Loop Instruction Caching for Energy-Efficient Embedded Multitasking Processors

Ji Gu, Tohru Ishihara and Kyungsoo Lee
Department of Communications & Computer Engineering
Graduate School of Informatics
Kyoto University
Outline

1. Background
2. Research overview
3. Design Approach
4. Experimental Results
5. Conclusions
Where does the processor energy go?

• Cache memories are energy-consuming due to instruction/data supply
Research Problem (1/2)

- Observed behavior of embedded applications [1]
  - 77% of execution time spent in loops
  - 47% of execution time spent in loops of size 64 or less
  - 46% of execution time spent in loops that iterate 5 times or more

- Loop behavior can be exploited for low-energy design

---

Objective: Caching instructions for most of loops to a special cache, which has small energy consumption

- To avoid repeated instruction fetching from the energy-consuming I-cache memories as much as possible

<table>
<thead>
<tr>
<th></th>
<th>Size</th>
<th>Latency</th>
<th>Energy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loop Cache</td>
<td>Small</td>
<td>Small</td>
<td>Small</td>
</tr>
<tr>
<td>Normal Cache</td>
<td>Large</td>
<td>Small</td>
<td>Large</td>
</tr>
<tr>
<td>Off-chip mem.</td>
<td>Huge</td>
<td>Huge</td>
<td>Huge</td>
</tr>
</tbody>
</table>
Hardware Design

- SMLIC: Shared Multitasking Loop Instruction Cache
- Three operation modes of SMLIC
  - Non-loop execution: SMLIC inactive, I-cache fetching
  - First loop iteration: I-cache fetching & SMLIC buffering
  - Following loop iterations: I-cache inactive, SMLIC fetching
Software Design

- Four special instructions: slp, brb, brf, elp
  - Inserted into program code at design time – statically
  - Controlling SMLIC operations at run time - dynamically
SMLIC Architecture

- First loop iteration
SMLIC Architecture

- Second loop iteration
SMLIC for Context Switch

During Context Switch (CS)
- IF SMLIC is NOT used: Normal CS operation
- IF SMLIC is used: Normal CS operation + SMLIC CS operation
  - SMLIC CS operation: convert Local PC to Global PC
Experimental Setup

- Multitasking application
  - MiBench, Powerstone and some other DSP kernel programs
  - merge: five DSP kernel programs (convolution, ftt, fir, lms, matrixmul) into a single task

<table>
<thead>
<tr>
<th>Multitasking set</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set 1</td>
<td>crc32, cryptcode, des</td>
</tr>
<tr>
<td>Set 2</td>
<td>blit, rijndael, seal</td>
</tr>
<tr>
<td>Set 3</td>
<td>adpcm, jpeg, rawaudio, stringsearch</td>
</tr>
<tr>
<td>Set 4</td>
<td>blowfish, merge, rawaudio, sha</td>
</tr>
<tr>
<td>Set 5</td>
<td>adpcm, bcnt, jpeg, rawaudio, sha</td>
</tr>
<tr>
<td>Set 6</td>
<td>blit, blowfish, cryptcode, merge, rawaudio</td>
</tr>
</tbody>
</table>
Multitasking System

- Multiprocessor system with a task scheduler
  - Simulate the multitasking system
Design Size of SMLIC

- **Index table**
  - The maximum loop size (number of loop instructions)

- **Branch target table**
  - The maximum number of loop branch instructions
Experimental Setup

- Processor specified at RTL level for simulation (ISS)

- 1KB SMLIC, 16KB I-cache
  - CACTI, DesignCompiler used for energy/area evaluation

- Round Robin task scheduling, with switching intervals of 20K, 40K, 60K, 80K and 100K clock cycles

- Compare with: Filter Cache (FC), Partitioned Loop Instruction Cache (PLIC)
Baseline

- **Filter cache**
  - Act as like a L0 cache
  - Always accessed first in each fetch cycle, an FC miss will incur an extra cycle for the I-cache fetch

- **Partitioned Loop Instruction Cache (PLIC)**
Results: Reduction of I-cache access

- FC: 27-76%, 44% on average
- PLIC: 50-86%, 65% on average
- SMLIC: 12-86%, 49% on average
Results: Energy reduction

- FC: 15-67%, 34% on average
- PLIC: 45-78%, 59% on average
- SMLIC: 11-79%, 45% on average
Overhead: Area and Delay

<table>
<thead>
<tr>
<th></th>
<th>Area (65nm)</th>
<th>Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>FC</td>
<td>9.6%</td>
<td>21-66%*</td>
</tr>
<tr>
<td>PLIC</td>
<td>19.1%</td>
<td>1.2%</td>
</tr>
<tr>
<td>SMLIC</td>
<td>10.2%</td>
<td>1.2%</td>
</tr>
</tbody>
</table>

* FC is always accessed first in each fetch cycle, an FC miss will incur an extra cycle for the I-cache fetch. Delay ranges in 21-66% due to various switching intervals for multitasking system.

- FC: Large performance overhead
- PLIC: Large area overhead
- SMLIC: Reasonable for both
Conclusions

- Loops are common in applications of most embedded systems.
- A shared loop instruction cache is proposed for multitasking applications:
  - A set of special instructions are used to control the operation of the cache.
  - A hierarchical structure is used for storing instructions at the low on-chip memory cost.
  - A low-cost Global PC update logic for context switch at hardware level.
- An average of 45% energy saving with 1.2% performance overhead and 10.2% area cost.
Thank you! & Questions?
Filter Cache

Conventional configuration

Filter cache configuration