LEAKAGE ENERGY REDUCTION IN ON-CHIP MICROPROCESSOR CACHES

Zhang Chengyi, Zhang Minxuan and Xing Zuocheng
Lab 610, School of Computer Science
National University of Defense Technology
410073, Changsha, China
E-mail: chengyizhang@nudt.edu.cn

KEYWORDS
Leakage energy, Microprocessor, Cache, LRU-assist, ADSR.

ABSTRACT

Leakage power is becoming dominant part of the microprocessor chip power budget as feature size shrinks. Leakage energy consumption is of particular concern in memory structures, such as on-chip caches, for large scale transistors and rare access. Chipmakers have proposed many low leak circuit techniques for cache leakage control, in which gated-vdd and DVS are two effective methods. In this paper, based on these two circuits we propose two architectural mechanisms, LRU-assist and ADSR, to reduce leakage energy consumption in on-chip cache hierarchies. LRU-assist decay combines time-based decay with existing LRU information to aggressively cut off lines in L1 cache. ADSR (Always Drowsy Speculatively Recover) puts the whole L2 cache in low leakage state all the time and speculatively recovers line's supply voltage using prefetch-like mechanism. We run SPEC CPU2000 benchmarks on a cycle accurate simulator to evaluate their efficiency. LRU-assist decay reduces L1 cache leakage power by 55% on average and improves EDP by 2%. ADSR with next-line recover reduces L2 cache leakage power by 66% on average and improves EDP by 47%.

INTRODUCTION

Controlling power dissipation is becoming a challenging job in high performance microprocessor design. Processor power contains mostly two parts (N.H.E. Weste 2005): dynamic power and static power. Dynamic power arises from circuit switches, and is determined by voltage, switch frequency and effective capacitance. Static power, which comes from leakage current, is always there no matter what state the circuits are. Leakage current is sensitive to voltage, technical parameter and temperature. As feature size shrinks leakage current increases exponentially due to threshold voltage drop, short channel effect, drain-induced barrier lowering, gate-induced drain leakage and thermal feedback. According to ITRS reports (SIA 2004), static power dissipation on chip will surpass dynamic power in the next several generations. Leakage power warrants new approaches for managing it.

On-chip caches are the most leakage energy intensive components in microprocessor for large scale transistors and low access frequency. In order to close the speed gap between CPU and memory, bigger (more bytes), wider (more ways) and deeper (more hierarchies) caches are integrated. So reducing cache leakage energy dissipation is a first-line work.

Many circuit techniques have been proposed in the past to reduce leakage power, such as MTCMOS (K. Nii et al. 1998), gated-vdd (M. D. Powell et al. 2002) and DVS (T. Pering et al. 1998). When used to reduce SRAM leakage power, these circuits can be categorized into state-destroying and state-preserving. Gated-vdd is state-destroying circuit which results in the holding contents lost when the cell is gated. Accesses to the gated cells need data reloading from the lower memory level, which leads to extra dynamic energy consumption. DVS can be state-preserving method when a lower voltage is switched on instead of completely cutting off. So the ability of reducing leakage energy is not as aggressive as gated-vdd. Cells supplied by this lower voltage can not be accessed either, but the contents are just preserved, as we call them in drowsy state. Drowsy cells need several cycles to wake up (drive to the normal voltage) before they are able to be accessed, which results in pipeline stall and performance drop. All these special circuits can reduce leakage current efficiently but inflexibly. Which transistors to be control and when to act need to be decided by architectural or micro-architectural structures according to real-time parameters.

Cache decay (S. Kaxiras et al. 2001) and drowsy cache (K. Flautner et al. 2002) use time-based strategy to control leakage power. Counters are associated with each cache line, so the extra leakage power of counters and extra dynamic power of counter clocking are considerable. Moreover, the counters can not be dynamic configured, so before a cache line is put into low leakage state, the preset number of cycles must elapse since its last access. If cache lines are determined dead effectively before counters overflow, a great portion of leakage energy can be saved.

According to Lin Li (L. Li 2002), L1 cache should use state-destroying strategy to aggressively reduce the leakage energy consumption, and L2 cache should use state-preserving strategy to avoid severe performance effect

Proceedings 20th European Conference on Modelling and Simulation
Wolfgang Borutzky, Alessandra Orsoni, Richard Zobel © ECMS, 2006
ISBN 0-9553018-0-7 / ISBN 0-9553018-1-5 (CD)

because off chip access is a time-consuming and energy-consuming work. In this paper, we propose several new algorithms for leakage power controlling in on-chip cache hierarchies. The rest of this paper is organized as follows. Section 2 introduces the experiment framework and evaluation metric of our work. Section 3 introduces LRU-assist algorithm for L1 cache leakage power reduction. ADSR mechanism for L2 cache leakage control is proposed in section 4. Section 5 concludes this paper and plans our future work.

METHODOLOGY

Architectural level power optimization is always based on power analyzing tools and true application simulating. This section describes our simulation environments, including the underlying processor architecture and benchmarks. Evaluation metric is also described.

Simulator And Benchmarks

Simulations in this paper are based on the SimpleScalar framework (D. Burger and T. Austin 1997). Our processor model closely resembles Alpha 21264 (R. Kessler 1999) (Table 1), and on-chip caches are the main objective for leakage control evaluation. Power estimation models are based on Princeton Wattch (D. Brooks 2000; P. Shivakumar and N. P. Jouppi 2001) and Virginia HotLeakage (Y. Zhang 2003). We choose 70nm technology with 0.9V supply voltage and 0.3V drowsy voltage, 5.6GHz clock speed and 80℃ environment temperature. The threshold voltages of N-transistor and P-transistor are 0.19V and 0.21V respectively.

The benchmark suite for this study consists of a set of thirteen SPEC CPU2000 benchmarks: ammp, applu, art, equake, lucas, mgrid, swim, bzip2, gcc, gzip, mcf, twolf and vortex. They are compiled for the Alpha instruction set using Compaq Alpha compiler with SPEC peak settings. In order to capture the most important program behaviors while at the same time accelerating simulation, we use the simulation points that were described and verified in SimPoint (T Sherwood et al. 2002).

Metric

The leakage power of caches is mainly determined by active ratio, so off-ratio can be used to evaluate the optimization ability of leakage control mechanism. Considering the extra misses and extra stalls as a result of putting cache lines into low leakage mode, energy-delay product (EDP) is used to trade-off performance and energy consumption. Lower EDP represents higher energy efficiency. Simulation cycles are gathered to represent delay. Energy is computed as the sum of dynamic energy and leakage energy of only on-chip cache hierarchies. This is reasonable because a large transistor budget is allocated for on-chip memories and our leakage control methods only impact the power of cache hierarchies.

Table1: Configuration of Simulated Processor

Processor core	
Instruction Window	80-RUU, 40-LSQ
Issue width	4
Function Unit	4-IntALU, 1-IntMULT 2-FpALU, 1-FpMULT 2-MemPort
Memory Hierarchy	
L1 DCache	64KB, 4-way, 32Bblock, WB
L1 ICache	64KB, 2-way, 32B block, WB
Unified L2 Cache	1MB, 4-way, 32B block, WB, 6-cycle latency
Memory	100 cycles, 16 bus width

LEAKAGE OPTIMIZATION IN L1 CACHES

L1 cache is close to processor core and relatively small. State-destroying low leakage circuits are appropriate to L1 cache because the overhead of data reloading from lower level memory hierarchy is small. To reduce the miss rate, set associative L1 cache is always equipped. LRU algorithm is a simple but the most popular replacement strategy in set-associative caches. It can be described as a function $LRUi : \{0,1,\cdots,n-1\} \rightarrow Wi$. n is associativity, and Wi is the blocks in set i . LRU can effectively reduce cache miss rate and can be implemented with systolic ordering array [19]. $LRU(n-1)$, which has the longest idle time since its last access, is always the most preferential candidate for replacement. In high performance processors, high associative (≥4) caches are essential. But as associativity increases, cache occupancy toboggans and accesses hardly hit in $LRU(\geq n/2)$. We simulated SPEC CPU2000 benchmarks on our framework. Figure 1 shows the $LRU(\geq n/2)$ hit rate for 2, 4, 8-way 64KB L1 DCache. Hit rate is approximately 1% on average, no more than 3%. So LRU information gives us some hints that which cache lines can be candidates for leakage power optimization.

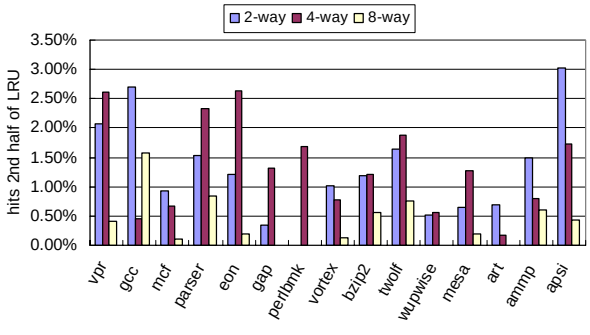

Figure 1: Hit Rate in the Second Half of LRU List

LRU-based Algorithm

In order to reduce hardware and power overheads, LRU information can be directly used to cut off cache lines as soon as possible. Ordinary LRU-based algorithm constantly puts the latter w/n part (w least recently used lines) of each set into low leakage mode. n is associativity and w is a number between 0 and n. $w=0$ represents the conventional cache, and $w=n$ represents a slower cache (each access needs a next-level fetch). So $w=1,2,\cdots,n-1$ is our concern.

Obviously, ordinary LRU-based algorithm is not very useful for state-destroying leakage control circuits like gated-vdd, because w/n LRU-based low leakage cache is almost the same as $(n-w)$ -way set associative cache, or even worse. To confirm this observation, we run 13 benchmarks in the framework mentioned above with 2/4 LRU-based leakage control enabled and without any leakage control but half associativity cache (32KB, 2-way) respectively. The EDP results are illustrated in figure 2. EDP shows the poor ability of ordinary LRU-based algorithm in energy-performance trade-off. So ordinary LRU-based leakage control mechanism is worthless and gilds the lily. But it can be combined with time-based gated-vdd strategy to more aggressively optimize leakage power.

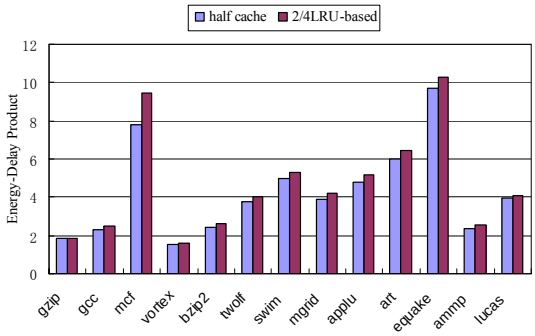

Figure 2: EDP of Ordinary LRU-based Algorithm

LRU-assist Algorithm

In essence, the function of LRU list and line-associated counters in time-based strategy partially overlaps. Cache line order in LRU list is always consistent with the counter value order. In one set the cache line whose counter overflows firstly must be $LRU(n-1)$, and anytime the counter value of $LRU(n-1)$ is also affirmatively the biggest one. Without regard to access latency and miss penalty, the counters for leakage control can completely replace LRU list, while it is impractical especially for large scale caches. So LRU is still necessary, but it can complement counters as an assistant for controlling cache leakage power more aggressively. We call this LRU-assist.

Considering the blindness to runtime states when use ordinary LRU-based algorithm to control cache leakage, we extend it with adaptivity. Instead of toggled in ordinary LRU-based algorithm, w is alterable in LRU-assist algorithm. Each set has its own way-mask wi. Corresponding to the cold start of cache, wi is initialized n which means all cache lines are cut off. A cache miss in set i decreases wi by 1 until $wi=0$. When $LRUi(k-1)$ (the k^{th} most recently used cache line in the LRU list of set i) is shut off due to counter overflow, which means this way has been idle for a long time, we predict the $LRUi(k-2)$ way will also overflow in the near future and set $wi=n-k+1$. The cache is controlled by LRU and counter jointly. wi swings from 0 to n in this manner, which implies a metaphorical balance between performance and power. Figure 3(a) shows the state machine of wi in a 4 way set associative cache.

To reduce the area and energy overhead of way masks ($n*sets$ bits), one alternative is sharing only one way mask w throughout the whole cache. Then all sets have the same off ratio. There is a little change in algorithm. When a cache line is cut off due to counter overflow, w increases by 1 until $n-1$. Figure 3(b) illustrates the state machine.

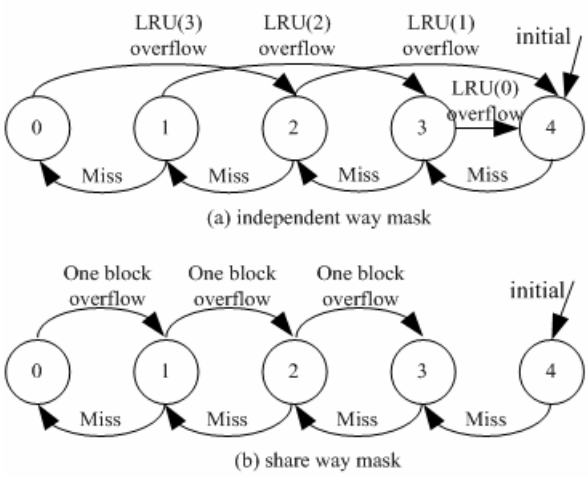

Figure 3: State Machine of wi in 4 Way Caches

Results

In essence, LRU-assist is a kind of prediction or even gamble based on statistical information instead of obtrusion, indicating the trade-off among performance and leakage energy. In this section, we will evaluate the LRU-assist algorithm by running true applications. We apply LRU-assist decay to 4-way set associative L1 Dcache, which dissipates a considerable portion of energy in modern microprocessors. We examine the L1 Dcache off ratio of LRU-assist decay and counter-only decay strategy (decay interval is 32k cycles). We also show the EDP metric for performance-energy trade-off evaluation. Figure 4 shows the experimental results. LRU-assist decay can gain approximately 9% more leakage power saving than counter-only decay on average. Normalized EDP results for 4-way L1 Dcache based on conventional cache without any leakage control method

are illustrated in figure 5. LRU-assist gains the lowest EDP on average, approximately 2% lower than baseline. All the energy efficiencies of gzip decrease because the miss rate of this benchmark is very sensitive to cache size and performance is greatly reduced when leakage control methods are used.

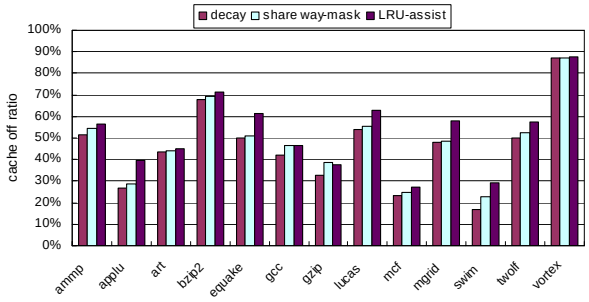

Figure 4: L1 Cache Off Ratio Using LRU-assist Decay

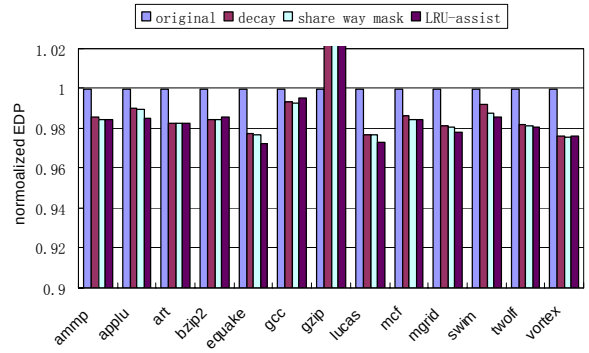

Figure 5: Normalized EDP Using LRU-assist Decay

LEAKAGE OPTIMIZATION IN L2 CACHE

L2 cache is the biggest component on microprocessor chip, but it is also rarely accessed due to L1 cache filter. So leakage power is the dominant item in L2 cache. Theoretically all other leakage control mechanisms are also work on L2 cache. But L2 cache has two larruping feature: high miss penalty due to off chip access and large capacity due to inclusion of L1 cache. State-destroying leakage control methods are rarely used because data reloading from off chip memory is time-consuming and energy-consuming. LRU-assist or other time-based DVS strategy are also impractical because the energy consumption of large number of additional counters can offset the leakage energy saving. Low overhead state-preserving strategy should be developed.

Always Drowsy Speculatively Recover

As access frequency of L2 cache is very low, all L2 cache lines can be put into drowsy state. Currently the tags are also put into drowsy mode along with the data for aggressive power saving. When a cache line is accessed, supply voltage of the whole set it located must be recovered to the normal level firstly for tag comparison, which costs several cycles and impacts the processor performance. L2 cache accesses can be partially predicted using prefetch-like mechanism, such as next-line prediction or stride-based prediction (S. Sair et al. 2002). We call this leakage control mechanism ADSR (Always Drowsy Speculatively Recover). Whenever an access to L2 cache line occurs, synchronized with tag compare and data read (wake up first if this objective line is drowsy), speculatively recover the whole set where the predicted line locates. If the next access exactly indexes this recovered set, recover latency is saved and performance is protected whether the tag comparison is match or not. Note that predicted set is only recovered to normal state instead of fetched from off-chip memory for dynamic energy saving. Each set is drowsed back as soon as the access is finished. Also the whole speculatively recovered set is drowsed back when the prediction is proved wrong by the next access.

Results

We implemented ADSR based on next-line prediction and stride-based prediction, as well as time-based drowsy. Two-cycle recover latency is first evaluated and drowsy threshold is set to 32k cycles. Thirteen SPEC CPU2000 benchmarks are simulated. Off ratio, IPC and EDP results are gathered and illustrated in figure 6, figure 7 and figure 8. We can see that ADSR can achieve more aggressive leakage power reduction than drowsy, and comparative performance (sometimes even better). The energy efficiency is also satisfying, which is better than drowsy and approximately 50% of the conventional L2 cache without any leakage control method. Though sometimes the IPC of ADSR is lower than time-based method, when the recover latency rises to 4-cycle, the IPC of time-based strategy drops obviously relative to ADSR (figure 9). So ADSR is more robust. We have also evaluated the recover accuracy of ADSR, and the result is not so exciting (only about 15% on average). It is suggested that the next-line and stride-based prediction does not work very well. Evaluating other pre-recovering mechanism and devising more accurate speculation such as compiler-based prediction are our plans in the near future work.

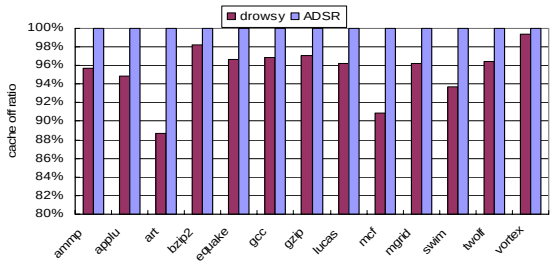

Figure 6: L2 Cache Off Ratio Using ADSR Algorithm

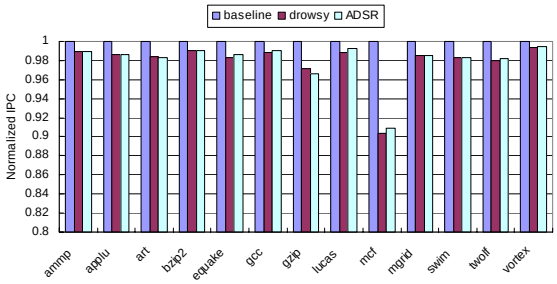

Figure 7: Normalized IPC Using ADSR Algorithm

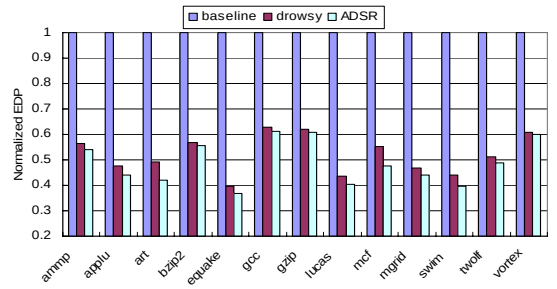

Figure 8: Normalized EDP Using ADSR Algorithm

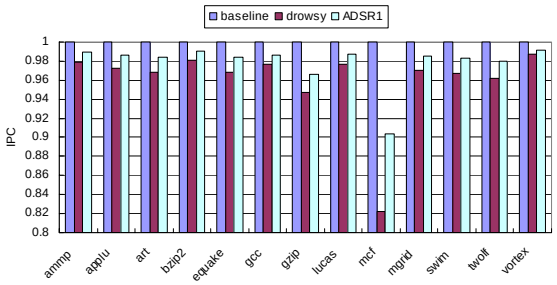

Figure 9: Normalized IPC Using ADSR with 4-cycle Recover Latency

CONCLUSION AND FUTURE WORK

This work presents two improved approaches, LRU-assist decay and ADSR, to respectively control L1 and L2 cache leakage energy consumption. Evaluation results show that they all work well in reducing leakage energy consumption and improving energy efficiency, with little hardware cost and without drastic performance decrease. These two methods can also be used simultaneously in the same memory system to control leakage power of the whole cache hierarchies. Current LRU-assist algorithm still needs counters to adaptively control the way mask. These counters consume considerable dynamic and static energy. A more saving LRU-assist mechanism will be developed in our future work. As mentioned above, evaluating other pre-recovering mechanism and devising more accurate speculation such as compiler-based prediction are also our plans.

ACKNOWLEDGEMENT

This work was supported by NSFC (No. 90207011 and No. 60273069) in China.

REFERENCES

D. Brooks, V. Tiwari, and M. Martonosi. 2000. “Wattch: A Framework for Architectural-level Power Analysis and Optimization.” *In International Symposium on Computer Architecture*, 83-94.

D. Burger and T. Austin. 1997. “The SimpleScalar Tool Set, version 2.0.” *Computer Architecture News*, 25(3):13-25, June.

K. Flautner, N.S.Kim, S.Martin, D.Blaauw, and T.Mudge. 2002. “Drowsy Caches: Simple Techniques for Reducing Leakage Power.” *In International Symposium on Computer Architecture*, 147-157.

K. Nii, H. Makino et al. 1998. “A low power SRAM using auto-backgate-controlled MT-CMOS.” *In International Symposium on Low Power Electronics and Design*, 293-298.

L. Li, I. Kadayif, Y.-F. Tsai, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, and A. Sivasubramaniam. 2002. “Leakage energy management in cache hierarchies,” *In the Eleventh International Conference on Parallel Architectures and Compilation Techniques,* 131–140.

M. D. Powell et al. 2002. “Gated-Vdd: A Circuit Technique to Reduce Leakage in Deep-Submicron Cache Memories.” *In International Symposium on Low Power Electronics and Design*, 90-95.

N.H.E. Weste. 2005. *CMOS VLSI Design: A Circuits and Systems Perspective, 3rd Edition.* Pearson Education.

P. Shivakumar and N. P. Jouppi. 2001. “Cacti 3.0: An Integrated Cache Timing, Power, and Area Model.” Technical Report WRL-2001-2, HP Labs Technical Reports.

R. Kessler. 1999. “The Alpha 21264 Microprocessor”. *In Annual International Symposium on Microarchitecture*, 24-36.

S. Kaxiras, Z. Hu and M. Martonosi. 2001. “Cache Decay: Exploiting Generational Behavior to Reduce Cache Leakage Power.” *In International Symposium on Computer Architecture*, 240-251.

S. Sair, T. Sherwood, and B. Calder. 2002. “Quantifying load stream behavior.” The 8th International Symposium on High-Performance Computer Architecture, 197-210.

SIA. 2004. *International Technology Roadmap for Semiconductors*, Update.

T Sherwood et al. 2002. “Automatically characterizing large scale program behavior.” *In Proceedings of the Tenth International Conference on Architectural Support for Programming Languages and Operating Systems,* San Jose, USA.

T. Pering, T. Burd and R. Brodersen. 1998. “The Simulation and Evaluation of Dynamic Voltage Scaling Algorithms.” *In International Symposium on Low Power Electronics and Design*, 76-81.

Y. Zhang, D. Parikh, K. Sankaranarayanan, K. Skadron and M. R. Stan. 2003. “Hotleakage: An Architectural, Temperature-aware Model of Subthreshold and Gate Leakage.” Tech. Report CS-2003-05, Department of Computer Sciences, University of Virginia.

AUTHOR BIOGRAPHIES

ZHANG CHENGYI was born in Hebei, China and went to the National University of Defense Technology, where he studied computer science and obtained his bachelor’s degree in 2000. Now he is a doctor candidate in School

of Computer Science, National University of Defense Technology. His main research interests are high-performance and low power microprocessor design. His e-mail address is: chengyizhang@nudt.edu.cn

ZHANG MINXUAN was born in Hunan, China and is a professor of National University of Defense Technology now. His main research interests are high performance computer architecture and VLSI design. His e-mail address is: mxzhang@nudt.edu.cn

XING ZUOCHENG was born in Anhui, China and is a professor of School of Computer Science, National University of Defense Technology now. His main research interests are high performance and low power microprocessor design. His e-mail address is: zcxing@nudt.edu.cn