DRAM POWER MANAGEMENT

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

- Upcoming deadline
  - April 4th: sign up for your student paper presentation

- This lecture
  - DRAM power components
  - DRAM refresh management
  - DRAM power optimization
DRAM Power Consumption

- DRAM is a significant contributor to the overall system power/energy consumption

**Bulk Power Breakdown:**
(midrange server)

- Processors
- Memory
- IO
- Interconnect chips
- Cooling
- Misc

*IBM data, from WETI 2012 talk by P. Bose*
A significant portion of the DRAM energy is consumed as IO and background.

**DDR4 DRAM Power Breakdown**

1. Reduce Refreshes
2. Reduce IO energy
3. Reduce precharges
4. ...

[data from Seol’2016]
If software is able to tolerate errors, we can lower DRAM refresh rates to achieve considerable power savings.
Critical vs. Non-critical Data

- Important for application correctness e.g., meta-data, key data structures
  - High refresh: No errors
  - Low refresh: Some errors

- Does not substantially impact application correctness e.g., multimedia data, soft state

Flikker DRAM

Mobile applications have substantial amounts of non-critical data that can be easily identified by application developers.
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

High Refresh

Low Refresh

1

3/4

1/2

1/4

1/8

[Song’14]
Power Reduction

- Up to 25% reduction in DRAM power

**Standby DRAM Power Reduction**

- conservative
- aggressive
- crazy

**Overall DRAM Power Reduction**

- conservative
- aggressive
- crazy

[Song’14]
Quality of the Results

original

degraded (52.0dB)

[Song’14]
Refresh Energy Overhead

% DRAM energy spent refreshing

Device capacity

Present

Future

2 Gb 4 Gb 8 Gb 16 Gb 32 Gb 64 Gb

15% 47%

[Liu’2012]
Conventional Refresh

- Today: Every row is refreshed at the same rate

- Observation: Most rows can be refreshed much less often without losing data [Kim+, EDL’09]

- Problem: No support in DRAM for different refresh rates per row

[Liu’2012]
Observation: Only very few rows need to be refreshed at the worst-case rate.

Can we exploit this to reduce refresh operations at low cost?

[Liu’2012]
Reducing DRAM Refresh Operations

- **Idea:** Identify the retention time of different rows and refresh each row at the frequency it needs to be refreshed

- **(Cost-conscious) Idea:** Bin the rows according to their minimum retention times and refresh rows in each bin at the refresh rate specified for the bin
  - e.g., a bin for 64-128ms, another for 128-256ms, …

- **Observation:** Only very few rows need to be refreshed very frequently [64-128ms] → Have only a few bins → Low HW overhead to achieve large reductions in refresh operations

[Liu’2012]
RAIDR Results

- DRAM power reduction: 16.1%
- System performance improvement: 8.6%

[Liu’2012]
Limit Activate Power

- Refresh timings

- Limit the power consumption
DRAM Power Management

- DRAM chips have power modes
- Idea: **When not accessing a chip power it down**

- Power states
  - Active (highest power)
  - All banks idle
  - Power-down
  - Self-refresh (lowest power)

- State transitions incur latency during which the chip cannot be accessed
Queue-aware Power-down

1. Read/Write instructions are queued in a stack

2. Scheduler (AHB) decides which instruction is preferred

3. Subsequently instructions are transferred into FIFO Memory Queue
Queue-aware Power-down

1. Rank counter is zero -> rank is idle
   &
2. The rank status bit is 0 -> rank is not yet in a low power mode
   &
3. There is no command in the CAQ with the same rank number -> avoids powering down if a access of that rank is immanent

<table>
<thead>
<tr>
<th>Read/Write Queue</th>
</tr>
</thead>
<tbody>
<tr>
<td>C:1 - R:2 – B:1 – 0 - 1</td>
</tr>
<tr>
<td>C:1 - R:2 – B:1 – 0 - 2</td>
</tr>
<tr>
<td>C:1 - R:2 – B:1 – 0 - 3</td>
</tr>
<tr>
<td>C:1 - R:2 – B:1 – 0 - 4</td>
</tr>
<tr>
<td>C:1 - R:2 – B:1 – 0 - 5</td>
</tr>
<tr>
<td>C:1 - R:2 – B:1 – 0 - 6</td>
</tr>
<tr>
<td>C:1 - R:2 – B:1 – 0 - 7</td>
</tr>
<tr>
<td>C:1 - R:1 – B:1 – 0 - 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set rank1 counter to 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decrement counter for rank 2</td>
</tr>
<tr>
<td>Set rank2 status bit to 8</td>
</tr>
<tr>
<td>Decrement counter for rank 1</td>
</tr>
<tr>
<td>Set rank2 status bit to 8</td>
</tr>
<tr>
<td>Decrement counter for rank 1</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>Set rank2 status bit to 8</td>
</tr>
<tr>
<td>Power down rank 1</td>
</tr>
</tbody>
</table>
Power/Performance Aware

- An adaptive history scheduler uses the history of recently scheduled memory commands when selecting the next memory command.

- A finite state machine (FSM) groups same-rank commands in the memory as close as possible -> total amount of power-down/up operations is reduced.

- This FSM is combined with performance driven FSM and latency driven FSM.
Adaptive Memory Throttling

Processors/Caches

Reads/Writes

Read/Write Queues

Scheduler

Memory Queue

Memory CONTROLLER

Throttle Delay Estimator

Throttling Mechanism

Model Builder
(a software tool, active only during system design/install time)

determines how much to throttle, at every 1 million cycles

decides to throttle or not, at every cycle

sets the parameters for the delay estimator

Power Target

sets the parameters for the delay estimator

Reads/Writes

DRAM
Adaptive Memory Throttling

- Stall all traffic from the memory controller to DRAM for T cycles for every 10,000 cycle intervals

• How to calculate T (throttling delay)?
Adaptive Memory Throttling

Model Building

- Throttling degrades performance

- Inaccurate throttling
  - Power consumption is over the budget
  - Unnecessary performance loss
Results

- Energy efficiency improvements from Power-Down mechanism and Power-Aware Scheduler
  - Stream: 18.1%
  - SPECfp2006: 46.1%
DRAM IO Optimization

- DRAM termination

- Hamming weight and Energy

[Seol’2016]
Bitwise Difference Encoding

- Observation: Similar data words are sent over the DRAM data bus
- Key Idea: Transfer the bit-wise difference between a current data word and the most similar data words
Bitwise Difference Encoding

- 48% reduction in DRAM IO power

[Seol’2016]