

Computer Architecture

Lecture 2a: Memory Refresh

Prof. Onur Mutlu

ETH Zürich

Fall 2019

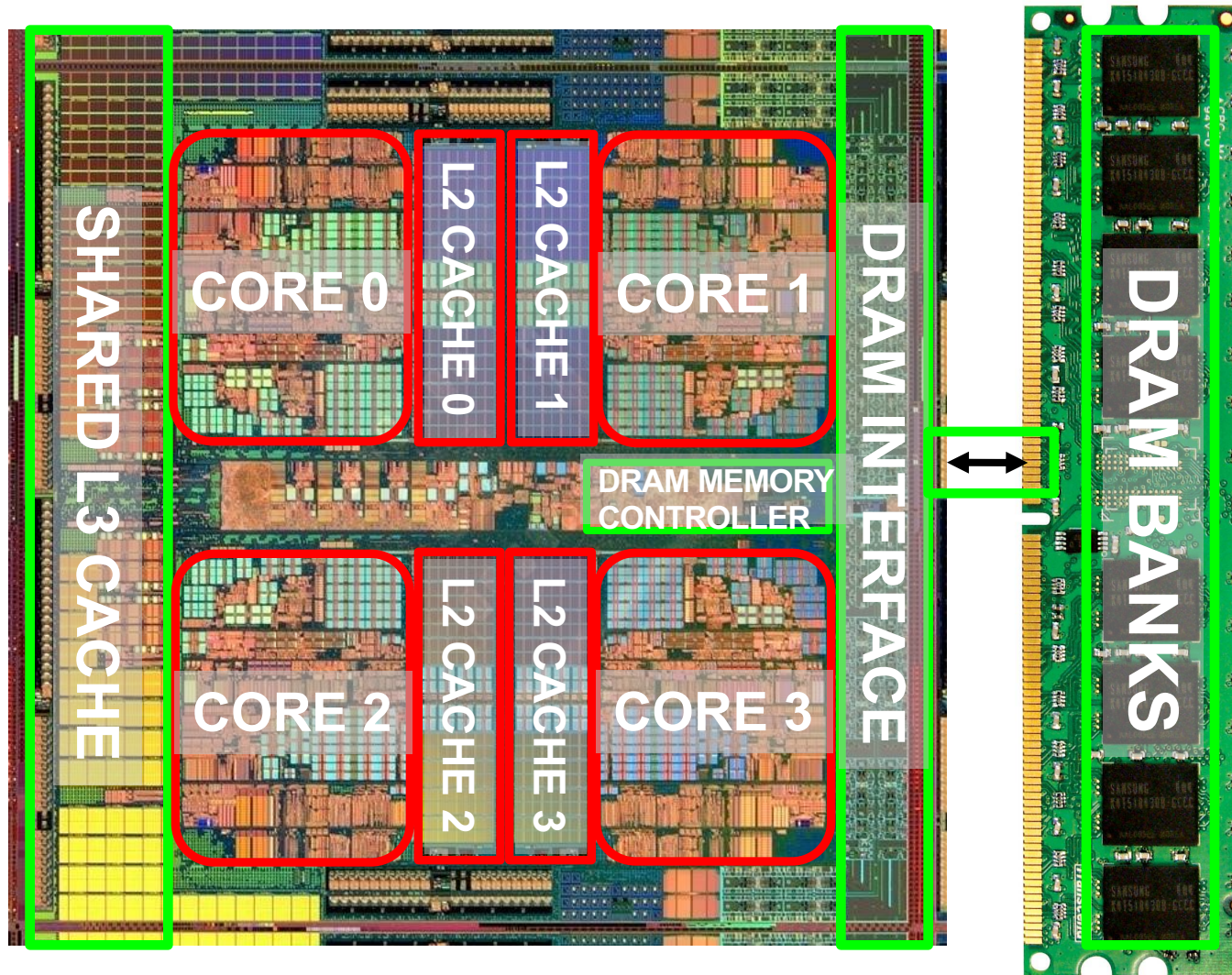
20 September 2019

Another Example

- DRAM Refresh

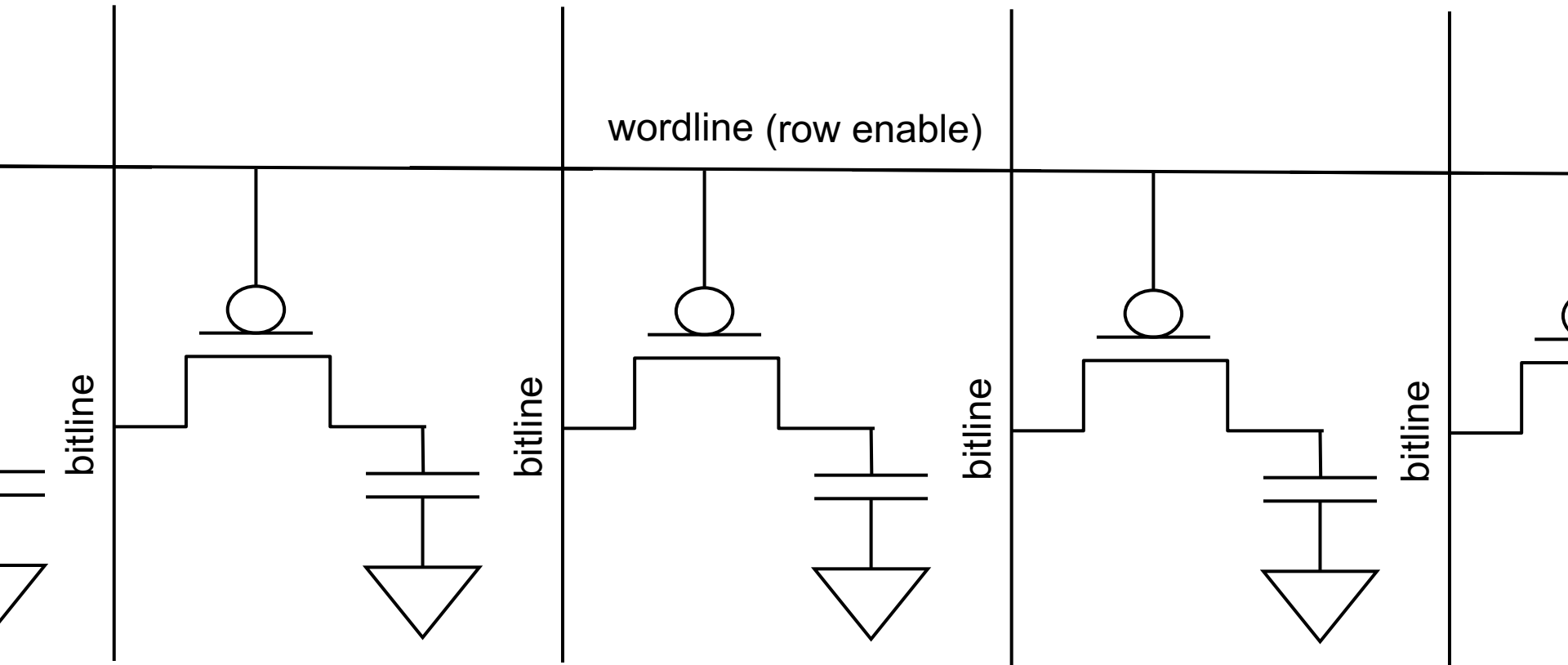
DRAM in the System

Multi-Core
Chip



*Die photo credit: AMD Barcelona

A DRAM Cell



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

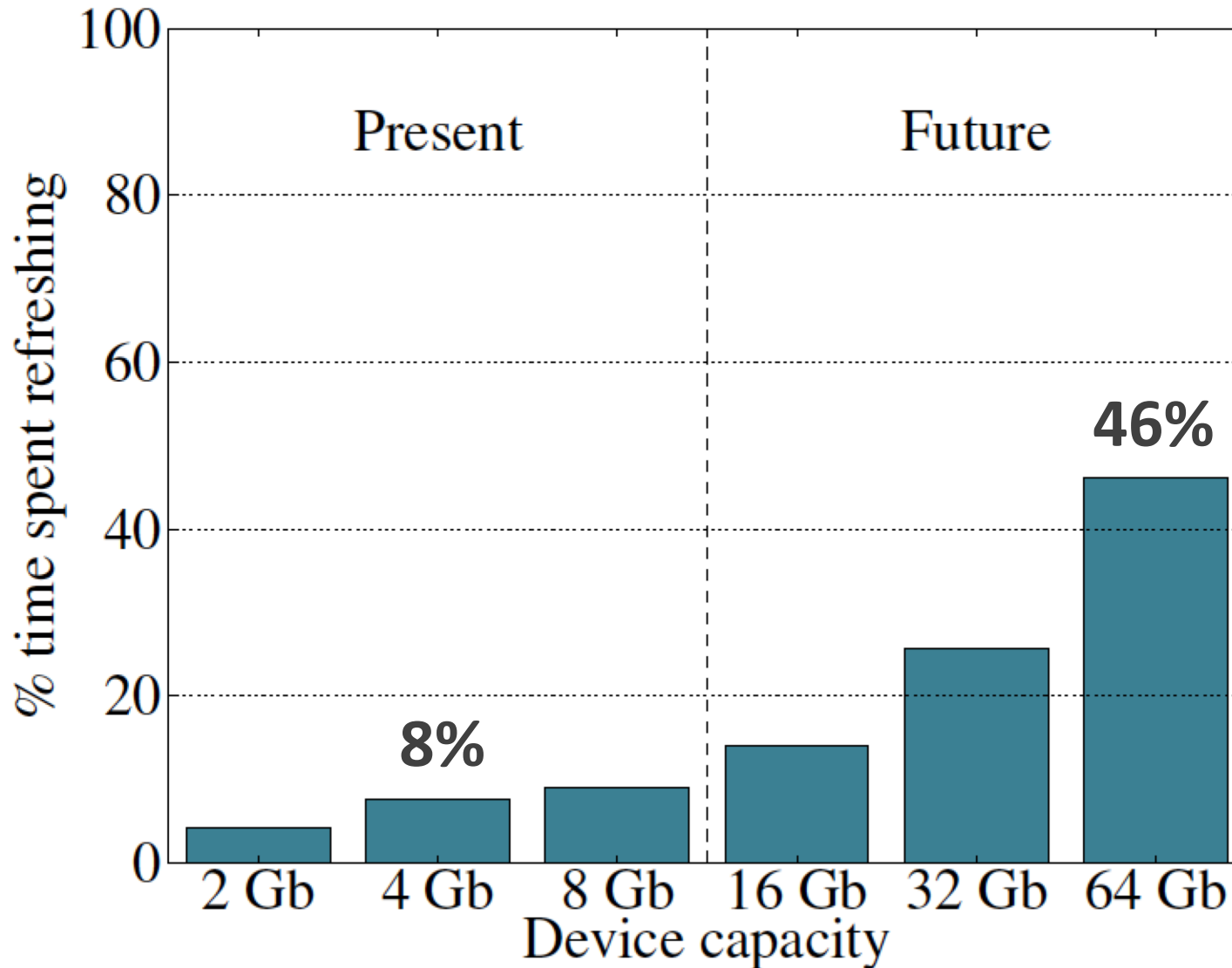
DRAM Refresh

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

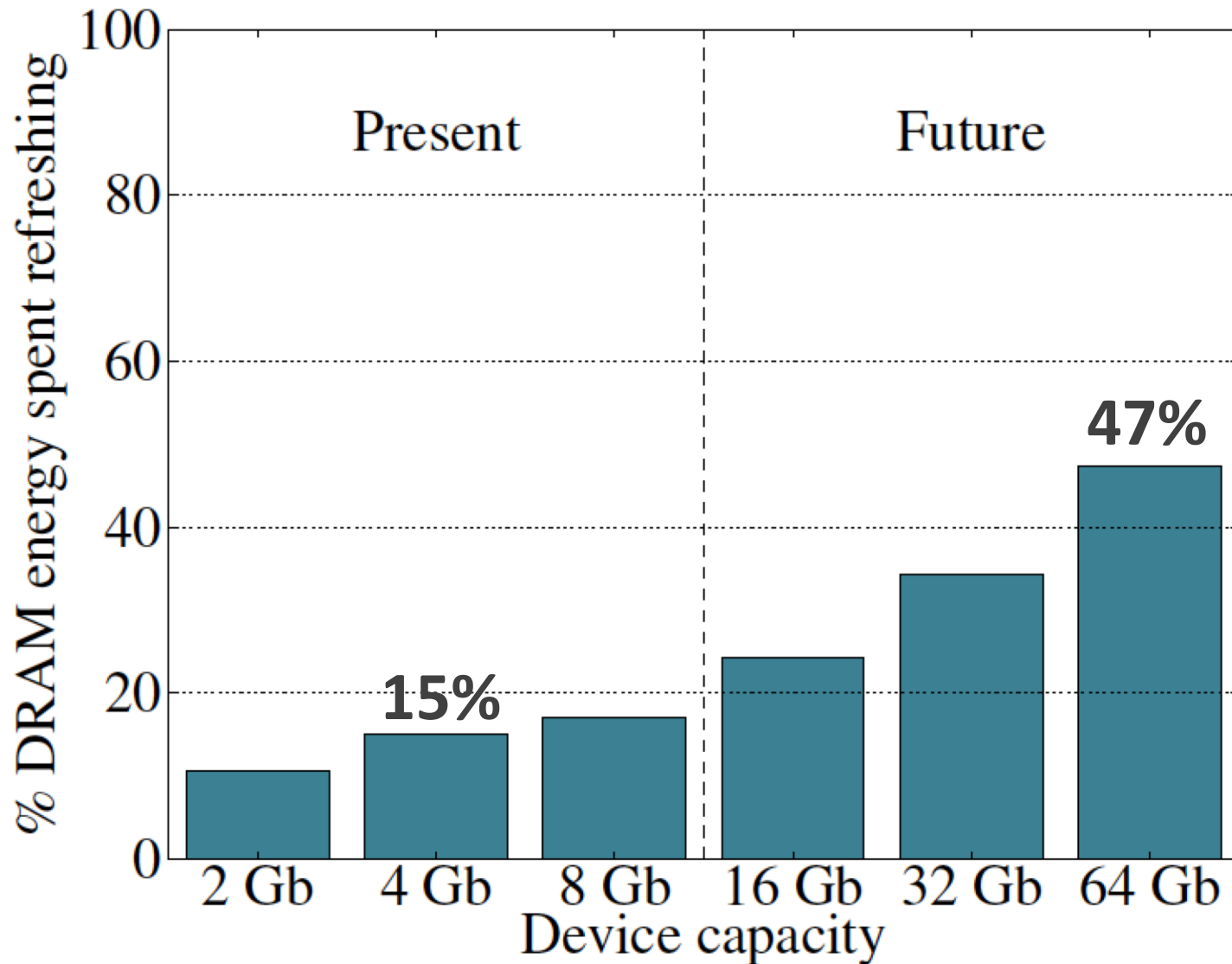
First, Some Analysis

- Imagine a system with 8 ExaByte DRAM (2^{63} bytes)
- Assume a row size of 8 KiloBytes (2^{13} bytes)
- How many rows are there?
- How many refreshes happen in 64ms?
- What is the total power consumption of DRAM refresh?
- What is the total energy consumption of DRAM refresh during a day?
- A good exercise...
- Brownie points from me if you do it...

Refresh Overhead: Performance



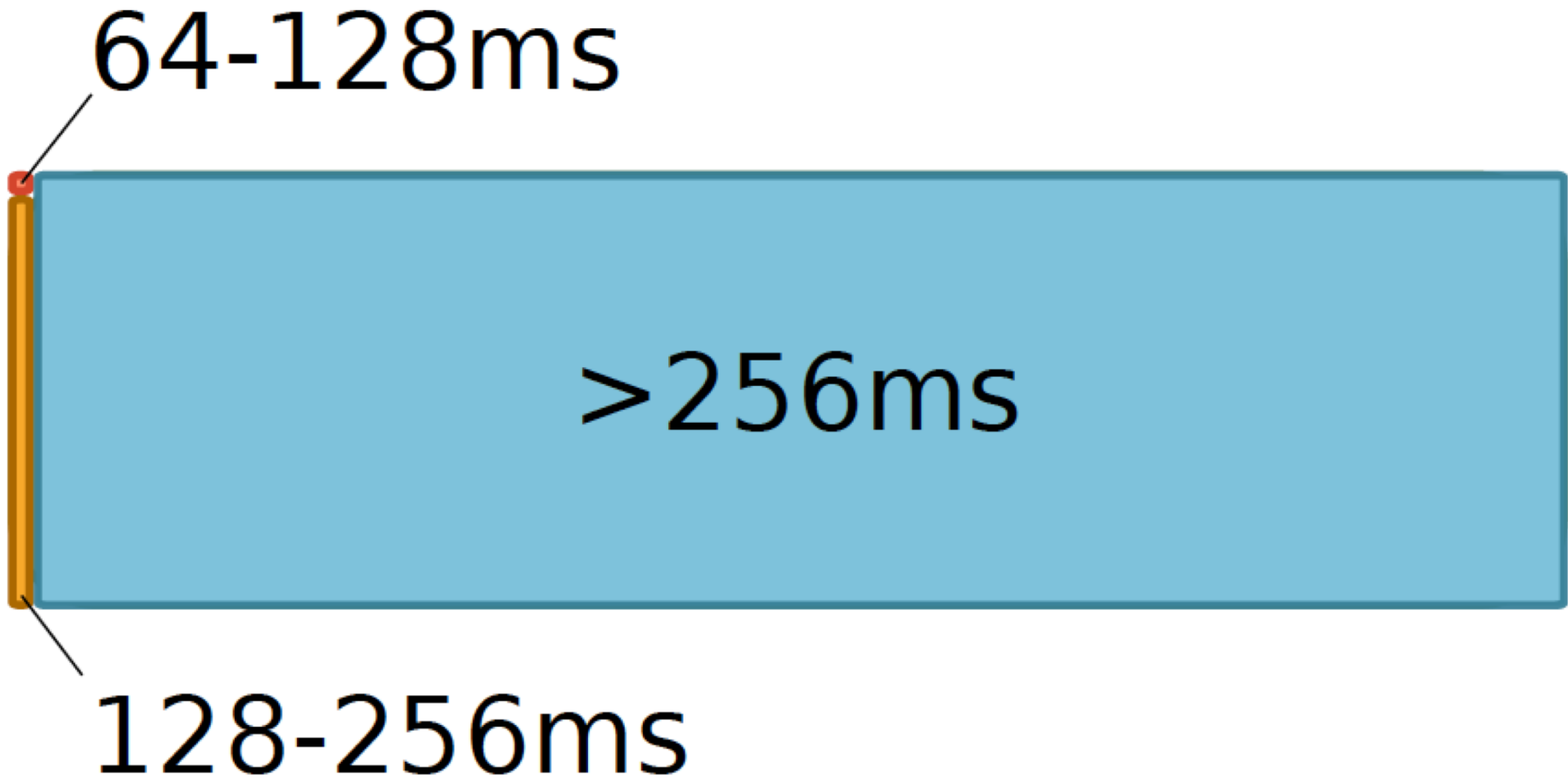
Refresh Overhead: Energy



How Do We Solve the Problem?

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

Underneath: Retention Time Profile of DRAM

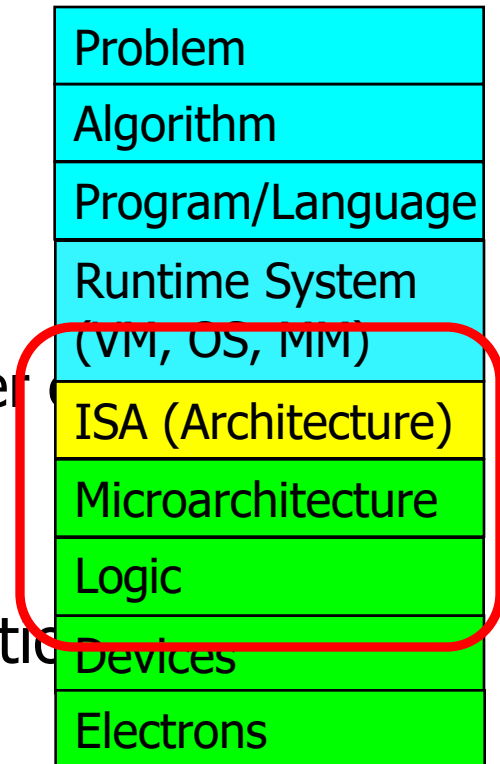


Aside: Why Do We Have Such a Profile?

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

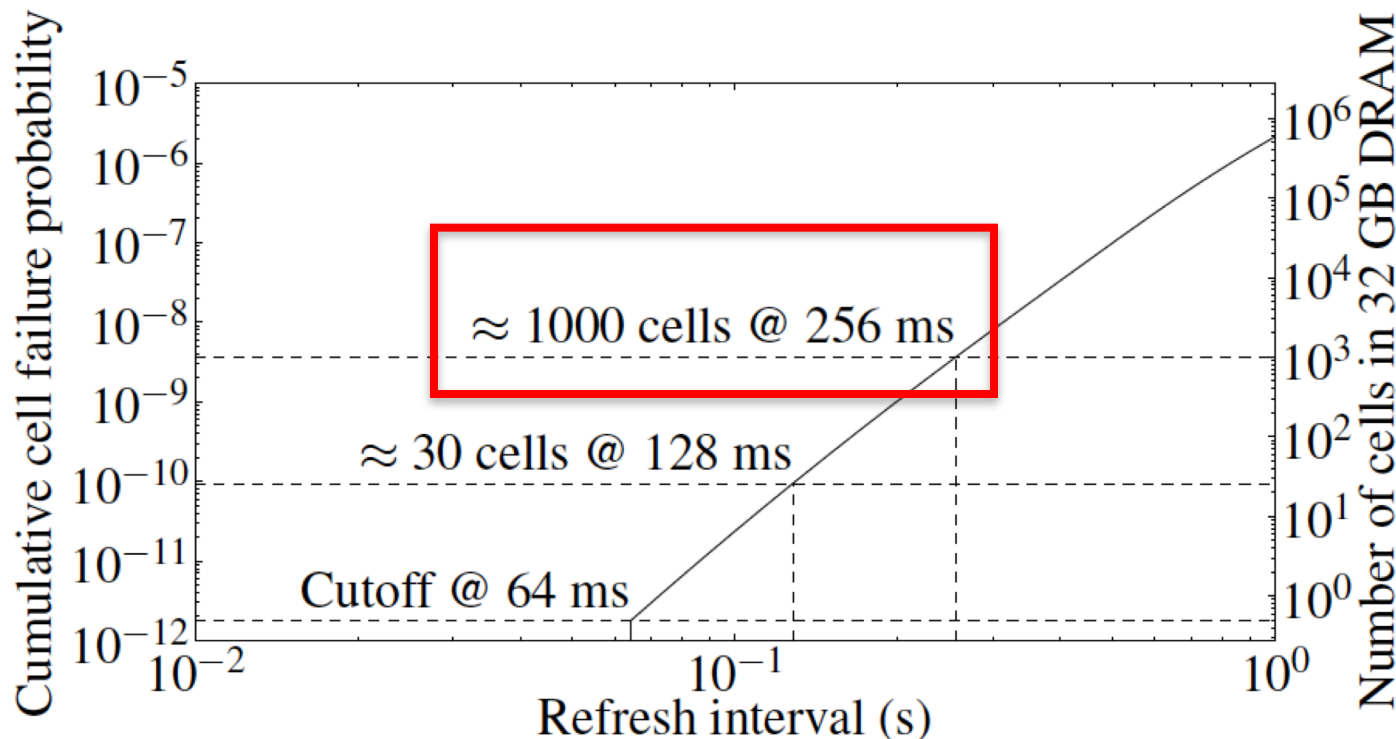
Opportunity: Taking Advantage of This Profile

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



Retention Time of DRAM Rows

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



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

RAIDR: Eliminating Unnecessary DRAM Refreshes

Liu, Jaiyen, Veras, Mutlu,
RAIDR: Retention-Aware Intelligent DRAM Refresh
ISCA 2012.

RAIDR: Mechanism

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

64-128ms

> 256ms

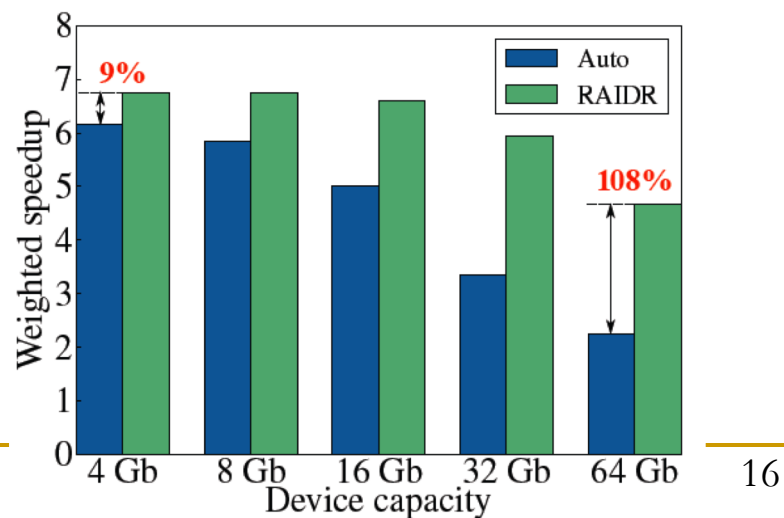
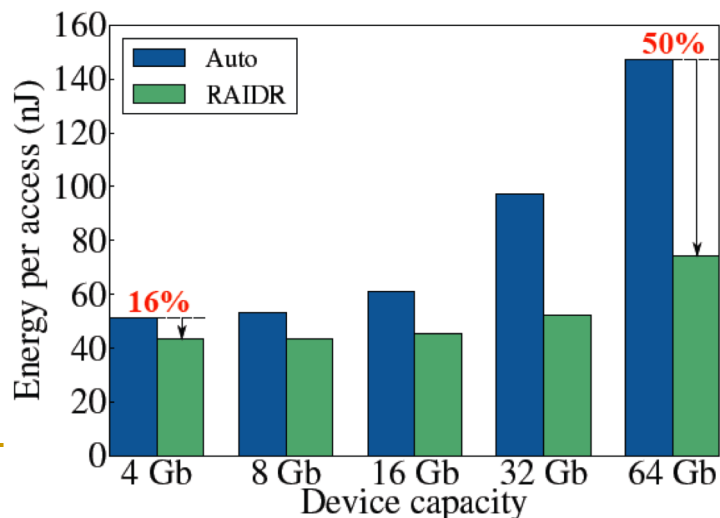
1.25KB storage in controller for 32GB DRAM memory

128-256ms

→ check the bins to determine refresh rate of a row

RAIDR: Results and Takeaways

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



Reading on RAIDR

- 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 \(pdf\)](#)
- One potential reading for your Homework 1 assignment

RAIDR: Retention-Aware Intelligent DRAM Refresh

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

If You Are Interested ... Further Readings

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

Takeaway 1

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

Takeaway 2

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

Digging Deeper: Making RAIDR Work

“Good ideas are a dime a dozen”

“Making them work is oftentimes the real contribution”

Recall: RAIDR: Mechanism

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

→ can be done at design time or during operation

2. **Binning:** Store rows into bins by retention time

→ use Bloom Filters for efficient and scalable storage

1.25KB storage in controller for 32GB DRAM memory

3. **Refreshing:** Memory controller refreshes rows in different bins at different rates

→ check the bins to determine refresh rate of a row

1. Profiling

To profile a row:

1. Write data to the row
2. Prevent it from being refreshed
3. Measure time before data corruption

	Row 1	Row 2	Row 3
Initially	11111111...	11111111...	11111111...
After 64 ms	11111111...	11111111...	11111111...
After 128 ms	11011111... (64–128ms)	11111111...	11111111...
After 256 ms		11111011... (128–256ms)	11111111... (>256ms)

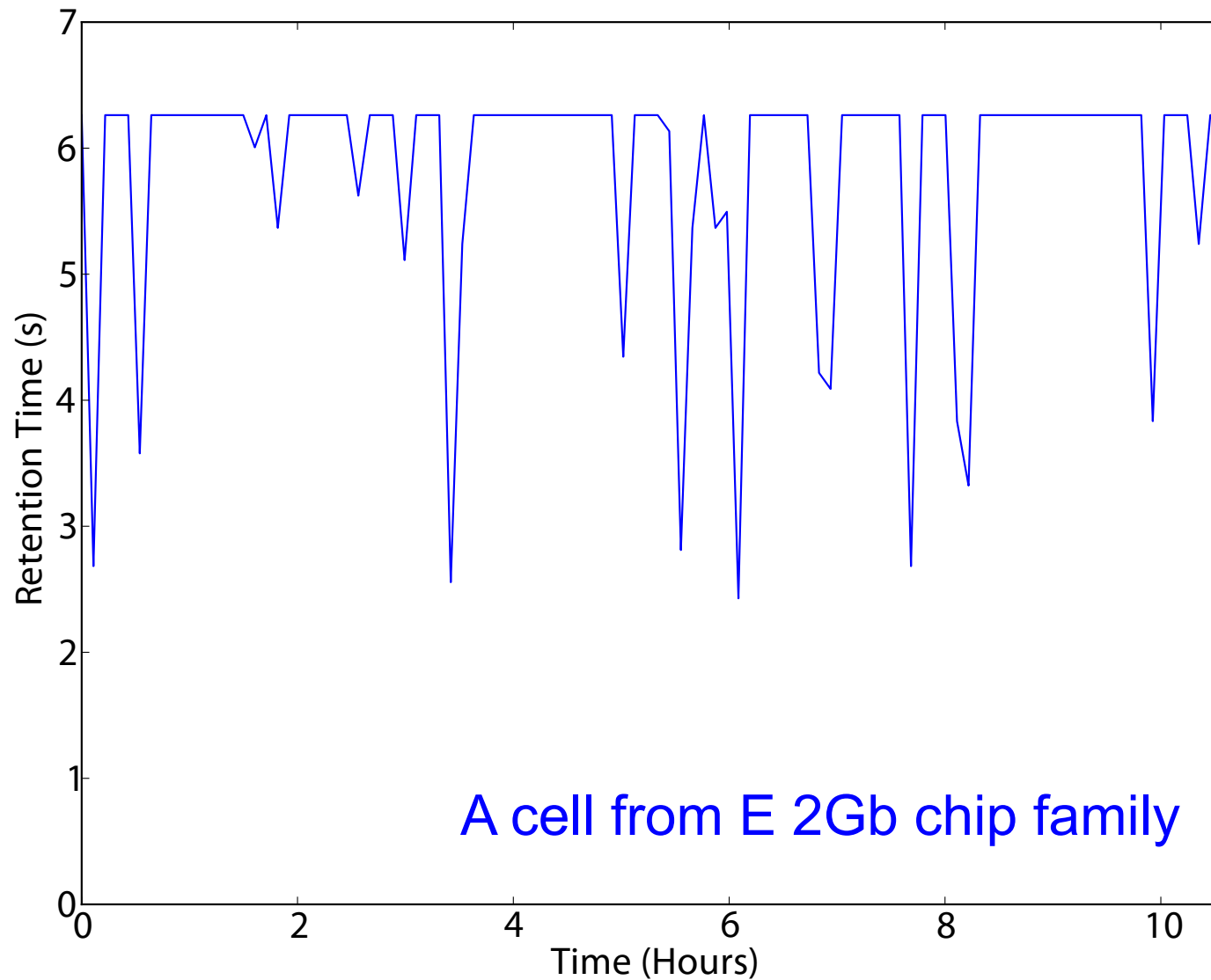
DRAM Retention Time Profiling

- Q: Is it really this easy?
- A: Ummm, not really...

Two Challenges to Retention Time Profiling

- Data Pattern Dependence (DPD) of retention time
- Variable Retention Time (VRT) phenomenon

An Example VRT Cell



VRT: Implications on Profiling Mechanisms

- Problem 1: There does not seem to be a way of determining if a cell exhibits VRT without actually observing a cell exhibiting VRT
 - VRT is a memoryless random process [Kim+ JJAP 2010]
- Problem 2: VRT complicates retention time profiling by DRAM manufacturers
 - Exposure to very high temperatures can induce VRT in cells that were not previously susceptible
 - can happen during soldering of DRAM chips
 - manufacturer's retention time profile may not be accurate
- One option for future work: Use ECC to continuously profile DRAM online while aggressively reducing refresh rate
 - Need to keep ECC overhead in check

More on DRAM Retention Analysis

- 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

Finding DRAM Retention Failures

- How can we reliably find the retention time of all DRAM cells?
- Goals: so that we can
 - Make DRAM reliable and secure
 - Make techniques like RAIDR work
 - improve performance and energy

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

Handling Variable Retention Time [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 { <i>moin, dhkim, pnair6</i> }@ece.gatech.edu		[‡] Carnegie Mellon University { <i>samirakhan, onur</i> }@cmu.edu		

Handling Data-Dependent Failures [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^{*}

^{*}University of Virginia

Donghyuk Lee^{†‡}

[†]Carnegie Mellon University

Onur Mutlu^{*†}

[‡]Nvidia

^{*}ETH Zürich

Handling Data-Dependent Failures [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

Handling Both DPD and VRT [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

2. Binning

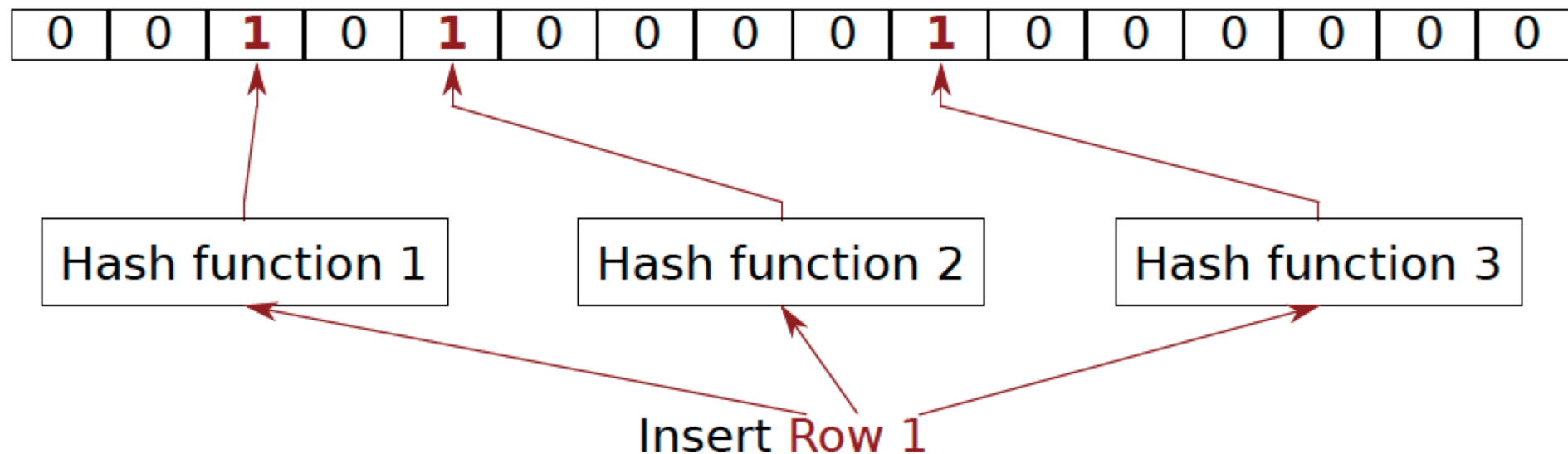
- How to efficiently and scalably store rows into retention time bins?
- Use Hardware Bloom Filters [Bloom, CACM 1970]

Bloom Filter

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

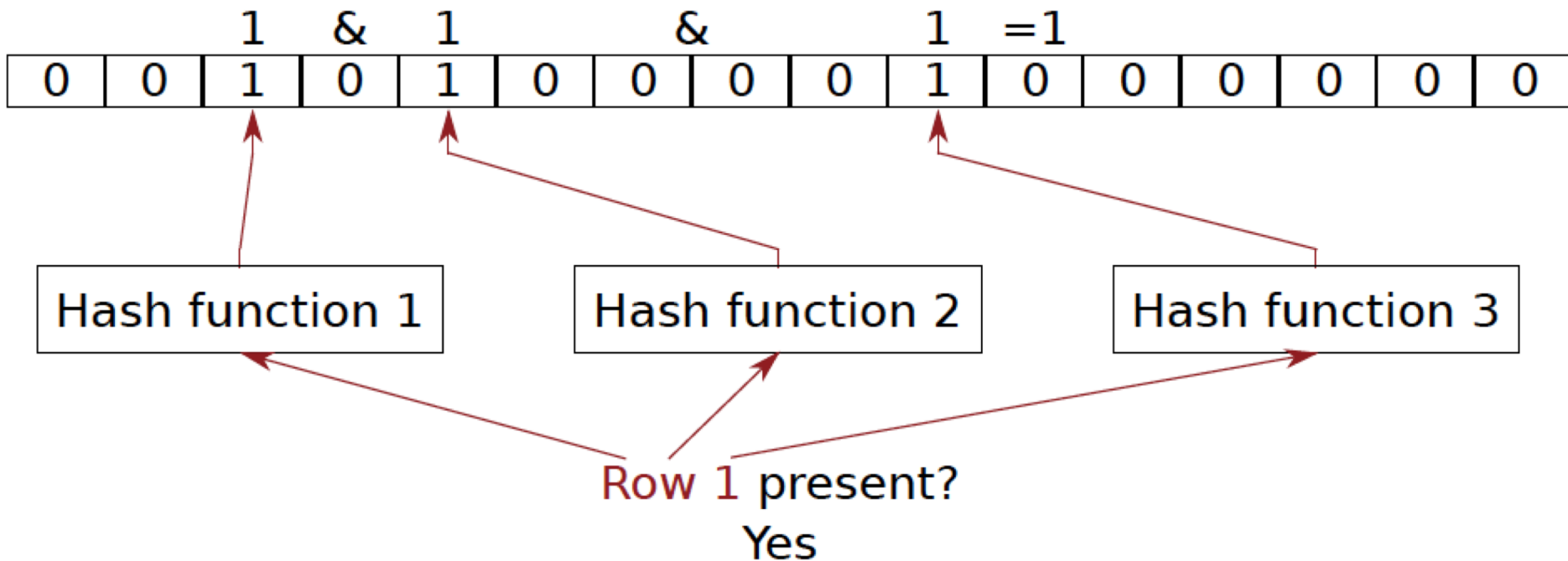
Bloom Filter Operation Example

Example with 64-128ms bin:



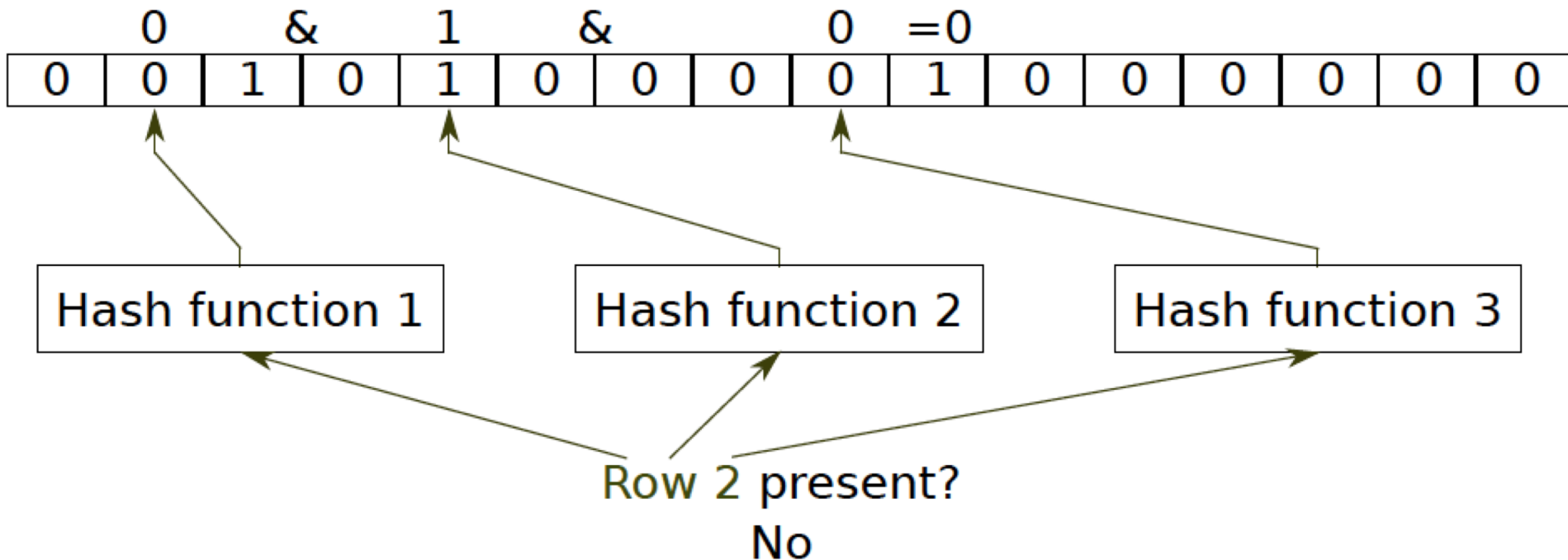
Bloom Filter Operation Example

Example with 64-128ms bin:



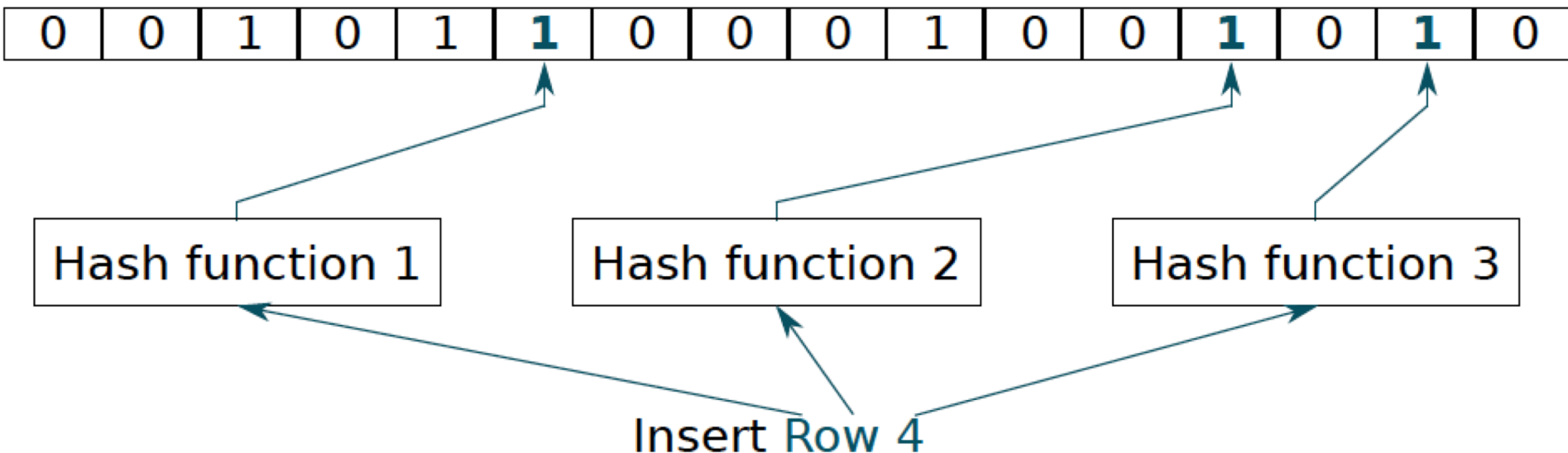
Bloom Filter Operation Example

Example with 64-128ms bin:



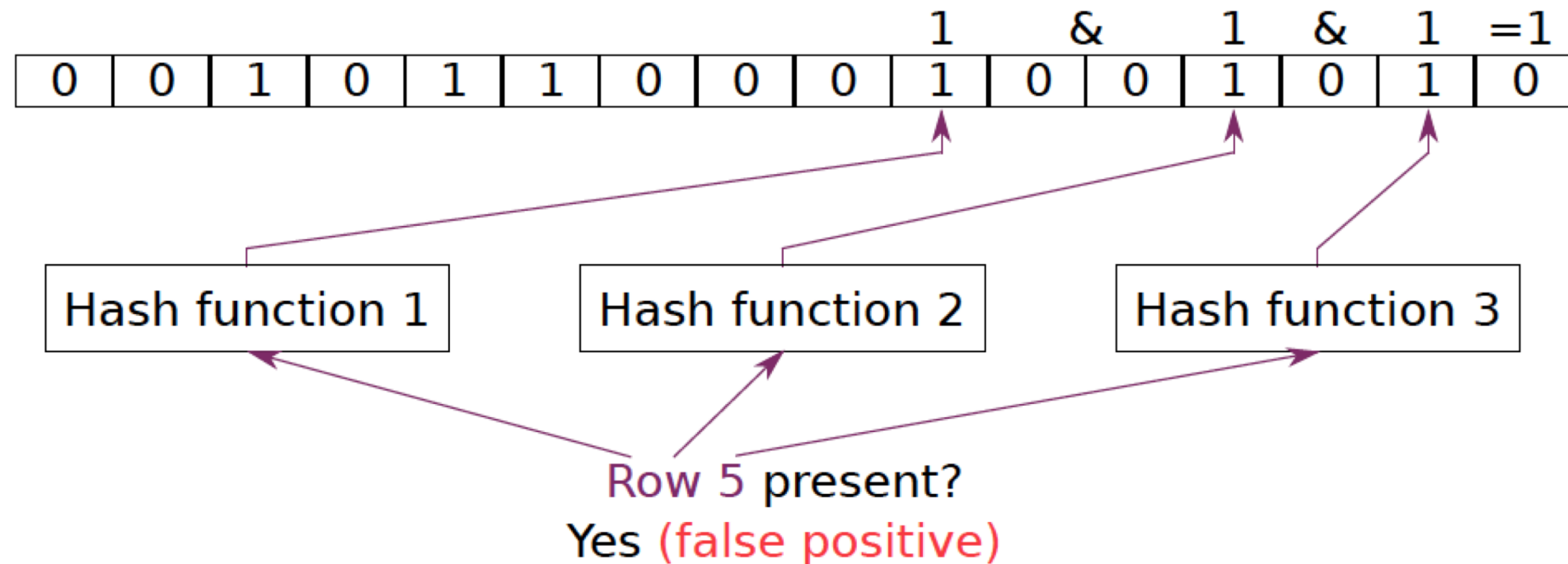
Bloom Filter Operation Example

Example with 64-128ms bin:



Bloom Filter Operation Example

Example with 64–128ms bin:



Bloom Filters

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

BURTON H. BLOOM

Computer Usage Company, Newton Upper Falls, Mass.

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

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

Bloom Filters: Pros and Cons

■ Advantages

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

■ Disadvantages

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

Not the right data structure when you cannot tolerate false positives

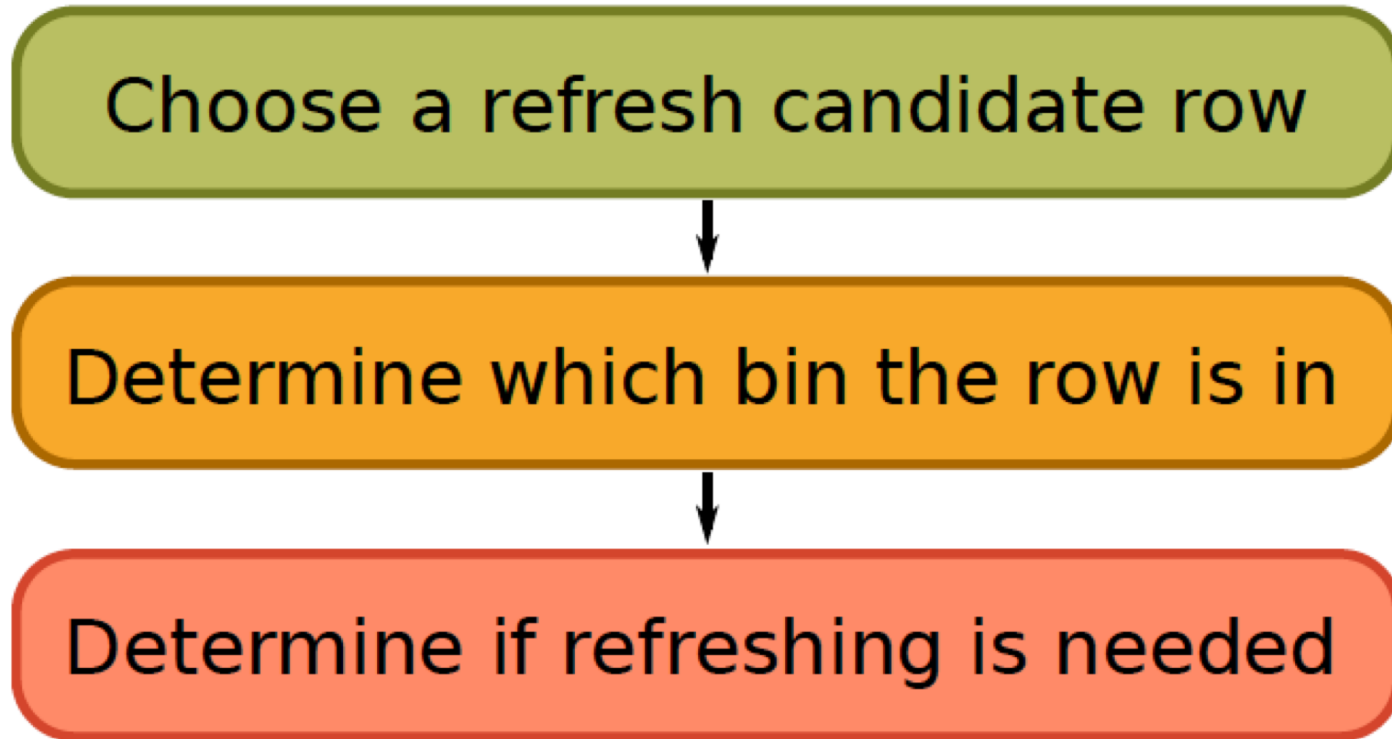
Benefits of Bloom Filters as Refresh Rate Bins

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

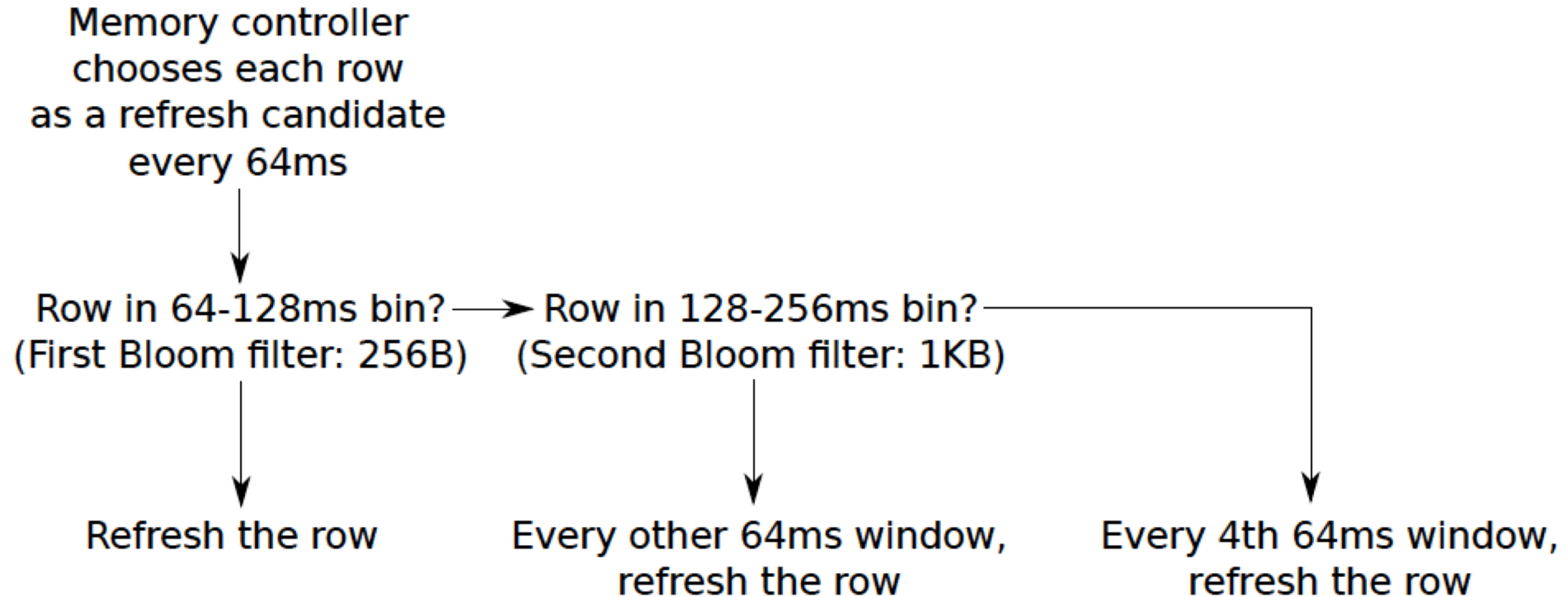
Use of Bloom Filters in Hardware

- Useful when you can tolerate false positives in set membership tests
- See the following recent examples for clear descriptions of how Bloom Filters are used
 - Liu et al., “[RAIDR: Retention-Aware Intelligent DRAM Refresh](#),” ISCA 2012.
 - Seshadri et al., “[The Evicted-Address Filter: A Unified Mechanism to Address Both Cache Pollution and Thrashing](#),” PACT 2012.

3. Refreshing (RAIDR Refresh Controller)

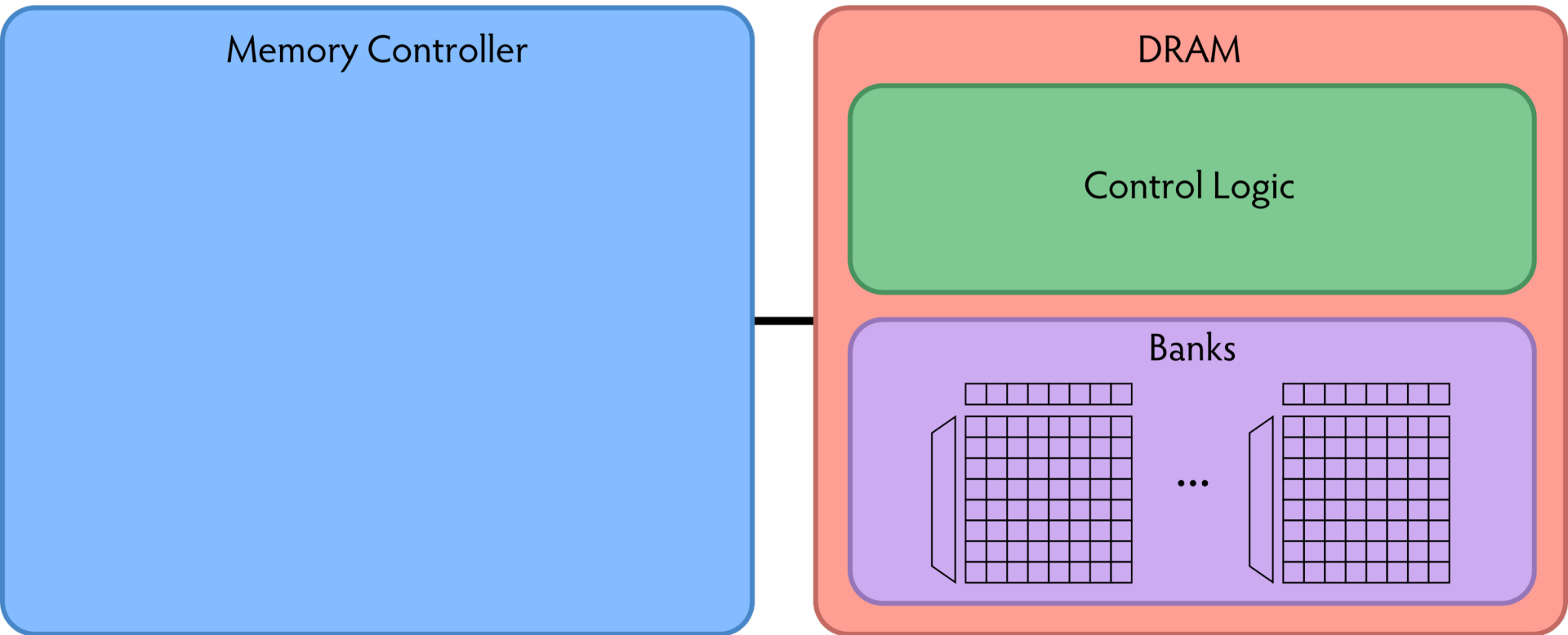


3. Refreshing (RAIDR Refresh Controller)



Liu et al., “[RAIDR: Retention-Aware Intelligent DRAM Refresh](#),” ISCA 2012.

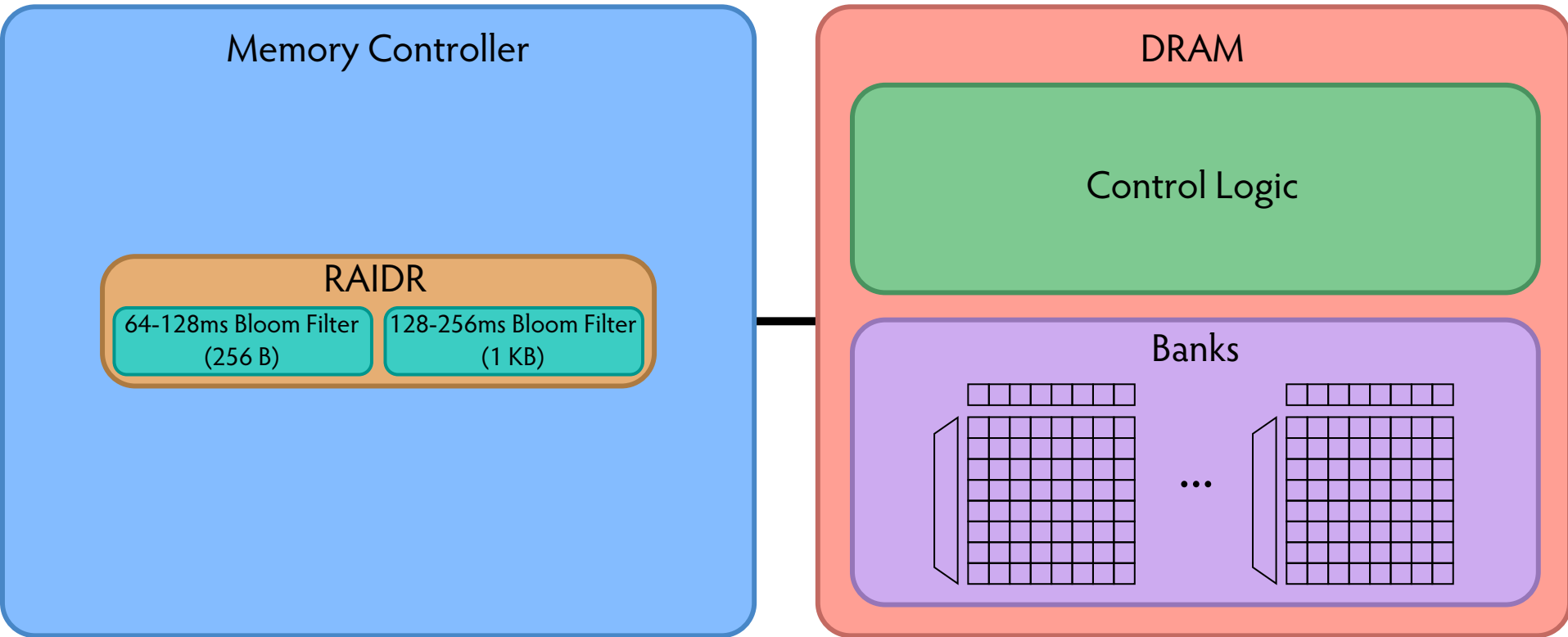
RAIDR: Baseline Design



Refresh control is in DRAM in today's auto-refresh systems

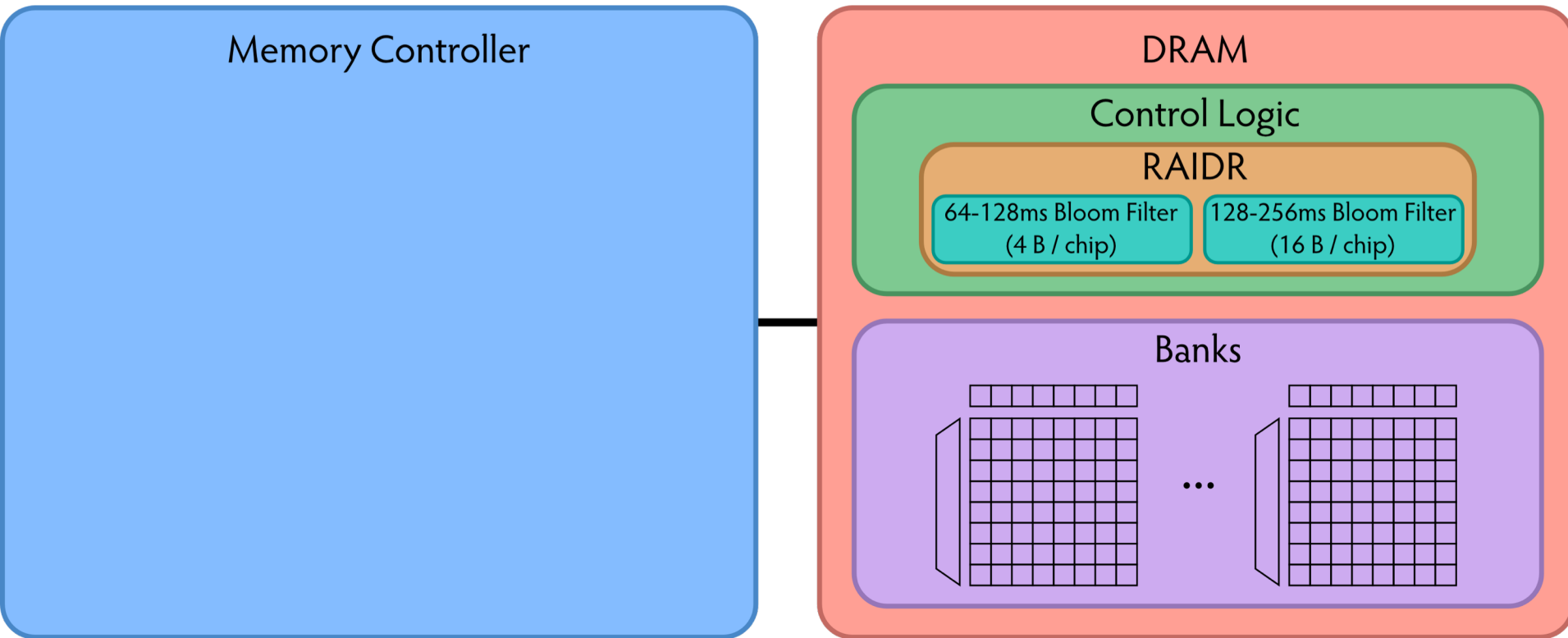
RAIDR can be implemented in either the controller or DRAM

RAIDR in Memory Controller: Option 1



Overhead of RAIDR in DRAM controller:
1.25 KB Bloom Filters, 3 counters, additional commands
issued for per-row refresh (all accounted for in evaluations)

RAIDR in DRAM Chip: Option 2



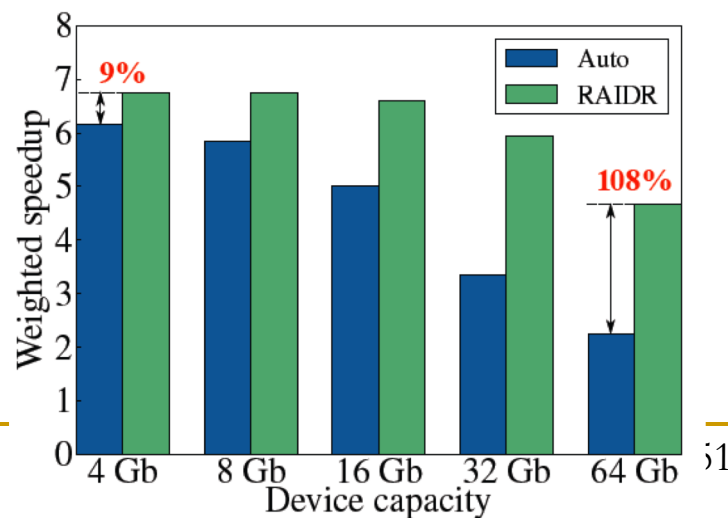
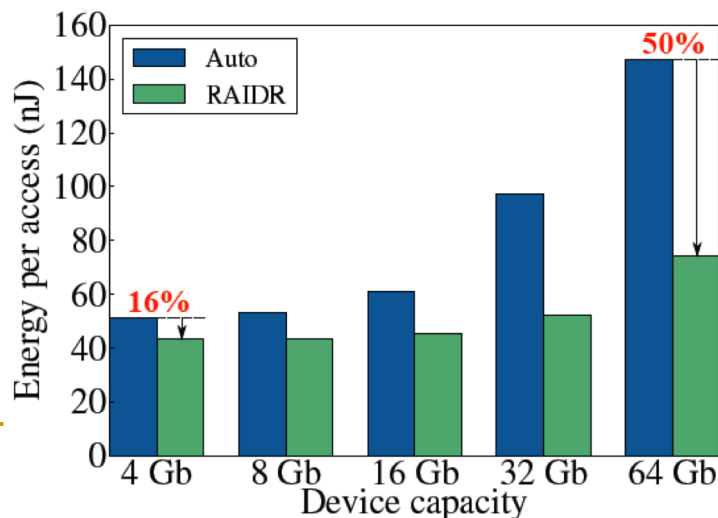
Overhead of RAIDR in DRAM chip:

Per-chip overhead: 20B Bloom Filters, 1 counter (4 Gbit chip)

Total overhead: 1.25KB Bloom Filters, 64 counters (32 GB DRAM)

RAIDR: Results and Takeaways

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



DRAM Refresh: More Questions

- What else can you do to reduce the impact of refresh?
- What else can you do if you know the retention times of rows?
- How can you accurately measure the retention time of DRAM rows?
- Recommended reading:
 - Liu et al., “An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms,” ISCA 2013.

Industry Is Writing Papers About It, Too

DRAM Process Scaling Challenges

❖ Refresh

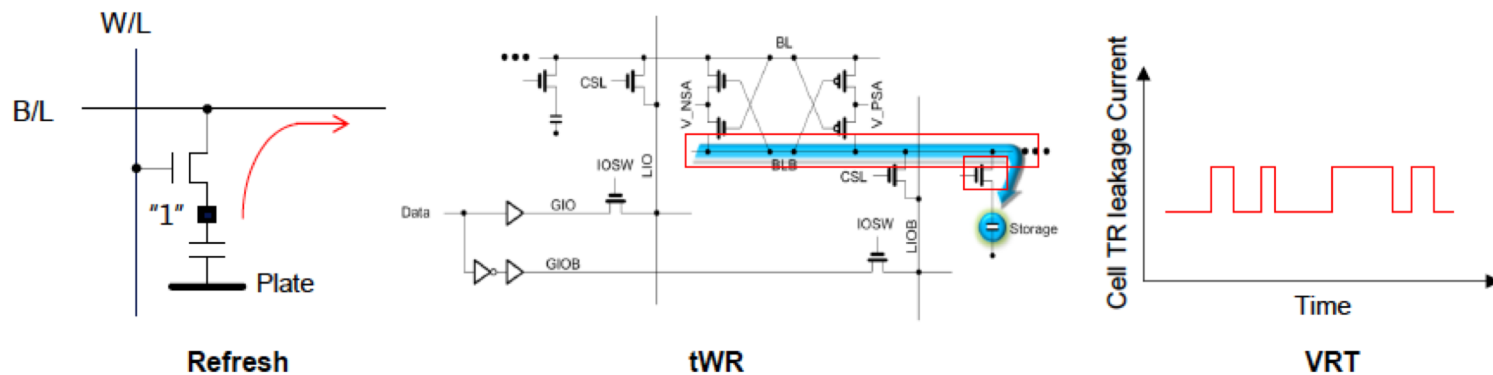
- Difficult to build high-aspect ratio cell capacitors decreasing cell capacitance
- Leakage current of cell access transistors increasing

❖ tWR

- Contact resistance between the cell capacitor and access transistor increasing
- On-current of the cell access transistor decreasing
- Bit-line resistance increasing

❖ VRT

- Occurring more frequently with cell capacitance decreasing



Call for Intelligent Memory Controllers

DRAM Process Scaling Challenges

❖ Refresh

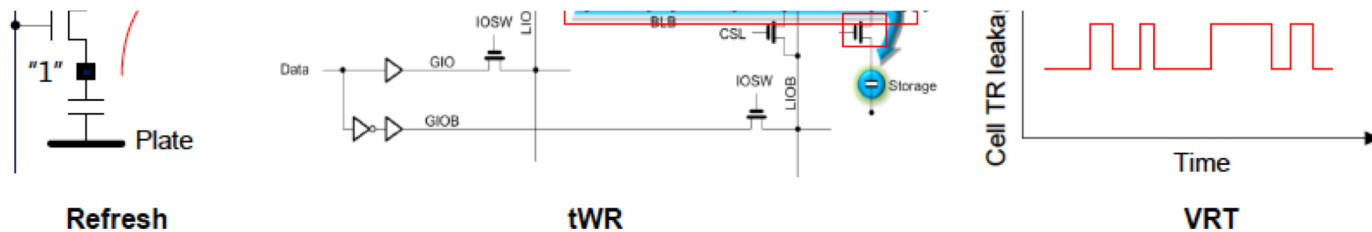
- Difficult to build high-aspect ratio cell capacitors decreasing cell capacitance

THE MEMORY FORUM 2014

Co-Architecting Controllers and DRAM to Enhance DRAM Process Scaling

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

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



We Will Dig Deeper More In This Course

“Good ideas are a dime a dozen”

“Making them work is oftentimes the real contribution”

Computer Architecture

Lecture 2a: Memory Refresh

Prof. Onur Mutlu

ETH Zürich

Fall 2019

20 September 2019