Improving DRAM Performance by Parallelizing Refreshes with Accesses

Kevin Chang†, Donghyuk Lee†, Zeshan Chisti§, Alaa Alameldeen§, Chris Wilkerson§, Yoongu Kim†, Onur Mutlu†
†Carnegie Mellon University, §Intel Labs

Problem
- **DRAM refresh** interferes with memory accesses, degrading system performance and energy efficiency
- **Goal**: Serve memory accesses in parallel with refreshes to reduce refresh interference on demand requests

Background and Motivation
- Memory controllers send **periodic refreshes** to DRAM ranks
  - tReflatenzy (TRFC): Varies based on DRAM chip density (e.g., 350ns)
  - tRefPeriod (TREFI): Remains constant
- 6.7%/23%/41% throughput loss for 4/32/64Gb DRAM
- **Two existing refresh modes**:
  - **All-Bank Refresh**: Employed in commodity DRAM (DDRx, LPDDRx)
  - **Per-Bank Refresh**: In mobile DRAM (LPDDRx)

  ![Timeline](image)

- **Shortcomings of per-bank refresh**:
  1) Per-bank refreshes are **strictly scheduled** in a **static round-robin order**
  2) A refreshing bank **cannot serve memory accesses**

  ![Can serve memory accesses in parallel with refreshes across banks](image)

Results
- **Methodology**
  - 8 OoO CPU cores
  - Caches: L1 – 32KB, Shared L2 – 4MB
  - DRAM: DDR3-1333, 64-bit channel, channels/ranks/banks = 2/2/8
  - Workloads: SPEC CPU2006, STREAM, TPC-C/H, random access

  ![System Performance](image)

  ![Energy Consumption](image)

  Consistent system performance improvement across DRAM densities (within 0.9%, 1.2%, and 3.8% of ideal)

  Consistent energy reduction

- **Our Solutions**
  1. **Dynamic Access Refresh Parallelization (DARP)**:
     - Improved scheduling policy for per-bank refreshes
     - **Component 1**: Out-of-order per-bank refresh
       - Schedule per-bank refreshes to idle banks opportunistically in a **dynamic order**

       ![Baseline: Round robin](image)

       ![Our mechanism: DARP](image)

     - **Component 2**: Write-refresh parallelization
       - Avoids refresh interference on latency-critical reads by refreshing with writes
       - Proactively schedules refreshes when banks are serving buffered writes

  2. **Subarray Access Refresh Parallelization (SARP)**:
     - Parallelizes refreshes and accesses within a bank

     ![Write-refresh parallelization](image)

- **0.71% DRAM die area overhead**

*Please read our paper in HPCA 2014 for more results