Evaluating the Reuse Cache for mobile processors

Lokesh Jindal, Urmish Thakker, Swapnil Haria
CS-752 Fall 2014
University of Wisconsin-Madison
Problem:
Mobile SOCs – Area is money!
Cache Area = 20-30% of SOC die area

Solutions?
Technology Scaling
   Expensive, diminishing returns
Reuse Caches
   Reduced cache size, comparable performance

Results=>
50% Area reductions for 5% performance loss
Outline

• Motivation
• Implementation
• Methodology
• Performance Evaluation
• Future Directions and Conclusion
Outline

• Motivation
• Implementation
• Methodology
• Performance Evaluation
• Future Directions and Conclusion
Caches in mobile SOCs
Area-Performance Tradeoffs*

![Graph showing Area-Performance Tradeoffs for conventional cache](image)

- Performance vs. Cache Size
- Conventional Cache

*Representative Graph
Area-Performance Tradeoffs*

*T Representative Graph
Motivation

Most of the inserted lines will not experience any hit because hits concentrate in a few lines.

Reuse Cache: Original idea

Data is stored only when \textbf{reuse} is detected

First access:
Miss $\rightarrow$ only tag insertion

Second access:
Tag hit $\rightarrow$ data insertion (reuse detected)

Reuse Cache: good idea for desktop processors (seemingly) but what about for mobile processors?

Same average performance* with 84% savings

*100 multiprogrammed SPEC CPU 2006 with a 3-core CMP
Revisiting Reuse Caches for the Mobile Environment

• Questions
  – Memory characteristics of mobile workloads?
  – Spatial/Temporal/Reuse locality at L2 level?
  – Reuse Cache performance for smaller, simpler caches?
Low temporal locality

% Dead Lines (on Fetch) = 1 - \( \frac{\text{# Lines Reused}}{\text{# Lines Fetched}} \)

% Dead Lines (on Eviction) = \( \frac{\text{# Unused Evicted Lines}}{\text{# Lines Evicted}} \)
Low temporal locality

85-90% dead lines!

% Dead Lines (on Fetch) = 1 - \( \frac{\# \text{Lines Reused}}{\# \text{Lines Fetched}} \)

% Dead Lines (on Eviction) = \( \frac{\# \text{Unused Evicted Lines}}{\# \text{Lines Evicted}} \)
Outline

• Motivation
• Implementation
• Methodology
• Performance Evaluation
• Future Directions and Conclusion
Organization

• Tag Array
  – Inclusive (aids coherence)
  – Set associative
  – (Forward) Pointer to data array entry

8 ways
....
Organization

• Tag Array
  – Inclusive (aids coherence)
  – Set associative
  – (Forward) Pointer to data array entry

• Data Array
  – Set associative
  – Stores reused lines
  – (Reverse) Pointer to tag array entry
Coherence

• TO-MOESI protocol
  – Small extension to the MOESI protocol
• New Tag-Only state
• Tag+Data states: Modified, Shared, Owned, Exclusive
• Transitions triggered by data insertions/evictions
• Read the paper!
Replacement

• Independent replacement policies for Tag and Data arrays

• Tag Array
  – Not Recently Reused (NRR)

• Data Array
  – Not Recently Used (NRU) + Shadow directory
Pomerene et al. proposed shadow directory
- Transient lines that must be flushed quickly
- Lines that become active after long periods of inactivity.

• Pomerene et al. proposed shadow directory
  – Transient lines that must be flushed quickly
  – Lines that become active after long periods of inactivity.
• Shadow bit added in reuse cache to break inefficient cycles
Outline

• Motivation
• Implementation
• Methodology
• Performance Evaluation
• Future Directions and Conclusion
Methodology

- Baseline
  - 32KB/4way L1 I$ + 32KB/4way L1 D$
  - 4MB/8way L2$ (Shared LLC)
  - 8 wide OOO pipeline
  - 1.4GHZ | 32nm | Low power device
  (Based on ARM A15 specifications)

- Workloads
  - AsimBench
  - SPEC CPU 2006
  - Self-written microbenchmarks
  (functional verification)
Workload – AsimBench*

- Popular Android applications
- 11 diverse applications

- 6 most memory-intensive apps selected for performance evaluation
  - BBench (Web Browser): Load web pages
  - K9Mail (Email): Load/Show emails
  - NeteaseNews (News): Check and load news
  - BaiduMap (Map): Load map information of a specific area
  - FrozenBubble (Game): Load game

Outline

• Motivation
• Implementation
• Methodology
• Performance Evaluation
• Future Directions and Conclusion
Performance - AsimBench

Simulation length of 5 Billion instructions after boot in Full System mode

Average performance loss of 4.16% (not bad!)

Simulation length of 5 Billion instructions after boot in Full System mode
Performance – SPEC CPU2006

Average performance drop of 7.17%

(Simulation length of 10B instructions in System Emulation mode)
Configuration Exploration (WIP)

Normalized IPC vs. Different Benchmarks

- mcf_32.scr
- omnetpp_32.scr
- gromacs_base_32.scr
- zeaump_base_32.scr
- soplex_base_32.scr

Reuse 4M + 2M
Reuse 4M + 2M
Multi-core Performance

SPEC benchmarks simulated on 4 cores, running 10 Billion instructions each (Full system + multi-core + AsimBench == Too slow)

~15% Degradation
## Area Improvements

<table>
<thead>
<tr>
<th></th>
<th>Conventional (4M + 4M)</th>
<th>Reuse Cache (4M + 1M)</th>
<th>Reuse Cache (4M + 2M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Area (in mm$^2$)</td>
<td>2.37</td>
<td>1.18</td>
<td>1.5</td>
</tr>
</tbody>
</table>

Area savings of **50%** for the 4+1 reuse cache, **37%** for the 4+2 cache. (Data generated using Cacti v6.5)
% Improvement in Power
% Improvement in Power

~10% Improvement

+ Reduced Leakage power
- Increased DRAM power
Live-ness Analysis - AsimBench

- % Live lines increased from 9.54 % to 47.95%
Pictures!

• AsimBench apps running on gem5 ...
Make yourself at home
You can put your favorite apps here.

OK

To see all your apps, touch the circle.
Outline

• Motivation
• Implementation
• Methodology
• Performance Evaluation
• Future Directions and Conclusion
Future Directions

• Replacement Policies (NRR + Shadow) for Fully Associative data array
• Explore allotment policies
• Better uses for extra tag entries
• Longer/Better simulations using simpoints
• Multi-core simulations using AsimBench to validate coherence protocols
• Comprehensive energy analysis
Summary

For mobile workloads, reuse cache promises
+ Significant area reduction (~50%)
+ Decent power savings (~10%)
- Marginal performance loss (~6%)
Questions?

Thank You!!
Configuration Exploration

Power in mW

Increased Power (mW)
L2 => 1 MB Cache
Animate this to cross out 1 MB and say 50% of data area

L2 => 4 MB Cache
Cross out and say Performance of 16 MB Cache
Results

• Mapping exploration -> IPC, ASIM + SPEC
• 4+4 vs (4+1) spec, asimbench
• Shadow vs NRR
• Boot times
• Bigger is better
• CACti graph
• Energy graph
• Dead lines
Baseline

- 32KB/4way L1 I$ + 32KB/4way L1 D$
- 4MB/8way L2$ (Shared LLC)
- 8 wide OOO pipeline
- 1.4GHZ | 32nm | Low power device

- Based on ARM A15 specifications
Challenges

- ARM aarch32 incompatible with gem5’s Ruby memory model
- Simulation infrastructure for ARM in full system mode
Contributions