A Power-Efficient Migration Mechanism for D-NUCA Caches

A. Bardine*, M. Comparetti*, P. Foglia*,G. Gabrielli*, C. A. Prete*
Dipartimento di Ingegneria dell'Informazione, Università di Pisa
Via Diotisalvi 2, 56126 Pisa, Italy
{alessandro.bardine,manuel.comparetti,foglia,giacomo.gabrielli,prete}@iet.unipi.it

Abstract

D-NUCA L2 caches are able to tolerate the increasing wire delay effects due to technology scaling thanks to their banked organization, broadcast line search and data promotion/demotion mechanism. Data promotion mechanism aims at moving frequently accessed data near the core, but causes additional accesses on cache banks, hence increasing dynamic energy consumption. We shown how, in some cases, this migration mechanism is not successful in reducing data access latency and can be selectively and dynamically inhibited, thus reducing dynamic energy consumption without affecting performances.

1 Introduction

CMOS technology trends and bandwidth demands of cores are leading to the use of large, on-chip L2 caches. For high clock frequency designs their access latency is dominated by the wire delay [1]. In order to reduce this effect, NUCA Caches have been proposed as a new paradigm for on-chip L2 cache memories [2].

In a NUCA architecture the cache is partitioned in a matrix of independent banks, and the communications between these cache banks and the central cache controller are supported by a switched on-chip network. In this organization the banks closer to the processor can be accessed independently from the other banks, hence allowing shorter access latencies with respect to banks located farther away. The aim of this architecture is maintaining low access latencies for L2 cache lines near to the controller, typical of small size cache banks, while guaranteeing a large capacity for the L2 cache. The mapping between cache lines and physical banks can either be Static or Dynamic (namely S-NUCA or D-NUCA). In D-NUCA caches each line can be mapped to one set of different banks, and most frequently accessed data are migrated towards the banks which are more closer to the processor (Fig. 1). This maximizes the utilization of the fastest banks during the application. As shown in Fig. 1 the banks are logically grouped in rows and columns, each bank containing a fixed group of lines.

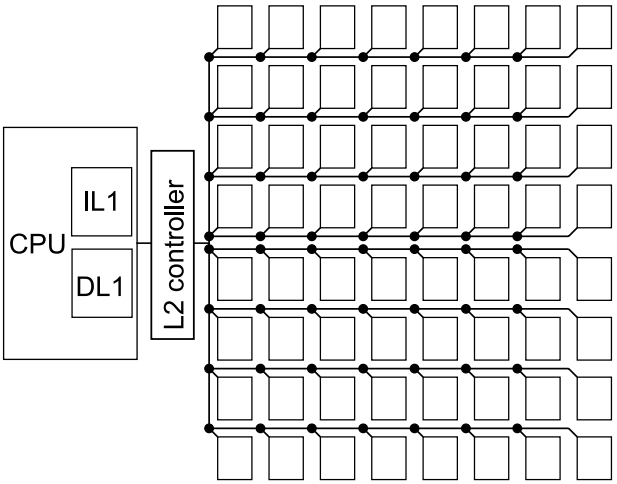


Figure 1. A D-NUCA L2 cache architecture made up by 8 rows of 8 banks connected by a switched network. The cache has 8 columns of banks, so it is globally 8 set-associative.

The entire address space is spanned on each column of banks. Each bank belonging to a row behaves as a single way of a set associative cache, and a line can reside only in a bank belonging to that row. A given cache line can be found at the same index in every bank pertaining to the same row, so the D-NUCA cache globally behaves as a set-associative cache whose associativity is given by the number of columns. When a cache line search is performed, the controller first determines the bank row which could contain the data, depending on the lowest order bits of the index field of the address. The remaining part of the field will be used to index every bank of the row, each of which is a candidate for a cache hit. Thus,

This work is partially supported by the SARC project funded by the European Union under the contract no. 27648.
*Members of HiPEAC – The European Network of Excellence on High-Performance Embedded Architecture and Compilation.

978-3-9810801-5-5/DATE09 © 2009 EDAA

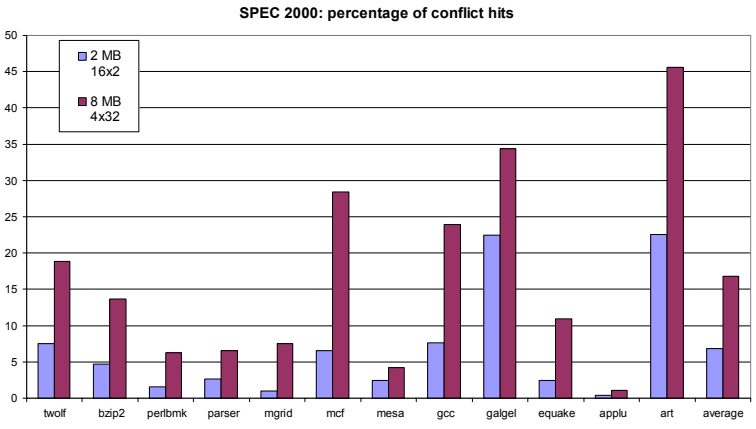


Figure 2. Percentage of conflict hits on global L2 hits for a 2 MB 2-way set associative D-NUCA and a 8 MB 32-way set associative D-NUCA. Applications selected from the SPEC CPU2000 suite.

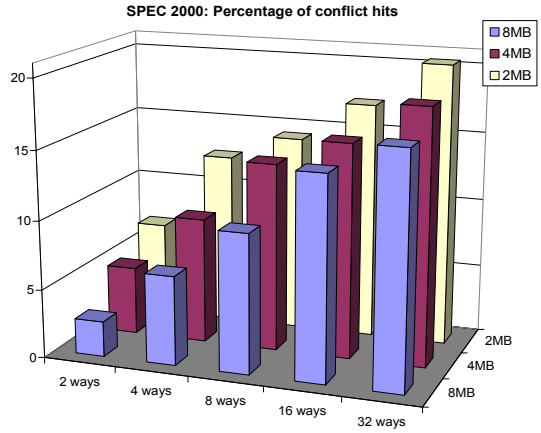


Figure 3. Percentage of conflict hits on global L2 hits for various cache sizes and associativities.

the controller broadcasts this request to all these banks along that row.

To further reduce access latencies, if a hit happens in a row other than the first, the cache line is promoted by swapping it with the line that holds the same position in the next column closer to the controller. The data swap mechanism is implemented writing the migrating data on the destination bank. Before this write, the corresponding data on the destination line must be read to be written back at the originating line. Consequently, every migration implies three additional accesses to the cache banks. This reduces data access latency because most frequently accessed data are dynamically migrated in fastest banks, but increases dynamic energy due to additional bank accesses.

In this work we evidence a phenomenon in the migration mechanism, namely "conflict hit", that can be avoided leading to energy savings with light impacts on performances. In the following we describe the causes of conflict hits, analyze the amount of the phenomenon for various applications and cache geometries, and propose a simple hardware technique which can save up to 17% of dynamic energy for some applications.

2 Related work

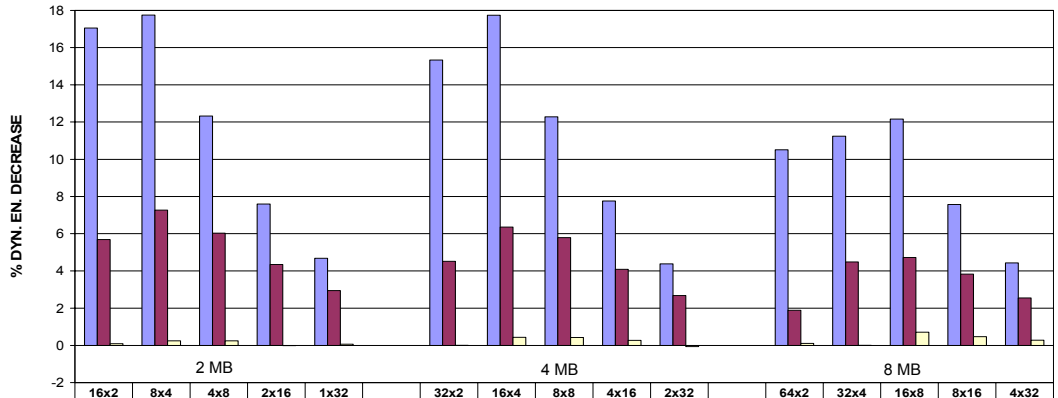
The NUCA paradigm for L2 on-chip caches has been firstly proposed by Kim et al. [2]. Chishti et al. [3] have proposed an optimization scheme, called NuRAPID, aiming at increasing the energy efficiency and the performance of NUCA caches in a single core configuration. An analysis of the various components of static and dynamic power consumption has been performed in [4]. In [5] a lightweight hardware technique which reduces static power consumption for D-NUCA architectures is proposed. The authors have shown how the migration mechanism can be leveraged to dynamically power down less utilized cache ways minimizing the number of additional cache misses. This work, instead, focuses on the dynamic power consumption caused by bank accesses in a D-NUCA architecture, and aims to optimize the migration mechanism in order to make it more energy efficient without impacting on performance. Huh et al. [8] proposed a CMP D-NUCA architecture and evidenced how two or more processors which share a cache line can generate migration patterns similar to the one presented in this work, but a study on the relevance of this phenomenon depending on the application and the architecture is not presented.

3 Analysis of the migration mechanism

Our studies suggest that not all data swaps succeed for moving data near the controller. This is due to concurrent subsequent hits to cache lines placed at the same index in two adjacent banks, and will be further called "conflict hit". Conflict hits cause these two competing data to mutually swap and demote each other, without moving towards the processor.

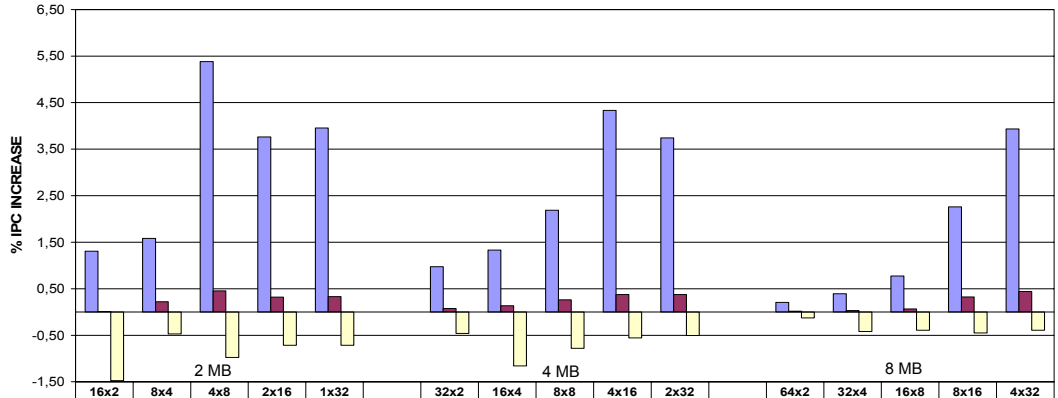
The relevance of conflict hits largely depends on how data are distributed among the banks and on the access patterns generated by the application during execution. We estimated how often conflict hits happen during the execution of several applications and for different L2 D-NUCA cache sizes and geometries. We studied the relevance of the phenomenon for applications from the SPEC CPU2000 suite, selected and simulated following the methodology reported in [2]. We utilized the *sim-alpha*[6] cycle accurate simulator, augmented to model D-NUCA caches and to trace conflict hits, which is based and validated for the Alpha 21264 architecture. The simulator models an Alpha 21264 core attached to 64 KB 2-way set-associative L1 caches. All the cache subsystem has a block size of 64 B. The L2 unified D-NUCA architecture is made up of 64KB banks with an hit latency of 3 cycles. The main memory latency is 300 cycles.

Fig. 2 shows the percentage of conflict hits on total L2 cache hits for 12 applications, evaluated for two different



	16x2	8x4	4x8	2x16	1x32	32x2	16x4	8x8	4x16	2x32	64x2	32x4	16x8	8x16	4x32
best	17,05	17,75	12,32	7,60	4,68	15,33	17,74	12,28	7,76	4,38	10,51	11,24	12,16	7,57	4,43
average	5,68	7,27	6,04	4,34	2,95	4,52	6,36	5,79	4,09	2,68	1,89	4,48	4,72	3,83	2,54
worst	0,09	0,25	0,25	-0,01	0,06	0,01	0,44	0,42	0,28	-0,06	0,11	0,00	0,71	0,47	0,28

Figure 4. SPEC CPU2000: Percentage reduction of dynamic energy applying the exposed technique. D-NUCA cache architectures of 2 MB, 4 MB and 8 MB are shown. "Best" is the value of the benchmark that, for every configuration, has incurred in the higher dynamic energy reduction.



	16x2	8x4	4x8	2x16	1x32	32x2	16x4	8x8	4x16	2x32	64x2	32x4	16x8	8x16	4x32
best	1,31	1,58	5,38	3,76	3,96	0,97	1,33	2,19	4,33	3,74	0,21	0,39	0,77	2,26	3,93
average	0,01	0,22	0,45	0,32	0,33	0,07	0,14	0,26	0,37	0,37	0,02	0,03	0,06	0,32	0,44
worst	-1,47	-0,47	-0,98	-0,72	-0,72	-0,46	-1,16	-0,78	-0,55	-0,51	-0,13	-0,42	-0,39	-0,45	-0,39

Figure 5. SPEC CPU2000: Percentage variation of IPC for the exposed technique. Previously mentioned L2 D-NUCA architectures and benchmarks are evaluated. "Best" is value of the benchmark which has shown the higher IPC increase for every configuration.

L2 D-NUCA cache architectures: a 2 MB cache made up by 16 rows of 2 banks (2 ways) and an 8 MB L2 D-NUCA architecture made up by 4 rows of 32 banks (32 ways). This examples were taken to show the dependency of the phenomenon on cache size and associativity. It can be noticed how the percentage of conflict hits varies sensitively depending on the application, and how highly associative caches are more prone to conflict hits.

Figure 3 reports the average percentage of conflict hits for all the benchmarks considered, as a function of cache size and cache associativity. Apart from noticing how heavily associative caches are more prone to conflict hits, as exposed before, it emerges how with a fixed associativiy the percentage of conflict hits tends slightly to grow as the cache capacity is lowered. This is mainly due to the reduced overall number of cache sets for small caches, which raises the probability for two cache accesses to be performed on the same set and conflict. This evaluations suggest that techniques for conflict hit detection and recovery can be useful to reduce the total number of bank accesses, reducing dynamic energy consumption due to useless migrations.

4 A simple optimization technique

We designed and evaluated a simple technique for reducing useless data swaps due to conflict hits. This technique detects conflicting migrations using a flag bit associated to each line. This bit is set when the line is promoted and subsequently demoted. When a hit occurs on a cache line whose bit is set, the migration does not occur and the bit is reset again. This aims to inhibit subsequent conflicting data swaps on the same couple of blocks in two adjacent cache ways. This technique needs only a bit per cache line and simple additional logic which is not in the latency-critical path. In order to evaluate the energy consumption associated to cache accesses we utilized the Cacti 4.2 tool [7], referring to a 70 nm technology.

As shown in Fig. 4 the adoption of this technique succeeds in reducing block accesses and consequently

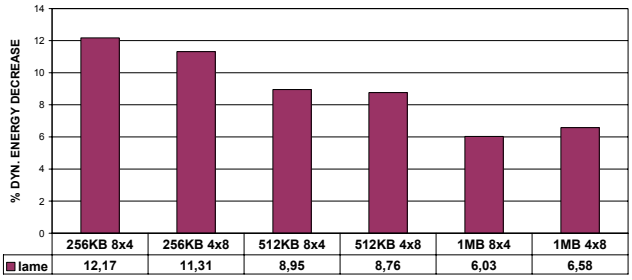


Figure 6. Case study for the lame encoder. For every cache capacity we analyzed a 4 way associative cache (8x4 geometry) and a 8 way associative cache (4x8 geometry)

their dynamic energy consumption, with light impacts on performance. Because of the high variability of the conflict hits amount on the application, the consequent reduction in dynamic energy consumption is more relevant in some applications (up to 17,75%), with an average reduction of 7,27%. for 2 MB architectures, 6,36% for 4 MB architectures and 4,72% for 8MB architectures

It is worth noticing from Fig. 5 that performances on average are slightly better with our technique, because the mechanism is able in some cases to unblock swapping data effectively moving them in faster banks.

5 Case study

Our simulations have shown a high variation across applications of the percentage of conflict hits and the savings in dynamic energy obtained by our technique. This is a typical behavior for a class of phenomena which can lead to optimizations on special purpose environments as embedded systems. Based on this consideration we explored the feasibility of the technique in the embedded environment. Although it is still not clear if there can be a performance advantage in using NUCA architectures instead of standard caches for the embedded environment, because of their reduced size and their different manifacturing technology, adopting a modular approach can be leveraged in power aware custom designs for embedded applications because, in such case, to selectively turn off unused portions of the cache is straightforward [5].

We present here the evaluations performed on a benchmark for the embedded environment, the multimedial encoder *lame* from the MIBench suite. We reparametrized our simulation environment for the embedded case, basing on a 130 nm technology and adopting a size of 16 KB for the L1s. We confronted three architectures for the L2 D-NUCA, with 256KB, 512KB and 1MB capacity respectively, and different geometries, as shown in Fig. 6.

Our results suggest that our policy can be used also in an embedded environment, leading to a maximum saving of 12% in dynamic energy consumption.

6 Conclusions and future work

This paper has shown the feasibility of optimizing the migration mechanism of D-NUCA caches to enhance their dynamic energy efficiency without impacting on performance. We have shown as an extra energy dissipation in migrations can be avoided adopting an easy to implement hardware mechanism. In this initial phase of our research we focused on the portion of dynamic energy dissipated by the L2, but the exposed technique has no harmful effects on the processor or the rest of the memory architecture, since on average the performances are not impacted as well as the amount of L2 misses. It is worth noticing that the technique shown here is orthogonal to other techniques proposed for D-NUCA caches addressing static power consumption [5], hence they can be used in combination in order to enhance the overall energetic efficiency. Our preliminary studies have highlited how a different migration mechanism can modify the distribution of cache hits, and this may cause a positive interference on techniques which leverage this aspect to turn off unused cache ways. Future extensions of this work will hence concern this aspect and we aim to evaluate the combined adoption of our proposed solution with this kind of techniques.

Finally, we plan to analyze the migration mechanism, its energetic impact and its possible enhancements in a multicore environment. We believe that with many cores accessing a shared L2 cache the frequency of conflict hits should raise. In particular, for multiprogrammed loads the opportunity for more conflict hits to happen is due to the higher data traffic injected in the L2 shared cache. Additionally, in multithreaded workloads, the migration mechanism for shared data is more likely to cause data to oscillate [8], thus making techniques similar to the one presented in this work even more profitable.

References

[1] V. Agarwal et al. Clock Rate versus IPC: The End of the Road for Conventional Microarchitectures. *Proc. 27th Int. Symp. on Comp. Arch., pp. 248-259, Vancouver, Canada, June 2000*

[2] C. Kim et al. An Adaptive, Non-Uniform Cache Structure for Wire-Delay Dominated On-Chip Caches. *Proc. 10th Conf. on Architectural Support for Programming Languages and Operating Systems, pp. 211-222, San José, CA, October 2002.*

[3] Z.Chisthi et al. Distance Associativity for High-Performance Energy-Efficient Non-Uniform Cache Architectures. *Proc. 36th Int. Symposium on Microarchitecture, pp 55-66, San Diego, CA, December 2003*

[4] A. Bardine et al. Analysis of Static and Dynamic Energy Consumption in NUCA Caches: Initial Results. *Proc. of the 2007 MEDEA Workshop, pp 105-112, Brasov, Romania, September 2007*

[5] A. Bardine et al. Leveraging Data Promotion for Low Power D-NUCA Caches. *Proc. of. the 11th EUROMICRO Conference on Digital System Design, pp. 307-316, Parma, Italy, September 2008*

[6] R. Desikan et al. Sim-alpha: a validated execution driven alpha 21264 simulator *Tech Report TR-01-23 Dept. of Computer Sciences, Univ. Texas at Austin, 2001*

[7] D.Tarjan et al. Cacti 4.0 *Tech Report HPL-2006-86, 2006, Hp Labs*

[8] Huh et al. A NUCA Substrate for flexible CMP Cache Sharing. *Proc. 19th Int. Conf. on Supercomputing, Cambridge, MA, June 2005*