Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 12, December 2014, pg.56 – 64

RESEARCH ARTICLE

AN ENERGY EFFICIENT CACHE DESIGN TECHNIQUE FOR EMBEDDED PROCESSORS

R.Azhagarpandi[1], Dr. G Mohanbabu[2]
[1]PG Scholar, [2]Assistant Professor Department of Electronics and Communication Engineering,
PSNA college of Engineering and Technology, Dindigul, Tamilnadu, India
[1] azhagarpandi91@gmail.com; [2] shyambabu@gmail.com

Abstract-- In this paper, a new cache design technique, referred to as early tag access (ETA) cache, to improve the energy efficiency of data caches in embedded processors. The proposed technique of ETAs to determine the destination ways of memory instructions before the actual cache accesses. It enables only the destination way to be accessed if a hit occurs during the ETA. The proposed early tag access cache can be configured under two operation modes to exploit the trade-offs between energy efficiency and performance. This shown that our technique is very effective in reducing the number of ways accessed during cache accesses. It enables significant energy reduction with negligible performance overheads.

Keywords: early tag access (ETA), energy efficiency, cache, embedded processor

1. INTRODUCTION

Multi-level on-chip cache systems have been widely adopted in high-performance microprocessors [1]–[3]. To retain data consistence throughout the memory hierarchy, write-through, write-back policies are commonly utilized. Under the write-back policy, a modified cache block is copied back to its corresponding lower level cache only when the block is about to be restored. While under the write-through policy, all copies of a cache block are up to date immediately after the cache block is modified at the current cache, even though the block might not be removed. As a result, the write-through policy maintains identical data copies at all levels of the cache hierarchy throughout most of their life time of execution. This characteristic is important as CMOS technology is scaled into the nanometer range, where soft errors have appeared as a major reliability issue in on-chip cache systems. It has been described that single-event multi-bit upsets are getting worse in on-chip memories [7]–[9]. Currently, this problem has been directed at different levels of the design abstraction. At the architecture level, an effective solution is to keep data consistent among different levels of the memory hierarchy to prevent the system from collapse due to soft errors [10]–[12]. Benefited from immediate up to date, cache write-through policy is inherently tolerant to soft errors because the data at all related levels of the cache hierarchy are always kept consistent. Due to this characteristic, many high-performance microprocessor designs have adopted the write-through policy [13]–[15]. While enabling better tolerance to soft errors, the write-through policy also suffers large amount of energy overhead, because of under the write-through policy, the caches at the lower level experience more accesses during write operations. Consider a two-level (i.e., Level-1 and Level-2) cache system for example. If the L1 data caches apply the write-back policy, a write hit in the L1 cache does not require accessing the L2 cache. In disparity, if the L1 cache is write-through, then both L1 and L2 caches need to be obtained for every write operation. Obviously, the write-through policy suffers more write accesses in the L2 cache, which increases the energy consumption of the cache system. Power dissipation is now contemplated as one of the critical issues in cache design. Studies have shown, on-chip caches can consume about 50% of the total power in high-performance microprocessors [4]–[6].

In this paper, a new cache technique, referred to as early tag access (ETA) cache, to improve the energy efficiency of L1 data caches. In physical tag and virtual index cache, a part of the physical address is stowed in the tag arrays while the conversion between the virtual address and the physical address is performed by the TLB. By accessing tag arrays and TLB during the LSQ stage, the destination ways of most memory instructions can be determined before accessing the L1 data cache. As a result, the one way in the L1 data cache needs to be accessed for these instructions, thereby reducing the energy consumption significantly. Note that the physical addresses generated from the TLB at the LSQ stage can also be used for subsequent cache accesses. Therefore, for most memory instructions, the energy overhead of way determination at the LSQ stage can be compensated for by skipping the TLB accesses during the cache access stage. For memory instructions whose destination ways cannot be determined at the LSQ stage, an enhanced mode of the ETA cache is proposed to reduce the number of ways accessed at the cache access stage. Note that in many high-end processors, accessing L2 tags is done in parallel with the accesses to the L1 cache [2]. Our technique is fundamentally different as ETAs are performed at the L1 cache.

2. PROPOSED ETA CACHE

In a conventional set-associative cache, all ways in the tag and data arrays are accessed simultaneously. The requested data, however, only resides in one way under a cache hit. The extra way accesses incur unnecessary energy consumption. In this section, new cache architecture referred to as ETA cache will be developed. The ETA cache reduces the number of unnecessary way accesses, thereby reducing cache energy consumption. To accommodate different energy and performance requirements in embedded processors, the ETA cache can be operated under two different modes: the basic mode and the advanced mode.

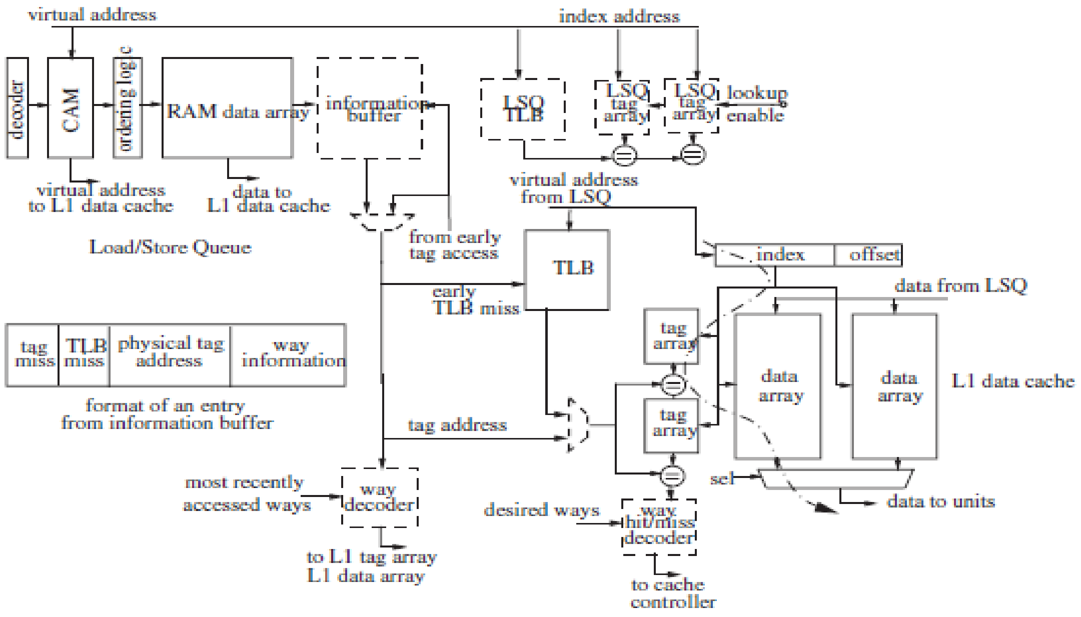


Fig.1. Proposed ETA cache (blocks in dash line are new components and Dotted line is the timing critical path).

A.LSQ TAG ARRAYS and LSQ TLB

To avoid the data contention with the L1 data cache, the LSQ tag arrays and LSQ TLB are implemented as a copy of the tag arrays and TLB of the L1 data cache, respectively. There are two types of operations in the LSQ tag arrays and LSQ TLB: lookup and update. Each time a memory address reaches the LSQ, the LSQ tag arrays and LSQ TLB will be searched for the early destination way. In case of a hit, the early destination way will be available; otherwise, the instruction will cause either an early tag miss For update operations, the contents of LSQ tag arrays and LSQ TLB are updated with the tag arrays and TLB of the L1 cache, so that they are identical to avoid cache coherence problems. The update logic of LSQ tag arrays and LSQ TLB is the same as that of the tag arrays and TLB of the L1 cache.

Fig.2 shows the implementation of the LSQ tag arrays, where only one way is shown as the other ways are the same Consider that generally at most N instructions can enter the LSQ while the L1 data cache allows M replacements to occur at the same time. Therefore, there might be at most N lookup operations and M update operations occurring at the LSQ tag arrays and LSQ TLB at the same time. In order to perform these operations simultaneously, the LSQ tag arrays and LSQ TLB have N read ports and M write ports.

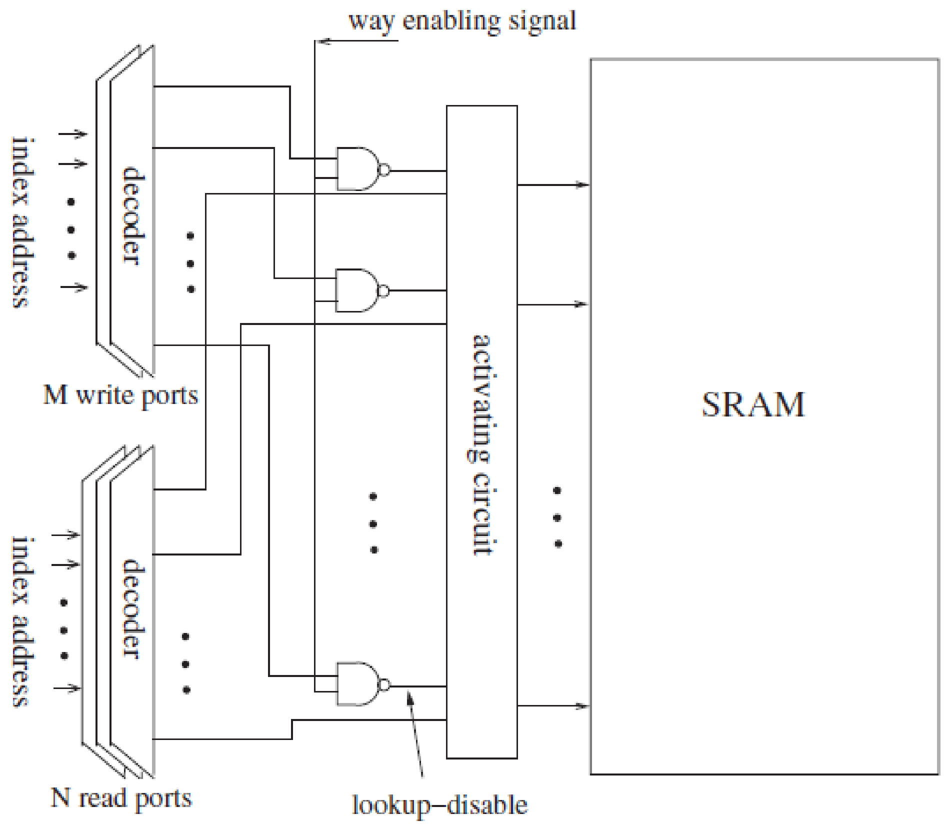


Fig.2. Implementation of LSQ tag arrays (only one way is shown for Simplicity).

In the simulations both M and N are chosen to be two for the purpose of demonstration. Write/read conflicts occur when the lookup and update operations target the same location of the LSQ tag arrays at the same time. To address this issue, we disable the lookup operation if an update operation is currently performed. This is achieved by the control signal lookup-disable, which are generated by the way enabling signals from the cache controller for cache replacements. Consider a two-way set-associative cache for example. Assume that there is a replacement occurring at the way 1 of the L1 data cache. As a result, the way enabling signal is set to “1” and then sent to the NAND gates in the way 1 of the LSQ tag arrays. If the write decoder outputs a “0,” i.e., no update operation on this entry of the tag array, the lookup-disable signal will be set to “1” and the activating circuit will not block the lookup operation on this entry. Otherwise the lookup-disable signal will be “0,” and the activating circuit will block possible lookup operations to avoid write/read conflicts.

B. INFORMATION BUFFER

This requires an information buffer to hold the early destination way until the corresponding load/store instruction is issued to the L1 data cache. The implementation of the information buffer is shown in Fig.3. It has the same number of entries as the LSQ. In each entry, the tag miss and TLB miss bits (“1” for hit and “0” for miss) indicate the status of early tag and early TLB accesses, respectively. The early destination way is stored in the way information bits. The physical tag address is generated by searching the LSQ TLB. For a memory instruction with an early tag hit, the access to the actual TLB during the cache access stage can be avoided by sending this physical tag address to the data cache. Since the information buffer is small (32 entries with 20 bits per entry in a typical configuration), the induced energy overhead can be easily offset by the energy savings from the cache access stage.

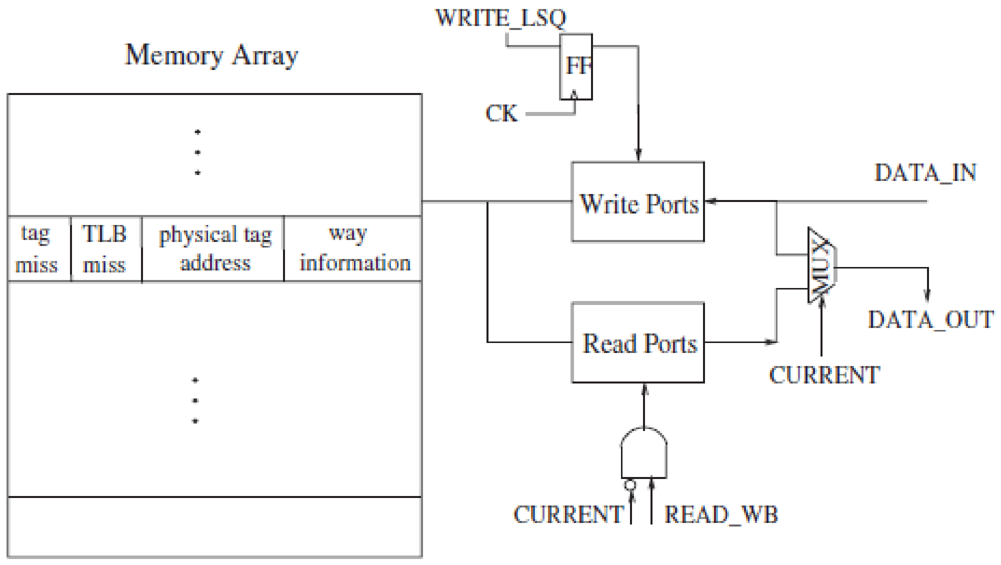
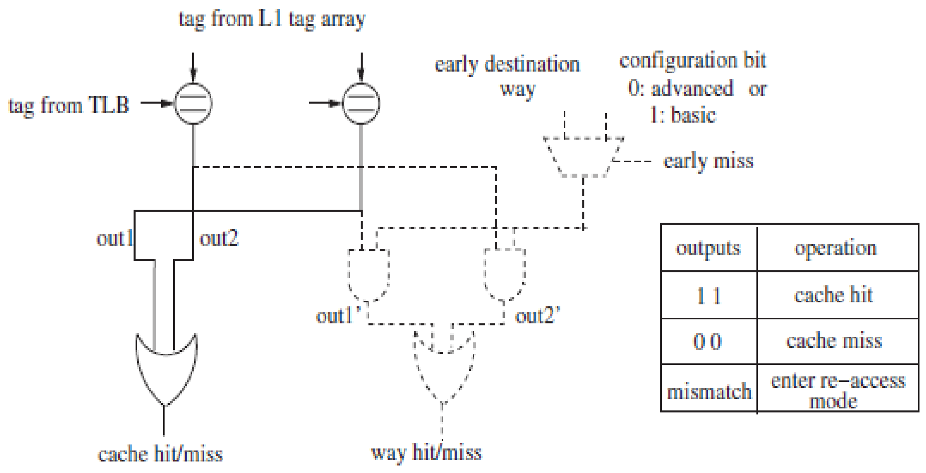


Fig.3. Implementation of information buffer.

The information buffer has separate write and read ports to support parallel write and read operations. The write operations of the information buffer always start one clock cycle later than the corresponding write operations in the LSQ. This is because the accesses to the LSQ, LSQ tag arrays, and LSQ TLB occur simultaneously. Since the way information is available after the write operations in the LSQ, this information will be written into the information buffer one clock cycle later than the corresponding write operation in the LSQ.

C. WAY HIT/MISS DECODER



outputs	operation
1 1	cache hit
0 0	cache miss
mismatch	enter re–access mode

Fig.4. Implementation of way hit/miss decoder.

If a cache coherence problem is detected, an additional access to the L1 data cache is required. Here, we introduce a way hit/miss decoder to determine whether the additional access is necessary. Fig.4 shows the implementation of this decoder with dotted lines. A conventional cache hit/miss decoder is also shown with solid lines. The configuration bit is used to set the ETA cache for the basic mode or the advanced mode. As shown in Fig.4, if both the cache hit/miss and way hit/miss signals indicate a hit (e.g., "1"), the cache access is considered a hit.

D.WAY DECODER

In the proposed ETA cache, way enabling signals are needed to control the access to the ways in the data arrays. Fig. 5 shows the implementation of the way decoder that generates these signals. When the instruction is associated with an early hit (e.g., "1"), the data arrays need to be accessed according to the early destination way. If the instruction experiences an early tag miss or an early TLB miss, the configuration bit shown in Fig. 14 determines which way in the data arrays o the L1 data cache needs to be accessed. Specifically, by setting the configuration bit to "1," the ETA cache will operate under the basic mode.

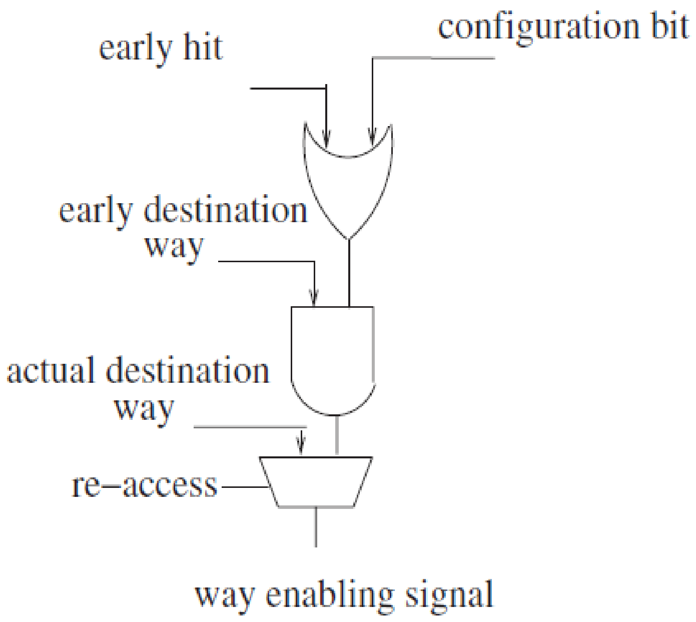
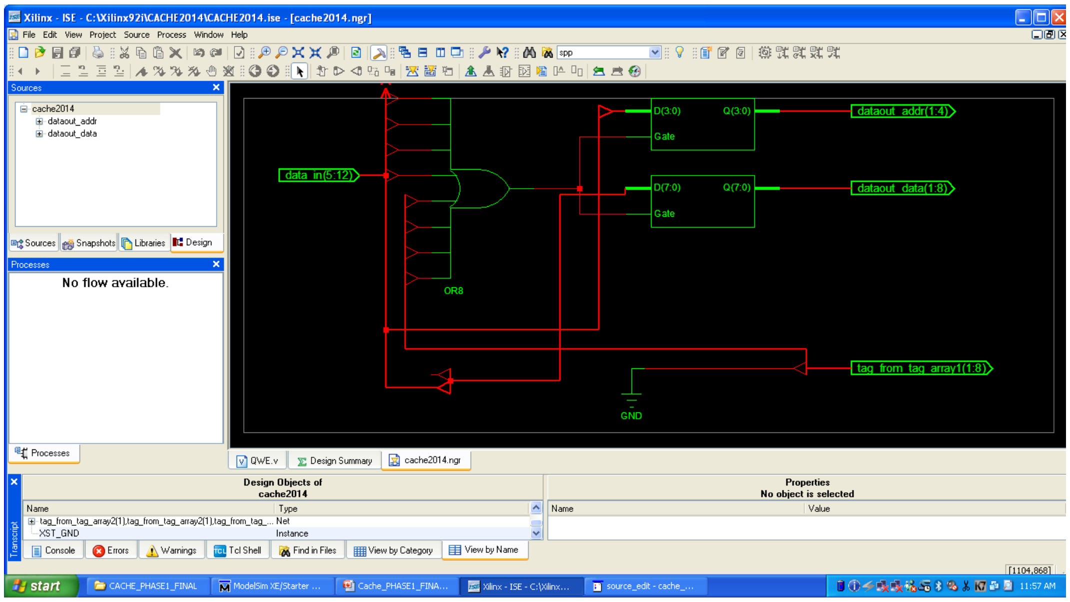


Fig.5. Implementation of way decoder.

3. RESULT AND DISCUSSION

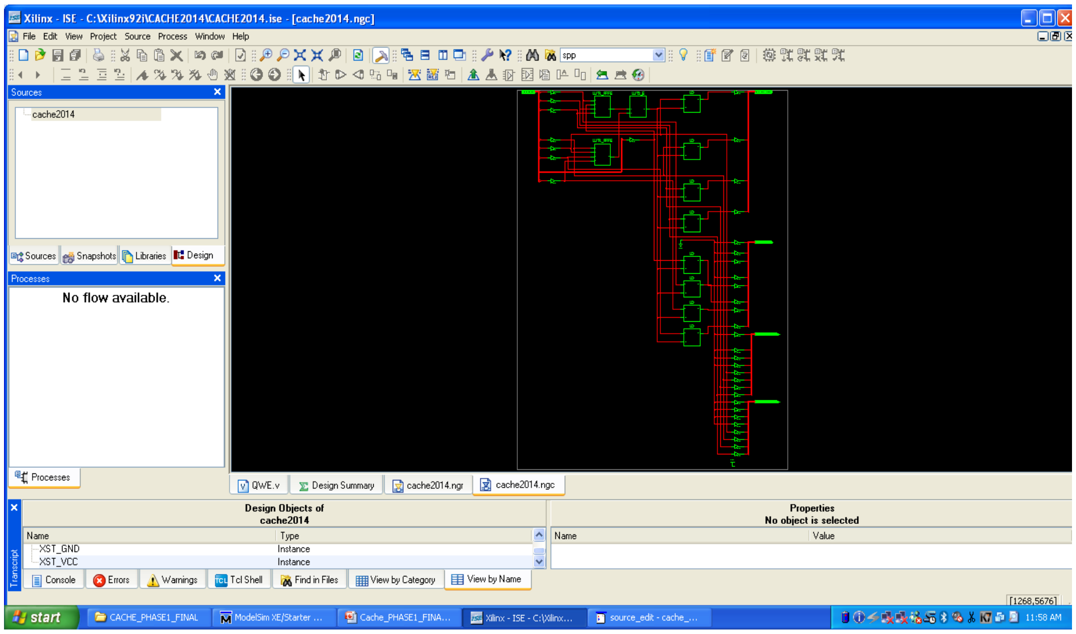
The proposed Cache architecture and its hardware architecture system is designed and tested on various version of Spartan-3E family device using Modelsim 6.1 and synthesis tool Xilinx 9.2i. This proposed scheme utilized 1728 gates and 155996KB memory usage at a maximum frequency of 120MHz. The proposed work results shows that the system incorporated with its hardware architecture leads to lower power consumption in terms of gates and Flip Flops. The proposed cache architecture is implemented in 90nm CMOS technology. The post layout results of the proposed cache architecture are summarized in the following sections, and the chip layout is shown in Figure.6&7. The power consumption is 203mW

A number of performance evaluation and resource utilization parameters are being used in the design of proposed cache architecture. The present research is focused on the design and development of efficient hardware architecture for low power applications. The parameters considered for investigation include latency and Power consumption (PC).



Name	Type
tag_from_tag_array2(1),tag_from_tag_array2(1),tag_from_tag_...	Net
XST_GND	Instance

Fig.6.RTL Schematic view



Name	Type
XST_GND	Instance
XST_VCC	Instance

Fig.7.Technology Schematic View

4. PERFORMANCE EVALUATION RESULT

Table.1. performance evaluation result

PERFORMANCE PARAMETERS	ACHIEVED RESULT
Power consumption	203mw
Memory usage	155996KB
Latency	5.077ns
Gate counts	1728

5. CONCLUSION

This paper presented a new energy-efficient cache design technique for low-power embedded processors. The proposed ETA technique predicts the destination way of a memory instruction at the early LSQ stage. Thus, only one way needed to be accessed during the cache access stage if the prediction is correct, thereby reducing the energy consumption significantly. By applying the idea of phased access to the memory instructions whose early destination ways cannot be determined at the LSQ stage, the energy consumption can be further reduced with negligible performance degradation. The Simulated results demonstrated the efficacy of the proposed ETA technique as well as the performance impact and design overhead. Our technique was demonstrated by a L1 data cache design, future work is being directed toward extending this technique to other levels of the cache hierarchy and to deal with multithreaded workloads.

REFERENCES

[1] M. Powell, A. Agarwal, T. Vijaykumar, B. Falsafi, and K. Roy, “Reducing set-associative cache energy via way-prediction and selective direct mapping,” in *Proc. 34th Int. Symp. Microarchit.* , pp. 54–65. 2001.

[2] C. Zhang, F. Vahid, and W. Najjar highly-configurable cache architecture for embedded systems,” in *Proc. 30th Annu. Int. Symp. Comput. Archit.*, pp. 136–146, Jun. 2003.

[3] S. Segars, “Low power design techniques for microprocessors,” in *Proc. Int. Solid-State Circuits Conf. Tuts.*, Feb. 2001.

[4] S. Manne, A. Klauser, and D. Grunwald, “Pipline gating: Spculation conrol for energy reduction,” in *Proc. Int. Symp. Comput. Archit.*, , pp. 132–141, Jun.–Jul. 1998.

[5] M. Gowan, L. Biro, and D. Jackson, “Power considerations in the design of the alpha 21264 microprocessor,” in *Proc. Design Autom. Conf.*, pp. 726–731, Jun. 1998.

[6] A. Malik, B. Moyer, and D. Cermak, “A Low power unified cache architecture providing power and performance flexibility,” in *Proc. Int. Symp. Low Power Electron. Design*,, pp. 241–243,2000.

[7] T. Lyon, E. Delano, C. McNairy, and D. Mulla, “Data Cache Design Considerations for the Itanium Processor,” in *Proc. IEEE Int. Conf. Comput. Design, VLSI Comput. Process.* , pp. 356–362, 2002.

[8] D. Nicolaescu, A. Veidenbaum, and A. Nicolau, “Reducing power consumption for high-associativity data caches in embedded processors,” in *Proc. Design, Autom., Test Eur. Conf. Exhibit.*, , pp. 1064–1068,
Dec. 2003.

[9] C. Zhang, F. Vahid, Y. Jun, and W. Najjar, “A way-halting cache for low-energy high-performance systems,” in *Proc. Int. Symp. Low Power Electron. Design*, , pp. 126–131, Aug. 2004.

[10] J. Montanaro, R. T. Witek, K. Anne, A. J. Black, E. M. Cooper, D. W. Dobberpuhl, P. M. Donahue, J. Eno, W. Hoeppner, D. Kruckemyer, T. H. Lee, P. C. M. Lin, L. Madden, D. Murray, M. H. Pearce, S. Santhanam, K. J.

Snyder, R. Stehpany, and S. C. Thierauf, “A 160-MHz 32-b 0.5- W CMOS RISC microprocessor,” *IEEE J. Solid-State Circuits*, vol. 31, no. 11, pp. 1703–1714, Nov. 1996.

[11] S. Santhanam, A. J. Baum, D. Bertucci, M. Braganza, K. Broch, T. Broch, J. Burnette, E. Chang, C. Kwong-Tak, D. Dobberpuhl, P. Donahue, J. Grodstein, K. Insung, D. Murray, M. Pearce, A. Silveria, D. Souydalay, A. Spink, R. Stepanian, A. Varadharajan, V. R. van Kaenel, and R. Wen, “A low-cost, 300-MHz, RISC CPU with attached media processor,” *IEEE J. Solid-State Circuits*, vol. 33, no. 11, pp. 1829–1838, Nov. 1998.

[12] D. Brooks, V. Tiwari, and M. Martonosi, “Wattch: A framework for architectural-level power analysis and optimizations,” in *Proc. Int. Symp. Comput. Archit.*, pp,83–94. Jun. 2000.

[13] A. Hasegawa, I. Kawasaki, K. Yamada, S. Yoshioka, S. Kawasaki, and P. Biswas, “SH3: High code density, low power,” *IEEE Micro*, vol. 15, no. 6, pp. 11–19, Dec. 1995.

[14] J. Dai and L. Wang, “An energy-efficient L2 cache architecture using way tag information under write-through policy,” *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 21, no. 1, pp. 102–112, Jan. 2013.