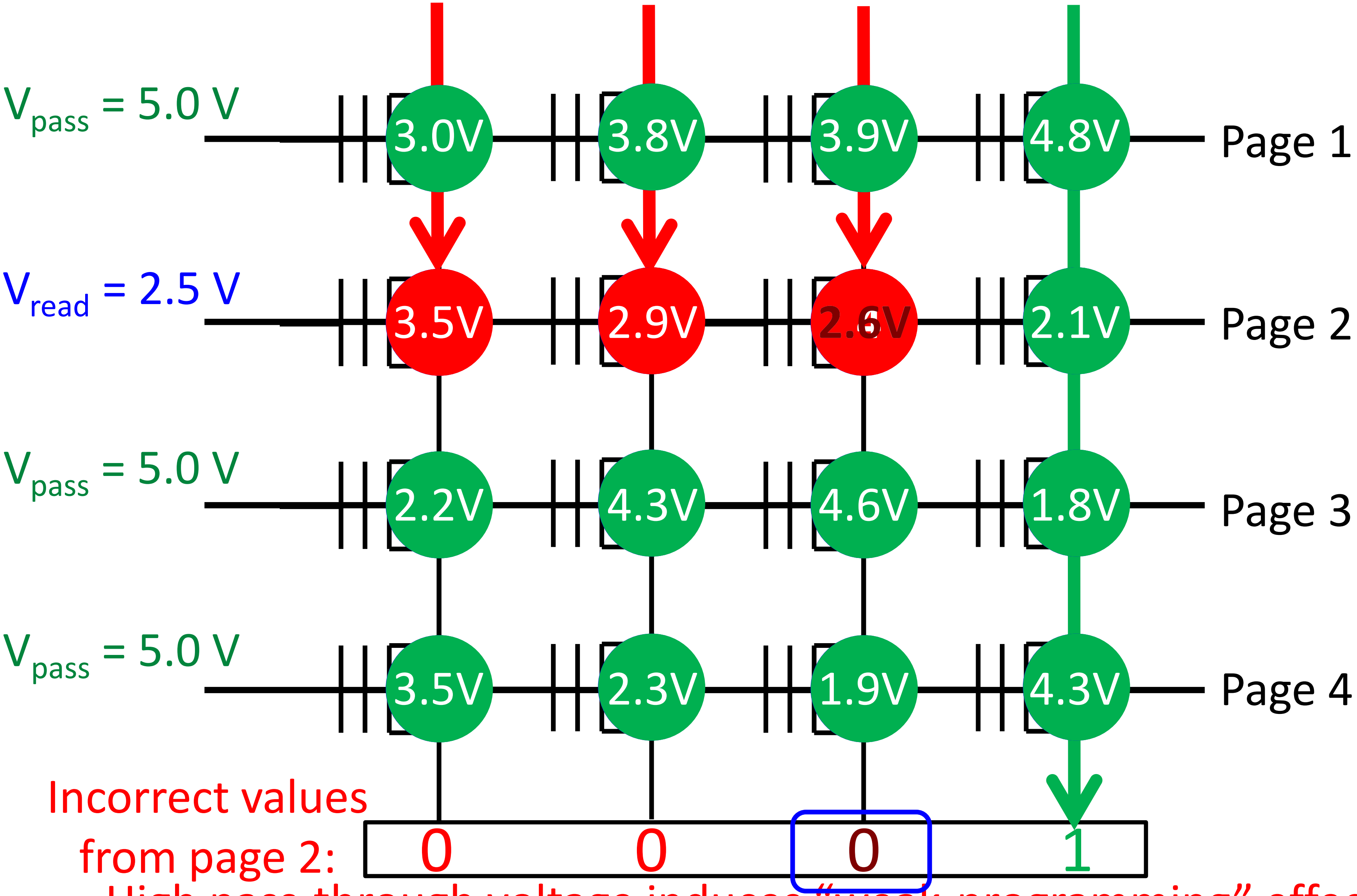
0	0	1	1

Read Disturb Problem: “Weak Programming” Effect



Repeatedly read page 3 (or any page other than page 2)

Read Disturb Problem: “Weak Programming” Effect

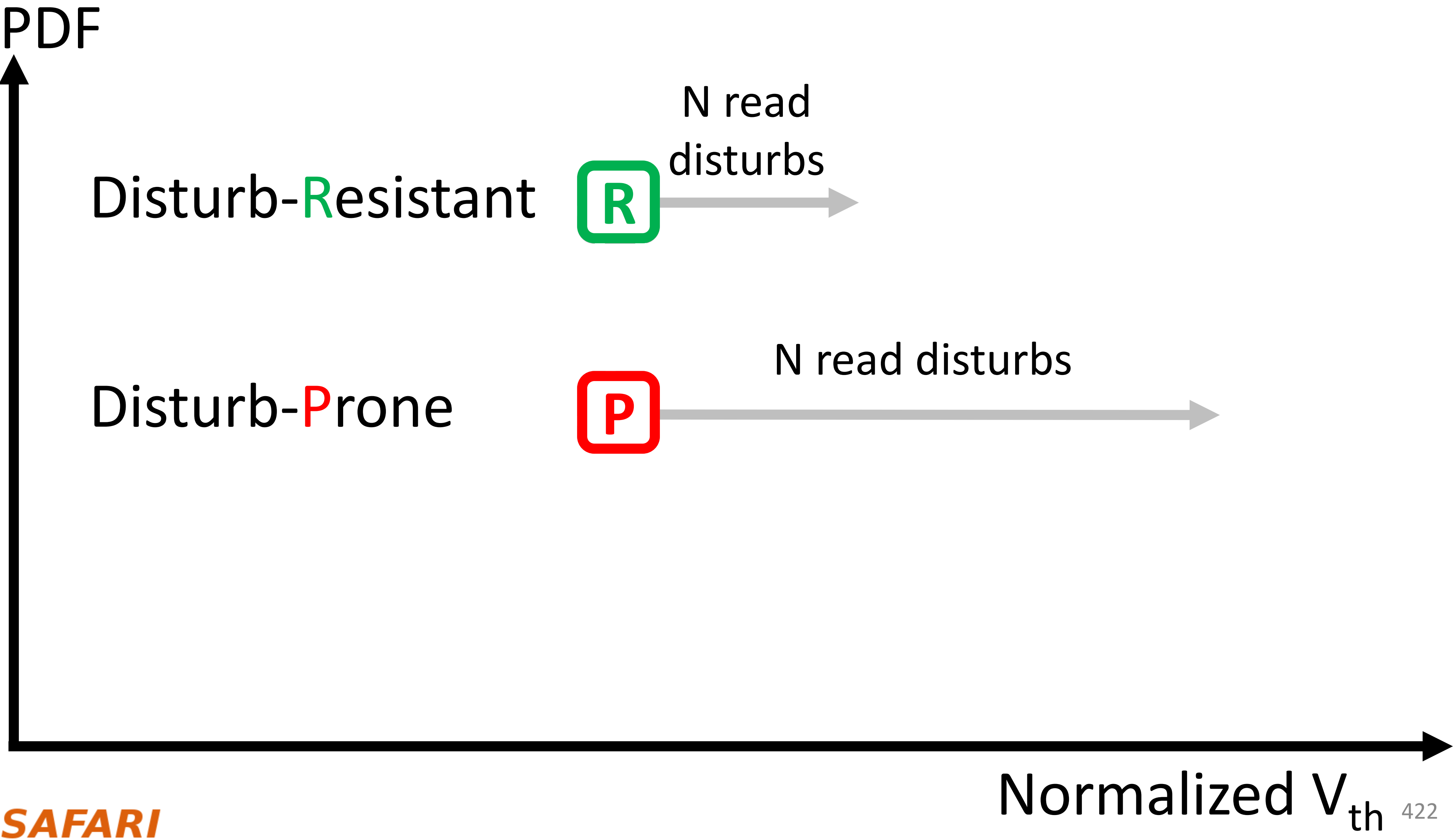


High pass-through voltage induces “weak-programming” effect

Executive Summary [DSN'15]

- ***Read disturb errors*** limit flash memory lifetime today
  - Apply a *high pass-through voltage (V_{pass})* to multiple pages on a read
  - Repeated application of V_{pass} can alter stored values in unread pages
- We **characterize read disturb** on real NAND flash chips
  - Slightly lowering V_{pass} greatly reduces read disturb errors
  - Some flash cells are more prone to read disturb
- **Technique 1:** Mitigate read disturb errors online
  - ***V_{pass} Tuning*** dynamically finds and applies a lowered V_{pass} per block
  - Flash memory lifetime improves by 21%
- **Technique 2:** Recover after failure to prevent data loss
  - ***Read Disturb Oriented Error Recovery*** (RDR) selectively corrects cells more susceptible to read disturb errors
  - Reduces raw bit error rate (RBER) by up to 36%

Read Disturb Prone vs. Resistant Cells



PDF

Disturb-Resistant

R

N read disturbs

Disturb-Prone

P

N read disturbs

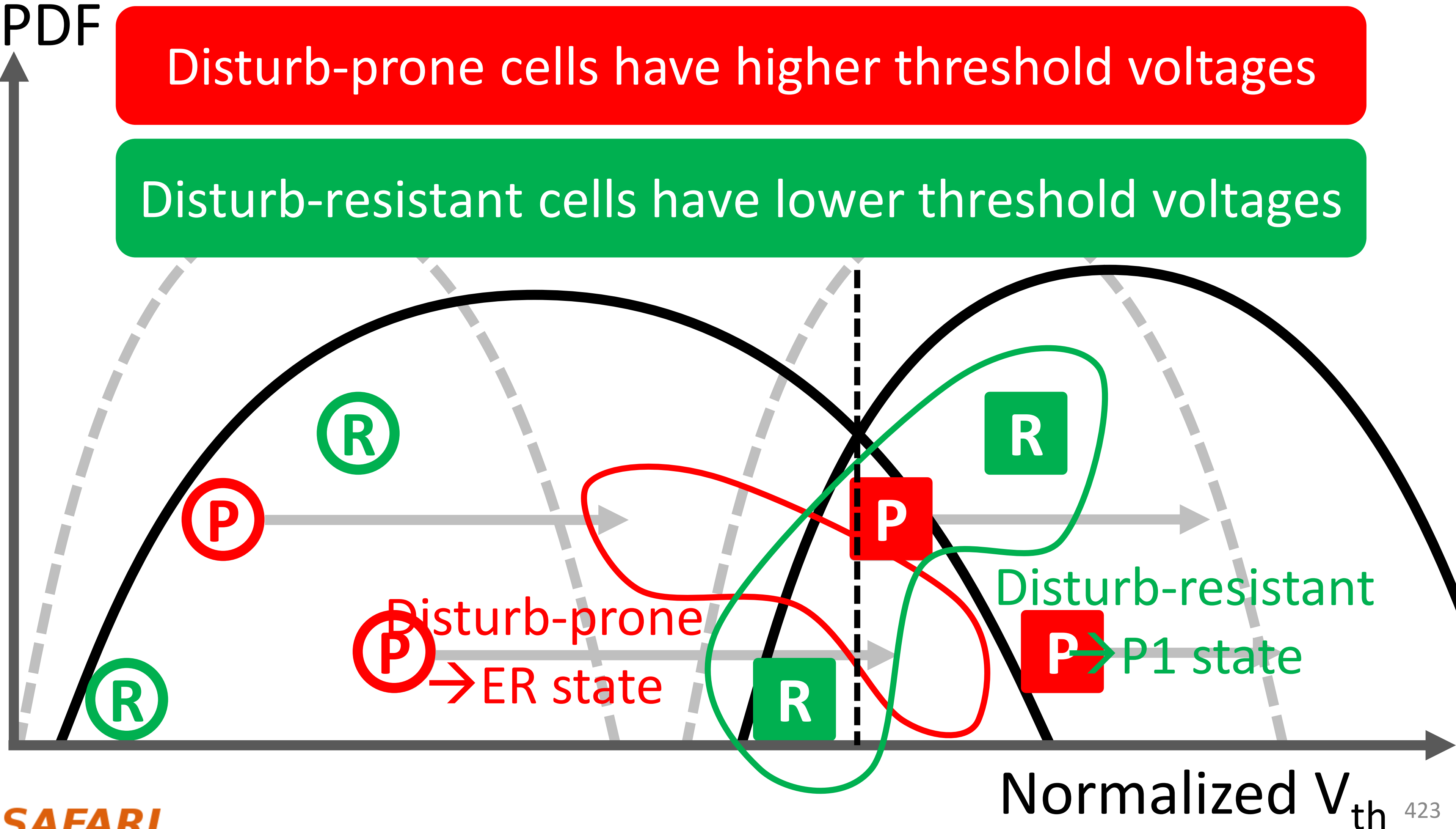
Normalized V_{th}

Observation 2: Some Flash Cells Are More Prone to Read Disturb

After 250K read disturbs:

Disturb-prone cells have higher threshold voltages

Disturb-resistant cells have lower threshold voltages



Read Disturb Oriented Error Recovery (RDR)

- Triggered by an uncorrectable flash error
  - Back up all valid data in the faulty block
  - Disturb the faulty page 100K times (more)
  - Compare V_{th}'s before and after read disturb
  - Select cells susceptible to flash errors ($V_{ref}−\sigma<V_{th}<V_{ref}−\sigma$)
  - Predict among these susceptible cells
    - Cells with more V_{th} shifts are disturb-prone → Higher V_{th} state
    - Cells with less V_{th} shifts are disturb-resistant → Lower V_{th} state

Reduces total error count by up to 36% @ 1M read disturbs
ECC can be used to correct the remaining errors

More on Flash Read Disturb Errors **[DSN'15]**

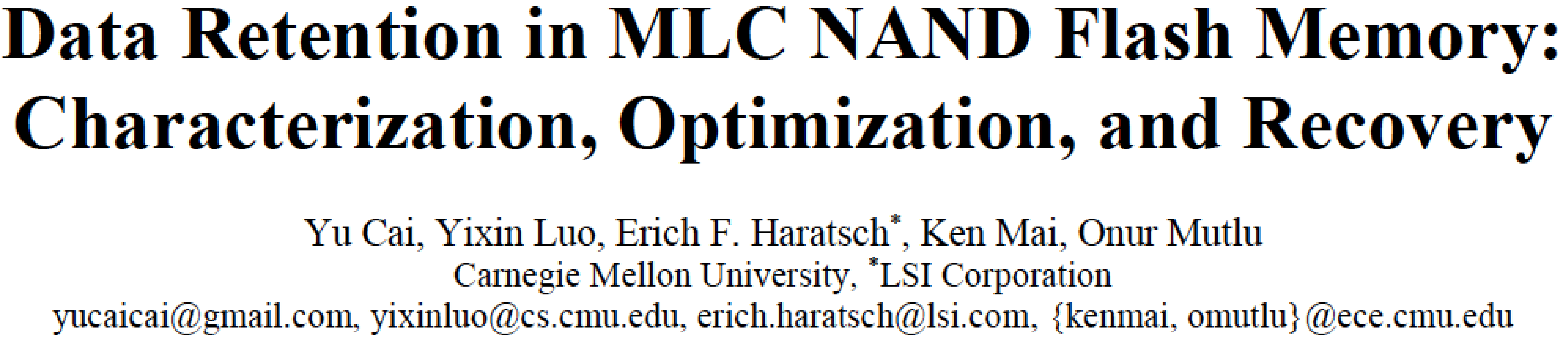
- Yu Cai, Yixin Luo, Saugata Ghose, Erich F. Haratsch, Ken Mai, and Onur Mutlu,
"Read Disturb Errors in MLC NAND Flash Memory: Characterization and Mitigation"
Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (***DSN***), Rio de Janeiro, Brazil, June 2015.

Read Disturb Errors in MLC NAND Flash Memory: Characterization, Mitigation, and Recovery

Yu Cai, Yixin Luo, Saugata Ghose, Erich F. Haratsch*, Ken Mai, Onur Mutlu
Carnegie Mellon University, *Seagate Technology
yucaicai@gmail.com, {yixinluo, ghose, kenmai, onur}@cmu.edu

Data Retention in Flash Memory

- Yu Cai, Yixin Luo, Erich F. Haratsch, Ken Mai, and Onur Mutlu, **"Data Retention in MLC NAND Flash Memory: Characterization, Optimization and Recovery"** *Proceedings of the 21st International Symposium on High-Performance Computer Architecture (**HPCA**)*, Bay Area, CA, February 2015. [Slides (pptx) (pdf)]



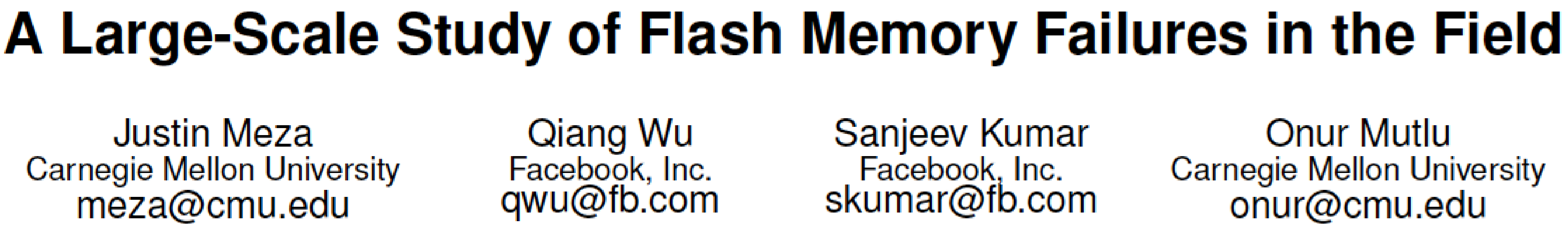
Data Retention in MLC NAND Flash Memory: Characterization, Optimization, and Recovery

Yu Cai, Yixin Luo, Erich F. Haratsch*, Ken Mai, Onur Mutlu
Carnegie Mellon University, *LSI Corporation
yucaicai@gmail.com, yixinluo@cs.cmu.edu, erich.haratsch@lsi.com, {kenmai, omutlu}@ece.cmu.edu

Large-Scale SSD Error Analysis [SIGMETRICS'15]

- First large-scale field study of flash memory errors

- Justin Meza, Qiang Wu, Sanjeev Kumar, and Onur Mutlu,
"A Large-Scale Study of Flash Memory Errors in the Field"
*Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (**SIGMETRICS**)*, Portland, OR, June 2015.
[Slides (pptx) (pdf)] [Coverage at ZDNet] [Coverage on The Register] [Coverage on TechSpot] [Coverage on The Tech Report]



A Large-Scale Study of Flash Memory Failures in the Field

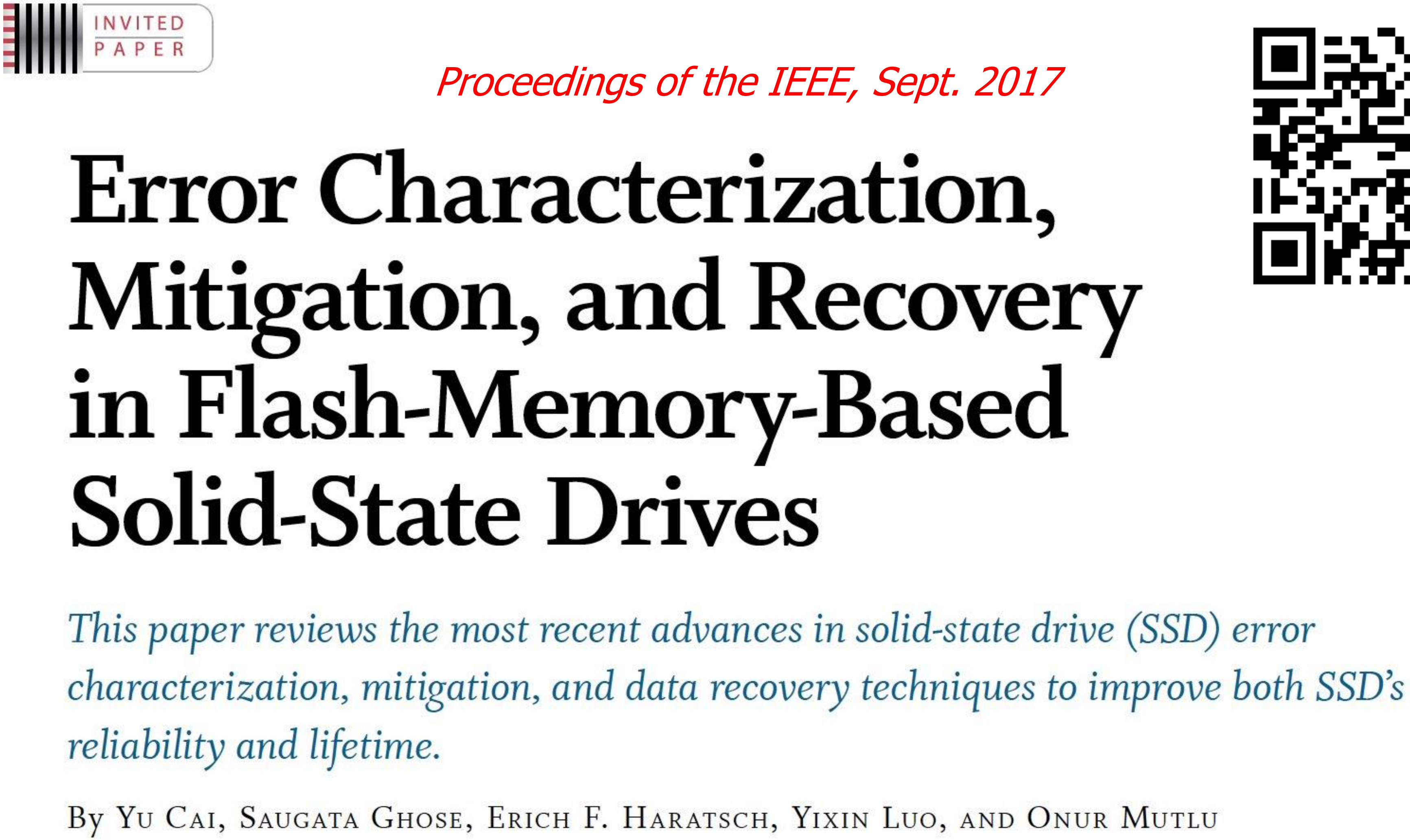
Justin Meza
Carnegie Mellon University
meza@cmu.edu

Qiang Wu
Facebook, Inc.
qwu@fb.com

Sanjeev Kumar
Facebook, Inc.
skumar@fb.com

Onur Mutlu
Carnegie Mellon University
onur@cmu.edu

Many Errors and Their Mitigation **[PIEEE'17]**



INVITED PAPER

Proceedings of the IEEE, Sept. 2017

Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives

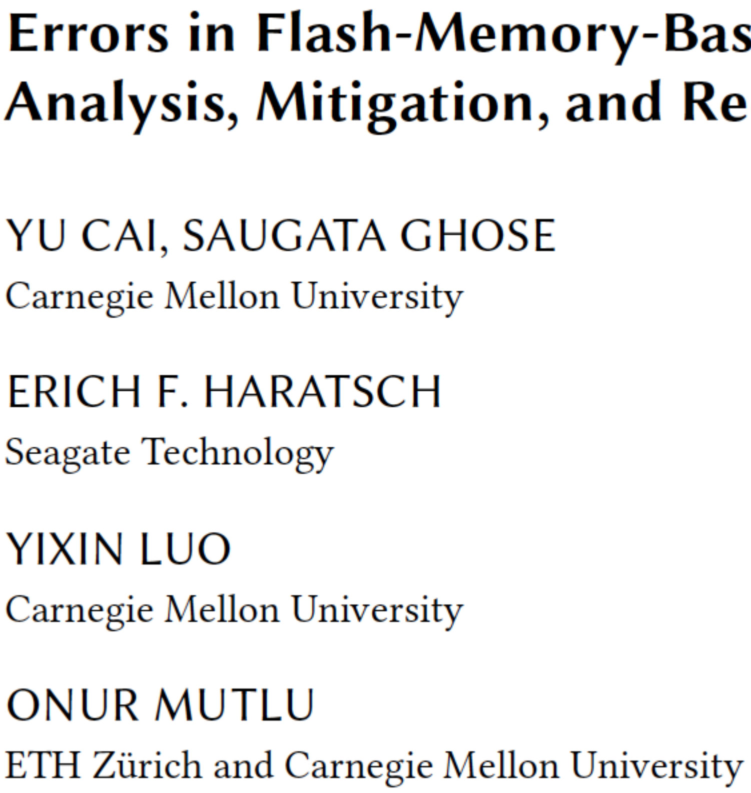
This paper reviews the most recent advances in solid-state drive (SSD) error characterization, mitigation, and data recovery techniques to improve both SSD's reliability and lifetime.

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

https://arxiv.org/pdf/1706.08642

More Up-to-date Version

- Yu Cai, Saugata Ghose, Erich F. Haratsch, Yixin Luo, and Onur Mutlu, **"Errors in Flash-Memory-Based Solid-State Drives: Analysis, Mitigation, and Recovery"**
  Invited Book Chapter in Inside Solid State Drives, 2018.
  [Preliminary arxiv.org version]



Errors in Flash-Memory-Based Solid-State Drives: Analysis, Mitigation, and Recovery

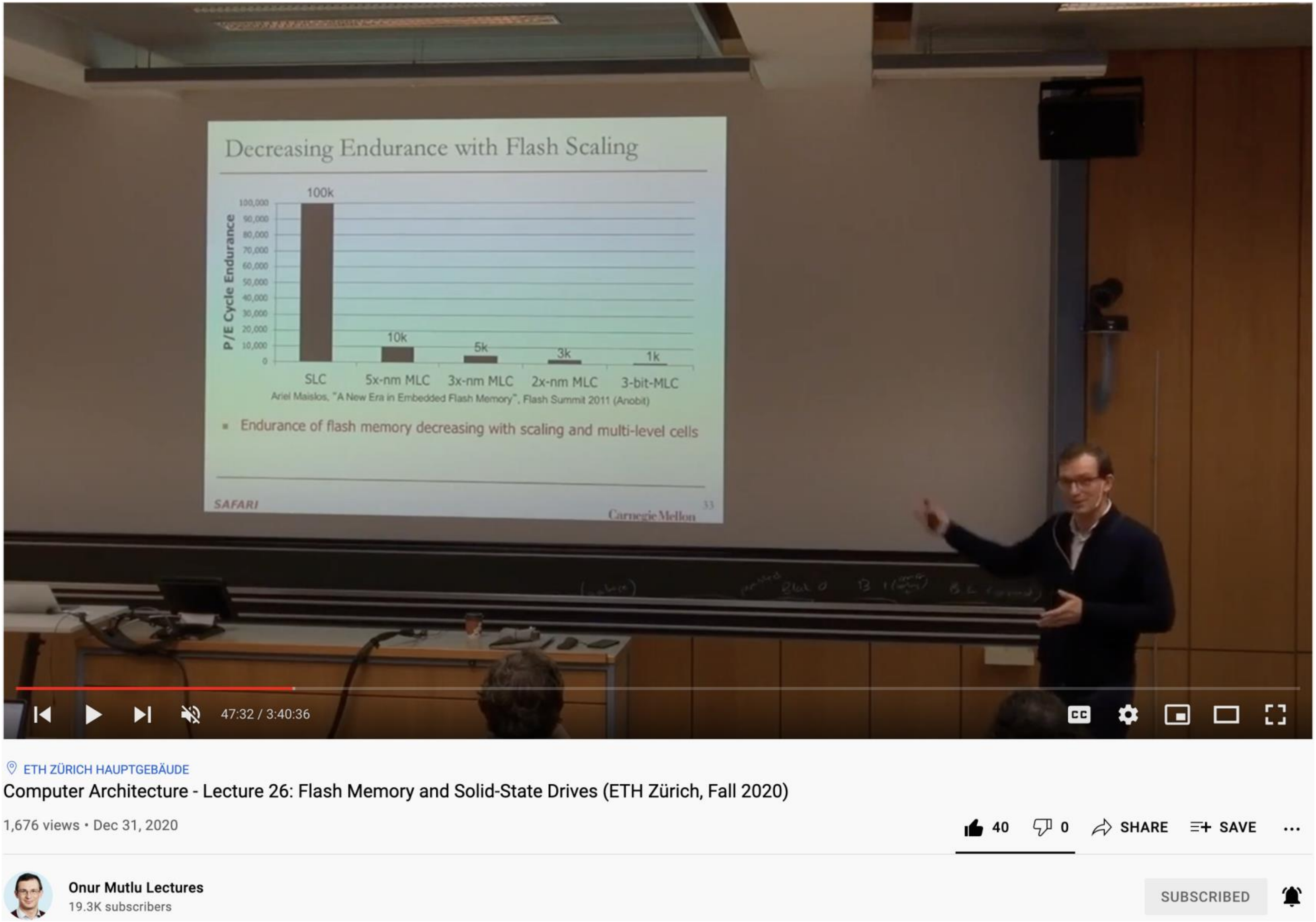
YU CAI, SAUGATA GHOSE
Carnegie Mellon University

ERICH F. HARATSCH
Seagate Technology

YIXIN LUO
Carnegie Mellon University

ONUR MUTLU
ETH Zürich and Carnegie Mellon University

More on Flash Memory Issues



Decreasing Endurance with Flash Scaling

Ariel Maislos, "A New Era in Embedded Flash Memory", Flash Summit 2011 (Anobit)

- Endurance of flash memory decreasing with scaling and multi-level cells

ETH ZÜRICH HAUPTGEBÄUDE

Computer Architecture - Lecture 26: Flash Memory and Solid-State Drives (ETH Zürich, Fall 2020)

1,676 views • Dec 31, 2020

Onur Mutlu Lectures

19.3K subscribers

https://www.youtube.com/watch?v=rninK6KWBeM&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=47