Design, Performance, and Energy Consumption of eDRAM/SRAM Macrocells for L1 Data Caches

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

Abstract—SRAM and DRAM cells have been the predominant technologies used to implement memory cells in computer systems, each one having its advantages and shortcomings. SRAM cells are faster and require no refresh since reads are not destructive. In contrast, DRAM cells provide higher density and minimal leakage energy since there are no paths within the cell from Vdd to ground. Recently, DRAM cells have been embedded in logic-based technology (eDRAM), thus overcoming the speed limit of typical DRAM cells.
In this paper, we propose a hybrid n-bit macrocell that implements one SRAM cell and n-1 eDRAM cells. This cell is aimed at being used in an n-way set-associative first-level data cache. Architectural mechanisms (e.g., special writeback policies) have been devised to completely avoid refresh logic.
Performance, energy, and area have been analyzed in detail. Experimental results show that using typical eDRAM capacitors, and compared to a conventional cache, a four-way set-associative hybrid cache reduces both energy consumption and area up to 54% and 29%, respectively, while having negligible impact on performance (less than 2%).

Index Terms—Retention time, static and dynamic energy, static and dynamic memory cells, way prediction.

1 Introduction

Static Random Access Memories (SRAM) and Dynamic RAM (DRAM) have been the predominant technologies used to implement memory cells in computer systems. SRAM cells, typically implemented with six transistors (6T cells) are usually designed for speed, while DRAM cells, implemented with only one capacitor and its associated pass transistor (1T-1C cells) are generally designed for density. Because of this reason, the former technology has been used to implement cache memories and the latter for main memory storage.

Cache memories use to occupy more than half the processor die in current multicore processors. Because of this reason, they dissipate an important amount of the total static energy or leakage, which is proportional to the number of transistors. This design concern is expected to aggravate in future technologies provided that the transistor size will continue shrinking [1]. Therefore, using large SRAM cells impact both in area and leakage.

Dynamic memory cells reduce leakage currents by design, since the power supply is removed after accessing the memory cells. However, these cells present two major drawbacks that make them not appropriate for caches: low speed and information loss. Regarding the first one, technology advances have recently allowed to embed DRAM cells using CMOS technology [10], making them faster. In this way, operation delays in these cells have been largely reduced, so that they have been already used in L2 caches [13] [14]. The latter drawback means that capacitors lose their charge, either when they are read or progressively with time. Thus, eDRAM cells must be recharged or *refreshed* with specific refresh logic so resulting not only in additional energy consumption but also in availability overhead, since the memory cannot be accessed when the refresh is being performed, which is a major design issue in L1 caches. Table 1 summarizes the main design characteristics of the discussed cells.

In [17], we proposed a *macrocell* which combines SRAM and eDRAM technologies aimed at being used at first-level data caches. The term macrocell refers to the proposed memory structure that consists of *n* memory cells, one SRAM cell and *n-1* eDRAM cells, and pass transistors that communicate SRAM to eDRAM cells. In this way, by design, leakage and area are attacked thanks to the use of

Technology	SRAM (6T)	DRAM (1T-1C)	eDRAM
Access time	fast	slow	slow
Density	low	high	high
Leakage	high	low	low
Refresh	no	yes	yes
Destr. reads	no	yes	yes

Table 1
Memory cell characteristics.

• *The authors are with the Department of Computer Engineering (DISCA), Universitat Politècnica de València, Camí de Vera s/n, 46022, Valencia, Spain.*
E-mail: alvabre@gap.upv.es; {spetit, jsahuqui, plopez, jduato}@disca.upv.es

eDRAM cells, which in turn provides six to eight times as much memory as SRAM in the same area [10]. Macrocells are designed to implement set-associative caches. In this context, the total number of cells within the macrocell defines the number of ways in the set-associative cache, that is, to implement an *n-way* set-associative hybrid eDRAM/SRAM cache, *n-bit* macrocells are required.

A hit in the SRAM cell is handled as in a conventional cache. However, a hit in an eDRAM cell involves reading the stored data, which is destructive. To avoid this drawback in the proposed hybrid cache, a swap operation that exchanges the data between the SRAM and the eDRAM cells is triggered on a hit in an eDRAM cell. To do so, the design has some paths interconnecting SRAM and eDRAM cells. On the other hand, eDRAM cells require refresh to avoid losing their contents with time. In the proposed design, we avoid refreshing these cells by merely allowing the cells to lose their data. This should not be a problem if the lost data is not actively used. We address this problem by keeping the Most Recently Used (MRU) block in the SRAM cell. Care must be taken in the case of *dirty* blocks allocated in the eDRAM cells, which must be written back to L2 before the retention time expires. Two writeback policies have been devised (*early* and *delayed*). The proposed delayed writeback policy performs a similar amount of writebacks as a conventional cache, so no energy wasting is incurred.

This paper extends the work in [17] in three main ways. First, a detailed energy evaluation for a 45nm node has been performed covering both dynamic energy and leakage currents. Second, area requirements for the tag and data array have been estimated. Third, contention in multi-banked caches have been also modeled.

Experimental results show that there are not performance losses when using eDRAM cells with a retention time (i.e., the time that an eDRAM cell can not longer hold the stored value) longer than 50K processor cycles. Compared to a conventional cache with the same organization, the proposal reduces the total energy consumption and area up to about 54% and 29%, respectively, with a minimal impact on performance.

The remainder of this paper is organized as follows. Section 2 discusses the design of the proposed cell. Section 3 presents the architectural innovations. Sections 4, 5, and 6 analyze the performance, energy savings, and area savings, respectively, provided by the proposal. Section 7 summarizes the related research, and finally, Section 8 presents some concluding remarks.

2 Memory Cell Proposal

Figure 1 depicts the implementation of an n-bit macrocell consisting of a typical 6T SRAM cell, n-1 eDRAM cells, and n-1 *bridge* transistors communicating the SRAM cell and the corresponding eDRAM cell. The *static* part is limited to only one SRAM cell mainly due to leakage and area reasons. It has the same structure as a typical 6T SRAM cell. Thus, read and write operations in this part are managed like in a typical static cell through the bitline (*BLs*) and its complementary (*/BLs*).

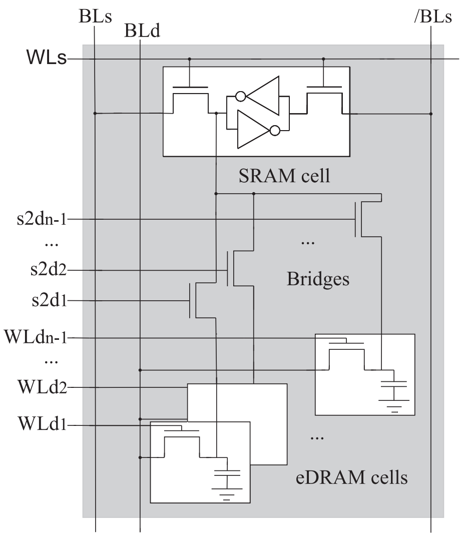


Figure 1. n-bit macrocell block diagram.

The *dynamic* part is composed of n-1 eDRAM cells, each one working like a typical 1T-1C DRAM cell. Each capacitor has an NMOS pass transistor (controlled by WLd_i) that keeps the capacitor charge insulated from the bitline (*BLd*). Likewise the static part, read and write operations are performed, like in a conventional eDRAM cell, through the pass transistor.

Information movements among SRAM to eDRAM cells imply two main steps in a conventional design: i) read the SRAM cell content and write it to an intermediate buffer, and ii) read the buffer and write its content to the eDRAM cell. The main novelty of the proposed design is that transistors act as unidirectional bridges controlled by the $s2d_i$ signals to move data from the SRAM to the target eDRAM cell. The information state in the SRAM cell can move directly to the capacitor, that is, no intermediate buffer is required. These transfers will be referred to as internal since no bitline is involved.

Notice that implementing a cache using macrocells instead of a split L1 cache (a standard SRAM and an independent eDRAM cache) has important advantages. First, some resources (e.g., decoder and wordline) can be shared. Second, when using two independent caches, the internal transfers (i.e., moving a block from static to dynamic part) involve much more circuitry and wire delays, prohibitively increasing latency and energy dissipation.

In order to check the correctness of the electronic behavior, the proposed cell has been modeled in NGSPICE, a Berkeley's Spice3f5 based circuit simulator. NGSPICE allows us to accurately simulate MOSFET behavior since it uses BSIM4 MOSFET model. All simulations used the Predictive Technology Models (PTM) [21]. Transistor features have been taken from the 2007 ITRS [1] for a 45nm technology node.

Two main design issues were addressed to ensure the correct functionality of the proposed cell: i) to check that the capacitor is properly charged, and ii) the absence of flips when moving data from the SRAM cell to an eDRAM cell. Regarding the former issue, the main problem in a typical 1T-1C cell is the voltage degradation when writing a logic '1' to the capacitor. This is due to NMOS pass transistors incur a voltage drop equal to *Vth* when they transfer a logic '1'. Thus, in order to charge the capacitor to the maximum *Vdd* voltage, wordlines are usually boosted to a voltage *Vpp* = Vdd + Vth. For eDRAM cells, Vth and Vdd are usually set to 0.4V and 1.1V, respectively, for a 45nm technology node. Regarding the macrocell, the wordlines controlled by WLd_i and $s2d_i$ must be also boosted.

Concerning the second design issue, read operations in conventional SRAM cells must be preceded by precharging high both bitlines. Precharge operations are necessary to optimize the cell speed, area, and stability relationship [18]. This is mainly due to the different features of NMOS and PMOS transistors. In this way, flips are avoided inside the cell since NMOS transistors are *stronger* (i.e., they can drive more current) than PMOS transistors. Analogously, to prevent flips in the proposed design, the capacitor of the eDRAM cell must be precharged to Vdd.

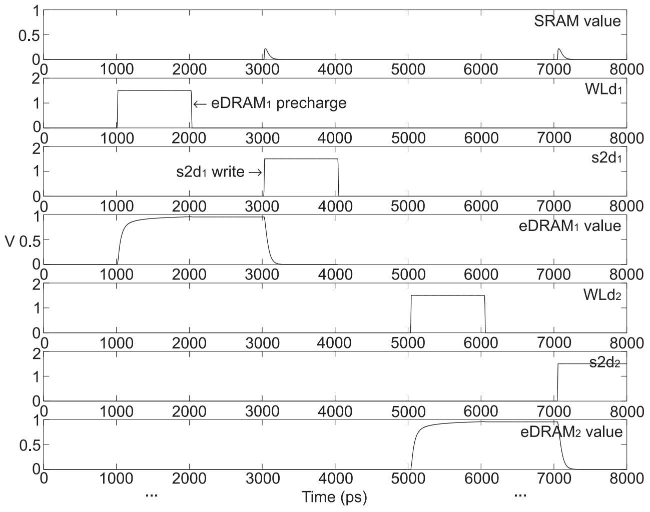
Figure 2 illustrates how the internal transfer operation between the SRAM cell and three eDRAM cells works in a 4-bit macrocell, highlighting both the precharge process and how flips are avoided [1]. Two internal transfers labeled as $eDRAM_1$ and $eDRAM_2$ are shown for illustrative purposes, both writing a '0' and a '1'. Finally, it can be also appreciated that the access time to write a logic '0' is about 1ns.

Besides correctness, as the capacitor loses charge with time, the retention time of the eDRAM cell has been calculated with CACTI 5.3 [16] [15] and assuming a 20fF capacitor used in typical implementations [18], which has a retention time around 0.12ms.

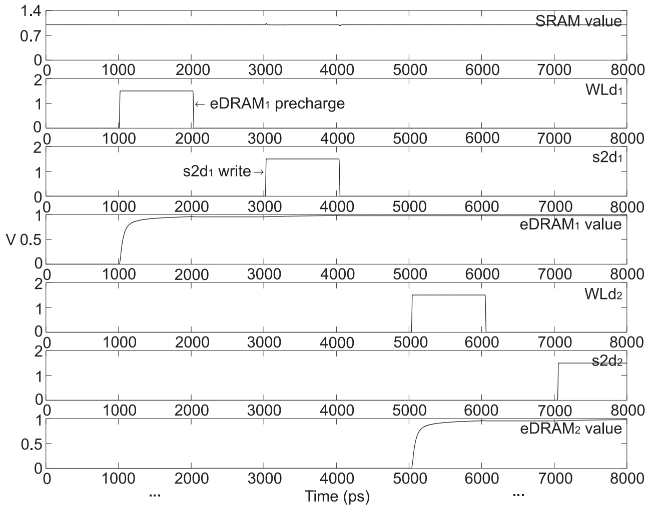
The proposed macrocell reduces both leakage and area. Since leakage is mainly produced by the SRAM cells, the macrocell reduces leakage proportionally to the number of implemented eDRAM cells. For instance, assume a 4-bit macrocell that has 1 SRAM cell and 3 eDRAM cells; ideally, as only the SRAM cell has leakage, the 4-way hybrid cache would achieve 75% (i.e., 3/4) less leakage than a conventional cache with the same capacity implemented with SRAM cells (see Section 5 for a deeper and accurate analysis). On the other hand, area is also reduced since eDRAM cells are implemented with less transistors than SRAM cells (see Section 6). Finally, notice that the higher the cache associativity degree, the higher the leakage savings and area reduction achieved with macrocells.

As read operations in an eDRAM cell are destructive and capacitors progresively lose their charge with time, refresh actions are usually required. In this context, new architectural innovations have been devised to avoid extra logic and energy dissipation due to refresh operations.

1. Unidirectional bridge transistor features have been assumed the same as the pass transistors of the eDRAM cells, whose channel length and width are 45nm and 90nm, respectively.



a) Transfering a logic '0'



b) Transfering a logic '1'

Figure 2. Static to dynamic write operation details.

3 Architectural Design Issues

As mentioned above, the proposed n-bit macrocell has been designed with the aim of implementing the data array of an n-way set-associative cache, having one way implemented with SRAM cells (*SRAM* way) and n-1 ways with eDRAM cells (*eDRAM* ways). From now on, we assume that the SRAM cells implement the way-0 of the cache. The tag array of such caches is assumed to be implemented with typical 6T memory cells, thus the reading of any tag is not a destructive operation.

The architectural scheme we devise to this cell consists of three main steps: i) accessing the eDRAM cells only in case of a hit in the corresponding tag, ii) increasing the percentage of hits in the SRAM blocks in order to minimize the number of data movements between SRAM and eDRAM cells, and iii) minimizing the number of writebacks to L2. Below, we detail these steps.

3.1 Accessing the eDRAM Cells

Typically, to achieve high performance, modern microprocessors overlap the access to the data and tag arrays. In this way, the requested data can be already available in case of a successful tag comparison (i.e., a cache hit). Figure 3(a) shows the accessed parts (dark boxes) in a four-way cache. The narrow box in the left represents the tag array and the right box represents the data array. This working behavior is not adequate for data arrays implemented with macrocells, since regardless the result of the access (i.e., cache hit or cache miss), reading the eDRAM cells is destructive and the design does not include refresh logic. Due to this reason, architectural solutions have been devised to avoid that cache blocks lose their contents.

In order to guarantee correct execution, the problem is not that a given cache block loses its state with time but that the state became unrecoverable. In other words, the information loss is not a problem for correctness when it can be recovered from another location, for instance, from the L2 cache.

The solution we devise to deal with this problem has two main design goals: first, to reduce the number of state losses due to reads, and second, to provide a recoverable location without negatively impacting on performance.

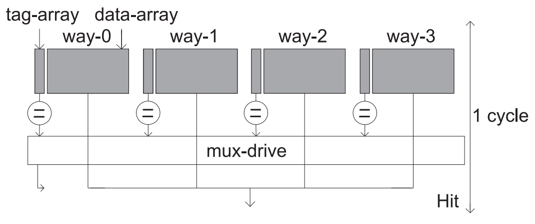
To achieve the first design goal, the tags of all ways in the set are accessed in parallel with the data array of the block located in way-0 (see the upper side of Figure 3(b)). On a hit in way-0 (*SRAM* hit), the processor is satisfied and no eDRAM cell is subsequently read. This access could obtain the hit speed of a direct-mapped cache since the multiplexer control signals could be set early, similarly as done in way prediction techniques [3] [6]. On a miss in way-0 but a hit in other tag associated to an eDRAM block, the corresponding data array is accessed subsequently. To illustrate this case, the lower part of Figure 3(b) represents a hit in way-2 (*eDRAM* hit). In this case, the data is read and sent to the processor. However, since this operation is destructive, the capacitor state is lost. A straightforward solution to address this drawback is to implement a write-through policy so that a copy of the data can be always read from L2 if the capacitor is discharged. However, this will lead to a huge energy wasting in L2 caches as well as a poor performance.

3.2 Increasing the Percentage of Hits in the SRAM Cell

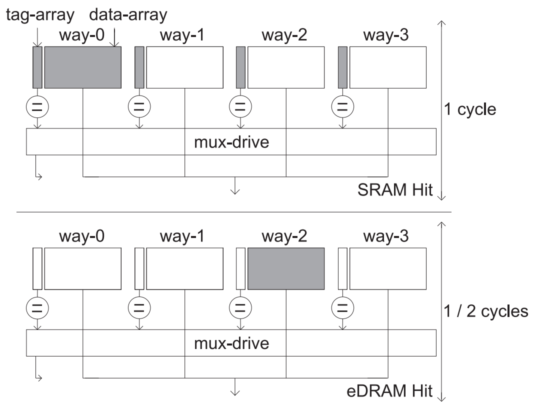
Some previous works [11] concluded that, because of data locality, in a two-way set-associative cache, most of the accesses hit the MRU block while few of them hit the Least Recently Used (LRU) block, as shown in Figure 4. This conclusion can be generalized to caches with larger associativities as shown in Section 4. Based on this assert, a simple way predictor could be implemented by predicting the MRU way.

The previous reason implies a modification of the cache controller in order to keep the MRU block always in the SRAM way of the hybrid cache. To this end, we enhance the cache controller in order to internally manage a swap between eDRAM and SRAM cells in case of a hit in the eDRAM cells. The swap operation consists of three main steps. First, the data in the eDRAM cells is transfered to an intermediate buffer. Then, the data stored in the SRAM cells is internally transferred to the eDRAM cells previously accessed. Finally, the data from the intermediate buffer is written to the SRAM cells. If each step takes one cycle, the controller will be busy for three processor cycles. However, the processor can get the requested data earlier from the intermediate buffer. Remark that in case of a cache miss, the requested data comes from a lower level of the memory hierarchy instead of from the intermediate buffer.

Notice that writes in the eDRAM cells are always done as internal movements from the SRAM cells, which are not



a) Conventional cache



b) Hybrid cache

Figure 3. Access and timing for conventional and hybrid caches.

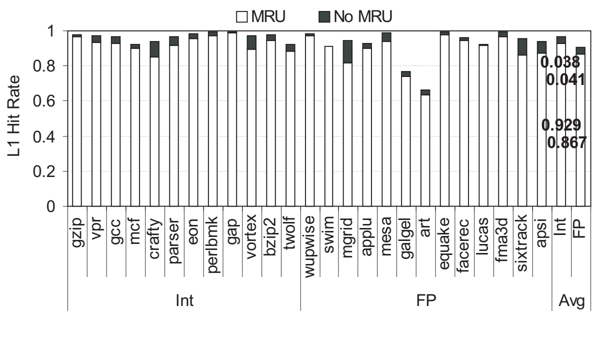


Figure 4. L1 hit rate in a conventional 16KB-2way 64B/line cache.

always triggered by store instructions. They may be caused by a hit (either a load or a store) in the eDRAM cells.

To sum up, capacitors lose their charge with time, but those capacitors storing useful information (i.e., those cells where there has been a hit) will not need to be recharged since before losing their state this information will be saved in the SRAM cells.

Finally, since we assume that the tag array is implemented with typical 6T memory cells, tags do not need to be swapped. Therefore, control information is needed to track the correspondence between tags and the associated data block. In particular, for a 2-way cache, only the LRU bit is enough to link tags and data blocks, and for a 4-way cache, two additional bits can be used.

3.3 Devised Writeback Policies

Although the architectural decisions discussed in the previous section avoid many capacitor state losses, they do not guarantee that cache blocks always preserve their state. This happens when a *dirty* block stored in eDRAM cells is rarely accessed. Thus, incorrect execution could occur when accessing this block. To deal with this shortcoming, two new writeback policies have been devised. These policies will be referred to as *early* writeback and *delayed* writeback.

The early writeback policy avoids the problem by preventing dirty blocks to be stored in eDRAM cells. As a block can only be written into eDRAM cells due to an internal swap operation, this policy checks the state of the block located in way-0 (SRAM way) when a swap rises and, if dirty, the block is written back to L2. The main drawback of this policy is that it might increase the number of writes to L2 so turning on excessive energy wasting. This policy assumes that the *valid* bit of each eDRAM way is implemented as a 1T-1C cell, including a capacitor with less capacitance than the macrocell capacitors to work as a sentry bit. In this way, if the capacitor of the sentry bit is not discharged, the contents of the associated data bits will be correct. On the contrary, if the capacitor of the sentry bit is discharged, the macrocell may still contain valid data, but the design conservatively assumes that the associated data has expired, what is equivalent to a non-valid content.

The delayed writeback policy is aimed at minimizing the number of writebacks to L2. To this end, this policy allows that dirty blocks move to the eDRAM cells. Once moved, capacitors maintain their state as dirty during the retention time. In this case, if the block is accessed again within the retention time, the block is moved to the SRAM cells so preserving its state, otherwise, the state is lost. To avoid the latter case, the solution we propose is to access each block periodically before the retention time expires, and if it is dirty, the block is written back to L2. This idea can be implemented with a single global binary counter [7] [9] for all eDRAM blocks. The counter is decreased every cycle, and each time the counter reaches zero, an eDRAM block accessed following a given criterion (e.g., round-robin) is checked to be written back. The counter must be initialized to the retention time divided by the number of eDRAM blocks. In this policy, the valid bit can be implemented in two different ways, i) by using a sentry bit similarly as done in the previous policy, and ii) by clearing the valid bit each time the block is checked whether it is written back to L2 or not. Since both of them provide similar performance, the first one was assumed across the experiments due to homogeneity reasons.

4 Performance Evaluation

This section presents the simulation environment and benchmarks used to evaluate performance in microprocessors implementing hybrid caches with macrocells. These cells have been modeled on top of an extensively modified version of the Hotleakage simulation framework [20]. Table 2 summarizes the architectural parameters used through the experiments. The delayed writeback policy has been used by default.

Experimental results were obtained configuring the Hotleakage framework for the Alpha ISA and running the SPEC2000 benchmark suite [2]. Both integer (Int) and floating-point (FP) benchmarks have been evaluated using the *ref* input sets, where we skip for all benchmarks 1B instructions before collecting statistics, and then simulate 500M instructions.

Multi-banked caches have been modeled to represent a more realistic approach. Each bank can support only one access at a time, but accesses to different banks can be performed concurrently. If a memory request refers to a bank that is already being accessed, that request must wait until the previous access (e.g., hit and writeback) finishes

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, and choice predictor has 4K 2-bit counters
Branch predictor penalty	10 cycles
Fetch, issue, commit	4 instructions/cycle
ROB size (entries)	256
# ALUs	4 Int, 4 FP
Memory hierarchy	
Memory ports	4
L1 data cache	Variable, 64 byte-line
L1 data cache hit latency	Split into hits in the dynamic and static parts
L1 data cache banks	8
L2 data cache	512KB, 8 ways, 64 byte-line
L2 data cache hit latency	10 cycles
L2 data cache banks	Infinite
Memory access latency	100 cycles

Table 2
Machine parameters.

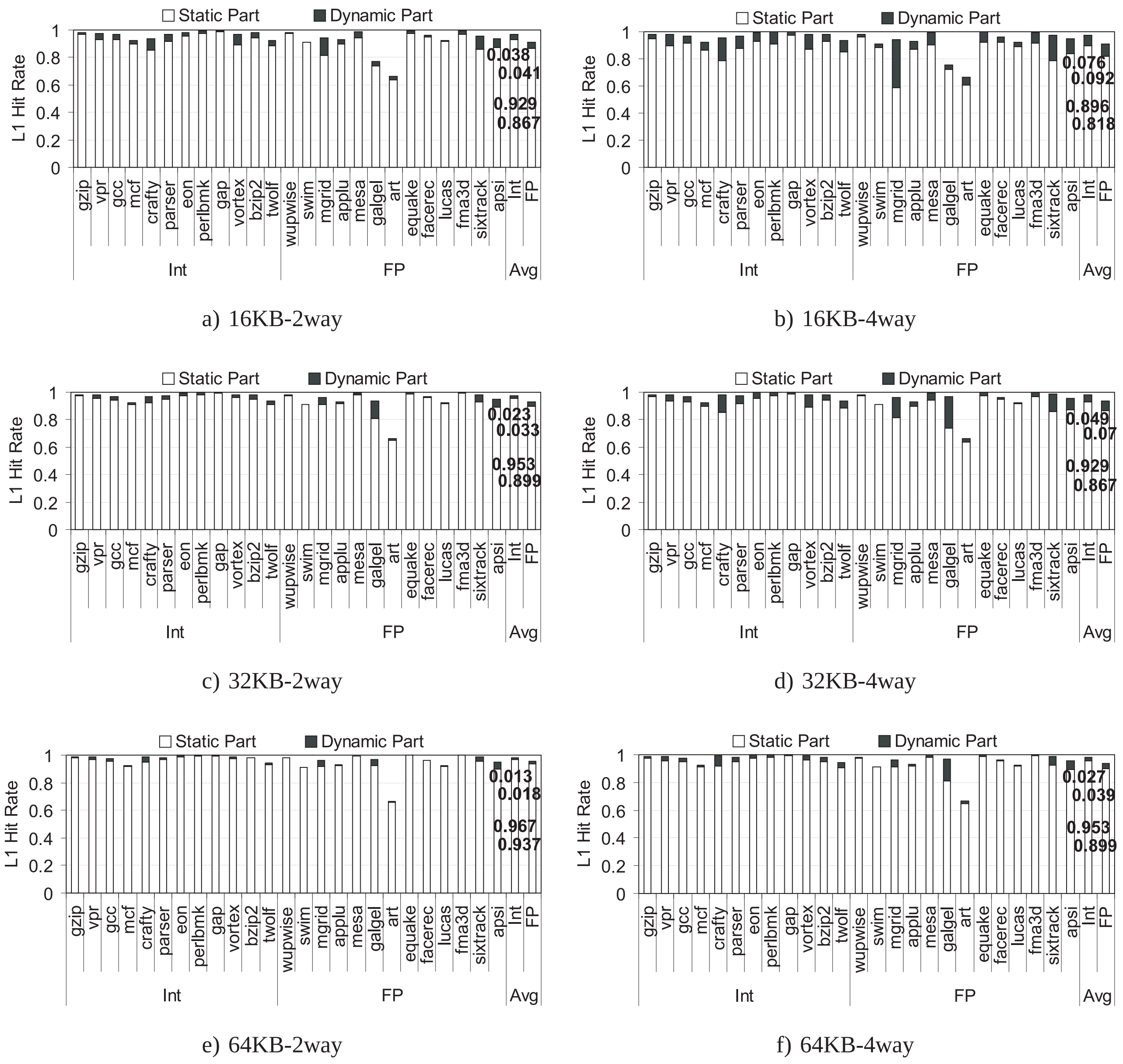


Figure 5. L1 hit rate for different hybrid cache organizations.

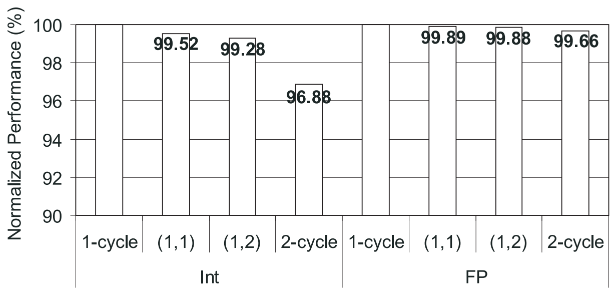
and the cache controller releases the corresponding bank. The waiting time also includes the time taken for a swap operation (if needed) in hybrid caches.

To reduce the likelihood of waiting due to bank contention, the cache sets have been interleaved among banks. Experimental results showed that, regardless the cache scheme and organization, eight banks provide less than 0.7% performance degradation with respect to an unlimited number of banks. Therefore, the number of banks has been set to eight across all the experiments. This number of banks is reasonable, since it is common to find designs having more banks in the literature [8]. Finally, since one bank only can perform one swap operation at a given time, the number of banks also matches the number of intermediate buffers. This amount of buffers does not adversely affect the length of the bitlines, since their capacitances, as obtained with CACTI, range from 14 to 29fF depending on the cache organization. These values fall in the same range as the ones obtained with conventional eDRAM caches implemented with the same cell capacitance and number of banks.

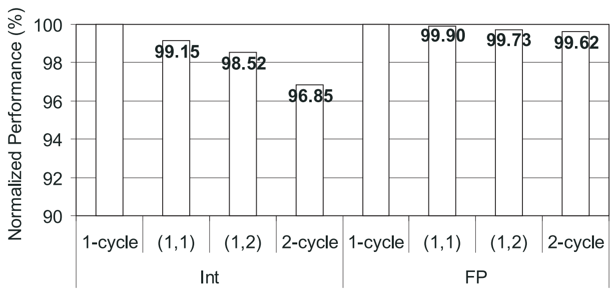
4.1 SRAM and eDRAM Hit Rate

A read in the eDRAM cells requires a refresh operation since the read operations are destructive, even if the access results in a cache miss. Hence, to avoid *unnecessary* refresh operations, eDRAM cells are only accessed after checking the tags. This section characterizes such accesses.

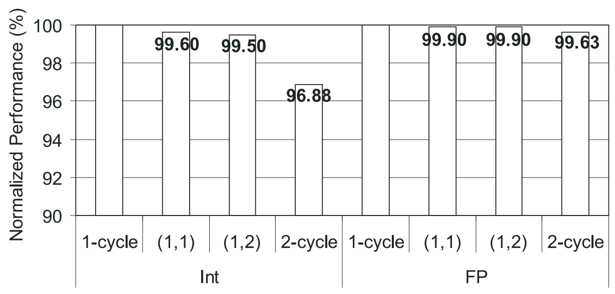
We use the terms SRAM hit rate and eDRAM hit rate to refer to the percentage of cache accesses that hit the static and the dynamic part, respectively. The sum of both gives



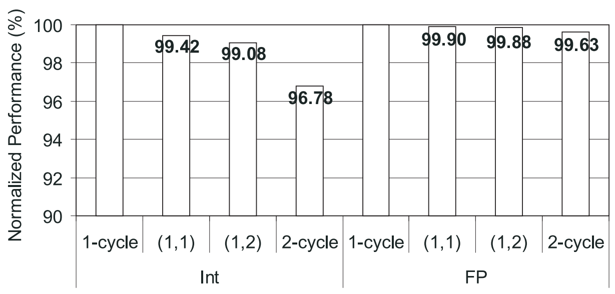
a) 16KB-2way



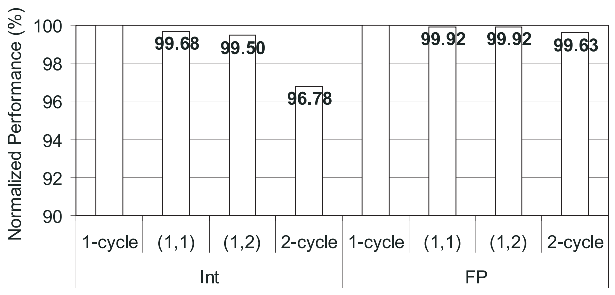
b) 16KB-4way



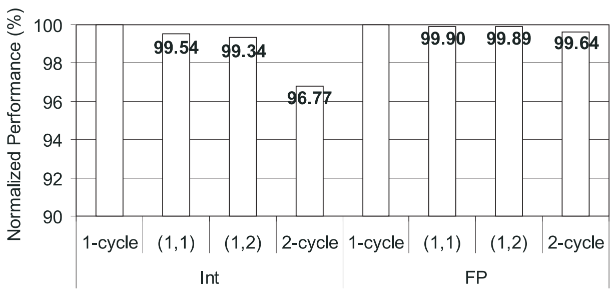
c) 32KB-2way



d) 32KB-4way



e) 64KB-2way



f) 64KB-4way

Figure 6. Normalized performance (IPC) with respect to the conventional cache for different access times (in processor cycles).

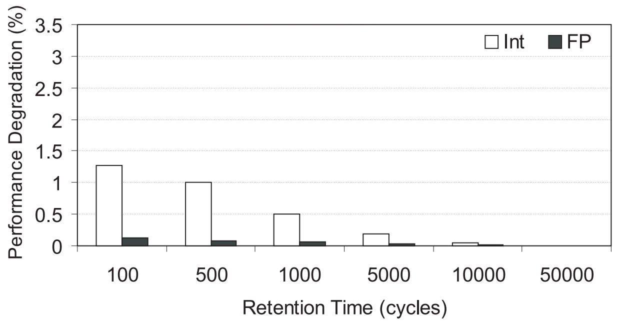
the L1 hit rate. Figure 5 shows the results for different cache sizes (16KB, 32KB, and 64KB) and number of ways (2 and 4).

As an n-bit macrocell has only one SRAM cell, the storage capacity of the SRAM cells is the cache size divided by the number of bits of the macrocell, that is, divided by the number of cache ways. For instance, the static storage capacity of a 16KB-2way and a 32KB-4way cache is 8KB.

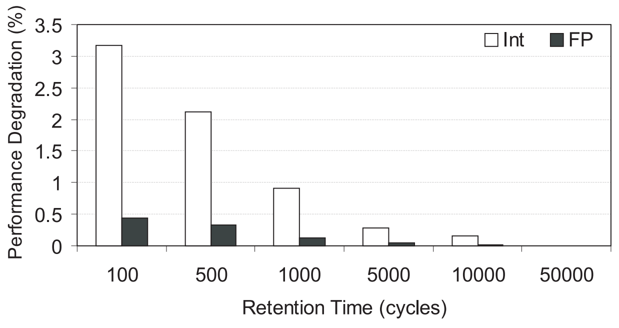
An important observation is that, due to the architectural mechanism that enforces the storage of the MRU block in the SRAM way, the number of hits in this way only depends on its storage capacity. For instance, the hit rate of the static part of a 16KB-2way cache matches the one of a 32KB-4way cache, since both caches have a static data array of the same size (i.e., 8KB). Hence, both arrays will have the same amount of misses, being the difference that the incoming block may be fetched from different memory structures (e.g., eDRAM cells or other level of the memory hierarchy).

Notice that for a given cache size, the higher the associativity degree, the smaller the size of the SRAM way. Hence, although increasing the number of ways leads to a higher overall L1 hit rate, the SRAM hit rate decreases with the number of ways and the eDRAM hit rate increases. In contrast, for a given associativity degree, enlarging the cache size results in a lower eDRAM hit rate and a higher overall L1 hit rate.

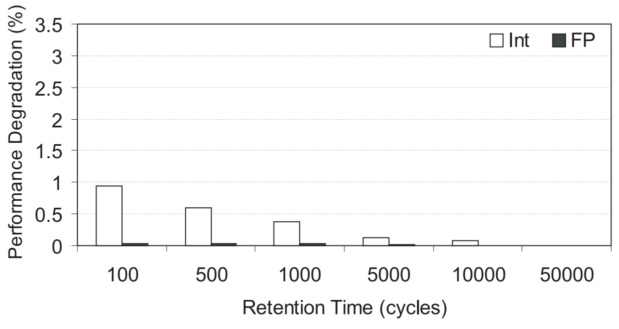
Results show the effectiveness of the devised architectural mechanisms since the eDRAM hit rate is, on average, less than 4.1% for 2-way caches regardless the benchmark type (integer or floating-point) and cache size. This value grows up to 9.2% for FP benchmarks in the 16KB-4way cache configuration, but in this case, the overall L1 hit rate is also larger than a 16KB-2way cache. Notice that a low



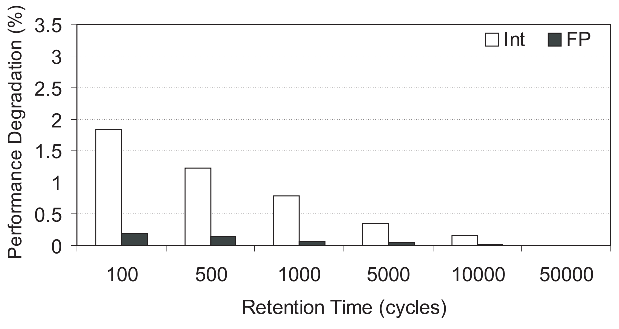
a) 16KB-2way



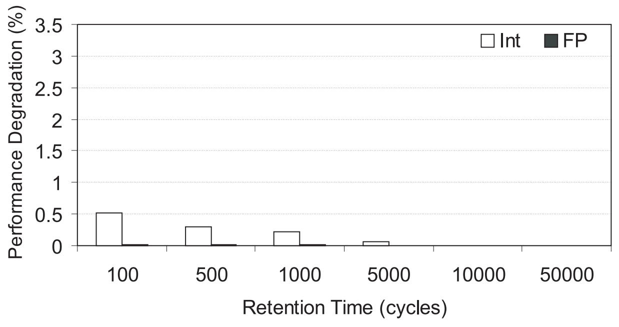
b) 16KB-4way



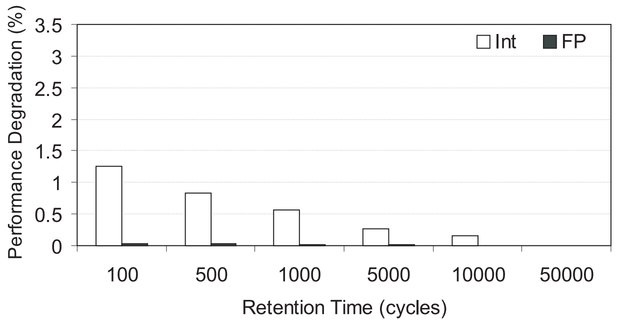
c) 32KB-2way



d) 32KB-4way



e) 64KB-2way



f) 64KB-4way

Figure 7. IPC losses with respect to the perfect hybrid cache for different retention times (in processor cycles).

eDRAM hit rate is interesting due to two main reasons: first, the overall access time of the eDRAM cells requires an additional processor cycle for tag comparison, and second, a hit in an eDRAM way incurs a swap between the static and dynamic parts.

4.2 Impact of eDRAM Access Time on Performance

Results provided by CACTI 5.3 indicate that an eDRAM cell can be accessed as fast as an SRAM, as also stated in [10], although writes can be slightly slower.

This section evaluates the impact on performance of the hybrid cache varying the access time of the eDRAM ways (1 and 2 cycles), while keeping constant (1 cycle) the access time of the SRAM way. Notice that a *fast* conventional cache with just 1-cycle access time imposes an upper-bound in performance (i.e., IPC) to the proposal, since there is no performance loss.

Figure 6 shows the normalized performance with respect to the conventional cache with the same organization. For comparison purposes, the values of a 2-cycle access time conventional cache are also plotted. The X axis shows the access times of the different cache organizations. Conventional caches are labeled as 1-cycle and 2-cycle. Access times of the hybrid caches are labeled as pairs (1,*x*), where the first and second elements refer to the access time of the SRAM and eDRAM ways in processor cycles, respectively. Remember that a hit in an eDRAM way would require an additional processor cycle to check the tags.

The performance degradation of the hybrid cache mainly comes from the access to eDRAM ways and swap operations. Anyway, the performance degradation is always less than 2%. In contrast, the performance of the 2-cycle conventional cache drops down to about 3%. Regarding floating-point benchmarks, minor performance differences appear both in the slower 2-cycle conventional cache and in the proposed hybrid cache.

4.3 Impact of Retention Time on Performance

In the previous section, the performance degradation of the hybrid cache has been evaluated with respect to a conventional SRAM cache. In that study, a perfect capacitor with no charge loss has been assumed. Nevertheless, real capacitors lose their state after a given retention time. Therefore, accessing the eDRAM cells after this time requires the access to a lower level of the memory hierarchy to get the requested data, thus, adversely impacting the performance. This section explores the impact of retention time on performance with respect to a hybrid cache with perfect capacitors.

Figure 7 shows the results ranging the retention time from 100 to 50K processor cycles. Retention times longer than 50K cycles do not improve the cache hit rate in any of the analyzed organizations; thus, no performance degradation can be observed.

For a given associativity degree, the larger the cache size (e.g., 64KB) the lower the performance losses (due to the larger static part of the cache). In contrast, for a given cache size, the higher the associativity degree the larger the performance losses (due to the larger number of eDRAM hits -i.e. larger dynamic part). Finally, large dynamic parts require long retention times to reach the performance of the baseline.

To sum up, the proposal just requires from capacitors that retain their charge for a few thousands of processor cycles in order to achieve its maximum performance. This means that the eDRAM capacitance can be several orders of magnitude lower than the analyzed in Section 2, which will be enough to avoid performance losses due to retention time.

4.4 Writeback Policies Evaluation

A writeback to L2 in a conventional writeback cache arises when a dirty block is evicted from L1. In addition to writebacks due to replacements, new types of writebacks can be identified in the devised policies. Regarding the early writeback policy, two types of writebacks can be distinguished. First, the SRAM block must move to an eDRAM way in order to make room to the missing block. In such a case, if the SRAM block is dirty it must be written back to L2. In addition, a writeback can be also performed due to a swap risen as a consequence of an eDRAM hit. Thus, there are writebacks both due to replacements and due to internal swaps. Regarding the delayed policy, writebacks due to replacements only can occur in the eDRAM ways. In addition, writebacks can also rise driven by the internal counter. This kind of writebacks will be referred to as *sporadic* writebacks.

To evaluate the effectiveness of the devised policies, we obtained the corresponding writeback rate for each writeback class, calculated as the number of blocks written back divided by the number of accesses to L1.

Figure 8 shows the average writeback rate of both policies in the hybrid cache configurations for a 50K-cycle retention time. For comparison purposes, the figure also depicts the average writeback rate of the conventional cache. As observed, for a given cache organization and benchmark type, the replacement writeback rate is quite uniform across the analyzed policies. Regarding the early policy, its swap writeback rate leads to an important amount of blocks written back to L2. Comparing this rate to the sporadic rate of the delayed policy, we conclude that the early policy is too conservative since many blocks are written back unnecessarily, so incurring in an energy wasting.

In short, the proposed delayed writeback policy performs, with no refresh, a similar amount of writebacks as a conventional cache with the same organization, so no energy wasting is incurred.

5 Analysis of Energy Savings

This section discusses the methodology used to obtain both dynamic and leakage energy consumed by each cache scheme. Then, energy results are presented and analyzed for different cache organizations using a 45nm technology node.

5.1 Methodology

An access to a conventional cache consumes energy due to the fact that all components in the cache set (i.e., all tag and data ways) are accessed in parallel. In contrast, only the SRAM way of the data array is accessed in the hybrid cache, although in parallel with all tags of the set. Therefore, on an SRAM hit, the hybrid cache consumes less dynamic energy than the conventional cache. Nevertheless, additional energy is consumed on an SRAM miss (i.e., an

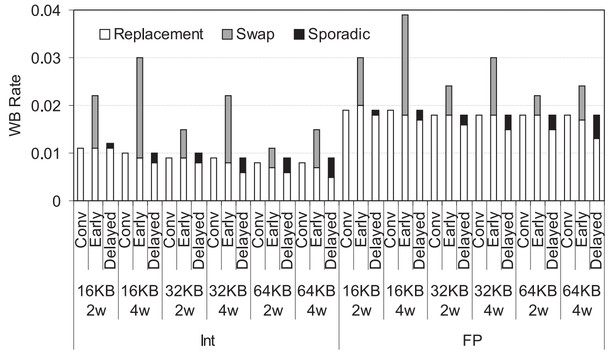


Figure 8. Writeback rate for the devised policies and the conventional cache.

eDRAM hit or cache miss), since the cache controller must drive a swap operation between the corresponding eDRAM and SRAM cells.

To estimate the energy consumption, each of the three steps involved in a swap operation (see Section 3.2) has been modeled as follows: i) a destructive read in an eDRAM way, ii) a store in an eDRAM way, and iii) a store in the SRAM way. While the dynamic energy consumed in the second step involves only the row decoder, in the first and third steps it also involves all the data array components (e.g., bitlines, senseamps, subarray output drivers, and so on). In addition, the energy consumption due to wordline voltage boosting and precharging capacitors in the second step (see Section 2) has been also taken into account in the results.

On a cache miss, the requested block is fetched from L2 (or main memory) and the victim block of L1, if dirty, is written back to L2. These read and write requests to L2 have been also taken into account to compute the dynamic energy.

To quantify energy savings, the proposed hybrid cache has been modeled with CACTI 5.3 using the ITRS high-performance device type for the SRAM cells and the logic-process based DRAM for the eDRAM cells. CACTI provides the dynamic energy consumed per access and categorized by cache component. Using these values, dynamic energy was calculated per access according to different events: SRAM hit, eDRAM hit, misses, and writebacks. Then, we measured the times that each event rises during each benchmark execution (i.e., 500M instructions). Finally, the overall dynamic energy was computed by multiplying the energy per access for a given event by the times that the event rises. Regarding leakage consumption, it is accumulated each cycle taking into account that there is no leakage energy associated to the eDRAM cells.

5.2 Energy Results

The proposed hybrid cache is compared not only to the conventional (Conv) scheme but also to another conventional scheme making use of way prediction as done by the proposal. This model will be referred to as WP scheme. In this way, the benefits due to the prediction technique can be identified. For the hybrid cache, a 50K-cycle retention time and the delayed writeback policy were used.

Figure 9 illustrates the dynamic and leakage energy results for the analyzed schemes. The dynamic energy consumed by the L1 data array is split according to the type of event (i.e., loads, stores, writebacks, and misses), whereas the tag array energy comprises all the accesses to this structure. Finally, the represented leakage refers to the leakage consumption of the whole L1 cache. The sum of all these values gives the total energy consumed by the cache.

Notice that leakage energy consumption is obtained by accumulating each cycle the leakage consumption of the whole cache. In contrast, dynamic energy is only consumed when there is an access to the cache. Thus, in those execution periods where the cache is not accessed, the leakage energy consumption dominates the overall energy consumption of the L1 cache. Such periods use to be common since many memory reference instructions access to the cache in a burst mode. Moreover, when the processor pipeline is stalled due to a long latency event (e.g., an L2 miss), the L1 cache is not accessed for hundreds of cycles.

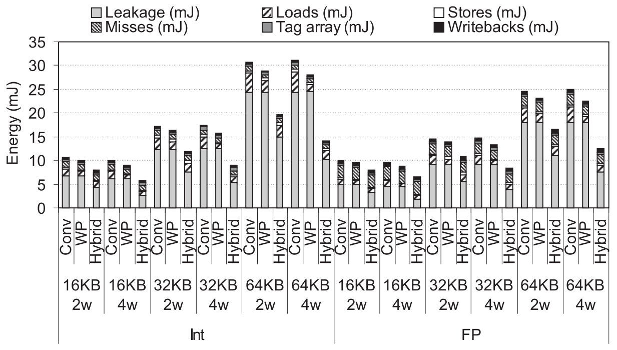


Figure 9. Energy consumption (in mJ) for the analyzed schemes.

Due to the aforementioned reasons, leakage dominates the overall energy consumption. As a consequence, the hybrid cache, which incurs in less leakage consumption because of the use of eDRAM cells, shows the best energy results. In particular, regarding integer benchmarks, the proposed cache reduces the overall energy consumption by about 36% and 54% for a 64KB-2way and a 64KB-4way cache organizations with respect to the Conv scheme, respectively. This trend is similar in the remaining analyzed cache organizations as well as for floating-point benchmarks. Notice that the leakage energy consumption is similar in the Conv and WP schemes, since both schemes use only SRAM cells for data storage.

Leakage consumption increases with the cache size for both the Conv and WP schemes. However, in order to this assert become true for the hybrid cache, the associativity degree must be fixed too. Although theoretically the hybrid cache should reduce leakage by 50% and 75% for 2 and 4 ways, respectively, experimental results do not reach these values. The main reason is that this theoretical reduction is restricted to cells in the data array (where the eDRAM technology is being applied) while other parts of the cache structure, like the tag array and the remaining logic are not being considered.

Regarding dynamic energy, the WP cache exhibits the lowest consumption, closely followed by the hybrid cache. This is because both schemes save a significant amount of dynamic energy by applying way prediction. However, the overall consumption is higher in the hybrid cache mainly due to swap operations, wordline voltage boosting, and precharge of capacitors. The consumption due to loads and stores also increases with the cache size (although more slowly than leakage consumption) since the dynamic energy consumed per access grows with the cache size. On the other hand, the consumption due to writebacks is quite uniform across the analyzed schemes since the delayed writeback policy performs a similar amount of writebacks as the Conv and WP schemes. Finally, larger caches show lower miss and writeback rates, thus devoting less energy to such events.

All the analyzed schemes implement the same tag array (i.e., using typical 6T SRAM cells). Therefore, the energy consumed by this structure slightly differs across the schemes for a given cache organization. Nevertheless, its contribution to the overall energy consumption increases with the cache size and associativity.

6 Analysis of Area Savings

The area occupied by the macrocell has been estimated with CACTI by adding the area of the conventional SRAM and eDRAM cells. In addition, the area overhead due to the wordline voltage boosting as well as the bridge transistors have been taken into account, assuming a conservative design where each bridge transistor occupies the same area as a 1T-1C cell.

The area of an SRAM and an eDRAM cell is 0.296μm^2 and 0.062μm^2 for a 45nm technology node, respectively. Using these values, since an n-bit macrocell consists of one SRAM cell and n-1 eDRAM cells, the area of a 2-bit and a 4-bit macrocell is 0.566μm^2 and 0.836μm^2, respectively. These area values were used as input to CACTI to obtain the area of the entire hybrid cache.

Table 3(a) shows the area of the tag and data arrays of both conventional and hybrid schemes varying the size (from 16KB to 4MB) of a 2-way associative cache. Table 3(b) shows the corresponding results for a 4-way set-associative cache. The last column of each table shows the total reduction (in percentage) achieved by the hybrid cache. This percentage is quite uniform regardless the cache size, and it is around 6% and 29% for 2 and 4 ways, respectively.

Cache size	Tag array (mm^2)	Data array (mm^2)		Red. (%)
		Conv	Hybrid	
16KB	0.007	0.060	0.056	7
32KB	0.013	0.113	0.105	7
64KB	0.018	0.214	0.201	6
128KB	0.033	0.407	0.381	6
256KB	0.063	0.793	0.743	6
512KB	0.114	1.541	1.445	6
1024KB	0.233	3.064	2.874	6
2048KB	0.442	6.048	5.670	6
4096KB	0.833	11.955	11.216	6

a) 2 ways

Cache size	Tag array (mm^2)	Data array (mm^2)		Red. (%)
		Conv	Hybrid	
16KB	0.007	0.060	0.040	31
32KB	0.013	0.113	0.075	30
64KB	0.018	0.214	0.147	29
128KB	0.033	0.407	0.287	27
256KB	0.063	0.793	0.549	29
512KB	0.115	1.542	1.070	28
1024KB	0.233	3.065	2.100	29
2048KB	0.437	6.050	4.177	29
4096KB	0.829	11.957	8.268	29

b) 4 ways

Table 3
Tag array and data array area (in mm^2) for both conventional and hybrid schemes.

7 Related Work

Leakage reduction has been widely analyzed in recent years in conventional SRAM caches. The proposed techniques can be classified in two main groups depending on whether the block state is preserved or not. In the first group, the voltage supply to selected cache lines is reduced and the lines are put in a low-power drowsy mode, so increasing the access time to such lines [4] [11]. In the second group, the voltage supply to the cache blocks with poorer locality is removed, thus, losing the block information [7] [12]. Subsequent accesses to such blocks will result on a cache miss, thus, the next level of the memory hierarchy must be accessed.

Implementation of 1T-1C eDRAM cells has been made in two main ways: DRAM-based and logic-based implementation. Initially, the consensus was that the use of *DRAM-based* eDRAM was the correct strategy for system evolution, because of its cheaper processing although its slower access time. In contrast, *logic-based* eDRAM produces fast devices but it requires adding the eDRAM deep trench to the logic process which was seemingly more difficult and expensive.

Matick and Schuster [10], from the IBM technology group, thought that the use of *logic-based* eDRAM to embed 1T-1C cells would be the most feasible way to reduce the increasing gap in speed between logic and DRAM memory technologies. Posterior work by the same technology group led to the inclusion of *logic-based* eDRAM in the IBM ASICs product. As a consequence, several recent microprocessors implement L2 caches in this technology [13] [14].

Other research works have concentrated on the design of new DRAM cells to reduce leakage dissipation. Most of these cells could be fabricated using logic-based technology. In this context, Liang et al. [9] proposed the 3T1D DRAM cell for L1 data caches. This cell consists of three transistors and a diode with the aim to reduce leakage currents. This cell can be quickly accessed; however, the diode charge get lost over time, which requires refresh actions. Hu et al. [5] proposed a DRAM cell based on 6T SRAM cell but only with 4 transistors, resulting in a non-static cell (the quasi-static 4T cell). The 4T cells offer an easy method for DRAM implementation in a logic process production, especially in embedded systems. 4T cells mainly differ from the 6T SRAM cells in that they do not include the two transistors connected to Vdd that restore the charge loss due to leakage currents. Using the same transistor type, the 4T cell requires less area than the 6T SRAM cell while achieving almost the same performance.

As opposite, the data access is a bit slower and destructive. Like in the 1T-1C cell, this problem can be solved by re-writing the read data immediately after the read operation.

Mixing different memory technologies in the cache hierarchy has been evaluated by Wu et al. [19]. They proposed two types of hybrid cache architecture to construct on-chip cache hierarchies by combining SRAM, eDRAM, MRAM, and PRAM technologies: i) the inter-layer hybrid cache, which means that the L3 cache is implemented with eDRAM, MRAM, or PRAM and the other levels are implemented with SRAM, and ii) the region-based hybrid cache, which means that the L2 and L3 caches are flatten into two regions forming a single level (i.e., one small fast (SRAM) region and a large but slow eDRAM, MRAM, or PRAM region) while the L1 cache is implemented with SRAM technology. Performance and power consumption are evaluated in each type of hybrid cache architecture. The latter approach is improved by intra-cache data movements.

8 Conclusions

Memory cells design for cache memories is a major concern in current microprocessors mainly due to the involved leakage, dynamic energy, area, and access time.

In this paper, we have introduced the macrocell memory with an n-bit capacity storage, and we have shown that combining SRAM and eDRAM technologies to implement a macrocell is feasible. An n-bit macrocell can be straightforwardly used to implement first-level n-way set-associative caches with minimal impact on performance, since it addresses by design speed, area, and leakage consumption.

Experimental results showed that, compared to a 4-way set-associative conventional cache with the same organization, the total energy consumption and area savings are up to 54% and 29%, respectively, for a 45nm technology node. These percentages can be further reduced for a higher associativity degree. In addition, a relatively low capacitor charge is enough to avoid performance losses due to the retention time constraint. Finally, remark that the devised delayed writeback policy allows to avoid the refresh logic, required in typical eDRAM memories, with minimal impact on performance and energy consumption.

Acknowledgments

This work was supported by the Spanish *Ministerio de Ciencia e Innovación* (MICINN), and jointly financed with *Plan E* funds under Grant TIN2009-14475-C04-01 and by Consolider-Ingenio 2010 under Grant CSD2006-00046.

References

[1] *Semiconductor Industries Association, " International Technology Roadmap for Semiconductors", 2007, available online at http://www.itrs.net/*.

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

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

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

[5] Z. Hu, P. Juang, P. Diodato, S. Kaxiras, K. Skadron, M. Martonosi, and D. W. Clark. Managing Leakage for Transient Data: Decay and Quasi-Static 4T Memory Cells. *Proceedings of the 2002 International Symposium on Low Power Electronics and Design*, pages 52–55, 2002.

[6] K. Inoue, T. Ishihara, and K. Murakami. Way-Predicting Set-Associative Cache for High Performance and Low Energy Consumption. *Proceedings of the 1999 International Symposium on Low Power Electronics and Design*, pages 273–275, 1999.

[7] S. Kaxiras, Z. Hu, and M. Martonosi. Cache Decay: Exploiting Generational Behavior to Reduce Cache Leakage Power. *Proceedings of the 28th Annual International Symposium on Computer Architecture*, pages 240–251, 2001.

[8] T. Kirihata, P. Parries, D. R. Hanson, H. Kim, J. Golz, F. G., 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*, 40(6):1377–1387, 2005.

[9] X. Liang, R. Canal, G.-Y. Wei, and D. Brooks. Process Variation Tolerant 3T1D-Based Cache Architectures. *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 15–26, 2007.

[10] R. E. Matick and S. E. Schuster. Logic-Based eDRAM: Origins and Rationale for Use. *IBM Journal of Research and Development*, 49(1):145–165, 2005.

[11] S. Petit, J. Sahuquillo, J. M. Such, and D. Kaeli. Exploiting Temporal Locality in Drowsy Cache Policies. *Proceedings of the 2nd conference on Computing frontiers*, pages 371–377, 2005.

[12] M. Powell, S.-H. Yang, B. Falsafi, K. Roy, and T. N. Vijaykumar. Gated-Vdd: A Circuit Technique to Reduce Leakage in Deep-Submicron Cache Memories. *Proceedings of the 2000 International Symposium on Low Power Electronics and Design*, pages 90–95, 2000.

[13] B. Sinharoy, R. N. Kalla, J. M. Tendler, R. J. Eickemeyer, and J. B. Joyner. POWER5 System Microarchitecture. *IBM Journal of Research and Development*, 49(4/5):505–521, 2005.

[14] J. M. Tendler, J. S. Dodson, J. S. Fields, H. Le, and B. Sinharoy. POWER4 System Microarchitecture. *IBM Journal of Research and Development*, 46(1):5–25, 2002.

[15] 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. *Proceedings of the 35th Annual International Symposium on Computer Architecture*, pages 51–62, 2008.

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

[17] 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. *Proceedings of the 42th Annual IEEE/ACM International Symposium on Microarchitecture*, 2009.

[18] N. H. E. Weste, D. Harris, and A. Banerjee. *CMOS VLSI Design: A Circuits and Systems Perspective*. Pearson/Addison-Wesley, 2005.

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

[20] Y. Zhang, D. Parikh, K. Sankaranarayanan, K. Skadron, and M. Stan. Hotleakage: A Temperature-Aware Model of Subthreshold and Gate Leakage for Architects. *University of Virginia Department of Computer Science, Technical Report*, 2003.

[21] W. Zhao and Y. Cao. Predictive Technology Model for Nano-CMOS Design Exploration. *Journal on Emerging Technologies in Computing Systems*, 3(1):1–17, 2007.

Alejandro Valero was born in Valencia, Spain, on November 14, 1985. He received the B.S. and M.S. degrees in computer engineering from the Universitat Politècnica de València, Valencia, Spain, in 2009 and 2011, respectively. He has earned a 4-year Spanish Research Grant, and he is currently working towards a Ph.D. degree at the Department of Computer Engineering in the same university. His Ph.D. research focuses on the design of hybrid caches. His research topics include energy consumption and memory hierarchy design.

Salvador Petit (M'07) received his Ph.D. degree in computer engineering from the Universitat Politècnica de València, Valencia, Spain. Currently, he is an Associate Professor in the Department of Computer Engineering at the UPV where he has several courses on computer organization. His research topics include multithreaded and multicore processors, memory hierarchy design, as well as real-time systems. He is a member of the IEEE Computer Society.

Julio Sahuquillo (M'04) received his B.S., M.S., and Ph.D. degrees in computer science from the Universitat Politècnica de València, Valencia, Spain. Since 2002 he is an Associate Professor at the Department of Computer Engineering. He has taught several courses on computer organization and architecture. He has published over 90 refereed conference and journal papers. His research topics have included multiprocessor systems, cache design, instruction-level parallelism, 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.

Pedro López (M'02) is a Full Professor in computer architecture and technology at the Department of Computer Engineering, in the Universitat Politècnica de València, Valencia, Spain. He received the BEng degree in electrical engineering and the M.S. and Ph.D. degrees in computer engineering from the same university in 1984, 1990 and 1995, respectively. He has taught several courses on computer organization and architecture. His research interests include high performance interconnection networks for multiprocessor systems and clusters and networks on chip. Prof. López has published more than 100 refereed conference and journal papers. He is a member of the editorial board of Parallel Computing journal. He is a member of the IEEE Computer Society.

José Duato received the M.S. and Ph.D. degrees in electrical engineering from the Universitat Politècnica de València, Valencia, Spain, in 1981 and 1985, respectively. He was an Adjunct Professor with the Department of Computer and Information Science, The Ohio State University, Columbus. He is currently a Professor with the Department of Computer Engineering (DISCA) in the Universitat Politècnica de València. 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).

Dr. Duato 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 cochair, 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.