#### Dynamic Management Technique to Mitigate Performance Degradation for Low-Leakage Caches

Komiya, Reiko JSPS Research Fellow DC | Department of Informatics, Kyushu University | Department of Informatics, Kyushu University

Inoue, Koji Department of Informatics, Kyushu University

Murakami, Kazuaki Department of Informatics, Kyushu University

https://hdl.handle.net/2324/6376

出版情報:Proceedings of Cool Chips X, pp.173-184, 2007-04-19 バージョン: 権利関係:

## Dynamic Management Technique to Mitigate Performance Degradation for Low-Leakage Caches

#### Reiko Komiya, Koji Inoue, Kazuaki Murakami

Kyushu University

# Outline

- Introduction
  - Leakage energy of cache memory
  - Conventional low leakage cache : Cache decay
- Problem of cache decay approach
- Solution: "Always-Active scheme"
- Results
- Conclusions

# Introduction

Energy consumption = Dynamic energy + Static energy



#### Cache leakage reduction is very important!!

\*1 Fred Pollack (Intel Fellow): New Microarchitecture Challenges in the Coming Generations of CMOS Process Technologies, Micro32

\*2 Simon Segars, "Low Power Design Techniques for Microprocessors," ISSCC2001

# Normal cache & Conventional Iow-leakage cache



Two operation modes

✓ active mode : high-leakage ☺ , preserve the data ☺
 ✓ sleep mode : low-leakage ☺ , destroy the data ☺

| majority | sleep mode    | active mode   |
|----------|---------------|---------------|
| good     | Leakage ↓     | Performance 1 |
| bad      | Performance ↓ | Leakage 1     |

2007/04/19

# Mode transition algorithm for cache decay

Cache decay: Conventional low-leakage cache



**Extra-misses lead to performance degradation!!** 

#### Performance impact of extra-misses



Many extra-misses cause large performance degradation!

2007/04/19

# **Our Goal**

Achieve high-performance, low-leakage cache!

- Problem of conventional low-leakage caches
  - Degrades the performance due to extra-misses
- Our approach
  - Reduces extra-misses and improve performance
  - Prohibits some cache lines from going to sleep mode

## Analysis of extra-misses

• Extra-Miss Density (*EMD*):

 $EMD_i = \frac{\text{number of extra-misses at the cache line }i}{\text{average number of extra-misses for all cache lines}}$ 

#### • Example

Number of extra-misses at each cache line



## Characteristics of extra-misses

 $\blacksquare$  EMD < 1  $\blacksquare$  1  $\leq$  EMD < 2  $\blacksquare$  2  $\leq$  EMD < 4  $\blacksquare$  4  $\leq$  EMD

# Breakdown of cache lines in terms of *EMD*

Breakdown of extra-misses in terms of *EMD* 



2007/04/1

# **Always-Active scheme**

- Introduces "Always-Active mode (AA mode)"
- Identifies cache lines which would cause frequently extra-misses, and marks the line AA mode line
- Prohibits AA mode line from going to sleep mode
- Reduces extra-misses in AA mode line

# How to identify AA mode lines

(a) Decay interval has passed since the last access (b) Tag hits

- In the case of normal cache, this access would be hit
- In the case of cache decay, this access would be extra-miss

The accesses which satisfy (a) and (b) are called "extra-misscandidate (EMC)"

• If an access is EMC, we make the following judgment

number of EMC at the cache line *i*  $EMCD_i =$ —— > Threshold average number of EMC for all cache lines

• If this equation is satisfied, the line transits to AA mode line Cool Chips X

### Mode transition algorithm for AA scheme

A line which causes EMC frequently  $\Rightarrow$  AA mode line



### How to measure EMCD<sub>i</sub> dynamically



 $1 > 2 \times 3 \cdots$  We judge this equation during program execution

Example) Number of cache lines =  $1024 (=2^{10})$ , Threshold =  $2 (=2^{10})$ 



### Hardware implementation for AA scheme



# **Experimental setup**

- Evaluation models
  - Base: normal cache
  - Decay: conventional low-leakage cache
  - AA2: cache decay with AA Scheme (threshold value=2)
- Cache configuration
  - L1 data cache
    - Cache size: 32KB
    - Associativity: 32way
    - Hit latency: 1 clock cycle
    - Miss penalty: 64 clock cycles
- Evaluation items
  - Normalized execution time (against Base model)
  - Normalized energy consumption (against Base model)

2007/04/19  $E_{total} = LE_{L1} + DE_{L1} + DE_{memory}$ 

#### Performance & Energy consumption



# Conclusions

- We have proposed a high-performance, low-leakage cache: AA Scheme
  - Detects lines which cause EMC frequently at run time
  - Improves the performance by means of making the lines
    AA mode and prohibiting AA mode lines from going to
    sleep mode
- Evaluation results
  - Higher performance and lower energy consumption
  - Best case (183.equake):
    - Performance degradation: 22%  $\rightarrow$ 7%
    - Energy consumption: 13% reduction