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
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]
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]
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]
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]
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?
Moving Data Inside DRAM?
Moving Data Inside DRAM?

Bank
Bank
Bank
Bank

DRAM

Bank

DRAM cell

...
Moving Data Inside DRAM?

Bank
Bank
Bank
Bank

DRAM

Subarray 1
Subarray 2
Subarray 3
\vdots
Subarray N
Moving Data Inside DRAM?

Bank
Bank
Bank
Bank

DRAM

Subarray 1
Subarray 2
Subarray 3
...
Subarray N

512 rows

8Kb
Moving Data Inside DRAM?

Bank
Bank
Bank
Bank
DRAM

512 rows

Subarray 1
Subarray 2
Subarray 3
...
Subarray N

Internal Data Bus (64b)

8Kb
Low connectivity in DRAM is the fundamental bottleneck for bulk data movement.
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
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

Subarray 1

Subarray 2
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

![Diagram of Subarrays](Image)
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

```
Subarray 1
...
Subarray 2
```

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

![Diagram of Subarray 1 and Subarray 2 with arrows indicating data flow]

• **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

Subarray
DRAM Internals

Subarray

Row Decoder

Wordline
DRAM Internals

Subarray

Row Decoder

Bitline

Wordline
DRAM Internals

Subarray

Bitline

Wordline

Row Decoder

Sense amplifier
DRAM Internals

Subarray

Bitline

Wordline

Row Decoder

Sense amplifier

Precharge unit
DRAM Internals

Subarray

Bitline

Wordline

Row Buffer

S Sense amplifier

P Precharge unit
DRAM Internals

Subarray

Bitline

Wordline

Row Decoder

Row Buffer

S Sense amplifier

P Precharge unit

Subarray

512 x 8Kb
DRAM Internals

Subarray

Row Buffer

Sense amplifier S

Precharge unit P

Bitline

Wordline

Row Decoder

Subarray

512 x 8Kb

Internal Data Bus

64b

I/O
DRAM Internals

Subarray

Bitline

Row Decoder

Wordline

Row Buffer

Sense amplifier

Precharge unit

Bank (16~64 SAs)

Internal Data Bus

512 x 8Kb

I/O

64b
DRAM Internals

**Subarray**

- **Bitline**
- **Wordline**
- **Row Buffer**
  - **Sense amplifier**
  - **Precharge unit**

**Bank** (16~64 SAs)
8~16 banks per chip
To Bank I/O

Bitline Voltage Level: $V_{dd}/2$
To Bank I/O

1 ACTIVATE: Store the row into the row buffer

Bitline Voltage Level: $V_{dd}/2$
1 **ACTIVATE**: Store the row into the **row buffer**

**Bitline Voltage Level**: $V_{dd}/2$
DRAM Operation

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

Bitline Voltage Level: $V_{dd}$
**DRAM Operation**

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

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

**Bitline Voltage Level**: $V_{dd}$
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**

**Bitline Voltage Level**: $V_{dd}$
**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

**Bitline Voltage Level**: $V_{dd}$
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

Bitline Voltage Level: $V_{dd}$
**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**

**Bitline Voltage Level**: $V_{dd}/2$
Outline

• Motivation and Key Idea
• DRAM Background
• **LISA Substrate**
  – **New DRAM Command to Use LISA**
• Applications of LISA
Observations

Internal Data Bus (64b)
1 Bitlines serve as a bus that is as wide as a row
Observations

Bitlines serve as a bus that is as wide as a row
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)
Low-Cost Interlinked Subarrays (LISA)
Low-Cost Interlinked Subarrays (LISA)

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

64b
Low-Cost Interlinked Subarrays (LISA)

Interconnect bitlines of adjacent subarrays in a bank using isolation transistors (links)
Low-Cost Interlinked Subarrays (LISA)

Interconnect bitlines of adjacent subarrays in a bank using isolation transistors (links)
Low-Cost Interlinked Subarrays (LISA)

Interconnect bitlines of adjacent subarrays in a bank using isolation transistors (links)
Low-Cost Interlinked Subarrays (LISA)

LISA forms a wide datapath b/w subarrays

Interconnect bitlines of adjacent subarrays in a bank using isolation transistors (links)
New DRAM Command to Use LISA

Row Buffer Movement (RBM): Move a row of data in an activated row buffer to a precharged one
New DRAM Command to Use LISA

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

Subarray 1

Subarray 2
New DRAM Command to Use LISA

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

Subarray 1

Activated

Subarray 2

Precharged
New DRAM Command to Use LISA

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

Subarray 1

Activated

Subarray 2

Precharged

V_{dd}

V_{dd}/2
New DRAM Command to Use LISA

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

Subarray 1

Activated

RBM: SA1 → SA2

Subarray 2

Precharged
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: SA1 → SA2

Subarray 1
Activated

Subarray 2
Precharged

V_{dd}

V_{dd}/2
New DRAM Command to Use LISA

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

Subarray 1

Activated

RBM: SA1 → SA2

Subarray 2

Precharged

Charge Sharing

\[ V_{dd} \]

\[ V_{dd}/2 \]
New DRAM Command to Use LISA

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

Subarray 1

Activated

RBM: SA1→SA2

Subarray 2

Precharged

Charge Sharing

V_{dd}-\Delta

V_{dd}/2+\Delta
New DRAM Command to Use LISA

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

Subarray 1 → Subarray 2

Amplify the charge

$v_{dd} - \Delta$

$v_{dd}/2 + \Delta$
New DRAM Command to Use LISA

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

Subarray 1

Subarray 2

RBM: SA1 → SA2

Activated

Activated

Amplify the charge

\[ V_{dd} \]

\[ V_{dd}/2 \]
New DRAM Command to Use LISA

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

Subarray 1

Activated

RBM: SA1 → SA2

Subarray 2

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
RBM Analysis

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

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

[Diagram showing subarrays 1, 2, and 3 with arrows indicating data movement]
RBM Analysis

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

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

  ![Subarray Diagram]

• Validated with SPICE using worst-case cells
  – NCSU FreePDK 45nm library
**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
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
1. Rapid Inter-Subarray Copying (RISC)

- **Goal:** Efficiently copy a row across subarrays
- **Key idea:** Use RBM to form a new command sequence
1. Rapid Inter-Subarray Copying (RISC)

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

1. **Activate** src row

![Diagram showing subarray activation and copying process](image-url)
1. Rapid Inter-Subarray Copying (RISC)

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

1. **Activate src row**

2. **RBM SA1 → SA2**
1. Rapid Inter-Subarray Copying (RISC)

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

1. **Activate src row**

2. **RBM SA1 → SA2**

3. **Activate dst row**
   (write row buffer into dst row)
1. Rapid Inter-Subarray Copying (RISC)

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

1. Activate *src* row

2. RBM *SA1* → *SA2*

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
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
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)
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

WS Improvement

-24%

DRAM Energy Reduction

5%
System Evaluation: RISC

WS Improvement

-24% Degrades bank-level parallelism

DRAM Energy Reduction

5%

Over Baseline (%)
System Evaluation: RISC

- 66% WS Improvement
- 55% DRAM Energy Reduction

RowClone

- 24% Degradation of bank-level parallelism
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)

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

**Long Bitline (DDRx)**
2. Variable Latency DRAM (VILLA)

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

![Diagram of Long Bitline (DDRx) and Short Bitline (RLDRAM)]
2. Variable Latency DRAM (VILLA)

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

![Diagram showing Long Bitline (DDRx) and Short Bitline (RLDRAM)]

- Shorter bitlines $\rightarrow$ faster activate and precharge time
2. Variable Latency DRAM (VILLA)

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

![Diagram showing Long Bitline (DDRx) and Short Bitline (RLDRAM)]

- Shorter bitlines → faster activate and precharge time
- High area overhead: >40%
2. Variable Latency DRAM (VILLA)

- **Key idea**: Reduce access latency of hot data via a heterogeneous DRAM design [Lee+ HPCA’13, Son+ ISCA’13]
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
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

![Diagram of VILLA architecture](image-url)
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
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

**Challenges**:
- VILLA cache requires frequent movement of data rows
- **LISA**: Cache rows rapidly from slow to fast subarrays
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

VILLA Cache Hit Rate

Workloads (50)

Normalized Speedup

VILLA

---VILLA Cache Hit Rate

VILLA Cache Hit Rate (%)
System Evaluation: VILLA

50 quad-core workloads: memory-intensive benchmarks

Average VILLA Cache Hit Rate: 5%
Maximal VILLA Cache Hit Rate: 16%

Normalized Speedup vs. Workloads (50)
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
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

![Diagram of precharge process]

Conventional DRAM
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

![Diagram comparing Conventional DRAM and LISA DRAM in Linked Precharge (LIP)]
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
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

**Conventional DRAM** vs **LISA DRAM**

- Precharging process is highlighted differently in each diagram.
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

LIP

Normalized Speedup

Avg: 8%

Max: 13%
50 quad-core workloads: memory-intensive benchmarks

Accelerating precharge using LISA improves system performance
Other Results in Paper

• Combined applications
Other Results in Paper

- Combined applications
- Single-core results
Other Results in Paper

• Combined applications
• Single-core results
• Sensitivity results
  – LLC size
  – Number of channels
  – Copy distance
Other Results in Paper

• Combined applications
• Single-core results
• Sensitivity results
  – LLC size
  – Number of channels
  – Copy distance
• Qualitative comparison to other hetero. DRAM
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
Summary

• Bulk data movement is inefficient in today’s systems
• Low connectivity between subarrays is a bottleneck
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
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
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
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
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