A Leakage-Energy-Reduction Technique for Highly-Associative Caches in Embedded Systems

Akihito Sakanaka*
Panasonic Communications
akihito@mickey.ai.kyutech.ac.jp

Seiichirou Fujii
Kyushu Institute of Technology
seiichi@mickey.ai.kyutech.ac.jp

Toshinori Sato
PRESTO, JST
toshinori.sato@computer.org

ABSTRACT

Power consumption is becoming one of the most important constraints for microprocessor design in nanometer-scale technologies. Especially, as the transistor supply voltage and threshold voltage are scaled down, leakage energy consumption is increased even when the transistor is not switching. This paper proposes a simple technique to reduce the static energy. The key idea of our approach is to allow the ways within a cache to be accessed at different speeds and to place infrequently accessed data into the slow ways. We use dual-V_t technique to realize the non-uniform set-associative cache, and propose a simple replacement policy to reduce average access latency. Experimental results on 32-way set-associative caches demonstrate that any severe increase in clock cycles to execute application programs is not observed and significant static energy reduction can be achieved, resulting in the improvement of energy-delay product.

Categories and Subject Descriptors

C.1.1 [Processor Arhcitectures]: Single Data Stream Architectures—RISC/CISC,VLIW architectures.

General Terms

Performance, Design, Experimentation.

Keywords

Leakage current, cache memories, embedded processors.

1. INTRODUCTION

Power consumption is becoming one of the most important concerns for microprocessor designers in nanometer-scale technologies. Until recently, the primary source of energy consumption in digital CMOS circuits has been the dynamic power that is caused by dynamic switching of load capacitors. The trend of the reduction in transistor size reduces capacitance, resulting in less dynamic power consumption. Microprocessor designers have relied on scaling down the supply voltage, resulting in further dynamic power reduction[10, 11, 15]. In addition, many architectural technologies to reduce power have been proposed by reducing the number of switching activities[9, 13, 17]. To maintain performance scaling, however, threshold voltage must also be scaled down with supply voltage. Unfortunately, this increases leakage current exponentially. The International Technology Roadmap for Semiconductors (ITRS) predicts an increase in leakage current by a factor of two per generation[21]. Borkar estimates a factor of 5 increases in leakage energy in every generation[1].

Many of techniques[4, 12, 14, 19] proposed to address this problem have focused on cache memory that is a major energy consumer of the entire system because leakage energy is a function of the number of transistors. For example, the Alpha 21264 and the StrongARM processors use 30% and 60% of the die area for cache memories[17]. Current efforts at static energy reduction have focused on dynamically resizing active area of caches[4, 12, 14, 19]. These architectural techniques require additional circuits to control cache activities. The control circuitry including history tables consumes power, and thus these dynamic approaches are not suitable for embedded processors due to the area and power overhead and design complexity. In order to solve the problem, we propose a simple technique for leakage energy reduction.

The organization of the rest of this paper is as follows: Section 2 discusses the motivation of our work. Section 3 presents our concept to reduce leakage energy in caches, and explains a non-uniform set-associative cache as an implementation of the concept. Section 4 presents experimental results and discussion on the effectiveness of our approach. Finally, Section 5 concludes the paper.

2. MOTIVATIONS

2.1 Leakage Energy

Power consumption in a CMOS digital circuit is governed by the equation:

$$P = P_{active} + P_{off} \quad (1)$$

where P_{active} is the active power and P_{off} the leakage power. The active power P_{active} and gate delay t_{pd} are given by

$$P_{active} \propto fC_{load}V_{dd}^2 \quad (2)$$

*Akihito Sakanaka did this work while at Kyushu Institute of Technology.

$$t_{pd} \propto \frac{V_{dd}}{(V_{dd} - V_t)^{\alpha}} \quad (3)$$

where f is the clock frequency, C_{load} the load capacitance, V_{dd} the supply voltage, and V_t the threshold voltage of the device. α is a factor dependent upon the carrier velocity saturation and is approximately 1.3–1.5 in advanced MOSFETs[8]. Based on Eq.(2), it can easily be found that a power-supply reduction is the most effective way to lower power consumption. However, Eq.(3) tells us that reductions in the supply voltage increase gate delay, resulting in a slower clock frequency, and thus diminishing the computing performance of the microprocessor. In order to maintain high transistor switching speeds, it is required that the threshold voltage is proportionally scaled down with the supply voltage.

On the other hand, the leakage power can be given by

$$P_{off} = I_{off} V_{dd} \quad (4)$$

where I_{off} is the leakage current. The subthreshold leakage current I_{off} is dominated by threshold voltage V_t in the following equation:

$$I_{off} \propto 10^{-\frac{V_t}{S}} \quad (5)$$

where S is the subthreshold swing parameter and is around 85mV/decade[21]. Thus, lower threshold voltage leads to increased subthreshold leakage current and increased static power. Maintaining high transistor switching speeds via low threshold voltage gives rise to a significant amount of leakage power consumption.

2.2 Dual-V_t CMOS

The dual-threshold (dual-V_t) technique[20, 23] addresses the tradeoff decision between high performance and low leakage power. Transistors located on critical paths are assigned low threshold voltage, whereas transistors that are not critical to timing can tolerate high threshold voltage and slow switching speeds. The selection of threshold voltages are conducted at design time, and no additional circuits to dynamically control threshold voltages are required. Table 1 shows leakage current for high and low threshold voltage transistors in a 70nm process technology[6]. We can see that the leakage energy of transistors with high threshold voltage is a factor of 75 smaller than that of transistors with low threshold voltage. Hence, replacing a low-V_t transistor with a high-V_t transistor results in substantial energy reduction. In addition, different from gated-V_{dd}[19] technique, which cannot maintain data in cache memories when the power supply is cut off and thus is not applicable to cache memories, the dual-V_t technique does not occur any additional cache misses. In summary, dual-V_t is a simple and efficient technique for static energy reduction of embedded processors, for which die size is of large concern as well as energy consumption.

Table 1: Impact of V_t on Leakage Current

Tr type	V_{dd} (V)	V_t (V)	I_{off} (nA)
High-V_t	0.75	0.4	26
Low-V_t	0.75	0.2	1941

2.3 Related Work

Current efforts at static energy reduction have focused on dynamically resizing active area of caches[4, 12, 14, 19]. These architectural techniques employ circuit techniques. VT+ADR cache[4] and SA cache[12] use VT-CMOS[16] and ABC-MOS[18], respectively, both of which control the substrate bias to reduce leakage current in sleep mode by raising up threshold voltage. Decay cache[14] and DRI cache[19] use gated-V_{dd}[19], which shuts off the supply voltage to SRAM cells to reduce leakage current. These circuit techniques have some disadvantages. Gated-V_{dd} loses the state within the memory cell in the sleep mode. Thus, additional cache misses might occur, resulting in an additional dynamic power consumption. In contrast, VT-CMOS and ABC-MOS can retain stored data in the sleep mode. However, VT-CMOS require a triple-well structure and a charge-pump circuit, and ABC-MOS requires an additional power line that must be distributed throughout the memory array. In addition, these architectural techniques require additional circuits to control cache activities. The control circuitry consumes power, however, most studies did not consider its effect. In summary, these dynamic approaches are not suitable for embedded processors due to the area and power overhead and design complexity.

3. NON-UNIFORM SET-ASSOCIATIVE CACHE

3.1 Our Approach

In this paper, we propose a simple technique for leakage energy reduction. It uses the dual-V_t technology[20, 23]. As explained above, the selection of threshold voltages are conducted at design time, and thus no additional circuitry is required for dual-V_t. While loss of adaptability might consume more leakage power than the dynamic techniques, removal of the power overhead as well as the complex circuitry is beneficial especially for embedded processors. From these considerations, we adopt dual-V_t to our proposed technique for static energy reduction. The key idea of our approach is to allow the ways within a cache to be accessed at different speeds and to place infrequently accessed data into the slow ways. This exploits dynamic information regarding data criticality in order to reduce power, as circuit techniques, such as transistor size optimizations[7] and clustered voltage scaling technique[22], exploit static information regarding timing criticality. Only critical data will be placed into the fast ways. While this technique is originally proposed for high-performance processors[3], we adopt it for low-power embedded processors and we call our proposed cache non-uniform set-associative (NUSA) cache.

3.2 Implementation Details

While we use a 2-way set-associative cache for explanation, our proposal is applicable to any set- and full-associative caches.

The non-uniform 2-way set-associative cache consists of a pair of a fast and a slow ways. Transistors with low threshold voltage are used for implementing the fast way. Similarly, transistors with high threshold voltage are used for implementing the slow way. In this NUSA, once a way is allocated to a referred datum, the datum is placed in the same way until replacement. Thus, frequently used data might be

placed in the slow way. This is not desirable, since processor performance is diminished. In order to place frequently used data into the fast way, we extend the NUSA by exploiting the locality in way reference. The locality means that a way recently accessed will be referred again in the near future[9]. Thus, it is desirable to place data used last into the fast way. If the way accessed last is the slow way, or if the cache does not hold referred data and the slow way is allocated, we exchange the lines between the fast and the slow ways as shown in Figure 1. It should be noted that a structural hazard can occur during every way-exchange. For example, consider two load instructions are executed. When the first datum is found in one of the slow way, the first load takes 2 cycles. However, the datum needs to be written into the fast way in the following cycle. As a result, data cache cannot be used for other access in the cycle. This structural hazard increases the slow-way access latency. In addition, the way-exchange consumes dynamic power. However, if the reduction in the leakage energy is larger than the increase in the dynamic energy, the NUSA with the way-exchange is a good solution since it improves execution cycles.

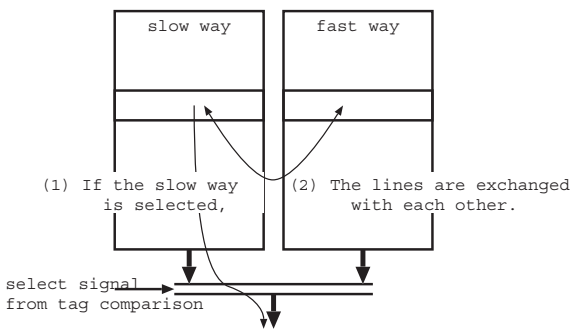

Figure 1: Way-Exchange

4. EXPERIMENTAL RESULTS

4.1 Simulation Methodology

We implemented our simulator using the SimpleScalar/ARM tool set[2]. We use ARM instruction set architecture in this study. The processor model evaluated is based on Intel XScale processors[10]. A single-ported 32KB, 32B block, 32-way set-associative L1 instruction and data caches are used. They have a load latency of 1 cycle and a miss latency of 32 cycles. The non-uniform 32-way set-associative L1 instruction and data caches consist of one primary fast way and remaining 31 slow ways. It has the same load latency of 1 cycle when the requested data is placed in the primary way and otherwise has a latency of 2 cycles. The structural hazard in the port during every way-exchange is considered in detail. During the event of an exchange, a new instruction or datum cannot be fetched. Hence, while instruction fetch suffers a large penalty every way-exchange, data fetch may suffer a smaller penalty if data cache is not accessed immediately after a slow-way access. We also evaluate caches whose load latency is always 2 cycles for comparison. This slow cache is pipelined and has the same throughput of 1 with the baseline model. The replacement policy is based on FIFO. No L2 cache is used. A memory operation that follows a store whose data address is unknown cannot be executed.

The MiBench[5] is used for this study. It is developed for use in the context of embedded, multimedia, and communications applications. It contains image processing, communications, and DSP applications. We use original input files provided by University of Michigan. We select two categories: Consumer and Telecommunications. Table 2 lists the benchmarks we used. All programs are compiled by the GNU GCC with the optimization options specified by University of Michigan. Each program is executed to completion.

Table 2: Benchmark programs

Consumer	
cjpeg	JPEG encode
jpeg	JPEG decode
lame	MP3 encode
mad	MPEG audio decode
tiff2bw	TIFF convert
tiff2rgba	TIFF convert
tiffdither	TIFF convert
tiffmedian	TIFF convert
Telecommunications	
FFT	Fast Fourier Transform
IFFT	Inverse FFT
Rawcaudio	ADPCM encode
Rawdaudio	ADPCM decode
Toast	GSM encode
Untoast	GSM decode
CRC	32-bit Cyclic Redundancy Check

4.2 Energy Parameters

To use data from Hanson et al.'s[6], we assume a 70nm process technology, a 2.5GHz clock frequency, a 0.75V supply voltage, and a 110C operating temperature. To compute total energy reduction, we first compute the leakage energy using the numbers shown in Table 1 and the number of clock cycles to execute each program. To account for the cost of the way-exchange, we include the dynamic energy to read and store data between two ways. Hanson et al. reported that the energy to access the 64K L1 caches is 0.07 nJ, and we assume that an additional energy of 0.07 * 2 = 0.14 nJ is required for each way-exchange. This assumption leads to a higher energy estimate, because the caches modeled in [6] are 2 times larger in capacity than the caches we consider. Thus, our results are conservative in the sense that the dynamic energy for the way-exchange will decrease.

4.3 Results

Figures 2 – 7 show simulation results. Figures 2 and 3 show way hit ratio. The hit means that a cache access refers the primary fast way. The left bar is for the instruction cache, and the right is for the data cache. As can be easily seen, both caches achieve 85% of the way hit ratio for most programs.

Figures 4 and 5 show relative processor performance. For each group of 3 bars, the left one indicates the performance which has the conventional set-associative cache, whose load latency is always 1 cycle (denoted by Fast). The middle indicates that of the NUSA cache (denoted by NUSA), and the right indicates that of the slow conventional cache, which

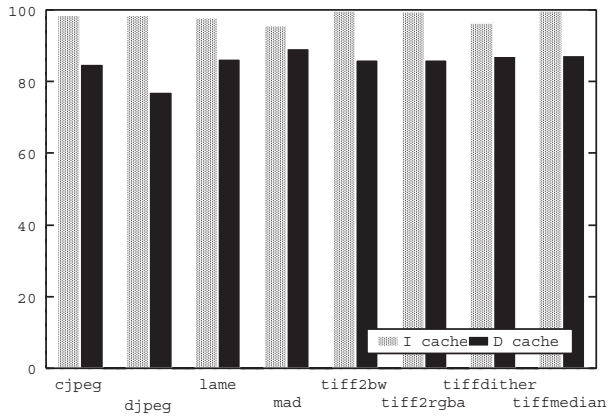

Figure 2: %Way Hit Ratio (consumer)

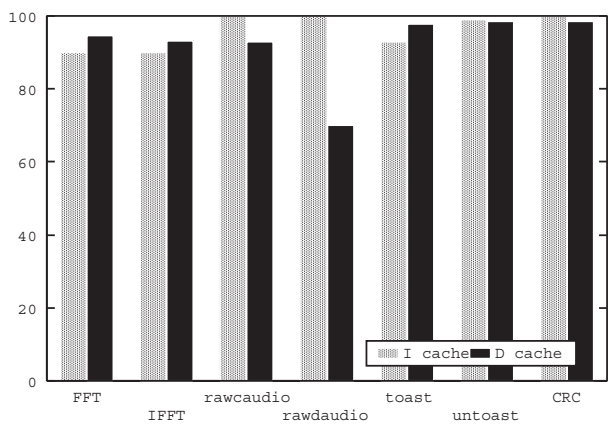

Figure 3: %Way Hit Ratio (telecomm)

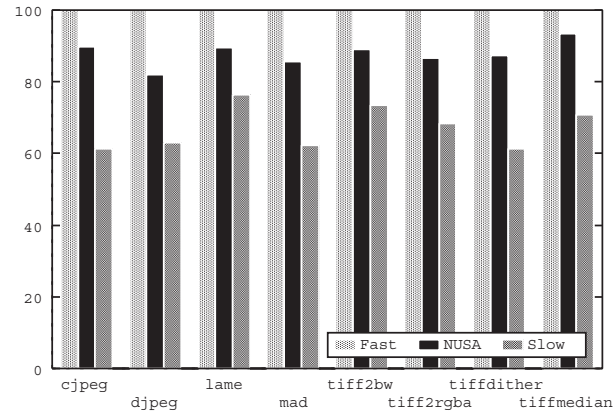

Figure 4: %Processor Performance (consumer)

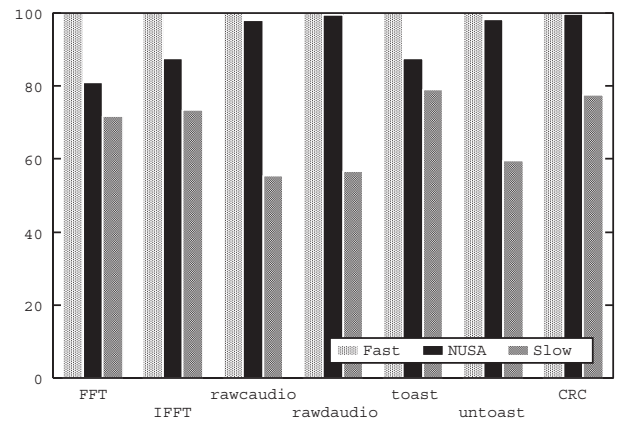

Figure 5: %Processor Performance (telecomm)

has a load latency of 2 cycles (denoted by Slow). Every result is normalized by that of the Fast model. We can find the model with the NUSA cache has no considerable difference from that with the conventional fast and energy-hungry cache. In addition, it has significant performance gain over that with slow and energy-efficient cache.

The static energy consumed in the NUSA is reduced by approximately the factor of 30, when we do not consider the energy consumed during the way-exchange. Since the way-exchange rarely occurs as we have seen in Figures 2 and 3, the efficiency of the NUSA on leakage current reduction will not be diminished. Based on the discussions in Section 4.4.2, the energy consumed during the way-exchange is calculated. Figures 6 and 7 show the percent energy-delay product of the NUSA cache normalized by that of the conventional power-hungry cache. Each bar is divided into to parts. The bottom part indicates the energy-delay product due to leakage power consumption, and the upper indicates that due to the energy consumed during the way-exchange. We can find up to 92% reduction of energy-delay product is achieved. In general, programs that do not degrade their performance (see Figures 4 and 5) achieve more improvement of energy-delay product.

5. CONCLUSIONS

In this paper, we have proposed a simple technique to reduce the static energy consumed in caches. The key idea of our approach is to allow the ways within a cache to be accessed at different speeds and to place infrequently accessed data into the slow ways. Simulation results showed that any severe increase in clock cycles to execute the application program was not observed and significant static energy reduction could be achieved, resulting in the improvement of energy-delay product.

6. REFERENCES

[1] S. Borker, "Design challenges of technology scaling," IEEE Micro, volume 19, number 4, 1999.

[2] D. Burger and T. M. Austin, "The SimpleScalar tool set, version 2.0," ACM SIGARCH Computer Architecture News, volume 25, number 3, 1997.

[3] D. Burger, "Technology scaling challenges for microprocessors and systems," Invited Lecture, COOL Chips V, 2002.

[4] R. Fujioka, K. Katayama, R. Kobayashi, H. Ando, and T. Shimada, "A preactivating mechanism for a VT-CMOS cache using address prediction," International Symposium on Low Power Electronics and Design, 2002.

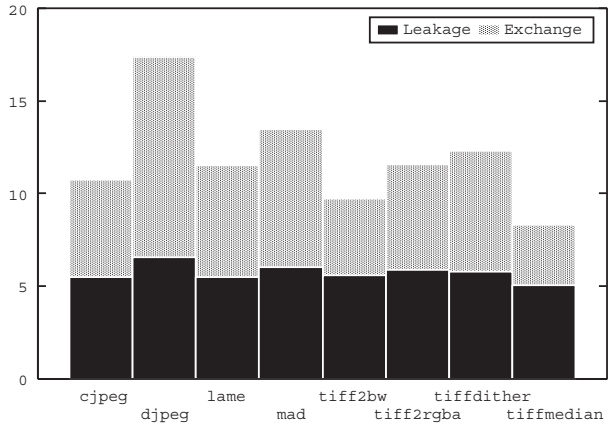

Figure 6: %Relative Energy-Delay Product (consumer)

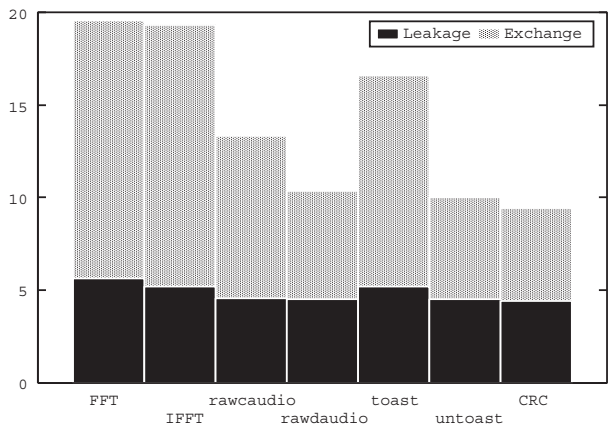

Figure 7: %Relative Energy-Delay Product (telecomm)

[5] M. R. Guthaus, J. S. Ringenberg, D. Ernst, T. M. Austin, T. Mudge, R. B. Brown, "MiBench: A free, commercially representative embedded benchmark suite," Workshop on Workload Characterization, 2001.

[6] H. Hanson, M. S. Hrishikesh, V. Agarwal, S. W. Keckler, and B. Burger, "Static energy reduction techniques for microprocessor caches," International Conference on Computer Design, 2001.

[7] M.Hashimoto and H.Onodera, "Post-Layout transistor sizing for power reduction in cell-based design," IEICE Transactions on Fundamentals, volume E84-A, number 11, 2001.

[8] T. Hiramoto and M. Takamiya, "Low power and low voltage MOSFETs with variable threshold voltage controlled by back-bias," IEICE Transactions on Electronics, volume E83-C, number 2, 2000.

[9] K. Inoue, T. Ishihara, and K. Murakami, "Way-predicting set-associative cache for high performance and low energy consumption," International Symposium on Low Power Electronics and Design, 1999.

[10] Intel Corp., "Intel XScale technology," http://developer.intel.com/design/intelxscale/, 2002.

[11] T. Ishihara and K. Asada, "A system level memory power optimization technique using multiple supply and threshold voltages," Asia and South Pacific Design Automation Conference, 2001.

[12] T. Ishihara and K. Asada, "An architectural level energy reduction technique for deep-submicron cache memories," Asia and South Pacific Design Automation Conference, 2002.

[13] A. Iyer and D. Marculescu, "Power aware microarchitecture resource scaling," Design, Automation and Test in Europe Conference and Exhibition, 2001.

[14] S. Kaxiras, Z. Hu, G. Narlikar, and R. McLellan, "Cache-line decay: a mechanism to reduce cache leakage power," Workshop on Power Aware Computer Systems, 2000.

[15] A. Klaiber, "The technology behind Crusoe processors," Transmeta Corporation, White Paper, 2000.

[16] T. Kuroda, T. Fujita, S. Mita, T. Nagamatsu, S. Yoshioka, F. Sano, M. Norishima, M. Murota, M. Kato, M. Kinugasa, M. Kakumu, and T. Sakurai, "A 0.9V, 150MHz, 10mW, 4mm^2, 2-D discrete cosine transform core processor with variable-threshold-voltage scheme," International Solid State Circuit Conference, 1996.

[17] S. Manne, A. Klauser, and D. Grunwald, "Pipeline gating: speculation control for energy reduction," International Symposium on Computer Architecture, 1998.

[18] K. Nii, H. Makino, Y. Tujihashi, C. Morishima, and Y. Hayakawa, "A low power SRAM using auto-backgate-controlled MT-CMOS," International Symposium on Low Power Electronics and Design, 1998.

[19] M. Powell, S. H. Yang, B. Falsafi, K. Roy, and T. N. Vijaykumar, "Gated-Vdd: a circuit technique to reduce leakage in deep-submicron cache memories," International Symposium on Low Power Electronics and Design, 2000.

[20] S. Sirichotiyakul, T. Edwards, C. Oh, J. Zuo, A. Dharchoudhury, R. Panda, and D. Blaauw, "Stand-by power minimization through simultaneous threshold voltage selection and circuits sizing," International Design Automation Conference, 1999.

[21] D. Sylvester and H. Kaul, "Power-driven challenges in nanometer design," IEEE Design & Test of Computers, volume 18, number 6, 2001.

[22] K.Usami and M.Horowiz, "Clustered voltage scaling technique for low-power design," International Symposium on Low Power Design, 1995.

[23] L. Wei, Z. Chen, M. Johnson, and K. Roy, "Design and optimization of low voltage and high performance dual threshold CMOS circuits," International Design Automation Conference, 1998.