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. 10, October 2014, pg.304 – 310

RESEARCH ARTICLE

AN ENERGY EFFICIENT DATACACHE EMBEDDED PROCESSOR

B.BalaBhuvanapriya, bhuvi.cloud9@gmail.com, **S.Udhayakumar,** udhaya2005@gmail.com
Department Of Electronics and Communication Engineering
Sri Eshwar College of Engineering, Kinathukadavu, Coimbatore-641202

ABSTRACT: This paper presents a new cache design technique, referred to as early tag access (ETA) cache, to improve the energy efficiency of data caches in embedded processors, to determine the destination ways of memory instructions before the actual cache accesses. It, thus, enables only the destination way to be accessed if a hit occurs during the ETA. The new ETA cache can be configured under two operation modes to exploit the tradeoffs between energy efficiency and performance. It is shown that our technology is very effective in reducing the number of ways accessed during cache accesses. The ETA cache achieves over 52.8% energy reduction on average in the L1 data cache and translation look aside buffer. It is more effective in energy reduction while maintaining better performance and this technique is used to other levels of cache hierarchy and deals with multi threaded workloads.

Keywords: Cache, Low power, LSQ Tag array and TLB

I. INTRODUCTION

A cache is a component that transparently stores data so that future requests for that data can be served faster. The data that is stored within a cache might be values that have been computed earlier or duplicates of original values that are stored elsewhere. If requested data is contained in the cache (cache hit), this request can be served by simply reading the cache, which is comparatively faster. Otherwise (caches miss), the data has to be recomputed or fetched from its original storage location, which is comparatively slower. Hence, the greater the number of requests that can be served from the cache, the faster the overall system performance becomes. Here Miss/Hit occurs means data cache is used. Small amount of fast memory Sits between normal main memory and CPU May be located on CPU chip or module. In data's are continuously stored like RAM. In L2 data's are store just like Hard disk. Cache memory is fast and it is expensive. It is categorized in levels that describe its closeness and accessibility to the microprocessor. Level 1 (L1) cache, which is extremely quick but relatively small, is located close to the processor and it is used to access the easy and fast. Level 2 (L2) cache is located half-way between the process and the system bus; it is fairly high-speed and medium-sized. The modified load store queue "caches" all previously accessed data values going beyond existing store-to-load forwarding techniques. Both load and store data are placed in the LSQ and are retained there after a corresponding memory access instruction has been committed .A memory management unit (MMU) that fetches page table entries from main memory has a specialized cache, used for recording the results of virtual address to physical address translations. This specialized cache is called a translation look aside buffer (TLB).Dividing the cache into separate tag and data arrays reduces the access time of the cache. The tag array typically contain many fewer bits than the data array can therefore be accessed more quickly than either the data array or a single combined tag/data array

II. SURVEY OF RELATED WORK

In [1], [2] describes the problem of energy consumption in data caches of contemporary entrenched systems a cache policy to diminish performance penalty techniques to identify data regions and rate their Criticality using an energetic critical pathway model to design an optimization that can lessen energy use in a data cache.[3] a new cache architecture referred to as way-tagged cache to improve the energy efficiency of write-through caches.. The way tag is sent to the way-tag arrays in the L1 cache when the data is loaded from the L2cache to the L1 cache. Utilizing the way tags stored in the way-tag arrays, the L2 cache can be accessed as a direct-mapping cache during the following inscribe hits, thus dropping cache energy consumption. High-power microprocessor design is the biggest problem encountered, and current multi-core, multithreaded processor architecture exacerbated the urgency of issue power consumption. High-power will limit the routine improve of the processor, if want to extra improve the frequency or increase the store size, processor power will continue rising, and then into a aggressive circle. In front of high-power pressure of multi-core, multi-threaded processors, low power design has become the core issue in the future microprocessor design. Multi-threaded processor design optimizing, computing employment and the power frequency of use of a solitary transistor radically boost up; the new low power devices will play a more important role in dipping the leakage control and switching power. In the phased tag cache, the tag is compared in two phases[4].A 4-MB L2 Cache is integrated with a 64-bit 1.6-GHz RISC Microprocessor. The expertise used a 90-nm node transistor with eight layers of metal. Familiar write and read circuits were used for Data, Tag, and LRU L2 cache blocks. Redundancy was implementing in order to exploit inclusion flexibility and to accommodate this great eight-layer dense metal cache [5].

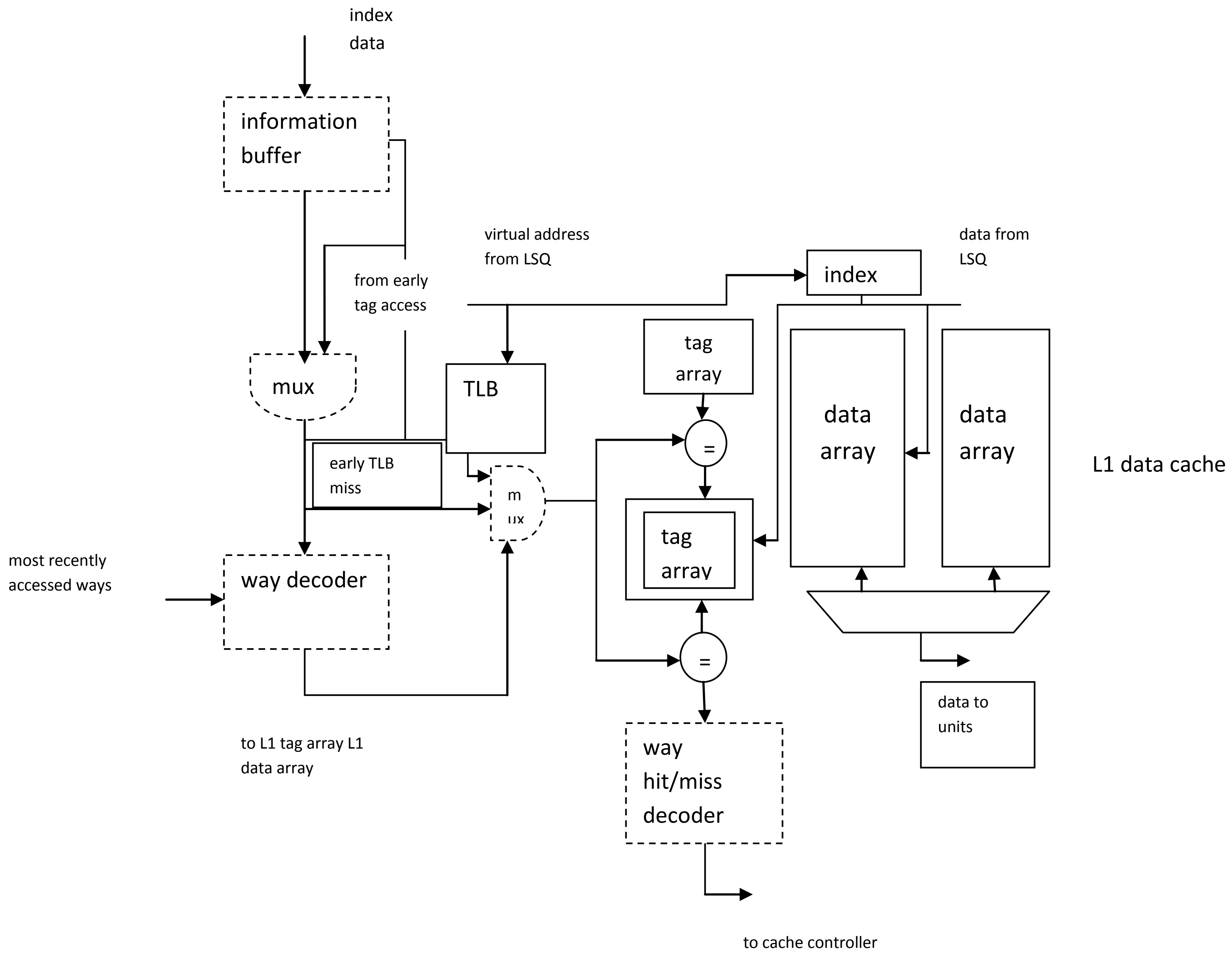
In [6-8], a method of saving energy by reducing the number of data cache access. It does so by modifying the Load/Store Queue design to allow caching of earlier access data values on both loads and stores after the consequent memory access instruction has been dedicated. It is revealed that a 32-entry LSQ design allow an average of 38.5% of the heaps in the SpecINT95 benchmark and 18.9% in the SpecFP95 benchmarks to get their data from the LSQ. The reduction in the number of L1 cache accesses results in up to a 40% reduction in the L1 data cache energy exploitation and in an up to a 16% improvement in the force remain product while involve about no further hardware or composite control be applied to other types of Load/Store Unit organization: units that enclose divide queues for loads and supplies, and unit that store full cache lines in each entry. Set-associative caches achieve low miss rates for typical applications but result in significant energy dissipation. Set-associative caches minimize access time by probing all the data ways in parallel with the tag lookup, although the output of only the matching way is used. The energy spent accessing the other ways is wasted. Eliminating the wasted energy by performing the data lookup sequentially following the tag lookup substantially increases cache access time, and is unacceptable for high-performance Ll caches. Be appropriate two previously-proposed techniques, way-prediction and selective direct-mapping, to reducing LI cache dynamic energy while maintaining high performance. The techniques predict the matching way and explore only the predicted way and not all the ways, achieve energy savings. While these techniques were originally proposed to improve set-associative cache access times, to apply them to reducing cache energy. We assess the effectiveness of these techniques in sinking L1 cache, and overall processor energy. Using these techniques, our caches achieve the energy-delay of sequential access while maintain the concert of equivalent access. Relative to parallel access LI cache, the technique achieve overall processor energy-delay decline of 8%, while great way-prediction with no performance degradation achieves 10% fall. The routine humiliation of the technique is less than 3%, compared to an aggressive, 1-cycle, 4-way, and parallel access cache [7]. A technique for reducing the D–cache power consumption and shows its collision on power and performance of an embedded processor. A design for a Way Determination Unit (WDU) that reduces the D-cache power consumption by allow the cache controller to only access one cache way for a load/store operation was accessible dropping the number of way accesses [8]. In [9], [10] Caches consume a significant amount of energy in modern microprocessors. To design an energy-efficient microprocessor, it is imperative to optimize cache energy consumption. This text examine routine and power trade-offs in cache designs and the efficiency of force reduction for numerous novel cache design techniques embattled for low power. The cache contact time is calculated based on an diagnostic form that 0.8 mm CMOS technology is assumed. The on chip third level cache is 3MB and is designed to provide the low latency and the large size needed by salable and procedural applications.

III. PROPOSED TECHNIQUE

A. LSQ TAGARRAY AND LSQ TLB

To ignore the data conflict 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 andLSQ TLB: lookup and update. Each time a recollection address the LSQ, the LSQ tag arrays and LSQ TLB will be searched for the premature destination way. In case of a hit, the early destination way will be available; If not, the instruction will cause either an early tag miss (if the access to the LSQ tag arrays encounters a miss) or an early TLB miss (if the address is not in the LSQ TLB). For inform operations, the contents of LSQ tag arrays and LSQ TLB is efficient with the tag arrays and TLB of the L1 cache, so that they are identical to avoid cache consistency harms. The update logic of LSQ tag arrays and LSQ TLB is the equivalent as that of the tag arrays and TLB of the L1 cache.

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* substitute to occur at the same time. Consequently, there force is 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. Write/read conflicts occur when the lookup and update operations purpose the same location 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 is a generated by the way enable signal from the cache controller for cache replacements. Regard as a two-way set-associative cache for example. Assume that there is a replacement occurring at the way 1 of the L1 data cache. While a result, the way enabling signal is position 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 activate circuit will not block the lookup operation on this entry. Otherwise the lookup-disable signal will be "0," and the activate circuit will block possible lookup operations to avoid write/read conflicts. The lookup operation, if any in this case, is considered as a miss. This miss might cause some performance degradations if it turns out to be a cache hit in the cache access stage. We observed from simulations less than 0.01% of the total LSQ accesses experienced this issue. Note that if the way enabling signal is "0," i.e., no update operation, the lookup operation will not be affected. This scheme is also used in the LSQ TLB. Since the activating circuit introduces no performance consequence, the enabling circuits increase the critical path of the LSQ tag arrays and LSQ TLB by the delay of an NAND gate. This delay is slight as compared with the critical path of the L1 data cache.

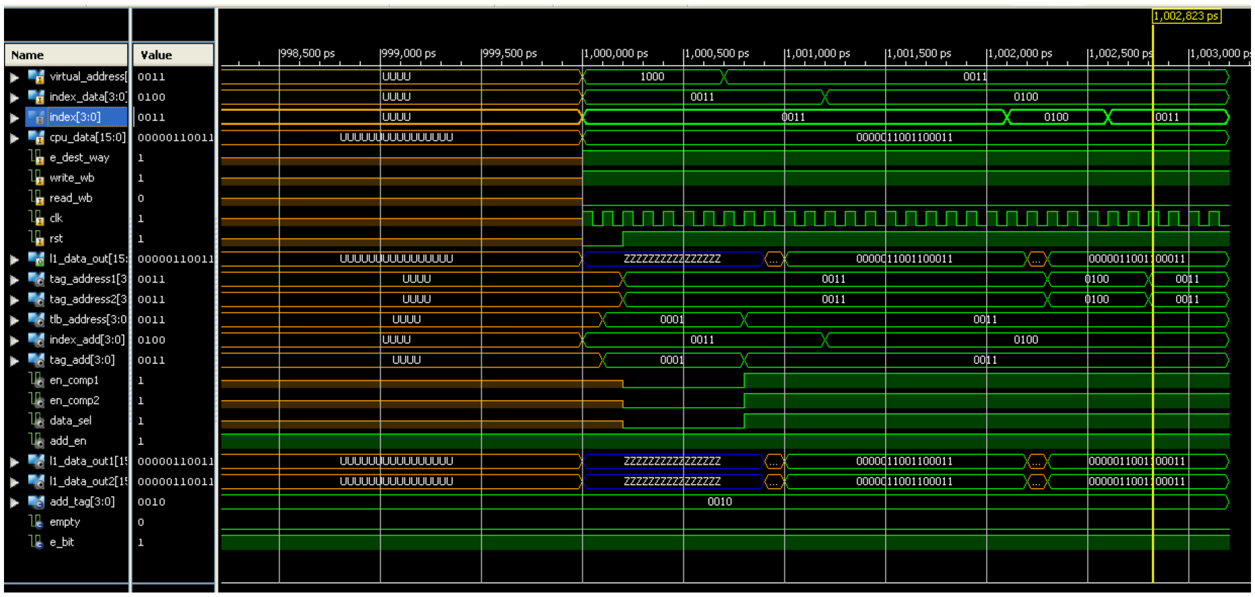


In the above block diagram index data is given as input to the information buffer and its values are passed to multiplexer and TLB.Then, TLB split into two tag arrays (one for odd priority and another one for even priority) with help of comparator. When hit/miss occurs in way decoder, it can be controlled by cache controller. If hit/miss '0', the value will go to the first data array otherwise it will go to the second data array. The communication protocol is used for transferring data's and TLB is used for data address and RS232.The way decoder is used for the fast access and early destination way. If value is set to'1' ETA will be enabled or else disabled.TLB and Index address both are equal, it reduces the latency.

B. INPUT PARAMETERS

INPUT	VALUES
dest_way	1
write	1
read	0
Clk	1
Reset	0

IV. SIMULATION RESULT



V. CONCLUSION

In this paper we proposed a new energy-efficient cache design technique for low-power embedded processors. The proposed technique predicts the destination way of a memory instruction at the early LSQ stage. Thus, only one way needed to be access during the cache access stage if the prediction is correct, thereby reducing the energy consumption significantly. By apply the idea of phased access to the memory instructions whose early destination ways cannot be determine at the LSQ stage, the energy consumption can be further reduced with insignificant performance degradation. While 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 error workload.

REFERENCES

1) Ananda Vardhan and Y N Srikant," **Exploiting Critical Data Regions to Reduce Data Cache Energy Consumption",** Indian Institute of Science, in Copyright 2014
2) Jianwei Dai and Lei Wang, "**An Energy-Efficient L2 Cache Architecture Using Way Tag Information Under Write-Through Policy**", IEEE transactions on very large scale integration (VLSI) systems, vol. 21, no. 1, January 2013
3) Yang Xinfeng and Guo Yongli LiuKechen, "**Research of Low Power Design Techniques for Microprocessor**", 20 II International Conferences on Computer Science and Network Technology Department of Computer Science and Technology
4) Rui Min, Wenben Jone and Yiming Hu, "**An efficient Low Power Cache System**", University of Cincinnati, ISCAS,2004
5) Hugh McIntyre, K. James Lin, P. Kaushik, Suresh Seshadri,Alfred Wang, V. Sundararaman, Ping Wang, Song Kim, Wen-Jay Hsu, Hee-Choul Park, Gideon Levinsky, Jiejun Lu, M. Chirania, Raymond Heald and Sanjaya Dharmasena , "**A 4-MB On-Chip L2 Cache for a 90-nm 1.6-GHz 64-bit Microprocessor"**, IEEE Journal Of Solid-State Circuits, Vol. 40, No. 1, January 2005
6) Dan Nicolaescu, Alex Veidenbaum and Alex Nicolau, "**Reducing Data Cache Energy Consumption via Cached Load/Store Queue",** Copyright 2003 ACM 158113682X/03/0008, University of California, August 25–27, 2003
7) Michael **D.** Powellr, Amit Agarwalr, T. N. Vijaykumarr, Babak Falsafit, and Kaushik Royr, " **Reducing Associative Cache Energy via Way-Prediction and Selective Direct Mapping",**IEEE 2001
8) Dan Nicolaescu Alex Veidenbaum and Alex Nicolau, " **Reducing Power Consumption for High-Associativity Data Caches in Embedded Processors",** Dept. of Information and Computer Science Proceedings of the Design,Automation and Test in Europe Conference and Exhibition (DATE'03)1530-1591/03 © 2003 IEEE
9) Ching-Long Su and Alvin M. Despain, "**Cache Design Trade-offs for Power and Performance Optimization",** International Conferences on Computer Science and Network Technology of Department of Computer Science and Technology
10) Terry Lyon and Eric Delano, " **Data Cache Design Considerations for the Itanium**® 2 **Processor",** Proceedings of the 2002 IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD'02) © 2002 IEEE