Cloak: Tolerating Non-Volatile Cache Read Latency

Apostolos Kokolis, Namrata Mantri†, Shrikanth Ganapathy*, Josep Torrellas and John Kalamatianos‡

University of Illinois Urbana-Champaign, †NVIDIA (work performed while at UIUC), *Rivos Inc. (work performed while at AMD† Research), ‡AMD Research

ACM International Conference On Supercomputing 2022
Introduction

- Data intensive workloads put increased pressure on LLCs
- Researchers examined other memory technologies for LLC
  - eDRAM
  - Non-Volatile Memories (NVM)

**NVM advantages**
- Higher density than SRAM
- Lower leakage power than SRAM
- Non-volatility
- Immunity to soft errors
- No Refresh or Row buffer over DRAM
NVM Limitations

- High latency
  - For both read and write requests
  - 10-30x higher for reads, 25-100x higher for writes
- Low Bandwidth
  - Non-pipelined accesses to the NVM
- High dynamic energy consumption per access

Need a low-cost solution to overcome the high latency problem of NVM
Prior Work

- Previous research focused on write latency
- Solutions:
  - Device and Circuit level
    - Sacrifice non-volatility and retention time
    - Adjust the transistor size
  - Architecture level with hybrid SRAM-NVM caches
    - Monitor access and swap lines between SRAM-NVM

Reducing retention time introduces refreshes
Increasing transistor size limits capacity benefits

Increased complexity/area to monitor accesses
Increased number of writes to NVM
Motivation

- NVMs (i.e., STT-RAM) is a viable alternative
- Need a low-cost architectural solution
  - No sacrifices to capacity, reliability and non-volatility

**Insight:** Exploit page level data re-use at the LLC to hide read latency
In our benchmarks, we find that on average 94.9% of LLC hits are for re-used pages.

As the size of LLC increases the number of LLC hits for re-used pages increases.
Contribution: Cloak

- Uses L1 TLB misses to identify page re-use in the LLC

- Adds small SRAM buffers (called Page Buffers or PB) to the LLC to service anticipating LLC requests

- Alters the LLC data layout to facilitate the activation of same page cache lines to the PBs and develops an adaptive replacement policy for the PBs

- Improves performance by 23.8% and ED$^2$ by 39.9% compared to an SRAM LLC (8.9% and 17.5% compared to NVM-Only)
Cloak Design Overview

- Cloak identifies page re-use from L1 TLB misses

- LLC is augmented with small SRAM buffers, Page Buffers (PB)

- PBs hold a subset of the LLC data for a specific page

- Same page cache lines are in the same physical LLC row
  - Fast retrieval of a page’s cache lines
Cloak Operation Overview

**L1 TLB Hint Operation**

- **L1 TLB Fill**
  - NO: Re-used Page
    - NO: Do nothing
    - YES: Page in PBs
      - NO: Do nothing
      - YES: Page lines > Threshold
        - NO: Search for available PB
          - Do nothing
        - YES: Transfer LLC resident lines to PB
          - Service from NVM Data Array

- YES: Do nothing

**LLC Request Operation**

- **LLC Read Request**
  - NO: Check LLC SRAM Tags
    - HIT: PB Tag Hit
      - YES: Service from the PB
      - NO: Service from NVM Data Array
    - MISS: Send to Mem
Data Layout

- NVM to PB transfer → find all LLC cache lines from same page
- Place same page lines in the same row of NVM

**Example for a 64MB slice**

<table>
<thead>
<tr>
<th>Physical Page Number</th>
<th>Page Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>36 bits</td>
<td>12 bits</td>
</tr>
</tbody>
</table>

- 36 bits for Page Offset
- 12 bits for Offset

**Processor Chip**

- Core
- L1
- L1 TLB
- L2
- L2 TLB
- MMU
- L2/L3 Interface Network
- L3 Ctrl

**L3 SRAM Tags**

- Set 0
- Set 1
- Set 2
- Set 3

**L3 Ctrl**

- Page Buffer Tags

- Page Buffer

Transfer Cache Lines to the Page Buffers

- LLC may have a subset of a page’s lines → Smaller 2KB PBs
- Multiplex lines from 2KB regions into the same PB
- Record the region of each cache line
- If lines contest for a spot
  - Pick the line that is from the same half of the page as the address that triggered the PB activation

\[
\begin{array}{c}
\text{2KB Region} \\
A1 - \ldots - A2 - A3 A4 \ldots - A5 \\
\downarrow \\
A1 - \ldots - A2 - \\
\text{Page Buffer}
\end{array}
\quad
\begin{array}{c}
\text{2KB Region} \\
A1 - \ldots - A2 - A3 A4 \ldots - A5 \\
\downarrow \\
A3 A4 \ldots A2 A5 \\
\text{Region Bits}
\end{array}
\quad
\begin{array}{c}
\text{2KB Region} \\
A1 - \ldots - A2 - A3 A4 \ldots - A5 \\
\downarrow \\
A3 A4 \ldots A2 A5 \\
\text{Region Bits}
\end{array}
\quad
\begin{array}{c}
\text{2KB Region} \\
A1 - \ldots - A2 - A3 A4 \ldots - A5 \\
\downarrow \\
1 1 \ldots 0 1 \\
\text{Region Bits}
\end{array}
\]
Tag Checking

Page Buffer Tags for keeping track of the cache lines that are present in the PBs

Page Buffer Tags

<table>
<thead>
<tr>
<th>PPN</th>
<th>Replacement</th>
<th>Residency</th>
<th>Region</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SRAM Tag Array

Hit/Miss

NVM Data Array

Sense Ampl

Page Buffer

PB Hit/Miss

LLC Read Request
Tag Checking

- **PPN**: of the activated page
- **Replacement**: counter selecting PBs
- **Residency**: #valid lines
- **Region**: 2KB region of lines

![Diagram of Tag Checking]

- LLC Read Request → SRAM Tag Array
- Hit/Miss
- NVM Data Array
- Sense Ampli
- Page Buffer
- PB Hit/Miss

Page Buffer Tags

- PPN: of the activated page
- Replacement: counter selecting PBs
- Residency: #valid lines
- Region: 2KB region of lines
Tag Checking

- **PPN**: of the activated page
- **Replacement**: counter selecting PBs
  - **Residency**: #valid lines
  - **Region**: 2KB region of lines

---

**SRAM Tag Array**

- LLC Read Request
- Hit/Miss

**NVM Data Array**

**Page Buffer**

**Page Buffer Tags**

- PPN
- Replacement
- Residency
- Region
Page Buffer Replacement Policy

- **Cloak** needs to find an available PB
- Uses a dynamic algorithm considering
  - #resident cache lines in a PB
  - Frequency of PB accesses

- **Replacement counter**
  - Residency * *Activation Period*
  - Decreases every cycle
  - Recalculated on PB accesses
  - 0 counter → can be replaced

- **Residency counter**
  - PB reads → decrease by 1
  - PB writes → increase by 1

### Page Buffer Tags

<table>
<thead>
<tr>
<th>PPN</th>
<th>Replacement</th>
<th>Residency</th>
<th>Region</th>
</tr>
</thead>
<tbody>
<tr>
<td>PPN</td>
<td>Replacement</td>
<td>Residency</td>
<td>Region</td>
</tr>
<tr>
<td>PPN</td>
<td>Replacement</td>
<td>Residency</td>
<td>Region</td>
</tr>
<tr>
<td>PPN</td>
<td>Replacement</td>
<td>Residency</td>
<td>Region</td>
</tr>
</tbody>
</table>
Huge Page Management

- System utilize huge pages of 2MB and 1GB
- **Cloak** transfers lines from 4KB regions
  - Records the region in the L1 TLB
  - Only sends a hint to the L3 Ctrl when a new chunk of the huge page is accessed
Methodology

Simulator
• Simics + SST
• Modified the LLC latency for NVM and added PBs for Cloak

Configurations Compared
• Baseline: SRAM based LLC
• NVM-Only: STT-RAM LLC
• Cloak: STT-RAM LLC and PBs
• O-SRAM: Optimistic LLC with the density of STT-RAM and the latency of SRAM

Workloads
• 10 benchmarks from SPEC\(^1\) CPU\(^\circ\) 2006 and SPEC\(^1\) CPU\(^\circ\) 2017+ 4 benchmarks from CORAL and CORAL2

\(^1\) SPEC\(^\circ\) and SPEC CPU\(^\circ\) are registered trademarks of the Standard Performance Evaluation Corporation. See www.spec.org for more information
Efficiency of the Page Buffers

<table>
<thead>
<tr>
<th>Application</th>
<th>PB Hits over LLC Hits (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>505.mcf_f</td>
<td>54%</td>
</tr>
<tr>
<td>519.lbm_f</td>
<td>57%</td>
</tr>
<tr>
<td>557.xz_f</td>
<td>54%</td>
</tr>
<tr>
<td>450.soplex</td>
<td>57%</td>
</tr>
<tr>
<td>459.GemsFDTD</td>
<td>54%</td>
</tr>
<tr>
<td>473.astar</td>
<td>54%</td>
</tr>
<tr>
<td>462.libquantm</td>
<td>54%</td>
</tr>
<tr>
<td>433.milc</td>
<td>54%</td>
</tr>
<tr>
<td>471.omnetpp</td>
<td>54%</td>
</tr>
<tr>
<td>437.lesle3d</td>
<td>54%</td>
</tr>
<tr>
<td>Kripke</td>
<td>54%</td>
</tr>
<tr>
<td>XSbench</td>
<td>54%</td>
</tr>
<tr>
<td>QLA</td>
<td>57%</td>
</tr>
<tr>
<td>Iulesh</td>
<td>54%</td>
</tr>
<tr>
<td>Mean</td>
<td>54%</td>
</tr>
</tbody>
</table>
### Efficiency of the Page Buffers

- On average, more than half of the requests hit in the PBs.
- Especially for benchmarks with high L2 MPKI like XSBench, this leads to great performance improvement.
- Most TLB hints can activate the cache lines to a PB.

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>PB Hits over LLC Hits (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>505.mcf_f</td>
<td>54%</td>
</tr>
<tr>
<td>519.ibm_f</td>
<td>57%</td>
</tr>
<tr>
<td>557.xz_f</td>
<td>57%</td>
</tr>
<tr>
<td>450.soplex</td>
<td>60%</td>
</tr>
<tr>
<td>459.GemmFDTD</td>
<td>60%</td>
</tr>
<tr>
<td>473.astar</td>
<td>60%</td>
</tr>
<tr>
<td>462.libquantum</td>
<td>60%</td>
</tr>
<tr>
<td>433.milc</td>
<td>57%</td>
</tr>
<tr>
<td>471.omnetpp</td>
<td>54%</td>
</tr>
<tr>
<td>497 Leslie3d</td>
<td>54%</td>
</tr>
<tr>
<td>Kripke</td>
<td>54%</td>
</tr>
<tr>
<td>XSBench</td>
<td>57%</td>
</tr>
<tr>
<td>QLA</td>
<td>54%</td>
</tr>
<tr>
<td>Iulesh</td>
<td>54%</td>
</tr>
<tr>
<td>Mean</td>
<td>54%</td>
</tr>
</tbody>
</table>
Read Latency Reduction of Cloak over NVM-Only

- Decrease in time spent for LLC reads (%)
  - Mean: 42.5%
Read Latency Reduction of Cloak over NVM-Only

- NVM-Only and Cloak have similar L3 MPKI
- The PBs speed-up Cloak by servicing requests faster and by not blocking the NVM data array

Decrease in time spent for LLC reads (%)

505.mcf_f  519.ibm_f  557.x2_f  450.soplex  459.GemsFDTD  473.astar  462.libquantum  433.milc  471.omnetpp  437.leslie3d  Kripke  XSbench  QLA  lulesh  Mean

42.5%
Performance Evaluation

Normalized Speedup

- Baseline
- NVM-Only
- Cloak
- O-SRAM

[Graph showing normalized speedup for various benchmarks with categories and specific values for each benchmark]
Performance Evaluation

- **Cloak** is 23.8% faster than SRAM
- NVM-Only is 14.9% faster than Baseline, O-SRAM is 27.9%
- L2 miss response time decreases by 30.5% for **Cloak** and by 15.9% for NVM-Only compared to Baseline
Energy Evaluation

Normalized ED

Baseline | NVM-Only | Cloak | O-SRAM

Cloak: Tolerating Non-Volatile Cache Read Latency
Energy Evaluation

- **Cloak** reduces $ED^2$ by 39.9%
- NVM-Only reduces it by 22.4% and O-SRAM by 43.3%
- **Cloak** consumes more dynamic energy in LLC, but it’s faster
Page Buffer Characterization

**PB hits over fetched cache lines (%)**

<table>
<thead>
<tr>
<th>Group</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group A</td>
<td></td>
</tr>
<tr>
<td>Group B</td>
<td></td>
</tr>
<tr>
<td>Group C</td>
<td></td>
</tr>
</tbody>
</table>

**Promoted cache lines over LLC resident lines (%)**

<table>
<thead>
<tr>
<th>Group</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group A</td>
<td></td>
</tr>
<tr>
<td>Group B</td>
<td></td>
</tr>
<tr>
<td>Group C</td>
<td></td>
</tr>
</tbody>
</table>

Group A: SPEC CPU® 2017, Group B: SPEC CPU 2006, Group C: CORAL and CORAL2
Page Buffer Characterization

On average, 51.1% of the promoted lines are referenced by the benchmarks.

On average, 68.8% of the lines are promoted to a PB.

Group A: SPEC CPU® 2017, Group B: SPEC CPU 2006, Group C: CORAL and CORAL2
Sensitivity Analysis

Size Sensitivity

<table>
<thead>
<tr>
<th>Size</th>
<th>NVM-Only</th>
<th>Cloak</th>
<th>O-SRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>4MB</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>8MB</td>
<td>1.2</td>
<td>1.2</td>
<td>1.2</td>
</tr>
<tr>
<td>16MB</td>
<td>1.4</td>
<td>1.4</td>
<td>1.4</td>
</tr>
<tr>
<td>32MB</td>
<td>1.6</td>
<td>1.6</td>
<td>1.6</td>
</tr>
</tbody>
</table>

Read Latency Sensitivity

<table>
<thead>
<tr>
<th>Latency</th>
<th>NVM-Only</th>
<th>Cloak</th>
</tr>
</thead>
<tbody>
<tr>
<td>3ns</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>6ns</td>
<td>1.2</td>
<td>1.2</td>
</tr>
<tr>
<td>9ns</td>
<td>1.4</td>
<td>1.4</td>
</tr>
</tbody>
</table>

Cloak: Tolerating Non-Volatile Cache Read Latency
Sensitivity Analysis

Size Sensitivity

- Cloak always achieves higher performance than NVM-Only.
- Even for 4MB Cloak achieves the performance of Baseline.

Read Latency Sensitivity

- Cloak can tolerate higher read latency than NVM-Only.
- It is always faster than Baseline.
Also in the paper . . .

- More details on Cloak
  - Area and Energy evaluation of Cloak’s HW
  - Implementation details
  - Handling Huge Pages

- Evaluation Results
  - More performance evaluation
  - Alternative prefetch design
  - Breakdown of energy evaluation
Conclusion

- **Cloak** uses L1 TLB misses to predict page reuse in the LLC *ahead of time*

- Changes the LLC layout to facilitate the use of PBs

- **Cloak** improves performance
  - by 23.8% and ED$^2$ by 39.9% compared to an SRAM LLC
  - by 8.9% and 17.5% compared to an NVM-Only design
Cloak: Tolerating Non-Volatile Cache Read Latency

Apostolos Kokolis, Namrata Mantri†, Shrikanth Ganapathy*, Josep Torrellas and John Kalamatianos‡

THANK YOU!!

University of Illinois Urbana-Champaign, †NVIDIA, *Rivos Inc., ‡AMD Research

ACM International Conference On Supercomputing 2022