A Reconfigurable Cache Architecture for Energy Efficiency

Karthik T. Sundararajan
School of Informatics
The University of Edinburgh
United Kingdom
t.s.karthik@ed.ac.uk

Timothy M. Jones
Computer Laboratory
The University of Cambridge
United Kingdom
timothy.jones@cl.cam.ac.uk

Nigel Topham
School of Informatics
The University of Edinburgh
United Kingdom
npt@staffmail.ed.ac.uk

ABSTRACT

On-chip caches often consume a significant fraction of the total processor energy budget. Allowing adaptation to the running workload can significantly lower their energy consumption. In this paper, we present a novel Set and way Management cache Architecture for efficient Run-Time reconfiguration (Smart cache), a cache architecture that allows reconfiguration in both its size and associativity. Results show the energy-delay of the Smart cache is on average 18% better than state-of-the-art reconfiguration architectures.

Categories and Subject Descriptors

B.3.2 [**Memory Structures**]: Cache memories

General Terms

Management, Performance, Experimentation

Keywords

Configurable cache, Cache tuning, Smart cache

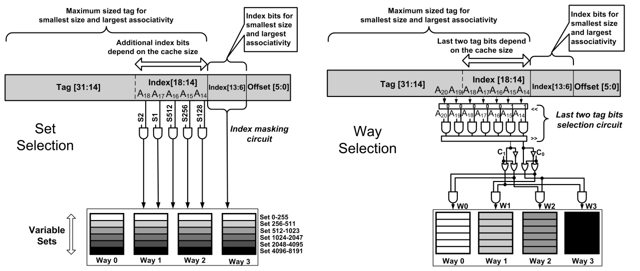
1. INTRODUCTION

The power dissipation of modern microprocessors is a primary design constraint across all processing domains, from embedded devices to high performance chips. Cache memories contain a large number of transistors and consume a large amount of energy. Customization of cache parameters may be static or dynamic; in a static approach the designer sets the cache parameters before synthesis, whereas in a dynamic scheme the cache parameters can be modified within a certain range to adapt to the running application.

This paper proposes a configurable cache architecture that allows reconfiguration of both the size and associativity, providing maximum flexibility to the application. We compare our approach, called the Smart cache, against state-of-the-art cache reconfiguration techniques and show that our scheme's energy-delay product is on average 14% better than these prior works.

2. THE SMART CACHE ARCHITECTURE

This section describes the Smart architecture that we use for the level-2 cache within the system. Figure 1 shows how each address is mapped onto a 2MB level-2 cache. There are two complementary circuits used in parallel that perform the mapping to route the address to the correct set and way.


Maximum sized tag for smallest size and largest associativity
Additional index bits depend on the cache size
Index bits for smallest size and largest associativity
Tag [31:14]
Index[18:14]
Index[13:6]
Offset [5:0]
Set Selection
S2
S1
S512
S256
S128
Index masking circuit
Variable Sets
Set 0-255
Set 256-511
Set 512-1023
Set 1024-2047
Set 2048-4095
Set 4096-8191
Way 0
Way 1
Way 2
Way 3
Last two tag bits depend on the cache size
Index [18:14]
Way Selection
Last two tag bits selection circuit
W0
W1
W2
W3

(a) Set Selection (b) Way Selection

Figure 1: Organization of the Smart cache architecture. Varying the cache size (through the set selection circuits) is performed in parallel with altering the associativity (through the way selection logic).

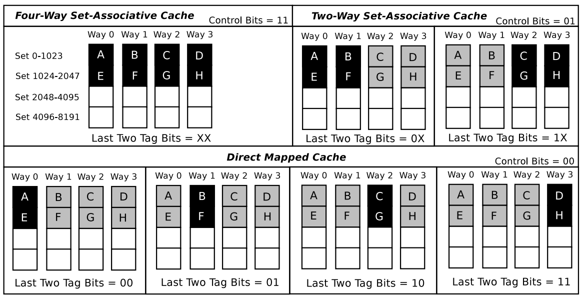


Figure 2: Behavior of the Smart cache. Black, grey and white regions show accessed, unaccessed and disabled sets respectively. Control bits determine the associativity and the last two tag bits determine the ways.

We group the sets in each bank by augmenting the cache with size selection bits that determine the sets that are enabled. These are then *AND*ed with bits from the index to determine the sets to access. A 64KB cache (sets 0-255) is always enabled even when all size selection bits are 0.

In order to control the associativity of the cache, we augment it with a way selection circuit. This uses the last two tag bits to route accesses to the ways that are enabled. Since the cache size can vary, the size selection bits are required to correctly identify the last two tag bits.

CF'11, May 03-05, 2011, Ischia, Italy.
ACM 978-1-4503-0698-0/11/05.

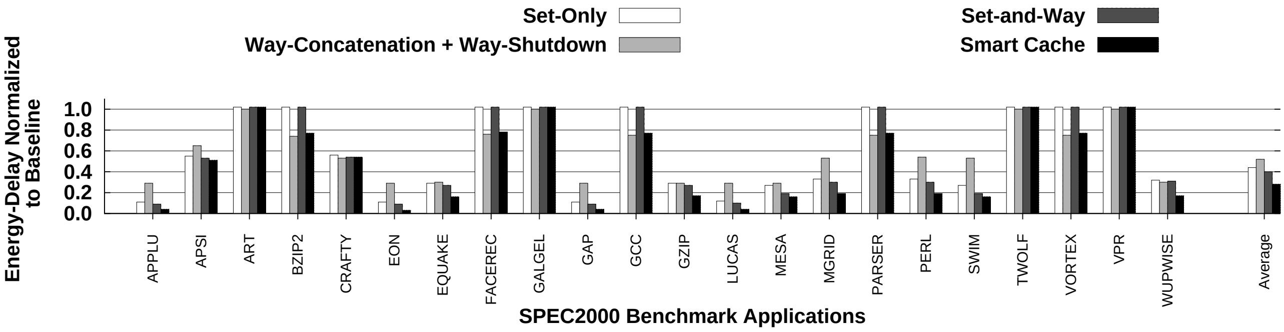


Figure 3: Energy-delay values for different cache architectures running on the same L2 baseline cache.

As the cache size and associativity varies, so does the number of bits needed for the tags. In our architecture, we store the maximum sized tag for each line (i.e., for the smallest cache and largest associativity). Figure 2 shows how the control and tag bits map to the ways that are enabled in our Smart cache.

2.1 Overheads

The way selection circuit does not appear in the cache's critical path because it can operate in parallel with the tag and data array address decoders [5]. The set selection circuit can be folded into the decoders to avoid any delay in calculating the index bits [1]. Therefore there is no increase in the cycle time for accessing the cache using our approach. However, after reconfiguring the cache to a smaller size, we turn off unused sets and ways, destroying their contents. Therefore we must flush any dirty lines back to the next level in the memory hierarchy. In all our simulations we add the flushing cost that includes both power and performance costs for copying back the dirty lines. We have calculated the area overheads of the cache as 0.5% over the baseline. This is due to the extra control circuitry required to perform set selection, way selection and reconfiguration.

2.2 Relation to Prior Work

There are several key differences between our Smart cache and state-of-the-art reconfiguration techniques. In our approach the associativity and size are varied in parallel by using the way control signals and the size control registers. The Smart cache organizes ways at set boundaries, which avoids flushing data back to memory when increasing the associativity but keeping the cache size fixed. This addresses the shortcomings in previous techniques [5], allowing dynamic reconfiguration of the cache. In addition to this, the Smart cache offers 3x more cache configurations than the set-only [3] and hybrid [4] schemes which combine way-concatenation with way-shutdown.

3. RESULTS

This section evaluates our Smart cache approach against prior cache architectures on static configurations of the level-2 cache. We implemented these approaches in the HotLeakage simulator [6], updating the underlying power models to use a more recent version of Cacti (v5.3) [2] that has been modified to support our new circuitry for 70nm process technology. We also altered the simulator to include the power and performance overheads of reconfiguring each cache. We modelled an Alpha out-of-order superscalar whose cache configurations are similar to an Intel Core 2 processor.

Figure 3 shows the energy-delay values achieved when running each benchmark on the best static configuration for that application for each cache architecture. The best static configuration is the one that has the lowest energy-delay and a maximum 2% performance loss, from all the possible configurations. Figure 3 shows on average level-2 cache energy-delay achieved by our approach is 0.24, which is 14% better than set-only cache [1], set-and-way cache [4] and 25% better than the way-concatenation combined with the way-shutdown cache [5]. This clearly demonstrates the benefits of using our Smart cache architecture for run-time cache reconfiguration.

4. CONCLUSIONS

This paper has presented a novel configurable cache architecture that can be reconfigured in both size and associativity to match the requirements of an application. We have demonstrated that our Smart cache architecture has an energy-delay product that is on average 18% better than state-of-the-art approaches. Future work will harness this architecture for dynamic cache adaptation to fit the cache to the requirements of each application phase as it runs.

5. REFERENCES

[1] M. Powell et al. Gated-vdd: a circuit technique to reduce leakage in deep-submicron cache memories. In *ISLPED*, 2000.

[2] S. Thoziyoor et al. Cacti 5.1. Technical Report HPL-2008-20. *HP Laboratories Palo Alto*, 2008.

[3] S.-H. Yang et al. Dynamically resizable instruction cache: An energy-efficient and high-performance deep-submicron instruction cache. *Purdue University*, 2000.

[4] S.-H. Yang et al. Exploiting choice in resizable cache design to optimize deep-submicron processor energy-delay. In *HPCA*, 2002.

[5] C. Zhang et al. A highly configurable cache architecture for embedded systems. *ISCA*, 2003.

[6] Y. Zhang et al. Hotleakage: A temperature-aware model of subthreshold and gate leakage for architects. *Technical Report,CS-2003-05*, 2003.