Design of Hybrid Second-Level Caches

Alejandro Valero, Julio Sahuquillo,
Salvador Petit, Pedro López, and José Duato

Abstract—In recent years, embedded Dynamic Random-Access Memory (eDRAM) technology has been implemented in last-level caches due to its low leakage energy consumption and high density. However, the fact that eDRAM presents slower access time than Static RAM (SRAM) technology has prevented its inclusion in higher levels of the cache hierarchy. This paper proposes to mingle SRAM and eDRAM banks within the data array of second-level (L2) caches. The main goal is to achieve the best trade-off among performance, energy, and area. To this end, two main directions have been followed. First, this paper explores the optimal percentage of banks for each technology. Second, the cache controller is redesigned to deal with performance and energy. Performance is addressed by keeping the most likely accessed blocks in fast SRAM banks. In addition, energy savings are further enhanced by avoiding unnecessary destructive reads of eDRAM blocks. Experimental results show that, compared to a conventional SRAM L2 cache, a hybrid approach requiring similar or even lower area speedups the performance on average by 5.9%, while the total energy savings are by 32%. For a 45nm technology node, the energy-delay-area product confirms that a hybrid cache is a better design than the conventional SRAM cache regardless of the number of eDRAM banks, and also better than a conventional eDRAM cache when the number of SRAM banks is an eighth of the total number of cache banks.

Index Terms—Cache memories, eDRAM, energy-aware systems, hybrid systems, SRAM

1 Introduction

Technologies

Multilevel on-chip cache hierarchies have been typically built with Static Random-Access Memory (SRAM) technology, which is the fastest existing electronic memory technology. Nowadays, alternative technologies are being used and explored since SRAM presents important shortcomings like low density and high leakage currents, which are proportional to the number of transistors. These shortcomings have become meaningful design challenges, in such a way that it is unlikely the implementation of future cache hierarchies with only SRAM technology, especially in the context of Chip Multi-Processors (CMPs).

New advances in technology enable to build caches with other technologies, like embedded Dynamic RAM (eDRAM), Magnetic RAM (MRAM), or Phase-change RAM (PRAM). Table 1 summarizes some properties of these technologies. Embedded DRAM presents high density and low leakage power, and has been already used to build large Last-Level Caches (LLCs) in some commercial processors [1] [2] [3] [4] [5]. This capacitor-based memory integrates trench DRAM storage cells into a logic-circuit technology [6], which reduces significant area over typical 6-transistor bit cells used in SRAM. More precisely, compared to SRAM, eDRAM increases the storage capacity by a $3x$ factor for a given silicon area, thus giving important area savings, especially for large LLCs.

Regarding other technologies like MRAM or PRAM, manufacturing constraints prevent from mingling them in conventional two-dimensional (2D) chips. In addition, the low speed and dynamic energy consumed by these technologies, in particular for write operations, suggest that they are more appropriate for main memory storage instead of caches.

Embedded DRAM-based caches are not normally implemented in the highest levels of the cache hierarchy like first-level (L1) or second-level (L2) caches since eDRAM technology is slower than SRAM and performance is more sensitive to the latency of these levels in current microprocessors. For example, both Intel Haswell microarchitecture [1] and IBM POWER7 [5] implement SRAM-based private 256KB L2 caches with a 10-cycle access time. Nevertheless, as each technology presents both advantages and shortcomings, there are several proposals that combine SRAM and eDRAM technologies in different microprocessor components such as L1 data caches [7], Non-Uniform Cache Architectures (NUCAs) [8] [9], and register files [10].

TABLE 1
Features of different memory technologies.

Feature	SRAM	eDRAM	MRAM	PRAM
Speed	fast	slow	very slow	very slow
Density	low	high	high	very high
Leakage	high	low	very low	very low
Refresh	no	yes	no	no
Dest. reads	no	yes	no	no

- *The authors are with the Department of Computer Engineering, Universitat Politècnica de València, Camí de Vera s/n, 46022 Valencia, Spain. Part of this work was done while A. Valero was in the Department of Electrical and Computer Engineering at Northeastern University, Boston, MA, USA.*
E-mail: alvabre@gap.upv.es, {jsahuqui, spetit, plopez, jduato}@disca.upv.es

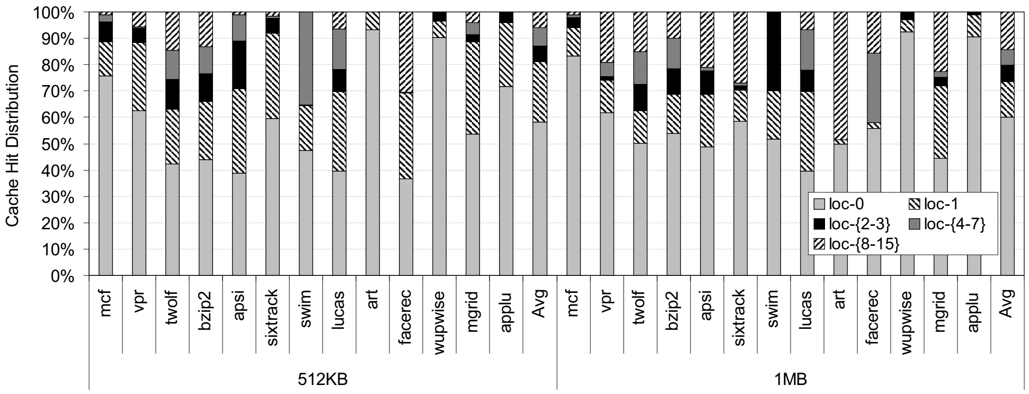


Fig. 1. Percentage of cache hits across the locations of the LRU stack.

Cache Block Behavior

In [7], authors present a hybrid eDRAM/SRAM L1 data cache design that leverages the fact that such caches concentrate most of their hits (e.g., more than 90%) in the Most Recently Used (MRU) blocks. Consequently, performance can be sustained by implementing only a cache way with SRAM technology and force this cache way to store the MRU block.

However, data locality in L2 caches is much poorer than it is in L1 caches, thus this implementation might yield to unacceptable performance in L2 caches. Figure 1 plots the distribution of cache hits in a conventional 512KB 16-way and 1MB 16-way L2 caches for the SPEC CPU2000 benchmarks [11][1]. This distribution has been obtained for the LRU stack with the aim of analyzing if hits concentrate only in a few blocks at the top of the stack. Label *loc-0* refers to the location storing the MRU block, while *loc-15* is the position storing the Least Recently Used (LRU) block. Label *loc*-{*x*-*y*} denotes hits falling in between locations *x* and *y* of the stack, both inclusive.

As observed, hits are distributed among different locations of the LRU stack in L2 caches. Although the distributions are clearly skewed to the first ways, the 512KB cache requires half of the cache ways (8 ways) to cover by 95% of the cache hits, while this percentage drops down to 85% for the 1MB cache. Notice too that, for the 512KB cache, the MRU way captures *only* around 50% of the cache hits in 7 of 13 applications. The number of applications grows up to 9 for the 1MB cache. On average, the percentage of hits in the MRU way is by 60% for both cache sizes. Thus, implementing only that way with SRAM technology would yield to unacceptable performance.

Proposal

Taking into account the previous observation, this paper proposes a hybrid L2 cache that mingles SRAM and eDRAM technologies to provide by design leakage energy and area savings with respect to typical SRAM caches. The cache controller is designed to address both performance and energy. To achieve minimal performance losses over SRAM caches, two main design choices have been taken: i) the most likely referenced blocks are placed in fast SRAM banks, and ii) the optimal percentage of fast SRAM banks is estimated. On the other hand, to further increase energy savings two main design choices have been studied: i) avoid unnecessary destructive eDRAM reads, and ii) estimate the optimal percentage of low-leakage eDRAM banks. Notice that the mentioned second design choices imply a trade-off between them, since a higher percentage of SRAM banks means better performance but at expenses of energy and area. That is, a pure SRAM cache presents the maximum performance and a pure eDRAM cache the minimum energy and the lower area. The optimal hybrid eDRAM/SRAM cache design falls in between these extremes and pursues to minimize performance losses, energy consumption, and area over SRAM caches, and to maximize performance over eDRAM caches. This paper presents a detailed study covering performance, energy (split into leakage and dynamic) and area for various storage capacities. Results are analyzed from both area and storage capacity points of views. A preliminary study of the results can be found in [12].

Experimental results show that, compared to a conventional SRAM L2 cache, a hybrid cache with similar or even lower area improves performance, on average, by 5.9%, while the total energy reduction is by 32%. For a 45nm technology node, the performance, energy, and area trade-off analysis reveals that, on average, a hybrid cache is a better design than a conventional SRAM cache regardless of the percentage of eDRAM banks, and also better than a conventional eDRAM cache when implementing the eighth part of its banks with SRAM technology. In addition, the energy-delay-squared product shows that both hybrid designs with an eighth and a quarter of their banks built with SRAM technology are better design options than conventional SRAM and eDRAM caches.

The rest of this paper is organized as follows. Section 2 presents the design of the proposed hybrid L2 caches. Section 3 analyzes the area, energy and power consumption, performance, energy-delay-area product, and energy-delay squared product achieved by the proposed caches. Section 4 summarizes the related work, and finally, conclusions are given in Section 5.

[1] Those applications exhibiting an L2 hit ratio greater than 85% in both 512KB and 1MB caches were skipped for this study.

2 Hybrid L2 Cache Design

We assume that each cache bank stores a pair of ways, which results in an L2 cache with 8 banks for the studied 16-way caches. This number of banks is reasonable and it is common to find other cache designs in the literature with much more banks and the same or lower bank storage capacity [5] [13]. Nevertheless, such a number of banks can be reduced by implementing the eDRAM banks with more than two ways. Notice that this has a minimal impact on performance and energy since the design ensures that eDRAM banks are rarely accessed, thus minimizing their bank contention (see Section 3.3).

As each bank of the data array is built with either SRAM or eDRAM technology, several hybrid cache configurations can be implemented. Table 2 summarizes the studied hybrid design choices, specifying the number of SRAM and eDRAM ways and banks of each configuration and the ratio of SRAM banks. The conventional schemes, the pure SRAM (16S) and the pure eDRAM (16D), have all their banks implemented with SRAM and eDRAM technology, respectively. The tag array is built with SRAM cells regardless of the cache scheme, since it is much smaller than the data array. Thus, much lower energy and area benefits can be obtained with this structure. Moreover, implementing it with eDRAM technology can significantly affect the cache access time.

2.1 Accessing the Hybrid Cache

Conventional cache designs usually overlap the access of the tag array with that of the data array to make the access time shorter. However, this might yield to energy wasting since all the cache ways are accessed in parallel. Many research work has addressed this shortcoming by predicting the cache way that contains the target data [14] [15] [16]. These approaches usually perform well in L1 caches given the high data locality exhibited in this cache level. Unfortunately, data locality is much less predictable in L2 caches. Because of this reason, the design proposed in this work predicts several ways (instead of only one) that are accessed in a first stage, similarly to as done in the L2 caches of the IBM POWER7 processor [5].

The access is split into two stages as depicted in Figure 2. In the first stage, the tag array and all the SRAM banks (SRAM data array) are accessed in parallel. If the requested data are stored in an SRAM way, the access time of the hybrid cache is as fast as a hit in a conventional SRAM cache and the second stage is skipped (i.e., no eDRAM way is accessed). This mechanism allows the hybrid cache to avoid unnecessary destructive reads in the eDRAM data array. On a miss in the SRAM data array but a hit in a tag associated to an eDRAM way, only the target eDRAM way is accessed in a second stage. In such a case, the access time includes the tag comparison plus the access to the eDRAM data. On a cache miss, no eDRAM way is accessed and the requested data are fetched from the main memory.

TABLE 2
Conventional and hybrid caches with their number of ways, banks, and ratio (%) of SRAM banks.

Cache config.	SRAM ways	eDRAM ways	SRAM banks	eDRAM banks	SRAM ratio
16S	16	0	8	0	100
8S-8D	8	8	4	4	50
4S-12D	4	12	2	6	25
2S-14D	2	14	1	7	12.5
16D	0	16	0	8	0

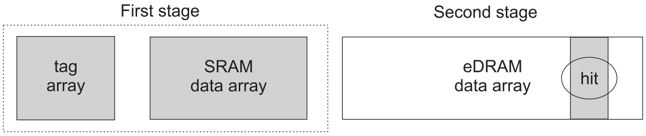


Fig. 2. Diagram of the hybrid cache access. Dark boxes represent the accessed parts of the cache. The second stage is performed only on a hit in an eDRAM way detected in the first stage.

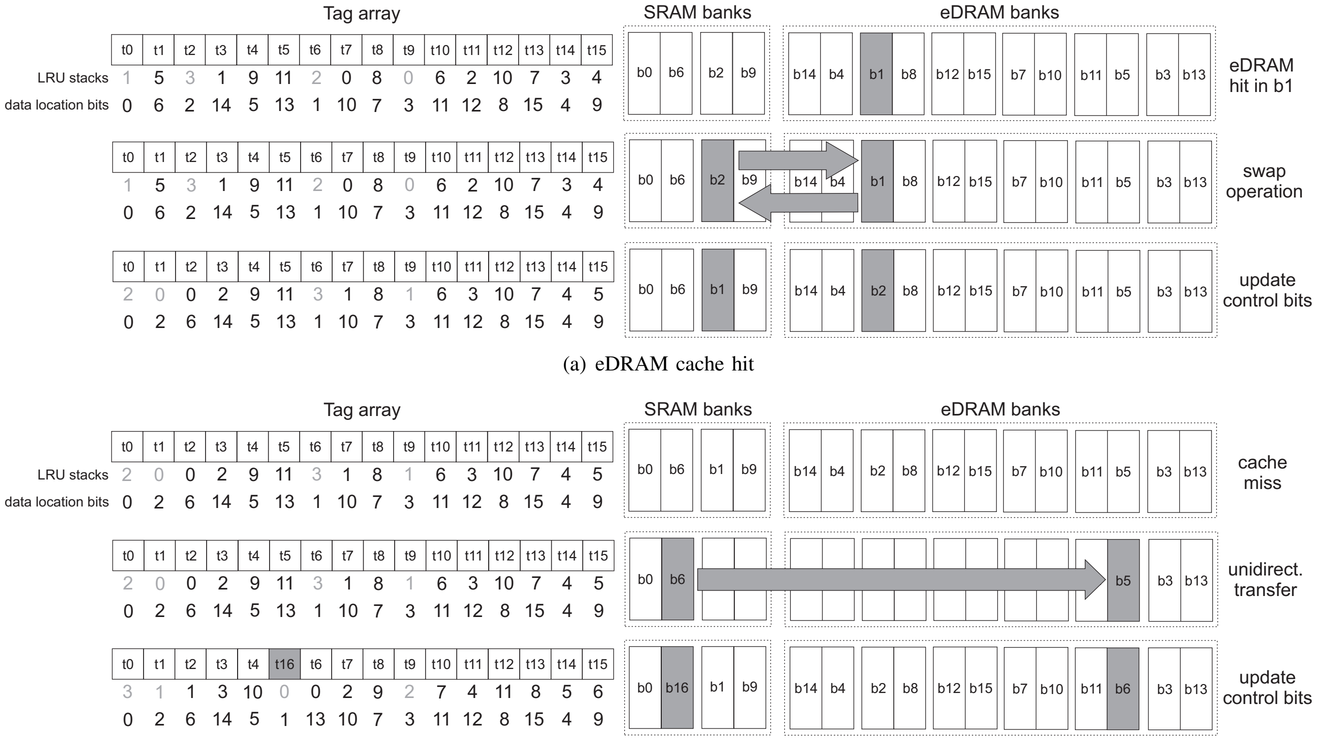
2.2 Keeping the Last Accessed Blocks in SRAM Banks

To keep the MRU data in fast SRAM banks, the cache controller manages a swap operation between SRAM and eDRAM banks. To properly select the blocks to be transferred, each SRAM and eDRAM data arrays maintain its own LRU stack, which allows reducing the number of LRU control bits.

The design assumes that tags are not swapped. Instead, four control bits per tag (*data location bits*) are needed to maintain the relationship between tags and ways in 16-way caches. Notice that accessing these control bits is not in the critical path since they are read together with the tag array and all the SRAM ways during the first stage.

Figure 3(a) and Figure 3(b) illustrate the 4S-12D cache configuration with the actions carried out on an eDRAM hit and a cache miss, respectively, to keep the MRU data in the SRAM banks. The examples show a possible set of values for the data location bits (values from 0 to 3 represent SRAM locations and 4–15 refer to eDRAM ones) and the LRU stacks in the tag array. Grey and black colors in the LRU counters refer to the stack of the SRAM and the eDRAM data array, respectively.

On an eDRAM hit, the requested eDRAM block (labeled as *b1*) is transferred from its eDRAM bank to the SRAM bank that holds the LRU block of the SRAM data array (location 2, block *b2*), which in turn is moved to the target eDRAM bank. To properly perform this swap operation, block *b1* is temporarily placed in an auxiliary buffer associated to the eDRAM bank, while block *b2* moves to this bank. Finally, block *b1* is transferred from the auxiliary buffer to the SRAM bank. After the swap operation, the involved blocks are set as the MRU ones of each data array by updating both LRU stacks, and the data location bits are exchanged. Note that the swap operation is not in the critical path since block *b1* is delivered to the processor as soon as it is read (i.e., before starting the swap process). However, subsequent accesses to the two involved banks in the swap operation are blocked until the data transfer is


Tag array
SRAM banks
eDRAM banks
t0 t1 t2 t3 t4 t5 t6 t7 t8 t9 t10 t11 t12 t13 t14 t15
LRU stacks 1 5 3 1 9 11 2 0 8 0 6 2 10 7 3 4
data location bits 0 6 2 14 5 13 1 10 7 3 11 12 8 15 4 9
b0 b6 b2 b9
b14 b4 b1 b8 b12 b15 b7 b10 b11 b5 b3 b13
eDRAM hit in b1
1 5 3 1 9 11 2 0 8 0 6 2 10 7 3 4
0 6 2 14 5 13 1 10 7 3 11 12 8 15 4 9
swap operation
2 0 0 2 9 11 3 1 8 1 6 3 10 7 4 5
0 2 6 14 5 13 1 10 7 3 11 12 8 15 4 9
b0 b6 b1 b9
b14 b4 b2 b8 b12 b15 b7 b10 b11 b5 b3 b13
update control bits

(a) eDRAM cache hit

Tag array
SRAM banks
eDRAM banks
t0 t1 t2 t3 t4 t5 t6 t7 t8 t9 t10 t11 t12 t13 t14 t15
LRU stacks 2 0 0 2 9 11 3 1 8 1 6 3 10 7 4 5
data location bits 0 2 6 14 5 13 1 10 7 3 11 12 8 15 4 9
b0 b6 b1 b9
b14 b4 b2 b8 b12 b15 b7 b10 b11 b5 b3 b13
cache miss
b0 b6 b5 b3 b13
unidirect. transfer
t0 t1 t2 t3 t4 t16 t6 t7 t8 t9 t10 t11 t12 t13 t14 t15
3 1 1 3 10 0 0 2 9 2 7 4 11 8 5 6
0 2 6 14 5 1 13 10 7 3 11 12 8 15 4 9
b0 b16 b1 b9
b14 b4 b2 b8 b12 b15 b7 b10 b11 b6 b3 b13
update control bits

(b) Cache miss

Fig. 3. Involved actions in an eDRAM hit and a miss to keep the MRU data in SRAM banks.

completed. Such a bank contention has been considered in the evaluation section.

On a cache miss, the LRU block of the eDRAM data array (location 13, block *b5*) is selected for replacement. The cache controller triggers a unidirectional transfer from the SRAM LRU block (location 1, block *b6*) to the eDRAM bank that contains the victim block, while the data block being fetched from the main memory (block *b16*) is placed in the SRAM bank.

In order to allow performing multiple swap operations in parallel, the hybrid caches include as many auxiliary buffers as eDRAM banks. The area overhead introduced by these buffers is negligible [17]. For the 512KB 2S-14D hybrid design, which is the worst-case configuration regarding swap overhead, the data array overhead is only by 0.00039%.

A cache block is neither evicted from the SRAM data array nor moves to the eDRAM array until the block becomes the LRU and it is selected to be swapped. Before being evicted from the SRAM array, a block always resides in the same SRAM bank. In other words, an SRAM hit does not imply any data movement between banks.

Remark that the proposed hybrid cache resembles a design with two exclusive SRAM and eDRAM caches. However, for the same storage capacity, the hybrid design presents important advantages. First, the exclusive caches serialize the access to the associated tag arrays, which would damage the performance. Second, some resources like the decoder, wordlines, and the tag array can be shared in the hybrid cache, which yields to area benefits compared to the split caches. Third, the data transfers between the exclusive caches, which are equivalent to swap operations in the hybrid design, consume more energy because not only the data are transferred but also the tag information.

2.3 Distributed Refresh

Although swapping eDRAM and SRAM data on an eDRAM hit avoids refreshing the accessed eDRAM contents, data in eDRAM banks that are not accessed for long may be lost since capacitors lose their contents with time. Merely losing eDRAM contents will hurt the performance because of these data, if required again, must be fetched from the main memory. To avoid such situations, refresh operations should be performed for eDRAM blocks both in hybrid caches and in the pure eDRAM 16D scheme before capacitors lose the stored value (i.e., before their retention time expire).

Retention time depends on eDRAM capacitance. In this work, we consider eDRAM cells implemented with trench capacitors [18] with a 10fF capacitance, which corresponds to a retention time of 190K processor cycles for a 3GHz processor [17] [19]. In order to mitigate the refresh penalty, we assumed a distributed refresh interleaved among banks following a round-robin policy, where each eDRAM block is regularly refreshed. The period between two consecutive refresh operations is established as the retention time divided by the number of eDRAM blocks. This guarantees that all the eDRAM blocks are refreshed before their retention time expire.

3 Experimental Evaluation

This section presents the simulation environment used to evaluate area, energy, and performance of the studied schemes. The hybrid caches have been modeled on top of an extensively modified version of the SimpleScalar simulation framework [20]. The simulation results include the execution time of the applications and the generated memory events (i.e., cache hits, misses, swaps, writebacks, and refreshes) required to estimate leakage and dynamic energy, respectively. The cache controller models bank conflicts and contention due to all these memory events in hybrid and pure eDRAM caches.

Leakage, dynamic energy per access type, area, and timing values were estimated for a 45nm technology node and 3GHz processor frequency with CACTI 5.3 [17] [21], which includes an analytical model for caches implemented either with SRAM or eDRAM banks. The overall energy was calculated combining the results of both simulators. We assumed the ITRS high-performance device type for the SRAM banks and the logic process-based DRAM for the eDRAM banks. Compared to high-performance devices, implementing the SRAM banks with low-leakage devices significantly enlarges their access time (from a 1.32*x* to a 1.37*x* factor for the studied caches), which would induce severe performance degradation with respect to high-performance SRAM L2 caches like those implemented in the IBM POWER7 processor [5] [22]. The focus of this work is on saving energy while sustaining the IPC with respect to high-performance SRAM L2 caches.

Experimental results were performed configuring the SimpleScalar for the Alpha ISA and running the SPEC CPU2000 benchmarks with the *ref* input set. Statistics were collected simulating 500M instructions after skipping the initial 1B instructions. Table 3 summarizes the main architectural parameters used throughout the experiments. For the 512KB cache, the access time of the SRAM and eDRAM banks is 1.76ns and 2.73ns, respectively. Doubling the cache capacity (1MB cache) implies a higher access time (by 1.90ns and 2.84ns for SRAM and eDRAM banks, respectively). However, for a given memory technology, the access time in cycles is the same for both 512KB and 1MB caches since the increase is masked when the access time is rounded up to processor cycles. Finally, the tag array access time is 0.44ns and 0.60ns for 512KB and 1MB caches, respectively.

TABLE 3
Machine parameters.

Microprocessor core	
Issue policy	Out of order
Branch predictor type	Hybrid gshare/bimodal: gshare has 14-bit global history plus 16K 2-bit counters. Bimodal has 4K 2-bit counters. Choice predictor has 4K 2-bit counters
Branch predictor penalty	10 cycles
Fetch, issue, commit width	4 instructions/cycle
ROB size (entries)	128
# Int/FP ALUs	4/4
Memory hierarchy	
L1 instruction cache	64B-line, 16KB, 2-way, 2 cc
L1 data cache	64B-line, 16KB, 2-way, 2 cc
L2 unified cache	64B-line, 512KB/1MB, 16-way
L2 access time	Tag array: 2 cycles SRAM banks: 6 cycles eDRAM banks: 9 cycles
Memory access time	100 cycles

3.1 Area

The hybrid caches and the pure eDRAM cache require less area than the conventional SRAM cache since eDRAM cells have higher density than SRAM cells. The area values of SRAM and eDRAM cells obtained with CACTI are $0.296\mu m^2$ and $0.062\mu m^2$, respectively. To calculate the area of the data array, we first obtained the area of an SRAM and an eDRAM bank. Then, these values were accumulated according to the number of banks in each cache configuration. The presented results include not only the area of the data array but also the tag array and the cache controller logic (e.g., decoders, multiplexers, and sense amplifiers). In addition, the area overhead due to the control bits to keep the mapping between tags and ways as well as the area of the auxiliary buffers required to perform swaps have been taken into account in the hybrid caches.

Figure 4 plots the cache area (in mm^2) of the studied caches. Remember that no area benefits come from the tag array since it is built with SRAM technology regardless of the cache configuration. As can be seen, the higher the number of eDRAM banks of the data array the larger the area savings. Compared to the 16S cache with the same capacity, the 16D cache is the scheme that most area reduces (by 47% in the 512KB cache), closely followed by the 2S-14D hybrid cache (41%). These area reductions are larger for the 1MB cache size. In this case, area savings are up to 46% for the 2S-14D approach.

In Figure 4 it can be appreciated that 4S-12D, 2S-14D, and 16D (highlighted with the circle) 1MB eDRAM-based configurations present area savings with respect to the pure 512KB SRAM cache despite their storage capacity is twice as large. Based on this observation, it makes sense to compare different approaches not only on a capacity basis but also on an area basis. To perform the analysis on the basis of area, we compare the highlighted 1MB eDRAM-based caches against the conventional 512KB SRAM cache. Note that some of these 1MB caches significantly reduce

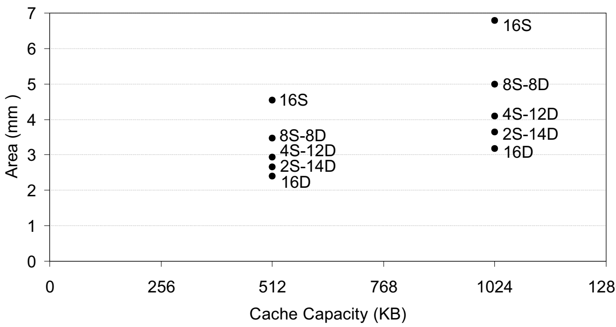


Fig. 4. Area (in mm^2) of the analyzed caches. The circle groups those 1MB eDRAM-based configurations with less area than the 512KB SRAM cache.

TABLE 4
Leakage and dynamic consumption with the overall reduction (%) compared to the pure SRAM cache.

	512KB					1MB				
Consumption	16S	8S-8D	4S-12D	2S-14D	16D	16S	8S-8D	4S-12D	2S-14D	16D
Total leakage (mJ)	70.7	42.6	28.2	21.0	13.8	120.2	70.2	44.4	31.6	18.9
Tag array (mJ)	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
SRAM hits (mJ)	4.9	2.3	1.1	0.5	0	8.7	3.5	1.6	0.7	0
eDRAM hits (mJ)	0	0.2	0.3	0.3	4.9	0	1.2	0.9	0.7	9.5
Swaps (mJ)	0	0.9	1.0	1.1	0	0	1.5	1.6	1.8	0
Writebacks (mJ)	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3
Misses (mJ)	7.5	4.2	2.5	1.7	7.4	8.6	4.8	2.9	1.9	9.3
Refreshes (mJ)	0	1.1	1.7	2.0	13.7	0	3.2	4.9	5.7	24.7
Total dynamic (mJ)	13.0	9.2	7.1	6.0	26.5	17.8	14.6	12.5	11.4	44.1
Total energy (mJ)	83.7	51.9	35.3	27.0	40.3	138.0	84.8	56.9	43.0	63.1
Total reduction (%)	–	38.0	57.9	67.7	51.8	–	38.5	58.8	68.8	54.3
Leakage (W)	0.73	0.43	0.28	0.21	0.14	1.34	0.77	0.49	0.34	0.20
Dynamic (W)	0.13	0.09	0.07	0.06	0.26	0.20	0.16	0.14	0.12	0.47
Power (W)	0.86	0.53	0.36	0.27	0.40	1.54	0.93	0.62	0.47	0.67
Reduction (%)	–	38.9	58.7	68.5	53.5	–	39.4	59.5	69.5	56.2

the area of the 512KB cache, thus this study provides conservative results for the proposed hybrid caches.

3.2 Energy and Power Consumption

This section analyzes both leakage and dynamic consumption of the studied 512KB and 1MB caches. Table 4 shows the energy results (in mJ). Leakage energy includes the consumption of the tag array, the data array, and the controller logic. To provide insights in energy savings, we analyzed separately the dynamic energy of the tag array, which is looked up on every cache access, and the energy of both data array and controller logic, which has been classified into six categories according to the access type: SRAM hits, eDRAM hits, swaps, writebacks, misses, and refreshes.

The SRAM hits category denotes the energy consumed by the access to the SRAM array, whereas the eDRAM hits category takes into account the access to the predicted SRAM banks at the first stage plus the actual access to the target eDRAM bank. The swap operation consists of three steps: a read access to the target eDRAM bank, a write access to that bank, and another write access to the target SRAM bank. The consumption of the first step is already considered in the eDRAM hits category, while the expenses of the two latter are taken into account in the swaps category. In addition, this category also includes the consumption of the unidirectional transfers from SRAM to eDRAM banks (write accesses to the target eDRAM bank) that arise on cache misses. The writebacks category considers the energy of accessing just the target bank. The misses category includes the energy required to access the SRAM banks (or eDRAM ones in the case of 16D), and the access to the bank where the incoming block is allocated. The refresh category takes into account the energy consumed by the periodic refresh in hybrid and 16D schemes, and also the consumption due to restoring the eDRAM contents after a destructive read in 16D caches. Notice that, as there is not information loss due to capacitor discharges regardless of the cache scheme, the energy penalty of accessing to the main memory has not been considered since it is the same for all the schemes. For the sake of completeness, results are also shown in terms of power consumption (in W). Leakage and dynamic power values were calculated as the total leakage and dynamic energy, respectively, divided by the execution time.

As observed, both eDRAM and hybrid approaches reduce leakage energy by design thanks to the use of eDRAM banks. Notice that leakage decreases with the number of eDRAM ways. Compared to the 16S cache, the 2S-14D approach reduces leakage by 70% for the 512KB cache. This percentage grows up to 74% for the 1MB cache size. For a given cache scheme, the 1MB caches consume a larger amount of leakage with respect to the 512KB caches since they double the cache capacity.

Regarding dynamic energy, the tag array consumption is almost negligible compared to that of the data array. As expected, for a given cache capacity, the 16S cache is by far the scheme that consumes more energy in the SRAM hits category since all the cache ways are accessed in parallel, while the values in this category decrease with the number of eDRAM ways. As opposite, the eDRAM hits energy increases with the number of eDRAM ways, although the fact of accessing the SRAM ways ahead of the target eDRAM way may prevent from obtaining low energy values even for a low number of eDRAM ways. This is the case of the 8S-8D configuration for the 1MB cache (1.2 mJ).

The expenses of the swap operation do not represent an important fraction of the dynamic energy consumption. The worst case can be found in the 2S-14D configuration for the 512KB cache, where the swap energy consumption (1.1 mJ) represents about 18% of the total dynamic energy (6.0 mJ). Nevertheless, in spite of this fact, this is the configuration that most reduces the dynamic energy among all the studied caches. The consumption due to writebacks slightly affects the total energy, and it is roughly the same across the studied schemes. In contrast, noticeable differences appear in the misses category. The 16S and 16D caches consume a large amount of energy because of the entire data array is involved on each access. For the hybrid caches, this

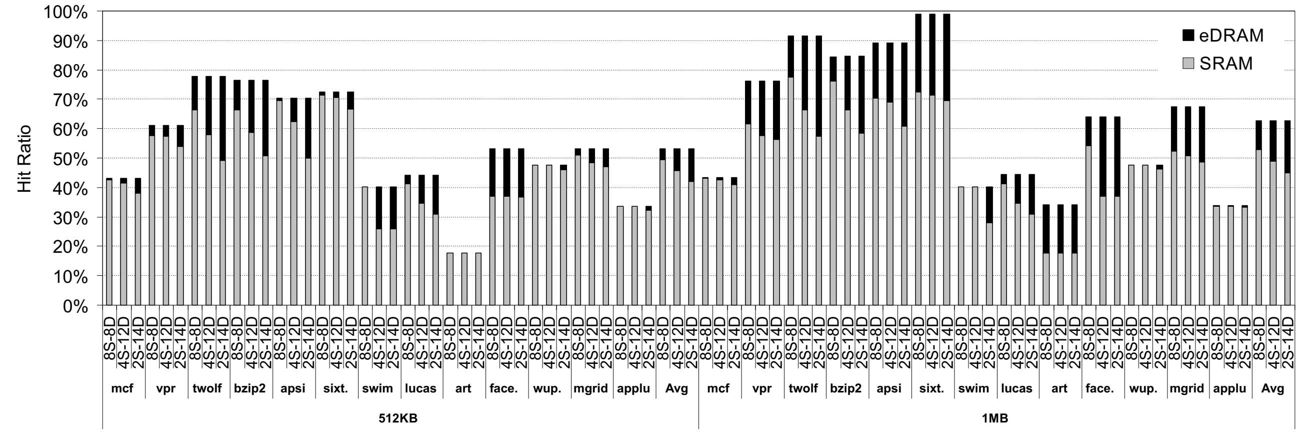


Fig. 5. Hit ratio (%) split into hits in SRAM and eDRAM banks.

consumption decreases with the number of eDRAM ways, similar to the SRAM hits category. The energy due to refresh operations increases with the eDRAM ways since more eDRAM blocks are checked to be refreshed. Recall that this category also includes the energy expenses due to rewriting the contents after a destructive read in the 16D approach.

Overall, for a given cache scheme, the dynamic energy increases with the cache capacity, similar to the leakage energy. Compared to the 16S scheme, the 2S-14D configuration reduces dynamic energy by 36% in the 1MB cache. This percentage is up to 54% for the 512KB cache size. Compared to the conventional scheme, the 16D approach doubles the dynamic energy in the 512KB cache, while in the 1MB cache this energy increases by a 2.48*x* factor.

An interesting observation is that, when comparing configurations on the basis of area (see Section 3.1), both 4S-12D and 2S-14D for the 1MB cache reduce the total dynamic energy in spite of having twice the capacity of the 512KB SRAM approach. On the other hand, the 1MB 16D cache significantly increases the total dynamic energy with respect to the 16S scheme.

Compared to the 16S configuration with the same storage capacity, 16D reduces the total energy consumption by 52% and 54% for the 512KB and 1MB caches, respectively. This percentage grows up to 58–59% and 68–69% with the 4S-12D and 2S-12D schemes. However, for the 8S-8D approach, the obtained energy savings are lower than those of the pure eDRAM scheme. This is mainly due to the high leakage consumption of SRAM banks, which are one half of the cache capacity. On the basis of area, both 4S-12D and 2S-14D hybrid schemes reduce the overall energy by 32% and 49%, respectively. This percentage drops down to 25% for the pure 1MB eDRAM cache.

Remark that, in general, the leakage contribution represents a high percentage of the total energy consumption. This is because leakage energy is always consumed regardless of whether the cache is accessed or not, while there is not L2 dynamic energy consumption if the data are found in L1 (apart from the periodic refresh).

Finally, similar conclusions can be drawn when analyzing the power results. In fact, the overall power reduction percentages are quite similar to those of energy consumption. Minor differences appear because each cache configuration obtains a different execution time.

3.3 Performance Evaluation

To provide insights in performance, we first quantify the hit ratio in the different cache banks since they work at different speeds. Remember that the design does not allow information loss due to capacitor discharges. Thus, the total hit ratio matches the obtained with pure caches. Figure 5 depicts the results.

As expected, for a given cache size, the hit ratio in the eDRAM banks (eDRAM hit ratio) increases with the number of eDRAM ways. Nevertheless, this is not the case in a few applications. For instance, the eDRAM hit ratio in *art* keeps constant for the 1MB cache regardless of the cache bank distribution. This behavior is because in this benchmark (see Figure 1) the MRU way and the following one (i.e., *loc-0* and *loc-1*) capture around 50% of cache hits, while locations from 8 to 15 in the LRU stack capture almost all the remaining hits.

For the 512KB cache, the eDRAM hit ratio is on average only by 4%, 7%, and 11% for 8S-8D, 4S-12D, and 2S-14D approaches, respectively. These percentages grow up to 10%, 14%, and 18% for the 1MB cache. Similarly, for the 512KB cache, the SRAM hit ratio is on average by 49%, 46%, and 42% for 8S-8D, 4S-12D, and 2S-14D configurations, respectively, while for the 1MB cache (with twice as large SRAM data array), these percentages are by 53%, 49%, and 45%. Overall, the 512KB cache achieves on average a higher miss ratio than the 1MB cache.

To enhance the performance in the hybrid cache, it is important that the percentage of eDRAM hits remains as low as possible since eDRAM is slower. Performance losses due to bank contention also rise because of periodic refresh operations. These losses are not constant across the different cache configurations, since the elapsed time between two consecutive periodic refreshes becomes shorter as the number of eDRAM lines increases. In addition, in the pure eDRAM cache (16D), reads require to refresh data since these operations are destructive, which also introduces bank contention. In contrast, bank contention on an eDRAM hit

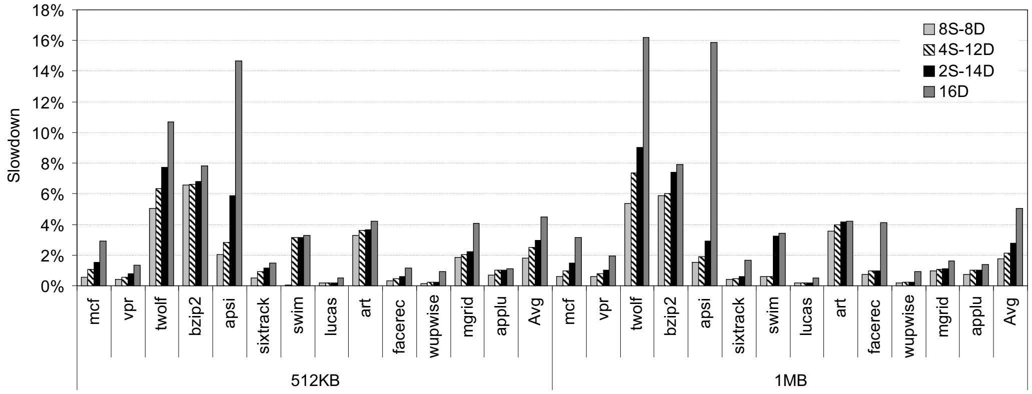


Fig. 6. Slowdown (%) of the studied configurations compared to the pure SRAM cache on the basis of capacity.

in hybrid caches is introduced by the three steps of the swap operation between the involved banks (see Section 3.2).

Figure 6 shows the performance slowdown of the studied caches with respect to a pure SRAM cache with the same capacity and associativity (the lower is the better). As observed, the slowdown increases with the number of eDRAM ways. Enlarging the eDRAM data array results in a higher number of refresh operations, slow accesses to eDRAM data, and swap operations. In general, the performance loss is higher in those applications with higher eDRAM hit ratio. For instance, in *twolf*, the eDRAM hit ratio can be as high as 29% in the 512KB cache with the 2S-14D configuration (see Figure 5), which yields to 7.7% slowdown. Compared to the 1MB cache, the eDRAM hit ratio is up to 34% and its slowdown is by 9%. The pure eDRAM architecture is strongly affected both by the slow access time and bank contention introduced by refresh operations. For example, in *apsi*, the slowdown is by 16% for the 1MB cache with the 16D scheme, resulting in very poor performance.

The slowdown for the 512KB cache is on average by 1.8%, 2.5%, and 3.0% in the 8S-8D, 4S-12D, and 2S-14D approaches, respectively. In comparison, minor differences appear in these percentages for the 1MB hybrid cache. In the 16D configuration, the slowdown grows up to 4.5% and 5.0% for the 512KB and 1MB caches, respectively. Notice too that 8 banks are enough to obtain a reasonably low slowdown for hybrid L2 caches.

For the sake of completeness, we evaluate the impact of the swap operations on the memory latency. For this purpose, latency has been split into latency due to swap and non-swap operations. The latter refers to cache operations others than swaps, which include cache hits, misses, writebacks, and refresh operations. Figure 7 shows the latency distribution of swap and non-swap operations in the 2S-14D cache, since this is the design performing more swaps due to it implements the largest eDRAM data array.

As observed, the percentage of memory latency added by swaps is quite low in most of the applications. Benchmarks that exhibit a high eDRAM hit ratio such as *twolf*, *bzip2*, or *apsi* (see Figure 5), and thus a high performance slowdown as shown above, are those presenting higher values. However, this is not the case of *sixtrack* in the 1MB cache because this benchmark rarely accesses the L2 cache, so minor overall performance differences appear with respect to the conventional cache.

In summary, the latency due to swap operations increases with the cache capacity for a given benchmark since more requests access the eDRAM banks. Nevertheless, its percentage is relatively low and below 1.6% and 5.7% on average for the 512KB and 1MB caches, respectively, which also confirms the low eDRAM bank contention of the hybrid design.

Finally, the analysis on the basis of area is presented. Figure 8 plots the speedup of the selected 1MB eDRAM-based caches with respect to the conventional 512KB SRAM cache. As observed, the hybrid caches, with longer access time when accessing eDRAM data, improve perfor-

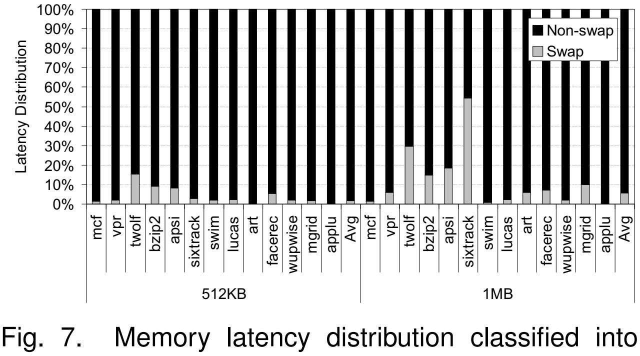


Fig. 7. Memory latency distribution classified into latency due to swap and non-swap operations.

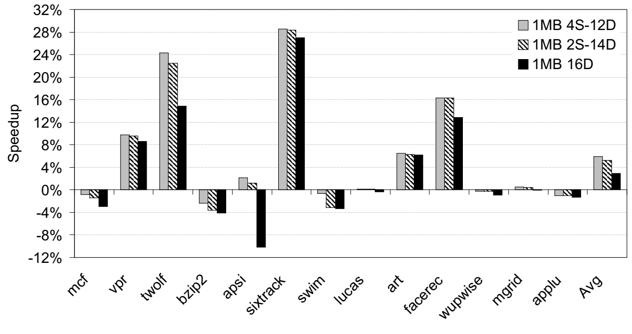


Fig. 8. Speedup (%) of the selected 1MB eDRAM-based caches with respect to the SRAM scheme on the basis of area.

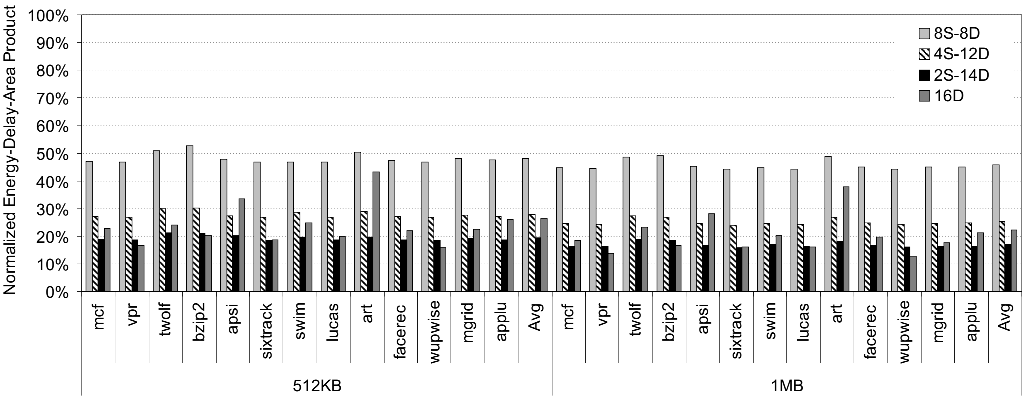


Fig. 9. Normalized EDAP (%) with respect to the pure SRAM approach on the basis of capacity.

mance in 8 of 13 applications. The performance speedup in applications like *vpr* and *twolf* comes from the fact that the hit ratio increases with the cache capacity (see Figure 5). For example, the hit ratio of *vpr* is by 61% and 76% for the 512KB and 1MB cache size, respectively. On the contrary, in other benchmarks such as *mcf* and *bzip2*, the increase in the hit ratio (if any) does not compensate the higher number of refreshes. To sum up, the speedup of 2S-14D and 4S-12D hybrid caches over the conventional 512KB SRAM is on average by 5.2% and 5.9%, respectively, whereas this percentage drops down to 3.0% for the pure eDRAM cache.

3.4 Energy-Delay-Area Product

This section evaluates the trade-off among area, energy, and performance using the recently proposed energy-delay-area product (EDAP) metric [23]. Figure 9 plots the EDAP results normalized over the conventional SRAM cache on the basis of capacity (the lower is the better). On average, for a given cache configuration, the EDAP reduction is quite uniform regardless of the cache organization.

As observed, compared to the pure SRAM cache, all the eDRAM-based configurations significantly reduce the EDAP despite the lower performance obtained for all the applications. This is mainly due to hybrid and pure eDRAM caches address leakage energy and area by design. These results point out the importance of eDRAM-based L2 caches. In particular, 2S-14D is the scheme that most reduces on average this metric compared to the pure eDRAM cache. For the 13 benchmarks analyzed in 512KB and 1MB caches, the 2S-14D design reduces the EDAP in 10 and 9 of them, respectively, over the other schemes. Although the 16D cache provides larger leakage and area savings compared to the hybrid design, its lower performance and higher energy consumption prevent it from being the best design choice. On the contrary, in spite of performing better than the pure eDRAM approach, both 4S-12D and 8S-8D hybrid caches achieve worse EDAP due to increased leakage energy and area. Overall, for the 2S-14D hybrid approach, the EDAP savings are on average by 81–83% depending on the cache organization. These percentages are by 74–78% for the 16D cache.

Figure 10 shows the normalized EDAP results on the basis of area. The reduction of EDAP is not as high as the savings obtained on the basis of capacity because the area differences between the eDRAM-based approaches and the conventional SRAM cache have been relaxed. Anyway, the proposed 2S-14D hybrid scheme, with EDAP savings on average by 61%, remains as the best design choice.

In short, the energy-delay-area product analysis reveals that, on the basis of equal capacity and similar area, the hybrid caches are better designs than pure SRAM caches regardless of the number of eDRAM banks, and also better than the pure eDRAM approach when 12.5% of their banks (2S-14D) are implemented with SRAM technology.

3.5 Energy-Delay Squared Product

This section evaluates the trade-off between performance and energy consumption with the energy-delay squared product (ED^2P) metric, since it reflects whether the hybrid design stands as a cost-effective cache design or not for the near future technologies. Figure 11 shows the normalized ED^2P results on the basis of capacity over the pure SRAM cache. The energy-delay squared product savings are on average quite uniform regardless of the cache organization.

Similar to the EDAP analysis, results indicate that all the eDRAM-based schemes achieve better ED^2P than the SRAM cache. Notice that, in this case, both 4S-12D and 2S-14D hybrid caches obtain on average a higher ED^2P reduction with respect to the 16D scheme. For the 512KB cache, 4S-12D and 2S-14D save ED^2P in 8 and 11 applications, respectively, over the pure eDRAM cache.

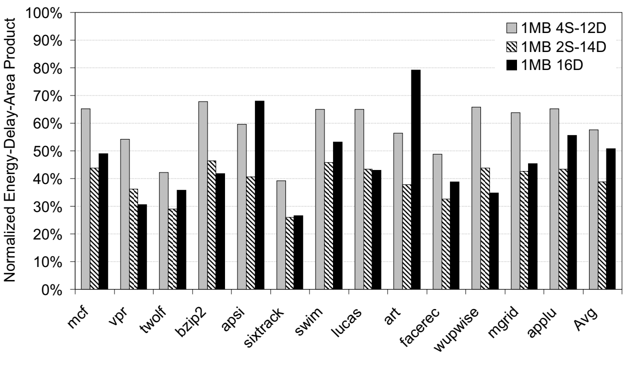


Fig. 10. Normalized EDAP (%) with respect to the pure SRAM cache on the basis of area.

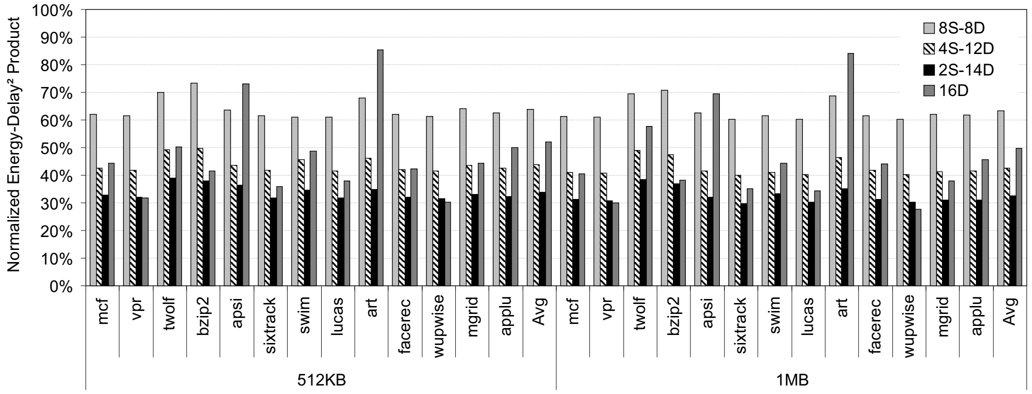


Fig. 11. Normalized ED^2P (%) with respect to the pure SRAM cache on the basis of capacity.

These numbers are 5 and 11, respectively, for the 1MB cache size. Compared to the EDAP analysis, the 4S-12D approach obtains better ED^2P than 16D because the area savings provided by the latter are not being considered in the trade-off. Instead, the lower execution time and consumed energy allow the hybrid cache to outperform the 16D scheme in most benchmarks. For the proposed 4S-12D and 2S-14D hybrid approaches, the ED^2P reduction is on average up to 56–57% and 66–67% depending on the cache organization. These results are by 48–50% in the 16D cache.

Figure 12 plots the normalized ED^2P on the basis of area. As observed, the ED^2P savings are lower than in the previous study on the basis of capacity. The reason is that, although the 1MB eDRAM-based caches perform better than 512KB 16S on average (see Figure 8), energy savings are considerably reduced when the cache capacity is doubled. Moreover, for the 16D cache, the ED^2P is above 100% in *apsi* and *art*. In contrast, regardless of the cache configuration, *twolf* and *sixtrack* applications achieve higher ED^2P reduction than in the analysis on the basis of capacity. This fact can be explained by looking at Figure 8, where these benchmarks reach the highest speedups. On average, the ED^2P reduction is by 40% and 54% for 4S-12D and 2S-14D, respectively.

In summary, the energy-delay squared product demonstrates that, on the basis of equal capacity and similar area, a hybrid cache design with 12.5% (2S-14D) or 25% (4S-12D) of its banks built with SRAM technology is a better cache design option than pure SRAM and eDRAM caches.

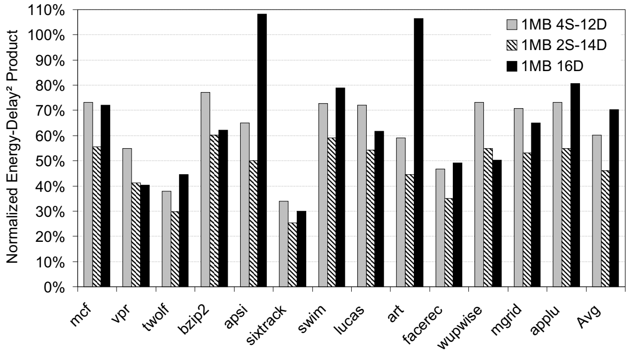


Fig. 12. Normalized ED^2P (%) with respect to the pure SRAM scheme on the basis of area.

3.6 Performance Evaluation in a Chip Multi-Processor System

This section explores the impact of the proposed hybrid L2 caches in a multicore processor. To focus the research, we have assumed a quad-core chip multi-processor. For evaluation purposes, the Multi2Sim simulation framework [24] was extensively modified to model hybrid caches. Two different L2 cache organizations have been considered: i) two separate 1MB 16-way caches, each one shared by a couple of cores and ii) a single 2MB 16-way cache shared by all of the 4 cores. For the latter cache, the access time of the tag array, SRAM banks, and eDRAM banks is 3, 7, and 10 cycles, respectively, as obtained with CACTI. Multiprogrammed mixes designed with benchmarks from the SPEC CPU2006 benchmark suite [25] with the *ref* input set were run skipping the initial 500M instructions and then collecting statistics during 600M cycles for each benchmark. We randomly generated 32 different benchmark mixes, but only a subset of 8 mixes showing the highest raw IPC differences among the studied caches is shown for illustrative purposes. Table 5 summarizes the selected benchmark mixes. The remaining machine parameters are the same as those assumed for the single-core processor evaluation (see Table 2).

The impact of the proposal on the performance of multiprogrammed workloads mainly depends on how hits concentrate on the fast SRAM ways. Note that due to interferences between applications sharing a given cache,

TABLE 5
Benchmark mixes for the multicore evaluation.

Mix	Benchmarks
Mix1	*gcc*, *libquantum*, *povray*, *xalancbmk*
Mix2	*astar*, *bzip2*, *gcc*, *GemsFDTD*
Mix3	*milc*, *sjeng*, *xalancbmk*, *tonto*
Mix4	*gromacs*, *astar*, *perlbench*, *zeusmp*
Mix5	*lbm*, *astar*, *soplex*, *wrf*
Mix6	*mcf*, *namd*, *omnetpp*, *soplex*
Mix7	*dealII*, *leslie3d*, *mcf*, *sjeng*
Mix8	*bzip2*, *lbm*, *wrf*, *xalancbmk*

$$HR_{L2,loc-\{0-1\}} = \sum_{i=A,B} (HR_{i,L2,loc-\{0-1\}} \times \%Accesses_i) - interferences \quad (1)$$

the hit distribution on the SRAM ways will be lower than the average hit distribution of the individual applications running alone.

Assuming the 2S-14D hybrid cache, Equation 1 models the relationship between the hit ratio in the SRAM ways (referred to as LRU locations 0 and 1 in the example) of individual applications (A and B) running alone and the same hit ratio when they share a common L2 cache. The summation term gives the weighted hit ratio considering the percentage of SRAM hits of a given individual benchmark with respect to its total number of accesses to the cache. The *interferences* term quantifies performance drops due to accesses of a given application that force transfers from SRAM to eDRAM ways of data blocks of the other application.

To estimate how the proposal behaves with multiprogrammed workloads, we measured the cache hit distribution across the LRU stack for the designed mixes. Figure 13 presents the results for the 1MB caches.

Notice that the lowest part of the bars represent the left side of Equation 1. As observed, results are encouraging since the $HR_{L2,loc-\{0-1\}}$ values range in between 55% and 83%. To provide insights on these results, we launched simulations for individual benchmarks and measured the values used in the summation. Then, we calculated the interference terms, and results show that they range in between 2% and 16%, with an average by 9%, which demonstrates that considering two SRAM ways for a shared hybrid L2 cache is a good design choice.

Finally, to check how the discussed cache performance impacts on the overall processor performance, the IPC degradation has been evaluated. Figure 14 plots the slowdown of hybrid and pure eDRAM caches compared to the conventional SRAM caches with the same storage capacity. Similarly to the single-core analysis, the slowdown increases with the number of eDRAM ways. For the 1MB caches, the performance loss of the hybrid designs is below 2% in most of the benchmark mixes, which confirms that the devised approach remains valid for multicore. As opposite, the slowdown of the pure 16D scheme is much worse than that of the hybrid caches; for instance, its performance degradation doubles that of the 2S-14D most aggressive hybrid approach in Mix1 and Mix3.

The IPC losses increase in the 2MB cache shared by four cores. The reason is that the overall number of banks (including the SRAM banks of the hybrid design) is reduced to the half from the two 1MB caches to the single 2MB cache, increasing bank contention. In addition, the higher the number of cores the higher the presence of inferences when accessing the cache, which affects the SRAM bank locality. In this context, new approaches addressing bank locality to make more effective the SRAM bank usage could help improving the performance. However, this research is out of the scope of this paper.

4 Related Work

To take advantage of the properties that each technology offers, previous works have focused on hybrid architectures in different memory structures such as on-chip caches, NUCAs, main memories, and multi-threaded register files.

Valero *et al.* [7] proposed a hybrid *n*-bit cell, namely *macrocell*, which consists of one SRAM cell, *n-1* eDRAM cells, and *n-1* bridge transistors that allow internal movements between SRAM and eDRAM cells. The macrocell is used to implement *n*-way set-associative L1 data caches, so that one cache way is built with SRAM cells and the remaining *n-1* ways are implemented with eDRAM cells. Due to the highly-predictable data locality in L1, the single way built with SRAM technology is used to store the MRU data. Unfortunately, the data locality widely differs in L2 caches, so a significant number of accesses would be performed in slow eDRAM cells. In addition, for high-associative caches, like 16-way L2 ones, the macrocell device would become too complex and expensive to implement. Unlike this work, our proposed hybrid caches combine both technologies at bank level; thus overcoming the difficulties that can be encountered using macrocells.

In [26], Mangalagiri *et al.* combined both SRAM and PRAM technologies to propose a hybrid L1 instruction cache. The L1 memory is split into an SRAM-based cache and a PRAM-based cache. To reduce leakage currents, the

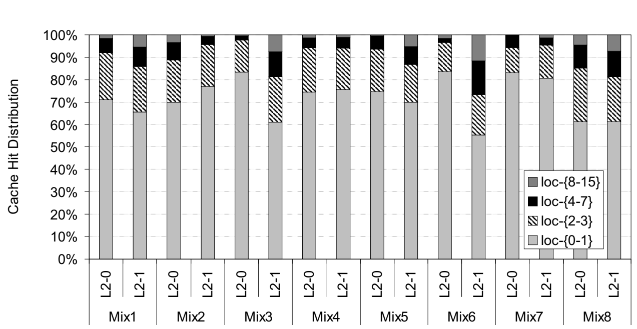


Fig. 13. Hit distribution across the LRU stack for the 1MB caches (L2-0 and L2-1) in the CMP.

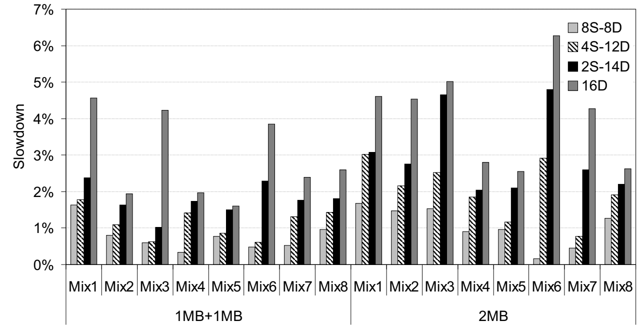


Fig. 14. Slowdown (%) of the analyzed caches with respect to the SRAM design in the CMP.

former cache is implemented as a drowsy cache [27] and it is much smaller than the PRAM memory. On a read access, the PRAM cache is accessed first. If there is a hit in this structure, the operation is completed. On a PRAM miss and an SRAM hit, the requested cache line is woken-up. To improve the write endurance of PRAM, a safe-write policy aware of the frequency of write accesses to PRAM cache lines distributes the writes between the PRAM and the SRAM cache.

In [8], Wu *et al.* proposed two hybrid designs: LHCA and RHCA. The former design implements the L3 cache with eDRAM, MRAM, or PRAM technologies, while both L1 and L2 levels are built with SRAM technology. In the latter design, both L2 and L3 caches are flatten into a pair of regions to form a single level. One region is SRAM-based and the other is eDRAM, MRAM, or PRAM-based, whereas the L1 is SRAM-based. The RHCA design adds more hardware complexity compared to our proposal to manage data movements between regions, since the design requires not only the LRU stack of all lines in a set, but also an additional *sticky bit* for the SRAM lines and a 2-bit saturating counter per eDRAM line. Unlike this work, there is not a design space exploration varying the size of the SRAM region, which is fixed to 256KB throughout the experiments.

Lira *et al.* [9] proposed two different architectures (homogeneous and heterogeneous) for a hybrid eDRAM/SRAM NUCA. In the homogeneous organization, the fast SRAM banks store the frequently accessed blocks and they are placed close to the cores, whereas the eDRAM banks are located in the center of the NUCA. However, this approach is penalized by the shared data, since they are usually located in slow eDRAM banks. On the other hand, the heterogeneous architecture distributes the number of SRAM and eDRAM banks according to their location (close to the cores or in the center of the NUCA). Authors argue that the same number of SRAM and eDRAM banks provide the best trade-off between performance, power, and area in this organization.

Qureshi *et al.* [28] proposed a PRAM-based main memory system that includes a DRAM buffer. The requested pages from hard disk and main memory are stored in the DRAM buffer, while the PRAM memory is only written (if required) when the page is evicted from the buffer. PRAM technology provides higher density with respect to DRAM, while the DRAM buffer allows reducing the number of accesses to the *slow* PRAM memory and the number of write operations to mitigate its write endurance problems.

In [10], Yu *et al.* presented an augmented 1-bit SRAM cell with several eDRAM cells, resulting in a multiple-bit eDRAM/SRAM cell to build register files. The fast SRAM cell is aimed at storing the active context, whereas each pair of eDRAM cells store a dormant context. An additional pair of eDRAM cells is used as a replica of the active context. A dormant context becomes active by transferring the data from the pair of eDRAM cells to the SRAM one.

Other recent works make use of hybrid caches to enhance the block placement. For instance, Hameed *et al.* [29] propose an adaptive line placement policy for a 64MB eDRAM L4 cache, which is coupled with a 6MB SRAM L3 cache to form a single hybrid L3 cache level. The policy discards some blocks from being stored in the eDRAM region to provide thrashing protection. On a miss in L3, the fetched line is always stored in the SRAM region, while the decision to place it in the eDRAM region is made by using the Set Dueling technique, which selects between two different competing insertion policies at runtime by testing them in a few sampled sets.

In [30], Wang *et al.* describe a low-cost adaptive block placement for hybrid MRAM/SRAM L2 caches referred to as Adaptive Placement and Migration (APM). With the aim to reduce long-latency and high-energy consumption of MRAM write operations, APM places a cache block into either MRAM or SRAM lines according to the write type operation (i.e., core-write, prefetch-write, and demand write) and its associated access pattern. An access pattern predictor identifies write-burst and dead blocks in the cache.

Finally, hybrid caches have been also used to address manufacturing imperfections that make SRAM cells unreliable at low voltages. In [31] authors propose a hybrid L1 data cache built with SRAM and eDRAM banks. When the processor works at low voltage to save energy, the effective storage capacity is reduced since the SRAM contents are replicated in some eDRAM banks. This allows the proposed design to cover SRAM faults by retrieving the requested data from such eDRAM banks.

5 Conclusions

Cache memories have been typically built with SRAM technology to achieve high speed accesses. However, this technology presents important drawbacks such as high leakage currents and low density. In contrast, new advances in technology allowed cache memories to be implemented with eDRAM technology, which presents low leakage and high density at the expense of a speed access not as fast as that provided by SRAM. Since both technologies are CMOS compatible, they have been mingled in the same die at the manufacturing process. The eDRAM technology has been used in last-level caches, where energy consumption is an important design concern. Some recent commercial processors, such as the IBM POWER7, incorporate a memory hierarchy with both SRAM-based L1 and L2 caches and an eDRAM-based last-level cache.

In this paper, both SRAM and eDRAM technologies have been mingled in the L2 cache, resulting in a novel hybrid cache design consisting of SRAM and eDRAM banks. The optimal percentage of SRAM banks has been explored to achieve the best trade-off among performance, energy, and area. Architectural mechanisms have been considered to maintain the most likely accessed data in SRAM banks and to avoid unnecessary destructive reads in eDRAM banks.

Experimental results have shown that, compared to a conventional SRAM L2 cache with the same storage capacity, performance degradation never exceeds on average 3%, whereas energy and area savings are on average by 69% and 46%, respectively, for a 1MB 16-way hybrid cache.

Compared to a conventional SRAM cache with similar area, the hybrid cache improves performance on average up to 5.9%, while the total energy reduction is by 32%. In addition, the energy-delay-area product analysis has revealed that, on average, a hybrid cache is a better design than a conventional SRAM cache regardless of the percentage of eDRAM banks, and also better than a conventional eDRAM cache when implementing the eighth part of its banks with SRAM technology (2S-14D). Moreover, the energy-delay-squared product has shown that both 2S-14D and the hybrid design with a quarter of its banks built with SRAM technology (4S-12D) are better design options than conventional SRAM and eDRAM caches.

Finally, the hybrid cache design has been also tested in a chip-multiprocessor system. Experimental results have shown that, similarly to the single-core analysis, the hybrid cache is a good design choice for such systems.

ACKNOWLEDGMENTS

This work was supported by the Spanish *Ministerio de Economía y Competitividad* (MINECO) and FEDER funds under Grant TIN2012-38341-C04-01. Additionally, it was also supported by the Intel Early Career Honor Programme Award and the Intel Doctoral Student Honor Programme Award.

REFERENCES

[1] D. Kanter, *Intel's Haswell CPU Microarchitecture, "Real World Technologies", 2012, available online at http://www.realworldtech.com/haswell-cpu/.*

[2] J. M. Tendler, J. S. Dodson, J. S. Fields, H. Le, and B. Sinharoy, "POWER4 system microarchitecture," *IBM Journal of Research and Development*, vol. 46, no. 1, pp. 5–25, 2002.

[3] B. Sinharoy, R. N. Kalla, J. M. Tendler, R. J. Eickemeyer, and J. B. Joyner, "POWER5 system microarchitecture," *IBM Journal of Research and Development*, vol. 49, no. 4/5, pp. 505–521, 2005.

[4] H. Q. Le, W. J. Starke, J. S. Fields, F. P. O'Connell, D. Q. Nguyen, B. J. Ronchetti, W. M. Sauer, E. M. Schwarz, and M. T. Vaden, "IBM POWER6 microarchitecture," *IBM Journal of Research and Development*, vol. 51, no. 6, pp. 639–662, 2007.

[5] B. Sinharoy, R. Kalla, W. J. Starke, H. Le, R. Cargnoni, J. A. Van-Norstrand, B. J. Ronchetti, J. Stuecheli, J. Leenstra, G. L. Guthrie, D. Q. Nguyen, B. Blaner, C. F. Marino, E. Retter, and P. Williams, "IBM POWER7 multicore server processor," *IBM Journal of Research and Development*, vol. 55, no. 3, 2011.

[6] R. E. Matick and S. E. Schuster, "Logic-Based eDRAM: Origins and Rationale for Use," *IBM Journal of Research and Development*, vol. 49, no. 1, pp. 145–165, 2005.

[7] A. Valero, J. Sahuquillo, S. Petit, V. Lorente, R. Canal, P. López, and J. Duato, "An Hybrid eDRAM/SRAM Macrocell to Implement First-Level Data Caches," in *Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, 2009, pp. 213–221.

[8] X. Wu, J. Li, L. Zhang, E. Speight, R. Rajamony, and Y. Xie, "Hybrid Cache Architecture with Disparate Memory Technologies," in *Proceedings of the 36th Annual International Symposium on Computer Architecture*, 2009, pp. 34–45.

[9] J. Lira, C. Molina, D. Brooks, and A. González, "Implementing a hybrid SRAM / eDRAM NUCA architecture," in *Proceedings of the 18th IEEE International Conference on High Performance Computing*, 2011, pp. 1–10.

[10] W.-k. S. Yu, R. Huang, S. Q. Xu, S.-E. Wang, E. Kan, and G. E. Suh, "SRAM-DRAM Hybrid Memory with Applications to Efficient Register Files in Fine-Grained Multi-Threading," in *Proceedings of the 38th Annual International Symposium on Computer Architecture*, 2011, pp. 247–258.

[11] *Standard Performance Evaluation Corporation, available online at http://www.spec.org/cpu2000.*

[12] A. Valero, J. Sahuquillo, S. Petit, P. López, and J. Duato, "Analyzing the Optimal Percentage of SRAM Banks in Hybrid Caches," in *Proceedings of the 30th IEEE International Conference on Computer Design*, 2012, pp. 297–302.

[13] T. Kirihata, P. Parries, D. R. Hanson, H. Kim, J. Golz, G. Fredeman, R. Rajeevakumar, J. Griesemer, N. Robson, A. Cestero, B. A. Khan, G. Wang, M. Wordeman, and S. S. Iyer, "An 800-MHz Embedded DRAM with a Concurrent Refresh Mode," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 6, pp. 1377–1387, 2005.

[14] B. Calder, D. Grunwald, and J. Emer, "Predictive Sequential Associative Cache," in *Proceedings of the 2nd International Symposium on High-Performance Computer Architecture*, 1996, pp. 244 –253.

[15] R. E. Kessler, R. Jooss, A. Lebeck, and M. D. Hill, "Inexpensive Implementations of Set-Associativity," *ACM SIGARCH Computer Architecture News*, vol. 17, no. 3, pp. 131–139, 1989.

[16] M. D. Powell, A. Agarwal, T. N. Vijaykumar, B. Falsafi, and K. Roy, "Reducing Set-Associative Cache Energy via Way-Prediction and Selective Direct-Mapping," in *Proceedings of the 34th Annual IEEE/ACM International Symposium on Microarchitecture*, 2001, pp. 54–65.

[17] S. Thoziyoor, N. Muralimanohar, J. H. Ahn, and N. P. Jouppi, "CACTI 5.1," *Hewlett-Packard Laboratories, Palo Alto, USA, Technical Report*, 2008.

[18] B. Keeth, R. J. Baker, B. Johnson, and F. Lin, *DRAM Circuit Design. Fundamental and High-Speed Topics*. John Wiley and Sons, Inc., Hoboken, New Jersey, USA, 2008.

[19] A. Valero, J. Sahuquillo, V. Lorente, S. Petit, P. López, and J. Duato, "Impact on Performance and Energy of the Retention Time and Processor Frequency in L1 Macrocell-Based Data Caches," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 20, no. 6, pp. 1108–1117, 2012.

[20] D. Burger and T. Austin, "The simplescalar tool set, version 2.0," *ACM SIGARCH Computer Architecture News*, vol. 25, no. 3, pp. 13–25, 1997.

[21] S. Thoziyoor, J. H. Ahn, M. Monchiero, J. B. Brockman, and N. P. Jouppi, "A Comprehensive Memory Modeling Tool and its Application to the Design and Analysis of Future Memory Hierarchies," in *Proceedings of the 35th Annual International Symposium on Computer Architecture*, 2008, pp. 51–62.

[22] R. Kalla, B. Sinharoy, W. J. Starke, and M. Floyd, "POWER7: IBM's Next-Generation Server Processor," *IEEE Micro*, vol. 30, no. 2, pp. 7–15, 2010.

[23] O. Azizi, A. Mahesri, B. C. Lee, S. J. Patel, and M. Horowitz, "Energy-Performance Tradeoffs in Processor Architecture and Circuit Design: A Marginal Cost Analysis," in *Proceedings of the 37th Annual International Symposium on Computer Architecture*, 2010, pp. 26–36.

[24] R. Ubal, J. Sahuquillo, S. Petit, and P. López, "Multi2Sim: A Simulation Framework to Evaluate Multicore-Multithreaded Processors," in *Proceedings of the 19th International Symposium on Computer Architecture and High Performance Computing*, 2007, pp. 62–68.

[25] *Standard Performance Evaluation Corporation, available online at http://www.spec.org/cpu2006.*

[26] P. Mangalagiri, K. Sarpatwari, A. Yanamandra, V. Narayanan, Y. Xie, M. J. Irwin, and O. A. Karim, "A low-power phase change memory based hybrid cache architecture," in *Proceedings of the 18th ACM Great Lakes Symposium on VLSI*, 2008, pp. 395–398.

[27] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. Mudge, "Drowsy Caches: Simple Techniques for Reducing Leakage Power," in *Proceedings of the 29th Annual International Symposium on Computer Architecture*, 2002, pp. 148–157.

[28] M. K. Qureshi, V. Srinivasan, and J. A. Rivers, "Scalable High Performance Main Memory System Using Phase-Change Memory Technology," in *Proceedings of the 36th Annual International Symposium on Computer Architecture*, 2009, pp. 24–33.

[29] F. Hameed, L. Bauer, and J. Henkel, "Adaptive Cache Management for a Combined SRAM and DRAM Cache Hierarchy for Multi-Cores," in *Proceedings of the Design, Automation, and Test in Europe Conference*, 2013, pp. 77–82.

[30] Z. Wang, D. Jimenez, C. Xu, G. Sun, and Y. Xie, "Adaptive Placement and Migration Policy for an STT-RAM-Based Hybrid Cache," *To Appear in Proceedings of the 20th International Symposium on High-Performance Computer Architecture*, 2014.

[31] V. Lorente, A. Valero, J. Sahuquillo, S. Petit, R. Canal, P. López, and J. Duato, "Combining RAM technologies for hard-error recovery in L1 data caches working at very-low power modes," in *Proceedings*

of the Design, Automation, and Test in Europe Conference, 2013, pp. 83–88.

Alejandro Valero received the BS, MS, and PhD degrees in Computer Engineering from the Universitat Politècnica de València, Spain, in 2009, 2011, and 2013, respectively. He is currently working as a postdoctoral researcher in the Department of Computer Engineering at the same university. In 2012, his research was recognized with the Intel Doctoral Student Honor Programme Award. His PhD research focuses on the design of hybrid caches, high-performance cache replacement algorithms, and refresh mechanisms for on-chip eDRAM caches. His research topics include energy consumption, multicore processors, reliability, and memory hierarchy design. He is member of the ACM.

Julio Sahuquillo received the BS, MS, and PhD degrees in Computer Engineering from the Universitat Politècnica de València, Spain. Since 2002 he is an Associate Professor in the Department of Computer Engineering at the same university. He has taught several courses on computer organization and architecture. He has published more than 100 refereed conference and journal papers. His current research topics include multi- and manycore processors, memory hierarchy design, cache coherence, and power dissipation. An important part of his research has also concentrated on the web performance field, including proxy caching, web prefetching, and web workload characterization. He is a member of the IEEE Computer Society.

Salvador Petit received the PhD degree in Computer Engineering from the Universitat Politècnica de València (UPV), Spain. Currently, he is an associate professor in the Computer Engineering Department at the UPV, where he has taught several courses on computer organization. His research topics include multithreaded and multicore processors, memory hierarchy design, as well as real-time systems. Prof. Petit is a member of the IEEE Computer Society.

Pedro López received the BS degree in Electrical Engineering and the MS and PhD degrees in Computer Engineering from the Universitat Politècnica de València (UPV), Spain, in 1984, 1990, and 1995, respectively. Since 2002 he is a Full Professor in the Computer Engineering Department at the UPV. He has taught several courses on computer organization and architecture. His research interests include high performance interconnection networks for multiprocessor systems, clusters and networks on chip, and, more recently, processor microarchitecture and cache design. Prof. López has published more than 120 refereed conference and journal papers. He has served in different conference program committees and in the editorial board of the *Parallel Computing Journal*.

José Duato received the MS and PhD degrees in Electrical Engineering from the Universitat Politècnica de València, Spain, in 1981 and 1985, respectively. He is currently a professor with the Department of Computer Engineering (DISCA) in the Universitat Politècnica de València. He was an adjunct professor with the Department of Computer and Information Science, The Ohio State University, Columbus. His research interests include interconnection networks and multiprocessor architectures. He has published more than 380 refereed papers. He proposed a powerful theory of deadlock-free adaptive routing for wormhole networks. Versions of this theory have been used in the design of the routing algorithms for the MIT Reliable Router, the Cray T3E supercomputer, the internal router of the Alpha 21364 microprocessor, and the IBM BlueGene/L supercomputer. He is the first author of *Interconnection Networks: An Engineering Approach* (Morgan Kaufmann, 2002). He was a member of the editorial boards of the *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Computers*, and *IEEE Computer Architecture Letters*. He was co-chair, member of the steering committee, vice-chair, or member of the program committee in more than 55 conferences, including the most prestigious conferences in his area: HPCA, ISCA, IPPS/SPDP, IPDPS, ICPP, ICDCS, EuroPar, and HiPC.