Vol.2009-ARC-184 No.13
2009/8/4

Leakage Efficient TLB Design for Embedded Processors

Zhao Lei,[†1] Hui Xu,[†1] Daisuke Ikebchi,[†1]
Toshiaki Kamata,[†1] Mitaro Namiki[†2]
and Hideharu Amano[†1]

This paper presents a leakage-efficient TLB (Translation Lookaside Buffer) design for embedded processors. By utilizing the spatial and temporary locality of TLB references, the unused entries or even the whole TLB can be put into a low power mode by dynamic voltage scaling technique. According to their different accessing patterns, two leakage control mechanisms are proposed for instruction TLB and data TLB respectively. Evaluation results with six MiBench programs show that the proposed design can reduce 50% and 35% leakage power of instruction TLB and data TLB on average, with only 0.01% performance degradation.

1. Introduction

In battery-driven embedded systems, power usually dominates over performance as the primary design concern, due to its impact on system density, operation reliability, battery life and integration costs. As process geometries shrink, leakage power increases exponentially, and is expected to become the dominant source of power consumption[1]. Hence, suppressing leakage is now a critical challenge facing all embedded processor designers.

Translation Lookaside Buffer (TLB) is an important component even in embedded processors. It provides storage attributes, access permissions and virtual to physical memory address translation to efficiently address huge amounts of physical memory. To avoid the performance degradation caused by TLB misses, modern processors usually adopt large size TLBs with high associative structure, which lead to a non-trivial energy dissipation of both dynamic and leakage consequently. Plenty of research has been devoted to exploring the dynamic power reducing mechanisms on TLBs[2][3][4], by either reducing the energy dissipation per access or the total number of accesses. However, when the leakage power has emerged a limiting factor in embedded processor design, power optimized TLB designs only addressing on dynamic power become insufficient.

Geyser-0[5] is a MIPS R3000 based prototype chips which was designed and implemented with 90nm technology. By utilizing the fine-grain runtime power gating technique, the function unit of the processor core, such as ALU, shifter, multiplier and divider can be dynamic put into low power mode and turned back. To remove the TLB from the critical path of the processor, caches in Geyser-0 are implemented in a virtual-index and physical-tag style, and the virtual address comparison and cache tag accesses are overlapped. To support such architecture, a 16-entry, two port-in and two port-out TLB are implemented by flip-flops (FFs). The evaluation results show the TLB consumes as much as 29% leakage power of the processor core, which is already a significant fraction of the total leakage budget. Further, technology improvement and feature size reduction will exacerbate such an issue in future embedded processor design.

In this paper we focus on reducing the leakage power of TLBs. Although TLB has a cache-like structure, it will be too bold to directly copy low leakage cache designs such as cache decay Kaxiras[6] and drowsy cache Kim[7] into leakage-efficient TLB design, due to their different access pattern, replace policy and miss recover penalty. Careless design will introduce unacceptable overheads of both performance and power consumption to a leakage-efficient TLB, and may even increase the total power consumption instead of saving.

To address this problem, a leakage-efficient TLB design is proposed in this paper. By utilizing the Dynamic Voltage Scaling (DVS) technique, TLB entries or even the whole TLB can be put into low leakage mode to reduce leakage power consumption. Two different leakage control mechanisms with different control granularity are proposed on instruction TLB (iTLB) and data TLB (dTLB) respectively, according their access patterns. The evaluation results show our design can reduce more than 50% and 35% leakage power of the iTLB and dTLB, with a negligible performance degradation.

The remainder of this paper is organized as follows. The next section presents the experimental platform for this work. In Section 3, the basic design philosophy and detailed leakage control mechanisms will be illustrated. Leakage saving results will be shown in Section 4, and Section 5 is for conclusion.

†1 Graduate School of Science and Technology, Keio University
†2 Department of Computer and Information Sciences,Tokyo University of Agriculture and Technology

© 2009 Information Processing Society of Japan

2. Experimental Setup

In this paper, trace-driven simulations are executed to capture necessary data for the purpose of evaluation and analysis. We choose the MIPS system emulator from QEMU[8] as the experimental platform. Since TLBs have a tighter connection with OS than other components in a processor, to better emulate the real working state of a TLB, OS (Debian in this paper) and application programs are running on the platform simultaneously. The configuration parameters are shown in Table1. A group of authors[9] modified the basic structure of QEMU, so it can be used to trace memory accesses and executed instructions. We traced all the memory accessing information by executing six application programs of different fields from MiBench[10]. When the application programs are executed, several OS related processes are running on the background, and there are also some basic processes, like ash, are running with application programs simultaneously.

Table 1 Configuration Parameters

Trace Environment	
CPU Type	MIPS R4000
Instruction Execution	In-order
OS Type	Debian
Kernel	Linux 2.6.15
Shell	ash
Compiler	GCC(4.2.2)

The power evaluation in this paper is based on the post-layout simulation. An actual layout of a 16-entry TLB is generated with Synopsys EDA tools (Design Compiler and Astro)[11]. Our design obeys the specification of MIPS R3000 processors[12], which employs 64 bits entry structure and is designed to cooperate with 4KB virtual-index physical-tag caches. Since the post-layout simulation is slow, emulating all memory reference information on the designed TLB will be desperately time costing. In this paper, we just simulate a small piece of the traced information, and treat the power consumption of this small segment as the average power of the whole application programs. Then, this value will be used to calculate the final power reduction effects, combined with the utilization ratio of logic ingredients. This method maybe not as accurate as the traditional way, but it can correctly reflect the overall power reduction effect of our design. The value of both dynamic power and leakage power used in this paper are obtained by PowerCompiler, while the leakage reduction ratio comes from the simulation result by HSIM[11].

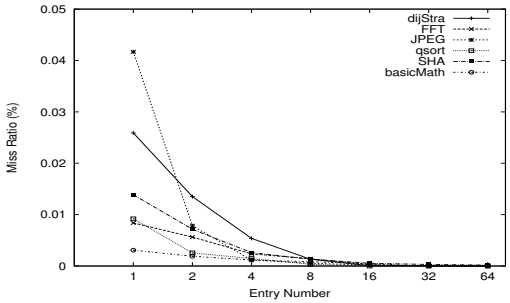
3. Design Philosophy

Leakage-efficient design based on the assumption that a certain fraction of the design target can be put into low leakage mode and instated to normal mode without significantly degrading the performance. The final leakage reduction effects rely on the scale of the leakage reduction target, the time stayed in low leakage mode, and the mode transition frequency. The TLB is one of most active components in embedded processors with a high utilization, which do not leave much space for leakage cutting. Fortunately, both iTLB and dTLB references exhibit a good spatial and temporal locality, which can be further exploited to improve leakage efficiency. Generally speaking, TLB references are evenly distributed on each entry. However, if the overall running time can be divided into small time slices, we can find that in a specific slice, TLB references only center on a small subset of entries. This observation provides an opportunity to cut leakage power by detecting the unnecessary TLB accesses and put these profitless entries into low leakage mode. The above observation inspires this work.

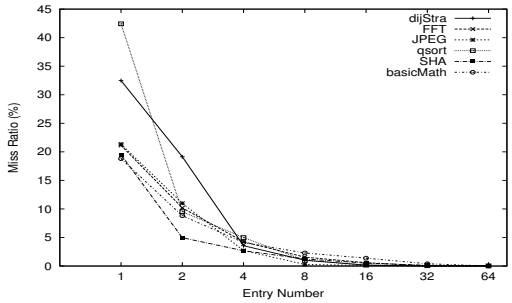
3.1 TLB Access Pattern Analysis

Fig.1(a) and Fig.1(b) show the miss ratio of iTLB and dTLB with different entry numbers. The miss ratio can be used as a metric of locality. Results presented in Fig.1(a) and Fig.1(b) show when increasing the TLB entry number from 1 to 64, the miss ratio of iTLB and dTLB will continually decrease. When choosing a 16-entry configuration, both iTLB and dTLB miss ratio can be reduced to a reasonable level. Note that entry number increases at low end of the x-axis will have much more miss ratio improvements than at high end of x-axis, for example, increase the entry number from 1 to 2 can reduce the miss ratio 25 times as much as changing TLB entry number from 32 to 64 for application program “basic math”. Both Fig.1(a) and Fig.1(b) show the good locality of both iTLB and dTLB, but we can see that the spatial locality of iTLB is much better than dTLB. For instance, a 2-entry iTLB can achieve a limpingly acceptable miss ratio, while same sized dTLB will caused a disaster. Another metric, Sequential Page Access Rate (SPAR) which means the ratio of numbers of sequential TLB accesses that hit on the same page to the total memory references number, is shown in Fig.1(c) to further analysis the difference of access pattern between iTLB and dTLB. While both iTLB and dTLB

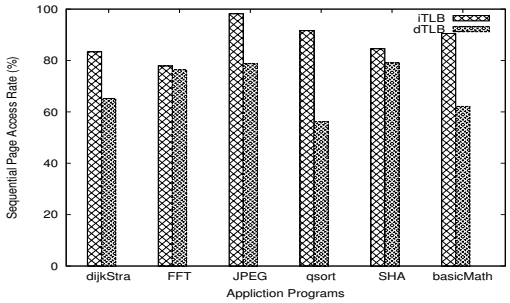
© 2009 Information Processing Society of Japan



(a) iTLB Miss Ratio



(b) dTLB Miss Ratio



(c) Sequential Page Access Rate

Fig. 1 TLB Access Pattern Analysis

present a good possibility that sequential TLB accesses will hit on the same page, the SPAR of dTLB is relative low, and this makes a request of different design methodology on the iTLB and the dTLB.

3.2 iTLB Leakage Control Mechanism

The iTLB leakage control mechanism comes from the observation that there is a considerable spatial locality in instruction streams. As was mentioned earlier, the average SPAR of iTLB is more than 80% of total instruction references, which means when a program enters a physical page, following instructions tend to be fetched from the same page for a certain time before it moves or jumps to another page. If the page-entering point and page-leaving point can be detected, accesses on other TLB entries will be unnecessary, and leakage reduction can be achieved by putting these entries into a low leakage mode. However, accurate page-entering and page-leaving point detection is hardly to be obtained at run-time. In our design, an alternative mechanism is proposed to approximate the ideal situation.

1-RAR solution: A Recently Accessed Register (RAR), which contains the latest address translation information, is inserted between the processor core and iTLB to filter out unnecessary TLB accesses. The RAR has the same structure as a one-entry TLB holding 64 bits for virtual page number(VPN), physical frame number(PFN), ASID, flag bits and preserved 14 bits. When the program enters a physical page, we save the page information into the RAR. If instructions are kept to be fetched from the same page, the RAR can be kept using without going to the iTLB. After a certain time, we can assume that such a same-page-hitting behavior will last, and the whole iTLB can be put into low leakage mode to save power. The working process of proposed mechanism works as follows: The virtual addresses generated by PC are compared with the virtual page number of the RAR. If they match, then the target instruction is in the same page as the previous one and the physical frame number is sent to L1 cache for tag comparison. If they do not match, then we force an iTLB lookup. If the iTLB is in the low leakage mode, there needs a way to turn it back in the active mode, then the whole iTLB will be accessed based on the virtual page number. If a hit happens, the target entry will be moved to the RAR. In case of TLB misses, the ordinary TLB miss process will be aroused and one iTLB entry will be updated. Because the RAR is implemented by registers, the RAR comparison can be executed as soon as the PC is updated. When a RAR miss happens, iTLB wake-up signal can be sent immediately, and only one clock cycle penalty is induced for a RAR-miss but iTLB-hit instruction reference (details will be discussed in the end of this section).

The time between the first RAR hit and the point when the iTLB is put into low leakage mode is also an important concern, and we refer it as warming-up time. If this time is too short, the low leakage mode transfer may be triggered by temporary references, and the iTLB can only stay in low leakage mode for a very short time, therefore, overheads of both performance and power will be induced. While a too long warming-up time will degrade the opportunity to put iTLB into low leakage mode. In next section, experimental results on different warming-up time will be presented.

© 2009 Information Processing Society of Japan

2-RARs Solution: A potential shortcoming of the 1-RAR solution is its awkward performance when the application program has loops crossing the page boundary. Under such circumstances, the contents of RAR will be kept updating, and the iTLB itself may be waged between the low leakage mode and the active mode constantly. Hence, significant degradation on both performance and power reduction effect will be caused. A simple solution is to use 2 RARs instead of one. The working process of the 2-RARs solution is identical to the 1-RAR solution except the RAR updating policy. In our design, when a RAR updating happens, the latest accessed RAR is kept while the other one is replaced.

Concatenation Solution: 2-RAR solution induces extra leakage power because of a second 64 bits register. By utilizing the preserved bits in RAR, we propose a concatenation solution which can approximately achieve the 2-RARs performance while using only one recent accessed register. The virtual page number and the physical frame number are served as a base address, while the preserved bits are served as address offsets. The preserved 14 bits are divided in 2 parts: 6 bits for virtual frame number ($Offset_{vpn}$) and 8 bits for physical frame number ($Offset_{pfn}$).The second address can be calculated by following expression:

$VPN_{sec} = \{VPN[19:6], Offset_{VPN}\}, PFN_{sec} = \{PFN[19:8], Offset_{PFN}\}$

When the distance between a new virtual page number and previous one is less than 2^6 page size, and the physical frame number distance is less than 2^8 page size, these address offsets can be used as the second RAR.

With the concatenation solution, every instruction reference will invoke the concatenation operation to complete the second address calculation and comparison. That will be too expensive. We can turn to the compiler to ease such a burden. The compiler will insert a special bit to tell the RAR explicitly whether the previous fetched instruction is a branch instruction or not, and the second address will not be calculated if the current instruction reference is not following a branch instruction. The RAR update process of concatenation solution is also different. Here, we classify page crossing cases into two categories: a) two successive instructions which are on the page boundary (we call this the WALKING case); and b) branch instructions whose target address are on other page (referred as the JUMPING case). Since the base address updating causes a recalculation of address offsets, in this design, only WALKING or iTLB miss can trigger the base address updating, otherwise only the address offsets are updated.

3.3 dTLB Leakage Control Mechanism

The spatial locality of dTLB is not as good as iTLB, and using the same leakage control mechanism on both will be insufficient. The dTLB leakage control mechanism comes from the observation that during a small time slice, the data references only center on a subset of the TLB entries. By putting unnecessary entries into low leakage mode, leakage reduction effect can be achieved.

One straight forward mechanism is to put all TLB entries into the low leakage mode periodically, regardless of access patterns. A dTLB entry is woken up only when it is accessed again. Such a mechanism needs an extra global counter and the mode control circuit must be implemented on an each entry granularity. The dTLB leakage control mechanism works in a 3-step fashion. First, the virtual page number of a data reference is compared with the virtual part in active TLB entries; a trace register is used here to keep tracking the active TLB entries. If the virtual page number hits on active entries, the corresponding physical frame number is sent to L1 cache without performance penalties. We refer such a case as active-hit and the corresponding miss as active-miss. When an active-miss happens, we first wake up the VPN, ASID and Flag bits of all entries, which are referred as TLB-TAG, and then this TLB-TAG part will be accessed. If a hit happens, the corresponding dTLB entry is fully woken up. In our design, such a case causes 3 clock cycles performance penalty. If a dTLB miss happens, the ordinary TLB miss handle process is called, the whole dTLB wake-up process can be overlapped, and only 1 clock cycle penalties are induced here.

TLB-TAG: The TLB-TAG accounts for a significant portion of the total size of a TLB entry. If the TLB-TAG of all low leakage entries are woken up when an active-miss happens, and such an active state is kept until the next slice, the leakage reduction opportunity will be degrade significantly. Here, the TLB-TAG and its periphery comparison circuits are designed capable of being accessed in low leakage, without having to be turned back to the active mode first. Working in low power mode will increase the circuit transition time, and therefore decrease the performing speed. However, as shown in Fig.2, an active-miss will stall the processor, and the path length of the TLB-TAG access is shorter than that of the common TLB accesses. By choosing a proper supply voltage, the low power accessible design will not bring in any frequency degradation. Further, the penalty of an active miss can be reduced to 2 clock cycles.

Fast-wake-up: If the spatial locality of a program segment is really bad, considerable performance and power overheads will be induced. A fast-wake-up policy is proposed to increase the immunity to bad locality program segments. If the number of active misses reaches a preset threshold (referred as fast-wake-up

© 2009 Information Processing Society of Japan

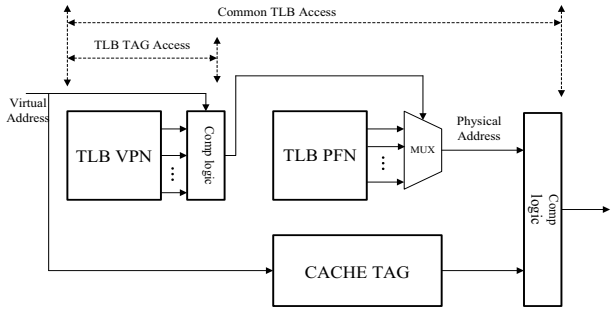


Fig. 2 TLB-TAG Path

threshold) in a specific time slice, the program segment executed during this slice will be recognized as a bad locality segment and the whole dTLB will be woken up to eliminated the penalty caused by staying low power mode.

Correlation between Time Slices: Another design concern is the correlation between sequential time slices. If the data references happen in this slice is highly depend on these of past one or several slices, then, keeping previous active TLB entries in a new time slice will be helpful.

Time Slice Length: The time slice length means how often TLB entries can be put into low leakage mode. Short time slice length induces high frequent mode transfers, therefore, the mode transfer penalty. While long time slice will increase the opportunity of a full active dTLB. The detailed discussion of the impact of fast-wake-up threshold, correlation between time slice and time slice length on performance and leakage reduction effect will be presented in next section.

3.4 Hardware Support

Leakage-efficient design needs the support from circuit level. Circuit-level leakage reduction technique suitable for proposed mechanisms should satisfy two requests: the state of circuits should be kept when in low leakage mode; and the mode transfer penalty should be small. In this paper, the Dynamic Voltage Scaling (DVS) technique was integrated into TLB design to reduce the leakage power of both the TLB entries and their periphery comparison logics. While voltage scaling technique has by wildly used for dynamic power reduction, short channel effects also make it very effective for leakage reduction[7]. When TLB entries are predicted unnecessary to be accessed for a near future, they can be put into a low power supply mode or drowsy mode. By fine tuning the supply voltage in drowsy mode, data stored in TLB entries can be preserved.

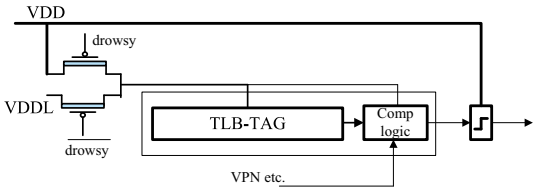


Fig. 4 TLB-TAG Entry Schematic

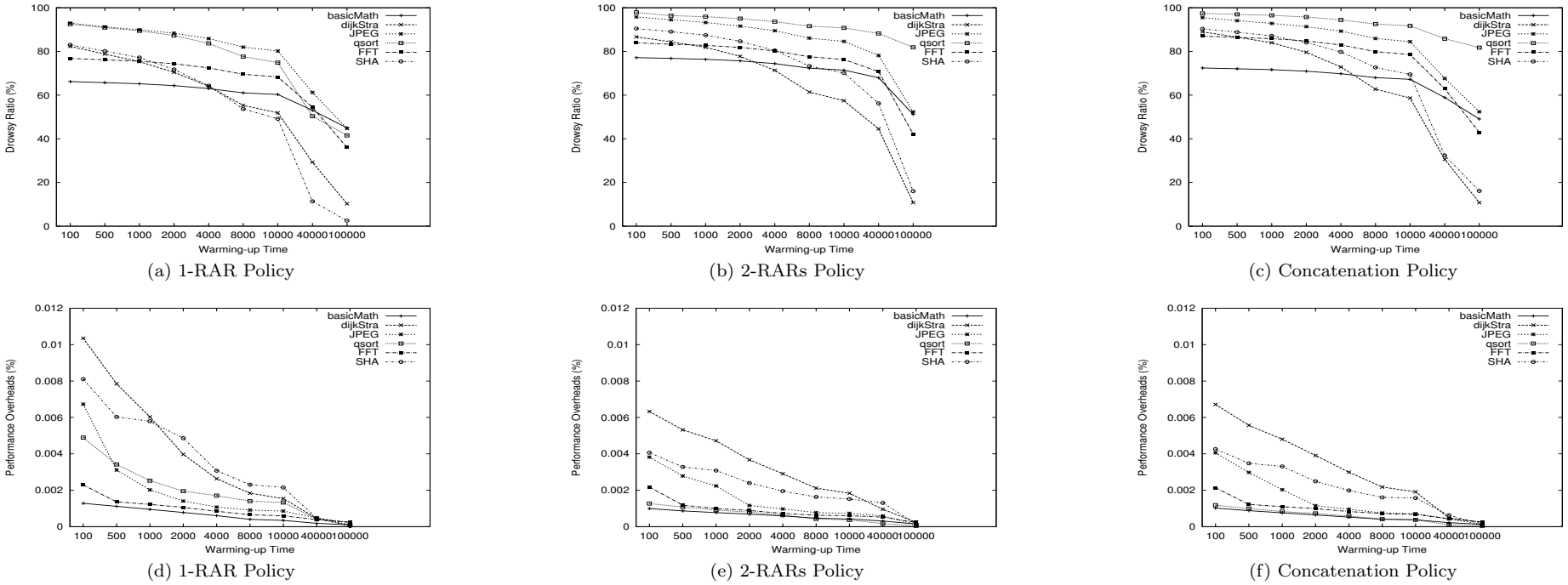
A dual supply network is also used to provide fast switching between supply voltages. Header PMOS transistors with complementary control signals are used to select between two different supply voltages. Note that, all components working in drowsy mode, except the TLB-TAG part, do not allow to be accessed until turned back to the normal voltage. According to the circuit level simulation result, the mode transfer time for a 16-entry TLB is less than 1ns. Here we assume the mode transfer will induce one clock cycle penalty. Fig.4 shows the structure of a TLB-TAG entry. Since TLB-TAG can be accessed in drowsy mode, a level shifter is appended after comparison logics. Because voltage scaling will degrade the operating speed, the VDDL must be carefully selected. Eq.1 shows the relationship between operating frequency and supply voltage, where, V_{norm} and F_{norm} are operating voltage and frequency which are normalized to the maximum voltage V_{max} and frequency F_{max}.

$$V_{norm} = V_{th}/V_{max} + (1 - V_{th}/V_{max}) * F_{norm}, Eq.1$$

In this paper, we choose the 0.9v as the lower voltage of dTLB entries, which means a 40% speed-down. As was mentioned in last subsection the slower operating does not affect the overall frequency because of the shorter path. Simulation results have confirmed such an assumption.

The soft error rate of FFs[13] is also a concern in very deep sub-micrometer technology. Since the soft error rate increases as supply voltage scaled, to alleviate the side effect of voltage scaling, a rather conserved lower voltage: 0.8v, was selected for the lower voltage supply for iTLB. Detailed power values of proposed mechanism are shown in Tab2.

© 2009 Information Processing Society of Japan



(a) 1-RAR Policy (b) 2-RARs Policy (c) Concatenation Policy

(d) 1-RAR Policy (e) 2-RARs Policy (f) Concatenation Policy

Fig. 3 iTLB Drowsy Ratio & Performance Overheads Comparison

Table 2 Power Parameters

Leakage	
Normal 16-entry	37.8μW
Drowsy 16-entry (0.8v)	9.9μW
Normal 1-entry	3.3μW
Drowsy 1-entry (0.9v)	1.4μW
Dynamic	
16-entry	688.6μW
RAR	14.1μW
Concatenation	17.4μW

4. Evaluation Results

The leakage reduction effect of proposed mechanisms will be evaluated here. The drowsy ratio, which means the ratio of the time when the leakage saving target staying in the drowsy mode to the whole execution time, has a direct impact on final power saving results, so it is employed as key metric to measure the leakage saving efficiency. In this section the drowsy ratio will be combined with performance overheads to measure the impact of different parameters or polices, which was mentioned in last section, on the design efficiency. Power evaluation models are also proposed. After selecting suitable design parameters, the final leakage reduction effects are presented. Note that, proposed mechanism can be easily implanted to low dynamic power design, in the end of each subsection, an approximate dynamic reduction effect is also presented.

4.1 iTLB Evaluation

Fig.3(a)-Fig.3(f) show the drowsy ratio and performance overheads of 1-RAR,

© 2009 Information Processing Society of Japan

2-RAR and concatenation policies varying with warming-up time from 100 clock cycles to 100000 clock cycles. As shown in these figures, the drowsy ratio will keep decreasing as the warming-up time increase, and performance overheads show a similar trend. It worth noting that before 8000 clock cycles the decreasing speed of drowsy ratio is not as much as that of performance overheads, and we find that 4000 clock cycles may be a suitable trade-off between drowsy ratio and performance overheads. The drowsy ratio of the 2-RARs policy is much better than the 1-RAR one because the 1-RAR is suffered from page-crossing loops. Since the design efficiency of concatenation policy is highly dependent on the memory allocation strategy of OS, the comparison of 2-RARs and concatenation policy is sort of tricky. With our experimental environment, the 2-RARs policy is 1%-5% ahead of the concatenation policy according to different application programs.

Power reduction effects are calculated with the evaluation methodology mentioned in Section 2. Leakage and dynamic evaluation model can be expressed as following:

$$L_{new} = L_{filter} + (1 - P_{Drowsy}) * L_{iTLB} + P_{Drowsy} * L_{iTLBL} + L_{counter}, (1)$$

$$D_{new} = D_{filter} + (1 - P_{Drowsy}) * D_{iTLB} + D_{counter} + D_{transition}, (2)$$

Where, L_{new} is the final leakage power of our design; L_{filter} is the leakage power the iTLB accesses filtering components, which can be one RAR, two RARs, or the concatenation register; P_{Drowsy} is the percentage when iTLB staying in drowsy mode; L_{iTLB} and L_{iTLBL} are leakage power of iTLB in active mode and drowsy mode. Since the warming-up time is selected as 4000 clock cycle, a 12-bit global counter is necessary in our design, and $L_{counter}$ is the leakage power of such a counter. In expression(2), D_{new}, D_{filter}, D_{iTLB} and D_{count} are the dynamic power counterpart of expression(1), while the $D_{transition}$ is the dynamic power consumed during the mode transition.

Fig.5 and Fig.6 show final leakage reduction effects of iTLB with a 4000 clock cycles warming-up time. The dynamic power and leakage power of a 4000-clock-cycle-counter is 3.4μW and 0.308μW. The energy dissipation for mode transfer is around 3e-19J, it only do a negligible contribution to the total dynamic power. Here, we omit the impact of mode transition energy when doing power calculations.

As shown in Fig.5 and Fig.6, the concatenation policy has the best leakage reduction efficiency, while the 2-RARs policy can save dynamic power best. If the spatial locality of an application program is really good, the 1-RAR policy may have a better performance than 2-RARs in terms of leakage saving because of the extra leakage induced by a second 64bits register. With our iTLB leakage control mechanism, average 50% leakage power and 70% dynamic power can be reduced with only 0.01% performance degradation.

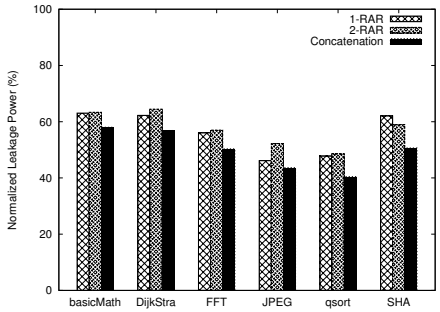


Fig. 5 Normalized Leakage Power Consumption

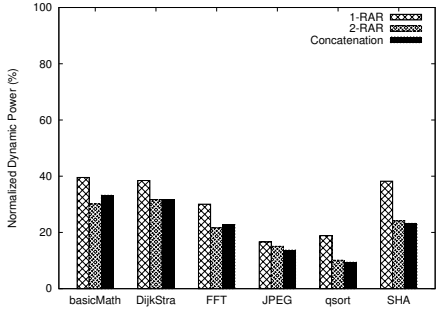
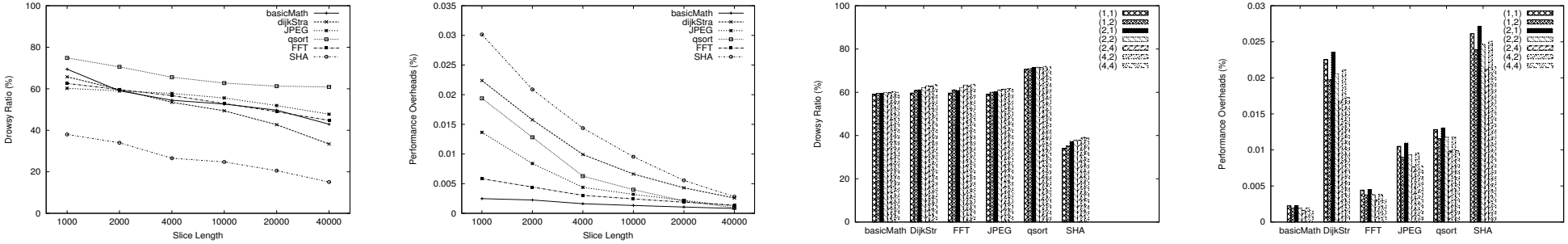


Fig. 6 Normalized Dynamic Power Consumption

4.2 dTLB Evaluation

In Section 3, we discussed three factors that may affect the design efficiency: fast-wake-up, correlation between time slices and the slice length. In this subsection, evaluation results on how these factors can impact the final leakage reduction effects are presented. Here, the impact of correlation between time slices is measured by the history length, which means the active state of history length slices before should be kept. Fig.7(a) and Fig.7(b) show the drowsy ratio and performance overheads of a dTLB by varying slice length without considering fast-wake-up or

© 2009 Information Processing Society of Japan



(a) Drowsy Ratio: Slice Length (b) Performance Overheads: Slice Length (c) Drowsy Ratio: HL & FT (d) Performance Overheads: HL & FT

Fig. 7 dTLB Drowsy Ratio & Performance Overheads Analysis

correlation between slices. Although the drowsy ratio and performance overheads change drastically from program to program, a good trade-off also can be obtained by selecting 4000 clock cycles slice length. Fig.7(c) and Fig.7(d) show the drowsy ratio and performance overheads by varying the history length(HL), fast-wake-up threshold(FT) pair, and we choose the (2,4) pair as the configuration in our leakage-efficient dTLB design. Note that, the drowsy ratio here is measured on an each-line granularity, which equals to the ratio of summary each line drowsy time to the production of execution time and TLB entry number.

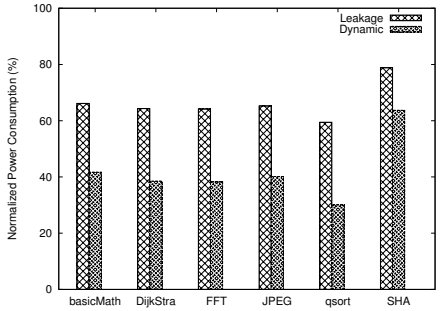


Fig. 8 Normalized Power Consumption: Leakage & Dynamic

The power evaluation of dTLB is more complex than iTLB because of the 3-step fashion design philosophy. To simplify the question, the dynamic power consumed by drowsy TLB-TAG accesses is treated the same as full active TLB accesses. The power evaluation mode is presented as follows:

$$L_{new} = (1 - P_{Drowsy}) * L_{entry} * 16 + L_{counter} + P_{Drowsy} * L_{entry_L} * 16, \quad (3)$$

$$D_{new} = (1 - P_{Drowsy} + P_{D\_a}) * D_{entry} * 16 + D_{counter} + D_{transition}, \quad (4)$$

Here, the L_{entry} and L_{entryL} mean the leakage power of a single TLB entry, when working in active mode and drowsy mode, and the D_{entry} is the average dynamic power of a TLB. P_{D_a} means the percentage of drowsy accesses to total memory accesses, it is used to approximate the dynamic power consumption of drowsy accesses. As was mentioned earlier the L_{entryL} is measured by 0.9v voltage supply.

Fig.8 gives the normalized power of proposed dTLB on both leakage and dynamic. From the evaluation result, more around 35% leakage power and 60% dynamic power can be saved by proposed mechanism with performance degradations less than 0.01%

5. Conclusions

Mechanisms for controlling the leakage power of TLB have been described. With the dynamic voltage scaling technique, a part or the whole TLB can be put into drowsy mode to cut leakage. By filtering out unnecessary TLB accesses, leakage power can be significant reduced. Proposed leakage control mechanisms also can be employed to reduce dynamic, with the help of clock gating technique. Evaluation result shows, for instruction TLB 50% leakag power and 70% dynamic power can be

© 2009 Information Processing Society of Japan

reduced, and more than 35% leakage power and 60% dynamic power can be saved for data TLB. Proposed mechanisms only induce 0.01% performance degradation.

Acknowledgments
The authors would like to thank VLSI Design and Education Center (VDEC), Synopsys, Cadence, STARC, and Japan Science and Technology Agency (JST) CREST for their support.

References

1) ITRS: Int'l Technology Roadmap for Semiconductor, http://public.itrs.net (2001).
2) Clark, L., Choi, B. and Wilkerson, M.: Reducing translation lookaside buffer active power, *Proceedings of the 2003 international symposium on Low power electronics and design*, pp.10–13 (2003).
3) Chang, Y.: An ultra low-power TLB design, *Proceedings of the conference on Design, automation and test in Europe: Proceedings*, pp.1122–1127 (2006).
4) Kadayif, I., Sivasubramaniam, A., Kandemir, M., Kandiraju, G. and Chen, G.: Optimizing instruction TLB energy using software and hardware techniques, *ACM Trans. Des. Autom. Electron. Syst.*, Vol.10, No.2, pp.229–257 (2005).
5) Seki, N. and et.al.: A fine-grain dynamic sleep control scheme in MIPS R3000, *ICCD 2008* (2008).
6) Kaxiras, S., Hu, Z. and Martonosi, M.: Cache decay: exploiting generational behavior to reduce cache leakage power, *Proceedings of the 28th annual international symposium on Computer architecture*, pp.240–251 (2001).
7) Kim, N., Flautner, K., Blaauw, D. and Mudge, T.: Circuit and microarchitectural techniques for reducing cache leakage power, *IEEE Transactions on Very Large Scale Integration(VLSI) Systems*, Vol.12, No.2, pp.167–184 (2004).
8) QEMU: http://fabrice.bellard.free.fr/qemu.
9) KazuyaMatsuo, M.S. and Namiki, M.: Development of System Evaluation Environment using QEMU for Low Power System, pp.61–68 (2007).
10) Guthaus M.R., Ringenberg J.S., e.: MiBench: A free, commercially representative embedded benchmark suite, *Workload Characterization, 2001. WWC-4. 2001 IEEE International Workshop on*, pp.3–14 (2001).
11) SYNOPSYS: http://www.synopsys.com.
12) Sweetman, D.: *See MIPS Run*, Morgan Kaufmann (2006).
13) Ramanarayanan, R., Degalahal, V., Vijaykrishnan, N., Irwin, M. and Duarte, D.: Analysis of soft error rate in flip-flops and scannable latches, pp.231–234 (2003).

© 2009 Information Processing Society of Japan