FIGARO: Improving System Performance via Fine-Grained In-DRAM Data Relocation and Caching

Yaohua Wang¹, Lois Orosa², Xiangjun Peng³, Yang Guo¹, Saugata Ghose⁴,⁵, Minesh Patel², Jeremie S. Kim², Juan Gómez Luna², Mohammad Sadrosadati⁶, Nika Mansouri Ghiasi², Onur Mutlu²,⁵

1 2 3 4 5 6
Executive Summary

- **Problem**: DRAM latency is a performance bottleneck for many applications
- **Goal**: Reduce DRAM latency via in-DRAM cache
- **Existing in-DRAM caches**:
  - Augment DRAM with **small-but-fast regions** to implement caches
  - Coarse-grained (i.e., multi-kB) in-DRAM data relocation
  - Relocation latency increases with physical distance between slow and fast regions
- **FIGARO Substrate**:
  - **Key idea**: use the existing shared global row buffer among subarrays within a DRAM bank to provide support for in-DRAM data relocation
  - Fine-grained (i.e., multi-byte) in-DRAM data relocation and distance-independent relocation latency
  - Avoids complex modifications to DRAM by using (mostly) existing structures
- **FIGCache**:
  - **Key idea**: cache only small, frequently-accessed portions of different DRAM rows in a designated region of DRAM
  - Caches only the parts of each row that are expected to be accessed in the near future
  - Increases row hits by packing frequently-accessed row segments into FIGCache
  - Improves system performance by 16.3% on average
  - Reduces energy consumption by 7.8% on average
- **Conclusion**:
  - FIGARO enables fine-grained data relocation in-DRAM at low cost
  - FIGCache outperforms state-of-the-art coarse-grained in-DRAM caches
Outline

Background

Existing In-DRAM Cache Designs
FIGARO Substrate
FIGCache: Fine-Grained In-DRAM Cache
Experimental Methodology
Evaluation
Conclusion
DRAM Organization
Outline

Background

Existing In-DRAM Cache Designs

FIGARO Substrate

FIGCache: Fine-Grained In-DRAM Cache

Experimental Methodology

Evaluation

Conclusion
Inefficiencies of In-DRAM Caches

1) Coarse-grained:
Caching an entire row at a time hinders the potential of in-DRAM cache

2) Area overhead and complexity:
Many fast subarrays interleaved among normal subarrays
# Outline

<table>
<thead>
<tr>
<th>Paragraph</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
</tr>
<tr>
<td>Existing In-DRAM Cache Designs</td>
</tr>
<tr>
<td>FIGARO Substrate</td>
</tr>
<tr>
<td>FIGCache: Fine-Grained In-DRAM Cache</td>
</tr>
<tr>
<td>Experimental Methodology</td>
</tr>
<tr>
<td>Evaluation</td>
</tr>
<tr>
<td>Conclusion</td>
</tr>
</tbody>
</table>

**SAFARI**
Observations and Key Idea

- **Observations:**

  1) All local row buffers (LRBs) in a bank are connected to a single shared global row buffer (GRB)

  2) The GRB has smaller width (e.g., 8B) than the LRBs (e.g., 1kB)

- **Key Idea:** use the existing shared GRB among subarrays within a DRAM bank to perform fine-grained in-DRAM data relocation
FIGARO Overview

FIGARO: Fine-Grained In-DRAM Data Relocation Substrate

- Relocates data across subarrays within a bank
- Column granularity within a chip
- Cache-block granularity within a rank
Key Features of FIGARO

- **Fine-grained**: column/cache-block level data relocation

- **Distance-independent latency**
  - The relocation latency depends on the length of global bitline
  - Similar to the latency of read/write commands

- **Low overhead**
  - Additional column address MUX, row address MUX, and row address latch per subarray
  - 0.3% DRAM chip area overhead

- **Low latency and low energy consumption**
  - Low latency (63.5ns) to relocate one column
    - Two ACTIVATEs, one RELOC, and one PRECHARGE commands
  - Low energy consumption (0.03uJ) to relocate one column
<table>
<thead>
<tr>
<th>Outline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
</tr>
<tr>
<td>Existing In-DRAM Cache Designs</td>
</tr>
<tr>
<td>FIGARO Substrate</td>
</tr>
<tr>
<td><strong>FIGCache: Fine-Grained In-DRAM Cache</strong></td>
</tr>
<tr>
<td>Experimental Methodology</td>
</tr>
<tr>
<td>Evaluation</td>
</tr>
<tr>
<td>Conclusion</td>
</tr>
</tbody>
</table>
FIGCache Overview

- **Key idea:** Cache only small, frequently-accessed portions of different DRAM rows in a designated region of DRAM

- **FIGCache** (Fine-Grained In-DRAM Cache)
  - Uses FIGARO to **relocate** data into and out of the cache at the fine **granularity** of a row segment
  - Avoids the need for a **large number of fast (yet low capacity) subarrays** interleaved among slow subarrays
  - Increases row buffer **hit rate**

- **FIGCache Tag Store (FTS)**
  - Stores information about which row segments are currently cached
  - Placed in the memory controller

- **FIGCache In-DRAM Cache Designs**
  - Using 1) fast subarrays, 2) slow subarrays, or 3) fast rows in a subarray
Benefits of FIGCache

- Fine-grained (cache-block) caching granularity
- Low area overhead and manufacturing complexity
## Outline

<table>
<thead>
<tr>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
</tr>
<tr>
<td>Existing In-DRAM Cache Designs</td>
</tr>
<tr>
<td>FIGARO Substrate</td>
</tr>
<tr>
<td>FIGCache: Fine-Grained In-DRAM Cache</td>
</tr>
<tr>
<td><strong>Experimental Methodology</strong></td>
</tr>
<tr>
<td>Evaluation</td>
</tr>
<tr>
<td>Conclusion</td>
</tr>
</tbody>
</table>
Experimental Methodology

- **Simulator**
  - Ramulator open-source DRAM simulator [Kim+, CAL’15] [https://github.com/CMU-SAFARI/ramulator]
  - 8 cores, DRAM DDR4 800MHz bus frequency

- **Workloads**
  - 20 eight-core multiprogrammed workloads from SPEC CPU2006, TPC, BioBench, Memory Scheduling Championship

- **Comparison points**
  - Baseline: conventional DDR4 DRAM
  - LISA-VILLA: State-of-the-art in-DRAM Cache
  - FIGCache-slow: Our in-DRAM cache with cache rows stored in slow subarrays
  - FIGCache-fast: Our in-DRAM cache with cache rows stored in fast subarrays
  - FIGCache-ideal: An unrealistic version of FIGCache-Fast where the row segment relocation latency is zero
  - LL-DRAM: System where all subarrays are fast
Outline

1. Background
2. Existing In-DRAM Cache Designs
3. FIGARO Substrate
4. FIGCache: Fine-Grained In-DRAM Cache
5. Experimental Methodology
6. Evaluation
7. Conclusion
The benefits of FIGCache-Fast and FIGCache-Slow increase as workload memory intensity increases.

Both FIGCache-slow and FIGCache-fast outperform LISA-VILLA.

FIGCache-Fast approaches the ideal performance improvement of both FIGCache-Ideal and LL-DRAM.
FIGARO: Improving System Performance via Fine-Grained In-DRAM Data Relocation and Caching

Yaohua Wang¹, Lois Orosa², Xiangjun Peng³,¹, Yang Guo¹, Saugata Ghose⁴,⁵, Minesh Patel², Jeremie S. Kim², Juan Gómez Luna², Mohammad Sadrosadati⁶, Nika Mansourí Ghiasi², Onur Mutlu²,⁵

¹ 2 3 4

ETH Zürich  The Chinese University of Hong Kong University of Illinois Urbana-Champaign

Carnegie Mellon University

SAFARI

MICRO 2020