Reducing Dynamic Energy of Variable Level Cache

Ko Watanabe, Takahiro Sasaki, Tomoyuki Nakabayashi, Kazuhiko Ohno, and Toshio Kondo

***Abstract*—Today, a high-performance and low-power processor is required. Even on an embedded processor, a large cache is implemented to achieve high performance, and leakage energy in such large cache has been increasing caused by scale down of semiconductor device. Therefore, reduction of cache leakage energy is very important. To reduce cache leakage energy, we propose a variable level cache (VLC) which reduces leakage energy with a little performance degradation. Generally, a required size of a cache depends on a program behavior. VLC dynamically estimates whether the current cache size is suitable for a running program, and if not VLC modifies its cache structure and cache hierarchy to change cache capacity. Since changing the cache construction and hierarchy incurs a large overhead, VLC adopts the low-overhead technique which reduces hierarchy changing overhead to prevent performance from degrading. However, previous VLC has a problem that dynamic energy consumption is increased because the technique needs many futile accesses. To solve the problem, this paper proposes the novel technique which reduces the number of the futile accesses to reduce dynamic energy consumption. According to our simulation results, the proposed VLC technique reduces 18% dynamic energy without performance degradation compared with the previous VLC.**

***Index Terms*—Low-power and high-performance cache, variable level cache, leakage energy, dynamic energy.**

I. Introduction

In mobile computer devices using finer CMOS process technology, leakage energy becomes one dominant factor of energy consumption. In particular, the leakage energy consumed by a cache system is one major factor of energy consumption. Therefore, to reduce cache leakage energy without decrease in performance, we have proposed a variable level cache (VLC) [1], [2]. VLC dynamically resizes the cache size according to the required cache size from a running program. In addition, VLC uses remained cache area after resizing as a pseudo lower level cache and an exclusive cache [3]. VLC swaps a cache line in the upper half area for a cache line in the lower half area (pseudo lower level cache) in low energy operation. The swapped lines become inaccessible lines when VLC returns to normal cache size (then these two segregated caches are unified) because the indexes of the swapped lines differ from the indexes before expanding. For this reason, VLC needs to write these lines back when VLC resizes the cache size to large capacity.

To solve this problem, previous VLC introduces a technique to reduce the writeback overhead. However, dynamic energy consumption of VLC increases on a low cache hit rate program because the technique needs extra accesses. We had estimated the energy consumption of VLC with large leakage energy based on a transistor roadmap. Because our previous works assumed pessimistic leakage current model, leakage energy was a dominant factor. However, state-of-the art CMOS device technologies [4], [5] enable us to design a low leakage energy transistor. A ratio of dynamic energy gets a large impact on the energy consumption. Hence, the problem which increases dynamic energy consumption in VLC cannot be ignored. In this paper, we propose a novel technique to reduce the dynamic energy consumption of VLC.

According to our simulation results, proposed technique reduces 18% dynamic energy without performance degradation compared to the previous VLC.

Manuscript received March 6, 2013; revised June 17, 2013.

The authors are with the Graduate School of Engineering Mie University, Tsu, Mie 514-8507, Japan (e-mail:{watanabe, sasaki, tomoyuki, ohno, kondo}@arch.info.mie-u.ac.jp).

II. Related Works

There are many kinds of approaches [6]-[9] which reduce cache leakage energy. However, we show only two approaches, Ke Meng's Approach [10] and DRI cache [11], because of the limitation of the page.

A. Ke Meng's Approach

This approach [10] dynamically increases/reduces the number of the associativity to reduce leakage energy. In addition, its cache tag is logically organized with MRU order in every access. We do not compare our proposed approach with this approach, because it needs cycles per instruction (CPI) and our trace-driven simulator cannot estimate CPI. Therefore, we will compare to this approach in the future.

B. DRI Cache

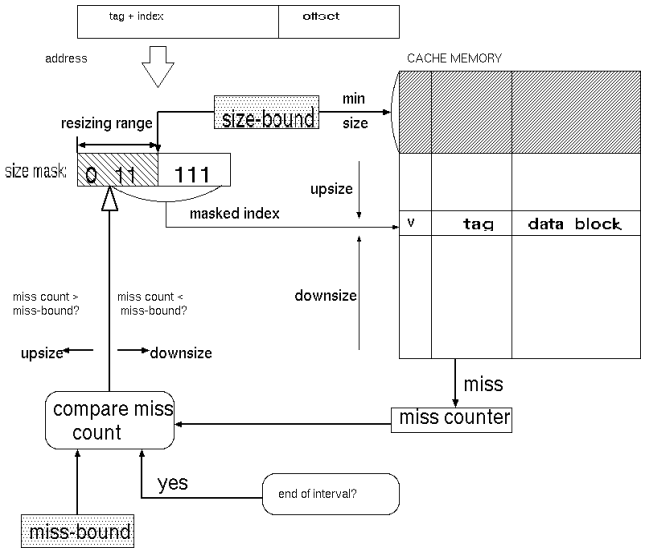


Fig. 1. Block diagram of DRI.

DOI: 10.7763/IJCEE.2013.V5.777

DRI cache [11] is a representative for cache leakage energy reduction techniques, and the base approach of our proposed cache. Fig. 1 shows a block diagram of DRI cache. DRI cache counts the number of cache misses in a fixed length interval with a miss counter. At the end of each interval, the cache system expands/reduces the cache capacity according to whether the miss counter value is lower/higher than the statically defined threshold. Resizing the cache capacity requires dynamic modification of index bit-width depending on the cache size. Every time the cache size reduction is required, the size mask shown in Fig. 1 is shifted to right in order to use the smaller number of index bit-width, and vice versa. DRI cache turns off the power supply for unused memory cell. DRI cache can be implemented on general SRAM modules because DRI cache resizes cache with a memory bank.

C. Problem of DRI Cache

Because DRI cache is specialized only for instruction cache, performance degradation becomes a problem when DRI cache is applied to data cache. Generally, before turning off the voltage for an unused cache bank, data in the bank must be written back to a lower level storage not to lost modified data (when DRI cache is used in instruction cache, this problem does not matter because no data is modified in instruction cache). Moreover, when DRI cache expands the cache capacity, a data that exists in the cache needs to be written back to lower level storage because a correct location for a data depends on the cache size.

III. Variable Level Cache

To solve the problem of DRI cache, we have proposed a VLC that improves DRI cache by reducing the penalty to write data back on resizing cache capacity [1], [11]. Our approach enables DRI technique to apply not only to a dedicated instruction cache but also to a data cache including unified L2 and/or L3 caches.

Fig. 2 shows basic idea of VLC. Similar to DRI cache, VLC predicts required cache capacity and resizes the cache size. Unlike the original DRI cache, if cache size is halved, the lower half cache area becomes stand-by (sleep) mode in which the power supply is lowered to the minimum voltage to maintain the memory content. However, if simply downsizing the cache size, performance degradation becomes a critical problem. Therefore, to keep performance, VLC uses the sleeping area as a lower level exclusive cache. In this paper, Mode 1 denotes a state of VLC which has one level (general cache structure), Mode 2 denotes the cache has two levels (L2 and L3), and Mode 3 denotes the cache has three levels (L2, L3 and L4) as shown in Fig. 2.

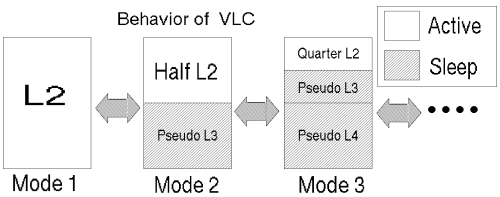


Fig. 2. Behavior of VLC.

A. Detail of Variable Level Cache

In this subsection, we describe mechanism of VLC. We assume that VLC starts operation in Mode 1 (operates as a normal cache) when a program starts.

A processor executes a program in Mode 1, and we assume that the program does not need a large cache capacity. VLC dynamically detects it, and VLC shifts into Mode 2 by putting the lower half memory into sleep mode. After VLC switches mode from Mode 1 to Mode 2, VLC accesses to the upper half active L2 at first, and then if this access makes a miss, the Pseudo-L3 (the lower half sleeping L2) is woken up and is accessed. If desired line exists in the Pseudo-L3, the access becomes a hit and VLC swaps the hit line for the corresponding line in the upper half cache memory. Pseudo-L3 operates as an exclusive cache to keep performance when the L2 cache size is halved. If access misses occur in both L2 and pseudo-L3, the access goes to an actual lower memory hierarchy (e.g., L3 cache or main memory).

Fig. 3 shows a block diagram of VLC to implement the above mechanism. The VLC is constructed with the circuit of DRI cache and additional two units: re-access unit and victim buffer. Both units are used in the modes except for Mode 1. We describe the operation of these units in Mode 2.

The re-access unit is used to wake up and access the pseudo L3 cache if the current access generates a miss and pseudo lower cache still exists. To use pseudo-L3 as an exclusive cache, the victim buffer is used to write a replaced data in the upper half area to the lower half area (Pseudo-L3). The replaced line is selected by LRU. In Mode 2, the index of VLC is masked to make the first access refer to the upper half non-sleep area. If a cache miss happens on the upper half L2, a replace data moves to the victim buffer (in Fig. 3) to write back to pseud-L3, and this load or store instruction accesses to pseudo-L3 by re-access unit (in Fig. 3). At that time, the pseudo-L3 area is woken up to be accessed. Therefore, one access to the pseudo-L3 takes the sum of wakeup latency and L2 access latency. If the access to pseudo-L3 makes a hit, the hit data moves to the upper half L2 area. If the access makes a miss, VLC accesses to main memory. The data in the buffer is written back to the lower half area in background after the end of the access.

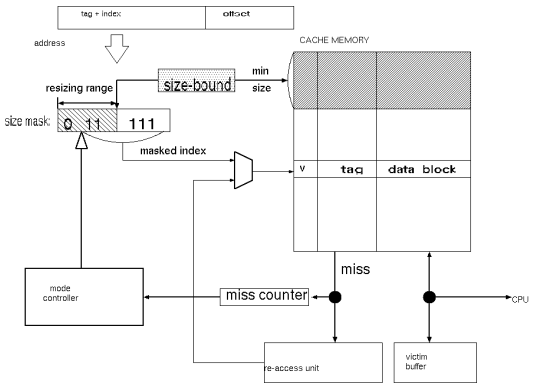


Fig. 3. Block diagram of VLC.

Advantage of VLC compared with DRI cache is that VLC does not need to write data in the lower half area back when VLC reduces the cache capacity. However, when shifting

from Mode 2 to Mode 1, basically VLC must write all data in the cache back because swapped lines are stored in inconsistent address. In our approach, in order to keep performance, the cache contents are not written back even if the mode shift occurs by introducing the trick described in section 4. Using the trick incurs incorrect data placement called violated set line after shifting to a mode which has fewer levels (e.g., shifting from Mode 2 to Mode 1).

B. Violated Set Line

Violated set line means a line which exists in incorrect position in Mode 1. Fig. 4 shows an example of a state after

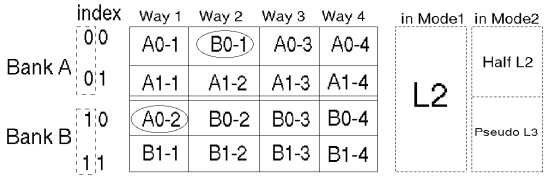


Fig. 4. Example of Violated Set Lines

VLC changes mode from Mode 2 to Mode 1 on 4-way cache. The boxes labeled 'A' mean those cache lines must exist in Bank A in Mode 1, and the succeeding number means index for each bank. In Mode 1, VLC constructs a L2 cache of both Bank A and Bank B. In Mode 2, Bank A becomes L2 cache, and Bank B becomes pseudo-L3 cache. Therefore, A0-2 can exist in the first row and third row in Mode 2. However in Mode 1, the line A0-2 in Fig. 4 must exist in the first row whose index is 00. As the result of operating the Bank B as an exclusive cache in Mode 2, A0-2 exists in an incorrect set whose index is 10 in Mode 1. We call lines like A0-2 violated set lines. If violated set lines remain in the cache after shifting from Mode 2 to Mode 1, the cache operation falls into an unrecoverable state. One solution is to write all lines back before changing mode to invalidate violated set lines. However, this solution takes a long time and causes the performance degradation.

IV. Our Previous Low-Overhead Method and It's Problem

A. Previous Low- Overhead Method

To solve the problem, we have proposed the following low overhead technique [2]. Fig. 5 shows the mechanism of the technique. We add the number of different line counters into each cache set, (NoD1 and NoD2 in Fig. 5) the counters count the number of violated set lines in the other bank. The VLC uses these counters to access violated set lines in Mode 1. Fig. 5 also shows an example of cache state after VLC shifts from

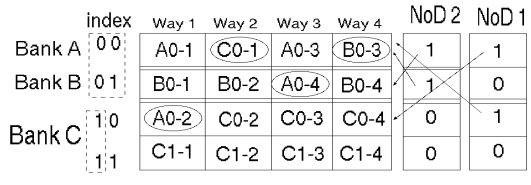


Fig. 5. Data placement on each way of VLC.

Mode 3 to Mode 1 via Mode 2. The 4 lines enclosed by the circles in Fig. 5 represent violated set lines. The NoD 1 is incremented with the line swapping in Mode 2, NoD 2 is incremented in Mode 3. This approach can handle violated set line A0-4 as the following steps.

Step 1: At first, VLC accesses the index 00, and then the access results in a miss.

Step 2: VLC refers to NoD 1 of index 00 and finds that a violated set line exists in the Bank C because the value is 1. Therefore, VLC accesses index 10 in the Bank C but cache miss occurs, again.

Step 3: VLC accesses index 01 in the Bank B, because NoD 2 value of index 00 is also 1, this means a violated set line exists in index 01, and the access generates a hit.

Step 4: VLC swaps A0-4 for B0-3 and NoD 2 counters are decremented because B0-3 is also violated set line.

In this case, the replaced line is selected from violated set lines. As a result of these steps, the number of violated set lines is gradually reduced without writeback. A NoD counter is required for each mode, but the cost of the NoD counter is small because the counter only adds log(#way) + 1 bits for each index (e.g., the counter requires 3 bits for 4 way associative).

B. Problem of the Previous Technique

The writeback reduction technique can prevent performance degradation caused by writing back the violated set lines when VLC shifts from higher mode to lower mode (e.g., from Mode 2 to Mode 1). On the other hand, the technique needs extra accesses in Mode 1 and Mode 2. In particular, the number of the extra access is increased when VLC executes a serious program that mainly has a low cache hit rate and occasionally gets a high cache hit rate because VLC operates in lower mode at that time. In ref [1], [2] the increase of extra accesses was not a problem because we estimated the energy consumption of VLC with a model for high leakage process according to previous trend. However, state-of-the-art CMOS device technologies reduce a ratio of leakage energy on a transistor contrary to the former prediction. Therefore, we revised the energy model for estimation based on recent technology.

In preliminary experiment, the dynamic energy consumption of VLC is increased to 170 % compared with normal cache due to increasing the number of cache accesses i.e., extra accesses in Mode 1 and Mode 2, and accesses to sleeping area in Mode 2 and Mode 3. With a program which has a high cache hit rate, extra accesses for violated set lines scarcely occur and VLC can reduce a large energy consumption because VLC operates most of time in Mode 3. However, with a program that mainly has a low cache hit rate and occasionally achieves a high cache hit rate, many extra accesses occur because VLC operates in Mode 1 or Mode 2 most of time. As a result, VLC consumes more dynamic energy and cannot reduce the energy consumption effectively. Hence, we need to consider a technique to prevent the increase of dynamic energy consumption of VLC.

V. Improvements of Variable Level Cache

A. Reduction of Bank Accesses in an Extra Access

This method reduces bank accesses on an extra access. An

extra access occurs to access only ways which have a violated set line. This means that the accesses to the other ways which do not have a violated set line are not necessary. Therefore, this approach reduces such futile accesses. VLC does not recognize which ways have a violated set line, therefore VLC accesses all ways for an extra access to obtain violated set lines. However, only 1 or 2 ways have violated set lines in average, and an extra access has many futile accesses to the other ways in which violated set lines do not exist. To reduce the number of the accessed ways on an extra access, NoD counter has to memorize which way has a violated set line. Therefore, we replaced NoD counters with a bitmap table to reduce accessed ways on an extra access. Fig. 6 shows VLC with bitmap table, we call this method 'violated set lines bitmap (VSL bitmap)'. VSL bitmaps memorize which ways have to be accessed when an extra access occurs. VLC can access only ways that have violated set lines using VSL bitmaps. In an extra access, VLC accesses only ways whose corresponding VSL bitmap is '1', this means the corresponding way has a violated set line. For example, in

Fig. 6, when VLC accesses to line A0-2 in Mode 1, the cache operates the following steps:

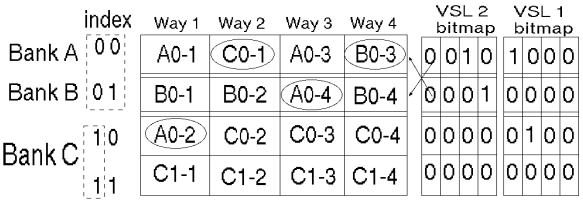


Fig. 6. Data placement on propose VLC with VSL bitmap

Step 1: At first, VLC accesses the index 00 because VLC operates like a normal cache in Mode 1.

Step 2: The access results in a miss. Then VLC refers VSL1 bitmap and notices that the Way 1 has a violated set line and should be accessed because the corresponding VSL1 bitmap is set to 1.

Step 3: VLC accesses only the Way 1 in Index 10 on the extra access.

The proposed VSL bitmap needs the number of ways bits in every set for a pseudo hierarchy, but the cost is less than 0.01% compared with 512KB 4-way cache memory. Accordingly, the hardware cost is very small compared with cache memory.

B. Dynamic Utilization of VLC

We also consider another method which has a potential to reduce a large dynamic energy. This method adds two new modes, normal cache mode and VLC mode, and uses these modes dynamically. This method uses VLC mode only for programs which have high hit rate, and normal cache mode for the other programs. In consequence the number of extra accesses is reduced in programs that have low hit rate. Whereas, it is difficult to get the attribute of a program before the program runs. In addition, it cannot react to the change of required cache size by a running program. In general, a hit rate of most programs occasionally gets high value while programs are running. VLC can also reduce the energy consumption for a short time in which the program gets the high hit rate. However, this method cannot reduce the energy consumption of a low hit rate program. If this method shifts from VLC mode to normal cache mode, this method needs to write all violated set line back because normal cache cannot access to violated set line. This writeback causes a large performance degradation. To prevent the writeback overhead, we adopted 2 thresholds, Thr I (threshold to increase the mode) and Thr D (threshold to decrease the mode), for changing the modes. Thr I is set to a high value which is defined to change the mode from VLC mode to normal cache mode and Thr D is set to a low value. These thresholds can prevent that the mode changing and the writeback in many time. We call this approach as 'Dyn Util'.

VI. Evaluation

A. Evaluation Environment

To evaluate proposed method, we developed a trace driven cache simulator, and we used M5 simulator [12] to generate trace data. Table I shows the processor architecture for this evaluation.

TABLE I: Environment of the Simulation

L1 I-Cache	Line Size:64 Byte, 32 Kb- 1 way, latency:1 cycle
L1 D-Cache	Line Size:64 Byte, 32 Kb- 1 way, latency:1 cycle
L2 Cache	Line Size:64 Byte, 32 Kb- 4way, latency:16 cycle
Main Memory	Infinity, latency:250 cycle
CPU clock frequency	1GHz
Wakeup overhead	10 cycle

The threshold of hit rate to change modes of the VLC is 30% that we decided by simulation experiment (30% and 20% are used to Thr I and Thr D on Dyn Util, respectively). We also use CACTI [13] to estimate energy consumption of these cache memory blocks. We adopted "total read dynamic power per read port at max frequency" as the energy of waking up the sleep mode memory because it is difficult to estimate the energy. Effect on the result of energy consumption of the VLC is less than 0.01% that is evaluated with the energy of waking up that is changed from 10 to 1000 times. We use 9 programs ammp, art, crafty, equake, gap, gcc, gzip, mcf and vpr from SPEC2000 benchmark suite [14]. The performance of the VLC is not affected by proposed VSL bitmap, and we measured only L2 cache energy consumption because it is difficult to measure energy of writing back to DRAM.

B. Calculation of Energy Consumption

The total energy consumption is calculated by sum of dynamic energy consumption and leakage energy consumption. Therefore, we calculated the energy consumption of VLC with the following expressions. The dynamic energy in VLC is consumed by two operations: cache access (E_{access}) and waking up from sleep mode (E_{wakeup}). Therefore, the dynamic energy of VLC is defined as

$$E_{dynamic} = E_{access} + E_{wakeup}. \quad (1)$$

E_{access} is defined as multiplication of the energy of one access and the number of accesses calculated by M5. E_{wakeup} in (1) is defined as multiplication of the energy of waking up once and the number of waking up. Next, we defined the leakage energy as

$$E_{static} = E_{set\_leak} \times N_{cache\_line} \times Cycles \div Freq. \quad (2)$$

We defined the leakage energy consumed by one set as $E_{set\ leak}$ that is calculated by the average ratio of sleeping sets and the leakage energy of one set, respectively. As we described above, the leakage energy in active is estimated with CACTI. The leakage energy in sleep mode is estimated by a ratio of power consumption on ACTIVE to STANDBY in SRAM data sheet [15]. Where $N_{cache\ line}$ is the number of cache line, Cycles is execution cycles of each program, and Freq is the clock frequency of the CPU.

Lastly, we estimate the energy consumed by the additional peripheral circuits that are needed to implement VLC. We estimated only the mode controller and the victim buffer by circuit simulation in transistor level using synthesized Verilog HDL design and PrimeTime PX. We do not make consideration for the energy of the other peripheral circuits in this evaluation because the energy of those components is very small (less than 0.1% compared to the mode controller and the victim buffer). All other values are calculated by our simulator.

C. Evaluation Results

Fig. 7, Fig. 8, and Fig. 9 show the simulation results of execution time, total energy consumption and dynamic energy consumption, respectively. In these figures, NoD Counter means the previous VLC [2] which has NoD counters, Dyn Util means the method which is described in section V-B, and VSL Bitmap means the proposed VLC which has novel VSL bitmap implementation. Each result is normalized by the result of the normal cache.

According to Fig. 7, both NoD Counter and VSL Bitmap increase execution time by less than 1% in all benchmarks compared to the normal cache. In addition, the execution times of the VLC are better than the normal cache in most benchmarks. This is a result of improving the cache hit rate caused by virtually increasing associativity due to the operation of the VLC that creates pseudo lower level cache. In particular, in Gzip, the cache hit rate improves approximately 3.6 times compared to the normal cache. As a result, VLC improves the execution time compared to the normal cache. The execution time of NoD Counter and VSL Bitmap are the same in principle. Dyn Util increases execution time by 8% in average compared to the normal cache. This is due to writing back of violated set lines when the mode changes from VLC mode to normal cache mode.

According to Fig. 8, in average, VSL Bitmap reduces 34% and 11% energy consumption compared to normal cache and NoD Counter, respectively. In particular, the energy consumption of Bzip2, Gap, Mcf and Vpr are significantly reduced. These benchmarks have low hit rate and are affected by increasing the number of ways. These four benchmarks run in Mode1 with many extra accesses most of time. Owing to this operation, the result of NoD Counter increases 35% energy consumption, but VLS Bitmap increases only 8% energy consumption at a maximum. Dyn Util increases the energy consumption by 1% in average compared to NoD Counter This is due to the leakage energy consumption caused by the increasing of the execution time.

According to Fig. 9, VSL Bitmap reduces 18% dynamic energy consumption compared to NoD Counter in average. In particular, in Mcf, VSL Bitmap reduces 37% dynamic energy consumption. Bzip, Gap, Mcf and Vpr that have low hit rate mostly run in Mode 1 with many extra accesses. Whereas, VSL Bitmap can save a large dynamic energy consumption consumed by the extra accesses because VSL Bitmap limits the number of ways accessed in extra accesses. For this reason, VSL Bitmap reduces more dynamic energy than NoD Counter. Dyn Util reduces 24% dynamic energy consumption compared to NoD Counter. However, it increases 20% leakage energy consumption. Therefore, Dyn Util is not helpful method to reduce the energy consumption of VLC.

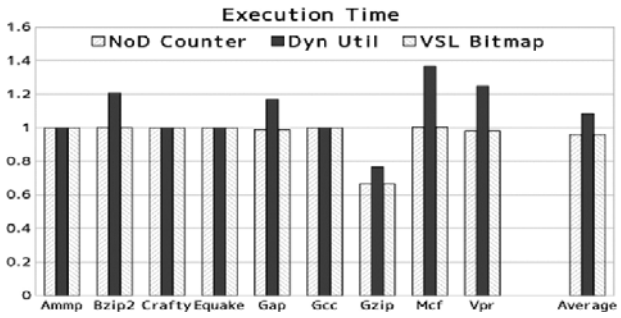


Fig. 7. Execution Time

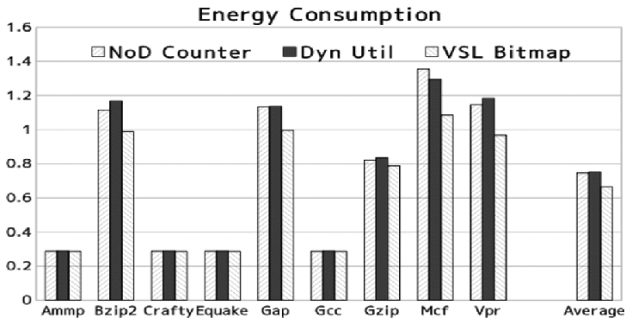


Fig. 8. Energy Consumption (Dynamic + Static)

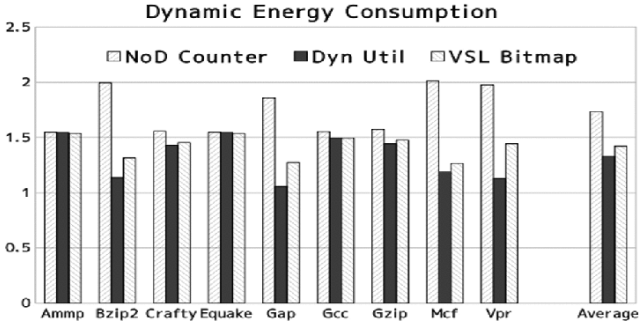


Fig. 9. Dynamic Energy Consumption.

VII. Conclusion

In this paper, we proposed a novel technique, VSL bitmap constructed with bitmap for VLC as dynamic energy reduction technique. The proposed VLC with VSL bitmap

reduces dynamic energy in an extra access than the previous VLC by limiting the number of the accesses to ways accessed in an extra access to violated set lines. We implemented a novel VSL bitmap to memorize which way has violated set lines. By the result of our evaluation, we show superiority of the novel VLC with VSL bitmap. We also show small hardware cost of the novel VSL bitmap. In the future work, we will design VLC VLSI chip and evaluate with the fabricated chip.

ACKNOWLEDGEMENTS

This work is supported by VLSI Design and Education Center, the University of Tokyo in collaboration with Synopsys Inc. and Rohm Corp.

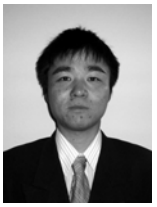
REFERENCES

[1] N. Matsubara, T. Sasaki, K. Ohno, and T. Kondo, "Evaluation of Variable Level Cache," in *Proc. of International Symposium on Information and Automation*, November 2010.
[2] K. Watanabe, T. Sasaki, K. Ohno, and T. Kondo, "Improvement of Writeback Mechanism of Variable Level Cache," in *Proc. of International Technical Conference on Circuits/Systems, Computers and Communications*, July 2012.
[3] Y. Zheng, B. T. Davis, and M. Jordan, "Performance Evaluation of Exclusive Cache Hierarchies," in *Proc. of IEEE International Symposium of Performance Analysis of Systems and Software*, pp. 89-96, 2004.
[4] S. J. Choi, J. W. Han, S. Kim, D. I. Moon, M. Jang, and Y. K. Choi, "Dopant- Segregated Schottky Source/Drain FinFET With a NiSi FUSI Gate and Reduced Leakage Current," in *Proc. of IEEE Transactions on Electron Devices*, November 2010.
[5] Y. Deng, R. Rupani, J. Johnson, and S. Springer, "Modeling of Gate Leakage, Floating Body Effect, and History Effect in 32nm HKMG PDSOI CMOS," in *Proc. of NSTI-Nanotech*, 2010.
[6] K. Flautner, N. Skim, S. Martin, D. Blaauw, and T. Mudge, "Drowsy Cache: Simple Techniques for Reducing Leakage Power," in *Proc. International Symposium on Computer Architecture*, pp. 148-157, May 2002.
[7] N. S. Kim, K. Flautner, D. Blaauwand, and T. Mudge, "Drowsy Instruction Caches; Leakage Power Reduction using Dynamic Voltage Scaling and Cache Sub-bank Prediction," in *Proc. International Symposium on Microarchitecture*, pp. 219-230, November 2002.
[8] N. Aziz, F. N. Najm, and A. Moshovos, "Low-Leakage Asymmetric-Cell SRAM," *IEEE Trans. On Very Large Scale Integration System*, vol. 11, no. 4, August 2003.
[9] U. Chaudhar and R. Jani, "A Study of Circuit Level Leakage Reduction Technique in Cache Memories," *IJERA*, vol. 3, pp. 457-460, April 2013.
[10] K. Meng, R. Joseph, and R. P. Dick, "Multi-optimization power management for chip multiprocessors," in *Proc. International Conference on Parallel Architecture and Compilation Techniques*, pp. 177-186, 2008.
[11] S. H. Yang, M. D. Powell, B. Falsafi, K. Roy, and T. N. Vijaykumar, "An Integrated Circuit / Architecture Approach to Reducing Leakage in Deep- Submicron High-Performance I-Caches," in *Proc. of the 7th Int. Symp. on High-Performance Computer Architecture*, pp. 147-157, 2001.
[12] N. L. Binkert, R. G. Dreslinski, L. R. Hsu, K. T. Lim, A. G. Saidi, and S. K. Reinhardt, "The M5 Simulator: Modeling Networked Systems," *IEEE Micro*, vol. 26, no. 4, pp. 52-60, Jul/Aug 2006.
[13] *CACTI 5.1 Shyamkumar Thoziyoor, Naveen Muralimanohar*, J. Ho Ahn, and N. P. Jouppi.
[14] SPEC -Standard Performance Evaluation Corporation. [Online]. Available: http://www.spec.org/.
[15] Alliance Semiconductor Corporation, Data sheet of "AS7C34096A 3.3V 512K 8 CMOS SRAM".

Ko Watanabe was born in Mie, Japan, on June 1, 1989. He received the B.S. degrees from Mie University in 2012. He has been a M.S. student at Mie University since 2012.

His current research interests are in low-energy cache memory.

Mr. Watanabe is a member of the Institute of Electronics, Information and Communication Engineers.



Takahiro Sasaki was born in 1975 in Aichi, Japan. He received the B.S., M.S. and Ph.D. degrees from Hiroshima City University in 1998, 2000, and 2003 respectively.

He is now assistant professor in the Graduate School of Engineering, Mie University. His research interests are in low-power and high-performance computing, multi-processor architecture, parallel processing and video codec hardware.

Dr. Sasaki is a member of the Institute of Electronics, Information and Communication Engineers and Information Processing Society of Japan.

Tomoyuki Nakabayashi was born in Mie, Japan, on February 6, 1987. He received the B.S., M.S. degrees from Mie University in 2009, 2011, respectively.

He has been a Ph.D. student at Mie University since 2011. His current research interests are in low-energy VLSI design and heterogeneous multi-core processor.

Mr. Nakabayashi is a member of the Institute of Electronics, Information and Communication Engineers.

Kazuhiko Ohno was born in Mie, Japan in 1970. He received the B.S., M.S. and Ph.D. degrees from Kyoto University in 1993, 1995, and 1998, respectively.

Since 2003, he has been an associate professor in the Graduate School of Engineering, Mie University. His research interests include the design, implementation, and optimization of parallel and distributed programming languages.

Dr. Ohno is a member of the Information Processing Society of Japan.

Toshio Kondo was born in Nagoya, Japan, on August 28, 1953. He received the B.S. and M.S. and Ph.D. degrees from Nagoya University in 1976, 1978 and 1996, respectively.

In 1978, he joined the Musashino Electrical Communication laboratories, Nippon Telegraph and Telephone Corporation (NTT), Tokyo, Japan, and had been engaged in research on SIMD parallel processors, character recognition systems and MPEG encoder LSIs. Since 2000, he has been a professor at Mie University, Mie, Japan. His current research interests include efficient motion estimators for UHDTV video compression, highly parallel processors for object recognition and processor instruction-set extensions for video processing.

Dr. Kondo is a member of the IEEE Circuits and Systems Society, the Institute of Electronics, Information and Communication Engineers and Information Processing Society of Japan.