

L2 cache way prediction techniques for low-power MPSoC design*

Chun-Mok Chung and Jihong Kim

School of Computer Science & Engineering
Seoul National University, Seoul 151-742, Korea
+82-2-880-1861, {chunmok, jihong}@davinci.snu.ac.kr

Abstract This paper proposes way prediction techniques for the set-associative L2 cache of MPSoC to reduce cache energy consumption. Exploiting the sequential access characteristics of an L2 cache, our techniques are based on two prediction modules, the look-ahead buffer and way-affinity table. The look-ahead buffer predicts the way number of the next sequential cache block while the way-affinity table maintains the way number that the most recent access has used. By combining the two modules, we predict correct ways for about 75% of L2 cache accesses, and achieve about 50% reduction in L2 cache energy consumption for an eight-way set-associative L2 cache.

Keywords: cache way prediction, set-associative cache, L2 access pattern, way-affinity, MPSoC

Introduction In designing low-power multiprocessor system-on-chips (MPSoCs), managing the power consumption of an on-chip L2 cache is very important. For most MPSoCs, a large on-chip L2 cache is employed to reduce the performance and power overhead of accessing off-chip memory, often making the L2 cache a dominant power consumer of MPSoCs. On-chip L2 caches usually take a set-associative organization for a higher hit ratio. However, set-associative caches are energy inefficient by its very nature: even though only a single way has a requested data in an n -way set associative cache, all the ways need to be searched simultaneously, thus wasting the energy spent for $(n-1)$ mismatched ways. To improve the energy efficiency of set-associative caches, way prediction techniques have been proposed [1, 2]. In these techniques, a way with the data for the next access is predicted, thus accessing the predicted way only for lower L2 cache energy consumption. In the existing schemes of [1, 2], the most recently used (MRU) way of each set is used as a predicted way for the next access. The MRU heuristic works fine for L1 caches because the L1 caches have a high degree of locality. However, in L2 caches, most of accesses with high locality are filtered by L1 caches, thus making the MRU heuristic less efficient.

In this paper, we propose new way prediction techniques which are optimized for the access characteristics of an L2 cache. Our proposed techniques are based on two observations of L2 cache access patterns. First, many L2 cache accesses are sequential. We call this *the sequentiality property*. The sequential accesses can be found in fetching instructions and loading/storing data blocks. Block-level processing, which is common for multimedia applications or matrix operations, requires multiple sequential L2 cache accesses when the data block size is larger than the L2 cache block size. Fig. 1 shows an example of set-associative L2 cache accesses for a data block load where the data block size is four times bigger than the L2 cache block size. The cache blocks in the same way of sequential sets tend to be accessed successively, because the cache blocks are mapped to the same data block. We call this *the way-affinity property*. Our techniques take account of the sequentiality property and way-affinity property in deciding the next way to be accessed. As shown in Fig. 2, we assume that our target MPSoCs have shared unified L2 caches.

Way prediction techniques for L2 cache Fig. 3 shows an L2 cache with our way prediction modules. The way prediction modules consist of the look-ahead buffer (LAB), the way-affinity table (WAT), and a mux. LAB, which predicts the next way based on the sequentiality property, stores the way numbers of the *next* cache blocks. After an L2 cache access, the way number of the next sequential cache block is looked-ahead and stored in LAB, for use in the next L2 cache access. LAB is implemented as a small set-associative cache whose set size is equal to the number of processors. The way size of LAB is configurable. The address of cache block is used as the tag. If address is matched, LAB returns a way number. On the other hand, WAT, which exploits the way-affinity property, stores the way numbers of the *last* L2 cache accesses. Since there is one last access for each processor, WAT is implemented using a register array whose size is equal to the number of processors. Each register stores a way number.

When an L2 cache access is requested, the processor ID and address are sent to both LAB and WAT at the same time. LAB searches the way number using the processor ID and address while WAT reads the last ac-

* This work was supported by the Brain Korea 21 Project and the Korea Science and Engineering Foundation grant funded by the Korea government (MEST) (R0A-2007-000-20116-0, R33-2008-000-10095-0). The ICT at Seoul Nat'l Univ. provided research facilities for this study.

cessed way of the processor. The mux determines the way from the search results of LAB and WAT. Since LAB always returns a correct way if matched while WAT may return an incorrectly predicted way number, we choose the way number from LAB over one from WAT when LAB finds a way for the requested L2 cache access. The L2 cache accesses only the predicted way. For an L2 cache hit, the way number is sent to WAT, which updates the last way of the requesting processor. In case of an L2 cache miss, the way number of a new cache block is sent to WAT. After an L2 cache access, the L2 cache tags are rechecked to find the way number of the next block. If the next block is found, the address and its way number are stored in LAB.

With the proposed way prediction techniques, three steps are responsible for most of dynamic energy consumed in obtaining data from the L2 cache, which are the way prediction step, the L2 cache access itself, and the way prediction module update step. Thus, the average energy consumption AE_{WP-L2} of an n -way set-associative L2 cache is the sum of the dynamic energy consumption of these three steps ($E_{predict}$, AE_{L2} , and E_{update}) and average leakage energy consumption $AE_{leakage}$. The energy models of four parts are expressed in Table I depending on the way prediction techniques (for notations, see Table II). Fig. 4 compares how the average energy consumption changes as the prediction hit ratio varies for the 8-way set-associative L2 cache. (Values were normalized to the base case of an original 8-way set-associative cache.) For LAB, WAT, and LAB+WAT, the break-even prediction hit ratios are 0.1, 0.2, and 0.3, respectively, indicating a high potential of saving energy when the way prediction techniques work reasonably. Depending on the prediction hit ratio, the best technique is changed. If the prediction hit ratio is lower than 0.7, LAB is the best choice, because it has the least miss penalty. However, if the prediction hit ratio is higher than 0.7, WAT is better than others, because it consumes the least energy in the way prediction module update step.

Experiments We performed several experiments using a modified MPSoC simulator based on CATS [3]. We have added to CATS the proposed way prediction modules with an appropriate energy model. As target parallel applications, we have used benchmark programs in SPLASH-2 and MiBench. Table III summarizes the parameters used in our experiments. We configured CATS to be similar to ARM11 MPCore [4]. The energy parameters of Table III were derived from CACTI [5] and HotLeakage [6] with a 130 nm technology which was used for recent low-power MPSoCs (e.g., ARM11 MPCore).

Fig. 5 shows the prediction hit ratios of way prediction techniques. As shown in Fig. 5, the MRU heuristic does not work well for predicting the next way in the L2 cache. LAB+WAT outperforms MRU all the benchmark programs except for `radix`. LAB+WAT also predicted better than LAB or WAT, because it usually takes the better prediction of two prediction results. In the case of `radix`, because most L2 accesses were random accesses and few accesses were sequential accesses, the prediction hit ratios of our techniques were lower than MRU. On average, LAB+WAT predicted correct ways for about 75% of L2 cache accesses. Fig. 6 shows the L2 cache energy consumption for the proposed way prediction techniques. Values were also normalized to the energy consumption of the baseline 8-way set-associative L2 cache (`Base` in the figure). The higher the prediction hit ratio, the lower the energy consumption. Therefore, our techniques consumed less energy than MRU, except for the case of `radix`. Although LAB+WAT required more leakage power and more dynamic energy to manage the way prediction modules than LAB or WAT, it consumed the least energy, because it had the highest prediction hit ratio and reduced the most L2 cache access energy. On average, LAB+WAT decreased L2 cache energy consumption to 50% of the baseline L2 cache.

Conclusions We have described new way prediction techniques for a set-associative shared L2 cache of MPSoC. Our techniques exploit two properties of L2 cache accesses, the sequentiality property and the way-affinity property using a look-ahead buffer and a way-affinity table, respectively. Experimental results based on simulations show that the proposed techniques accurately predict the next way number for most L2 cache accesses, thus saving about 50% of L2 cache energy consumption over a conventional cache.

- [1] K. Inoue, T. Ishihara, and K. Murakami, "Way-predicting set-associative cache for high performance and low energy consumption," in Proc. of ISLPED, pp. 273-275, 1999.
- [2] M. Powell, A. Agarwal, T. Vijaykumar, B. Falsafi, and K. Roy, "Reducing set-associative cache energy via way-prediction and selective direct-mapping," in Proc. of MICRO, pp. 54-65, 2001.
- [3] D. Kim, S. Ha, and R. Gupta, "CATS: cycle accurate transaction-driven simulation with multiple processor simulators," in Proc. of DATE, pp. 749-754, 2007.
- [4] ARM, "MPCore multiprocessor technical reference manual," ARM DDI 0360A, 2005.
- [5] D. Tarjan, S. Thoziyoor, and N. P. Jouppi, "CACTI 4.0," HPL-2006-86, HP Lab. Palo Alto, 2006.
- [6] Y. Zhang, D. Parikh, K. Sankaranarayanan, K. Skadron, and M. Stan, "HotLeakage: a temperature-aware model of subthreshold and gate leakage for architects," TR-CS-2003-05, U. of Virginia, Dept. of CS, 2003.

Table I Energy models of four parts depending on way prediction techniques.

	LAB	WAT	LAB+WAT
$E_{predict}$	E_{LAB}	E_{WAT}	$E_{LAB} + E_{WAT}$
AE_{L2}	$r_{hit}E_{1-way} + r_{miss}E_{n-way}$	$r_{hit}E_{1-way} + r_{miss}(E_{1-way} + E_{n-way})$	
E_{update}	$E_{tag} + E_{LAB}$	E_{WAT}	$E_{tag} + E_{LAB} + E_{WAT}$
$AE_{leakage}$	$AL_{LAB} \times (P_{LAB} + P_{n-way})$	$AL_{WAT} \times (P_{WAT} + P_{n-way})$	$AL_{LAB+WAT} \times (P_{LAB} + P_{WAT} + P_{n-way})$

Table II Definition of symbols in the energy model.

Symbol	Definition
$E_{LAB}, E_{WAT}, E_{1-way}, E_{n-way}, E_{tag}$	Dynamic energy values of LAB, WAT, 1-way L2 cache, n -way L2 cache, and L2 cache tag lookup
$P_{LAB}, P_{WAT}, P_{n-way}$	Leakage power values of LAB, WAT, and n -way L2 cache
$AL_{LAB}, AL_{WAT}, AL_{LAB+WAT}$	Average L2 cache latencies with LAB, WAT, and LAB&WAT
r_{hit}, r_{miss}	Way prediction hit and miss ratio

Table III Configuration of simulation parameters.

Parameter	Configuration		
Processors	4 ARM cores (550MHz)		
L1 cache (I, D)	Private, 32KB, 4-way, 32B block		
L2 cache	Shared, 1MB, 8-way, 32B block		
LAB	4 x 4	WAT	1 x 4
E_{LAB}	4.6419 pJ	E_{WAT}	0.0888 pJ
E_{MRU}	2.4265 pJ	E_{1-way}	0.1108 nJ
E_{8-way}	0.5524 nJ	E_{tag}	0.0374 nJ
P_{LAB}	38.1625 nW	P_{WAT}	3.8549 nW
P_{MRU}	370.066 nW	P_{8-way}	113448 nW
L_{8-way}	8 cycles	L_{1-way}	4 cycles
$L_{LAB}, L_{WAT}, L_{MRU}$	1 cycle		

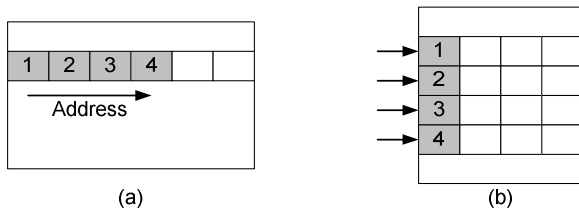


Fig. 1 (a) Data block (4x larger than L2 cache block size) in address space. (b) 4-way L2 cache accesses for a data block load.

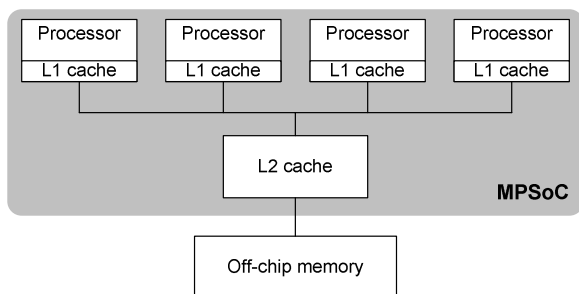


Fig. 2 Baseline MPSoC with shared L2 cache.

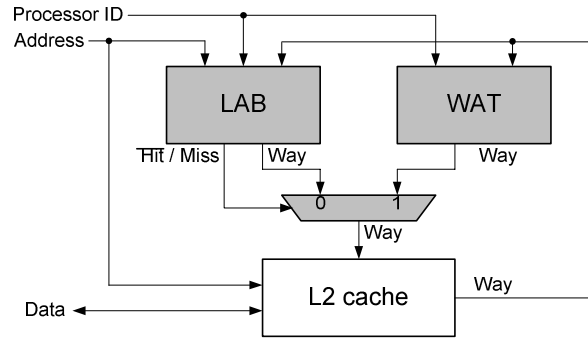


Fig. 3 L2 cache with way prediction module.

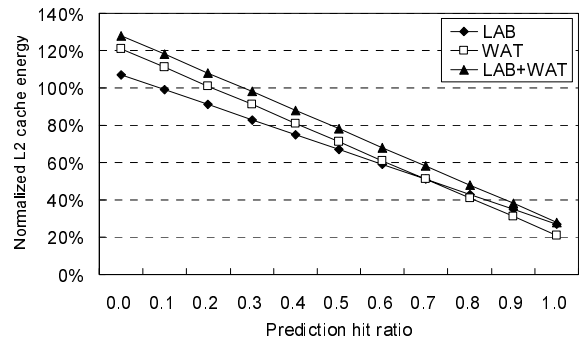


Fig. 4 L2 cache energy consumption according to prediction hit ratio.

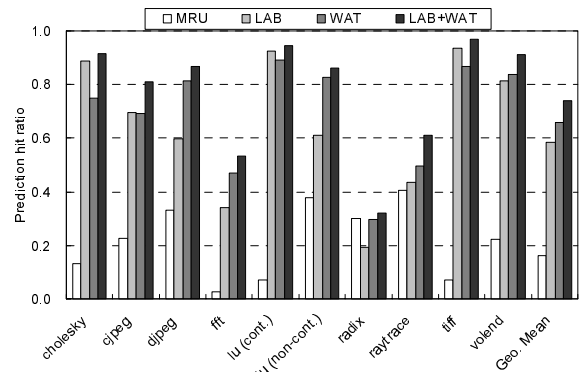


Fig. 5 Prediction hit ratios of way prediction techniques.

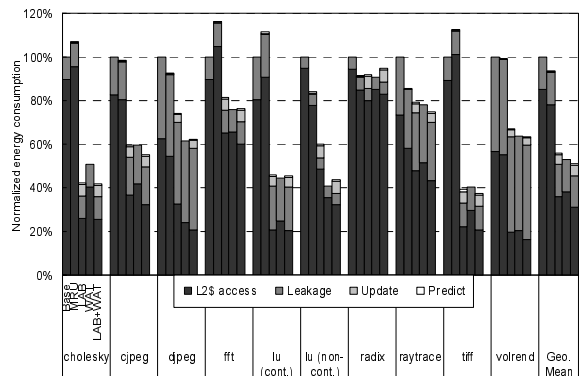


Fig. 6 L2 cache energy consumption with way prediction techniques.