

# CSC 2224: Parallel Computer Architecture and Programming Main Memory Fundamentals

Prof. Gennady Pekhimenko

University of Toronto

Fall 2020

*The content of this lecture is adapted from the slides of  
Vivek Seshadri, Donghyuk Lee, Yoongu Kim,  
and lectures of Onur Mutlu @ ETH and CMU*

# Review #4

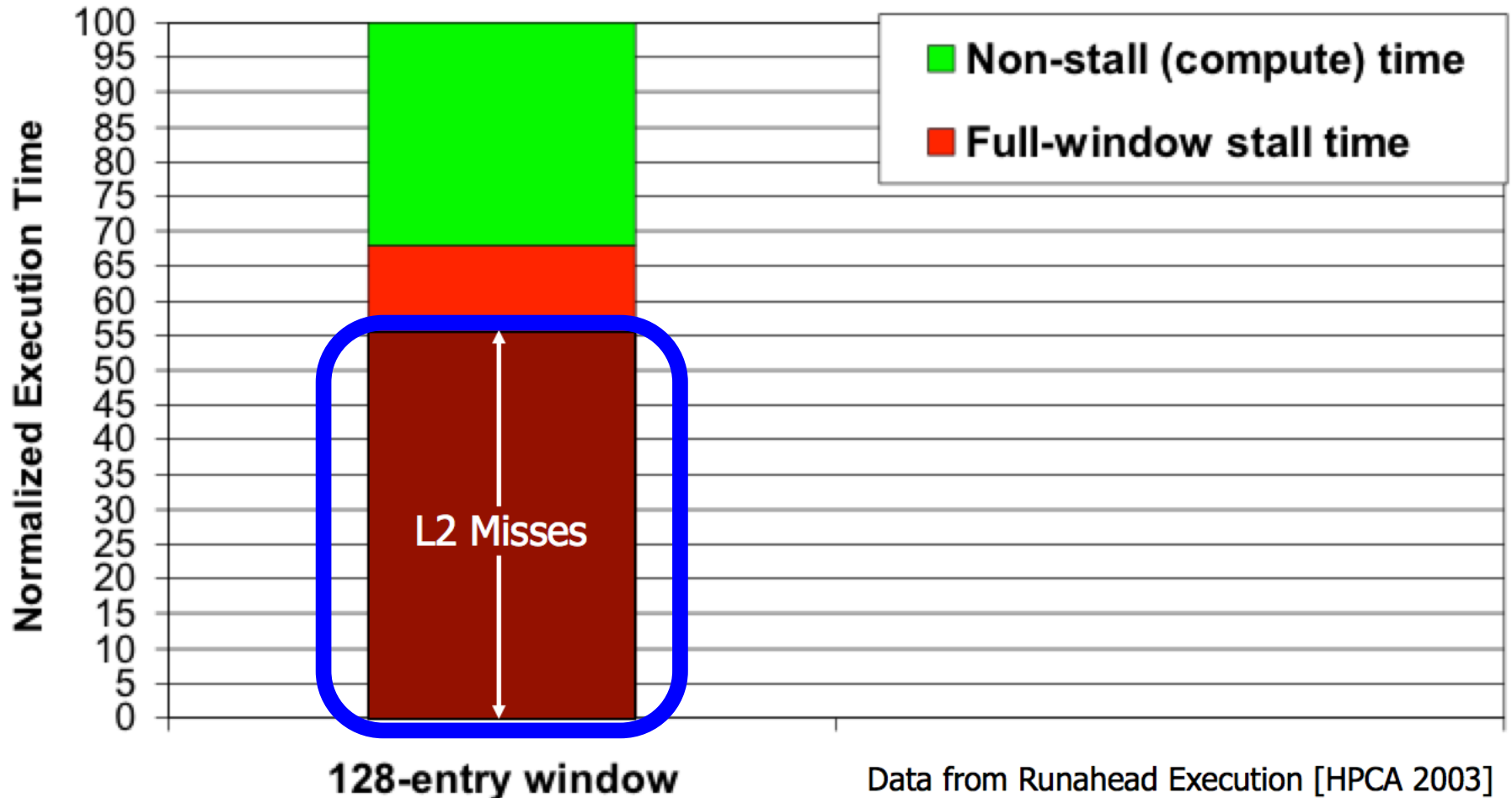
- **RowClone: Fast and Energy-Efficient In-DRAM Bulk Data Copy and Initialization**

Vivek Seshadri et al., *MICRO 2013*

# **Why Is Memory So Important? (Especially Today)**

# The Performance Perspective

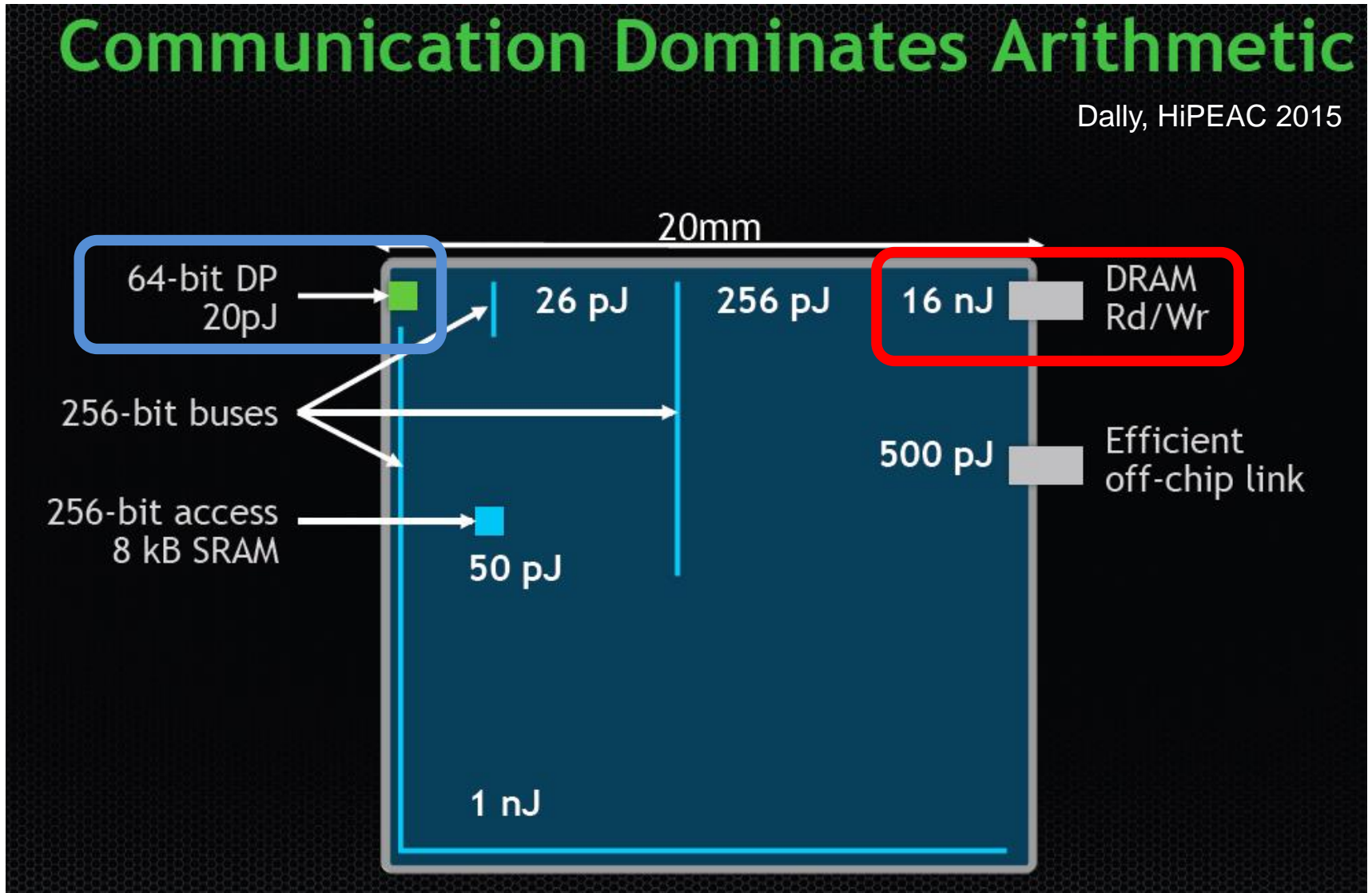
- “It’s the Memory, Stupid!” (Richard Sites, MPR, 1996)



# The Energy Perspective

## Communication Dominates Arithmetic

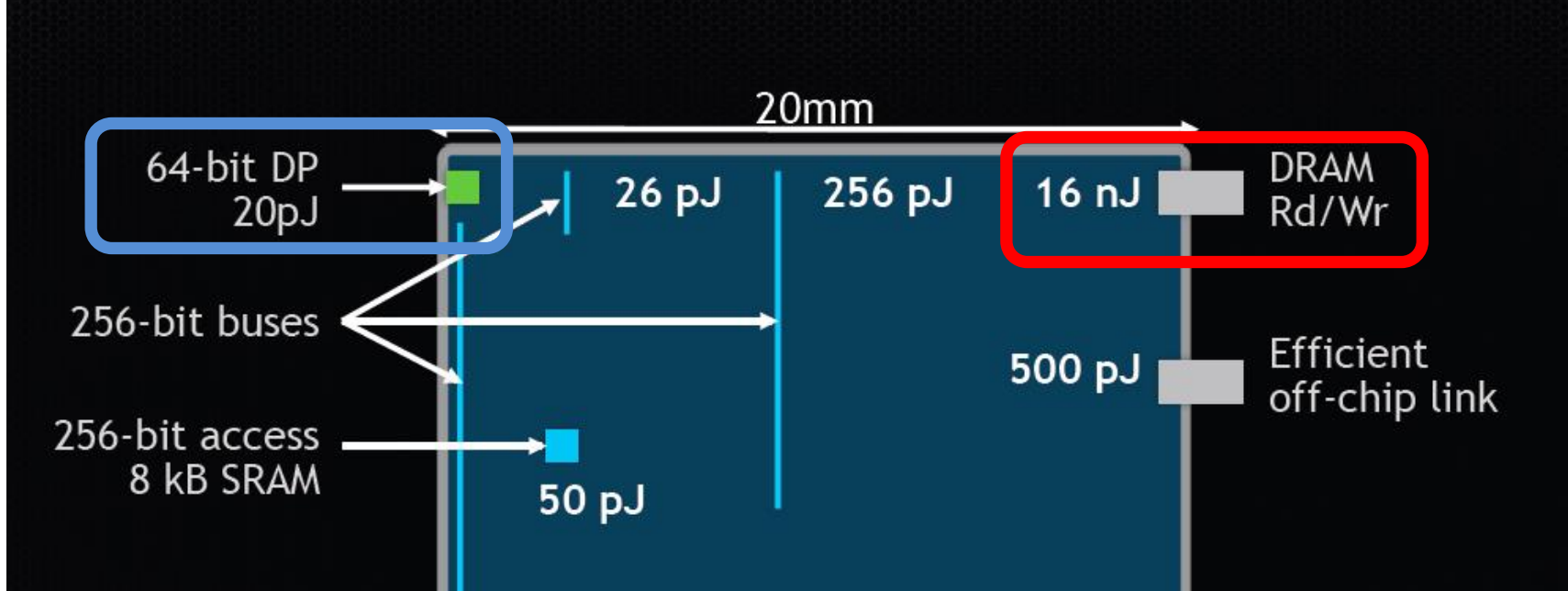
Dally, HiPEAC 2015



# The Energy Perspective

## Communication Dominates Arithmetic

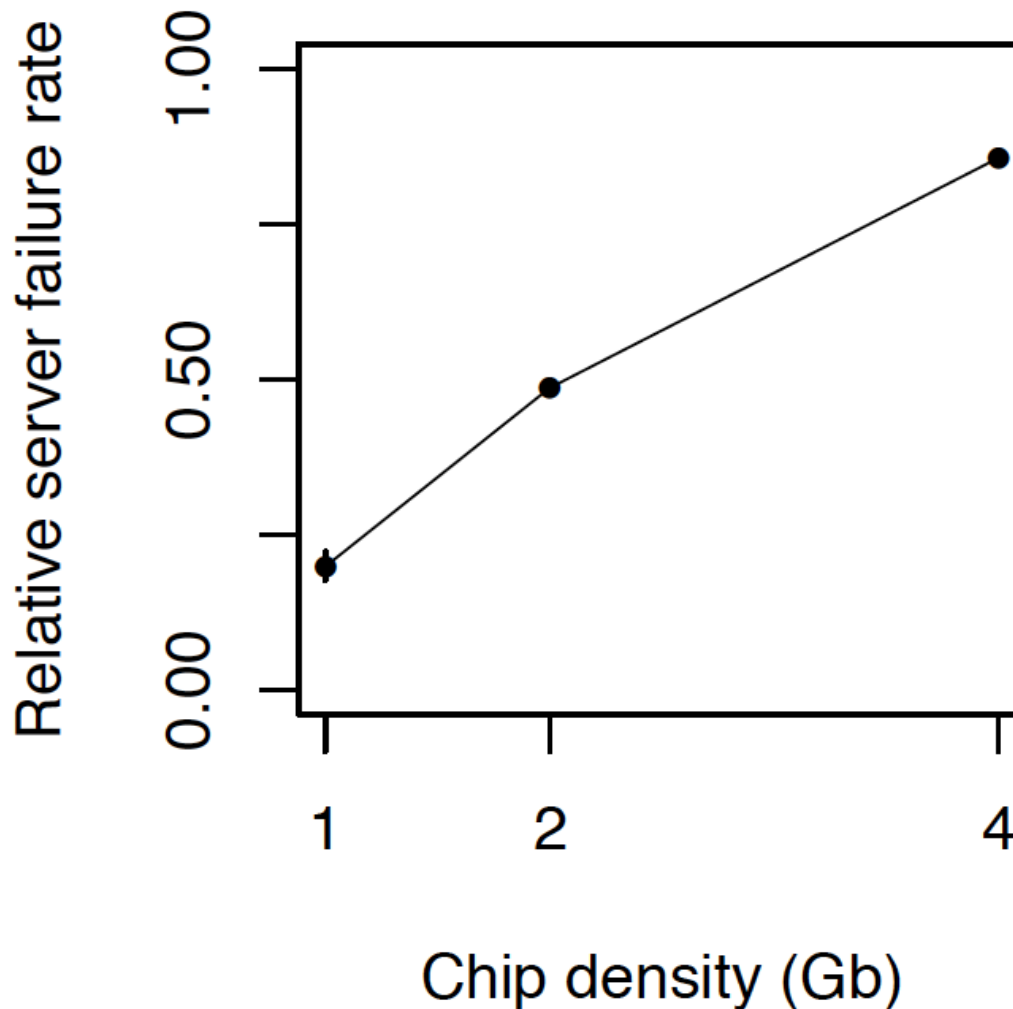
Dally, HiPEAC 2015



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

# The Reliability Perspective

- 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*

# The Security Perspective



Rowhammer

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



**Why Is DRAM *So* Slow?**

# Motivation (by Vivek Seshadri)

Conversation with a friend from Stanford

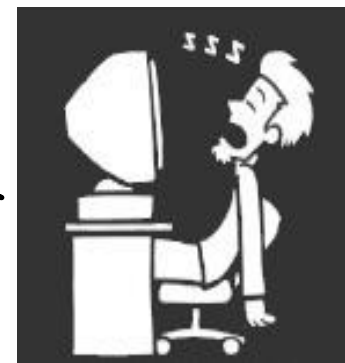


Him

Why is DRAM so slow?!

Really?

50 nanoseconds to serve one request? Is that a fundamental limit to DRAM's access latency?

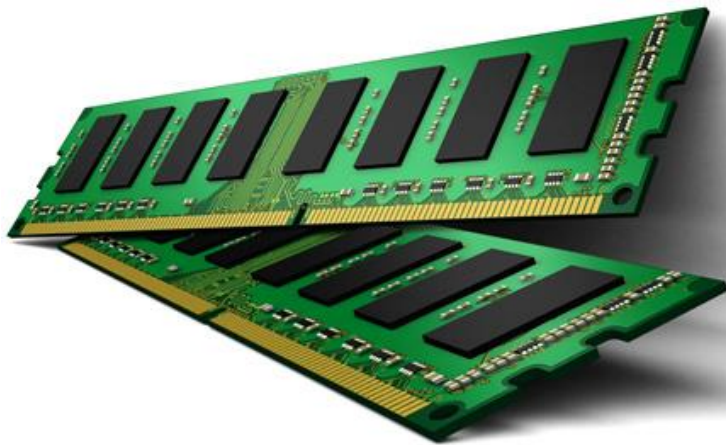


Vivek

# Understanding DRAM

DRAM

What is in here?



Problems related to today's  
DRAM design

Solutions proposed by our research

# Outline

1. What is DRAM?

2. DRAM Internal Organization

3. Problems and Solutions

- Latency (Tiered-Latency DRAM, HPCA 2013, Adaptive-Latency DRAM, HPCA 2015)
- Parallelism (Subarray-level Parallelism, ISCA 2012)

# What is DRAM?

**DRAM** – Dynamic Random Access Memory

Array of Values

3455434
43543
98734
0
847
42
873909
1729

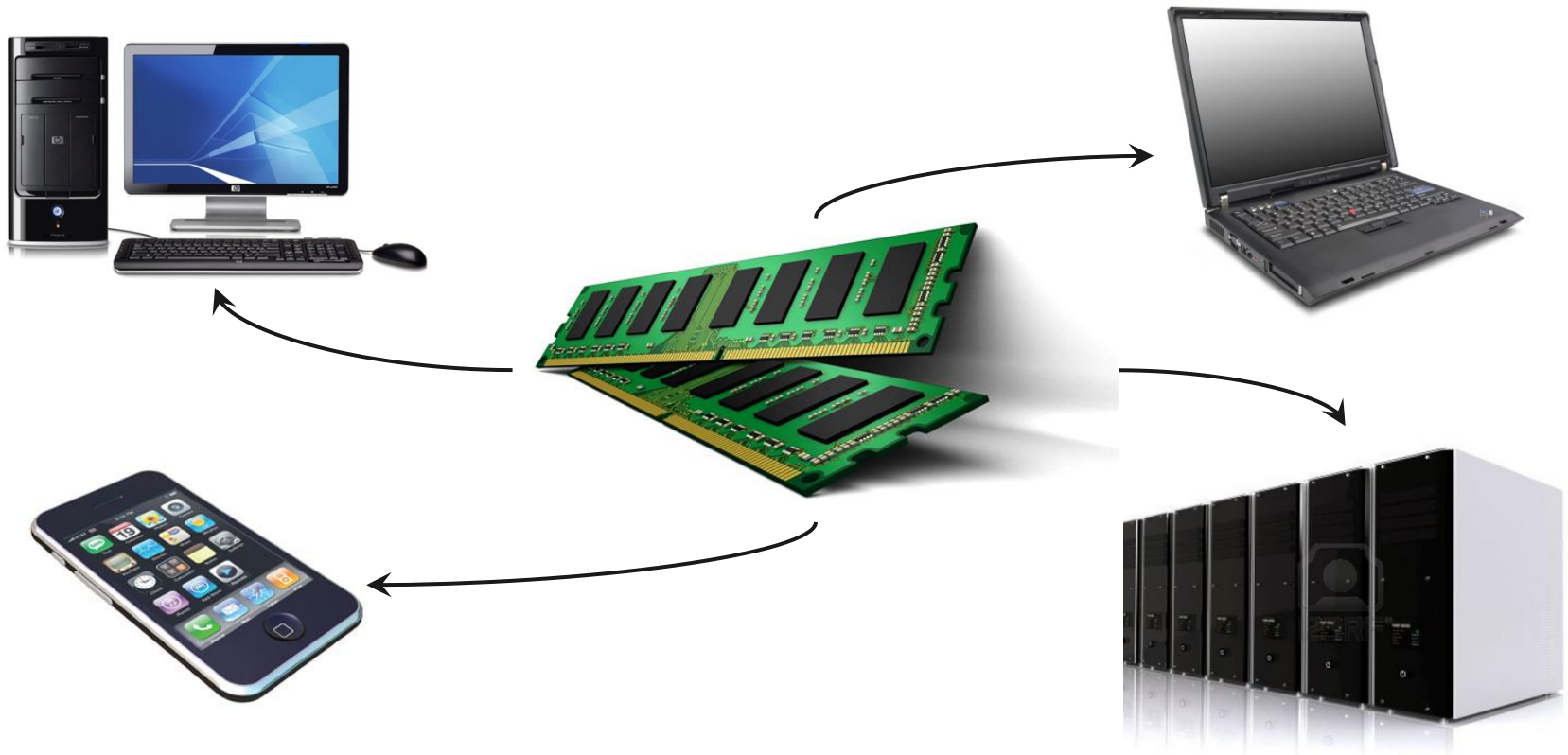
**READ** *address*

**WRITE** *address, value*

Accessing any location takes the same amount of time

Data needs to be constantly refreshed

# DRAM in Today's Systems



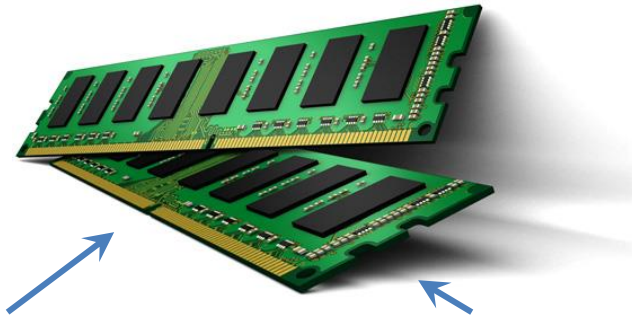
Why DRAM? Why not some other memory?

# Von Neumann Model

Processor



Memory



Program

Data

4 instruction accesses +  
3 data accesses

T1 = Read Data[1]
T2 = Read Data[2]
T3 = T1 + T2
Write Data[3], T3

3
4
8
2

Memory performance is important

# Factors that Affect Choice of Memory

## 1. Speed

- Should be reasonably fast compared to processor

## 2. Capacity

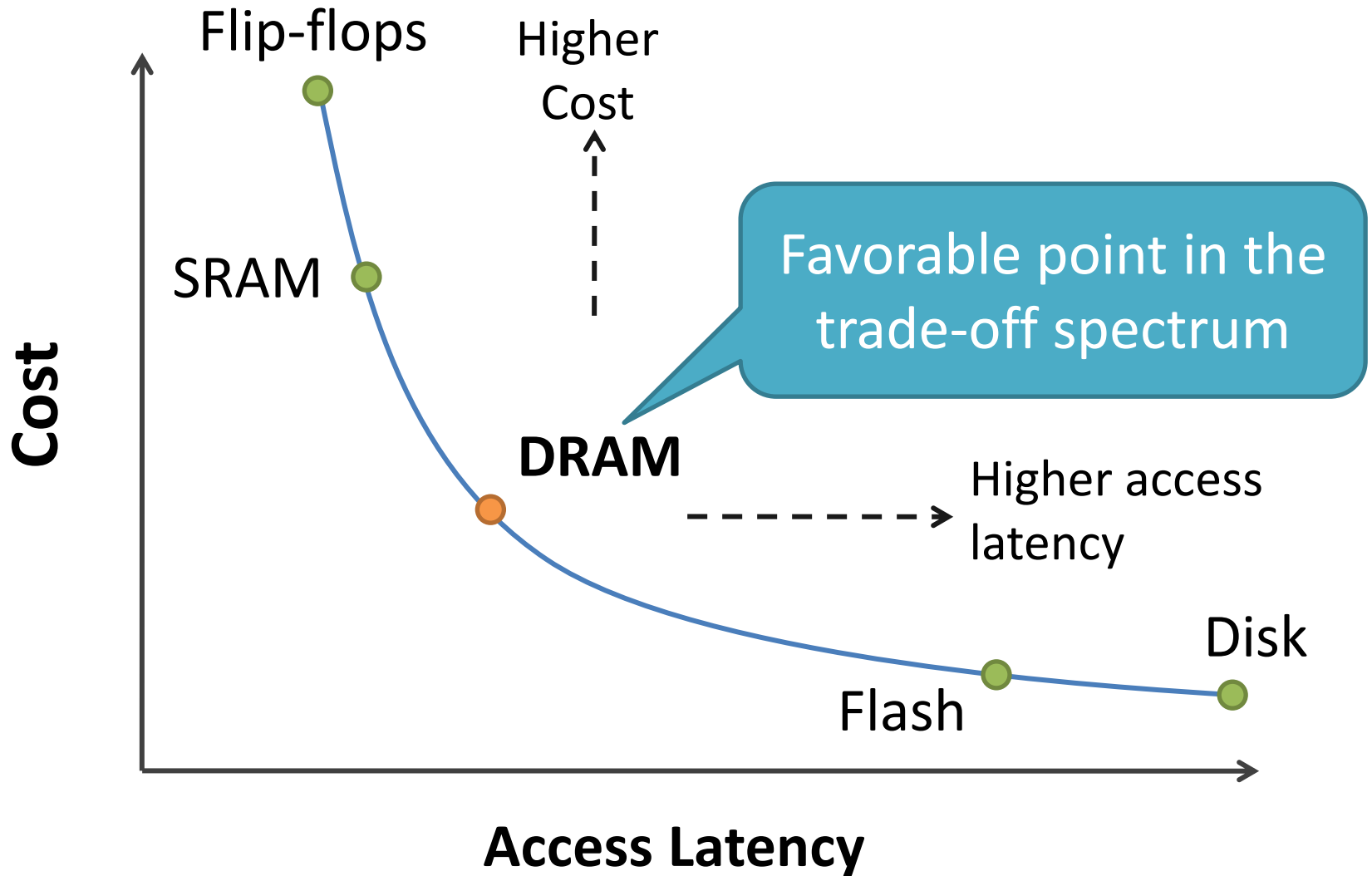
- Should be large enough to fit programs and data

## 3. Cost

- Should be cheap

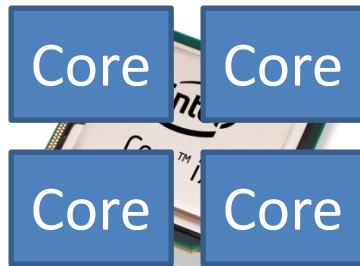


# Why DRAM?

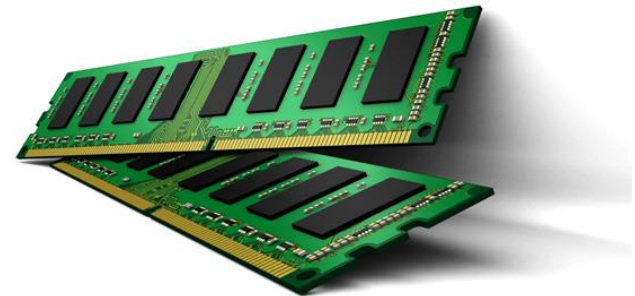


# Is DRAM Fast Enough?

Processor

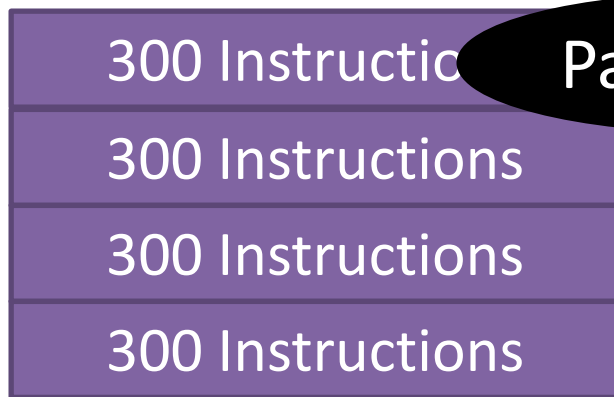


Commodity DRAM



3 GHz, 2 Instructions / cycle

50ns



Independent programs

Served in parallel?

# Outline

1. What is DRAM?

2. DRAM Internal Organization

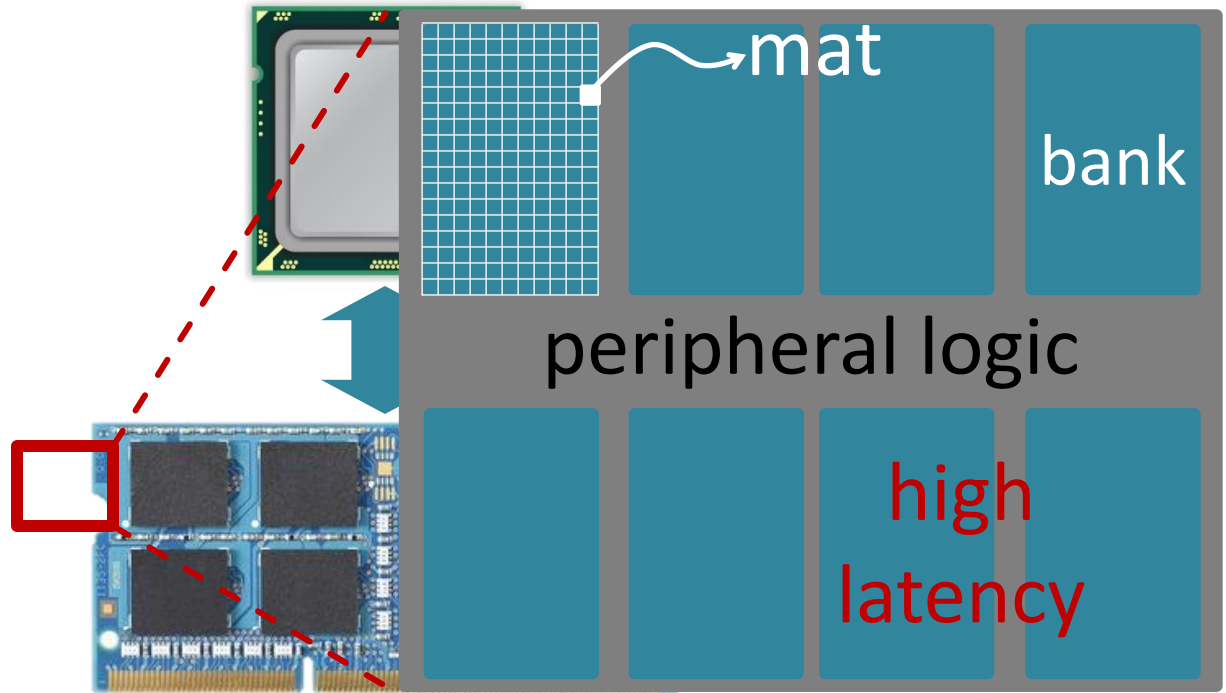
3. Problems and Solutions

- Latency (Tiered-Latency DRAM, HPCA 2013, Adaptive-Latency DRAM, HPCA 2015)
- Parallelism (Subarray-level Parallelism, ISCA 2012)

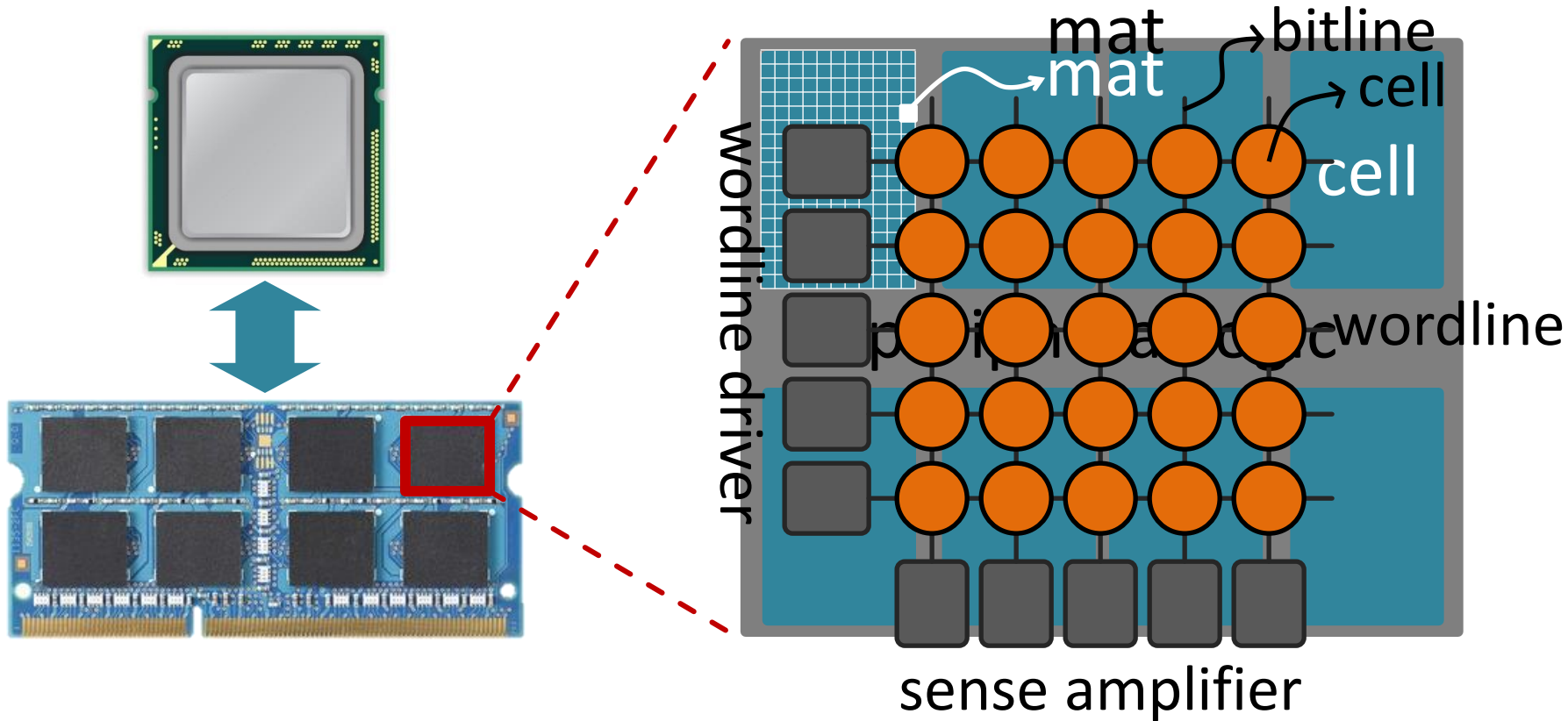
# DRAM Organization

processor

main  
memory

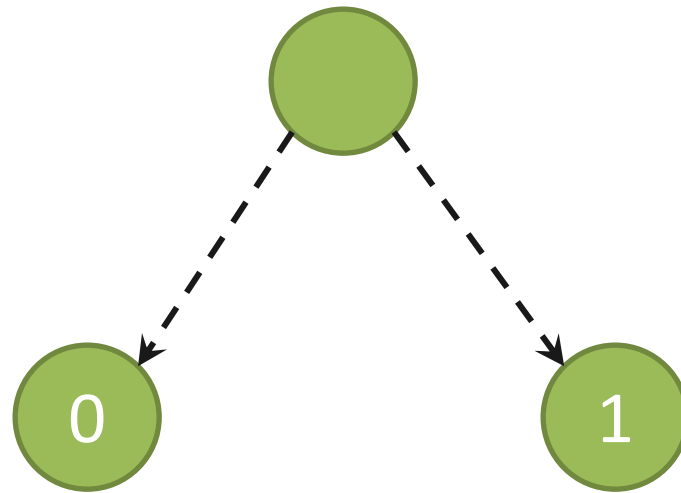


# DRAM Cell Array: Mat



# Memory Element (Cell)

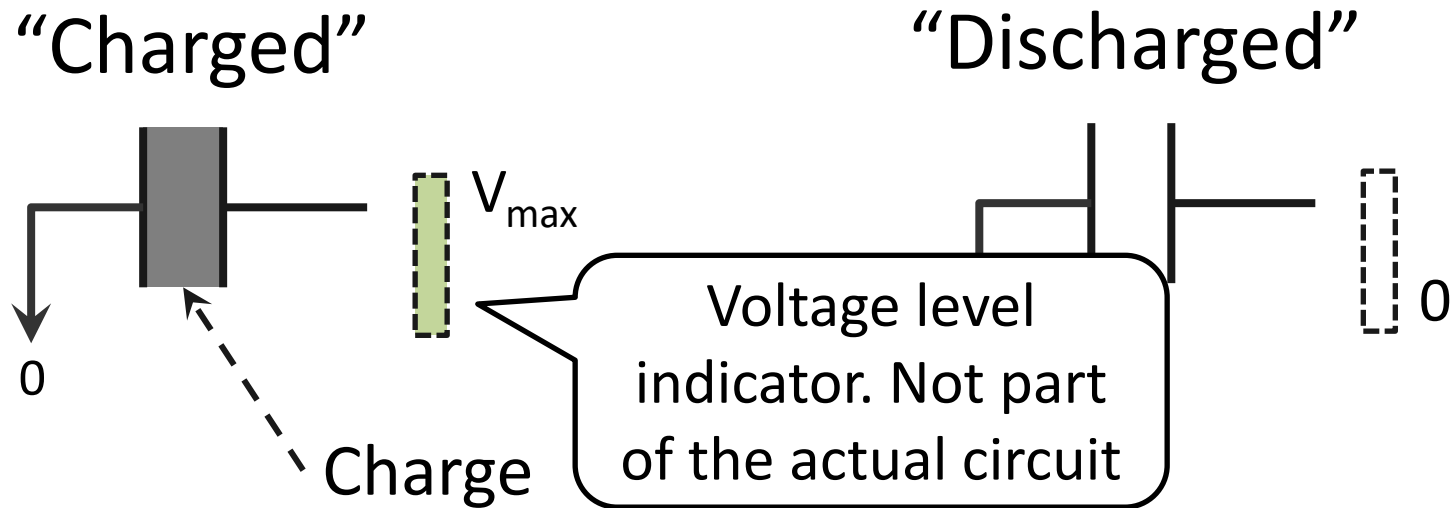
Component that can be in at least two states



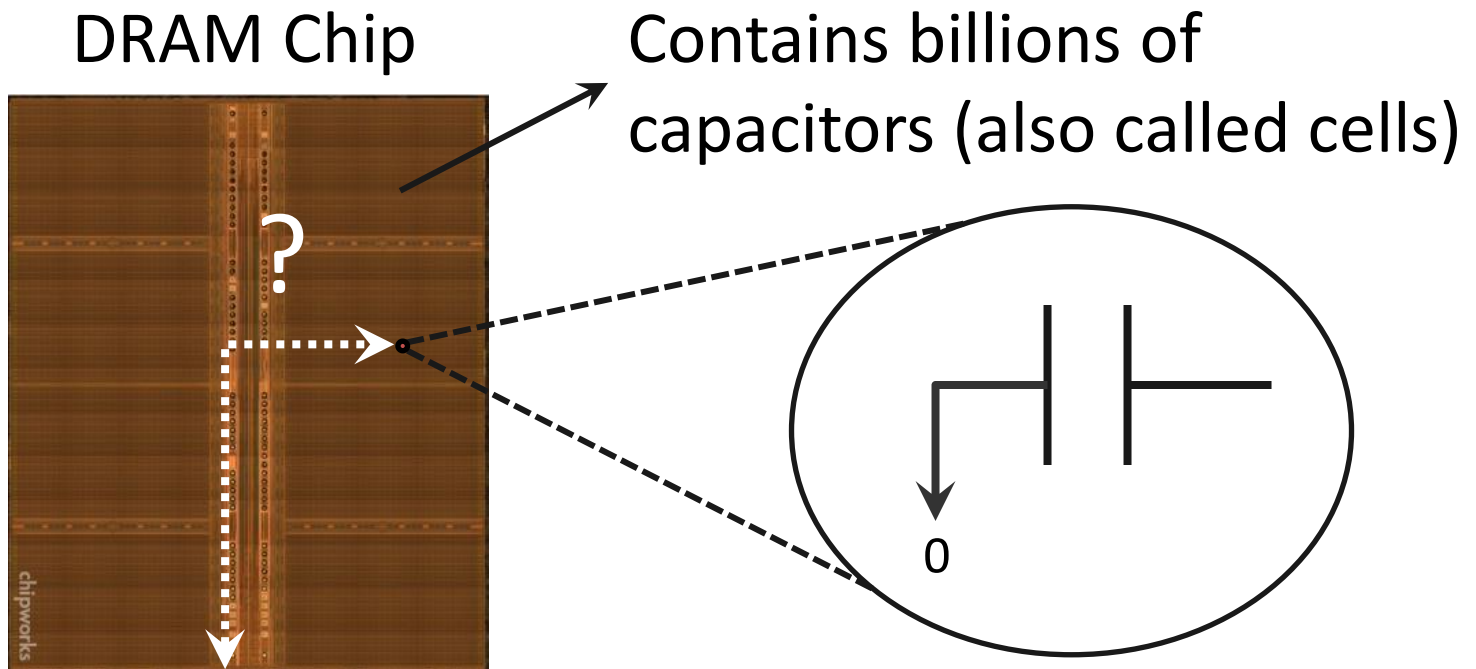
Can be electrical, magnetic, mechanical, etc.

DRAM → Capacitor

# Capacitor – Bucket of Electric Charge

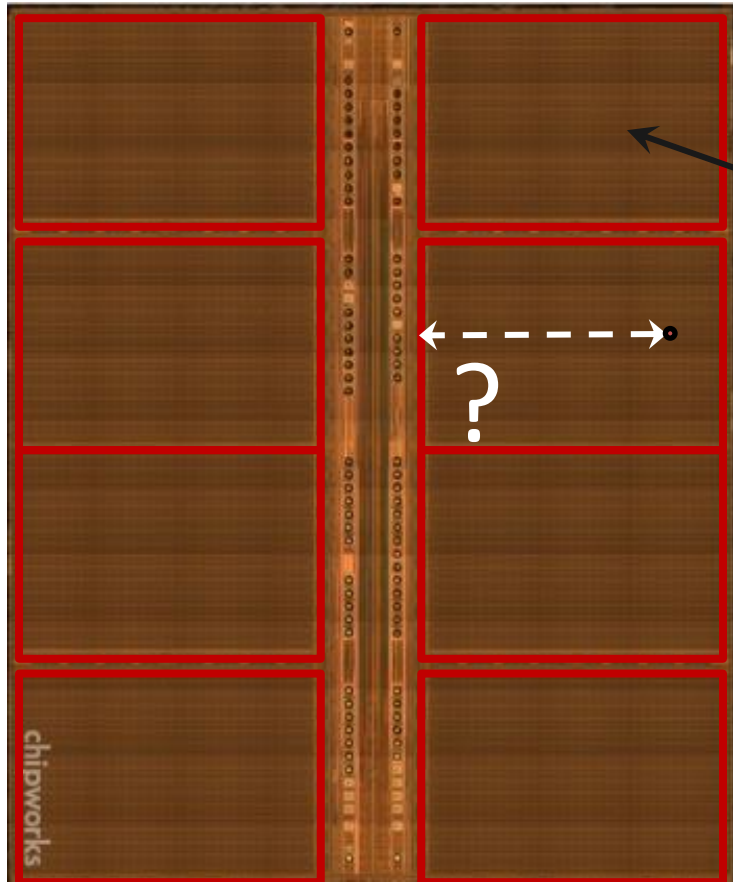


# DRAM Chip





# Divide and Conquer

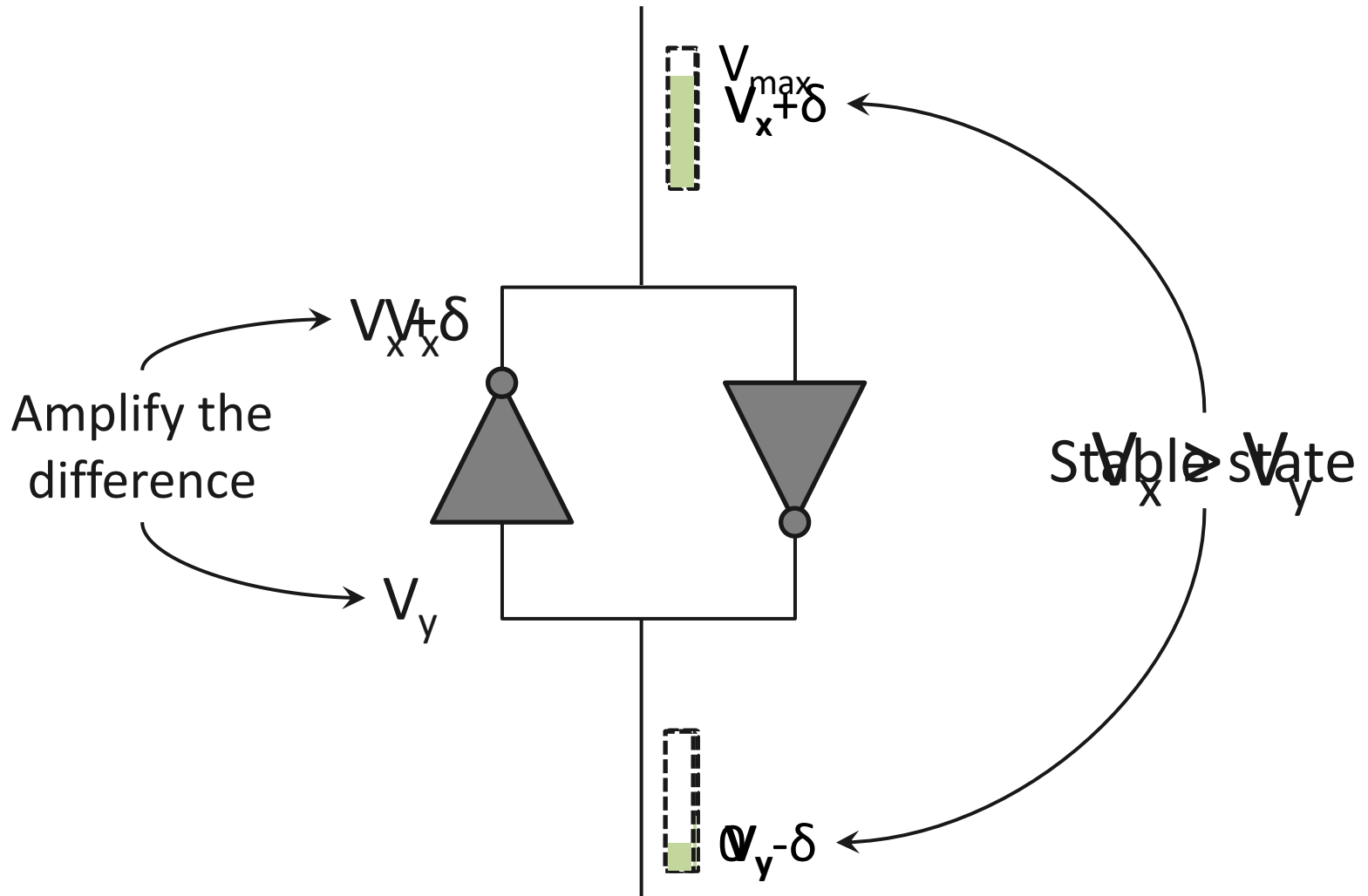


DRAM Bank

## Challenges

1. Weak
2. Reading destroys state

# Sense-Amplifier



# Outline

1. What is DRAM?

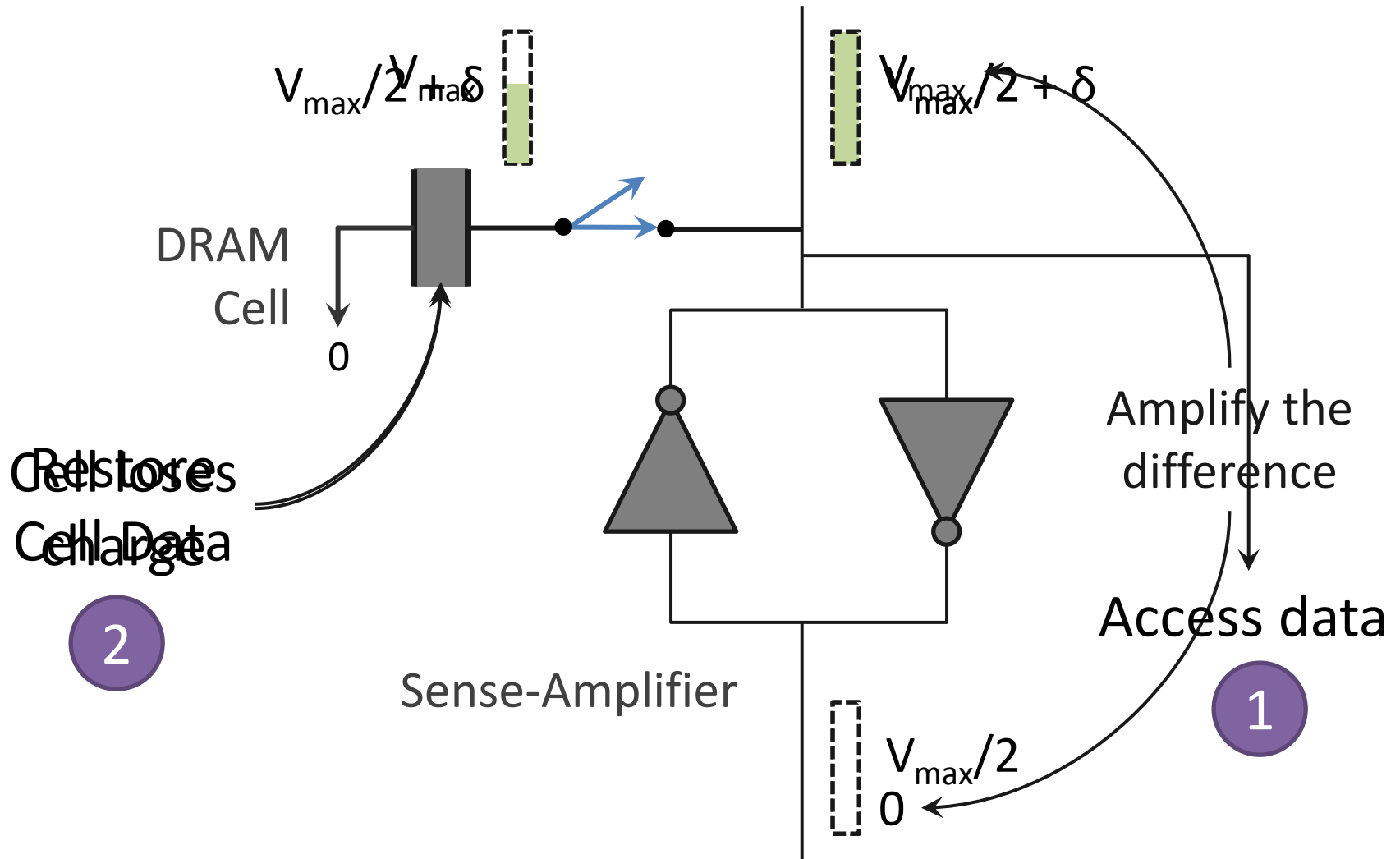
## 2. DRAM Internal Organization

- DRAM Cell
- DRAM Array
- DRAM Bank

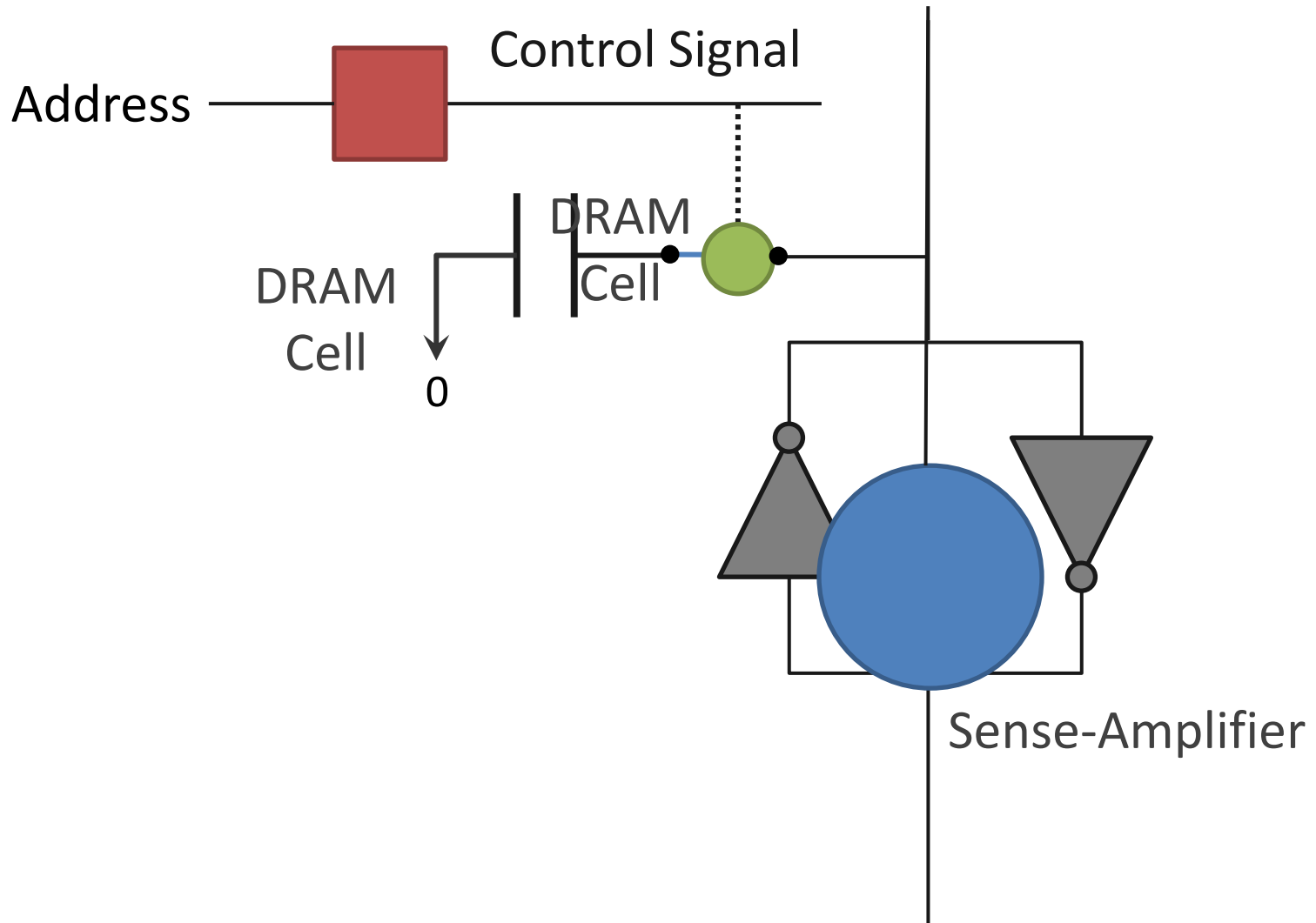
3. Problems and Solutions

- Latency (Tiered-Latency DRAM, HPCA 2013)
- Parallelism (Subarray-level Parallelism, ISCA 2012)

# DRAM Cell Read Operation



# DRAM Cell Read Operation



# Outline

## 1. What is DRAM?

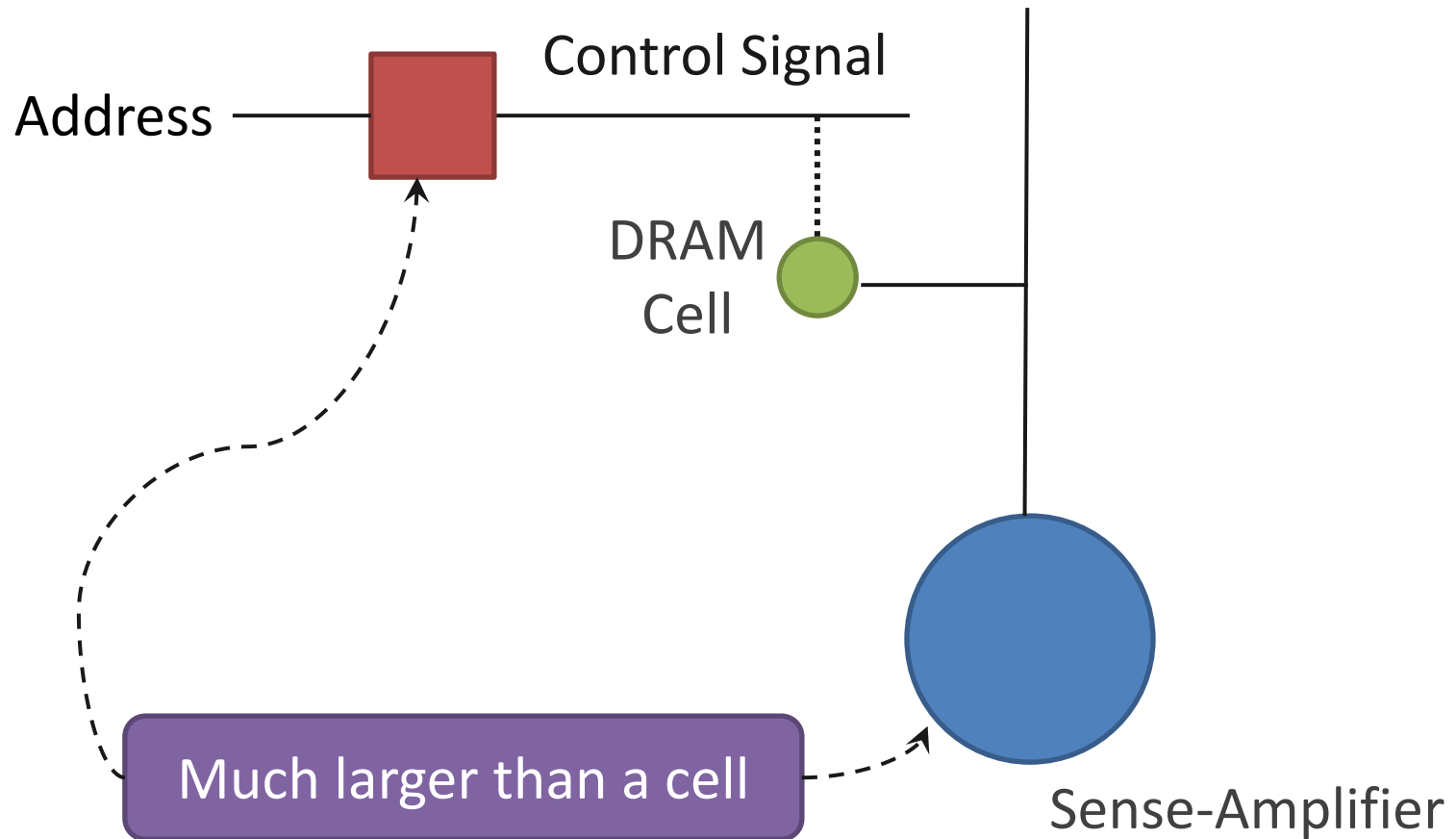
## 2. DRAM Internal Organization

- DRAM Cell
- DRAM Array
- DRAM Bank

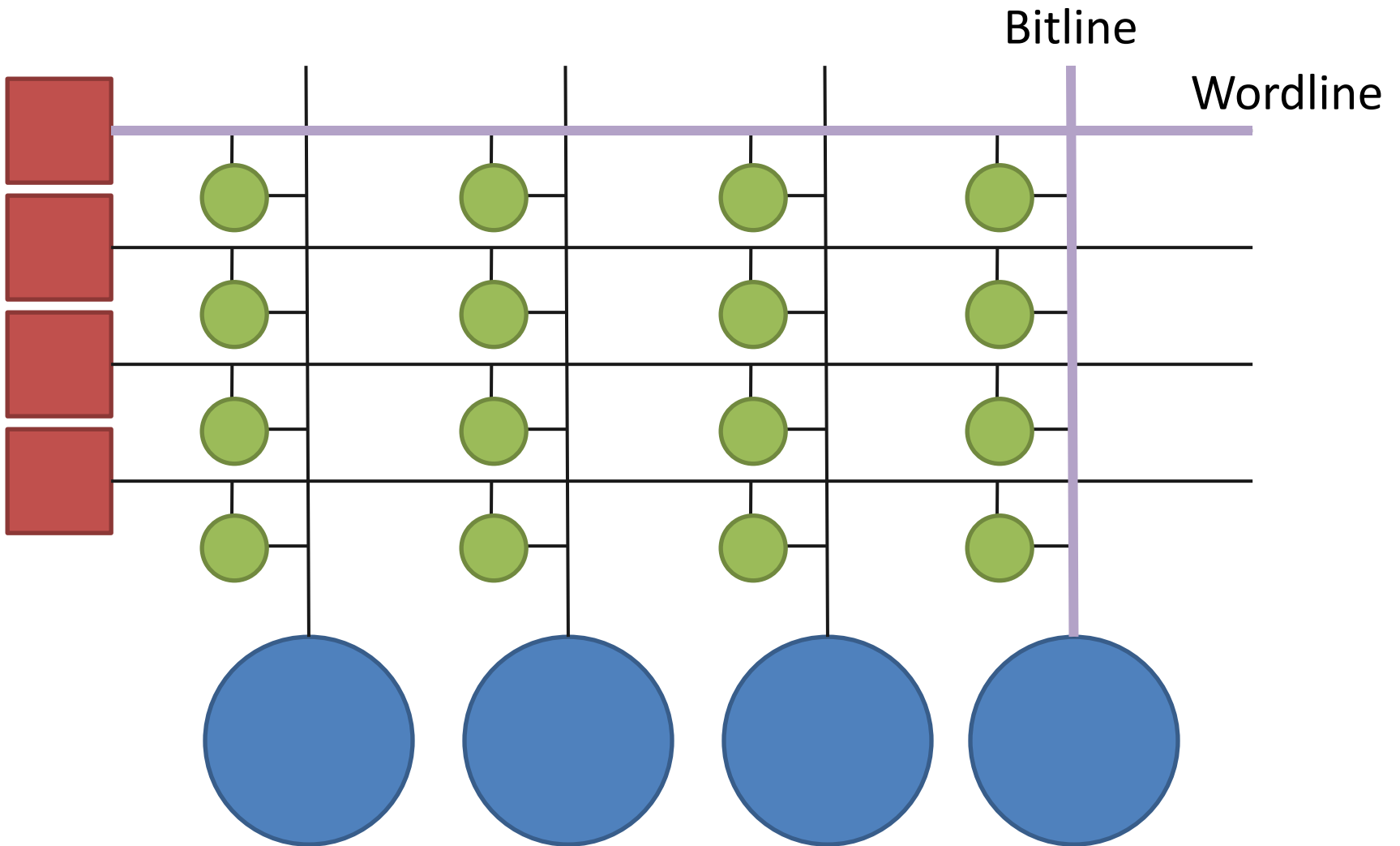
## 3. Problems and Solutions

- Latency (Tiered-Latency DRAM, HPCA 2013; Adaptive-Latency DRAM, HPCA 2015)
- Parallelism (Subarray-level Parallelism, ISCA 2012)

# Problem

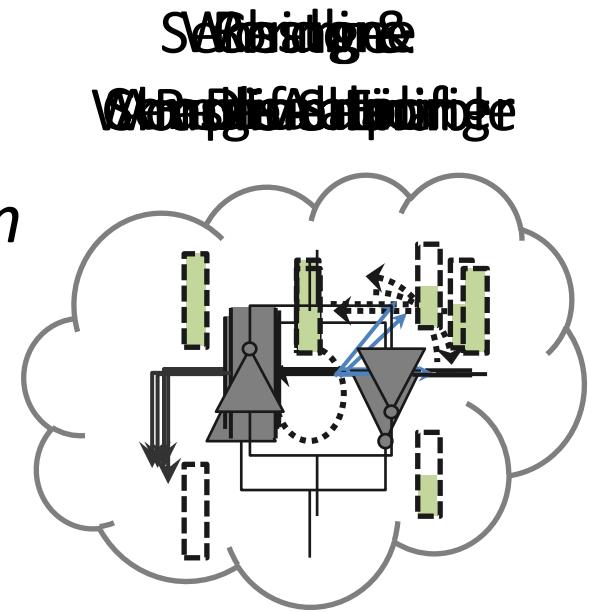
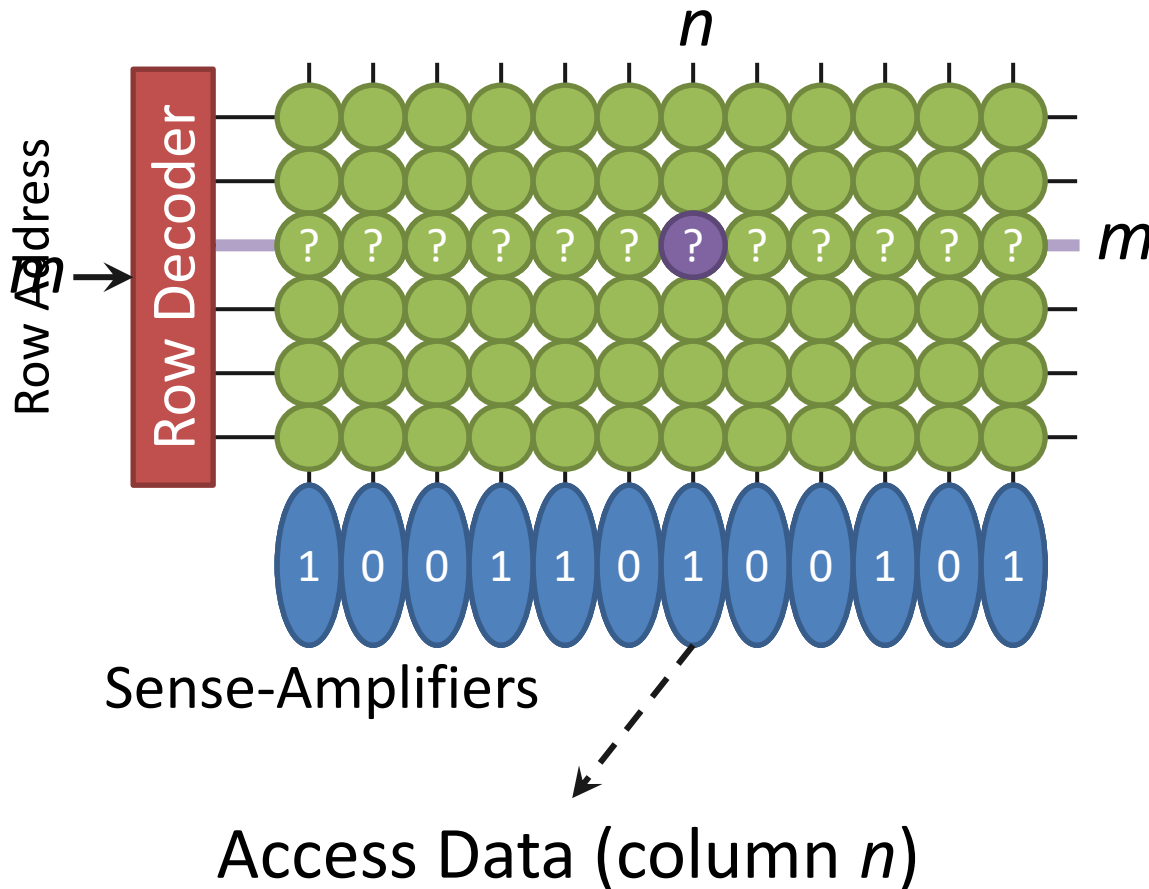


# Cost Amortization





# DRAM Array Operation



# Outline

## 1. What is DRAM?

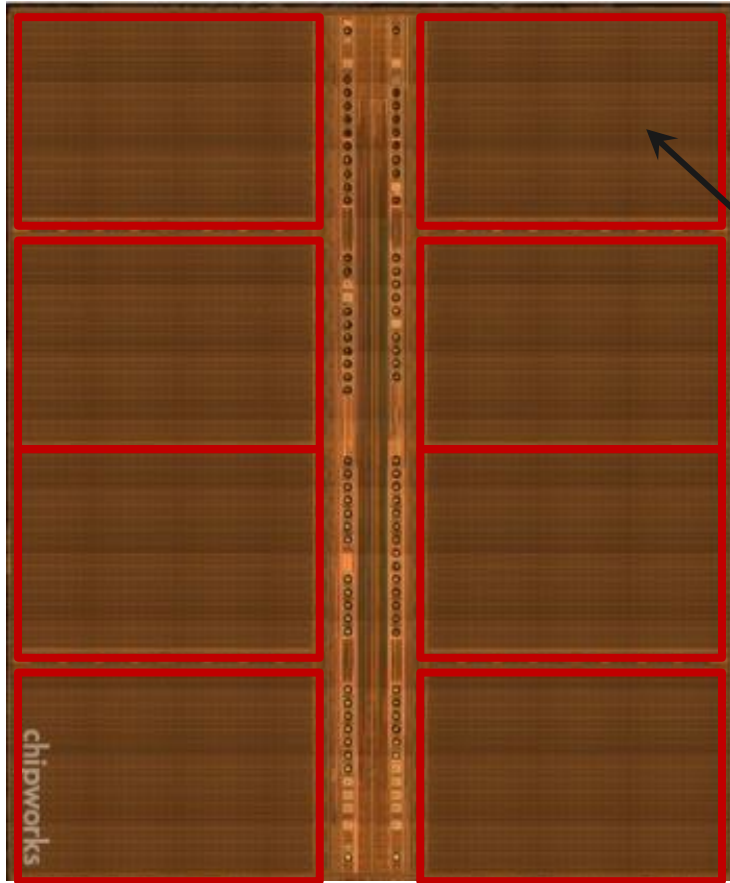
## 2. DRAM Internal Organization

- DRAM Cell
- DRAM Array
- DRAM Bank

## 3. Problems and Solutions

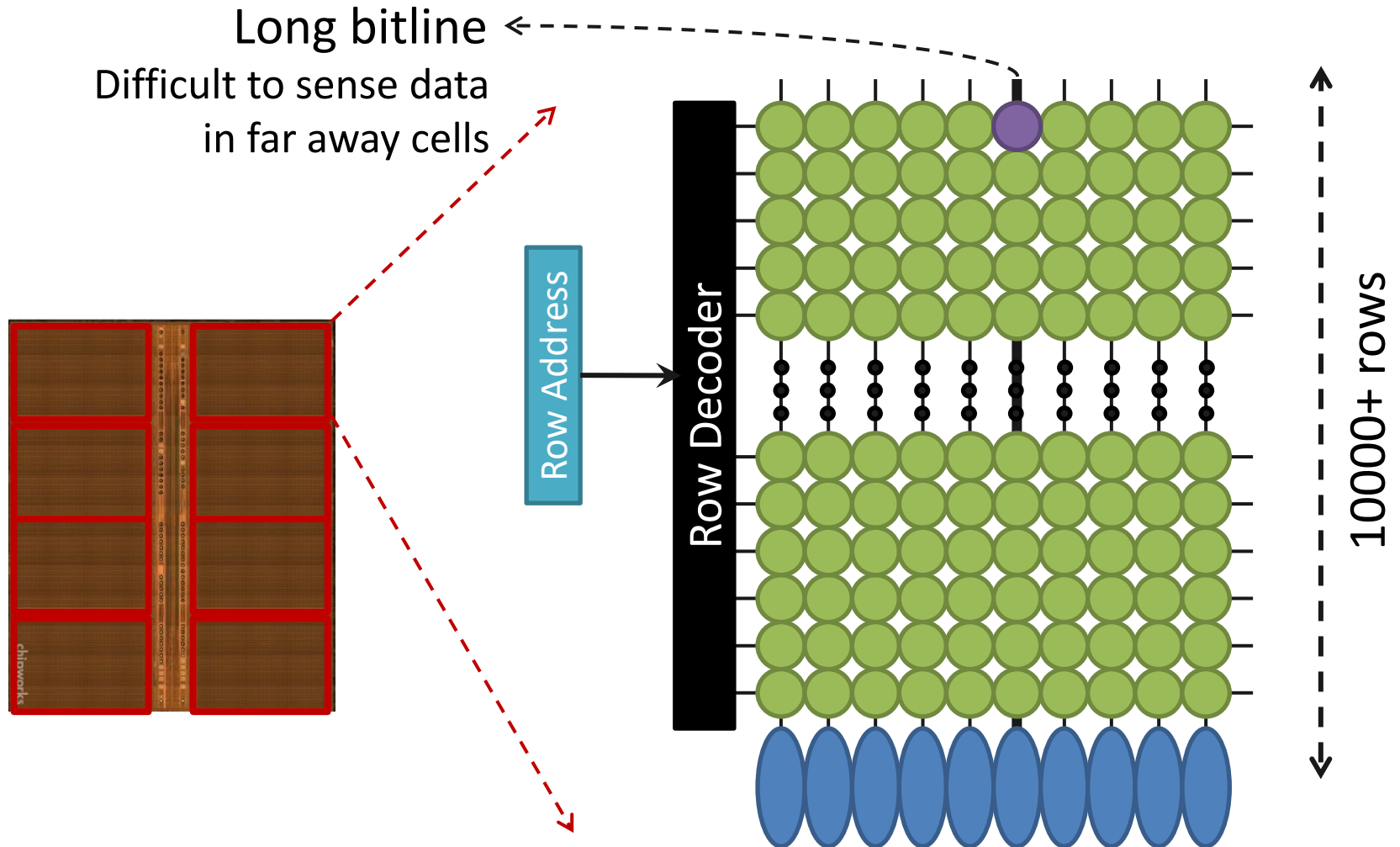
- Latency (Tiered-Latency DRAM, HPCA 2013  
Adaptive-Latency DRAM, HPCA 2015)
- Parallelism (Subarray-level Parallelism, ISCA 2012)

# DRAM Bank

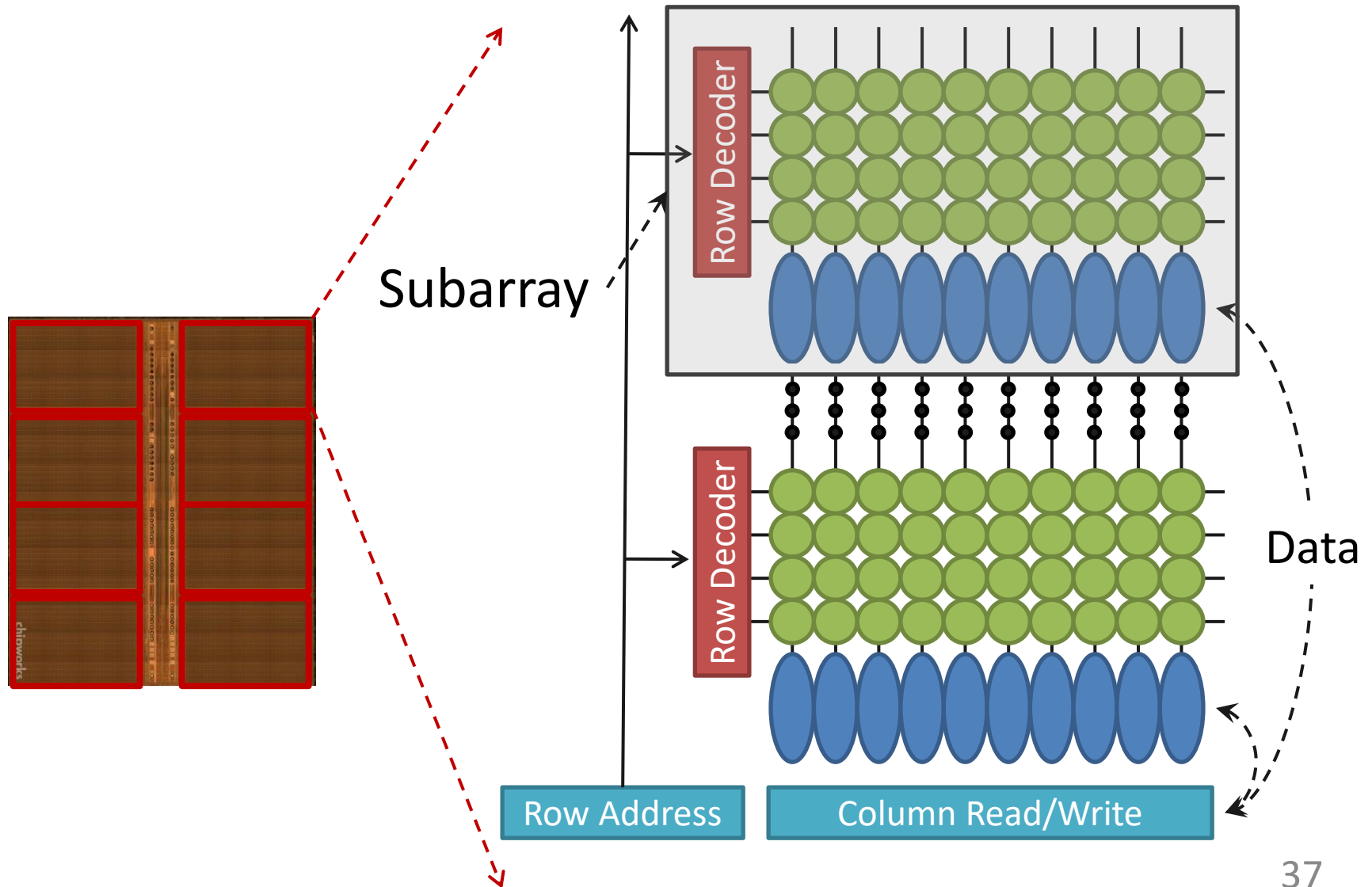


How to build a DRAM bank from a DRAM array?

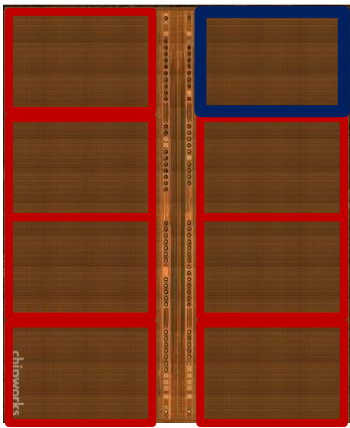
# DRAM Bank: Single DRAM Array?



# DRAM Bank: Collection of Arrays

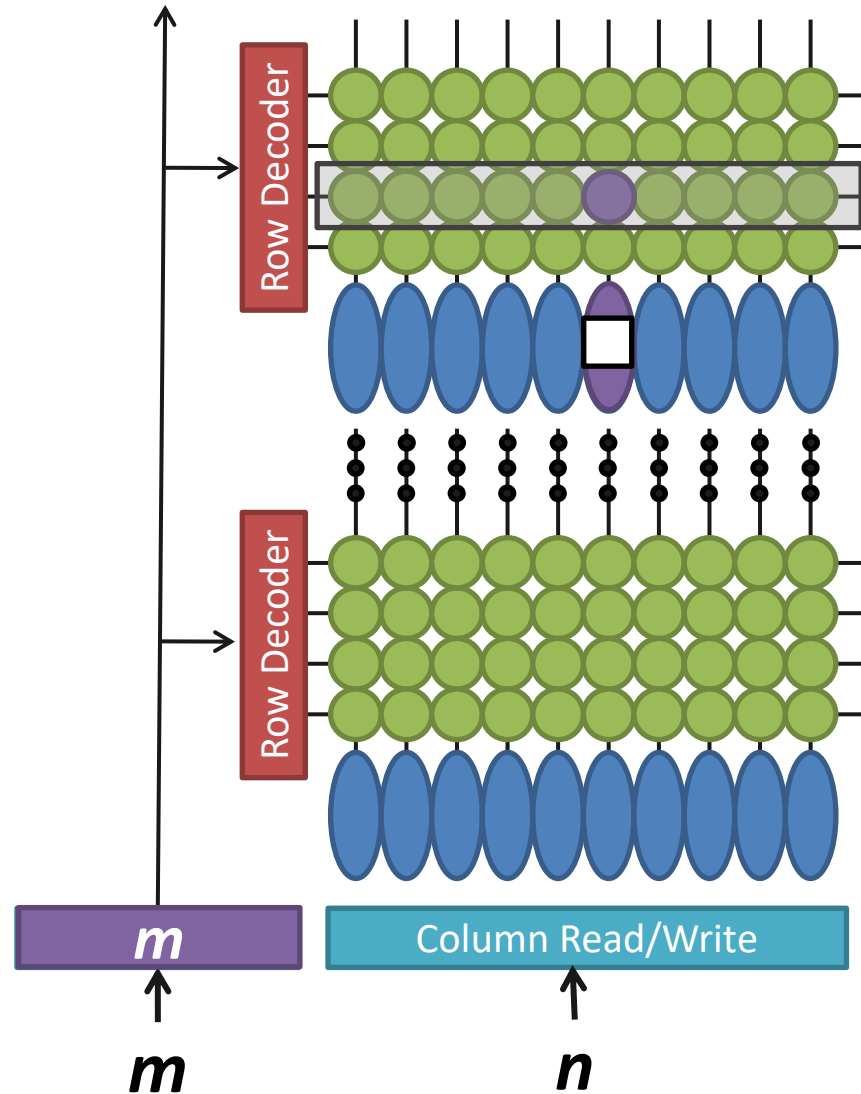


# DRAM Operation: Summary



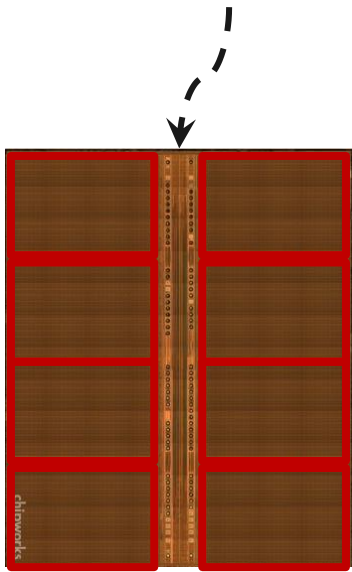
Row  $m$ , Col  $n$

1. Enable row  $m$
2. Access col  $n$
3. Close row

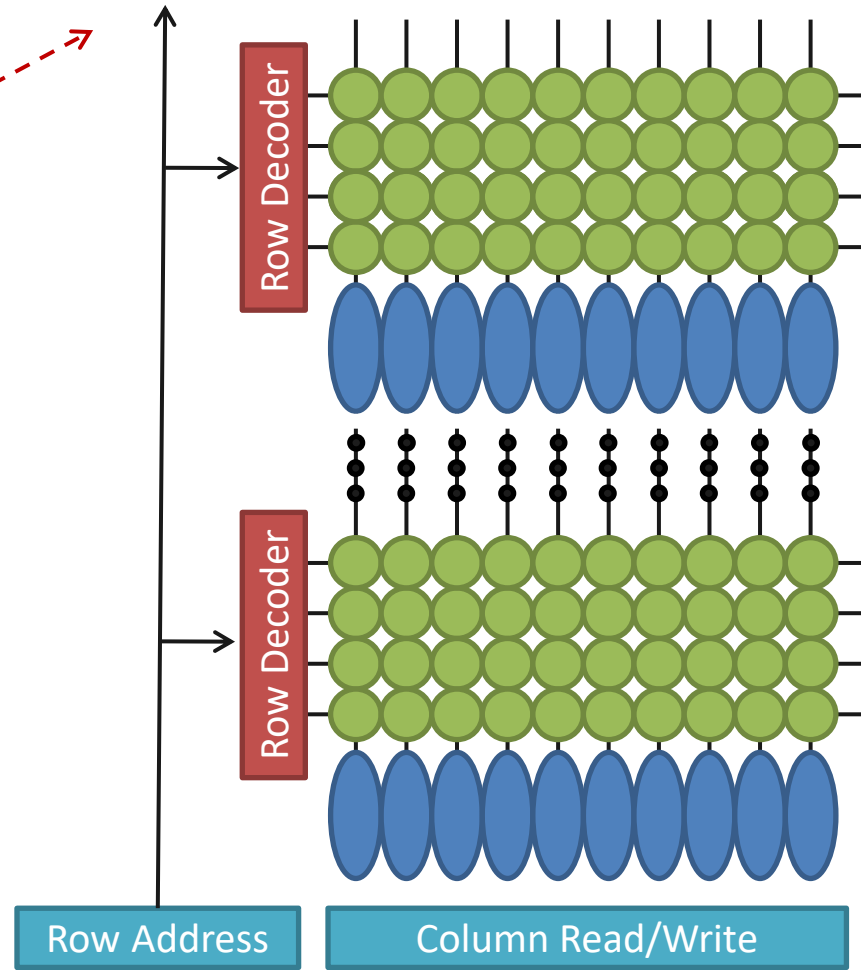


# DRAM Chip Hierarchy

Collection of Banks



Collection of Subarrays



# Outline

1. What is DRAM?

2. DRAM Internal Organization

3. Problems and Solutions

- Latency (Tiered-Latency DRAM, HPCA 2013; Adaptive-Latency DRAM, HPCA 2015)
- Parallelism (Subarray-level Parallelism, ISCA 2012)



# Factors That Affect Performance

## 1. Latency

- How fast can DRAM serve a request?

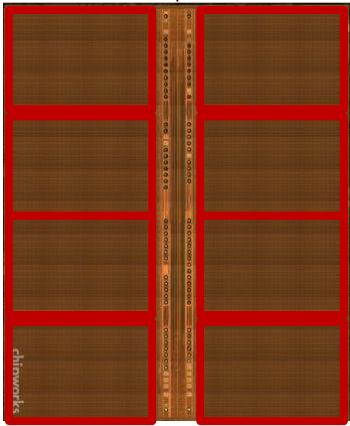
## 2. Parallelism

- How many requests can DRAM serve in parallel?

# DRAM Chip Hierarchy

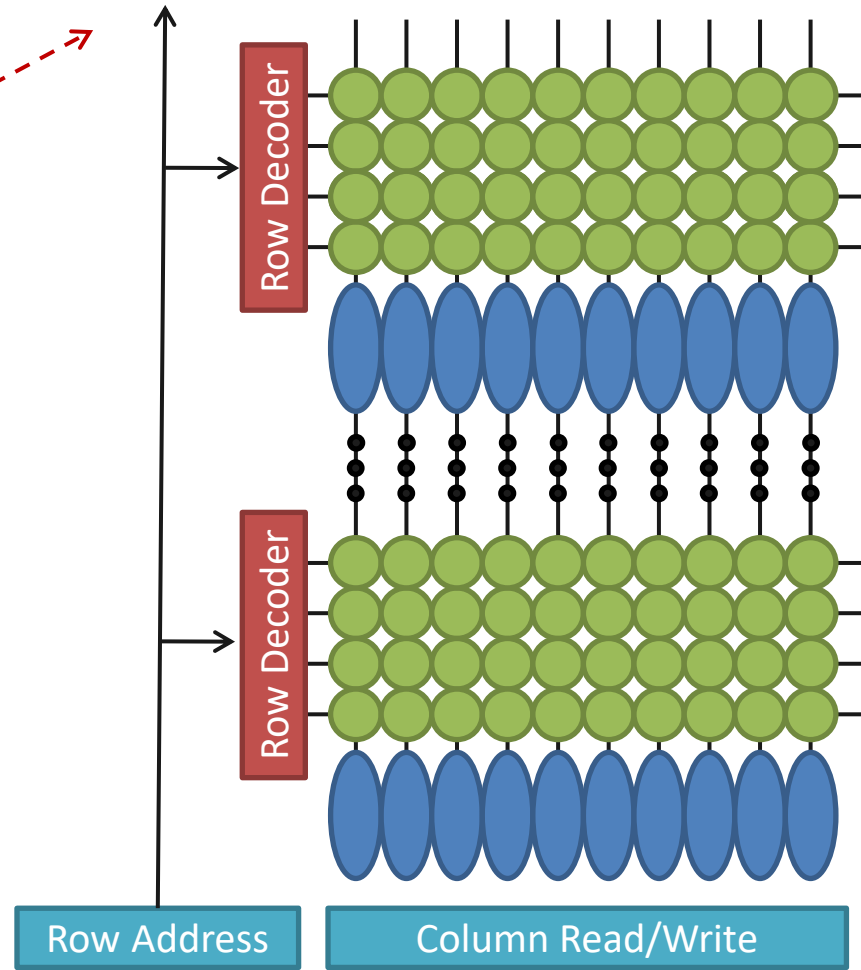
Collection of Banks

Parallelism



Latency

Collection of Subarrays



# Outline

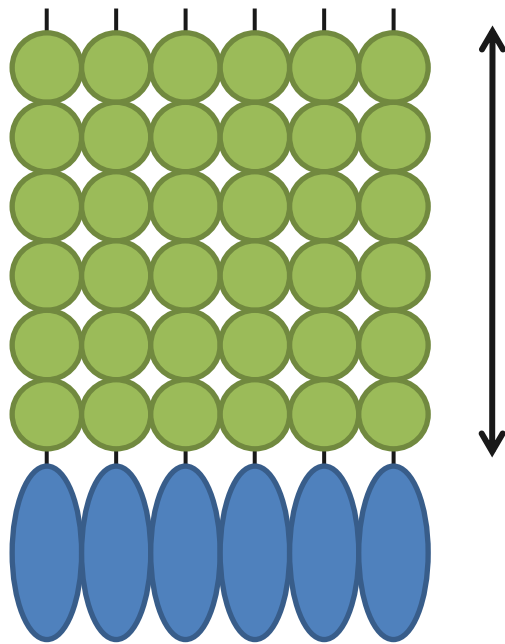
1. What is DRAM?

2. DRAM Internal Organization

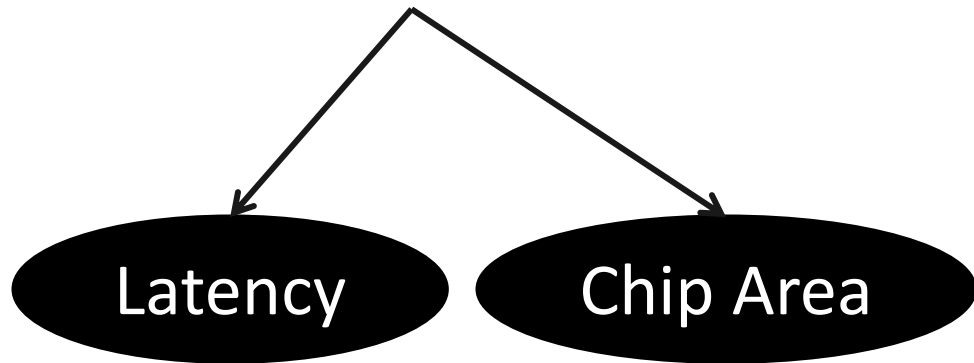
3. Problems and Solutions

- Latency (Tiered-Latency DRAM, HPCA 2013; Adaptive-Latency DRAM, HPCA 2015)
- Parallelism (Subarray-level Parallelism, ISCA 2012)

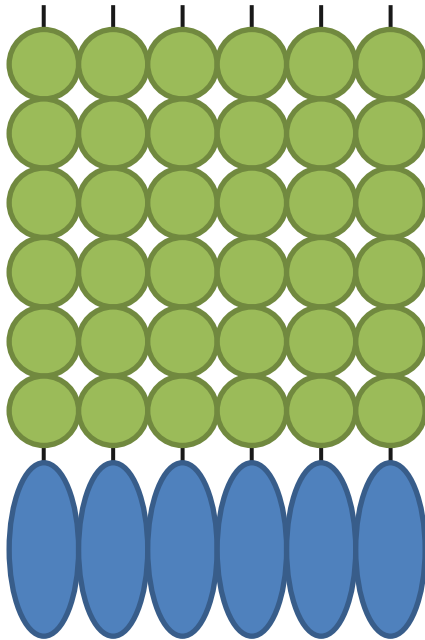
# Subarray Size: Rows/Subarray



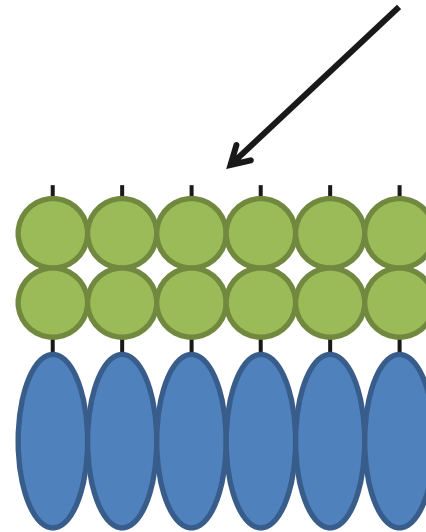
Number of rows in subarray



# Subarray Size vs. Access Latency



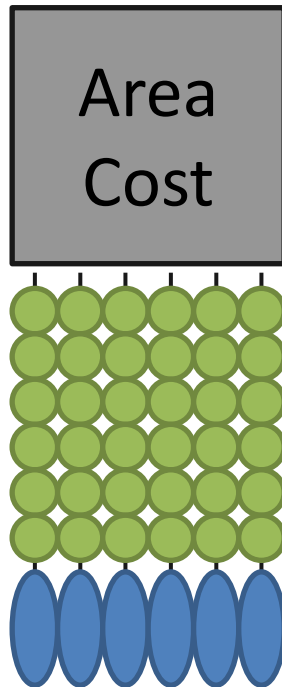
Shorter Bitlines => Faster access



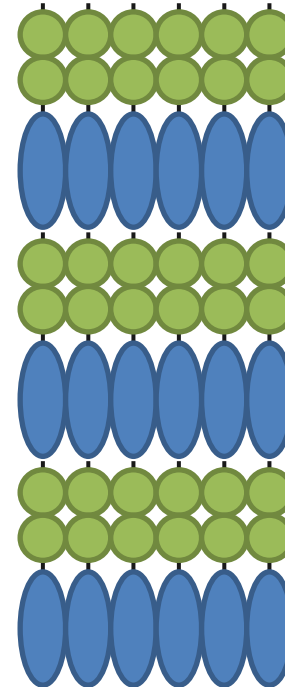
Smaller subarrays => lower access latency

# Subarray Size vs. Chip Area

Large Subarray

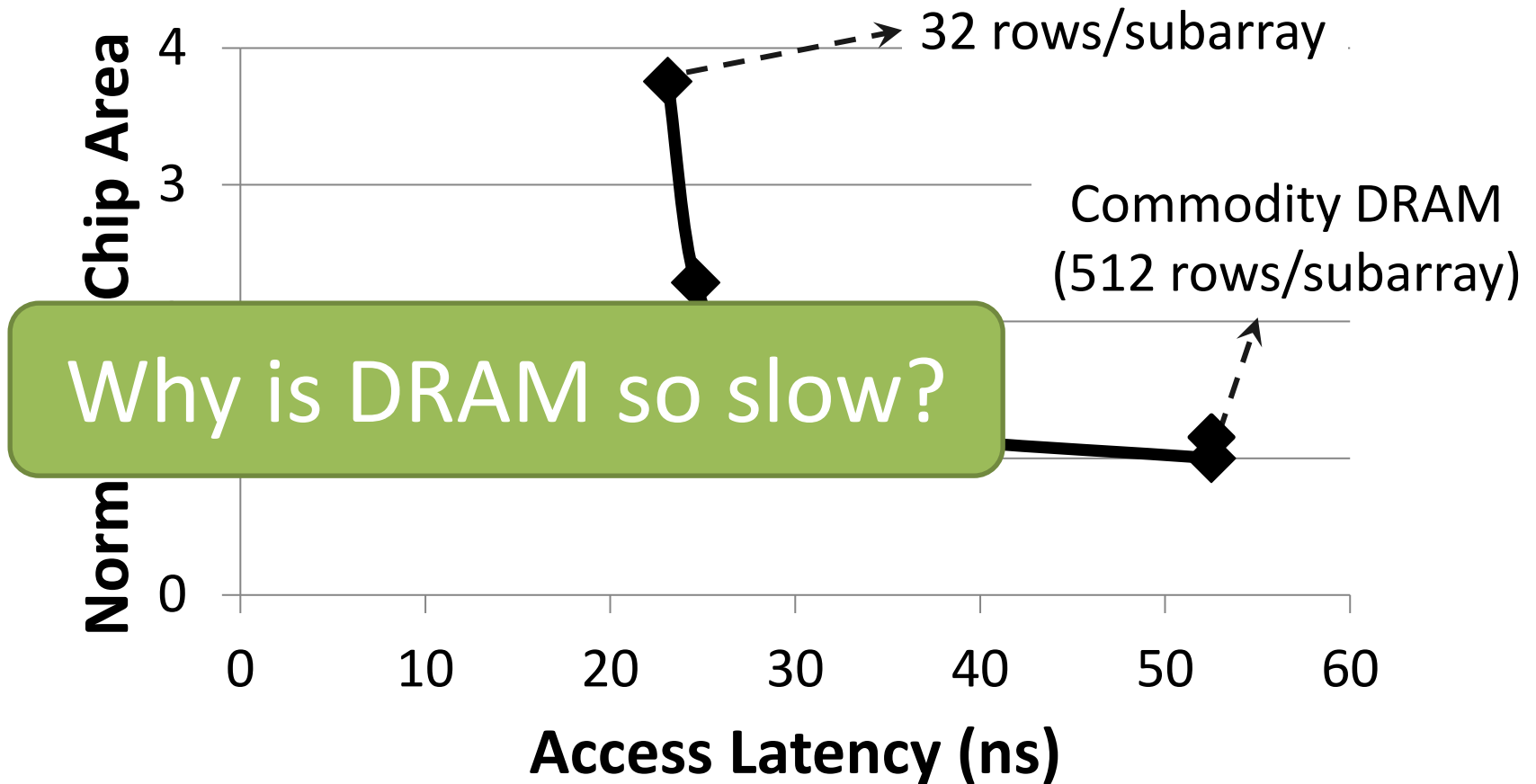


Smaller Subarrays

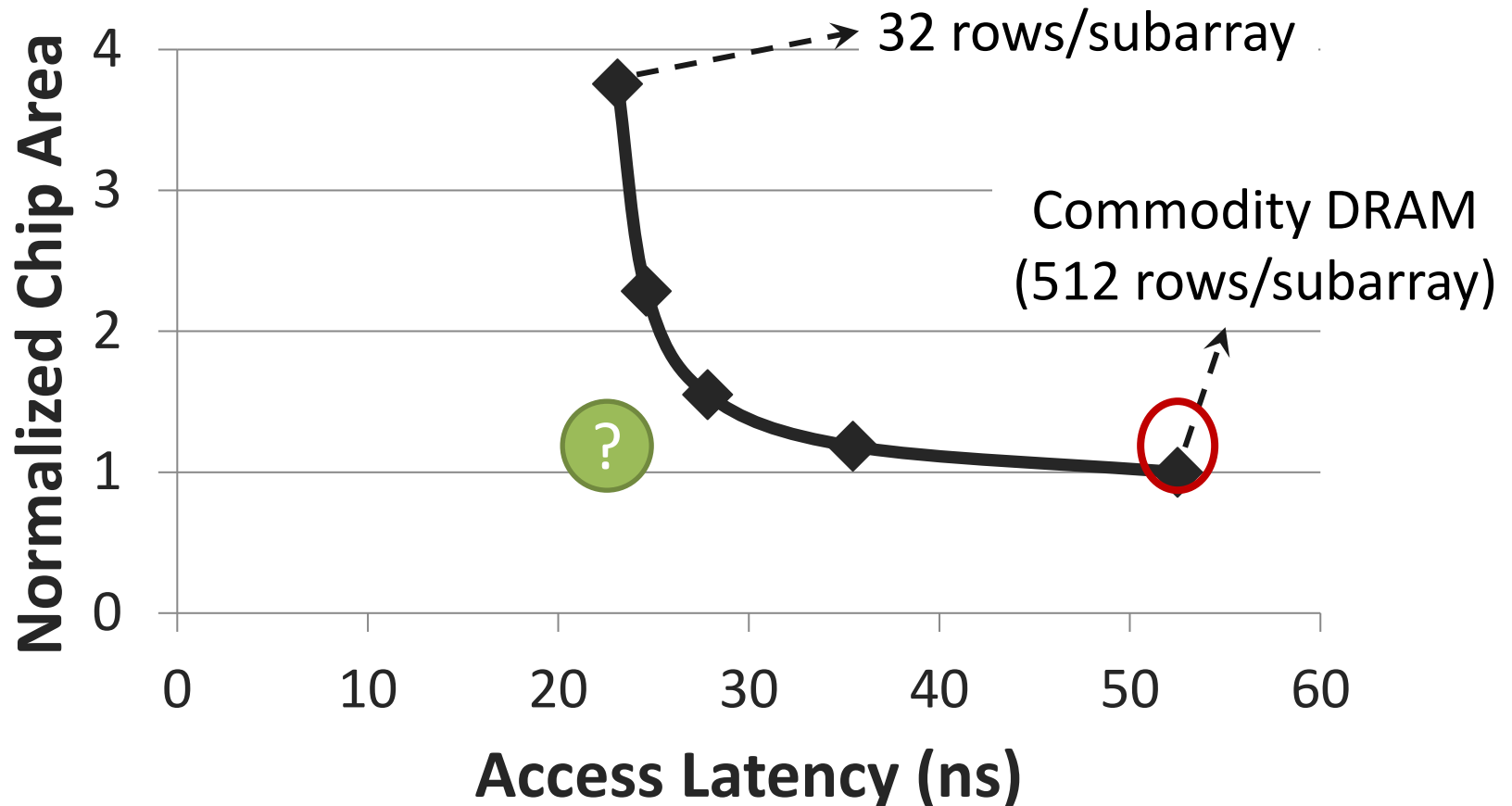


Smaller subarrays => larger chip area

# Chip Area vs. Access Latency



# Chip Area vs. Access Latency

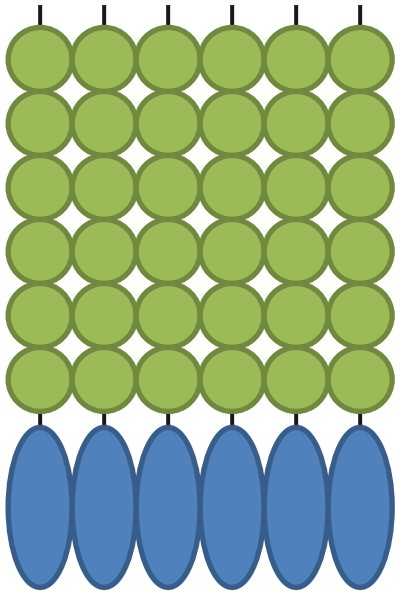


How to enable low latency without high area overhead?

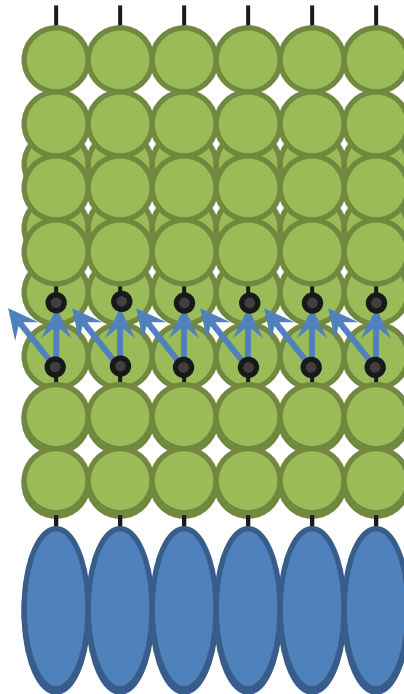


# New Proposal

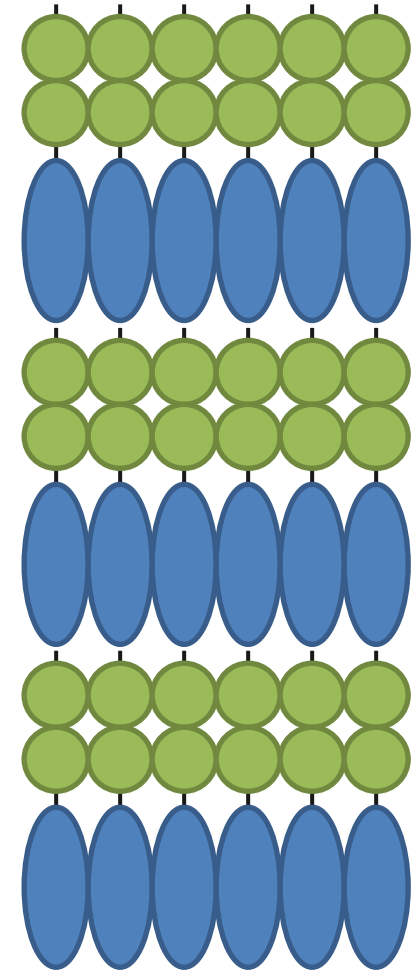
Low area cost



Large Subarray

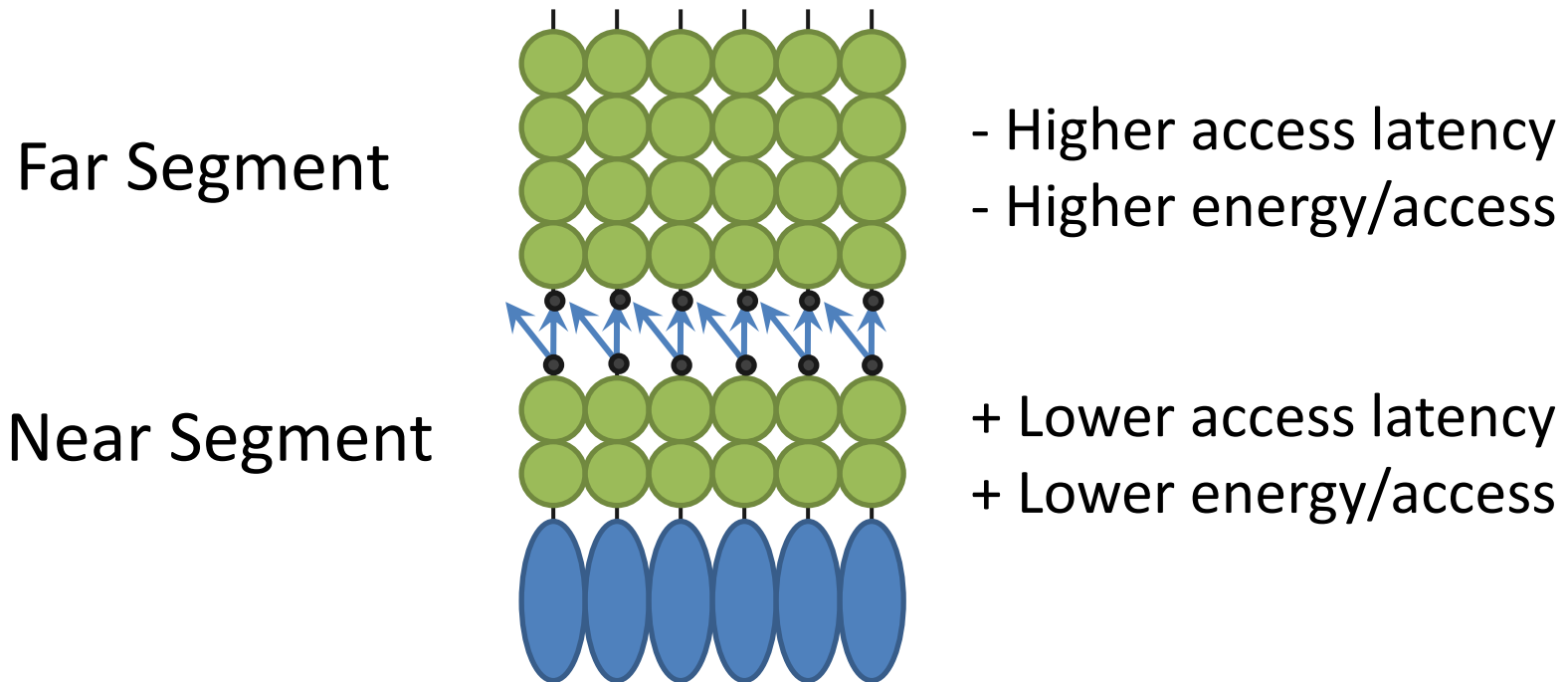


**Our Proposal**



Small Subarray

# Tiered-Latency DRAM

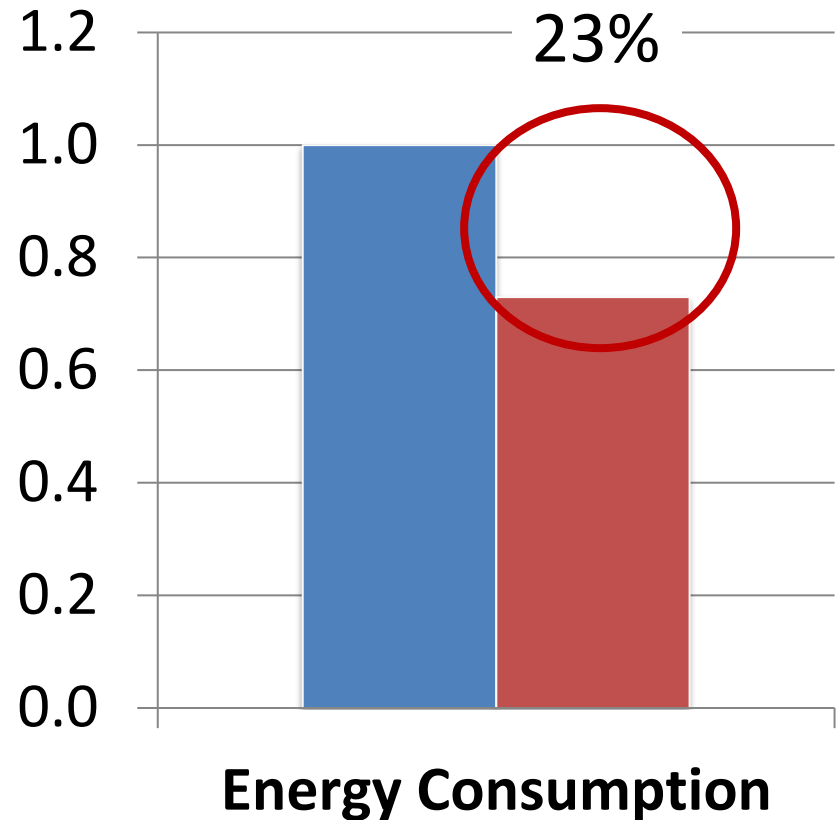
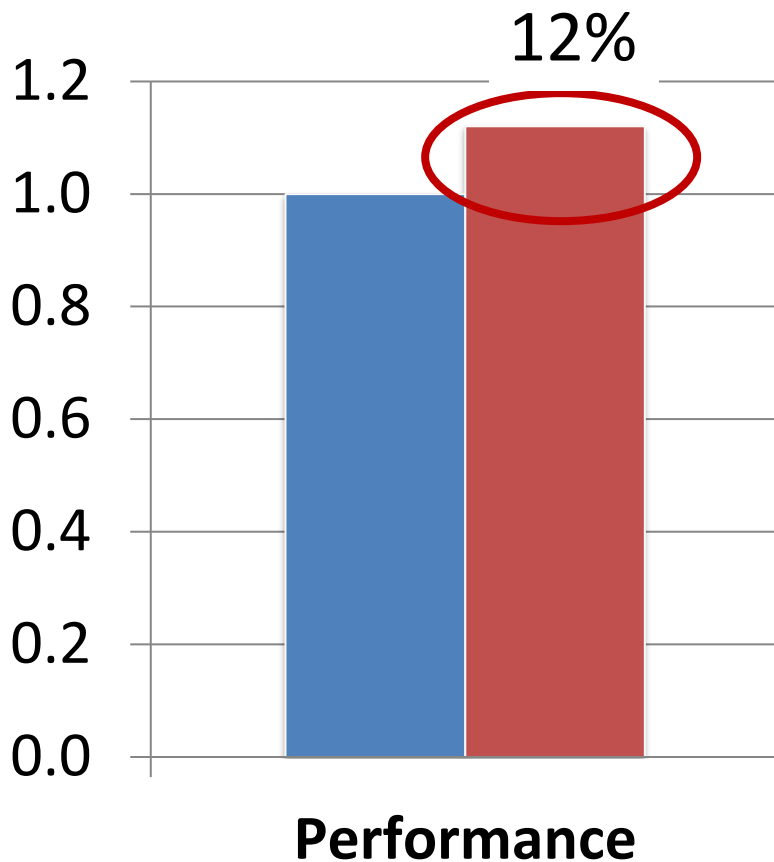


Map frequently accessed data to near segment

# Results Summary

■ Commodity DRAM

■ Tiered-Latency DRAM



# Tiered-Latency DRAM: A Low Latency and Low Cost DRAM Architecture

Donghyuk Lee, Yoongu Kim, Vivek Seshadri,  
Jamie Liu, Lavanya Subramanian, Onur Mutlu

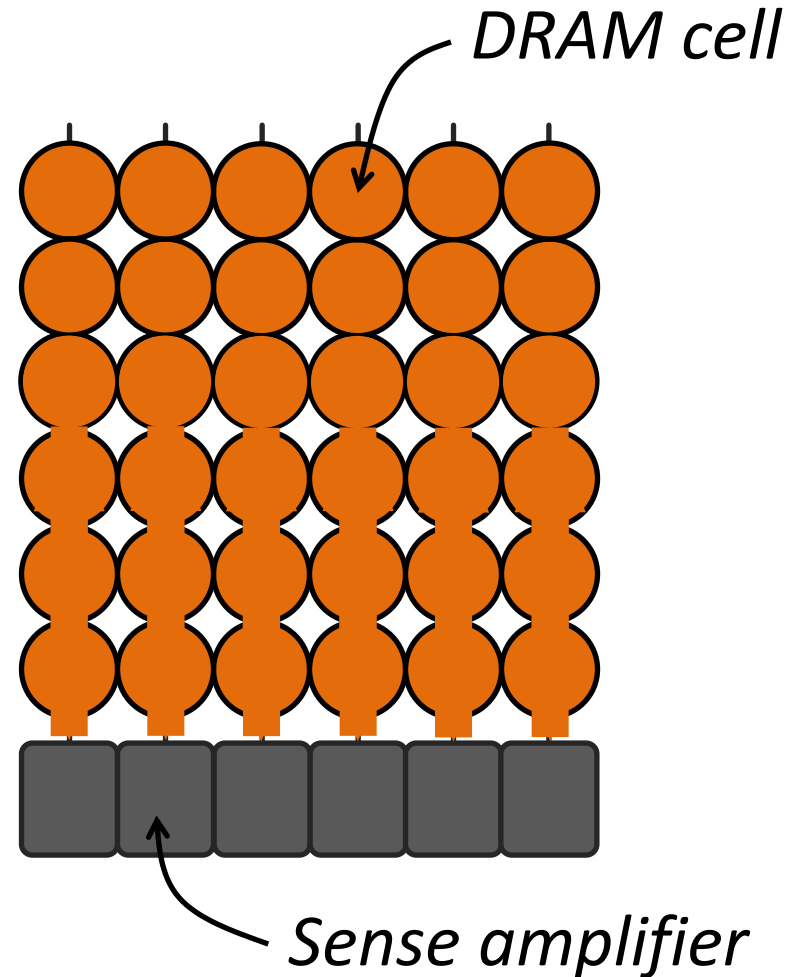
Published in the proceedings of 19<sup>th</sup> IEEE International  
Symposium on

**High Performance Computer Architecture 2013**

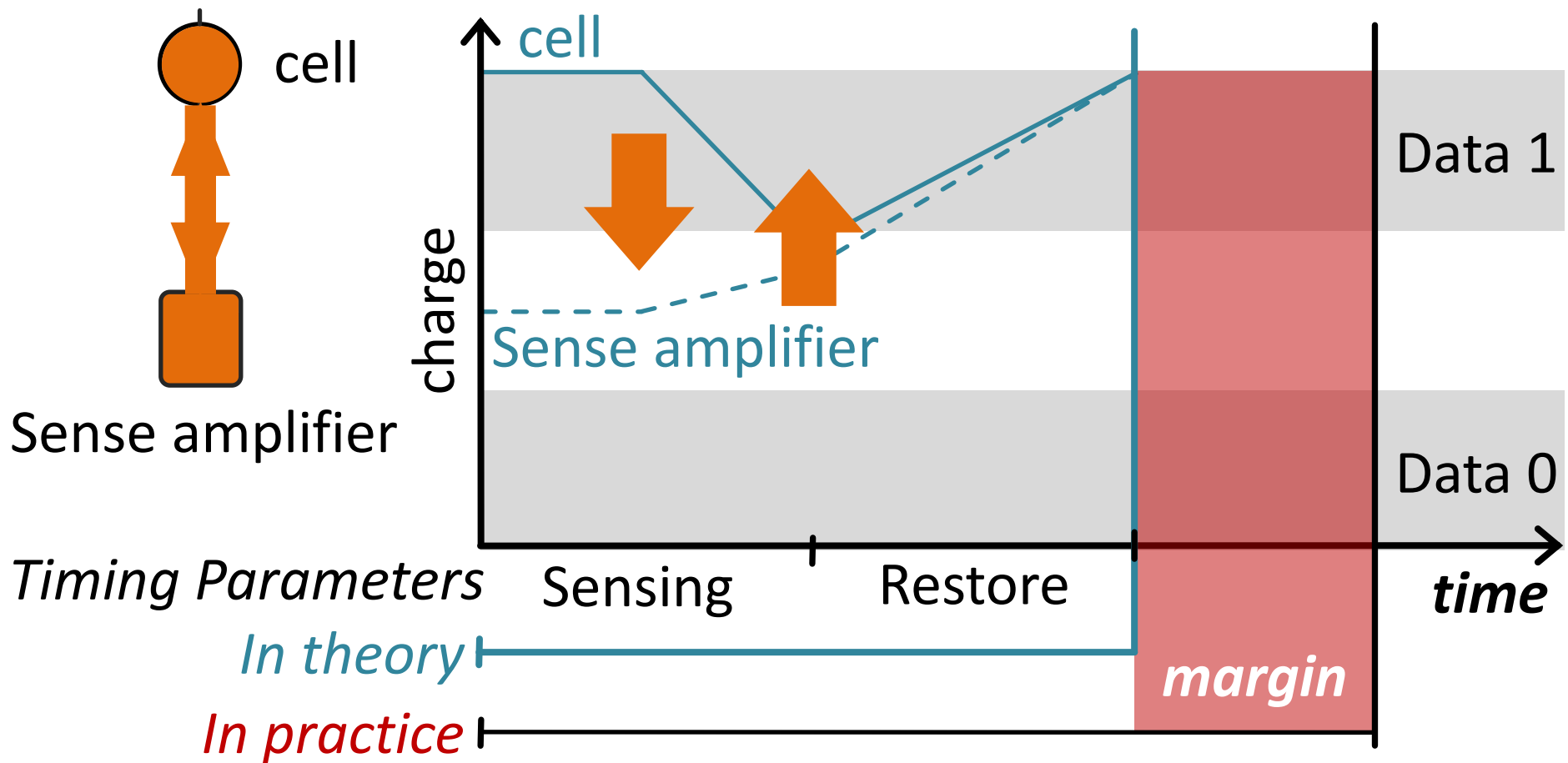
# DRAM Stores Data as Charge

Three steps of charge movement

1. Sensing
2. Restore
3. Precharge



# DRAM Charge over Time



Why does DRAM need the extra timing margin?

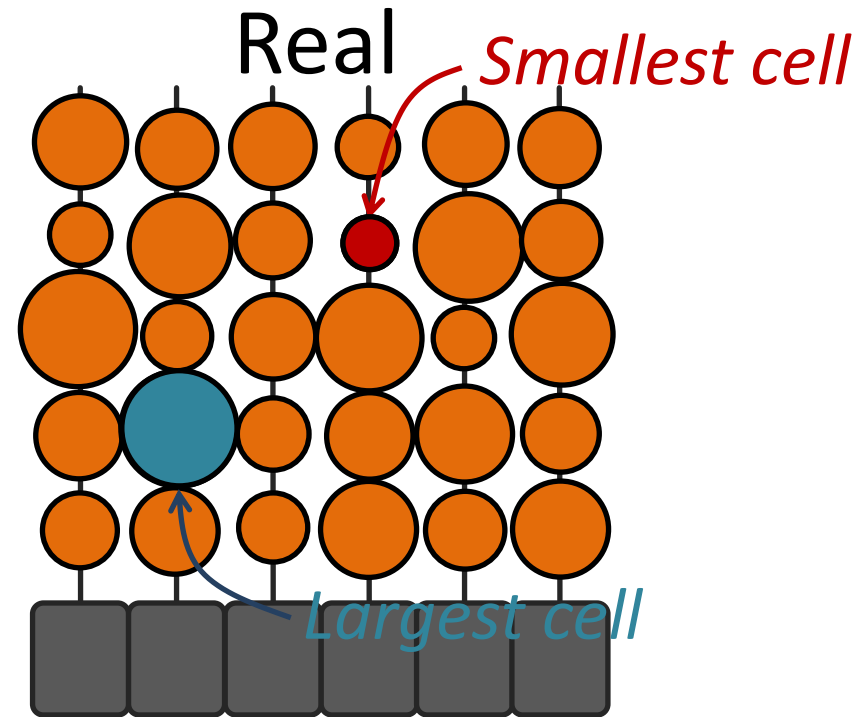
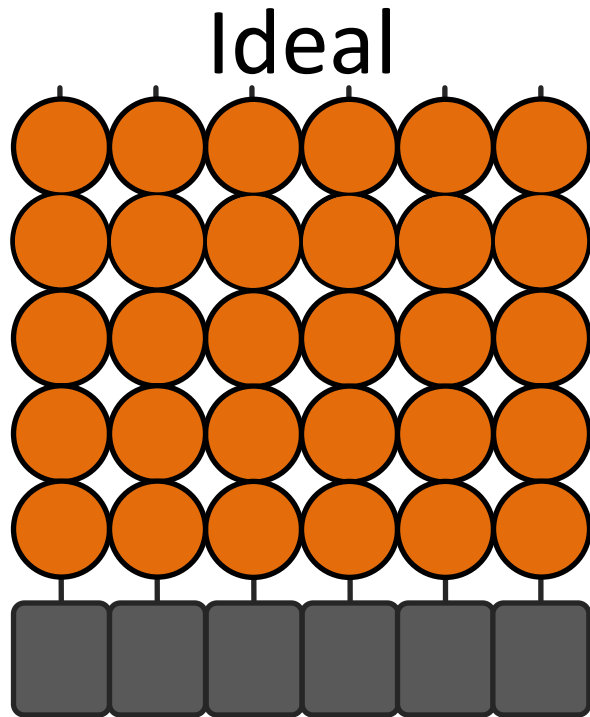
# Two Reasons for Timing Margin

## 1. Process Variation

- DRAM cells are not equal
- Leads to extra timing margin for cells that can store large amount of charge

## 2. Temperature Dependence

# DRAM Cells are Not Equal



Same size →  
Same charge →  
Same latency

Large variation in cell size →  
Large variation in charge →  
Large variation in access latency

Different size →  
Different charge →  
Different latency



# Two Reasons for Timing Margin

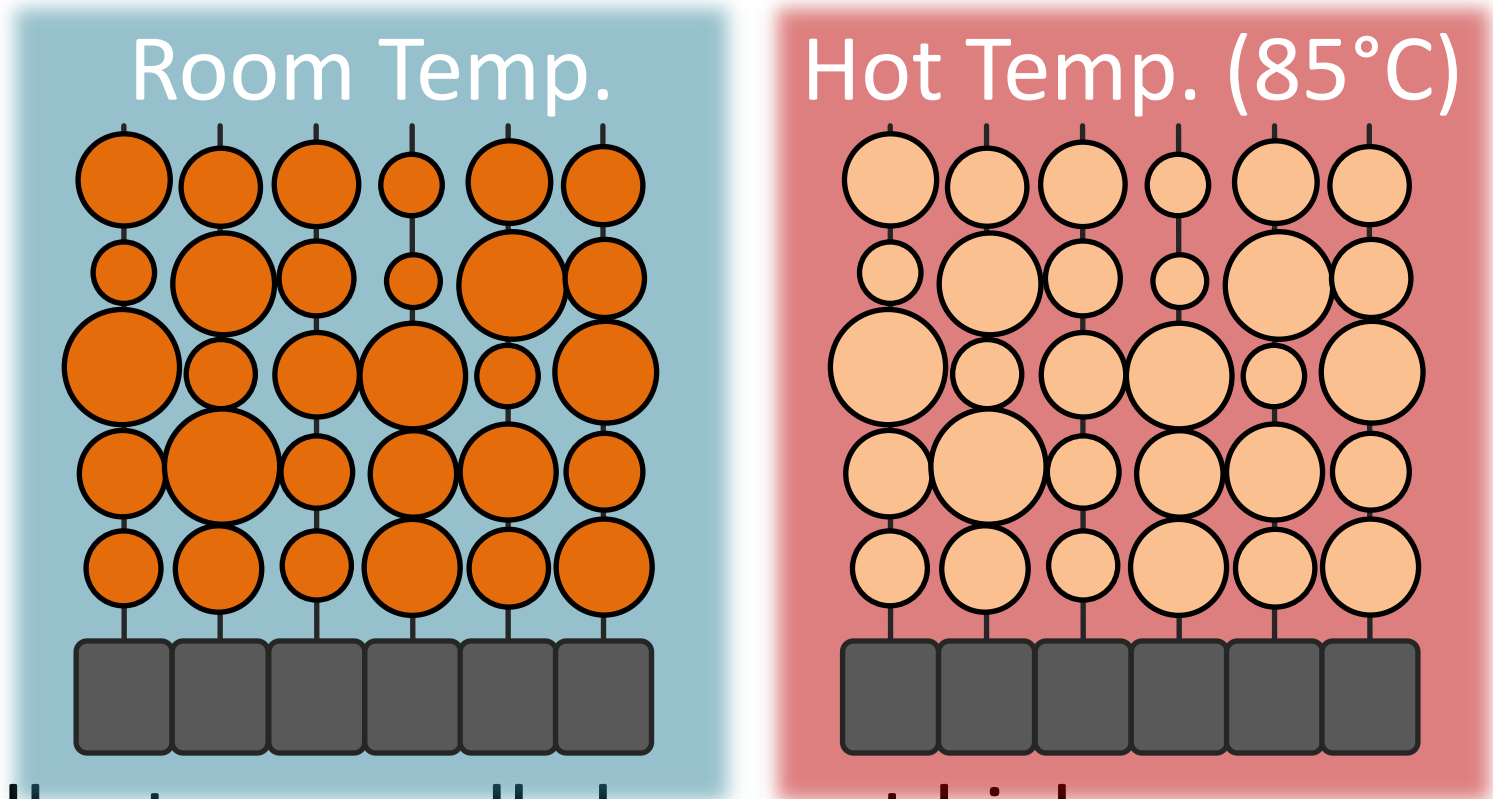
## 1. Process Variation

- DRAM cells are not equal
- Leads to *extra timing margin* for cells that can store large amount of charge

## 2. Temperature Dependence

- DRAM leaks more charge at higher temperature
- Leads to extra timing margin when operating at low temperature

# Charge Leakage $\propto$ Temperature



Cells store small charge at high temperature  
and large charge at low temperature  
→ Large variation in access latency

# DRAM Timing Parameters

- DRAM timing parameters are dictated by *the worst case*
  - The smallest cell with the smallest charge in all DRAM products
  - Operating at the highest temperature
- Large timing margin for the common case
  - Can lower latency for the common case

# DRAM Testing Infrastructure

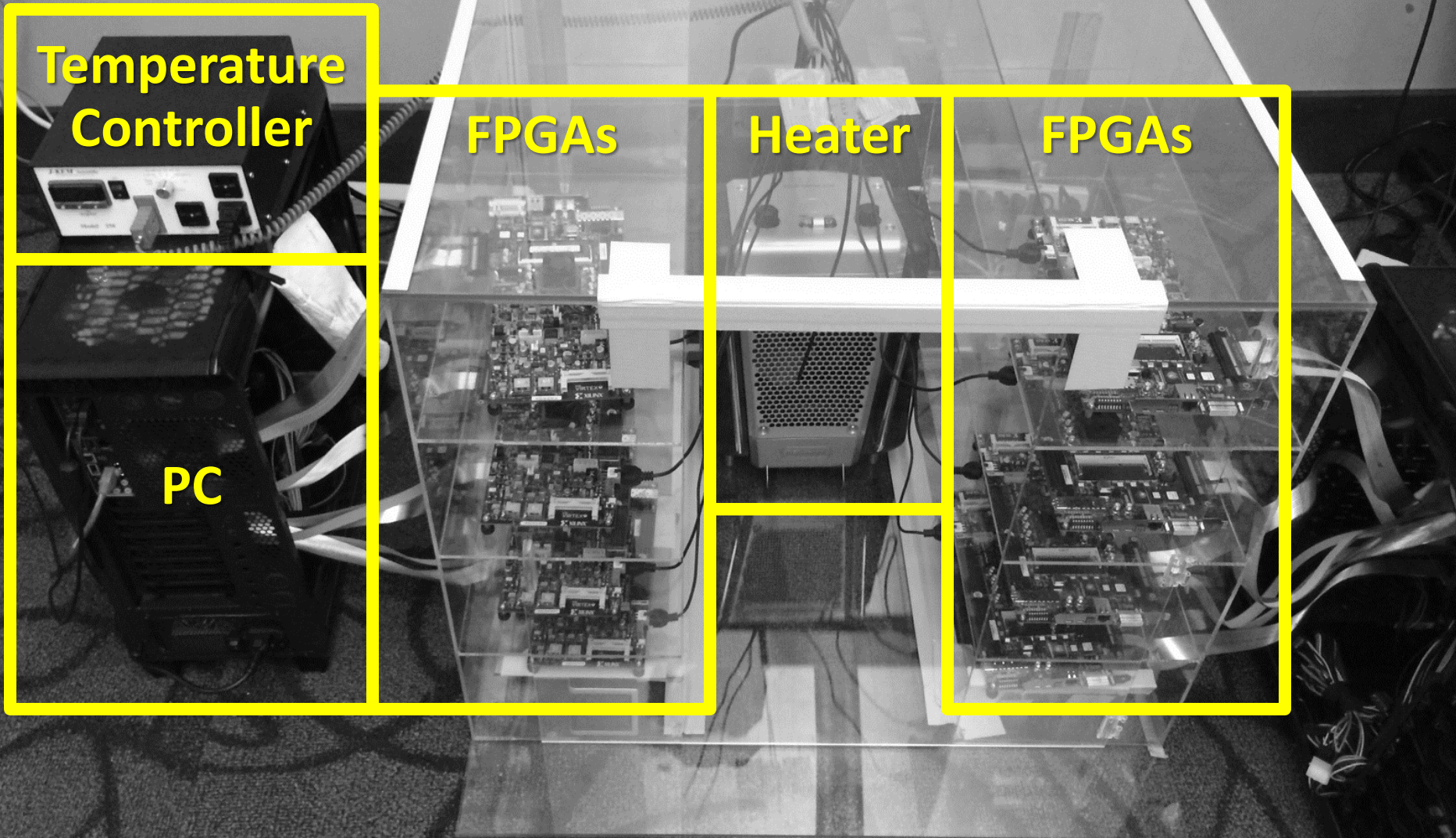
Temperature  
Controller

FPGAs

Heater

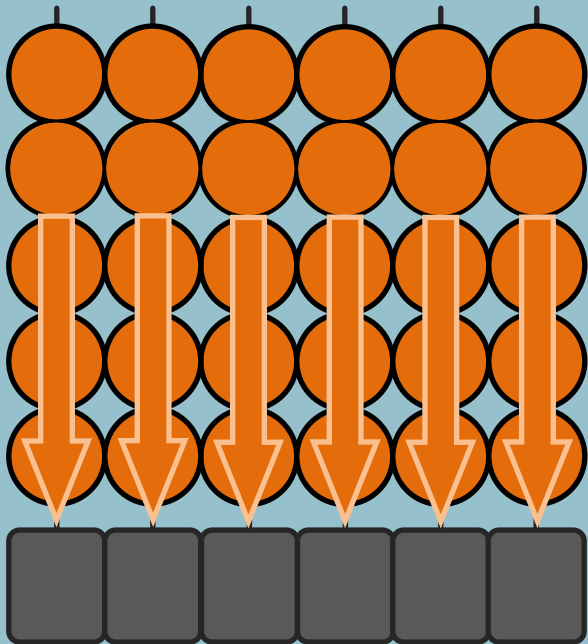
FPGAs

PC



# Obs 1. Faster Sensing

Typical DIMM at Low Temperature



More charge

Strong charge flow

Faster sensing

115 DIMM characterization

**Timing**  
( $t_{RCD}$ )

**17% ↓**

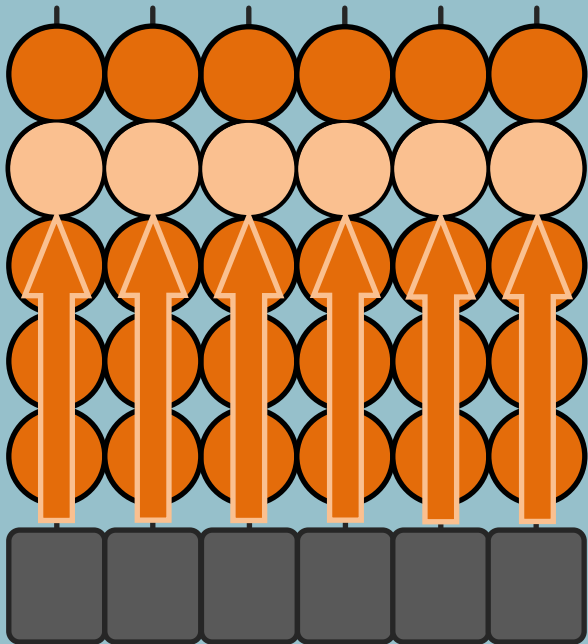
**No Errors**

Typical DIMM at Low Temperature

→ *More charge* → *Faster sensing*

# Obs 2. Reducing Restore Time

Typical DIMM at Low Temperature



Larger cell &  
Less leakage →  
Extra charge

No need to fully  
restore charge

115 DIMM  
characterization

**Read** ( $t_{RAS}$ )

**37% ↓**

**Write** ( $t_{WR}$ )

**54% ↓**

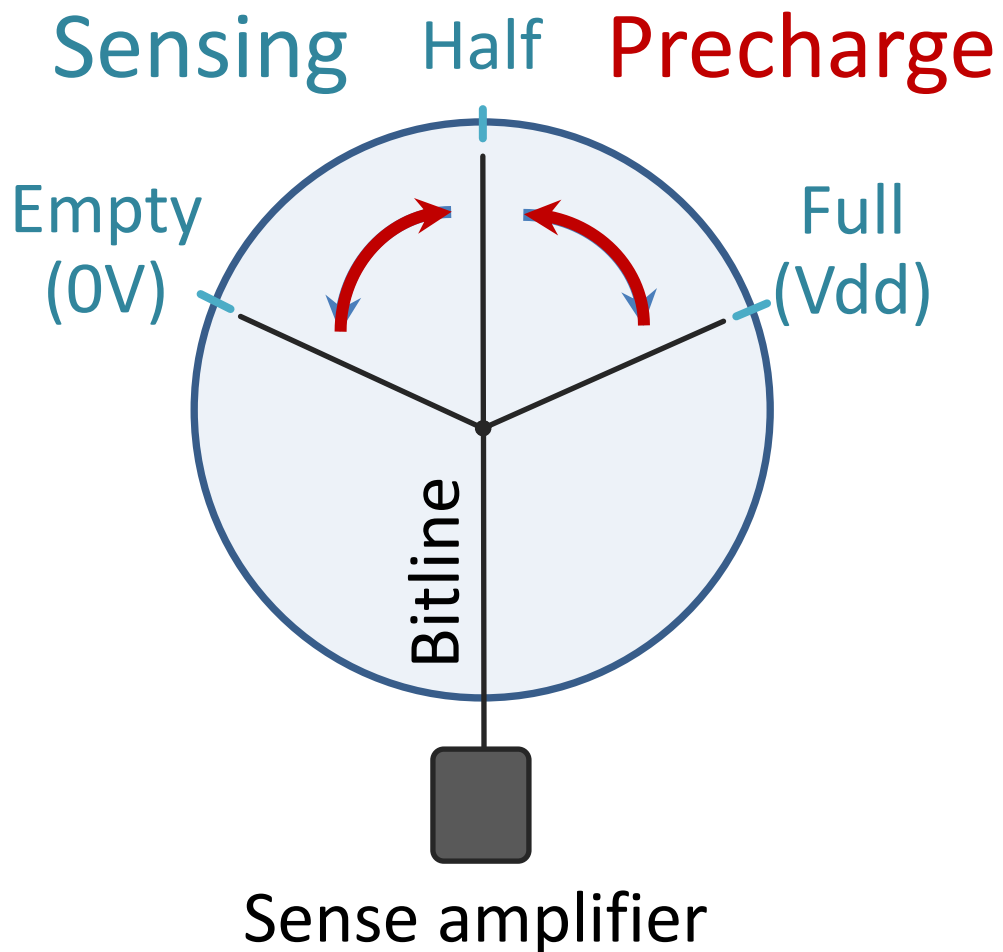
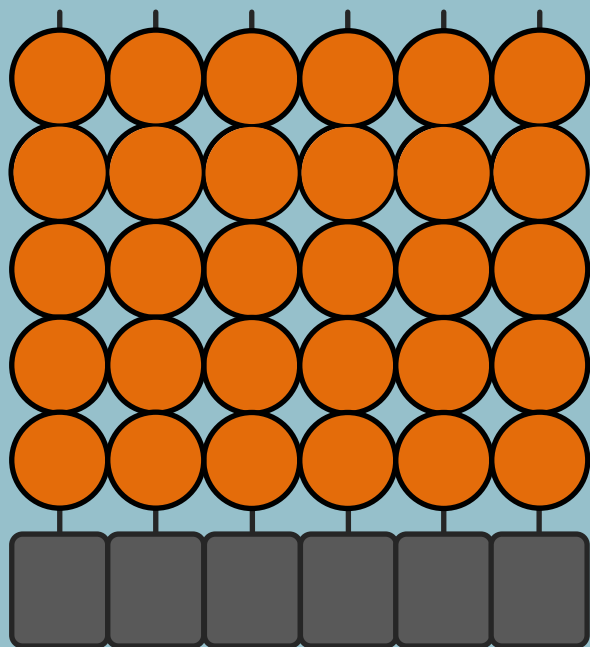
**No Errors**

Typical DIMM at lower temperature

→ More charge → Restore time reduction

# Obs 3. Reducing Precharge Time

Typical DIMM at Low Temperature



Precharge ? – Setting bitline to half-full charge

# Obs 3. Reducing Precharge Time

Access empty cell

Access full cell

115 DIMM  
characterization

Not fully  
precharged

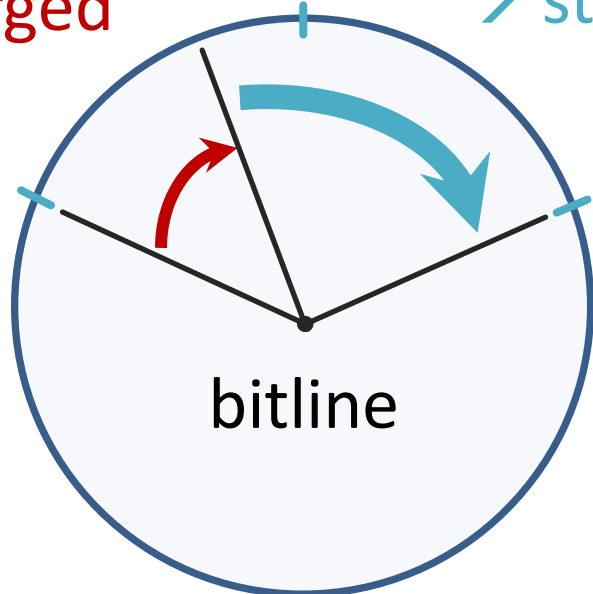
More charge

→ strong sensing

Empty  
(0V)

Half

Full (V<sub>dd</sub>)



**Timing**  
( $\tau_{RP}$ )

**35% ↓**

**No Errors**

Typical DIMM at Lower Temperature

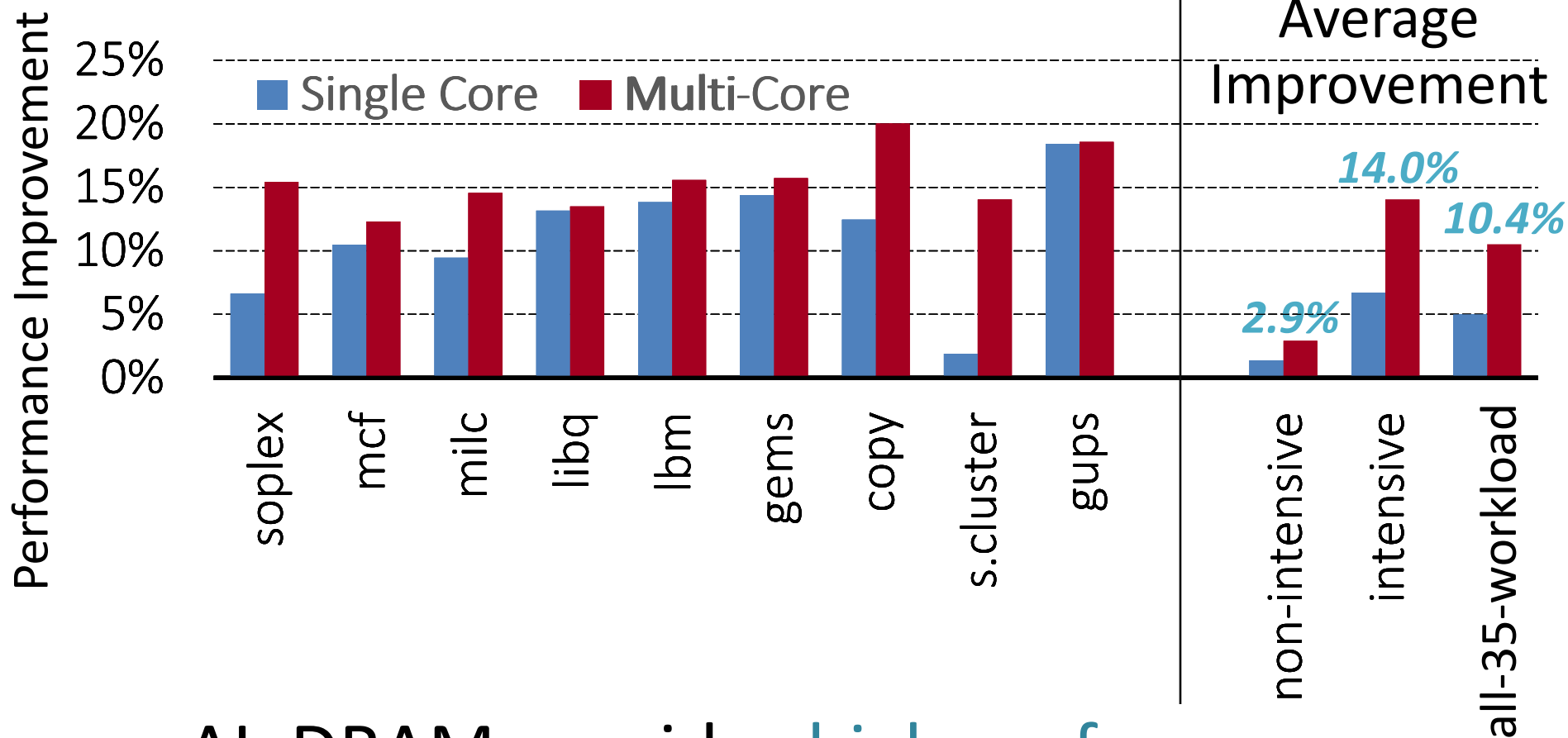
→ More charge → Precharge time reduction



# Adaptive-Latency DRAM

- Key idea
  - Optimize DRAM timing parameters online
- Two components
  - DRAM manufacturer profiles multiple sets of reliable DRAM timing parameters different temperatures for each DIMM
  - System monitors DRAM temperature uses appropriate DRAM timing parameters

# Real System Evaluation



AL-DRAM provides high performance improvement, greater for multi-core workloads

# Summary: AL-DRAM

- Observation
  - DRAM timing parameters are dictated by the worst-case cell (smallest cell at highest temperature)
- Our Approach: *Adaptive-Latency DRAM (AL-DRAM)*
  - Optimizes DRAM timing parameters for *the common case* (typical DIMM operating at low temperatures)
- Analysis: Characterization of 115 DIMMs
  - Great potential to *lower DRAM timing parameters (17 – 54%)* without any errors
- Real System Performance Evaluation
  - Significant *performance improvement (14%* for memory-intensive workloads) without errors (*33* days)

# Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case

Donghyuk Lee, Yoongu Kim,

Gennady Pekhimenko, Samira Khan, Vivek  
Seshadri, Kevin Chang, and Onur Mutlu

Published in the proceedings of 21<sup>st</sup>

**International Symposium on High Performance  
Computer Architecture 2015**

# Outline

1. What is DRAM?

2. DRAM Internal Organization

3. Problems and Solutions

– Latency (Tiered-Latency DRAM, HPCA 2013;  
Adaptive-Latency DRAM, HPCA 2015)

– Parallelism (Subarray-level Parallelism, ISCA 2012)

# Parallelism: Demand vs. Supply

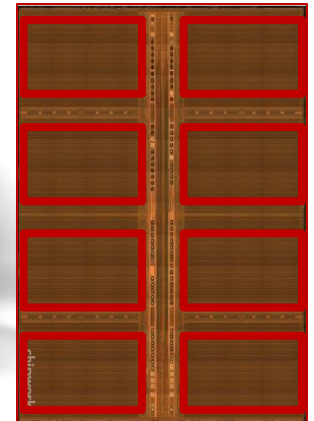
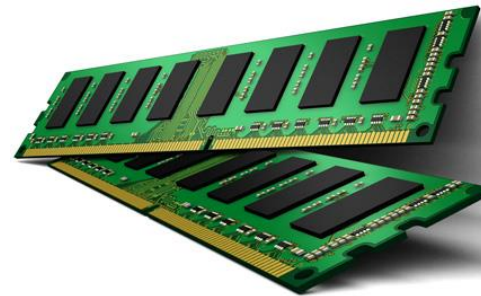
Demand

Supply

Out-of-order  
Execution

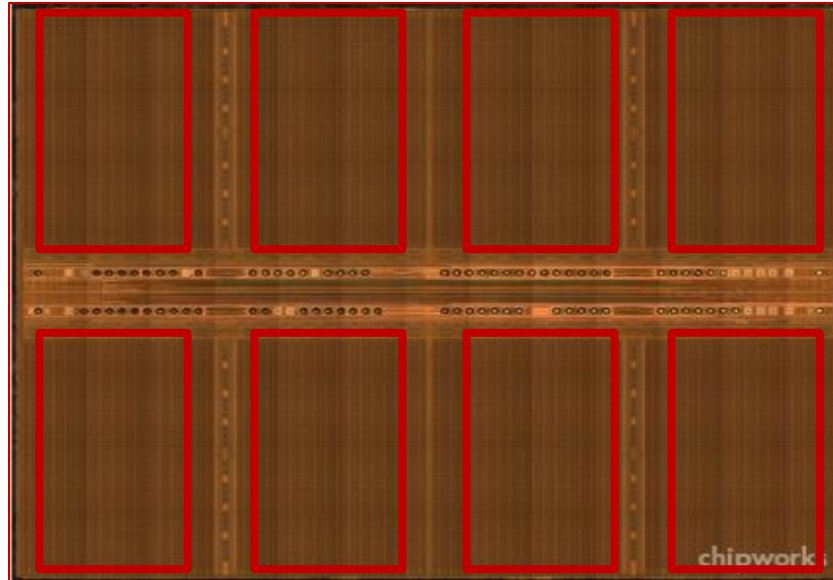
Multi-cores

Prefetchers



Multiple  
Banks

# Increasing Number of Banks?



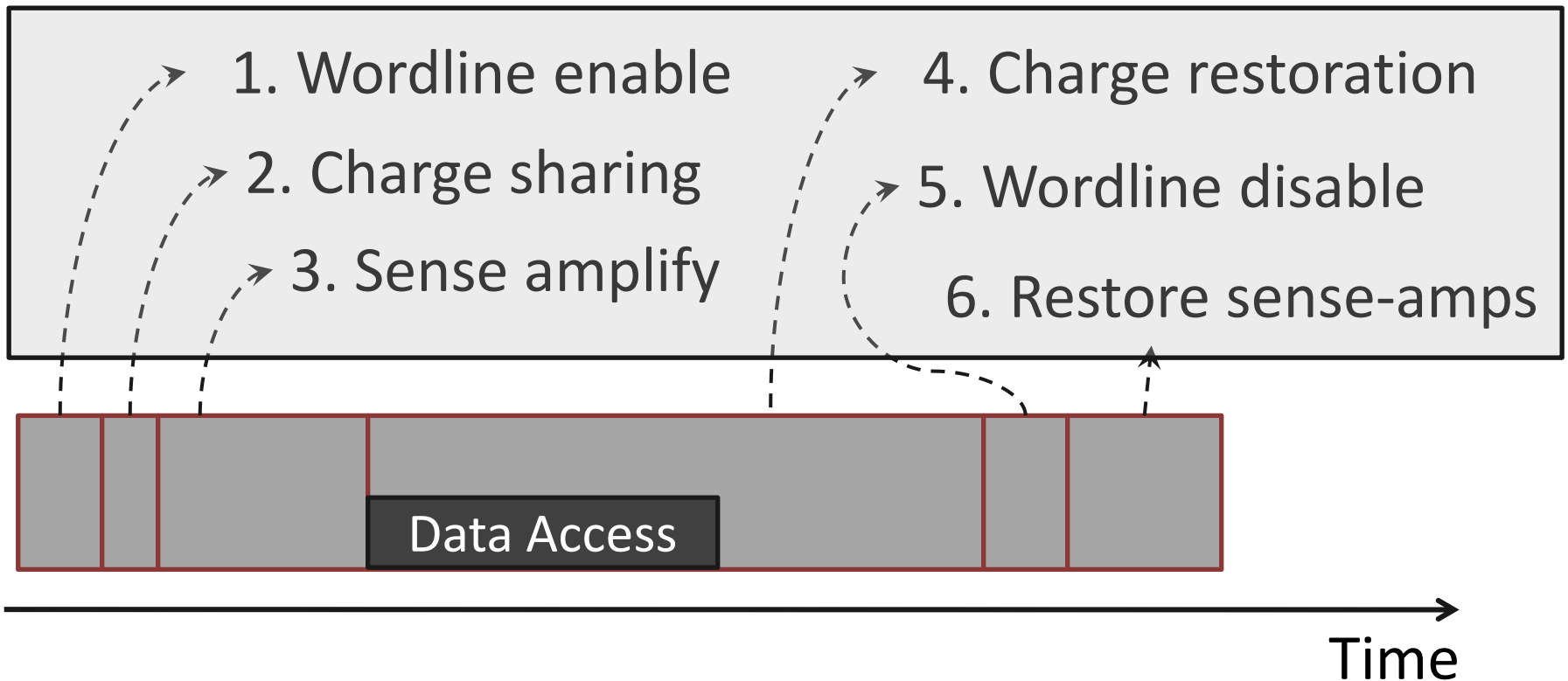
Adding more banks → Replication of shared structures

Replication → Cost

How to improve available parallelism within DRAM?

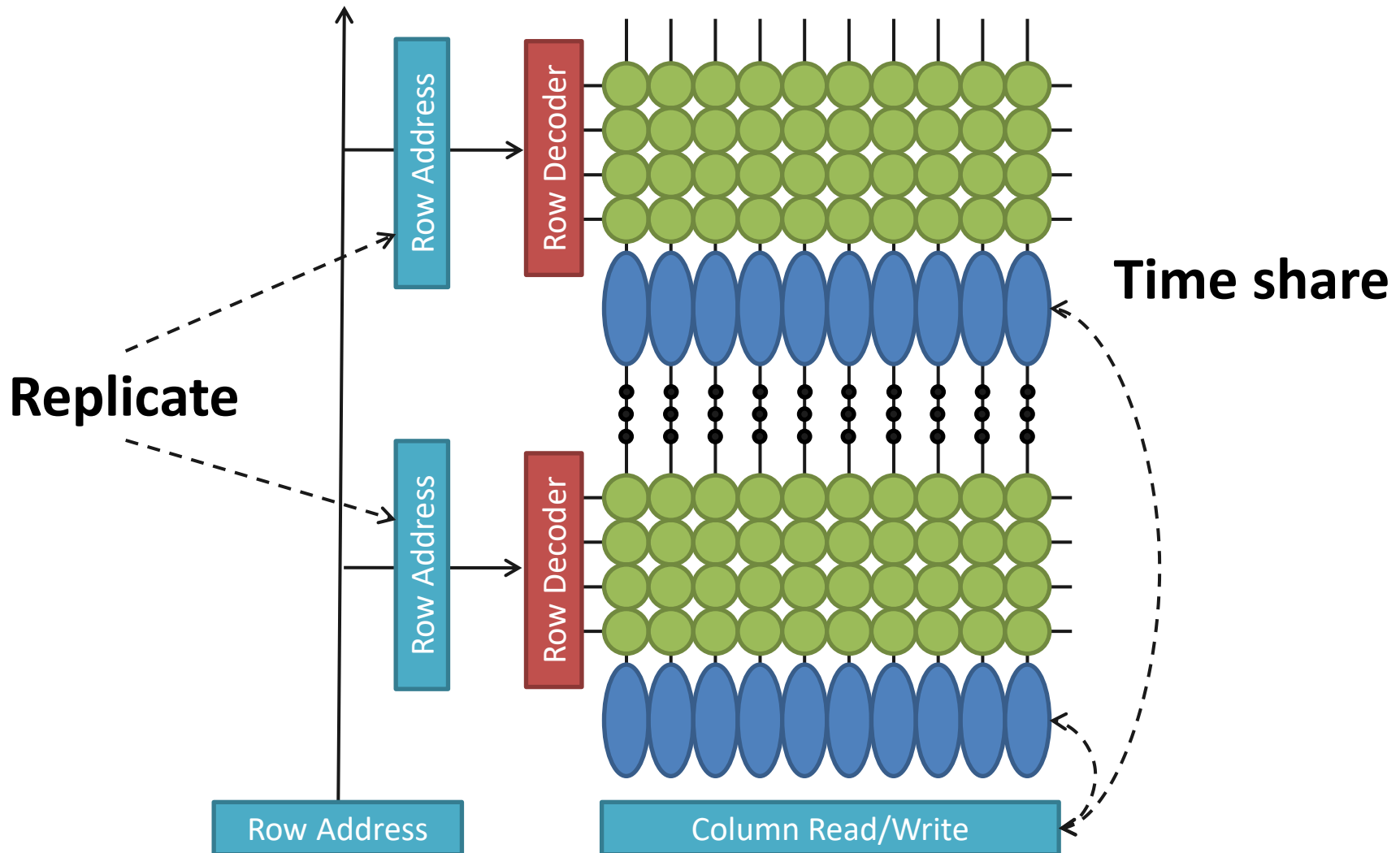
# Our Observation

Local to a subarray

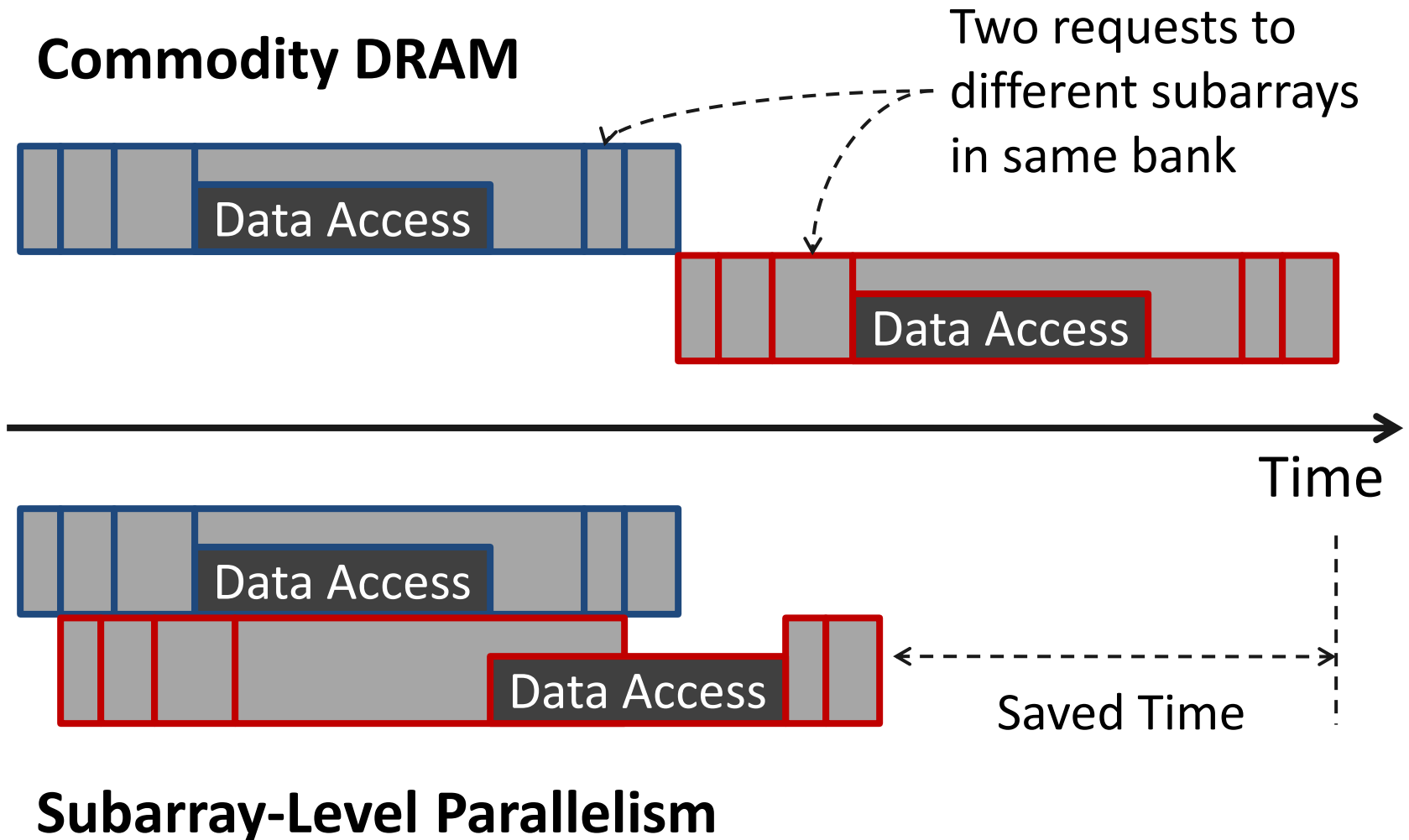




# Subarray-Level Parallelism



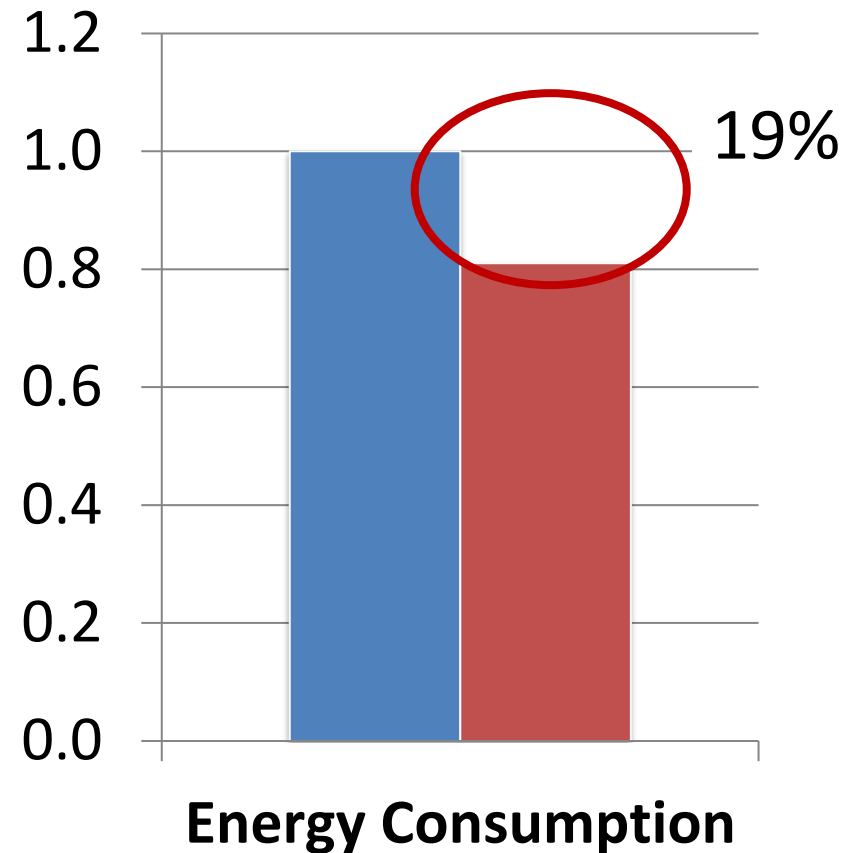
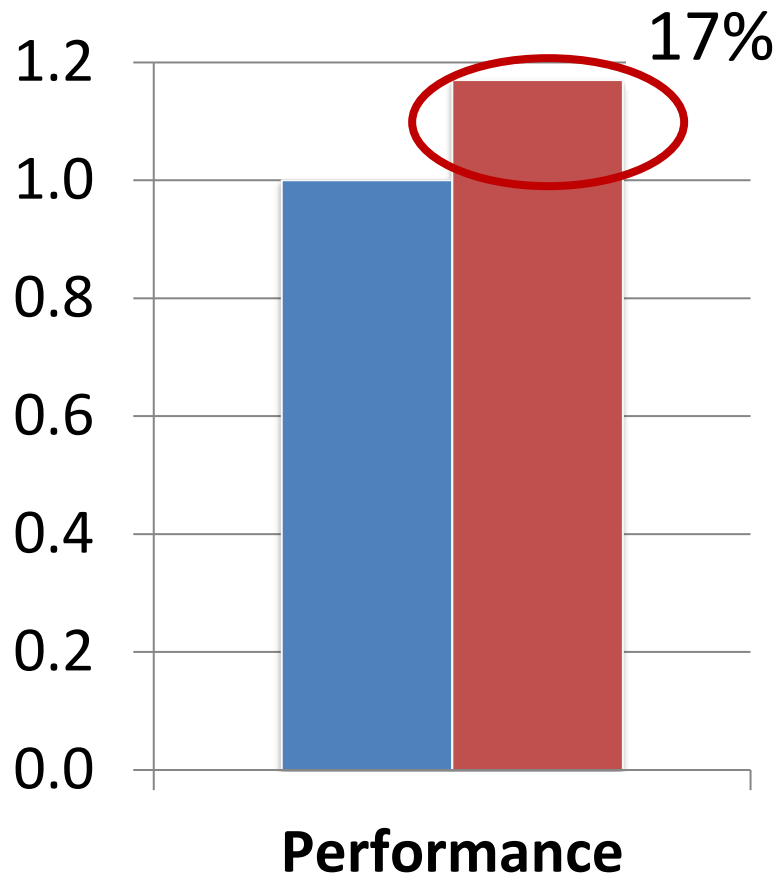
# Subarray-Level Parallelism: Benefits



# Results Summary

■ Commodity DRAM

■ Subarray-Level Parallelism



# A Case for Exploiting Subarray-Level Parallelism (SALP) in DRAM

Yoongu Kim, Vivek Seshadri, Donghyuk Lee,  
Jamie Liu, Onur Mutlu

Published in the proceedings of 39<sup>th</sup>

**International Symposium on Computer Architecture  
2012**

# Review #4

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

Vivek Seshadri et al., *MICRO 2013*

# CSC 2224: Parallel Computer Architecture and Programming Main Memory Fundamentals

Prof. Gennady Pekhimenko

University of Toronto

Fall 2020

*The content of this lecture is adapted from the slides of  
Vivek Seshadri, Donghyuk Lee, Yoongu Kim,  
and lectures of Onur Mutlu @ ETH and CMU*