Divide-and-Conquer Way Access for Low Power Mobile Caches

Satish Raghunath, Lakshmi Deepika Bobbala, Naveen Davanam and Byeong Kil Lee

***Abstract*—As multi-core design concept is becoming dominant, power consumption of the shared level-2 caches is one of the critical issues along with its performance. This is more significant for mobile processors which are used in battery-powered devices. Designing a cache memory, increasing the cache size or adding more set-associativity is one of the simplest ways to improve the performance for both mobile processor and even general-purpose processors. In mobile processors, however, simple increase of the cache size can significantly affect the chip area and power consumption. Modern integration technologies allow integrating the bigger caches into tiny single-chip multi-core mobile processors, but relevant architectural issues cannot be ignored. These issues will be worse in mobile processors which are integrated in a small form-factor without any cooling fan. In this paper, we propose a novel cache mechanism to reduce power consumption in level-2 cache by using the divide-and-conquer way-accessing technique. This idea is derived from the observation that cache way-hit distribution shows the unbalanced pattern in which primary-way(s) have more way-hits. Based on this way utilizations and activities, we came up with a biased way-access mechanism to get benefits from lower associativity which requires relatively lower power consumption.**

Considering unbalanced way-hit distribution of the cache, all other ways but primary way(s) can be accessed one step later when the primary way-access has across the miss. By doing that, if way-hits are achieved from the primary-way, power can be saved by turning only primary-way on. However, the penalties from the primary-way misses will not be ignored in access time. Through the careful tradeoff analysis between power-saving and penalty in access time, appropriate cache configurations can be chosen in early design stage.

***Index Terms*—cache memory, mobile processors, set associativity, low power design.**

I. Introduction

Multi-core (many-core) processor paradigm is one of main trends in microprocessor design. In addition to that, NoC (Network-on-Chip) and SoC (System-on-Chip) design concepts make it accelerating towards performance improvement in computing society. While processors with multi-cores have already thrived in general-purpose processor area, mobile processor companies are recently starting to release their multi-core version (Intel Atom [11], ARM Cortex A9 [9], etc.) which are used in netbooks, smart phones or tablet PCs. Even though it provides more IPC (Instruction per Cycle) and ILP (Instruction Level Parallelism) to the mobile system, many design issues are being presented as in general-purpose processors or are more critical. Especially, power and heating problems are among the most critical issues in battery-powered mobile devices. Since the performance improvement can be deteriorated without addressing power issues in applying multi-core technologies, those problems need to be analyzed and solved with appropriate solutions for future mobile processor designs.

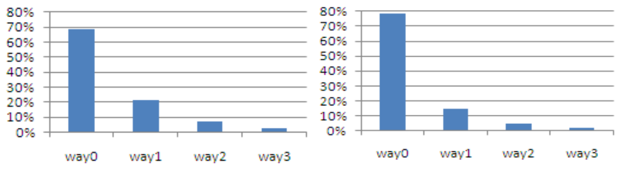
In mobile processor market, many companies are trying to take a leading position in the MID (Mobile Internet Devices) race. From the microprocessor architecture point of view, two directions can be observed: x86-based technology and ARM IP-based technology. Both groups have pros and cons in terms of operating system, performance, power consumption, processor size, area, backward compatibility, etc. Nobody knows which R&D group will be eventually leading in the MID race, but it is possible that they will meet at somewhere between those two development directions. Recently released Apple's iPad tablet PC [12] is one of the intermediate concepts between two groups. Along with this technical and market trend, mobile internet device (MID) is becoming one of major categories in embedded domain (smart phone) and even in general-purpose processor domain (netbook). Therefore, mobile processor designers need to explore wide spectrum of workloads – not only embedded workloads but also general-purpose applications (eg., SPEC CPU benchmark suite [10]).

As multi-core trends are becoming dominant, cache structures are being sophisticated and complicated, and bigger shared level-2 (L2) caches are demanded for higher cache performance. Generally, memory subsystem takes a large portion of the die area in the microprocessors, and caches consume over 40% of a processor's total power [3][7]. Therefore, reduction of energy consumption - especially in cache - is one of the main design goals for mobile computing devices.

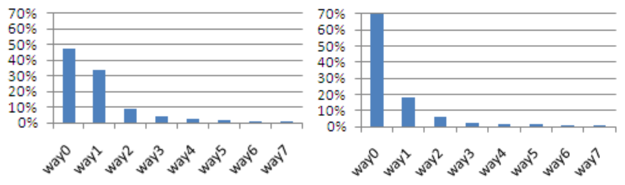
Designing a cache memory, increasing the cache size is one of the simplest ways to improve the performance for both mobile processor and even general-purpose processors. In mobile processors, however, simple increase of the cache size significantly affects chip area and power. Modern integration technologies allow integrating the bigger caches into tiny single-chip multi-core mobile processors, but relevant architectural problems cannot be ignored. These issues will be a bigger issues in mobile processors which are integrated in a small form-factor without any cooling fan (unlike general-purpose processors).

Manuscript received October 30, 2010.
B. Lee, is with the Department of Electrical and Computer Engineering, The University of Texas at San Antonio, San Antonio, Texas 78249 USA. Phone: +1 210 4585027; Fax: +1 210 4585947; e-mail: byeong.lee@utsa.edu.

In this paper, we propose a novel cache mechanism to reduce power consumption in level-2 cache by using divide-and-conquer way-accessing technique. As shown in Figure 1, way-hit distribution is not a balance form on all ways, but it is biased toward lower order of ways. For example, with 256KB 4-way set associative cache, almost 70% of way-hits are concentrated on way-0. With bigger cache size, the level of way-0 biasing is becoming more eminent as shown in Figure 1. Similar patterns can be observed in different size and different associative caches. Based on this way utilizations and activities, we came up with a biased way-access mechanism to get benefits from lower associativity (e.g., direct-map cache, 2-way cache, etc.) which requires relatively lower power consumption. We use the SPEC CPU2000 (Integer and Floating-point) benchmark suite [10] for simulation, since current mobile internet devices are required to run same applications which have been used in general-purpose processor.



(a) Way-hit distribution for 4-way set associative (left side: 256KB; right side: 512KB)



(b) Way-hit distribution for 8-way set associative (left side: 256KB; right side: 512KB)

Figure 1. Cache way-hit distribution

Considering unbalanced way-hit distribution of the cache, all other ways but primary way(s) can be accessed one step later when the primary way-access has across the miss. By doing that, if the way-hits are achieved from the primary-way, dynamic power can be saved by turning only primary-way on. However, the penalties from the primary-way misses will not be ignored in access time, because the decision of way hit/miss from the primary- way(s) will be required for both cases: way-hits and way-misses. This disadvantage is also analyzed in this paper. Based on the tradeoff analysis between power saving and penalty in access time, appropriate cache configurations can be chosen in early design stage. By applying divide-and-conquer way-access mechanism to level-2 cache, power consumption of cache memory can be reduced with the penalty in cache access time. The amount of power reduction and the level of penalty in access time will be varied depending on the set-associativity and cache size.

The rest of the paper is organized as follows: section 2 describes related works. The proposed divide-and-conquer way-access mechanism is explained in section 3. Section 4 shows simulation environment, results and analysis. Finally, concluding remarks and future works are presented in the last section.

II. Related Work

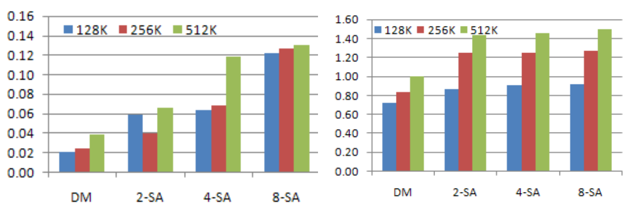
Several approaches [2-6][8][20] have been proposed to achieve low power consumption a set associative cache. While some researches focus on heterogeneous way-size techniques [14], other approaches [15-18] are based on predicting the way where the data is stored. Abella *et al.* [14] proposed a Heterogeneous Way-size Cache, in which different cache ways have different sizes. They applied this mechanism to L1 and L2 caches with dynamically adaptive version. Powell *et al.* [21] studied cache energy reduction mechanism using way-predicting and selective direct-mapping. Ting *et al.* [19] applied cycle-time-aware sequential way-access to set-associative cache.

III. Divide-and-Conquer Way Access for Low Power Consumption

Low power design strategies are very important in battery-powered mobile processors, and power estimation in early-stage is one of significant steps in processor design. As mobile internet devices (MID) are widely used and multi-core concepts are being merged into tiny mobile processors, the power reduction technique of shared level-2 cache is becoming more critical. Our research is focused on reducing dynamic power consumption on set-associative level-2 caches in mobile processors. The size of level-2 cache in mobile devices are ranging from 64KB to 8MB (up to 2MB for ARM Cortex-A8; up to 8MB for ARM Cortex-A9) [9], but commercially available L2 cache size is 256KB or 512KB.

A. *Analytical approach on cache access time and power consumption*

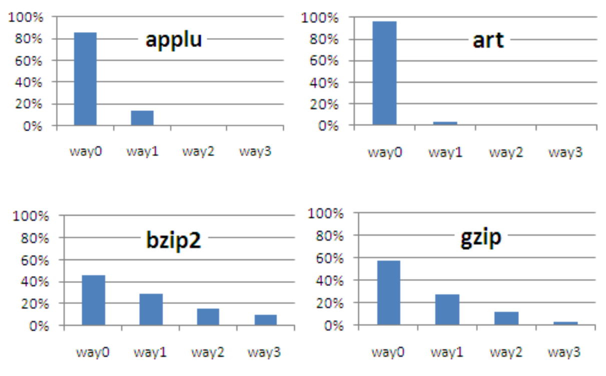
Figure 2 shows access time and energy consumption for various cache configurations from the experiment with Cacti [7] which is an integrated cache and memory access time, cycle time, area, leakage and dynamic power model. Performance behaviors are totally different with the cache size and set-associativity. Energy consumption also shows irregular pattern. For example, similar access time can be observed between 2-way set associative (SA) and 4-way SA in 128K cache. Also, similar behavior can be seen in 512K cache between 4-way SA and 8-way SA. However, designer should be careful to select set associativity in 256K cache since there will be a big difference among set associativity.



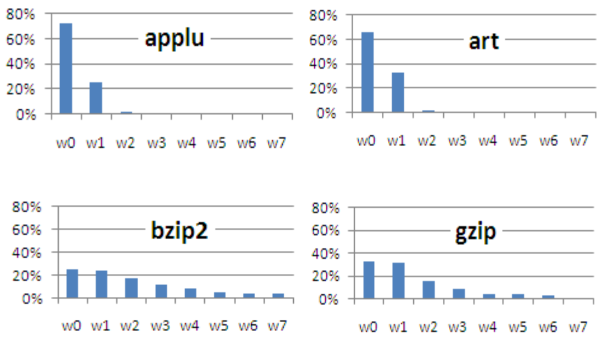
(a) Access time [ns] (b) Energy consumption [nJ]

Figure 2. Cache access time and energy consumption for various cache configurations

Similarly, energy consumption behaviors also show both regularity and irregularity. All three cache sizes show a big jump from direct-map cache to 2-way SA cache, and small difference can be monitored among 2-way, 4-way and 8-way set associative caches. Main idea of this research is to take advantage of lower energy consumption in direct-map cache.



(a) 4-way



(b) 8-way

Figure 3. Way-hit distribution of 4-way and 8-way set associative caches (256K cache size)

B. Way-hit distributions

In mobile processor design, cache size is one of the critical factors due to its impact to processor size (area) and power consumption. Even with those limitations, many mobile processor companies tend to apply bigger caches to improve the overall performance by doubling cache size. With the cache size increase, the increased set-associativity is also applied to maximize the overall cache performance. In this case, energy consumption might be an issue based on the discussion in section 3.1. Also, this will be a critical issue in battery-operated mobile devices.

As shown in Figure 3, it can be observed some patterns on way-hit distribution in dynamic simulation. While some applications including applu and art show that more than 80% of the way-hit is concentrated on way-0 in 4-way SA 256K cache, some other applications such as bzip2 and gzip show relatively spread and exponentially decreasing pattern from low-order way to high-order way. With 8-way SA and same size cache, the concentration on way-0 tends to spread all the way to the way-7. However, the behavior keeps the intrinsic features as in 4-way SA in terms of distribution pattern. Based on this observation, we proposed a way-access mechanism to save the energy consumption by using this unbiased way-hit distribution.

C. Divide-and-Conquer way access

In general, secondary cache (level-2 cache) should focus on reducing miss rate to reduce the penalty of long main memory access times with larger block sizes and higher levels of associativity, while primary cache (level-1 cache) should focus on minimizing hit time in support of a shorter clock cycle and smaller block sizes. The proposed divide-and conquer way-access scheme is designed for level-2 caches that handle missed instruction or data addresses from level-1 caches. The miss rate of the proposed scheme will be same as traditional cache.

1) Basic configuration

Several different approaches for divide-and conquer way-access can be considered to save dynamic power consumption from high associative cache configuration. As shown in Figure 4, the traditional set associative cache will access all ways in parallel (one time per one cache access), but only one way will be validly used. It does not quite affect access time because of simultaneous accesses to each ways, but these multiple accesses cause a waste of power consumption. All accesses to other ways, however, are required to decode set-associativity with a penalty of large amount of energy consumption.

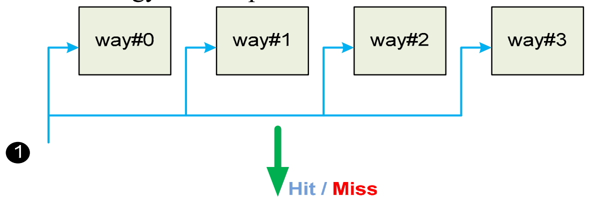
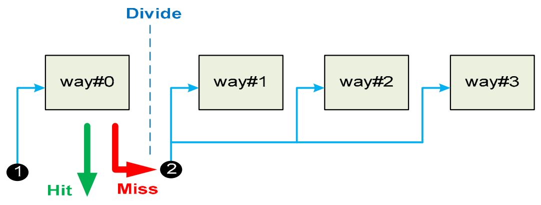


Figure 4. Traditional 4-way set associative cache structure

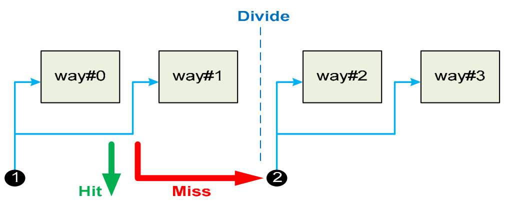
Considering unbiased way-hit distribution as shown in Figure 2, way-accesses scheme can be divided into two steps: (1) by accessing primary way(s), if the primary-way gets the hit, the remaining ways do not need to be checked. In this case, we only need the amount of energy consumption as in direct-mapped cache, and multiplexing way-data is not needed anymore (so, we can get a little bit of advantage in access time); (2) in the case of misses, the remaining ways will be accessed to check the validity. However, the penalties cannot be ignored in terms of access time and energy consumption because the access time and power consumption to decide the hit/miss from the primary-way(s) are required in all cases of way hit/miss.

2) Division schemes

Several different division schemes can be used for the proposed divide-and conquer way-access configuration. In the case of 4-way set associative cache, two divide-and-conquer schemes can be applied: (i) 1+3 (way-0 as primary-way); (ii) 2+2 (way-0 and way-1 as primary-ways) as shown in Figure 5.



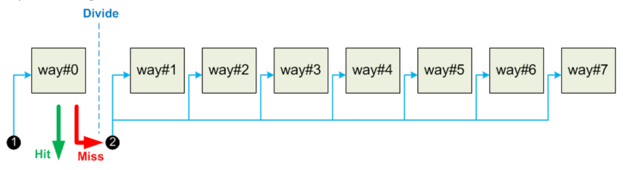
(a) **1+3** divide-and-conquer way access *(way-0 will be primary way in 4-way set associative cache structure)*



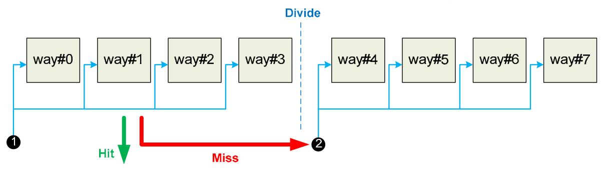
(b) 2+2 divide-and-conquer way access *(way-0 and way-1 will be primary ways in 4-way set associative cache structure)*

Figure 5. Divide-and-conquer way access in 4-way set associative cache structure (1+3 and 2+2)

In the case of 8-way set associative cache, various divide-and-conquer schemes can be applied: (i) 1+7 (way-0 as primary way); (ii) 2+6 (way-0 and way-1 as primary ways)); (iii) 3+5 (way-0, way-1 and way-2 as primary ways)); (iv) 4+4 (way-0, way-1, way-2 and way-3 as primary ways). Figure 6 shows 1+7 and 4+4 schemes.



(a) 1+7 divide-and-conquer way access *(way-0 will be primary way in 8-way set associative cache structure)*



(b) 4+4 divide-and-conquer way access *(way-0 to way-3 will be primary ways in 8-way set associative cache structure)*

Figure 6. Divide-and-conquer way access in 8-way set associative cache structure (1+7 and 4+4)

IV. Simulation and Analysis

A. Workloads and experimental environment

For the simulation, we used the SPEC CPU 2000 integer and floating-point suite [10] as the workloads and we also used the sim-cache of the Simplescalar [1] with skipping the initialization phase of the workload. First 500 million instructions were fast forwarded and following 500 million instructions are simulated with the ref input data sets. The reason we choose the SPEC benchmark for simulating the mobile processors is that general-purpose applications are becoming a major workloads in mobile internet devices (MID) along with increased processing capability from a single-chip multi-core paradigm. For power estimation, we used the Cacti and modified the Wattch [22].

TABLE 1. Benchmark Characteristics for Baseline L2-Cache *(Cache miss rate, %)*

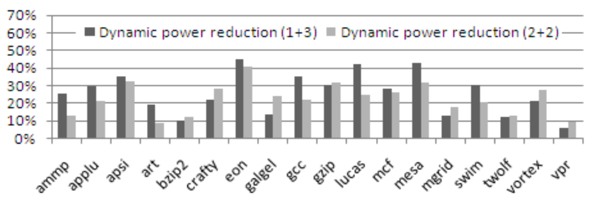
Bench-marks	256K		512K		1024K	
	4-way	8-way	4-way	8-way	4-way	8-way
ammp	48.8%	45.1%	48.9%	45.6%	45.1%	32.3%
applu	34.8%	34.8%	34.8%	34.8%	34.8%	34.8%
apsi	7.9%	7.6%	8.0%	7.6%	7.6%	7.6%
art	56.6%	56.3%	56.6%	56.6%	56.3%	49.5%
bzip2	33.4%	20.7%	32.8%	20.5%	20.7%	11.2%
crafty	4.7%	2.1%	3.7%	1.9%	2.1%	0.6%
galgel	23.5%	22.8%	23.7%	22.9%	22.8%	10.5%
gcc	32.1%	8.2%	32.1%	8.2%	8.2%	7.3%
gzip	2.9%	1.3%	2.7%	1.3%	1.3%	1.2%
lucas	28.6%	28.6%	28.6%	28.6%	28.6%	28.6%
mcf	21.5%	20.7%	21.3%	20.7%	20.7%	19.9%
mesa	8.8%	7.4%	8.0%	7.4%	7.4%	7.4%
mgrid	39.0%	38.1%	38.6%	37.3%	38.1%	24.9%
swim	29.9%	29.9%	29.9%	29.9%	29.9%	29.9%
twolf	36.9%	25.1%	36.5%	24.5%	25.1%	10.4%
vortex	7.1%	4.2%	5.9%	3.9%	4.2%	2.6%
vpr	36.8%	23.9%	35.3%	22.9%	23.9%	7.9%

Table 1 shows the benchmarks we used and L2 cache miss rates (4 and 8-way; 256KB, 512KB and 1024KB). The configurations of level-1 cache for the simulations are 32KB, 64B block size and 2-way with LRU replacement for L1 I-cache and D-cache. The unified L2 cache configuration is 128B block size with LRU replacement. We choose popular cache configurations based on a mobile processor - ARM Cortex A8/A9 configurations. While some applications such as crafty, gcc, gzip, mesa, mgrid, vortex, bzip2 and twolf show sensitively responding to the increase of cache size, other applications show little sensitivity to cache-size increase. Also, most applications including bzip2, crafty, galgel, gcc, gzip, twolf, vortex and vpr show very strong response to set-associativity increase. However, the penalty to improve hardware configuration cannot be ignored in terms of power consumption and relevant chip area.

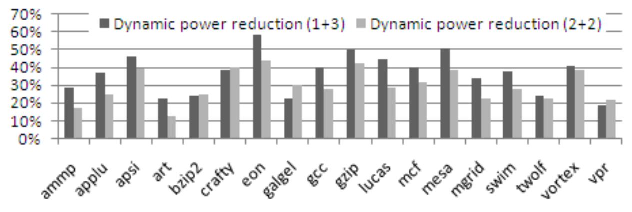
B. Results and analysis

Figure 7 shows the dynamic power reduction with two different divide-and-conquer way-access schemes. Each application shows different behaviors with cache size, set-associativity and way-access scheme. In the case of 256K 4-way set associative cache, 1+3 scheme shows better power reduction on ~60% of workloads than 2+2 scheme, but with 512K size cache, ~85% of workloads show better power reduction with 1+3 scheme. This is because the level of primary-way biasing is becoming more eminent with the bigger cache size as shown in Figure 1. If 8-way set-associativity is applied to 512K cache, the amount of primary-way hit is spread into the other ways but the way-hit biasing is still dominant at low-order of way.

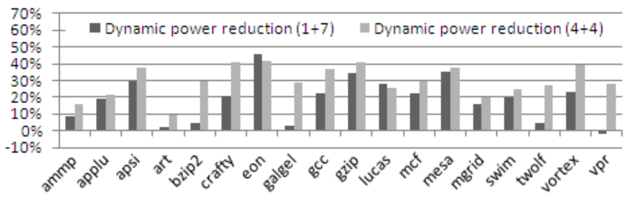
As shown in Figure 7, the scheme of divide-and-conquer way-access shows power reduction in most of applications with various configurations. However, it is interesting to note that some applications such as gzip, mgrid and vortex show the reversed behavior in 1+3 and 2+2 between 256K and 512K with same 4-way SA cache. The reason is that these applications have well-balanced way-hit distribution. Also, increasing cache size and associativity (1024K cache size with 8-way SA) results in different behaviors even with primary-way hit biasing, because its power consumption is exponentially increased with cache and associativity increase.



(a) 256K 4-way



(b) 512K 4-way

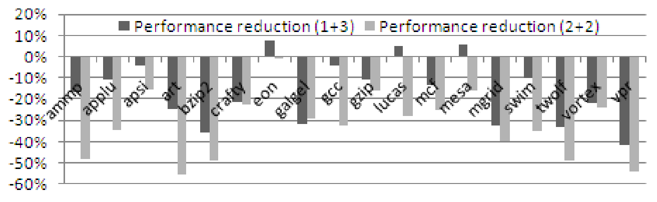


(c) 1024K 8-way

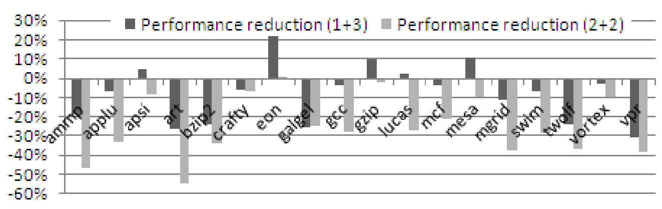
Figure 7. Dynamic power reduction with different cache sizes, set-associativities and way-access schemes

Based on the preliminary analysis, it is expected that the proposed divide-and-conquer way-access scheme will have bad impact on access time due to the required penalty from the misses on primary-way accesses. As shown in Figure 8, most of applications except eon, lucas and mesa show the performance degradation (in access time) with the proposed scheme in 256K 4-way cache configuration. However, it can be observed that the access time can be improved with the increased cache size and set-associativity. Figure 8 shows that the amount of performance reduction tends to be reduced, and especially in the case of 1024K 8-way, most applications except ammp, art, bzip2, galgel, twolf and vpr show positive improvement with 1+7 way-access scheme.

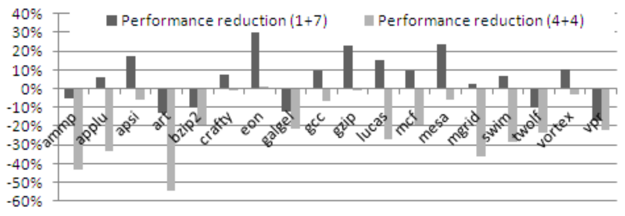
Most of modern mobile processors have intelligent power management mechanisms. For example, when the level of battery-power is low, the processor can be operated with lower performance. That way, users can operate the mobile devices for longer time. By applying the proposed scheme with careful tradeoff analysis for specific mobile application group, mobile processors can have more battery-operating time with reasonably degraded performance.



(a) 256K 4-way



(b) 512K 4-way



(c) 1024K 8-way

Figure 8. Cache performance reduction with different cache sizes, set-associativities and way-access schemes

V. Conclusion

Designing a cache memory, increasing the cache size or adding more set-associativity is one of the simplest ways to improve the cache performance for both mobile processor and even general-purpose processors. In mobile processors, however, simple increase of the cache size can significantly affect the chip area and power consumption. Modern integration technologies allow integrating the bigger caches into tiny single-chip multi-core mobile processors, but relevant architectural problems cannot be ignored. These issues will be worse in mobile processors which are integrated in a small form-factor without any cooling fan (unlike general-purpose processors). In this paper, we propose a novel cache mechanism to reduce power consumption in level-2 cache by using the divide-and-conquer way-accessing technique. This idea is from the observation that cache way-hit distribution shows the unbalanced pattern in which primary-way(s) have more way-hits. Based on this way utilizations and activities, we came up with a biased way-access mechanism to get benefits from lower associativity (e.g., direct-map cache, etc.) which requires relatively lower power consumption.

Considering unbalanced way-hit distribution of the cache, all other ways but primary way(s) can be accessed one step later when the primary way-access has across the miss. By doing that, if the way-hits are achieved from the primary-way, power can be saved by turning only primary way on. However, the penalties from the primary way misses will not be ignored in access time, because the decision of way hit/miss from the primary way(s) will be required for both the way-hit and the way-miss. Through the careful tradeoff analysis between power-saving and penalty in access time, appropriate cache configurations can be chosen in early design stage.

As future works, we will work on more efficient hardware mechanism for applying divide-and-conquer, effective power managing scheme and appropriate replacement scheme. We will also study on the tradeoff analysis for performance, power and temperature on the proposed cache mechanism.

References

[1] D. C. Burger and Todd M. Austin, “The Simplescalar Tool Set, Version 2.0,” UW Madison Computer Sciences Technical Report #1342, June 1997.

[2] B. Calder, D. G, and J. Elmer, “Predictive sequential associative cache,” IEEE International Symposium on High Performance Computer Architecture, pp. 244–253, 1996.

[3] Z. Chishti, M. D. Powell, and T. N. Vijaykumar, “Distance associativity for high-performance energy-efficient non-uniform cache

architectures," In Proceedings of the 36th Annual ACM/IEEE International Symposium on Microarchitecture, pp. 55–66, 2003.

[4] E. G. Hallnor and S. K. Reinhardt, "A fully associative software managed cache design," In Proceedings of the 27th Annual International Symposium on Computer Architecture, pp.107–116, 2000.

[5] N. P. Jouppi, "Improving direct-mapped cache performance by the addition of a small fully-associative cache and prefetch buffers.," In Proceedings of the 17th Annual International Symposium on Computer Architecture, pp. 364–373, 1990.

[6] A. Seznec, "A case for two-way skewed-associative caches," In Proceedings of the 20th Annual International Symposium on Computer Architecture, pp. 169–178, 1993.

[7] S. E.Wilton and N. Jouppi, "Cacti: an enhanced cache access and cycle time model," IEEE Journal of Solid-State Circuits, No. 31 pp. 677–688, May 1996.

[8] M. K. Qureshi, D. Thompson, and Y. N. Patt, "The V-Way Cache: Demand Based Associativity via Global Replacement," International Symposium on Computer Architecture 2005.

[9] ARM Cortex A9 processor, http://www.arm.com/products/processors/cortex-a/cortex-a9.php

[10] Standard Performance Evaluation Corporation (SPEC) website, http://www.spec.org/

[11] Intel Atom Processor: Intel's smallest chip, http://www.intel.com/technology/atom/

[12] Apple iPad tablet PC, http://www.apple.com/ipad/

[13] EEMBC, EDN Embedded Microprocessor Benchmark Consortium, http://www.eembc.org/

[14] J. Abella, A. González, "Heterogeneous way-size cache," International Conference on Supercomputing, 2006

[15] D. H. Albonesi, "Selective Cache Ways: On-Demand Cache Resource Allocation," MICRO 1999.

[16] C. Liu , A. Sivasubramaniam, M. Kandemir, "Organizing the Last Line of Defense before Hitting the Memory Wall for CMPs," In the Proceedings of High-Performance Computer Architecture 2004

[17] B. Batson, T. N. Vijaykumar, "Reactive-Associative Caches," PACT 2001.

[18] K. Inoue, T. Ishihara, K. Murakami, "Way-Predictive Set-Associative Cache for High Performance and Low Energy Consumption," ISLPED 1999.

[19] C. Ting, J. Huang and Y. Kao, "Cycle-Time-Aware Sequential Way-Access Set-Associative Cache for Low Energy Consumption," IEEE Asia Pacific Conf. on Circuits and Systems, pp. 854-857, Dec. 2008.

[20] D. Nicolaescu, A. Veidenbaum and A. Nicolau, "Reducing Power Consumption for High-Associativity Data Caches in Embedded Processors," Design, Automation, and Test in Europe, 2003.

[21] M. D. Powell, A. Agarwal, T. N. Vijaykumar, B. Falsafi and K. Roy, "Reducing set-associative cache energy via way-prediction and selective direct-mapping," International Symposium on Microarchitecture 2001.

[22] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: a framework for architectural-level power analysis and optimizations," In ISCA, pages 83–94, 2000.

[23] E. Witchel, S. Larsen, C. S. Ananian, and K. Asanovi, "Direct addressed caches for reduced power consumption," In Proceedings of the 34th Annual International Symposium on Microarchitecture, 2001.

Satish Raghunath received the B.S. degree in Electrical Engineering from University of Mumbai, India 2006. He is currently M.S. student in the University of Texas at San Antonio. His research interest includes computer architecture, workload characterization, workload reduction and GPGPU.

Lakshmi Deepika Bobbala received the B.S. degree in Electrical Engineering from GITAM-Vishakapatnam, India 2006. She is currently M.S. student in the University of Texas at San Antonio. Her research interest includes computer architecture, cache optimization, low power design and workload characterization.

Naveen Davanam received the B.S. degree in Electrical Engineering from SRI VENKATESWARA UNIVERSITY, India 2007. He is currently M.S. student in the University of Texas at San Antonio. His research interest includes computer architecture, multithreading, cache optimization and workload characterization.

Byeong Kil Lee received the Ph.D. degree in computer engineering from the University of Texas at Austin, Austin, in 2005. He is currently an assistant professor in the Electrical and Computer Engineering Department, University of Texas at San Antonio, San Antonio, where he joined the faculty in 2009. He was a senior design engineer in Texas Instruments, Inc., Austin from 2004 to 2009. Also, he was a senior research staff in the Agency for Defense Development (ADD), Korea. His current research interests include computer architecture, application-specific embedded systems (network processor, multimedia processor), low power mobile processors, workload characterization of emerging applications, parallel computing and parallel architecture design, power-aware design, and power estimation.