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

Abdullah Giray Yağlıkçi
Ataberk Olgun  Minesh Patel  Haocong Luo  Hasan Hassan
Lois Orosa  Oğuz Ergin  Onur Mutlu

SAFARI

ETH Zürich  CESGA  TOBB ETÜ
Executive Summary

• **Problem:** DRAM Refresh
  - is a *fundamental operation* to avoid bit flips due to *leakage* and *RowHammer*
  - incurs *increasingly large performance overhead* with DRAM chip *density scaling*

• **Goal:** Reduce the *performance overhead* of DRAM Refresh

• **Key Idea:** *Hide refresh latency* by *refreshing* a DRAM row *concurrently with activating* another row in a *different subarray* of the *same bank*

• **HiRA:** Hidden Row Activation – a new DRAM operation that
  - Issues *DRAM commands* in *quick succession* to concurrently open two rows in *different subarrays*
  - Works on *real off-the-shelf DRAM chips* by violating timing constraints
  - *Significantly reduces* (51.4%) the time spent for refresh operations

• **HiRA-MC:** HiRA Memory Controller – a new mechanism
  - Leverages HiRA to perform *refresh requests* *concurrently with DRAM accesses* and other *refresh requests*
  - *Significantly improves* system performance by *hiding refresh latency* for both *regular periodic* and *RowHammer-preventive* refreshes
Outline

Background and Problem

Goal and Key Idea

HiRA: Hidden Row Activation

HiRA in Real DRAM Chips

HiRA-MC: HiRA Memory Controller

Performance Evaluation

Conclusion
DRAM Organization

- DRAM Chip
- Bank
- Chip I/O
- Subarray
- DRAM Bank
- DRAM Row
- Bitline
- Wordline
- Row Buffer
- DRAM Subarray
- DRAM Rank

SAFARI
DRAM Operations

1. **ACTIVATE (ACT):** Fetch the row’s content into the *row buffer*

2. **Column Access (RD/WR):** Read/Write the target column and drive to I/O

3. **PRECHARGE (PRE):** Prepare the array for a new ACTIVATE
DRAM Refresh is the key maintenance operation to avoid bit flips due to charge leakage.

DRAM Refresh activates a row and precharges the bank.

Problem: DRAM Refresh blocks accesses to the whole bank / rank.
Two Main Types of DRAM Refresh

1. **Periodic Refresh**: Periodically restores the charge of DRAM cells that leak 
   over time.

2. **RowHammer**: Repeatedly accessing a DRAM row can cause bit flips in other physically nearby rows.

   - **Preventive Refresh**: Mitigates RowHammer by refreshing physically nearby rows of a repeatedly accessed row.
Periodic Refresh with Increasing DRAM Chip Density

A larger capacity chip has more rows to be refreshed.

A smaller cell stores less charge.

More periodic refresh operations incur larger performance overhead as DRAM chip density increases.
RowHammer and Preventive Refresh with Increasing DRAM Chip Density

RowHammer vulnerability worsens as DRAM chip density increases

Preventive refresh operations need to be performed more aggressively as DRAM chip density increases
Outline

Background and Problem

Goal and Key Idea

HiRA: Hidden Row Activation

HiRA in Real DRAM Chips

HiRA-MC: HiRA Memory Controller

Performance Evaluation

Conclusion
Our Goal

Reduce the **performance overhead** of DRAM Refresh (both **periodic** and **preventive**).
Key Idea

Hide refresh latency by refreshing a DRAM row concurrently with activating another row in a different subarray of the same bank.
Outline

Background and Problem

Goal and Key Idea

HiRA: Hidden Row Activation

HiRA in Real DRAM Chips

HiRA-MC: HiRA Memory Controller

Performance Evaluation

Conclusion
HiRA: Hidden Row Activation – Key Insight

Activating two rows in **quick succession** that are in **different subarrays** in the **same bank** can **refresh one row** concurrently with activating the other row.

[Diagram showing subarrays X and Y in a DRAM bank with ACT signals to Row A and Row B]
HiRA: Hidden Row Activation

Refresh **RowA** concurrently with **Activating RowB**

Without HiRA:
- **ACT RowA**
- **PRE**
- **ACT RowB**
- **RD**

With HiRA:
- **PRE**
- **ACT RowA**
- **ACT RowB**
- **RD**

**Saved time using HiRA**
Outline

Background and Problem

Goal and Key Idea

HiRA: Hidden Row Activation

HiRA in Real DRAM Chips

HiRA-MC: HiRA Memory Controller

Performance Evaluation

Conclusion
DRAM Testing Infrastructure

FPGA-based SoftMC (Xilinx Virtex UltraScale+ XCU200)

Xilinx Alveo U200 FPGA Board (programmed with SoftMC*)

DRAM Module with Heaters

PCle Host Interface

MaxWell FT200 Temperature Controller

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

*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-SAFAIR/SoftMC]
HiRA in Off-the-Shelf DRAM Chips: Key Result 1

- HiRA works in **56 off-the-shelf DRAM chips** from SK Hynix

| Module Label | Module Vendor | Module Identifier | Freq (MT/s) | Date Code | Chip Cap. | Die Rev. | Chip Org. | HiRA Coverage | Norm. $N_{RH}$ |
|--------------|---------------|-------------------|-------------|-----------|-----------|----------|-----------|---------------|---------------|
| A0           | G.SKILL       | DWCW (Partial Marking)*  | 2400        | 42-20     | 4Gb       | B        | x8        | 24.8%         | 1.75          |
| A1           |               | F4-2400C17S-8GNT [39] |             |           |           |          |           | 24.9%         | 1.72          |
| B0           | Kingston      | H5AN8G8NDJR-XNC KSM32RD8/16HDR [87] | 2400        | 48-20     | 4Gb       | D        | x8        | 25.1%         | 1.71          |
| B1           |               |                   |             |           |           |          |           | 25.0%         | 1.74          |
| C0           | SK Hynix      | H5ANAG8NAJR-XN HMAA4GU6AJR8N-XN [109] | 2400        | 51-20     | 4Gb       | F        | x8        | 25.3%         | 1.47          |
| C1           |               |                   |             |           |           |          |           | 29.2%         | 1.09          |
| C2           |               |                   |             |           |           |          |           | 26.5%         | 1.49          |

* The chip identifier is partially removed on these modules. We infer the chip manufacturer and die revision based on the remaining part of the chip identifier.

- HiRA performs a given row’s **refresh concurrently with activating** any of the **32% of the rows** in the same bank
HiRA in Off-the-Shelf DRAM Chips: Key Result 2

- **HiRA works in 56 off-the-shelf DRAM chips** from **SK Hynix**

| Module Label | Module Vendor | Module Identifier (Chip Identifier) | Freq (MT/s) | Date Code | Chip Cap. | Die Rev. | Chip Org. | HiRA Coverage | Norm. $N_{RH}$ |
|--------------|---------------|------------------------------------|------------|-----------|----------|----------|-----------|----------------|----------------|
| A0/A1        | G.SKILL       | DWCW (Partial Marking)* F4-2400C17S-8GNT [39] | 2400       | 42-20     | 4Gb      | B        | x8        | 24.8% 24.9% 25.0% 25.5% | 1.75 1.72 1.90 2.52 |
| B0/B1        | Kingston      | H5AN8G8NJR-XNC KSM32RD8/16HDR [87] | 2400       | 48-20     | 4Gb      | D        | x8        | 25.1% 25.0% 32.6% 36.8% | 1.71 1.74 1.89 2.34 |
| C0/C1/C2     | SK Hynix      | H5ANAG8NAJR-XN HMAA4GU6AJR8N-XN [109] | 2400       | 51-20     | 4Gb      | F        | x8        | 25.3% 29.2% 35.3% 39.5% | 1.47 1.09 1.89 2.23 |

* The chip identifier is partially removed on these modules. We infer the chip manufacturer and die revision based on the remaining part of the chip identifier.

- **51.4% reduction** in the time spent for refresh operations

**HiRA effectively reduces the time spent for refresh operations in off-the-shelf DRAM chips**
HiRA in Off-the-Shelf DRAM Chips: Key Results

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

A. Giray Yağlıkçı¹, Ataberk Olgun¹, Minesh Patel¹, Haocng Luo¹, Hasan Hassan¹
Lois Orosa¹,³, Oğuz Ergin², Onur Mutlu¹

¹ETH Zürich  ²TOBB University of Economics and Technology  ³Galicia Supercomputing Center (CESGA)

DRAM is the building block of modern main memory systems. DRAM cells must be periodically refreshed to prevent data loss. Refresh operations degrade system performance by interfering with memory accesses. As DRAM chip density increases with technology node scaling, refresh operations also increase because: 1) the number of DRAM rows in a chip increases; and 2) DRAM cells need additional refresh operations to mitigate bit failures caused by RowHammer, a failure mechanism that becomes worse with technology node scaling. Thus, it is critical to enable refresh operations at low performance overhead.

To this end, we propose a new operation, Hidden Row Activation (HiRA), and the HiRA Memory Controller (HiRA-MC) to perform HiRA operations. As DRAM density increases with technology node scaling, the performance overhead of refresh also increases due to three major reasons. First, as the DRAM chip density increases, more DRAM rows need to be periodically refreshed in a DRAM chip [55, 57–61]. Second, as DRAM technology node scales down, DRAM cells become smaller and thus can store less amount of charge, requiring them to be refreshed more frequently [10, 20, 67, 102, 103, 118, 122–124]. Third, with increasing DRAM density, DRAM cells are placed closer to each other, exacerbating charge leakage via a disturbance error mechanism called RowHammer [79, 84, 119, 120, 133, 134, 167, 180, 183], and thus requiring additional refresh operations (called preventive refreshes) to avoid data corruption due to RowHammer.

https://arxiv.org/pdf/2209.10198.pdf
# Outline

1. Background and Problem
2. Goal and Key Idea
3. HiRA: Hidden Row Activation
4. HiRA in Real DRAM Chips
5. **HiRA-MC: HiRA Memory Controller**
6. Performance Evaluation
7. Conclusion
**HiRA-MC: HiRA Memory Controller**

- **Goal**: Leverage HiRA’s parallelism as much as possible

- **Key Insight**: A time slack is needed to find a row activation and a refresh to perform HiRA

---

Ref(RowA) is generated → ACT RowX → ACT RowY → ACT RowZ → ACT RowT → Ref(RowA) deadline

*RowA and RowZ are in two electrically disconnected subarrays*
HiRA-MC: HiRA Memory Controller

1. Generates each periodic refresh and RowHammer-preventive refresh with a deadline.

2. Buffers each refresh request and performs the refresh request until the deadline.

3. Finds if it can refresh a DRAM row concurrently with a DRAM access or another refresh.
HiRA-MC: HiRA Memory Controller

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

A. Giray Yağlıkçı¹  Ataberk Olgun¹  Minesh Patel¹  Haocng Luo¹  Hasan Hassan¹
Lois Orosa¹,³  Oğuz Ergin²  Onur Mutlu¹
¹ETH Zürich  ²TOBB University of Economics and Technology  ³Galicia Supercomputing Center (CESGA)

DRAM is the building block of modern main memory systems. DRAM cells must be periodically refreshed to prevent data loss. Refresh operations degrade system performance by interfering with memory accesses. As DRAM chip density increases with technology node scaling, refresh operations also increase because: 1) the number of DRAM rows in a chip increases; and 2) DRAM cells need additional refresh operations to mitigate bit failures caused by RowHammer, a failure mechanism that becomes worse with technology node scaling. Thus, it is critical to enable refresh operations at low performance overhead. To this end, we propose a new operation, Hidden Row Activation (HiRA), and the HiRA Memory Controller (HiRA-MC) to perform HiRA operations.

As DRAM density increases with technology node scaling, the performance overhead of refresh also increases due to three major reasons. First, as the DRAM chip density increases, more DRAM rows need to be periodically refreshed in a DRAM chip [55, 57–61]. Second, as DRAM technology node scales down, DRAM cells become smaller and thus can store less amount of charge, requiring them to be refreshed more frequently [10, 20, 67, 102, 103, 118, 122–124]. Third, with increasing DRAM density, DRAM cells are placed closer to each other, exacerbating charge leakage via a disturbance error mechanism called RowHammer [79, 84, 119, 120, 133, 134, 167, 180, 183], and thus requiring additional refresh operations (called preventive refreshes) to avoid data corruption due to RowHam-

https://arxiv.org/pdf/2209.10198.pdf
# Performance Evaluation

- **Cycle-level simulations using Ramulator** [Kim+, CAL 2015]

## System Configuration:

| Component                | Specification                                                                 |
|--------------------------|-------------------------------------------------------------------------------|
| Processor                | 3.2 GHz, 8 core, 4-wide issue, 128-entry instr. window                       |
| Last-Level Cache         | 64-byte cache line, 8-way set-associative, 8 MB                              |
| Memory Scheduler         | FR-FCFS                                                                       |
| Address Mapping          | Minimalistic Open Pages                                                       |
| Main Memory              | DDR4, 4 bank group, 4 banks per bank group (16 banks per rank)               |
| Timing Parameters        | $t_1 = t_2 = 3\text{ns}$, $t_{RC} = 46.25\text{ns}$, $t_{FAW} = 16\text{ns}$ |

- **Workloads:** 125 different **8-core** multiprogrammed workloads from the SPEC2006 benchmark suite

- **DRAM Chip Capacity:** \{2, 4, 8, 16, 32, 64, 128\} Gb

- **RowHammer Threshold:** \{1024, 512, 256, 128, 64\} activations
  
  The **minimum number of row activations** needed to induce the first **RowHammer bit flip**
HiRA for Periodic Refreshes

- **No-Refresh**: No periodic refresh is performed (Ideal case)
- **Baseline**: Auto-Refresh (using conventional REF commands)

Periodic refreshes cause **significant (26%) performance overhead**

HiRA improves system performance by **12.6%** over the baseline

- **No Refresh**
- **Baseline**
- **HiRA**
HiRA for Preventive Refreshes

- **No Defense**: No RowHammer mitigation employed (i.e., no preventive refresh)
- **PARA [Kim+, ISCA'14]**: the RowHammer defense with the **lowest hardware overhead**

![Bar chart showing speedup and slowdown for No-Defense, PARA, and HiRA with varying RowHammer thresholds]

- **96% slowdown**
- **3.7x speedup**

**PARA significantly reduces (by 96%) system performance**

**HiRA improves system performance by 3.7x over PARA**
More in the Full Paper

• **Real DRAM Chip Experiments**
  - Verification of **HiRA's functionality**
  - **Variation** in HiRA's characteristics **across banks**

• Sensitivity to
  - length of **time slack** for refreshes
  - **number of channels**
  - **number of ranks**

• **Hardware Complexity Analysis**
  - Chip **area cost of 0.0023%** of a processor die per DRAM rank
  - **No additional latency** overhead

• **Experimental Methodology**
  - **Detailed algorithms** for each set of **real chip** experiments
  - Extensive **security analysis** for RowHammer-preventive refreshes

• **Detailed Algorithm of Finding Concurrent Refreshes**
HiRA: Hidden Row Activation for Reducing Refresh Latency of Off-the-Shelf DRAM Chips

A. Giray Yağlıkçı¹ Ataberk Olgün¹ Minesh Patel¹ Haocong Luo¹ Hasan Hassan¹
Lois Orosa¹,³ Oğuz Ergin² Onur Mutlu¹

¹ETH Zürich ²TOBB University of Economics and Technology ³Galicia Supercomputing Center (CESGA)

DRAM is the building block of modern main memory systems. DRAM cells must be periodically refreshed to prevent data loss. Refresh operations degrade system performance by interfering with memory accesses. As DRAM chip density increases with technology node scaling, refresh operations also increase because: 1) the number of DRAM rows in a chip increases; and 2) DRAM cells need additional refresh operations to mitigate bit failures caused by RowHammer, a failure mechanism that becomes worse with technology node scaling. Thus, it is critical to enable refresh operations at low performance overhead. To this end, we propose a new operation, Hidden Row Activation (HiRA), and the HiRA Memory Controller (HiRA-MC) to perform HiRA operations.

As DRAM density increases with technology node scaling, the performance overhead of refresh also increases due to three major reasons. First, as the DRAM chip density increases, more DRAM rows need to be periodically refreshed in a DRAM chip [55, 57–61]. Second, as DRAM technology node scales down, DRAM cells become smaller and thus can store less amount of charge, requiring them to be refreshed more frequently [10, 20, 67, 102, 103, 118, 122–124]. Third, with increasing DRAM density, DRAM cells are placed closer to each other, exacerbating charge leakage via a disturbance error mechanism called RowHammer [79, 84, 119, 120, 133, 134, 167, 180, 183], and thus requiring additional refresh operations (called preventive refreshes) to avoid data corruption due to RowHam-

https://arxiv.org/pdf/2209.10198.pdf
Conclusion

• **HiRA**: Hidden Row Activation – a new DRAM operation
  - First technique that refreshes a DRAM row concurrently with activating another row in the same bank in off-the-shelf DRAM chips
  - Real DRAM chip experiments:
    • HiRA works on 56 real off-the-shelf DRAM chips
    • 51.4% reduction in the time spent for refresh operations

• **HiRA-MC**: HiRA Memory Controller – a new mechanism
  - Leverages HiRA to perform refresh requests concurrently with DRAM accesses and other refresh requests
  - HiRA-MC provides:
    • 12.6% speedup by hiding periodic refresh latency
    • 3.7x speedup by hiding RowHammer-preventive refresh latency
HiRA: Hidden Row Activation for Reducing Refresh Latency of Off-the-Shelf DRAM Chips

Abdullah Giray Yağılıkçı
Ataberk Olgun  Minesh Patel  Haocong Luo  Hasan Hassan
Lois Orosa  Oğuz Ergin  Onur Mutlu

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

Backup Slides

Abdullah Giray Yağlıkçı
Ataberk Olgun  Minesh Patel  Haocong Luo  Hasan Hassan
Lois Orosa  Oğuz Ergin  Onur Mutlu

SAFARI

ETH zürich  CESGA  TOBB ETÜ
The RowHammer Vulnerability

Repeatedly **opening** (activating) and **closing** (precharging) a DRAM row in **real DRAM chips** causes **RowHammer bit flips** in nearby cells.
Activating a DRAM row refreshes the row and prevents RowHammer bit flips.
Mitigating RowHammer

ACT ➔ Row 0
ACT ➔ Row 1
ACT ➔ Row 2
ACT ➔ Row 3
ACT ➔ Row 4

⚠️ Row 2 is being hammered

Refresh neighbor rows

Preventive Refresh
Activating potential victim rows \textbf{mitigate RowHammer} by refreshing them
RowHammer and Preventive Refresh

• **RowHammer**: Repeatedly accessing a DRAM row can *cause bit flips* in other *physically nearby rows*.

• **Preventive Refresh**: Refresh a DRAM row when a physically nearby row is activated based on *activation counts* or *probabilistic processes*.

Preventive refresh mitigates RowHammer bit flips.
HiRA: Hidden Row Activation

• HiRA **concurrently activates** two rows in a DRAM bank
  - **Challenge 1**: Only one row can be activated in a DRAM bank at a given time
  - **Solution 1**: HiRA **violates timing constraints** for concurrent row activations

• HiRA issues **two row activation (ACT)** commands **in quick succession**
  - **Challenge 2**: DRAM chips **ignore the second** activation before precharge
  - **Solution 2**: HiRA issues a **precharge (PRE)** command between two ACTs

• HiRA activates **two DRAM rows** in the **same bank**
  - **Challenge 3**: The two rows can **override** each other’s data **via shared bitlines**
  - **Solution 3**: HiRA uses rows from two **electrically disconnected subarrays**

HiRA **violates DRAM timing constraints** by issuing a sequence of **ACT-PRE-ACT** commands that target two rows in two **electrically disconnected subarrays**
HiRA: Hidden Row Activation

Refreshing RowA concurrently with Activating RowB

The time saved using HiRA

Reduction in the time spent for two refreshes
HiRA Operation

HiRA refreshes RowA concurrently with activating RowB by issuing \textit{ACT-PRE-ACT} commands in quick succession.
HiRA in Off-the-Shelf DRAM Chips: Key Results

• HiRA works in 56 off-the-shelf DRAM chips from SK Hynix

• 51.4% reduction in the time spent for refresh operations

• HiRA performs a given row’s refresh concurrently with activating any of the 32% of the rows in the same bank

HiRA effectively reduces the time spent for refresh operations in off-the-shelf DRAM chips
HiRA Support in Off-the-Shelf DRAM Chips

• 56 off-the-shelf DDR4 DRAM chips support HiRA (from SK Hynix)

• HiRA Coverage of a given DRAM row:
  - Refresh a given DRAM row while activating other rows in the same bank
  - We sweep two timing parameters: $t_1$ and $t_2$

HiRA can refresh a DRAM row concurrently with 32% of any of the other DRAM rows in the same bank
HiRA’s Second Row Activation

- Does performing HiRA in between refresh the victim row?
  - If HiRA’s second row activation is performed, more activations are needed to induce RowHammer bit flips.
  - If HiRA’s second row activation is ignored, RowHammer threshold should not change.

![Graphs showing absolute and normalized RowHammer thresholds with and without HiRA activation.](image)
Variation across DRAM Banks

• Coverage: Identical across banks
• The effect of second row activation
HiRA-MC: HiRA Memory Controller

• **Goal**: Leverage HiRA’s parallelism as much as possible

• **Periodic** and **preventive** refresh controllers generate each refresh request **with a deadline**

• **Refresh Table** buffers a refresh request until its **deadline**

• **Concurrent Refresh Finder** finds if HiRA can refresh a row
  - **Concurrently with a memory request**
  - **Concurrently with another refresh request**
The Concurrent Refresh Finder

Case 1: Executes when a precharge is issued (completes before the precharge completes)

Case 2: Periodically executes after every $t_{RC}$ (completes before $t_{RC}$)
HiRA-MC Example

• Case 1: **Refresh – Access** Parallelism

  Memory Request Queue
  
  Refresh Table
  
  HiRA(Row SA:B Row:6, SA:A Row:0)

  
  Head
  
  RD SA:B Row:0
  
  RD SA:A Row:0

  REF SA:C Row:2
  
  REF SA:B Row:1
  
  REF SA:B Row:6

  ACT SA:B Row:6
  
  ACT SA:A Row:0

  PRE

  6ns

• Case 2: **Refresh – Refresh** Parallelism

  Memory Request Queue
  
  Refresh Table
  
  HiRA(Row SA:B Row:1, SA:C Row:2)

  
  Head
  
  RD SA:B Row:0
  
  REF SA:B Row:1

  REF SA:B Row:0

  REF SA:B Row:6

  ACT SA:B Row:1
  
  ACT SA:C Row:2

  PRE

  6ns

HiRA-MC provides **refresh-access** and **refresh-refresh** parallelism
HiRA-MC Hardware Complexity

• We use CACTI with 22nm technology node

| HiRA-MC Component       | Area (mm$^2$) | Area (% of Chip Area) | Access Latency |
|-------------------------|---------------|------------------------|----------------|
| Refresh Table           | 0.00031       | <0.0001%               | 0.07ns         |
| RefPtr Table            | 0.00683       | 0.0017%                | 0.12ns         |
| PR-FIFO                 | 0.00029       | <0.0001%               | 0.07ns         |
| Subarray Pairs Table    | 0.00180       | 0.0005%                | 0.09ns         |
| **Overall**             | **0.00923**   | **0.0023%**            | **6.31ns**     |

HiRA-MC consumes only **0.0023%** of CPU chip area per DRAM rank

HiRA-MC does not increase memory access latency

Precharge latency: ~14.5ns

HiRA-MC Overall Latency: 6.31ns

time

PRE

ACT
Estimating Periodic Refresh Overhead

\[ t_{RFC} = 110 \times C_{chip}^{0.6} \]

- Latency of a REF command
- DRAM Chip Capacity

2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture

Nonblocking Memory Refresh

Kate Nguyen, Kehan Lyu, Xianze Meng  
Department of Computer Science  
Virginia Tech  
Blacksburg, Virginia  
katevy@vt.edu, kehan@vt.edu, xianze@vt.edu

Vilas Sridharan  
RAS Architecture  
Advanced Micro Devices, Inc  
Boxborough, Massachusetts  
vilas.sridharan@amd.com

Xun Jian  
Department of Computer Science  
Virginia Tech  
Blacksburg, Virginia  
xunj@vt.edu
Reducing Overall Latency of Two Refreshes

• Refreshing two rows using nominal timing parameters:

- $t_{RAS}: 32\text{ns}$
- $t_{RP}: 14.25\text{ns}$
- $t_{RAS}: 32\text{ns}$

• Using HiRA:

- $t_1: 3\text{ns}$
- $t_2: 3\text{ns}$
- $t_{RAS}: 32\text{ns}$

Overall latency of refreshing two rows reduces by 51.4% from 78.25ns down to 38ns
# Tested DRAM Chips

Table 4: Characteristics of the tested DDR4 DRAM modules.

| Module Label | Module Vendor | Module Identifier Chip Identifier | Freq (MT/s) | Date Code | Chip Cap. | Die Rev. | Chip Org. | HiRA Coverage Min. | HiRA Coverage Avg. | HiRA Coverage Max. | Norm. $N_{RH}$ Min. | Norm. $N_{RH}$ Avg. | Norm. $N_{RH}$ Max. |
|--------------|---------------|-----------------------------------|------------|-----------|-----------|---------|----------|---------------------|---------------------|---------------------|-------------------|-------------------|-------------------|
| A0           | G.SKILL       | DWCW (Partial Marking)* F4-2400C17S-8GNT [39] | 2400       | 42-20     | 4Gb       | B       | x8       | 24.8%               | 25.0%               | 25.5%               | 1.75              | 1.90              | 2.52              |
| A1           |               |                                   |            |           |           |         |          | 24.9%               | 26.6%               | 28.3%               | 1.72              | 1.94              | 2.55              |
| B0           | Kingston      | H5AN8G8NDJR-XNC KSM32RD8/16HDR [87] | 2400       | 48-20     | 4Gb       | D       | x8       | 25.1%               | 32.6%               | 36.8%               | 1.71              | 1.89              | 2.34              |
| B1           |               |                                   |            |           |           |         |          | 25.0%               | 31.6%               | 34.9%               | 1.74              | 1.91              | 2.51              |
| C0           | SK Hynix      | H5ANAG8NAJR-XN HMMA4GU6AJR8N-XN [109] | 2400       | 51-20     | 4Gb       | F       | x8       | 25.3%               | 35.3%               | 39.5%               | 1.47              | 1.89              | 2.23              |
| C1           |               |                                   |            |           |           |         |          | 29.2%               | 38.4%               | 49.9%               | 1.09              | 1.88              | 2.27              |
| C2           |               |                                   |            |           |           |         |          | 26.5%               | 36.1%               | 42.3%               | 1.49              | 1.96              | 2.58              |

* The chip identifier is partially removed on these modules. We infer the chip manufacturer and die revision based on the remaining part of the chip identifier.

[https://arxiv.org/pdf/2209.10198.pdf](https://arxiv.org/pdf/2209.10198.pdf)
HiRA-MC: HiRA Memory Controller

- **Periodic** and **preventive** refresh controllers generate each refresh request **with a deadline**
- **Refresh Table** buffers a refresh request **until its deadline**
- **Concurrent Refresh Finder** finds if HiRA can refresh a row
  - **Concurrently with a DRAM access**
  - **Concurrently with another refresh request**
HiRA for Periodic Refreshes

a) HiRA's perf. overhead, compared to No Refresh

b) HiRA's perf. improvement compared to Baseline
RowHammer Thresholds

a) PARA's probability threshold ($p_{th}$) for different values of $N_{RH}$ and $t_{RefSlack}$

b) Overall RowHammer success probability for different values of $N_{RH}$ and $t_{RefSlack}$
HiRA for Preventive Refreshes

a) PARA's perf. overhead with and without HiRA

b) HiRA's perf. improvement compared to PARA

- Para  
- HiRA-2  
- HiRA-4  
- HiRA-8
HiRA for Periodic Refresh
Sensitivity to Number of Channels and Ranks
HiRA for Preventive Refresh
Sensitivity to Number of Channels and Ranks
Workload Memory Access Characteristics

• 125 different 8-core multiprogrammed workloads

• Three histograms showing MPKI, RBCPKI, and RBHPKI respectively
RowHammer Mitigation across Generations

J. S. Kim, M. Patel, A. G. Yaglikci, H. Hassan, R. Azizi, L. Orosa, and O. Mutlu, "Revisiting RowHammer: An Experimental Analysis of Modern Devices and Mitigation Techniques," in ISCA, 2020.
Refresh Delay

- DDRx protocols allow a REF command to be postponed for ~70us

- HiRA-MC’s current design does not leverage this flexibility

- A longer time slack allows
  - the baseline to better utilize DRAM idle time to perform refresh operations
  - HiRA to find more opportunities to perform a refresh operation concurrently with a DRAM access

- Future sensitivity study: the effect of long refresh delays

SAFARI
Energy

• HiRA does not change the number of refresh operations at a given time window
  - Overall energy consumed for refresh operations is the same

• HiRA improves system performance
  - Reduces the background energy consumption

• Evaluation requires an accurate power model based on real system measurements, similar to VAMPIRE [Ghose+ SIGMETRICS’17], but for HiRA operations
HiRA: Hidden Row Activation for Reducing Refresh Latency of Off-the-Shelf DRAM Chips

Abdullah Giray Yağlıkçı
Ataberk Olgun  Minesh Patel  Haocong Luo  Hasan Hassan
Lois Orosa  Oğuz Ergin  Onur Mutlu

SAFARI

ETHzürich  CESGA  TOBB ETÜ