Low-Cost Inter-Linked Subarrays (LISA)

Enabling Fast Inter-Subarray Data Movement in DRAM

Kevin Chang

Prashant Nair, Donghyuk Lee, Saugata Ghose,

Moinuddin Qureshi, and Onur Mutlu

SAFARI

CARET

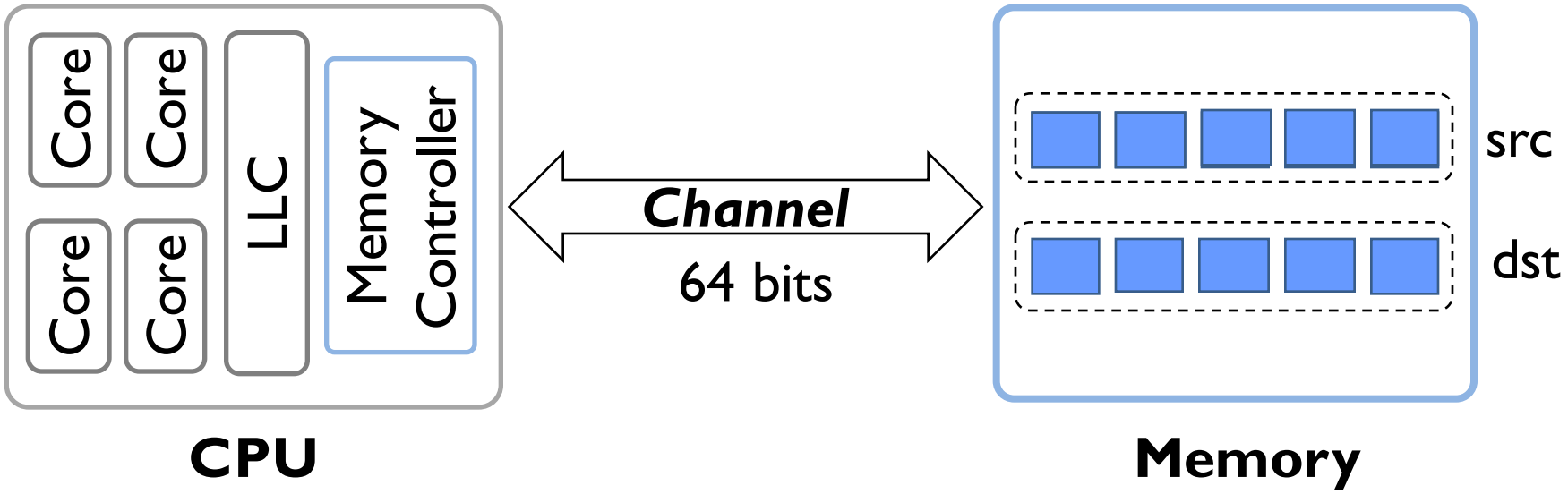
Carnegie Mellon

Georgia Tech

Problem: Inefficient Bulk Data Movement

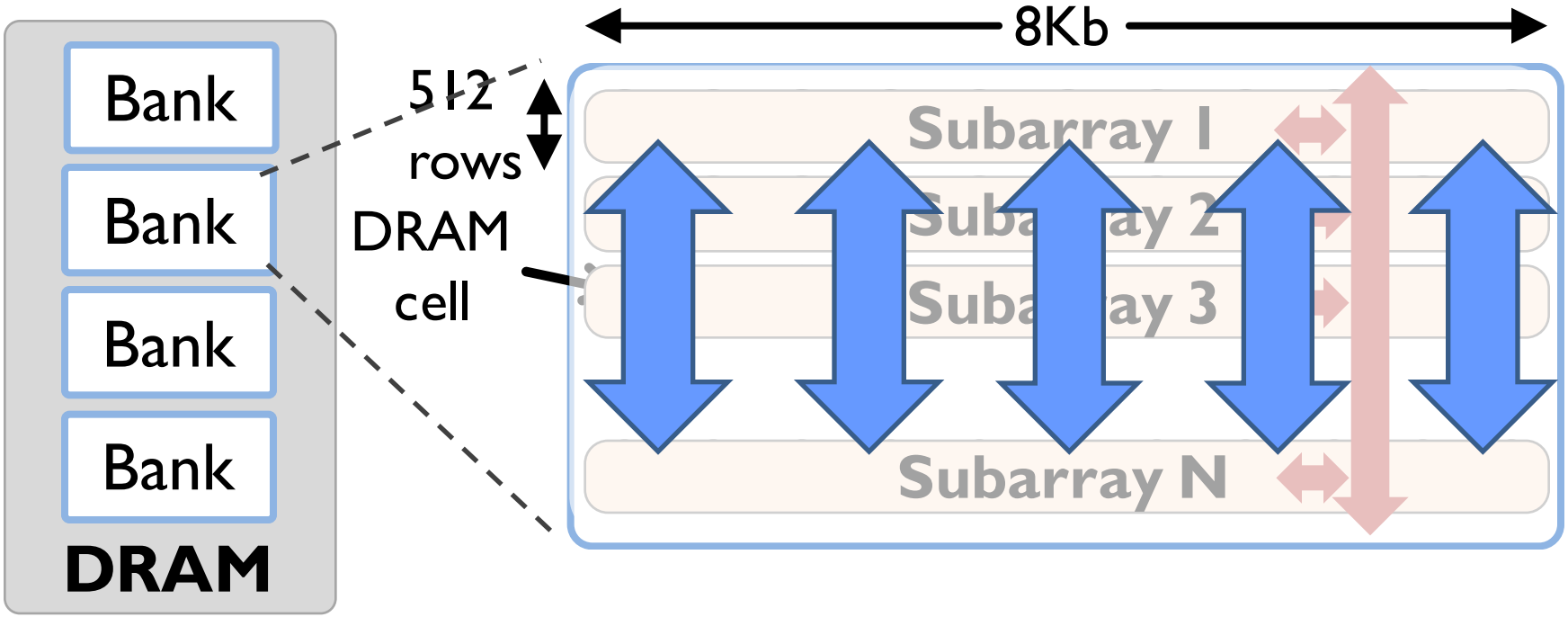
Bulk data movement is a key operation in many applications

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



Long latency and high energy

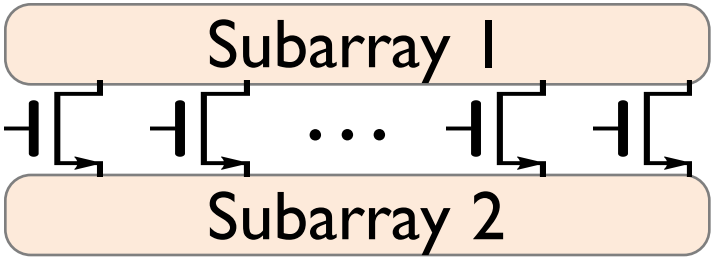
Moving Data Inside DRAM?



Goal: Provide a new substrate to enable wide connectivity between subarrays

Key Idea and Applications

- **Low-cost Inter-linked subarrays (LISA)**
  - Fast bulk data movement between subarrays
  - Wide datapath via isolation transistors: 0.8% DRAM chip area



- LISA is a **versatile substrate** → new applications

  Fast bulk data copy: Copy latency 1.363ms→0.148ms (9.2x)
  → 66% speedup, -55% DRAM energy

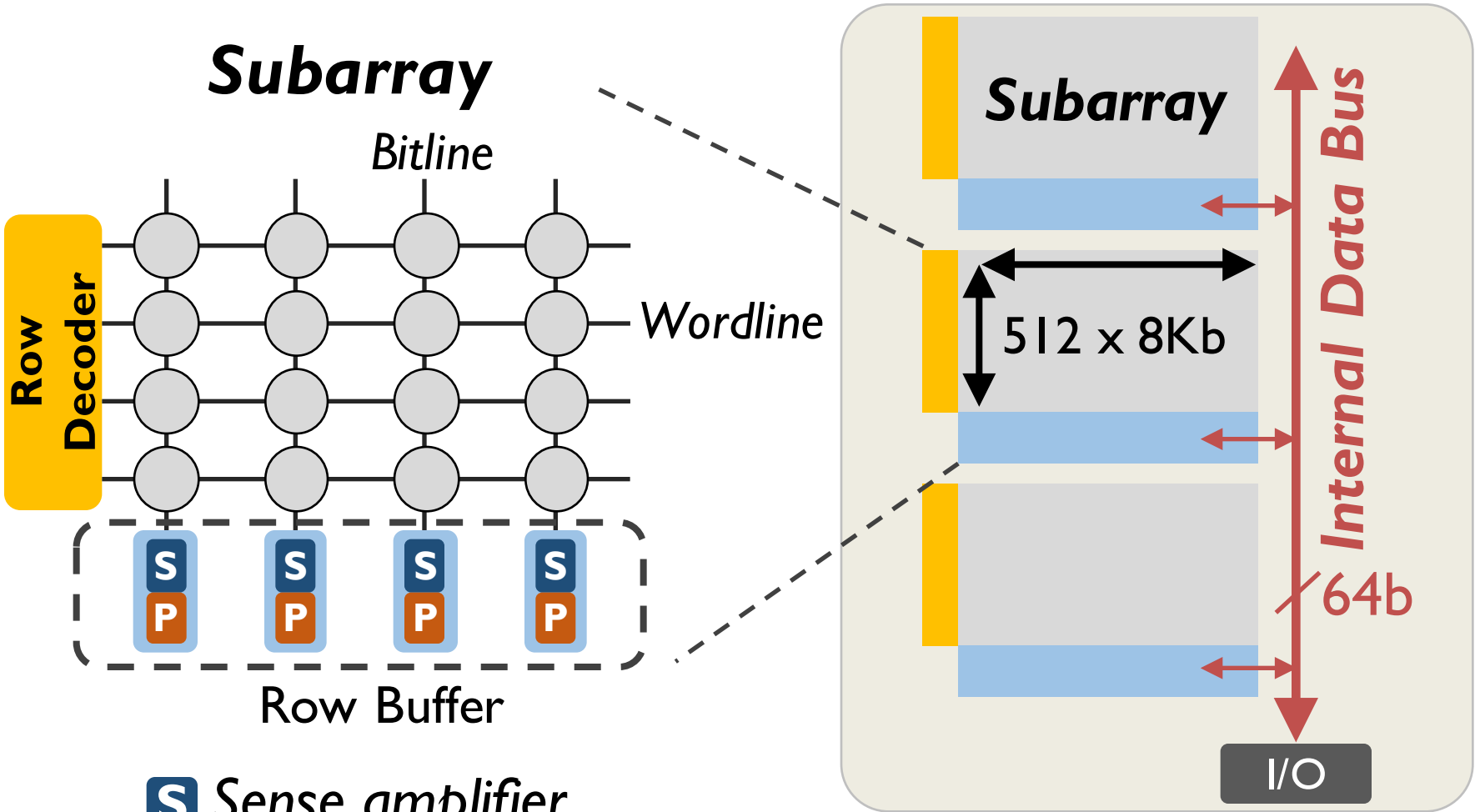
  In-DRAM caching: Hot data access latency 48.7ns→21.5ns (2.2x)
  → 5% speedup

  Fast precharge: Precharge latency 13.1ns→5.0ns (2.6x)
  → 8% speedup

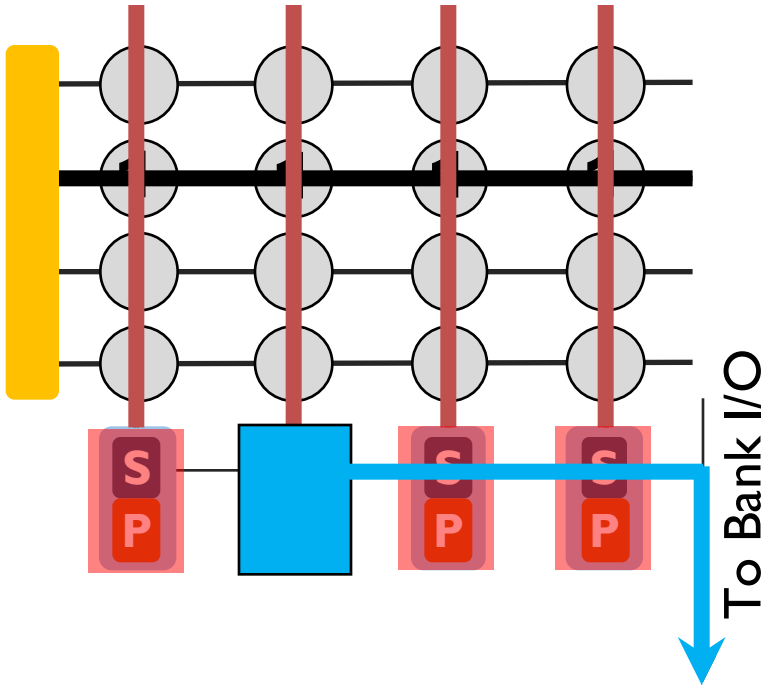
Outline

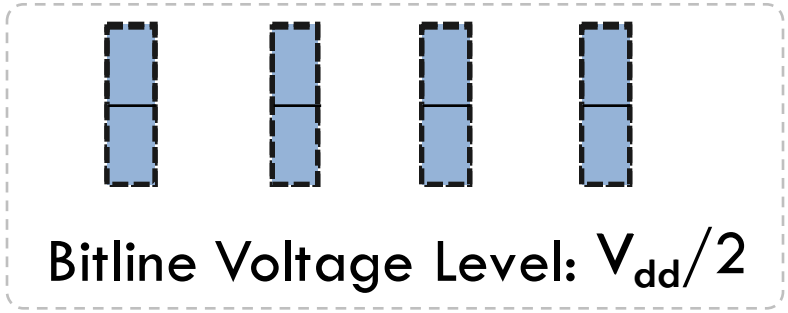
- Motivation and Key Idea
- DRAM Background
- LISA Substrate
  - New DRAM Command to Use LISA
- Applications of LISA

DRAM Internals



DRAM Operation





1 **ACTIVATE**: Store the row into the **row buffer**

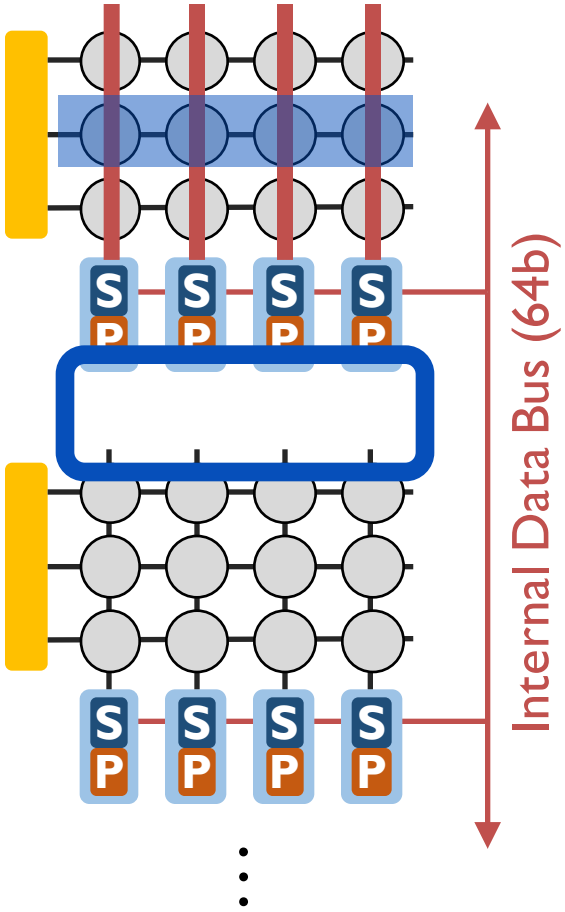
2 **READ**: Select the target column and drive to I/O

3 **PRECHARGE**: Reset the bitlines for a new ACTIVATE

Outline

- Motivation and Key Idea
- DRAM Background
- **LISA Substrate**
  - **New DRAM Command to Use LISA**
- Applications of LISA

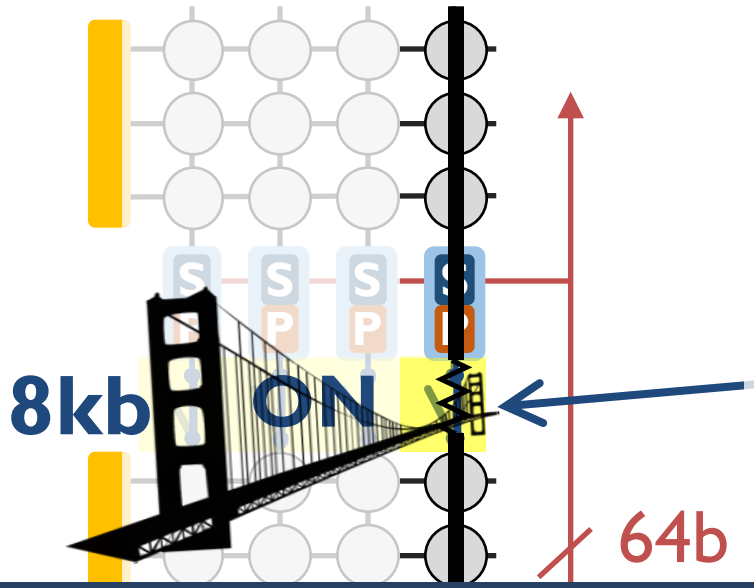
Observations



1 Bitlines serve as a bus that is as wide as a row

2 Bitlines between subarrays are close but disconnected

Low-Cost Interlinked Subarrays (LISA)

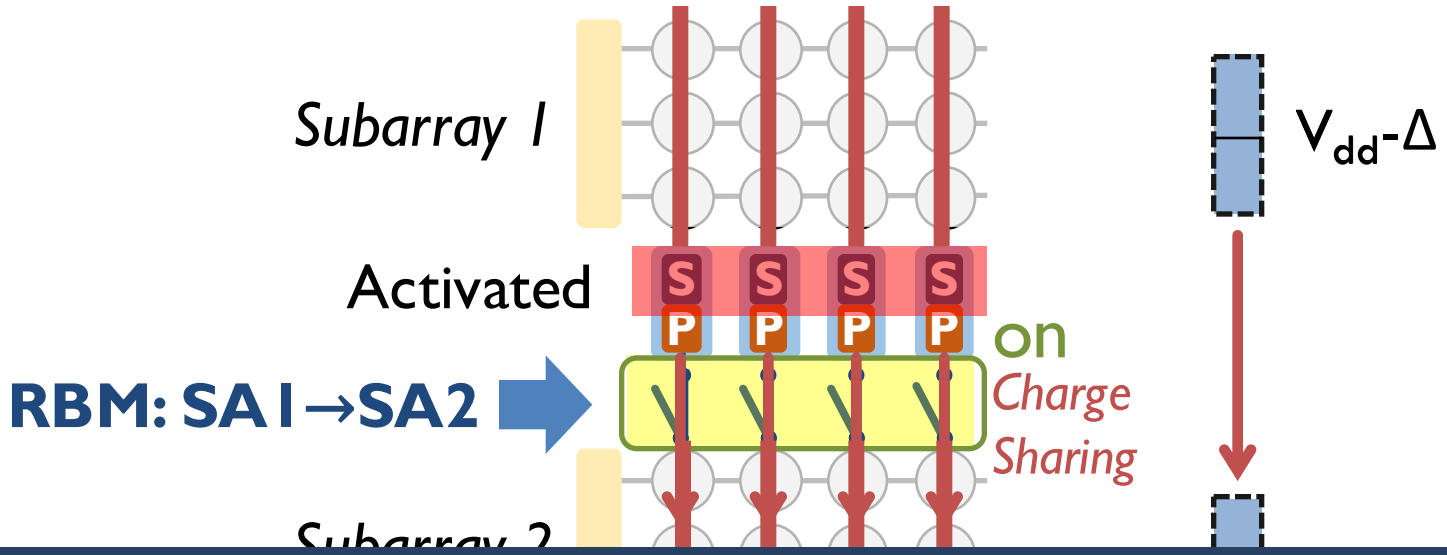


Interconnect bitlines of adjacent subarrays in a bank using **isolation transistors (links)**

LISA forms a wide datapath b/w subarrays

New DRAM Command to Use LISA

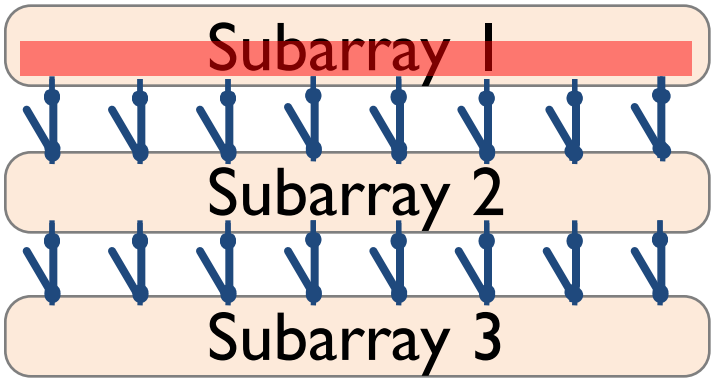
Row Buffer Movement (RBM): Move a row of data in an activated row buffer to a precharged one



RBM transfers an entire row b/w subarrays

RBM Analysis

- The range of RBM depends on the DRAM design
  - Multiple RBMs to move data across > 3 subarrays



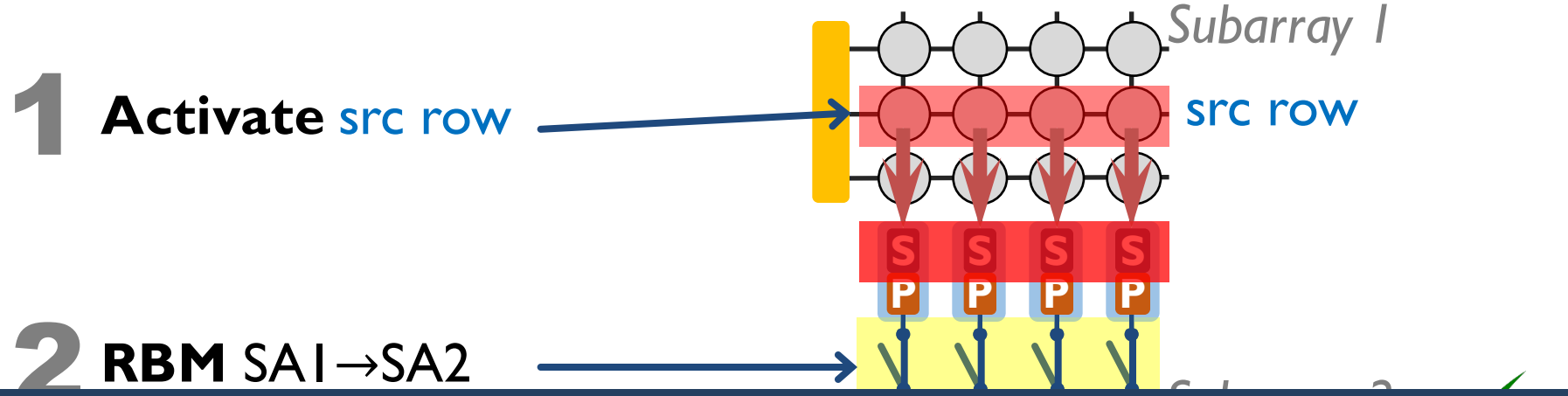
- Validated with SPICE using worst-case cells
  - NCSU FreePDK 45nm library
- **4KB data in 8ns** (w/ 60% guardband)
  → **500 GB/s, 26x** bandwidth of a DDR4-2400 channel
- **0.8%** DRAM chip area overhead [O+ ISCA'14]

Outline

- Motivation and Key Idea
- DRAM Background
- LISA Substrate
  - New DRAM Command to Use LISA
- **Applications of LISA**
  - **1**. Rapid Inter-Subarray Copying (RISC)
  - **2**. Variable Latency DRAM (VILLA)
  - **3**. Linked Precharge (LIP)

1. Rapid Inter-Subarray Copying (RISC)

- **Goal:** Efficiently copy a row across subarrays
- **Key idea:** Use *RBM* to form a new command sequence



Reduces row-copy latency by 9.2x, DRAM energy by 48.1x

Methodology

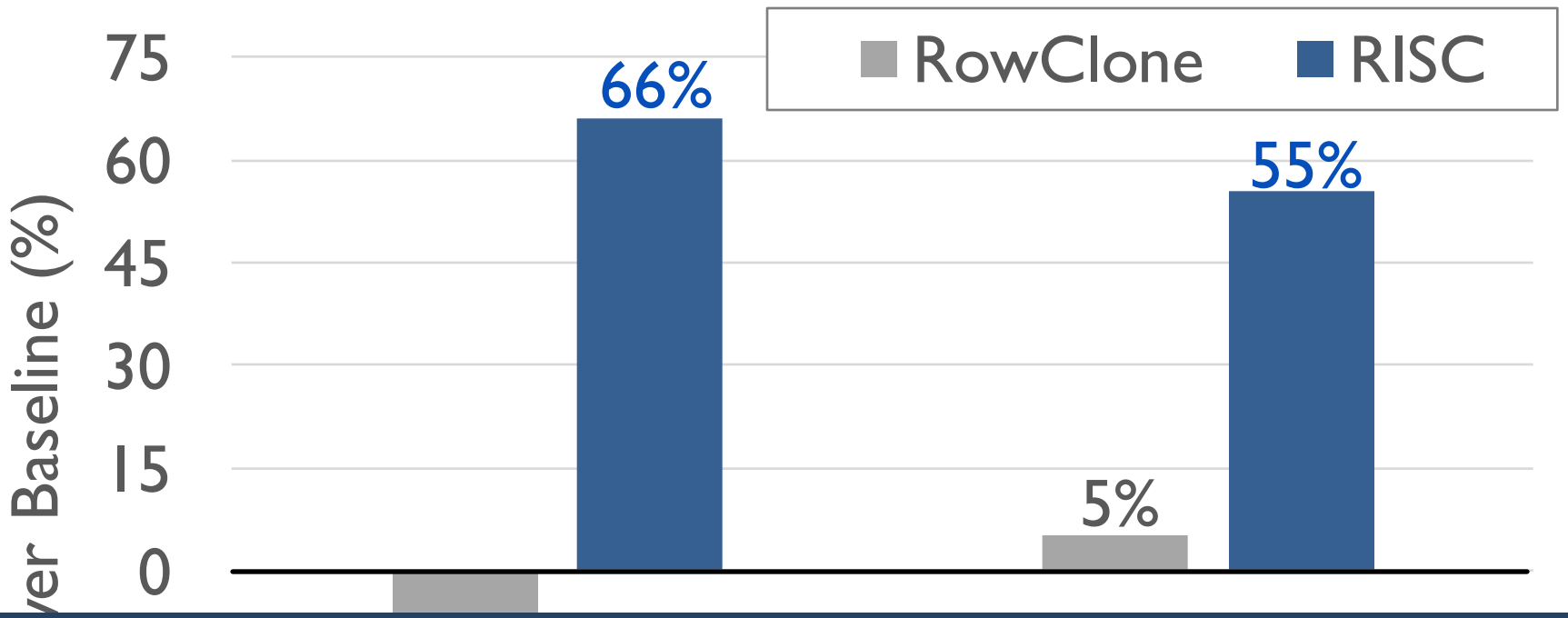
- Cycle-level simulator: Ramulator [CAL'15]
  https://github.com/CMU-SAFARI/ramulator
- CPU: **4 out-of-order cores**, 4GHz
- L1: 64KB/core, L2: 512KB/core, L3: shared 4MB
- DRAM: **DDR3-1600, 2 channels**
- Benchmarks:
  - **Memory-intensive**: TPC, STREAM, SPEC2006, DynoGraph, random
  - **Copy-intensive**: Bootup, forkbench, shell script
- 50 workloads: Memory- + copy-intensive
- Performance metric: Weighted Speedup (WS)

Comparison Points

- **Baseline**: Copy data through CPU (existing systems)

- **RowClone** [Seshadri+ MICRO'13]
  - In-DRAM bulk copy scheme
  - Fast **intra**-subarray copying via bitlines
  - Slow **inter**-subarray copying via internal data bus

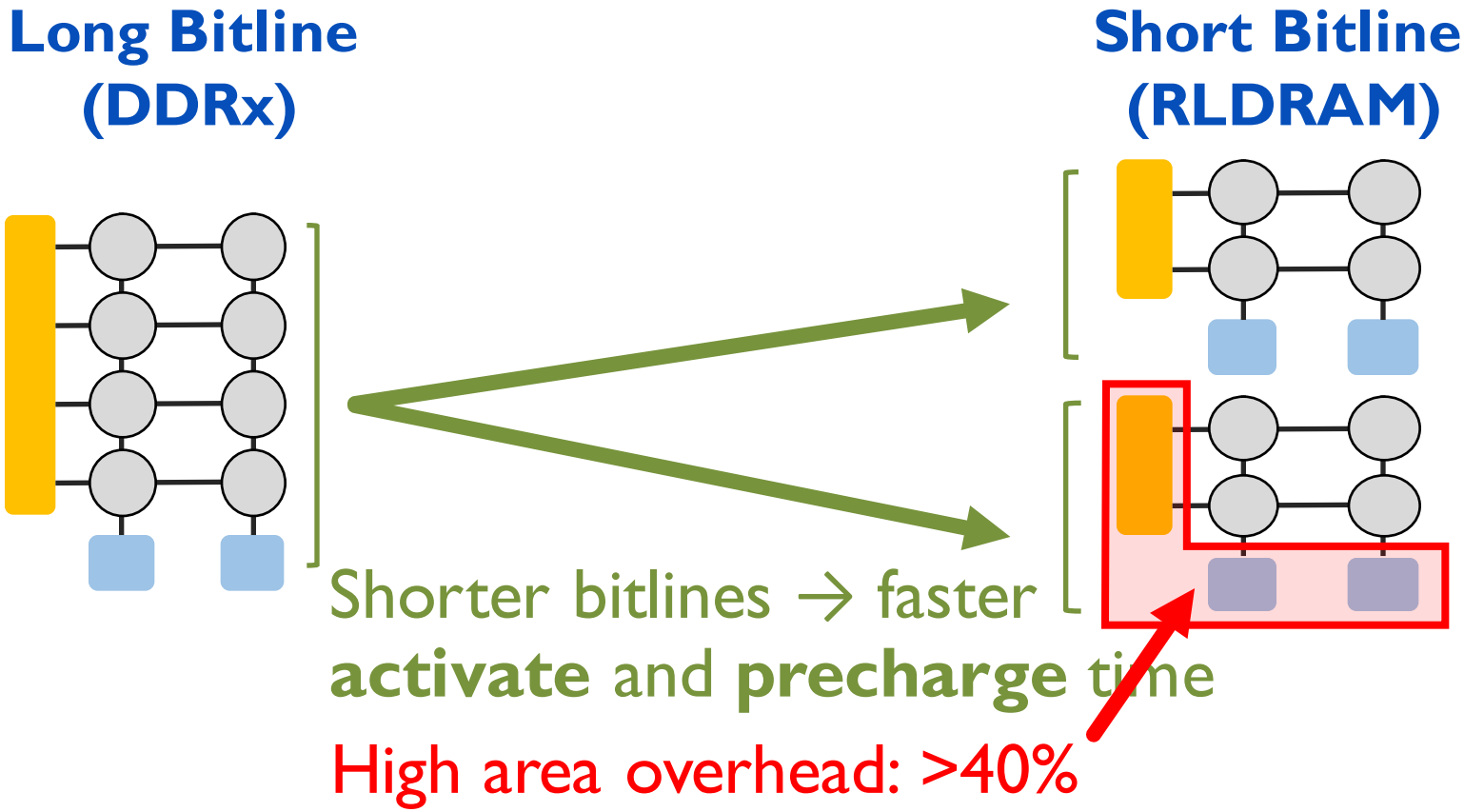
System Evaluation: RISC



Rapid Inter-Subarray Copying (RISC) using LISA improves system performance

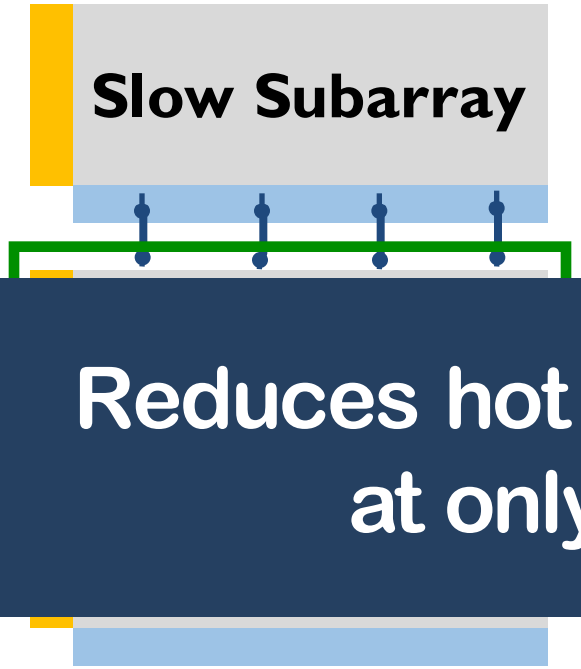
2. Variable Latency DRAM (VILLA)

- **Goal**: Reduce DRAM latency with low area overhead
- **Motivation**: Trade-off between area and latency



2. Variable Latency DRAM (VILLA)

- **Key idea**: Reduce access latency of hot data via a **heterogeneous DRAM** design [Lee+ HPCA'13, Son+ ISCA'13]
- **VILLA**: Add fast subarrays as a **cache** in each bank

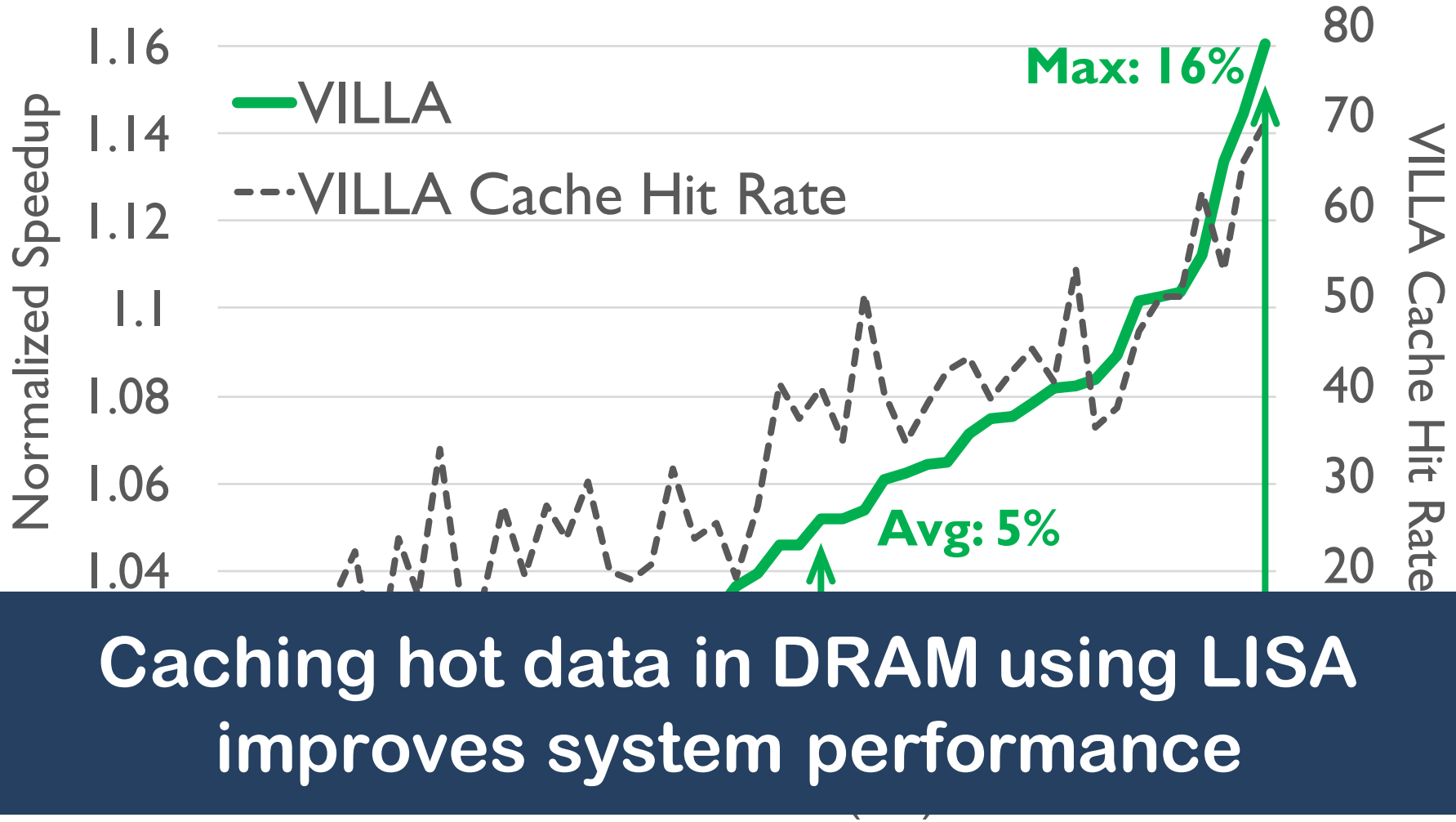


Challenge: VILLA cache requires frequent movement of data rows

Reduces hot data access latency by 2.2x at only 1.6% area overhead

System Evaluation: VILLA

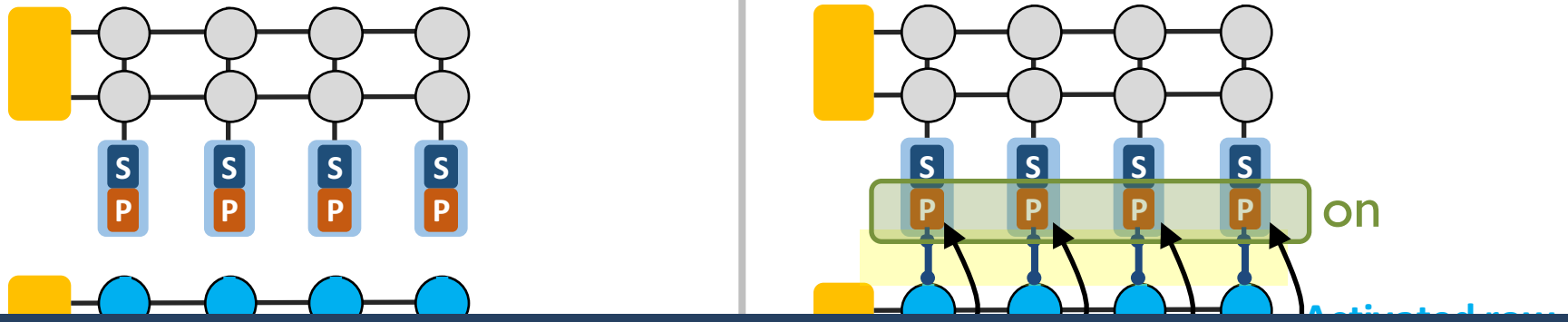
50 quad-core workloads: memory-intensive benchmarks



Caching hot data in DRAM using LISA improves system performance

3. Linked Precharge (LIP)

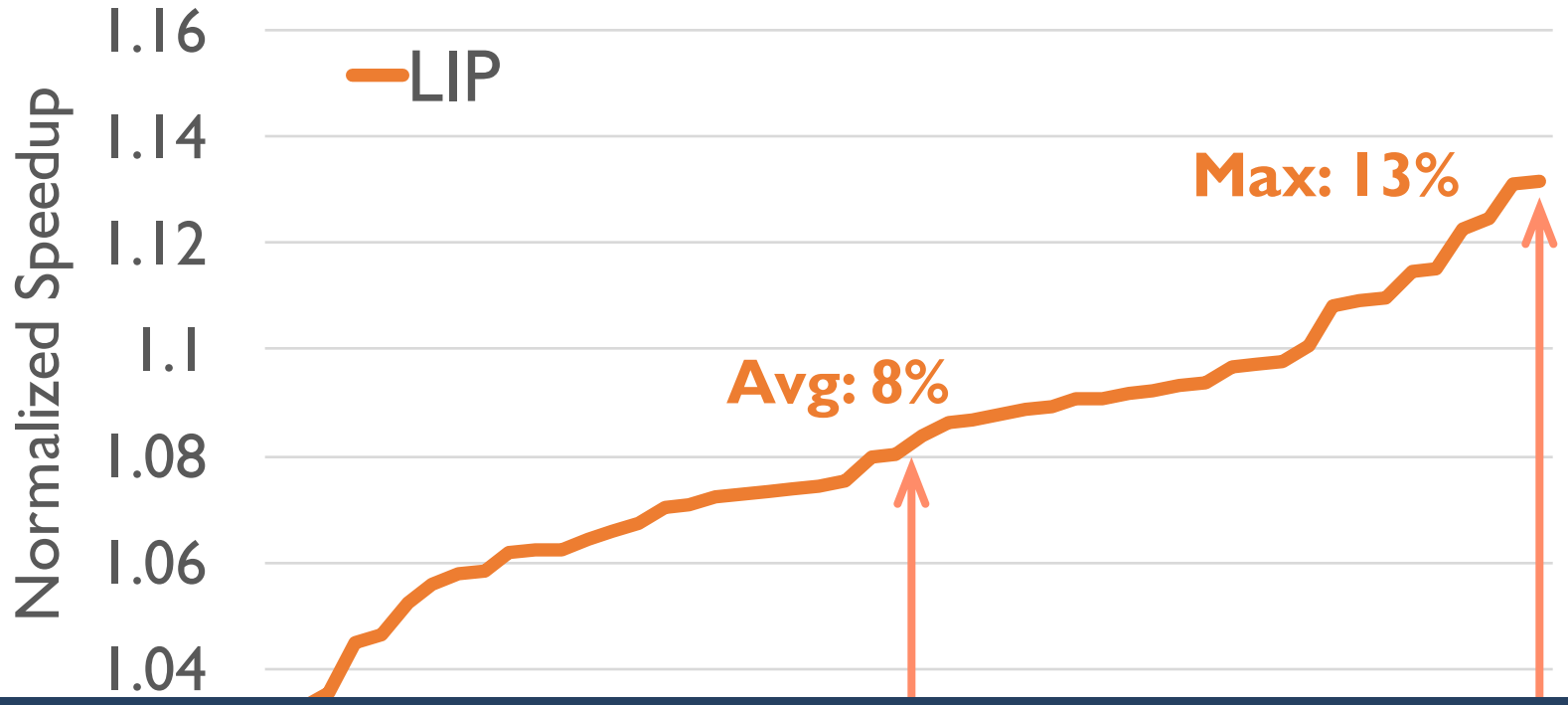
- **Problem**: The precharge time is limited by the strength of one precharge unit
- **Linked Precharge (LIP)**: LISA precharges a subarray using multiple precharge units



Reduces precharge latency by 2.6x (43% guardband)

System Evaluation: LIP

50 quad-core workloads: memory-intensive benchmarks



Accelerating precharge using LISA improves system performance

Other Results in Paper

- Combined applications
- Single-core results
- Sensitivity results
  - LLC size
  - Number of channels
  - Copy distance
- Qualitative comparison to other hetero. DRAM
- Detailed quantitative comparison to RowClone

Summary

- Bulk data movement is inefficient in today's systems
- Low connectivity between subarrays is a bottleneck
- **Low-cost Inter-linked subarrays (LISA)**
  - Bridge bitlines of subarrays via isolation transistors
  - Wide datapath with 0.8% DRAM chip area
- LISA is a **versatile substrate** → new applications
  - Fast bulk data copy: 66% speedup, -55% DRAM energy
  - In-DRAM caching: 5% speedup
  - Fast precharge: 8% speedup
  - LISA can enable other applications
- Source code will be available in April
  https://github.com/CMU-SAFARI

Low-Cost Inter-Linked Subarrays (LISA)

Enabling Fast Inter-Subarray Data Movement in DRAM

Kevin Chang

Prashant Nair, Donghyuk Lee, Saugata Ghose,

Moinuddin Qureshi, and Onur Mutlu

SAFARI

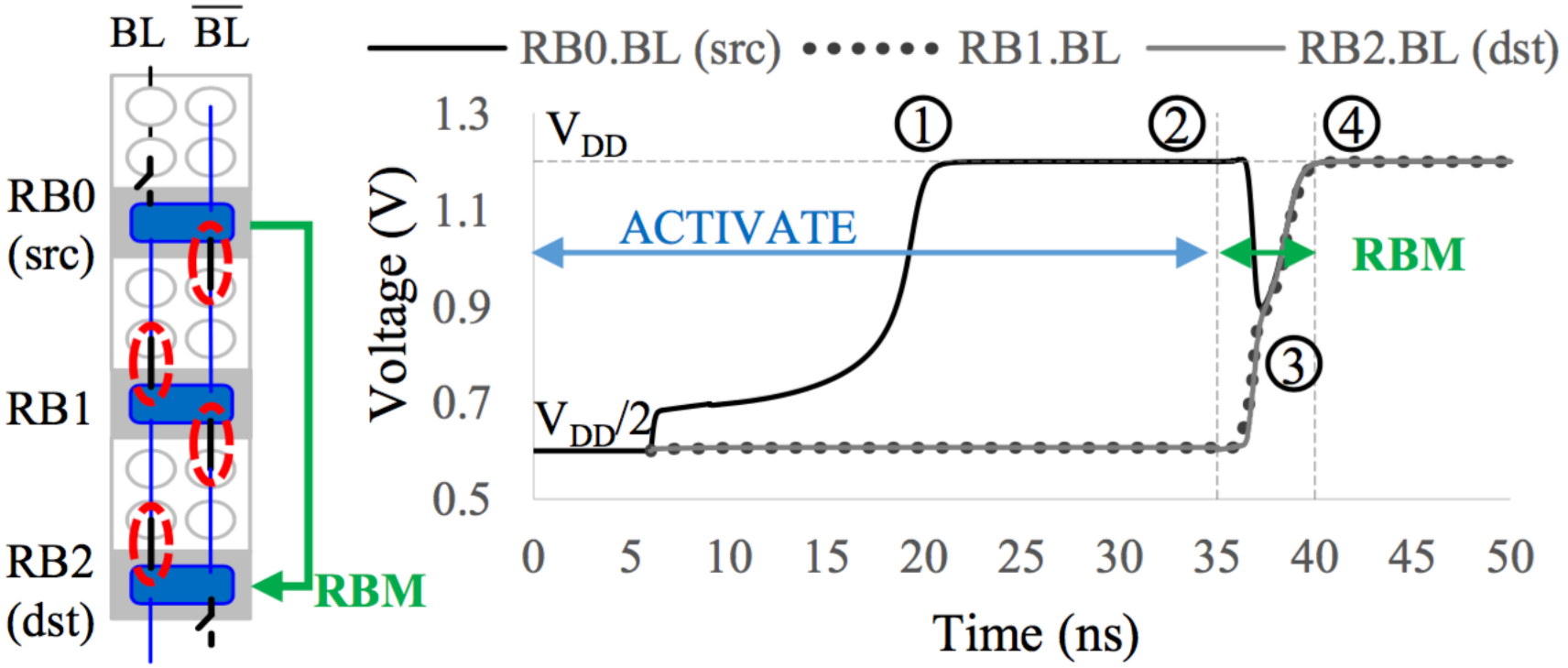
CARET

Carnegie Mellon

Georgia Tech

Backup

SPICE on RBM


BL
$\overline{BL}$
RB0
(src)
RB1
RB2
(dst)
RBM
RB0.BL (src)
RB1.BL
RB2.BL (dst)
Voltage (V)
1.3
1.1
0.9
0.7
0.5
V_{DD}
$V_{DD}/2$
ACTIVATE
RBM
①
②
③
④
0
5
10
15
20
25
30
35
40
45
50
Time (ns)

Comparison to Prior Works

Heterogeneous DRAM Designs	*TL-DRAM* (Lee+ HPCA'13)	*CHARM* (Son+ ISCA'13)	***VILLA***
Level of Heterogeneity	**Intra-Subarray**	**Inter-Bank**	**Inter-Subarray**
Caching Latency	✓	✗	✓
Cache Utilization	✗	✓	✓

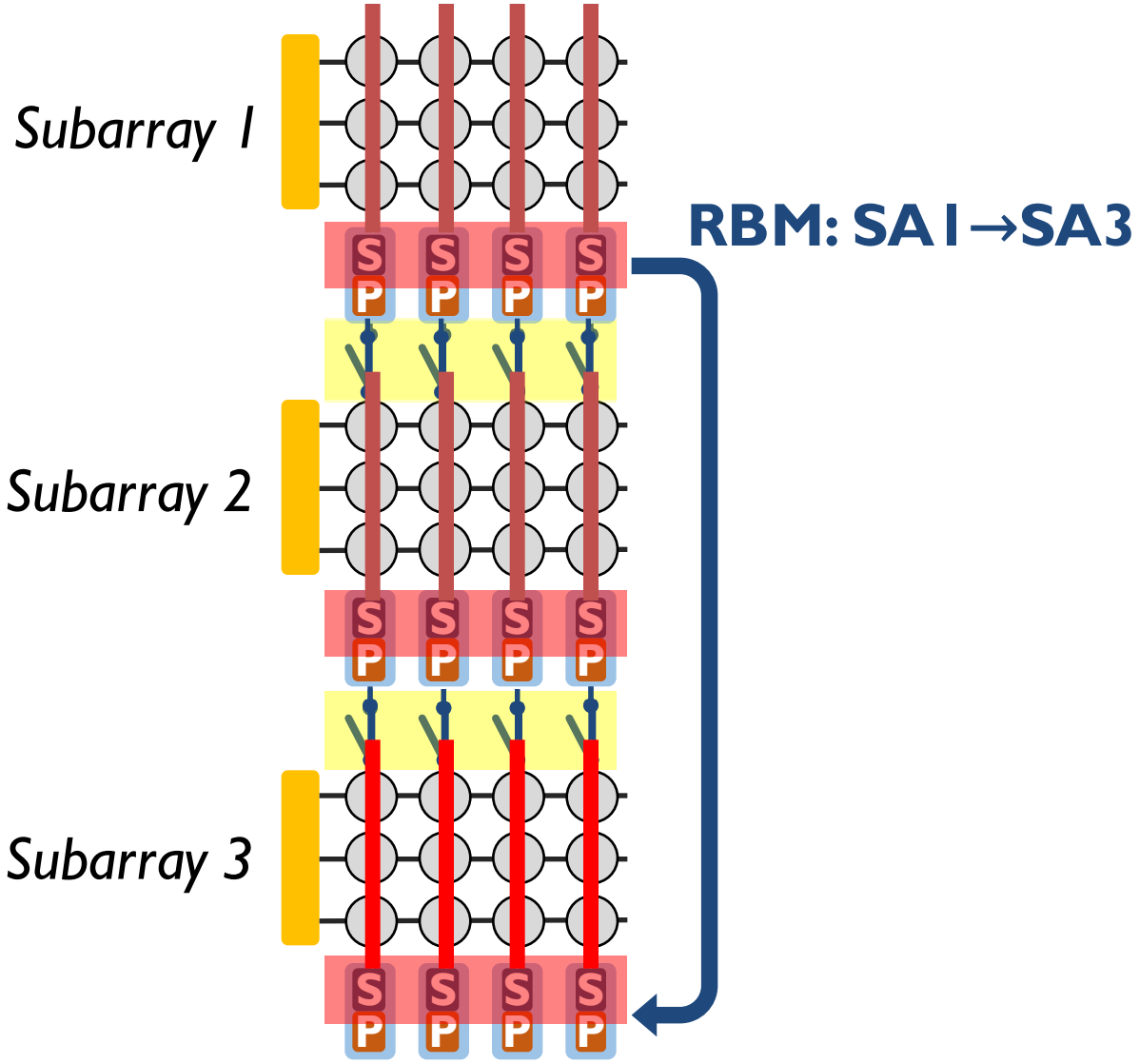
Comparison to Prior Works

DRAM Designs	*DAS-DRAM* (Lu+ MICRO'15)	*LISA*
Goal	Heterogeneous DRAM design	Substrate for bulk data movement
Enable other applications?		
Movement mechanism	Migration cells	Low-cost links
Scalable Copy Latency		

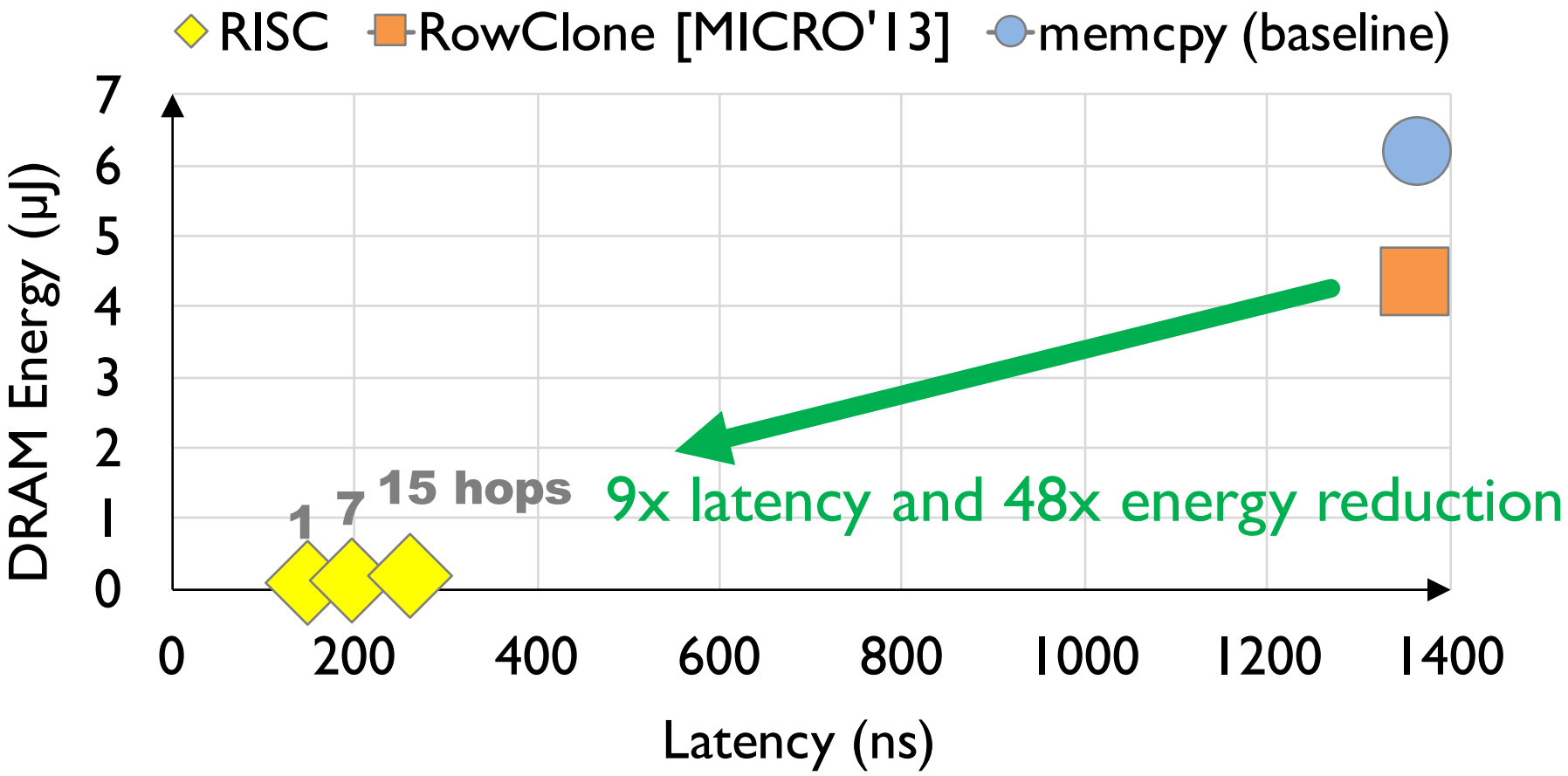
LISA vs. Samsung Patent

- S.-Y. Seo,
  "Methods of Copying a Page in a Memory Device and Methods of Managing Pages in a Memory System,"
  U.S. Patent Application 20140185395, 2014
- Only for copying data
- Vague. Lack of detail on implementation
  - How does data get moved? What are the steps?
- No analysis on the latency and energy
- No system evaluation

RBM Across 3 Subarrays



Comparison of Inter-Subarray Row Copying

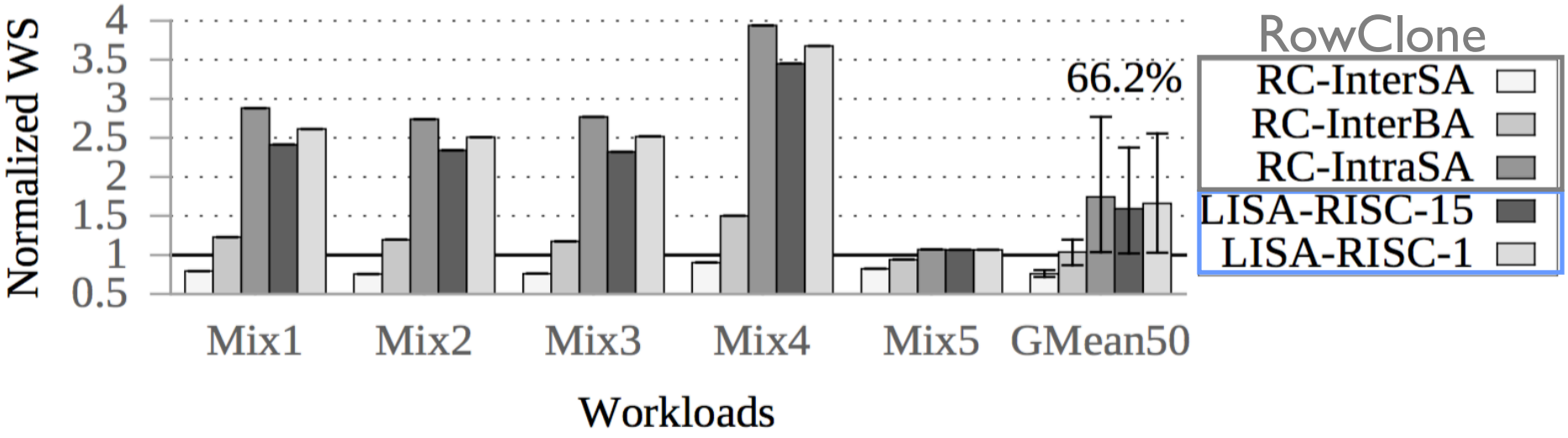


RISC: Cache Coherence

- Data in DRAM may not be up-to-date
- MC performs flushes dirty data (src) and invalidates dst
- Techniques to accelerate cache coherence
  - Dirty-Block Index [Seshadri+ ISCA'14]
- Other papers handle the similar issue
  [Seshadri+ MICRO'13, CAL'15]

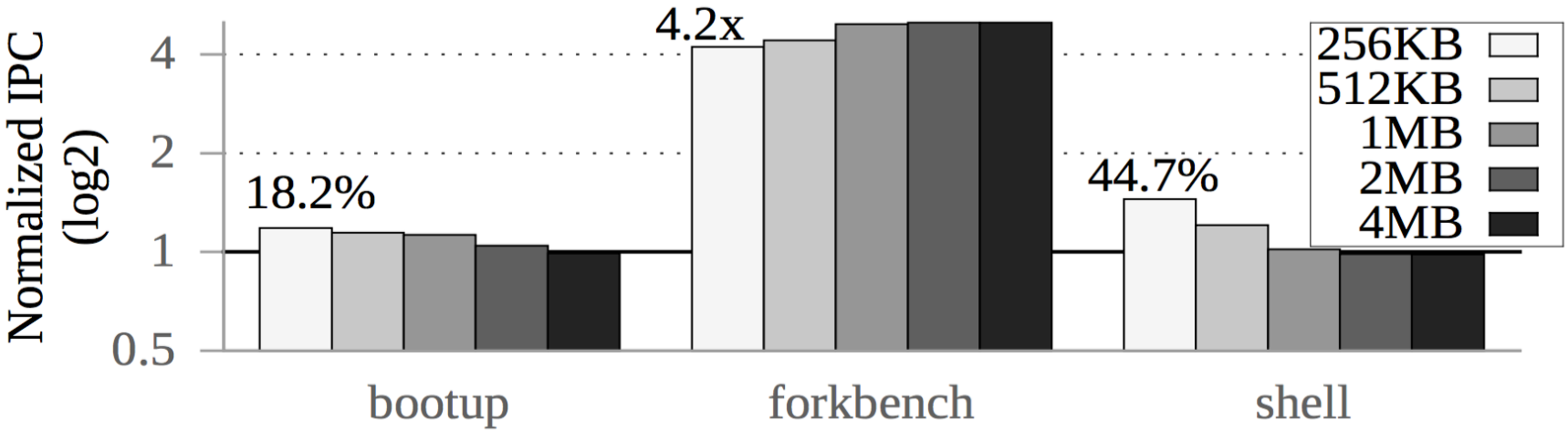
RISC vs. RowClone

4-core results



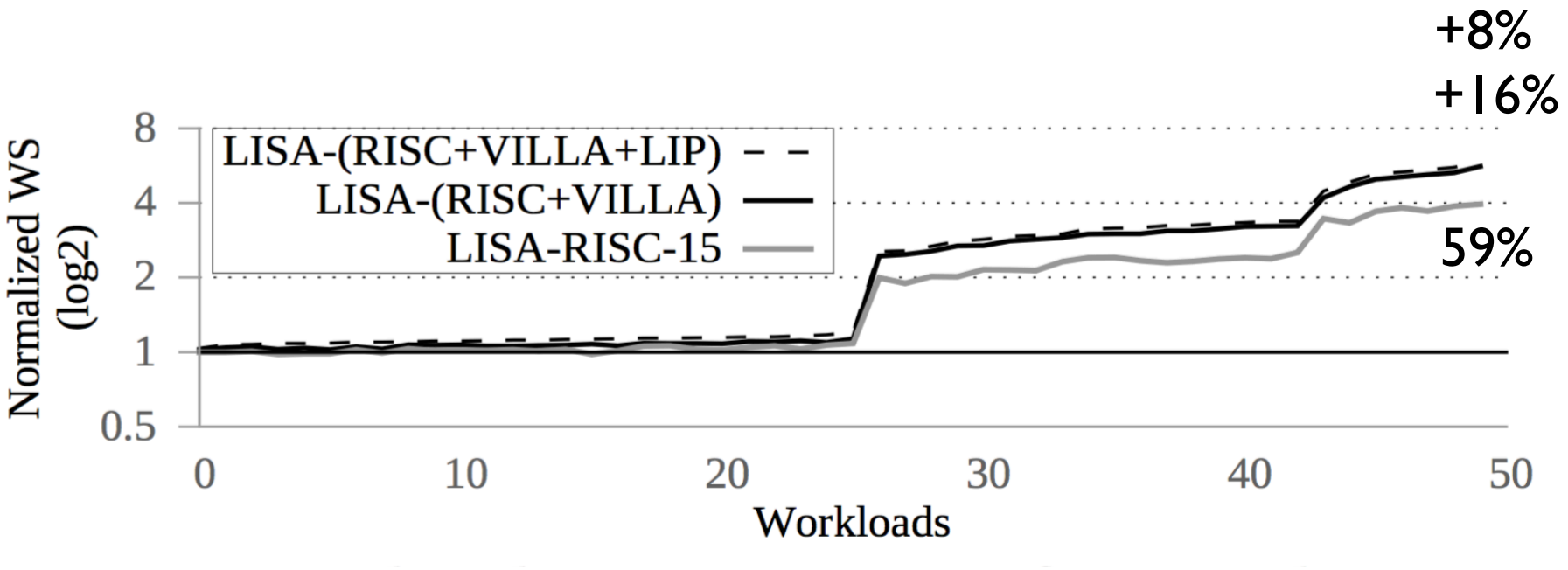
Sensitivity of Cache Size

Single core: RISC vs. baseline as LLC size changes



- Baseline: higher cache pollution as LLC size decreases
- Forkbench
  - RISC: Hit rate – 67% (4MB) to 10% (256KB)
  - Base: Hit rate – 20% to 19%

Combined Applications



Sensitivity to Copy Distance

Copy Distance (hops)	1	3	7	15	31	63
RISC Copy Latency (ns)	148.5	164.5	196.5	260.5	388.5	644.5
WS Improvement (%)	66.2	65.3	63.3	59.6	53.0	42.4
DRAM Energy Savings (%)	55.4	55.2	54.6	53.6	51.9	48.9

Table 4: Effect of copy distance on LISA-RISC.

VILLA Caching Policy

- Benefit-based caching policy [HPCA'13]
  - A benefit counter to track # of accesses per cached row
- Any caching policy can be applied to VILLA

- Configuration
  - 32 rows inside a fast subarray
  - 4 fast subarrays per bank
  - 1.6% area overhead

Area Measurement

- *Row-Buffer Decoupling* by O et al., ISCA'14
- 28nm DRAM process,
  - 3 metal layers
  - 8Gb and 8 banks per device

Other slides

Low-Cost Inter-Linked Subarrays (LISA)

Enabling Fast Inter-Subarray Data Movement in DRAM

Kevin Chang

Prashant Nair, Donghyuk Lee, Saugata Ghose,

Moinuddin Qureshi, and Onur Mutlu

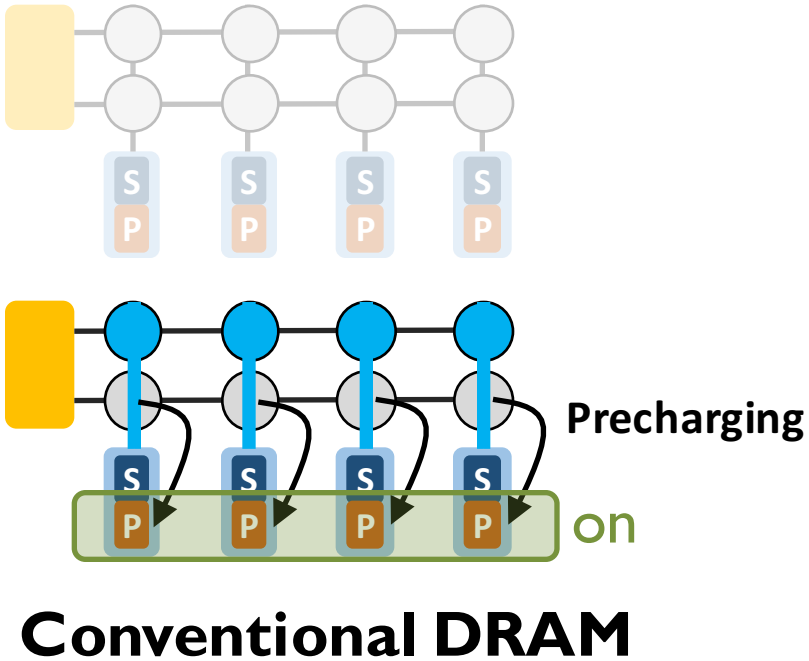
SAFARI

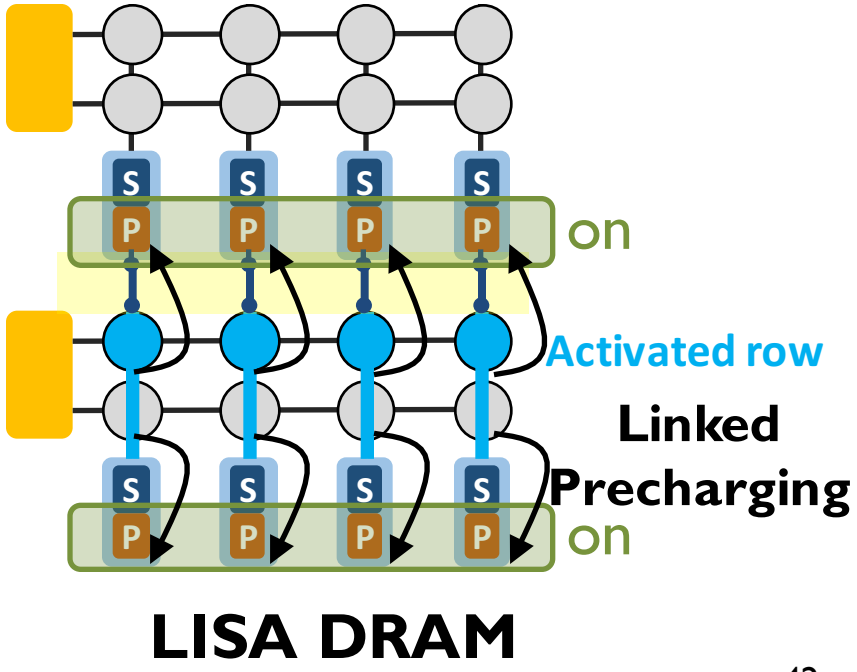
Carnegie Mellon University

Georgia Institute of Technology

3. Linked Precharge (LIP)

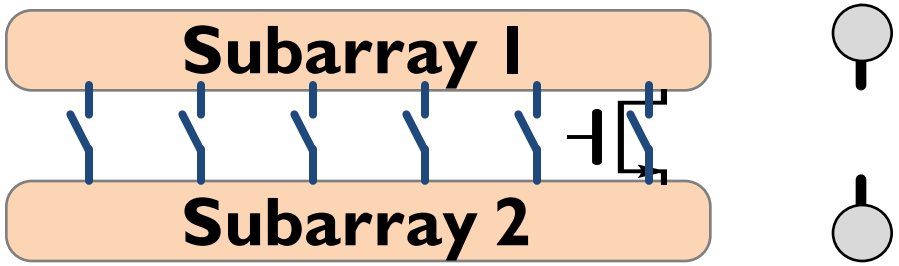
- **Problem**: The precharge time is limited by the strength of one precharge unit (PU)
- **Linked Precharge (LIP)**: LISA's connectivity enables DRAM to utilize additional PUs from other subarrays





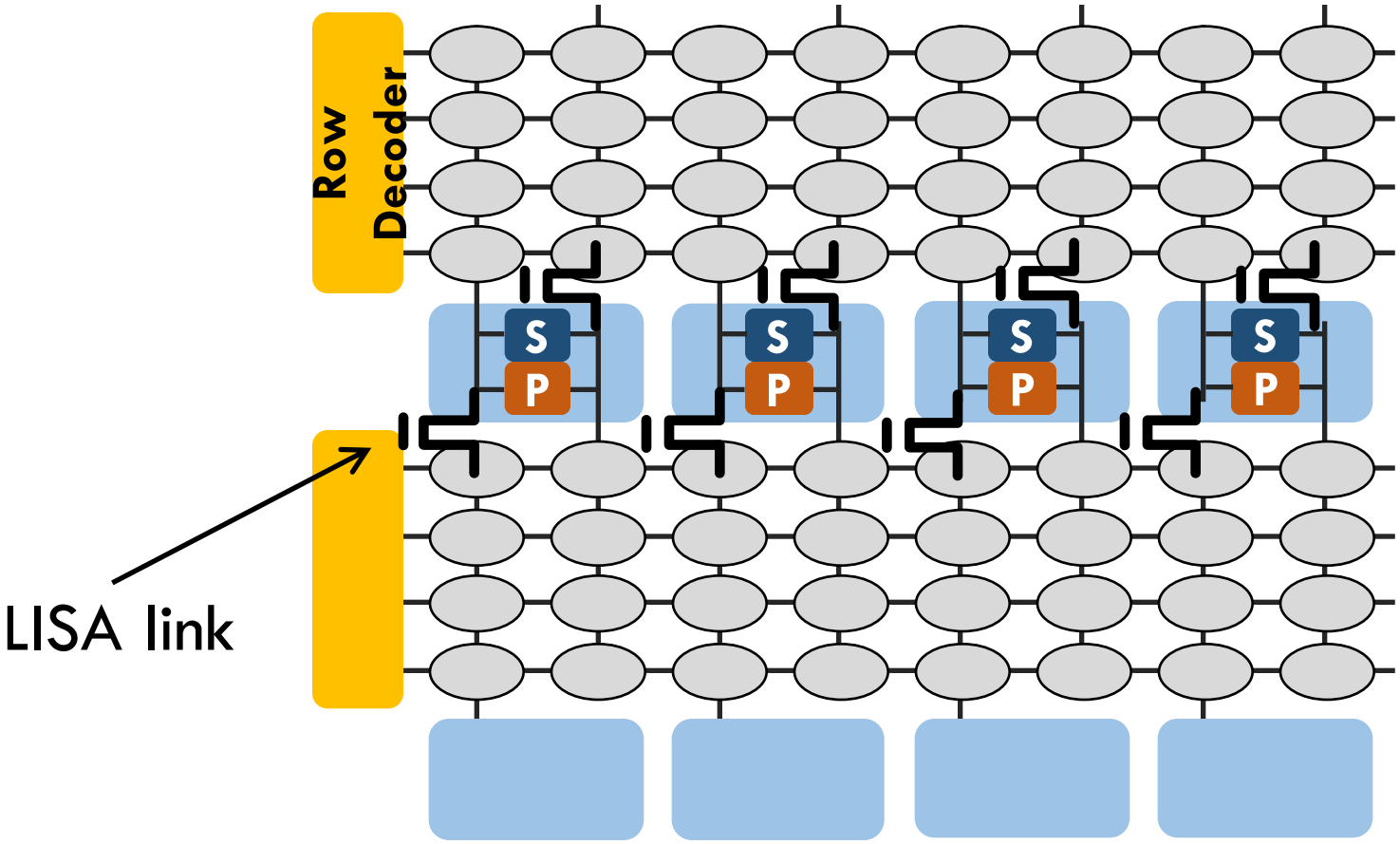
Key Idea and Applications

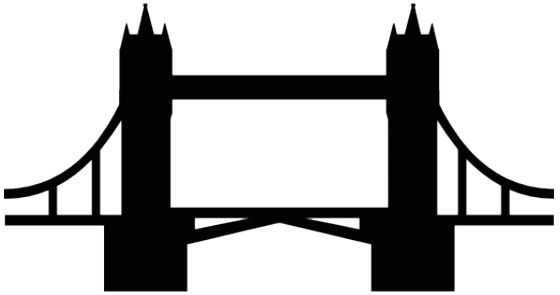
- **Low-cost Inter-linked subarrays (LISA)**
  - Fast bulk data movement b/w subarrays
  - Wide datapath via isolation transistors: 0.8% DRAM chip area



- LISA is a **versatile substrate** → new applications
  1. Fast bulk data copy: Copy latency 1.3ms→0.1ms (9x)
     ↑ 66% sys. performance and 55% energy efficiency
  2. In-DRAM caching: Access latency 48ns→21ns (2x)
     ↑ 5% sys. performance
  3. Linked precharge: Precharge latency 13ns→5ns (2x)
     ↑ 8% sys. performance

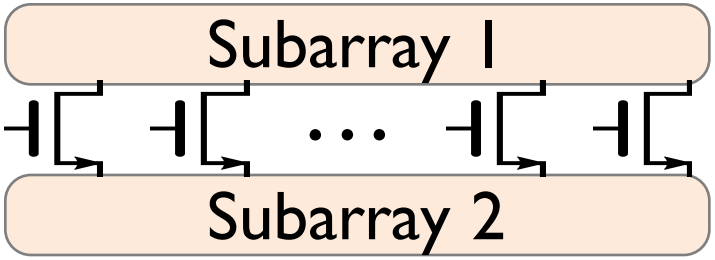
Low-Cost Inter-Linked Subarrays (LISA)





Key Idea and Applications

- **Low-cost Inter-linked subarrays (LISA)**
  - Fast bulk data movement b/w subarrays
  - Wide datapath via isolation transistors: 0.8% DRAM chip area



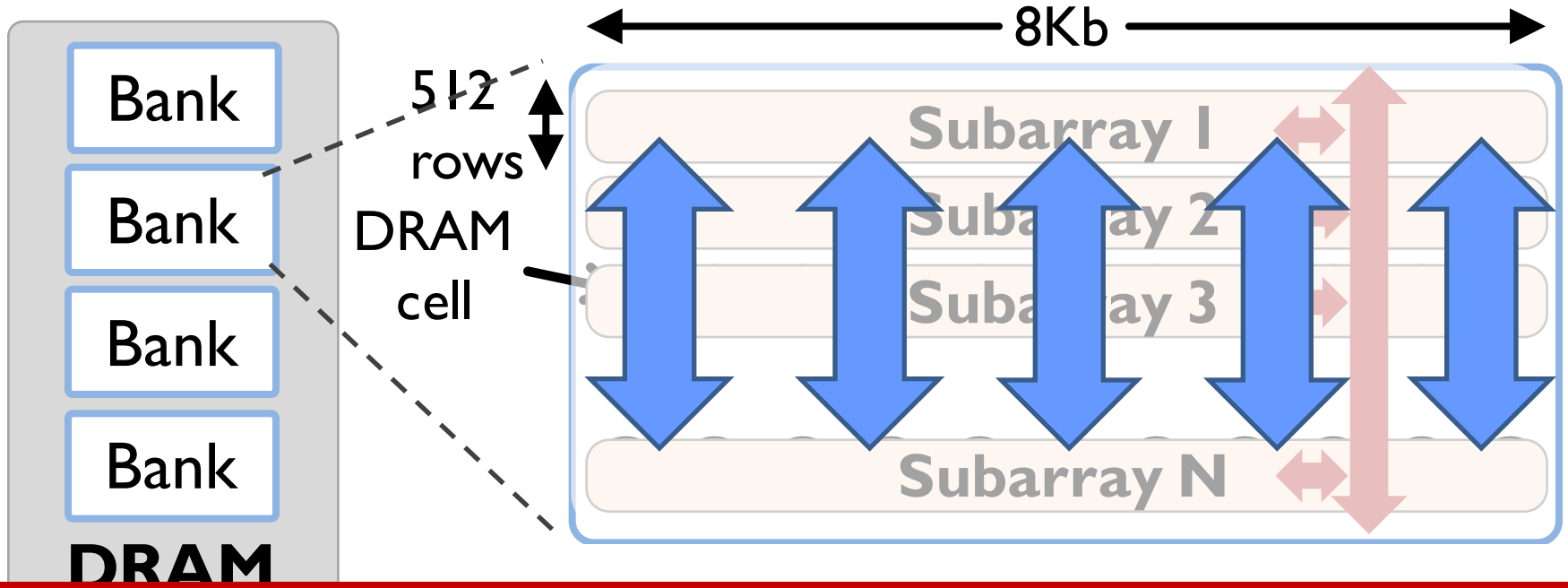
- LISA is a **versatile substrate** → new applications

  Fast bulk data copy: Copy latency 1.363ms→0.148ms (9.2x)
  +66% speedup and -55% DRAM energy efficiency

  In-DRAM caching: Hot data access latency 48.7ns→21.5ns (2.2x)
  ↑ 5% sys. performance

  Fast precharge: Precharge latency 13.1ns→5ns (2.6x)
  ↑ 8% sys. performance

Moving Data Inside DRAM?

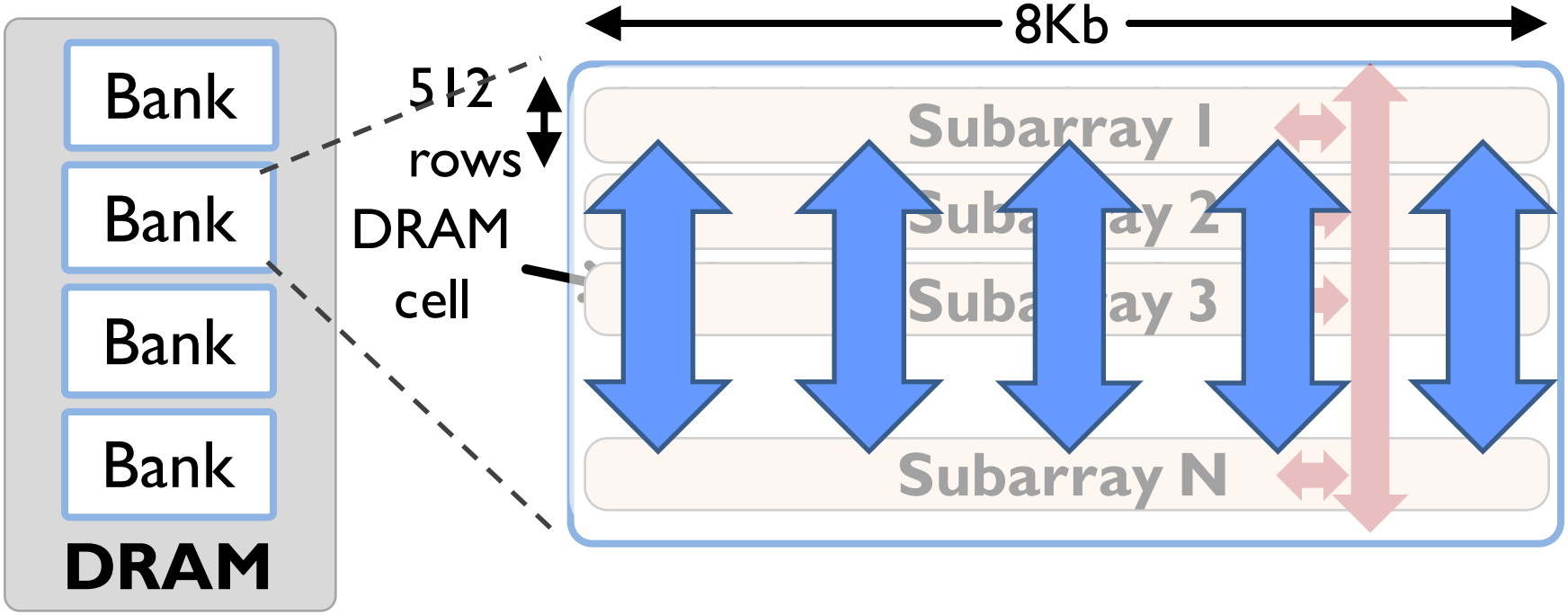


Low connectivity in DRAM is the fundamental bottleneck for bulk data movement

Goal: Provide a new substrate to enable wide connectivity between subarrays

Low Connectivity in DRAM

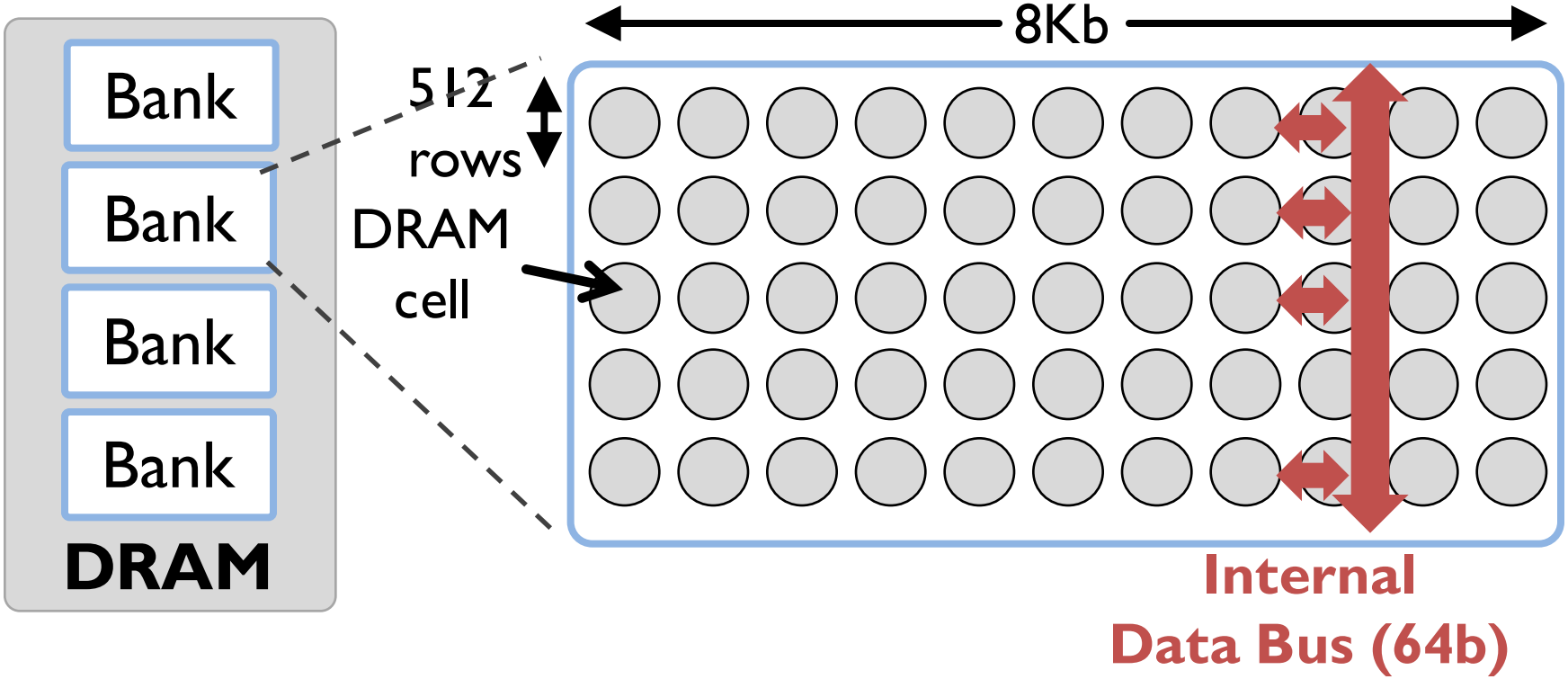
Problem: Simply moving data inside DRAM is inefficient



Goal: Provide a new substrate to enable wide connectivity b/w subarrays

Low Connectivity in DRAM

Problem: Simply moving data inside DRAM is inefficient



Low connectivity in DRAM is the fundamental bottleneck for bulk data movement