delay. Last, we found the area and delay penalties associated with using a rectangular EPLC over a square EPLC. This result does not mean that rectangular EPLCs are a bad idea; in many applications, the fixed shapes and sizes of the other cores will dictate that a rectangular EPLC is to be used. Our goal in this paper was to optimize the EPLC core for a given aspect ratio.

REFERENCES

[1] eASIC Corporate Website [Online]. Available: http://www.easic.com

[2] Actel Corporate Website [Online]. Available: http://www.actel.com

[3] M2000 Corporate Website [Online]. Available: http://www.m2000.fr

[4] V. Betz and J. Rose, "VPR: A new packing, placement, and routing tool for FPGA research," *Int. Workshop Field-Programmable Logic and Applications*, pp. 213–222, Aug. 1997.

[5] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAs*. Norwell, MA: Kluwer, 1999.

[6] J. Rose, A. E. Gamal, and A. Sangiovanni-Vincentelli, "Architectures of field-programmable gate arrays," *Proc. IEEE*, vol. 81, no. 7, pp. 1013–1029, Jul. 1993.

[7] S. J. E. Wilton, "Architectures and algorithms for field-programmable gate arrays with embedded memory," Ph.D. dissertation, Univ. Toronto, Dept. Elect. Comput. Eng., Toronto, ON, Canada, 1997.

[8] M. I. Masud and S. J. E. Wilton, "A new switch block for segmented FPGAs," in *Int. Workshop Field-Prog. Logic and Applic.*, Sep. 1999, pp. 274–281. Lect. Notes in Comp. Sci. 1673.

[9] Y.-W. Chang, D. Wong, and C. Wong, "Universal switch modules for FPGA design," *ACM Trans. Design Automation of Electron. Syst.*, vol. 1, pp. 80–101, Jan. 1996.

[10] H. B. Fan, J. P. Liu, Y.-L. Wu, and C. C. Cheung, "On optimum designs of universal switch blocks," *Field Programmable Logic Applicat.*, pp. 142–151, Sep. 2002.

[11] Y.-W. Chang, K. Zhu, G.-M. Wu, D. F. Wong, and C. K. Wong, "Analysis of FPGA/FPIC switch modules," *ACM Trans. Design Automation Electron. Syst. (TODAES)*, vol. 8, no. 1, pp. 11–37, Jan. 2003.

[12] H. Schmit and V. Chandra, "FPGA switch block layout and evaluation," in *Proc. ACM/SIGDA 10th Int. symp. Field-Programmable Gate Arrays*, Feb. 2002, pp. 11–18.

[13] G. Lemieux and D. Lewis, "Analytical framework for switch block design (PDF)," *Field-Prog. Logic Applicat.*, pp. 122–131, Sep. 2002.

[14] H. Fan, J. Liu, Y.-L. Wu, and C.-C. Cheung, "On optimum switch box designs for 2-D FPGA's," in *Proc. Design Automation Conf.*, 2001, pp. 203–208.

[15] V. Betz and J. Rose, "Directional and nonuniformity in FPGA global routing architectures," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, 1996, pp. 652–659.

[16] P. Hallschmid and S. J. E. Wilton, "Detailed routing architectures for embedded programmable logic IP cores," in *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*, Monterey, CA, Feb. 2001, pp. 69–74.

[17] V. Betz and J. Rose, "Automatic generation of FPGA routing architectures from high-level descriptions," in *Proc. ACM/SIGDA Int. Symp. Field Programmable Gate Arrays*, 2000, pp. 175–84.

[18] ——, "FPGA routing architecture: Segmentation and buffering to optimize speed and density," in *Proc. ACM/SIGDA Int. Symp. Field Programmable Gate Arrays*, 1999, pp. 59–68.

[19] *The Programmable Logic Data Book*, Xilinx Corp., 1996.

[20] S. Yang, "Logic Synthesis and Optimization Benchmarks, Version 3.0,", Tech. Rep., 1991.

[21] E. M. Sentovich *et al.*, "SIS, A System for Sequential Circuit Analysis,", Tech. Rep. UCB/ERL M92/41, 1992.

[22] J. Cong and Y. Ding, "Flowmap: An optimal technology mapping algorithm for delay optimization in lookup-table based FPGA designs," *IEEE Trans. Computer-Aided Des. Integr. Circuits Syst.*, vol. 13, no. 1, pp. 1–12, Jan. 1994.

[23] A. Marquardt, V. Betz, and J. Rose, "Using cluster-based logic blocks and timing-driven packing to improve FPGA speed and density," in *Proc. ACM/SIGDA Int. Symp. Field-Prog. Gate Arrays*, 1999, pp. 37–46.

[24] P. Hallschmid, "Detailed routing architectures for embedded programmable logic IP cores," Master's thesis, Univ. of British Columbia, Vancouver, BC, Canada, 2003.

Microarchitecture-Level Leakage Reduction With Data Retention

Weiping Liao, Joseph M. Basile, and Lei He

***Abstract*—In this paper, we study microarchitecture-level leakage energy reduction by power gating. We consider the virtual power/ground rails clamp (VRC) and multithreshold CMOS (MTCMOS) techniques and apply VRC to memory-based units for data retention and MTCMOS to the other units. We propose a systematic methodology for leakage reduction at the microarchitecture level, in which profiling of idle period distribution and ideal power gating analysis are used to select a target component for realistic power gating. We show that the ideal leakage energy reduction can be up to 30% of the total energy for the modern high-performance very long instruction word processors we study and that the secondary level (L2) cache contributes most to the reduction. We further improve the existing adaptive cache decay method for leakage reduction by using VRC for data retention and name it *VRC decay*. Applied to L2 cache, the VRC decay, on average, increases performance by 5.6% and reduces system energy by 24.1%, compared to the adaptive cache decay without data retention.**

***Index Terms*—Cache memories, circuit modeling, computer architecture, power.**

I. INTRODUCTION

The leakage current in nanometer devices has increased drastically due to reduction in threshold voltage, channel length, and gate oxide thickness [1]. In addition, an increasing number of modules in a highly integrated system are idle at any given time. The high-leakage devices and low activity rates both contribute to the growing significance of leakage power at the system level. The Intel Pentium IV processor at 3 GHz has an almost equal amount of leakage and dynamic power [2]. Therefore, leakage reduction has become important.

Power gating reduces leakage power by inserting sleep transistors between the power supply and logic or memory circuits, and these sleep transistors are turned off to cut off the power supply when the circuits are idle. The following circuit-level implementations of power gating have been proposed. Multithreshold CMOS (MTCMOS) [3] uses sleep transistors with high threshold voltage to reduce more leakage compared to using sleep transistors with normal threshold voltage. Virtual power/ground rails clamp (VRC) [4] improves the MTCMOS with data retention by inserting diodes parallel to sleep transistors. Similarly, drowsy cache [5] reduces the supply voltage of the idle cache to a small and predefined level, such that the cache leakage power is reduced with data retention. DRG cache [6] inserts an NMOS sleep transistor that is sized carefully for data retention between normal SRAM cells and the ground line. Additionally, focusing on memory units such as the level-one instruction cache and register file, the leakage-biased bitlines (LBB) method [7] tries to reduce the leakage energy consumed by inactive SRAM cells during precharge. The transition energy overhead

Manuscript received July 8, 2004; revised May 26, 2005. This paper was supported in part by the National Science Foundation under CAREER Award CCR-0401682 and SRC Grant 1116, by Mindspeed, Fujitsu Laboratories of America, and Intel through a UC MICRO grant, and by a Faculty Partner Award from IBM.

W. Liao and L. He are with the Electrical Engineering Department, University of California, Los Angeles, CA 90095 USA (e-mail: wliao@ee.ucla.edu; lhe@ee.ucla.edu).

J. M. Basile is with Intel Corporation, Santa Clara, CA 95054 USA (e-mail: joseph.m.basile@intel.com).

Digital Object Identifier 10.1109/TVLSI.2005.859560

and the minimum time for LBB to be effective for leakage reduction are considered in [7].

Microarchitecture-level leakage energy reduction has been studied using MTCMOS without data retention. The work in [8] leverages scheduling slacks among instruction bundles to increase the chance of power gating integer units in very long instruction word (VLIW) processors. [9] proposes an adaptive power gating to resize L1 cache for leakage reduction. The work in [10] develops the cache decay method to dynamically power gate a cache line, which contains dead cache data not to be used in the near future, as measured by the *decay interval*. The decay interval is fixed in [10] but can be dynamically adjusted according to benchmark behaviors in [11] for more leakage reduction. In [12], the authors further propose a feedback control-based method to adjust the decay interval and minimize leakage with respect to the targeted performance. L1 caches are also studied in that work.

The following papers consider microarchitecture-level leakage reduction with data retention. The work of [13] combines the circuit-level power gating design similar to the DRG cache and the cache decay method at the microarchitecture level for leakage reduction in the cache hierarchy containing L1 and L2 caches, but it does not consider the transition overhead (both time and energy) of power gating. A study similar to that in [13] was also presented in [14]. The initial study of this paper [15] proposes a simple, yet effective, time-out scheme applying VRC to reduce leakage in L2 cache. The time-out scheme is similar to cache decay, except that the entire L2 cache is turned off in [15] rather than individual cache lines in the cache decay.

All of the aforementioned work (except our initial study [15]) focuses only on parts of microprocessors (e.g., caches or integer units). However, in such cases, because leakage reduction techniques inevitably lead to a certain performance penalty and prolong the execution time, the leakage energy consumed by the other parts of microprocessors actually increases. Therefore, the leakage reduction on a specific microarchitectural component does not necessarily guarantee the reduction of total energy of the whole microprocessor. Clearly, a systematic study for leakage energy reduction from the perspective of the whole processor is necessary.

In this paper, we complete the initial study in [15] and study microarchitecture-level leakage reduction, considering modern high-performance VLIW microprocessors (but not just integer units or caches) and using power and timing models from circuit-level designs. Specifically, we make the following contributions.

1) We propose to evaluate leakage reduction techniques and pinpoint the power gating candidates for given circuit-level techniques, based on profiling of the idle time distribution and the minimum idle time of circuit-level techniques.[1] We show that, for general-purpose computing workloads (SPEC) on modern VLIW processors, ideal power gating with in-time scheduling can reduce up to 30% of the total energy, and L2 cache contributes most to the reduction.
2) We study realistic power gating by applying the feedback control-based cache decay method from [12] to L2 cache, called *adaptive cache decay* in this paper. We also employ VRC with data retention to improve the adaptive cache decay, and we name the new approach *VRC decay*. Targeting L2 cache, the VRC decay, on average, increases performance by 5.6% and reduces system energy by 24.1%, compared to the adaptive cache decay without data retention. On the other hand, ignoring the system-level impact, the adaptive cache decay reduces L2 cache leakage energy but increases the total processor energy for benchmark *art*. This indicates the importance of using the systematic methodology for microarchitecture-level leakage reduction proposed in this paper.

The remainder of the paper is organized as follows. In Section II, we discuss circuit-level power and timing models for VRC and MTCMOS. In Section III, we propose the systematic method for microarchitecture-level leakage reduction, and we apply the method to study microarchitecture-level leakage reduction with ideal power gating. In Section IV, we study realistic leakage energy reduction for L2 cache. Finally, we conclude in Section V. An extended abstract regarding the preliminary results of this study was published in [15].

II. Circuit-Level Leakage Power Reduction

Circuits that feature the use of power gating exhibit three operating modes, given as follows.

1) *Active mode*, in which a circuit performs operations and dissipates both dynamic power (P_d) and leakage power (P_s). The sum of P_d and P_s is defined as active power (P_a).
2) *Standby mode*, in which a circuit is idle but ready to execute an operation, dissipating only leakage power (P_s).
3) *Inactive mode*, also known as *sleep mode*, in which a circuit is deactivated by power gating or other leakage reduction techniques, dissipating a reduced static leakage power defined as inactive power (P_i).

The circuits in the inactive mode are not ready to execute any operation. There are two important transitions: 1) *shut-down*, when circuits are deactivated from the standby mode to the inactive mode, and 2) *wake-up*, when circuits are switched from the inactive mode to the standby mode. Leakage reduction techniques typically have a dynamic energy overhead dissipated during both shut-down and wake-up transitions, denoted as E_{sd} and E_{wk}, respectively. For an idle period t_{idle}, during which a circuit is shut down for power reduction and then woken up for operations, the leakage energy reduction should be larger than such overhead so that enforcement of leakage reduction techniques is worthwhile. Such constraint dictates a lower bound of the idle period. We name such lower bound as the *minimum idle time* (MIT), which is calculated as [16]

$$\mathrm{MIT} = \frac{E_{\mathrm{sd}} + E_{\mathrm{wk}} - P_i * (t_{\mathrm{sd}} + t_{\mathrm{wk}})}{P_s - P_i}. \tag{1}$$

Leakage power reduction techniques are beneficial only when the $t_{\mathrm{idle}} >= MIT$.

In this paper, we consider two leakage power reduction techniques: MTCMOS [3] and VRC [4]. MTCMOS uses high-V_t sleep transistors connected to GND, with the logic implemented by low-V_t transistors. The sleep transistors can be turned off to reduce leakage power. However, there is no data retention guaranteed by MTCMOS. VRC, on the one hand, meets the need of data retention by placing diodes across the sleep transistors for GND. On the other hand, it introduces more transition energy and has a lower leakage reduction ratio, compared to MTCMOS. The detailed assessments for MTCMOS and VRC are shown in [15]. In our subsequent microarchitecture-level experiments, we chose VRC for memory-based components and MTCMOS for the remaining components. We use the power model developed in [17], assuming a fixed temperature of 80 °C. Overall, the average errors of the power models are less than 7%, compared to SPICE simulation [17].

[1] Although the concept of minimum idle time was introduced in [7], it was not used at system level to identify the target components for leakage energy reduction.

TABLE I
POWER-RELATED PARAMETERS. VRC IS APPLIED ON MEMORY-BASED UNITS SUCH AS BTB, REG, IL1, DL1, AND L2, WHILE MTCMOS IS APPLIED TO THE OTHER UNITS. THE MIT AND t_{wk} ARE IN THE UNIT OF CYCLES. WE ASSUME A 3-GHz CLOCK FREQUENCY AND 100-nm TECHNOLOGY

Component	P_a (mW)	P_s (mW)	P_i (mW)	$E_{transition}$ (μ J)	M.I.T.	t_{sd}	t_{wk}
Branch target buffer (BTB)	251.1848	37.7506	0.3693	1.98×10^{-3}	158	5	5
Register file (Reg)	157.0423	1.6597	0.0163	7.88×10^{-5}	144	5	5
One cache set in L1 instruction cache (IL1)	1.7628	0.0686	2.74×10^{-3}	3.496×10^{-6}	155	5	5
One cache set in L1 data cache (DL1)	1.7627	0.0685	2.74×10^{-3}	3.496×10^{-6}	155	5	5
One cache set in L2 unified cache (L2)	1.3376	0.4993	0.01997	1.06×10^{-1}	157	5	5
One decode unit (Decode) out of six	78.836	20.443	0.204	4.82×10^{-6}	7	3	3
One integer unit (IALU) out of four	118.254	30.665	0.307	7.23×10^{-6}	7	3	3
One FP unit (FALU) out of two	236.508	61.331	0.613	1.45×10^{-5}	7	3	3

III. METHODOLOGY OF MICROARCHITECTURE-LEVEL LEAKAGE ENERGY REDUCTION

Targeting EPIC/VLIW architecture, we use PowerImpact [18] as our experiment platform. We choose the same system configuration and component partition of the target VLIW processor as that in [15]. Shown in Table I, the power consumption and the minimum idle times for each component are obtained based on our power model in the same ways as that in [15] and [17], for memory-based units and logic circuits, respectively.

A. Microarchitecture-Level Leakage Reduction Methodology

As we have shown in Section II, leakage energy reduction techniques are beneficial only when the target component has an idle period no less than its minimum idle time (MIT). The distribution of idle period depends on the workload, but MIT is inherited from the circuit-level leakage reduction technique. Simply taking a circuit-level leakage reduction technique and arbitrarily applying it to a microarchitecture component may not be effective for leakage energy reduction. Instead, we propose a systematic approach for microarchitecture-level leakage reduction as follows. First, we study the potential of leakage reduction on given workload and circuit-level techniques by calculating the percentage of idle periods longer than the MIT. Such study can eliminate ineffective circuit-level techniques or validate our selection of circuit-level techniques to components. Then, we study ideal power gating, which provides the upper bound of the leakage energy savings and pinpoint the appropriate components for microarchitecture-level leakage reduction. Finally, we can focus on the target components for microarchitecture-level leakage energy reduction.

B. Microarchitecture-Level Leakage Power Reduction With Profiling and Ideal Power Gating

For general purpose computing workloads (SPEC) on modern VLIW processors, it has been shown in [19] that, with MTCMOS and VRC, there is plenty of potential for leakage energy reduction for the whole processor at the microarchitecture level. We then study ideal power gating by assuming that we can schedule a power gating event *in time* for any idle period longer than the minimum idle time to maximize power savings, and we can wake up a component *in time* to avoid performance loss. We use the same methodology of ideal power gating as that in [15], except for the fact that we turn on/off each cache set for caches, instead of the whole cache in [15]. The power savings of ideal power gating provides a theoretical upper bound of the leakage power reduction without losing any performance.

TABLE II
WHOLE SYSTEM POWER WITH IDEAL SCHEDULING. *go*, *li*, *ijpeg*, *mcf*, *parser*, AND *bzip2* ARE INTEGER BENCHMARKS, WHILE *equake* AND *art* ARE FLOATING POINT BENCHMARKS

	No gating	Clock gating	Ideal Power Gating
go	100%	52.63%	14.43%
li	100%	55.04%	23.41%
ijpeg	100%	62.21%	29.79%
mcf	100%	51.93%	23.22%
parser	100%	53.08%	17.43%
bzip2	100%	56.89%	27.27%
equake	100%	57.35%	23.49%
art	100%	52.88%	28.07%
average	100%	55.25%	23.39%

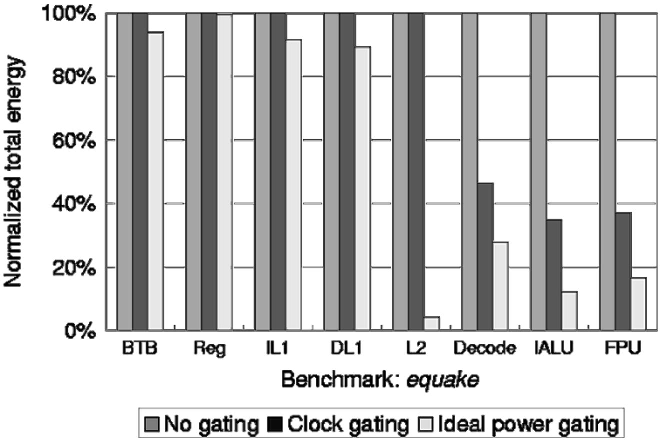

Fig. 1. Leakage power reduction under ideal power gating. Only one of the benchmarks are shown as the others bear a similar trend.

In Table II, we compare the total power of the entire processor for a number of benchmarks[2] under three situations: 1) no gating; 2) clock gating; and 3) ideal power gating. Compared to the no gating case, the total power can be reduced to 55.25% and 23.39% on average by using clock gating and ideal power gating, respectively. In other words, ideal power gating achieves up to 76.61% total power reduction. The gap between power savings values with clock gating and power gating is the upper bound of the leakage power for a system. Table II indicates that such an upper bound can be up to 30% for the modern high-performance VLIW processor we study.

IV. REALISTIC LEAKAGE ENERGY REDUCTION AT THE MICROARCHITECTURE LEVEL

We first decide on the candidate for realistic microarchitecture-level leakage energy reduction based on the results of ideal power gating. Fig. 1 presents the energy reduction for each component in our experiments. By observing the difference between energy consumed in clock gating and in ideal power gating, it is easy to see that power gating is effective to reduce power for the L2 cache, IALU, and FPU and obtains the largest power reduction for the L2 cache. Therefore, we focus on

[2] We use eight benchmarks in our experiments but are unable to use any benchmark written in Fortran language because the IMPACT toolset does not provide compiler support for Fortran programs.

the L2 cache for microarchitecture-level leakage energy reduction in this section.

A. Realistic Leakage Energy Reduction for the L2 Cache

We choose the cache decay method [10] in our realistic leakage energy reduction for the L2 cache. In the cache decay method, each cache line is individually turned off by MTCMOS if it has not been accessed for a given amount of time called the *decay interval*. The cache decay is justified because most of the data in the cache line are unlikely to be used in the near future, as measured by the decay interval [20]. Once a cache line is turned off, the data in that line are lost due to no data retention with MTCMOS. Such a property introduces additional cache misses called *induced misses*. The work in [12] further proposes a method based on the feedback control theory to adaptively adjust the decay interval and minimize leakage with respect to the target performance. In such a feedback control mechanism, the decay interval is updated for every given time window according to the number of induced misses during that time window. If the number is larger than a predefined threshold, the decay interval is increased so that cache lines stay at the standby mode for more time to avoid the induced misses; otherwise, the decay interval is decreased so that cache lines are turned off more frequently to reduce leakage.

We apply the feedback control-based cache decay method from [12] to each cache set in our set-associative L2 cache, and we name this method *adaptive cache decay* in this paper. In addition, we enhance this adaptive cache decay method by using VRC with data retention to replace MTCMOS, and we name the new approach *VRC decay*. With the VRC decay, since data in cache sets is preserved when turning them off, there are no additional cache misses such as induced misses. However, whenever a cache set is hit and it was turned off by VRC, we have to wake up the set before accessing it. We call this situation a *VRC miss*. The wakeup time and transition energy of the VRC misses lead to performance and energy overhead in the VRC decay method.

In our implementation of the feedback control mechanism for the adaptive cache decay and the VRC decay, a feedback controller is in charge of adjusting the decay interval. The controller has two preset parameters: the *gain* and the *setpoint*. The input of this controller is the number of the induced misses (in the adaptive cache decay) or the VRC misses (in the VRC decay) during the time window, and the controller updates the decay interval according to

$$\Delta T = \text{gain} \times (N_{\text{miss}} - \text{setpoint}) \tag{2}$$

where ΔT is the change in decay interval and N_{miss} is the number of the induced or VRC misses during the last time window.

We choose the same experiment settings for both the adaptive cache decay and the VRC decay, except for the setpoint. We set the gain as 8, similar to [12]. We limit the decay interval between 4k cycles and 512k cycles because these values cover a reasonable range of the decay interval as pointed out by [12]. The decay interval imposes a low bound on the time windows, as it is undesirable to change the decay interval in the middle of the counting interval [12]. Therefore, we choose the time window as 512k cycles in our experiments. The setpoint is the number of misses we want to maintain during a time window, and this can be decided according to the target performance. We target 5% IPC degradation, which corresponds to a clock cycle increase of roughly 5% due to induced misses or VRC misses. For the adaptive cache decay, we set the setpoint to 100 because 5% of the cycles for one time window corresponds to a miss penalty of 100 induced misses, given the miss penalty for each induced miss 255 cycles (miss penalty of the L2 cache). Following the same derivation, for VRC misses, with the miss penalty of each VRC miss at five cycles (wake-up time of VRC), we set the setpoint for the VRC decay as 5000.

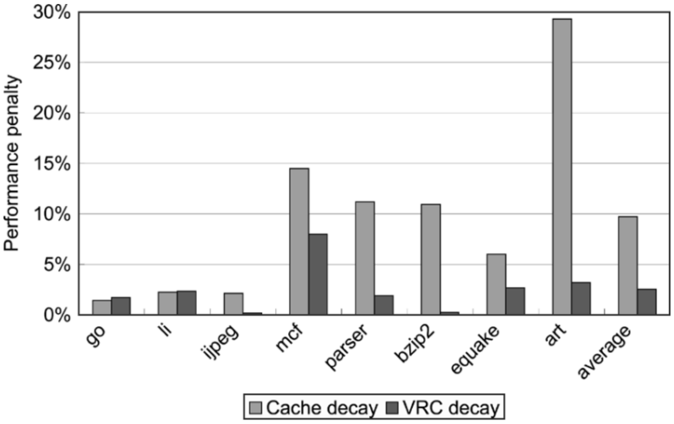

Fig. 2. Performance penalty for the adaptive cache decay and the VRC decay.

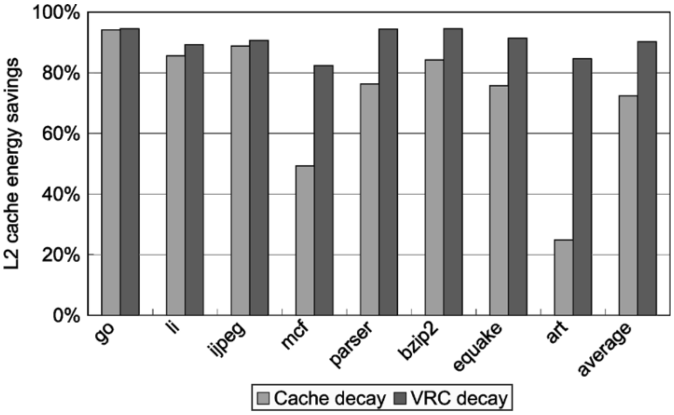

Fig. 3. L2 cache energy savings for the adaptive cache decay and the VRC decay.

We use IPC degradation to represent the performance penalty. In both the adaptive cache decay and the VRC decay, the performance penalty and energy savings are obtained by comparison to the case without any decay method applied. Fig. 2 compares the percentage of performance penalty between the adaptive cache decay and the VRC decay. For the adaptive cache decay, the induced misses have significant impact on system performance because: 1) our study focuses on L2 cache with a miss penalty as large as 255 cycles and 2) the in-order nature of VLIW processors makes it impossible to hide the memory latency by executing independent instructions out of order. As we can see from Fig. 2, the performance penalty can be up to 30% for the adaptive cache decay. Although we design the feedback controller to target a 5% performance penalty, the performance impact associated with the induced misses is so significant for some benchmarks, such as *art* and *mcf*, that even by constantly applying the upper bound of the decay interval in our system (512k cycle), we still cannot achieve the 5% target performance penalty (i.e., the performance penalty is beyond the adjustable range of our feedback controller). On the other hand, the performance penalty for the VRC decay is smaller than that for the adaptive cache decay due to: 1) no additional cache misses because of data retention and 2) smaller miss penalty for one VRC miss (the wakeup time of VRC, which is merely five cycles in our experiments) compared to that for induced misses. On average, with VRC decay, we can achieve as little as 2.5% performance penalty compared to 9.7% performance penalty with the adaptive cache decay. This difference is equivalent to a 5.6% performance increase with the VRC decay compared to the adaptive cache decay.

Fig. 3 compares the percentage of L2 cache energy savings between the adaptive cache decay and the VRC decay. Clearly, the VRC decay achieves larger energy savings than the adaptive cache decay for all

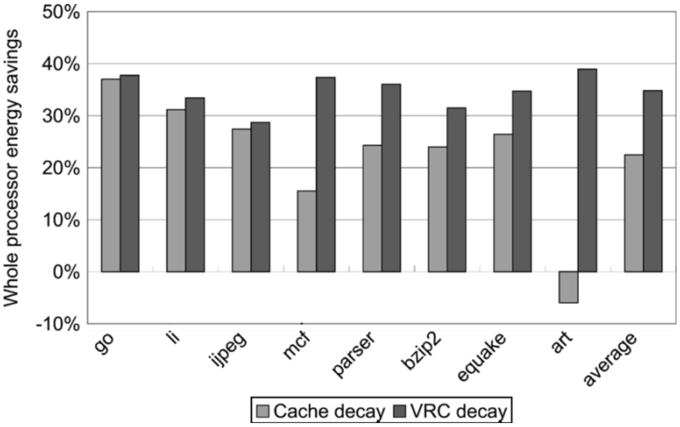

Fig. 4. Whole processor energy savings for the adaptive cache decay and the VRC decay.

benchmarks. Although the adaptive cache decay benefits from the larger leakage reduction and smaller transition energy associated with MTCMOS, two sources of additional energy overhead offsets this benefit: 1) leakage energy during longer execution time due to larger performance penalty, compared to VRC decay and 2) excessive dynamic energy during additional cache refills due to induced misses. Fig. 4 compares the percentage of whole processor system energy savings between the adaptive cache decay and the VRC decay. On average, the system energy savings by the adaptive cache decay and the VRC decay are 22.48% and 34.81%, respectively. Equivalently, compared to the adaptive cache decay, VRC decay, on average, reduces the system energy by 24.1%.

Furthermore, from Figs. 3 and 4, we observe that for benchmark *art* with the adaptive cache decay, although L2 cache energy is reduced by as much as 25%, the total energy of the whole processor does not decrease, but increases. The reason is that the performance penalty of leakage reduction techniques leads to additional execution time, and the increase of leakage energy consumed by system components due to such additional execution time exceeds the leakage energy reduction of L2 cache. Clearly, if the performance penalty is severe, it may not even be beneficial to turn off L2 cache for leakage energy savings from the total system energy point of view. Therefore, it is critically important to evaluate any leakage reduction technique with appropriate methodology such as the one in Section III-A before applying such technique.

V. Conclusion

In this paper, we have studied how to reduce leakage energy at the microarchitecture level considering power gating in the forms of VRC and MTCMOS, with VRC featuring data retention. We propose a systematic approach for realistic microarchitecture-level leakage reduction, based on profiling of the period distribution and ideal power gating. We have shown that the ideal leakage energy reduction can be up to 30% of the total energy for the modern high-performance VLIW processors we study, and the L2 cache contributes most to the reduction.

We have further enhanced the existing adaptive cache decay method for leakage reduction by using VRC for data retention, and we name it *VRC decay*. Targeting the L2 cache, the VRC decay on average increases performance by 5.6% and reduces system energy by 24.1%, compared to the best existing method, which is the adaptive cache decay without data retention.

Acknowledgment

The authors would like to thank R. Patel and W. Jiang at Intel Corporation for discussions related to the virtual power/ground rails clamp technique during the summer of 2001 and Dr. K. Lepak at Advanced Micro Devices, Inc., for helpful discussions.

References

[1] W. Liao, J. M. Basile, and L. He, “Leakage power modeling and reduction with data retention,” in *Proc. ICCAD*, Nov. 2002, pp. 714–719.

[2] S. Heo, K. Barr, M. Hampton, and K. Asanovic, “Dynamic fine-grain leakage reduction using leakage-biased bitlines,” in *ACM SIGARCH Computer Architecture News*, vol. 30, 2002, pp. 137–147.

[3] L. Li, I. Kadayif, Y.-F. Tsai, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, and A. Sivasubramaniam, “Leakage energy management in cache hierarchies,” in *Proc. Int. Conf. Parallel Architecture and Compilation Techniques*, 2002, pp. 131–140.

[4] Y. Li, D. Parikh, Y. Zhang, K. Sankaranarayanan, M. Stan, and K. Skadron, “State-preserving vs. nonstate-preserving leakage control in caches,” in *Proc. DATE*, Feb. 2004, pp. 22–27.

[5] A. Agarwal, C. H. Kim, S. Mukhopadhyay, and K. Roy, “Leakage mechanisms and leakage control for nano-scale cmos circuits,” in *Proc. DAC*, Jun. 2004, pp. 6–11.

[6] A. S. Grove, “Changing vectors of Moore’s law,” in *Int. Electron Devices Meet Dig.*, Keynote speech, Dec. 2002.

[7] S. Mutoh *et al.*, “1-V power supply high-speed digital circuit technology with multithreshold-voltage cmos,” *IEEE J. Solid-State Circuits*, vol. 30, no. 8, pp. 847–854, Aug. 1995.

[8] K. Kumagai *et al.*, “A novel powering-down scheme for low vt cmos circuits,” in *Symp. VLSI Circuits Dig. Tech. Papers*, 1998, pp. 44–45.

[9] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. Mudge, “Drowsy caches: Simple techniques for reducing leakage power,” in *Proc. Int. Symp. Computer Architecture*, May 2002, pp. 148–157.

[10] A. Agarwal, H. Li, and K. Roy, “DRG-cache: A data retention gated-ground cache for low power,” in *Proc. DAC*, Jun. 2002, pp. 473–478.

[11] C. Long and L. He, “Distributed sleep transistor network for leakage power reduction,” *Proc. DAC*, pp. 181–186, Jun. 2003.

[12] W. Zhang, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, D. Duarte, and Y.-T. Fai, “Exploiting vliw schedule slacks for dynamic and leakage energy reduction,” in *Proc. MICRO*, vol. 34, Dec. 2001, pp. 102–113.

[13] S.-H. Yang, M. D. Power, B. Falsafi, K. Roy, and T. Vijaykumar, “An integrated circuit/architecture approach to reducing leakage in deep-submicron high-performance i-caches,” in *Proc. ACM/IEEE Int. Symp. High-Performance Computer Architecture (HPCA)*, Jan. 2001, pp. 147–158.

[14] S. Kaxiras, Z. Hu, and M. Martonosi, “Cache decay: Exploiting generational behavior to reduce cache leakage power,” in *Proc. Int. Symp. Computer Architecture*, May 2001, pp. 240–251.

[15] H. Zhou, M. C. Toburen, E. Rotenberg, and T. M. Conte, “Adaptve mode control: A static-power-efficient cache design,” in *Proc. 10th Int. Conf. Parallel Architectures and Compilation Techniques*, 2001.

[16] S. Velusamy, K. Sankaranarayanan, D. Parikh, T. Abdelzaher, and K. Skadron, “Adaptive cache decay using formal feedback control,” in *Proc. Workshop on Memory Performance Issues*, May 2002, pp. 1–10.

[17] D. Duarte, Y. Tsai, N. Vijaykrishnan, and M. J. Irwin, “Evaluating runtime techniques for leakage power reduction techniques,” in *Proc. ASP-DAC*, 2002, pp. 31–38.

[18] W. Liao and L. He, “Coupled power and thermal simulation and its application,” *Lecture Notes in Computer Science*, vol. 3164/2004, pp. 148–163, 2004.

[19] Powerimpact (2002). [Online]. Available: http://eda.ee.ucla.edu/PowerImpact/

[20] L. He, W. Liao, and M. S. Stan, “System level leakage reduction considering the interdependence of temperature and leakage,” in *Proc. DAC*, Jun. 2004, pp. 12–17.

[21] D. A. Wood, M. D. Hill, and R. E. Kessler, “A model for estimating trace-sample miss ratios,” in *Proc. ACM SIGMETRICS*, Jun. 1991, pp. 79–89.