Transactions Briefs

A CAM With Mixed Serial-Parallel Comparison for Use in Low Energy Caches

Aristides Efthymiou and Jim D. Garside

***Abstract*—A novel, low-energy content addressable memory (CAM) structure is presented which achieves an approximately four-fold improvement in energy per access, compared to a standard parallel CAM, when used as tag storage for caches. It exploits the address patterns commonly found in application programs, where testing the four least significant bits of the tag is sufficient to determine over 90% of the tag mismatches; the proposed CAM checks those bits first and evaluates the remainder of the tag only if they match. Although, the energy savings come at the cost of a 25% increase in search time, the proposed CAM organization also supports a parallel operating mode without a speed loss but with reduced energy savings.**

***Index Terms*—Asynchronous design, content-addressable memory, low-power design.**

I. Introduction

The cache accounts for a considerable fraction of the energy consumption in an embedded system. For example, in the StrongARM processor, it is responsible for over 40% of the total power [1]. At the same time, it is usually performance critical, so implementations targeting low energy consumption cannot neglect the access time.

Two options are available for the cache tag storage: random access memory (RAM) or content addressable memory (CAM). There is an ongoing debate about which consumes less energy: a RAM-tagged associative cache with an intelligent order of accessing its tags and ways (e.g., way prediction [2]) or a CAM-tagged, high-associativity cache. In earlier work [3], we presented a comparison of these approaches for subbanked caches and argued that the difference in the two designs would arise from the energy consumption of the tags. With a more than one tag search per memory access (on average) for the RAM-based caches and their (usually) lower hit rate, a CAM which consumes for a search, no more that twice the energy consumption of an equivalent RAM can lead to a more energy efficient cache design.

This paper presents an adaptive, serial-parallel CAM organization with an energy consumption per search of only about 50% more than a RAM read. Compared to the earlier work [3], a variation with a low-leakage cell and more information on application behavior on cache tag matching is presented.

Section II discusses the two main sources of energy consumption in CAMs and how application behavior can be exploited to save energy. Section III proposes the new serial-parallel CAM organization that offers three to four times reduction in energy consumption per search. Finally, a comparison with the state of the art in CAM design is presented in Section IV and conclusions are given.

Manuscript received March 1, 2003; revised July 1, 2003. This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC), U.K., under Grant GR/M71466/01.

The authors are with the Department of Computer Science, University of Manchester, Manchester, M13 9PL, U.K.

Digital Object Identifier 10.1109/TVLSI.2004.824298

II. Energy Consumption in CAMs

A. Match-Line Energy Consumption

About half the energy consumed in a CAM is due to the repeated precharging and discharging of all but one of the match lines in each access. This is due to the "parallel" (or NOR type) implementation of the match operation [4]. "Serial" (or NAND type) CAM designs [5], [6] search one bit at a time (for each row) so that they do not discharge a single large capacitance when there is no match. Unfortunately, they are generally slower than parallel CAMs, as their search speed depends on the number of cells in a row.

Assuming a 50% probability of a match at any bit position (for each line), when a serial search method is used, after n bits, the probability of a line not matching is $1/2 + (1/2)^2 + \ldots + (1/2)^n$ or $1 - (1/2)^n$. Thus, after only four bits, this probability would be 93.75%, which means that the majority of the match operations could stop early. This can be exploited by a mixed, serial-parallel, matching method, where after searching a small number of bits serially, the remainder can be evaluated in parallel, for those lines that could still match.

1) Application Behavior in Cache Tag Matching: To check if these expectations are true in real programs, an analysis of the memory traces of various SPECInt95 benchmarks was undertaken. The benchmarks were compiled for an ARM processor, with all the speed optimizations enabled.

For each Benchmark, two cases were analyzed: *i*) a unified 16-KB cache and *ii*) split instruction and data caches, each 16 KB in size. The cache line size is 32 b and a write back, no write-allocate policy with round-robin line replacement is employed.

The memory traces from the simulations were used to measure *i*) how many tag comparisons would stop at each bit (starting from the least significant) and *ii*) the percentage of accesses which do not match at each tag bit. The results, presented in Figs. 1 and 2 show that the experimental data agree very closely with the expected behavior mentioned above: Over 90% of the tag line mismatches are determined within the four least significant bits (LSBs) of the tags. This figure is consistent for all the benchmarks and in both unified and split cache configurations.

Fig. 2 shows that at the least significant positions all bits have approximately 50% probability of matching. In data caches, this trend continues up to the most significant bit, while in instruction caches, the most significant half of each tag line seems to always match. As expected, unified caches have a behavior between the other two.

It should be noted that caches are assumed to use virtual addressing for the above evaluation. If the addresses were translated to physical addresses, the results may have been different, but for low-power operation, virtually addressed caches are preferred because they avoid address translation for most memory accesses.

B. Search-Line Energy Consumption

Traditional CAM cells combine the search lines with the bitlines [4]. This causes an increase in the capacitance of each search line as an extra transistor drain per cell is present (though it could be shared in the cell layout). Even after separating search and bit lines, driving the search lines accounts for almost half of the energy in search operations.

Apart from having a relatively high capacitance, in parallel CAMs, one search line per bit switches at every search, even when the same

1063-8210/04$20.00 © 2004 IEEE

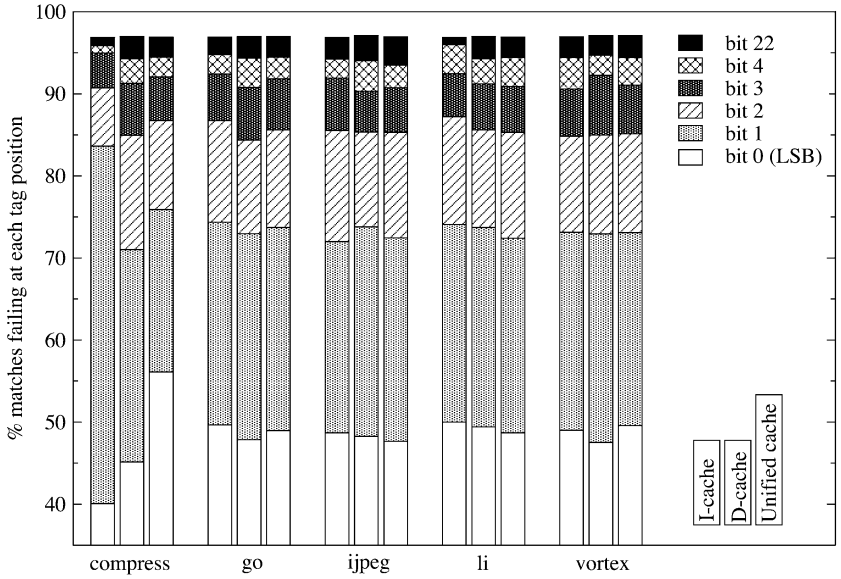

Fig. 1. Percentage of tag checks ending at each bit position.

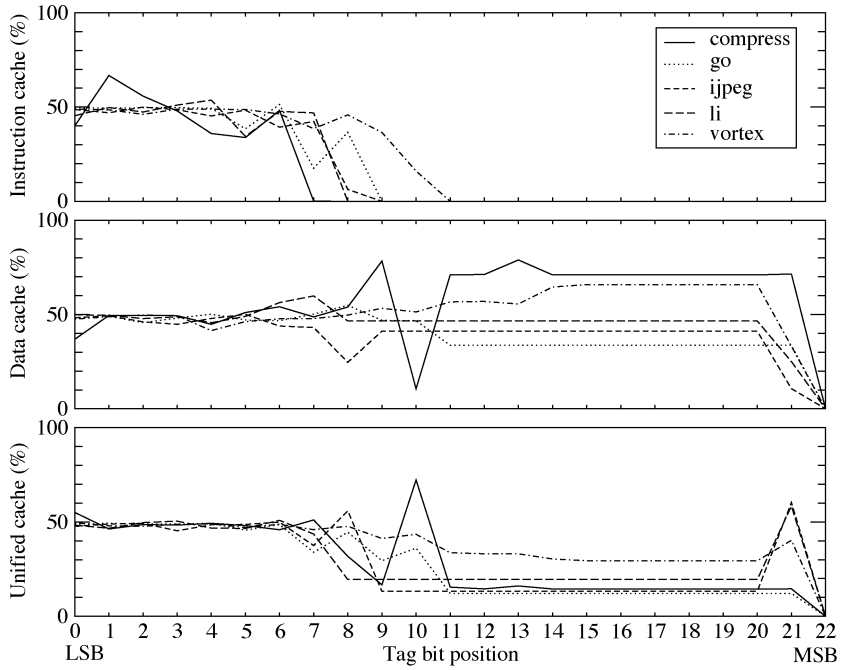

Fig. 2. Percentage of mismatches for each bit position.

value is searched each time. This is because the search lines must be driven low while the match lines are being precharged to avoid a direct short-circuit from supply to ground through the cells that do not match. On the other hand, serial CAMs form chains of transistors that propagate a value when all the cells match; evaluation is coordinated by precharging the intermediate nodes so that the search lines do not have to be precharged (or predischarged) for every search.

In a parallel CAM organization, the only way for the search lines not to be predischarged is to control the "ground" line, into which the match lines are discharged, with a timing signal, making it a "virtual ground" line. Juan *et al.* [7] show an implementation where a transistor is added in every cell in series with those pulling down the match line. Alternatively one transistor per CAM row can be used, which pulls down the virtual ground wire of all the cells in the row. Toggling match-line CAM [8] uses such a scheme where the virtual ground line is shared between two rows and alternate precharging to low or high in order to reduce the toggling of the match lines.

In this case, there would not be a threat of a direct short circuit, but charge sharing between the match line and the virtual ground line (via cells that do not match) must be addressed. In our experience, this is not a hard problem to solve, as explained later. Moreover, serial CAM implementations also suffer from charge sharing; Shafai *et al.* [5] had to predischarge the search lines to make sure that the intermediate nodes in the matching chain were all set to low before evaluation started. Hsiao *et al.* [6] also acknowledge the problem but do not present a solution in their paper.

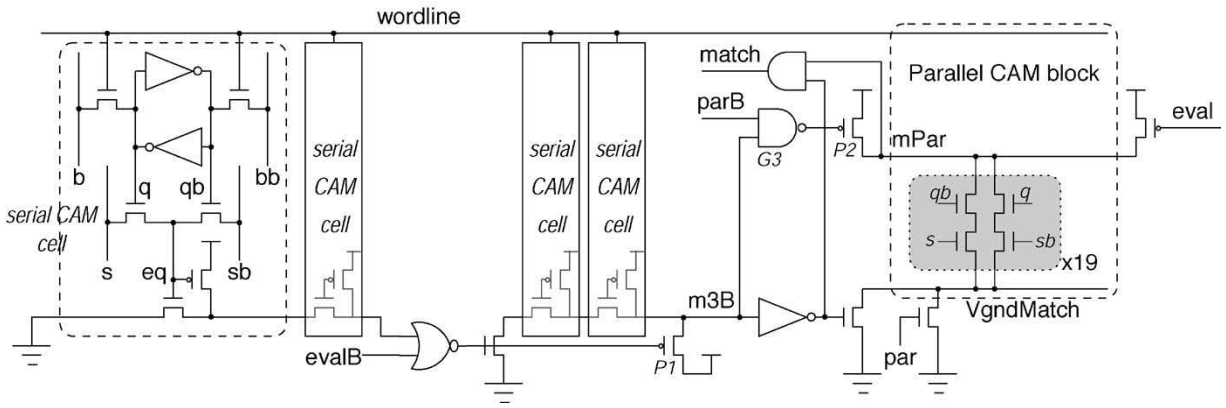

Fig. 3. Row of the proposed CAM.

III. Mixed Serial-Parallel CAM Architecture

Based on the results of the tag matching behavior of the applications, a mixed serial-parallel CAM (SPCAM) organization was implemented with the following design decisions based on the principles described by Hsiao *et al.* [6]:

- A mixed serial-parallel search method is used to minimize the transitions on the match lines.
- The search lines are separate from the bit lines and they are not forced to "0" or "1" while charging the match lines. The latter is accomplished by using a "virtual ground" line as described in Section II-B.
- The use of timing signals is minimized to reduce the energy consumed by their drivers.

A row of the circuit is shown in Fig. 3. Different CAM cells are used for the serial and parallel parts. The parallel CAM uses standard 10T cells [4] with the ground connection of the two pull down nMOS chains testing the equivalence, replaced with *VgndMatch*. This signal runs lengthwise and is connected to all the (parallel) CAM cells in the row. The four bits of the serial part are broken down into two sets of two bits to limit the maximum number of series transistors to three. The match signals of the two parts are ANDed together to produce the final result.

SPCAM can operate either in or mode. In mode, the four LSBs of each row are checked (serially) and, if they all match, the remaining bits are checked in parallel by pulling *VgndMatch* low. In parallel mode, both parts start matching at the same time; the four LSBs are still checked serially, but this operation is overlapped with the parallel matching of the second part, so that the final result is produced sooner.

A. Circuit Operation

In the serial part, a match propagates as a zero from the LSB to the most significant. A cell that matches opens its nMOS pass transistor, propagating the result from its less significant neighbor to its more significant neighbor. If it does not match, it breaks the chain and generates a one to pass on. If the first cell of a set does not match but the second does, a one is propagated through an nMOS transistor. This is allowed through only one device to minimize the speed loss. The gates at the output of these chains are designed so that their input threshold is lowered (with widened nMOS transistors) to compensate for the V_t voltage drop when their inputs are driven to high.

If the cells in the second set match, but those in the first do not, *P1* pulls up the serial-part match signal (*m3B*) signifying a "no match." This eliminates a potential charge sharing problem which could lower the voltage at *m3B* in the above situation.

If the four LSBs all match, the virtual ground *VgndMatch* is pulled down, allowing the rest of the row to evaluate the parallel match line (*mPar*); otherwise, *mPar* is precharged via transistor *P2*. As explained in Section II-A, most of the time *mPar* would not have been discharged, so *P2* just replaces the charge leaked from *mPar*.

TABLE I
Comparison of Tag Implementations

	RAM	CAM	Serial SPCAM	Parallel SPCAM	Serial LL SPCAM
cell height	3.65μm	5.3μm	5.8μm	5.8μm	7.3μm
array size	32×21	32×23	32×23	32×23	32×23
access time	0.5 ns	0.8 ns	1.0ns	0.8ns	1.0ns
energy/acc.	2.3 pJ	12.0 pJ	3.3pJ	8.0pJ	3.7pJ

In operation, the search lines are not predischarged; this saves energy as transitions only occur when the search data actually change. This means that the parallel CAM cells will be evaluating all the time, even when the match line is being precharged, which could lead to charge sharing: If some of these cells do not match, there will be a path from the parallel match line *mPar* to the undriven *Vgndmatch*.

To solve the problem, either the precharge time should be made longer so that the extra capacitance will be charged, or larger precharge transistors should be used that are able to charge the increased capacitance at the same time. Obviously neither of these options is desirable. Instead, we opted not to change the precharge transistor sizes or the precharge time. This could leave *mPar* not fully charged but it will not affect the operation. If the LSBs match, *VgndMatch* will be pulled down and, as some of the parallel part tags do not match, *mPar* will also be discharged. If the LSBs do not match, the final match line will stay low, driven from *m3B*, and the charging will continue, eventually reaching the appropriate level. Simulation indicates that the time between two accesses is sufficient to precharge *mPar* fully when this form of charge sharing happens.

Serial (NAND-type) CAMs generally need timing signals in the cell array to control the precharging and evaluation of the match chains. In the operating mode of SPCAM, the initial intention was not to use timing signals at all. Unfortunately, a situation can occur which can only be overcome by using a timing signal: assume a search where, in some row, the LSBs all match but the MSBs do not. This will leave the row's parallel match line (*mPar*) discharged. If the following search matches the whole row, it would have wrongly reported a mismatch in this row, since *mPar* did not have a chance to precharge.

For this reason *evalB*—the inverse of the parallel block evaluation signal *eval*—is connected to the NOR gate within the LSB match logic; this forces a "no match" for the LSBs, imposing a precharge of the parallel match line (*mPar*) for every search. This implementation was preferred to combining the *evalB* signal with *m3B* in a logic gate to drive *VgndMatch* and *mPar* because it has less capacitive load on *evalB*.

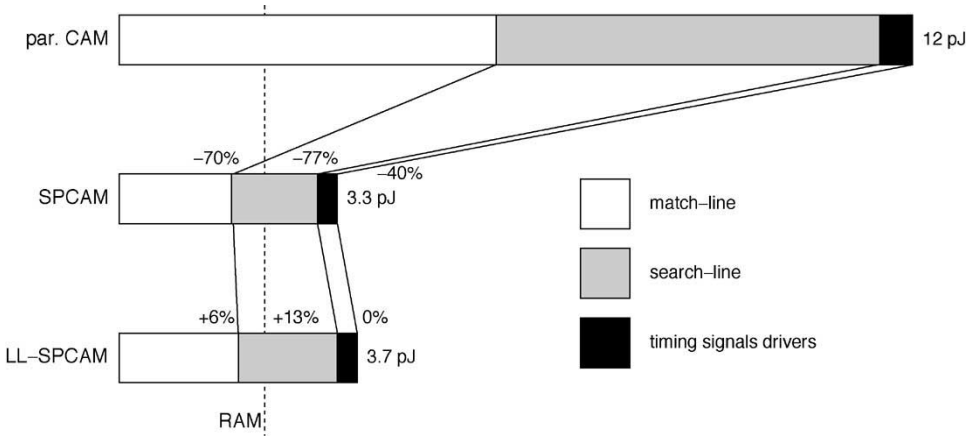

Fig. 4. Comparison of the energy consumption in CAMs.

When the CAM operates in mode, the virtual ground line (*Vgnd-Match*) of the parallel part of the CAM is always connected to the ground. In addition, the NAND gate *G3* isolates the precharging of the parallel part's match line (*mPar*) from the match signal of the serial part, *m3B*. A separate signal *eval* is then used for precharging. This signal is gated off in mode so that its driver does not consume power.

B. Low-Leakage Serial CAM Cell

The serial CAM cell in Fig. 3 can suffer from increased leakage power. When the cell matches, *eq* is driven high through one of the two nMOS transistors, which means that it suffers a V_t drop. Thus, the pMOS transistor driven by *eq* does not turn off completely, allowing leakage.

By replacing the pass transistors with transmission gates, *eq* is pulled high to a full V_{dd} voltage. Since both *q* and *qb* are present, no extra inverters are required. The CAM using these low-leakage cells is called "low-leakage SPCAM."

The added transistors increase the cell size considerably. Since the original floor-plan assumed that one CAM cell is pitch-matched with two RAM cells vertically, the cell area increase—just within the limit of two RAM cells—does not increase the cache bank area. Moreover, the search lines are not made longer, so there is no increase in their interconnection capacitance due to the increased cell size.

The low-leakage cells impose an increased capacitance on the search lines: a pMOS transistor drain per row is added and the capacitance of *eq*—which is also driven by one of the two searchlines—is also increased since it is connected to two extra transistors and its wiring is longer. The effect of this on the energy consumption is presented in Section III-C.

C. Evaluation

Table I summarizes the results for all the tag implementations, including the new low-leakage SPCAM, simulated in a 0.18-μm technology using Spectre. The array sizes and the cell height (the width is fixed) are also shown, to give an estimate of the tag block area. The distribution of mismatches to bit positions follows the findings of Section II-A1.

In serial mode, the energy consumption of the SPCAM is only 45% more than the RAM, excluding the decoder and comparator; this is almost a quarter of the standard CAM energy consumption. The cycle time is twice that of the RAM, but only 25% longer than the original CAM. Thus, SPCAM is much more energy efficient than the conventional CAM. In mode, the energy consumption is 3.5 times that of the RAM, still 33% better than the conventional parallel CAM, while the speed is the same.

Fig. 4 presents the energy consumption in the three CAM implementations. As expected, the low-leakage version of SPCAM (LL-SPCAM) consumes about 12% more dynamic energy than the original SPCAM, which is due to the increased capacitance on the search lines. In comparison to the RAM, LL-SPCAM consumes 61% more energy; this is still over three times less than the fully-parallel CAM design.

D. Implications for Cache Organization

With a CAM search energy consumption so close to that of a single RAM read, the fully associative cache organization becomes a lower energy choice compared to (pseudo) associative, way predicting caches. Caches with conventional CAMs are reported to have a similar access time to caches with RAM tags [9], thus, the effect of the decoder and the comparator must slow down the RAM-based designs to a similar speed to the CAM-based ones. With the results presented here, an SPCAM in mode is only 25% slower than a conventional CAM.

If this speed reduction is transferred to the clock cycle, such a cache does not seem attractive to high-speed processor architects. Since only the tag look-up operation is slowed down, it could be possible that the effect on the clock cycle would be less than the full 25% increase. For example in a two-cycle access (possibly pipelined) cache, with the first cycle allocated to the tag access and the other to the data SRAM access, it is likely that the tag access cycle could be underutilized and, thus, increasing it by 25% may not increase the system's cycle time.

The proposed CAM is able to switch from to mode, trading energy for speed. Thus, it could be normally operated in the full speed, mode, while the low-power, low-speed mode could be used in a power-saving system mode, e.g. for portable devices when operating on battery. The speed reduction in the memory system could then be balanced with a similar speed drop at the processor, combined with reduced activity for extra energy savings [10]. The clock generator will have to be able to produce the appropriate frequencies, though. As our design was originally intended for asynchronous processors, the variation in speed can be accommodated easily.

IV. Comparison to Related Work

A comparator circuit for a dispatch buffer design [11] compares the four LSBs first and then evaluates the rest of the eight bits in two cascaded domino stages, each of which is evaluated only if the previous one signals a match. This approach could be thought of as generating 2-b parallel match signals (4 b for the LSB) and serially evaluating the intermediate results. Compared to the solution proposed here it requires an extra voltage source, has long transistor stacks (notably four pMOS in series) and needs more timing signals.

The CAM design by Hsiao *et al.* [6] has good low-power properties since it evaluates the match lines serially (NAND-type) and does not require discharging of the search lines while the match line is precharged. However precharging and evaluating the match line segments requires more "clocking" power than the design proposed here. They report a 45.5 fJ/bit/search at 12-ns cycle time in a. 35-μm, 3.3-V technology. Converting their energy per bit per search result to our technology suggests about 11 fJ/bit/search, which is over twice that of the SPCAM in mode.

Juan *et al.* [7] report 40% to 60% power reduction on a small translation look-aside buffer (TLB). They use a fully parallel organization but they avoid having to predischarge the search lines by adding an nMOS transistor in every cell, controlled by a global timing signal, in series with those pulling down the match line. This greatly increases the energy consumed in driving timing-signals and possibly increases the CAM cell size. To reduce the power on the match lines, they restrict the lines which are allowed to do a comparison, thus effectively limiting the associativity.

Zhang and Asanovic [9] argue that CAM-based caches are preferable for low-power processors. They describe a CAM design with separate bit and search lines and they precharge the match lines through nMOS transistors to reduce voltage swing. As a speed enhancement they split each match line into two segments which, in view of the analysis here, would also save energy as the most significant part will be discharged less frequently. For further speed improvement they employ single-ended sense amplifiers on both segments of the match lines, but these are likely to consume significant power. Although the energy consumed in the tags is not directly compared, they showed that the total energy consumption of this cache is similar to a two-way associative conventional (RAM-based) cache. Their CAM-based cache has an almost identical performance to a conventional RAM-based cache, which is not phased, i.e., all tags and data are read in parallel in all cache ways.

Burd [12] presented a CAM which consumes twice the energy of an equivalent-sized RAM. He argues that caches using this CAM consume the same energy as a conventional two-way set associative cache built with RAM tags, so that the CAM-based design is preferable since a higher associativity is needed for his design.

V. Conclusions

A new serial-parallel CAM (SPCAM) design has been proposed which consumes about a quarter of the energy of a conventional low-power CAM, when used as a cache tag store/comparator. It exploits the address patterns commonly found in application programs, where testing the four LSBs of the tag is sufficient to determine over 90% of the tag mismatches; the proposed CAM checks those bits first and evaluates the remainder of the tag only if they match. In addition the search lines do not have to be forced to "0" or "1" while precharging the match line, which accounts for almost half of the energy of a conventional CAM. The proposed CAM is also adaptive, i.e., it can be configured to work serially as described above or it can operate as a parallel CAM with lower energy benefit than in serial mode, but at the same speed as a conventional CAM.

SPCAMs energy consumption is comparable to that of reading a RAM of similar capacity. Thus, using this CAM for the tag parts of a subblocked, highly associative cache, could make this cache more energy efficient than way predicting (pseudo) associative caches. Naturally comparing CAM-based and RAM-based caches, based on the speed and energy per tag access is not enough to produce concluding results for which architecture is the best. As the cycle time could be affected, the energy saved in the memory could be less than the increase in the energy spent in the processor. More work is, therefore, needed to create accurate models of the energy consumption of the cache designs and also take into consideration the effect on the processor core.

Acknowledgment

The authors thank K. Asanovic for pointing out the potential power leakage problem in the first SPCAM cell design, as well as the reviewers and the special issue editors for their helpful comments.

References

[1] J. Montanaro, R. Witek, K. Anne, A. Black, E. Cooper, D. Dobberpuhl, P. Donahue, J. Eno, W. Hoeppner, D. Kruckemyer, T. Lee, P. Lin, L. Madden, D. Murray, M. Pearce, S. Santhanam, K. Snyder, R. Stehpany, and S. Thierauf, "A 160-MHz, 32-b, 0.5-W CMOS RISC microprocessor," *IEEE J. Solid-State Circuits*, vol. 31, pp. 1703–1714, Nov. 1996.

[2] K. Inoue, T. Ishihara, and K. Murakami, "Way-predicting set-associative cache for high performance and low energy consumption," in *Proc. Int. Symp. Low-Power Electronics Design*, Aug. 1999, pp. 273–275.

[3] A. Efthymiou and J. D. Garside, "An adaptive serial-parallel CAM architecture for low-power cache blocks," in *Proc. Int. Symp. Low Power Electronics Design*, Aug. 2002, pp. 136–141.

[4] K. J. Schultz, "Content-addressable memory core cells a survey," *VLSI J. Integration*, vol. 23, no. 2, pp. 171–188, Nov. 1997.

[5] F. Shafai, K. Schultz, G. Gibson, A. Bluschke, and D. Somppi, "Fully parallel 30-MHz, 2.5-Mb CAM," *IEEE J. Solid-State Circuits*, vol. 33, pp. 1690–1696, Nov. 1998.

[6] I. Y.-L. Hsiao, D.-H. Wang, and C.-W. Jen, "Power modeling and low-power design of content addressable memories," in *Proc. Int. Symp. Circuits Systems*, vol. 4, May 2001, pp. 926–929.

[7] T. Juan, T. Lang, and J. J. Navarro, "Reducing TLB power requirements," in *Proc. 1997 Int. Symp. Low Power Electronics Design*, 1997, pp. 196–201.

[8] G. Thirugnanam, N. Vijaykrishnan, and M. J. Irwin, "A novel low power CAM design," in *Proc. Int. ASIC/SOC Conf.*, 2001, pp. 198–202.

[9] M. Zhang and K. Asanovic, "Highly-associative caches for low-power processors," in *Proc. Kool Chips Workshop with MICRO-33*, Dec. 2000.

[10] A. Efthymiou and J. D. Garside, "Adaptive pipeline depth control for processor power-management," in *Proc. Int. Conf. Computer Design*, Sept. 2002, pp. 454–457.

[11] G. Kucuk, K. Ghose, D. V. Ponomarev, and P. M. Kogge, "Energy: efficient instruction dispatch buffer design for superscalar processors," in *Proc. 2001 Int. Symp. Low Power Electronics Design*, 2001, pp. 237–242.

[12] T. Burd, "Energy-efficient processor system design," Ph.D. dissertation, Dept. Elect. Eng. Comput. Sci., Univ. California, Berkeley, May 2001.