DRAM REFRESH MANAGEMENT

Mahdi Nazm Bojnordi
Assistant Professor
School of Computing
University of Utah
Overview

- Upcoming deadline
  - Tonight: homework assignment will be posted

- This lecture
  - DRAM address mapping
  - DRAM refresh basics
  - Smart refresh
  - Elastic refresh
  - Avoiding or pausing refreshes
Where to store cache lines in main memory?

Typical Mapping

- Row
- Bank
- Column
- Block

Application A:

Good distribution of memory requests among DRAM banks.
Where to store cache lines in main memory?

Typical Mapping

Application B:
Unbalanced distribution of memory requests among DRAM banks.
How to compute bank ID?

Custom Mapping

Row Bank Row Column Block

Application B:
Good distribution of memory requests among DRAM banks.
Spatial locality is not well preserved!
Page Interleaving

Address format

<table>
<thead>
<tr>
<th>r</th>
<th>k</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>page index</td>
<td>bank</td>
<td>page offset</td>
</tr>
</tbody>
</table>
Cache Line Mapping

- Bank index is a subset of set index

### Cache line interleaving
- Page index
- Page offset
- Bank
- Page offset

### Page interleaving
- Page index
- Bank
- Page offset

### Cache-related representation
- Cache tag
- Cache set index
- Block offset
Problem: interleaving load and writeback streams with the same access pattern to the banks may result in row buffer misses.

Load

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x+b</td>
<td>x+2b</td>
</tr>
<tr>
<td></td>
<td>x+3b</td>
<td></td>
</tr>
</tbody>
</table>

Writeback

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>y+b</td>
<td>y+2b</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

the same row buffer
Key Issues

- To exploit spatial locality, use maximal interleaving granularity (or row-buffer size)
- To reduce row buffer conflicts, use only those bits in cache set index for “bank bits”

![Diagram of memory addresses]
Permutation-based Interleaving

L2 Cache tag

index  bank  page offset

k  XOR  k  k

page index  new bank  page offset

[Zhang’00]
Permutation-based Interleaving

- New bank index

L2 Conflicting addresses

1000 1010
1001 1010
1010 1010
1011 1010

Permutation-based interleaving

Different bank indexes

Conventional interleaving

Same bank indexes

Memory banks

0000
0001
0010
0011
0100
0101
0110
0111
1010
1011

[Zhang'00]
Permutation-based Interleaving

![Bar chart showing IPC improvement for various benchmarks: tomcatv, swim, su2cor, hydro2d, mgrid, applu, turb3d, wave5, TPC-C. Colors correspond to different mechanisms: cacheline, page, swap, permutation.](Zhang'00)
DRAM Refresh

- DRAM cells lose charge over time
- Periodic refresh operations are required to avoid data loss
- Two main strategies for refreshing DRAM cells
  - **Burst refresh:** refresh all of the cells each time
    - Simple control mechanism (e.g., LPDDRx)
  - **Distributed refresh:** a group of cells are refreshed
    - Avoid blocking memory for a long time
Refresh Basics

- **tRET**: the retention time of DRAM leaky cells (64ms)
  - All cells must be refreshed within tRET to avoid data loss

- **tREFI**: refresh interval, which is the gap between two refresh commands issues by the memory controller
  - MC sends 8192 auto-refresh commands to refresh one bin at a time
    - \( tREFI = \frac{tRET}{8192} = 7.8\text{us} \)

- **tRFC**: the time to finish refreshing a bin (refresh completion)

- What is the bin size?
Refresh Basics

- tRFC increases with chip capacity

![Impact of chip density on refresh completion time](image)

[Stuecheli’10]
Controlling Refresh Operations

- CAS before RAS (CBR)
  - DRAM memory keeps track of the addresses using an internal counter

- RAS only refresh (ROR)
  - Row address is specified by the controller; similar to a pair of activate and precharge

- Auto-refresh vs. self refresh
  - Every 7.8us a REF command is sent to DRAM (tRAS+tRP)
  - LPDDR turns off IO for saving power while refreshing multiple rows
Refresh Granularity

- All bank vs. per bank refresh

(a) All-bank refresh ($REF_{ab}$) frequency and granularity.

(b) Per-bank refresh ($REF_{pb}$) frequency and granularity.
Optimizing DRAM Refresh

- Observation: each row may be accessed as soon as it is to be refreshed
Smart Refresh

- Idea: avoid refreshing recently accessed rows

Figure 5: Smart Refresh Control Schematic

[Ghosh‘07]
Diverse Impacts of Refresh

Worst Case Refresh Hit DRAM Read

<table>
<thead>
<tr>
<th>DRAM capacity</th>
<th>tRFC</th>
<th>bandwidth overhead (95°C per Rank)</th>
<th>latency overhead (95°C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>512Mb</td>
<td>90ns</td>
<td>2.7%</td>
<td>1.4ns</td>
</tr>
<tr>
<td>1Gb</td>
<td>110ns</td>
<td>3.3%</td>
<td>2.1ns</td>
</tr>
<tr>
<td>2Gb</td>
<td>160ns</td>
<td>5.0%</td>
<td>4.9ns</td>
</tr>
<tr>
<td>4Gb</td>
<td>300ns</td>
<td>7.7%</td>
<td>11.5ns</td>
</tr>
<tr>
<td>8Gb</td>
<td>350ns</td>
<td>9.0%</td>
<td>15.7ns</td>
</tr>
</tbody>
</table>

[Stuecheli’10]
Elastic Refresh

- Send refreshes during periods of inactivity
- Non-uniform request distribution
- Refresh overhead just has to fit in free cycles
- Initially not aggressive, converges with delay until empty (DUE) as refresh backlog grows
- Latency sensitive workloads are often lower bandwidth
- Decrease the probability of reads conflicting with refreshes

[Stuecheli'10]
Elastic Refresh

- Introduce refresh backlog dependent idle threshold
- With a log backlog, there is no reason to send refresh command
- With a bursty request stream, the probability of a future request decreases with time
- As backlog grows, decrease this delay threshold

Key: to reduce REF and READ conflicts

[Stuecheli’10]
If software is able to tolerate errors, we can lower DRAM refresh rates to achieve considerable power savings.
Flikker

- Divide memory bank into high refresh part and low refresh parts
- Size of high-refresh portion can be configured at runtime
- Small modification of the Partial Array Self-Refresh (PASR) mode

![Flikker DRAM Bank Diagram]

[Song’14]
Refresh Pausing

Baseline system

Request B arrives

Pausing at arbitrary point can cause data loss

Pausing Refresh reduces wait time for Reads
Performance Results

Performance Comparison

Refresh Pausing  No Refresh

Speedup

COMMERICAL  SPEC  PARSEC  BIOBENCH  GMEAN