Analysis of Functional Unit Power Gating in Embedded Processors

Houman Homayoun and Amirali Baniasadi
ECE Department, University of Victoria, Victoria, Canada
{homayoun, amirali}@ece.uvic.ca

ABSTRACT

In this work we report leakage energy savings potentially achieveable by using power gating in embedded processors.

We investigate idle time distribution in embedded processors and show that there is an opportunity to apply power gating in the embedded space. Moreover, we show that a previously suggested simple dynamic idle period detection technique can reduce leakage power considerably while maintaining performance.

We also analyze how variations in design and circuit parameters impact performance and power savings.

We evaluate power gating for a representative subset of MiBench benchmarks and for a processor using a configuration similar to Intel's XScale processor.

1. INTRODUCTION

Developing power-efficient hardware has been an active research area in high-performance processor design for many years. However, in recent years, we have witnessed a rapid complexity increase in embedded processors. As a result, embedded processors' power dissipation has become one of the major barriers in their deployment in mobile devices.

Many embedded processors such as Intel's XScale rely on executing instructions in program order (in-order). In these processors, where performance is less of a concern, power dissipation is the most critical design constrain.

Moreover, as the semiconductor technology scales down, leakage (standby) power will start to account for an increasing share of processor power dissipation [1,9]. Leakage power may become a more serious issue in embedded processors where applications may require long periods of inactivity [12,13]. Previous study has introduced many techniques to reduce leakage in different processor units (*e.g.,* [3, 4, 5, 6]).

In most processors, including embedded processors, computational units' power dissipation accounts for a considerable share of total power dissipation. However, and as we show in this work, in a high-performance embedded processor, computational units may be idle for long periods of time depending on the application's required resources. As a result, execution units may consume energy without contributing to performance during such long idle periods.

One way to reduce power dissipation is to revisit conventional designs and to distribute power only to units that contribute to performance. This is done in many forms in modern processors including *power gating* [2]. In power gating we identify idle units and turn-off their power supply.

In this work we extend previous work [2] and investigate power gating as a solution to reduce leakage power in embedded processors. We analyze embedded processors and show that there is an opportunity to reduce leakage power dissipated by execution units. In particular, we show that execution units may be idle for long periods of time. Identifying such idle periods accurately provides an opportunity to apply power gating and to reduce power while maintaining performance.

To reduce power dissipation, we turn off the voltage supply for execution units that are detected to be in their idle time. Once a functional unit is needed we reactivate it by turning on the power supply. Unfortunately, there is a timing overhead associated with this switching. We show how variations in this overhead impact performance cost and energy savings.

One way to detect idle execution units is to monitor the units and to power gate them if they are idle for a consecutive number of cycles. In this work we apply this method to embedded processors.

The rest of the paper is organized as follows. In section 2 we discuss power gating in more detail. In section 3 we discuss our motivation. In section 4 we go over methodology, present our analysis framework and present performance and power savings results achieved by power gating in embedded processors. Finally, in section 5 we offer concluding remarks.

2. POWER GATING

Power dissipation in a CMOS circuit can be classified to dynamic and static. Dynamic power dissipation is the result of switching activity while static power dissipation is due to leakage current. Subthreshold leakage, gate oxide tunneling, drain-induced barrier lowering (DIBL), gate-induced drain leakage, hot carrier effects, reverse-biased PN junctions, and punchthrough currents [10,11] are various sources of leakage current. Among them the subthreshold leakage is considered to be an important contributor. Subthreshold leakage current (I_{Dsub}) flows from drain to source even when the transistor is off (see figure 1(a)).

I_{Dsub} can be expressed as:

$$I_{Dsub} = I_{S0}. [1 - e^{(-Vds/Vt)}]. [e^{(Vgs-VT-Voff)/nVt}] \quad \text{(Eq. 1)}$$

V_T is the threshold voltage, n is derived from other device parameters, V_{off} is a model parameter and V_t is a factor of temperature (Vt = KT/q where K and q are physical constants). I_{SO} current depends on transistor geometry and could be replaced by I'_{SO}.(W/L). V_{gs} is the voltage between gate and source and Vds is the voltage between drain and source. Butts and Sohi [9] have shown that for a single transistor in off state (where Vgs=0 and Vds=Vcc) by using the approximation Vds=VCC>>Vt equation 1 can be reduced to[9]:

$$I_{Dsub} = (W/L).K_{tech}.10\text{-}^{(-VT/St)} \quad \text{(Eq. 2)}$$

Where K_{tech}=Iso.exp(-Voff/(n.Vt)) and S_t=2.303.n.Vt. Note that W and L are transistor channel width and length respectively.

Assuming that subthreshold leakage current is the major contributor, and for a group of N transistors, transistor leakage current ($I_{leakage}$) can be expressed as:

$$I_{leakage}=N.K_{design}.K_{tech}.10^{(-V_T/S_t)} \quad \text{(Eq. 3)}$$

Equation 3 shows that the leakage current increases exponentially with decreasing V_T. This shows that the static power (or leakage power) portion in total power dissipation increases significantly as technology scales down the threshold voltage.

As power is the product of current and voltage, the static power dissipation is equal to:

$$P_{static}=Vcc.I_{leakage}=Vcc.N.K_{design}.K_{tech}.10^{(-V_T/S_t)} \quad \text{(Eq. 4)}$$

The parameters in equation 4 are divided to two categories: technology dependent and design dependent. Vcc, N and K_{design} are technology independent and may be varied independently targeting a specific design model.

V_T is a technology dependent parameter. As the technology scales down, V_T decreases which results in a significant increase in static power.

We use power gating to block Vcc and reduce leakage power to zero. In figure 1(b) we present how power gating is achieved using a header transistor to block voltage supply from reaching a circuit unit. The power gate detection circuit decides when is the appropriate time to turn off the voltage supply. Once the sleep signal is generated, and after a transition period, the Vcc signal will be blocked from reaching the functional unit.

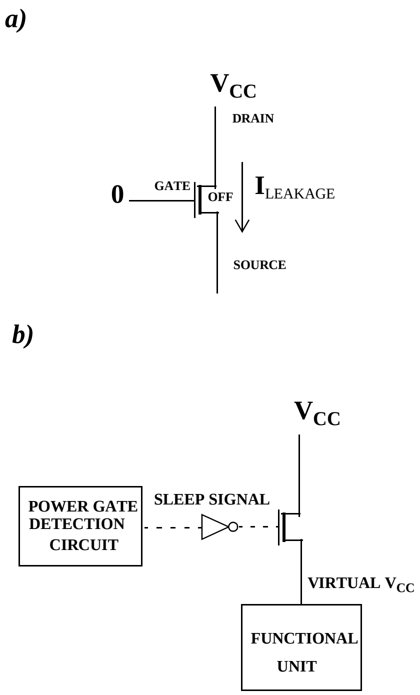

Figure 1: a) Turned off transistor dissipating leakage power b) Schematic showing major blocks exploited in power gating.

Applying power gating comes with timing overhead. To explain this in more detail in figure 2 we present transition states associated with power gating.

As presented the power gating process includes three separate intervals. We refer to the first interval as the *active to sleep transit period (ASP).* ASP starts the moment we decide to power gate a unit and ends when the voltage supply is completely blocked. We refer to the second interval as *the deep sleep period* or *DSP.* This is the period where the functional unit is gated and therefore does not dissipate power. Power dissipation reduction depends on how often and for how long units stay in DSP. We have to wakeup a unit as soon as its idle period ends. In the case of functional units, for example, this is when an instruction requires the functional unit to execute. Turning on the voltage supply to wakeup a unit takes time. The third interval presented in figure 2 represents this timing overhead and is the time needed to reactivate a unit. We refer to this period as *the sleep to active transition period (SAP).*

While saving leakage power during ASP and SAP is possible, in this study we assume that power reduction benefits are only achieveable when a unit is in DSP. As such we refer to ASP and SAP as timing overheads associated with power gating. Hu *et. al,* provide a detailed explanation of the three intervals [2].

One possible approach in implementing power gating is to monitor the state of an execution unit, and turn it off after seeing a number of consecutive idle cycles. This is referred to as the time-based power gating technique [2]. In this technique we gate an execution unit when the number of consecutive idle cycles seen exceeds a pre-decided threshold. We refer to this threshold as the *idle detect threshold* (*IDT*).

We analyze how time-based power gating performs for embedded processors in section 4.

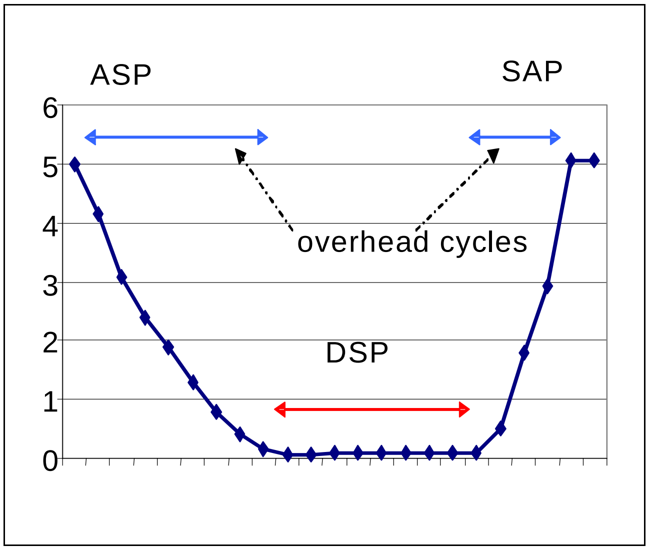

Figure 2: Transition states in power gating

3. MOTIVATION

In figure 3 we report leakage power reduction achieved by power gating in execution units. We report for both ideal and practical power gating. Bars from left to right report average savings for integer ALU, integer multiplier/divider, memory ports, floating point ALU and floating point multiplier/divider.

In figure 3(a) we present energy savings achievable by ideal (but not practical) power gating. We assume that the percentage of execution units' idle cycles indicates maximum leakage power reduction possible by using power gating. We also assume that the timing overhead with power gating is zero. As a result the data presented in figure 3(a) serves as an upper bound for power gating leakage power savings.

Through this study we report for a representative subset of MiBench benchmarks [8] and for a processor similar to that of Intel's XScale processor (*more on this later in section 4*).

In 3(a), and as an indication of potential leakage power savings, we report how often each of the five units used in the Intel's XScale are idle. On average, three of the units, *i.e.*, integer multiplier/divider, floating point ALU and floating point multiplier/divider are idle more than 95% of cycles. Average idle period is least for integer ALU (40%).

To provide insight to how timing limitations impact potential savings, in figures 3(b) and 3(c) we also report for the same processor when there is 10 and 20 cycle timing overhead associated with power gating respectively.

Comparing 3(a) with 3(b) and 3(c) reveals that, as expected, an increase in the timing overhead reduces power gating potential savings. Particularly, and as reported in 3(b) and 3(c), integer ALU loses high number of cycles to the increased overhead. Note that integer ALU can be power gated for only 10% of execution cycles when the timing overhead is increased to 20 cycles. The multiply/divide unit, on the other hand, has the highest number of idle cycles (more than 90% for different timing overheads presented here).

Note that 100% idle times for some units represented in figure 3 for some benchmarks is due to the fact that the benchmarks do not exploit that unit at all. Accordingly such units can be gated-off for the entire runtime.

We conclude from figure 3 that there is motivating opportunity in embedded processors to exploit idle times and to power gate execution units to reduce leakage power dissipation. However, identifying idle times early enough is a challenging problem. Moreover, reactivating the gated execution units soon enough is critical since stalling instruction execution could come with a performance penalty.

4. METHODOLOGY AND RESULTS

In this section we report our analysis framework and simulation results. To evaluate time-based power gating we report performance and the percentage of execution cycles a unit is gated-off. We use a subset of MiBench benchmark suite[8] compiled for MIPS instruction set. We report power savings for MiBench benchmarks here for different timing overheads (ASP and SAP) and IDTs. We also report how variations in ASP, SAP and IDT impact performance cost.

For simulation purpose we used a modified version of simplescalar v3.0 toolset[7]. We modeled a single issue in-order embedded processor with an architecture similar to Intel's XScale core. Table 1 shows the configuration we used.

Note that IDT is decided by the designer. ASP and SAP, on the other hand, are mainly determined by circuit design limits.

In figure 4 we report how changing the idle detect threshold or IDT impacts power gating. We assume that the active to sleep period is 3 cycles. We also assume that returning an execution unit from sleep to active takes 5 cycles. Later we will study how different ASP and SAP values impact our results.

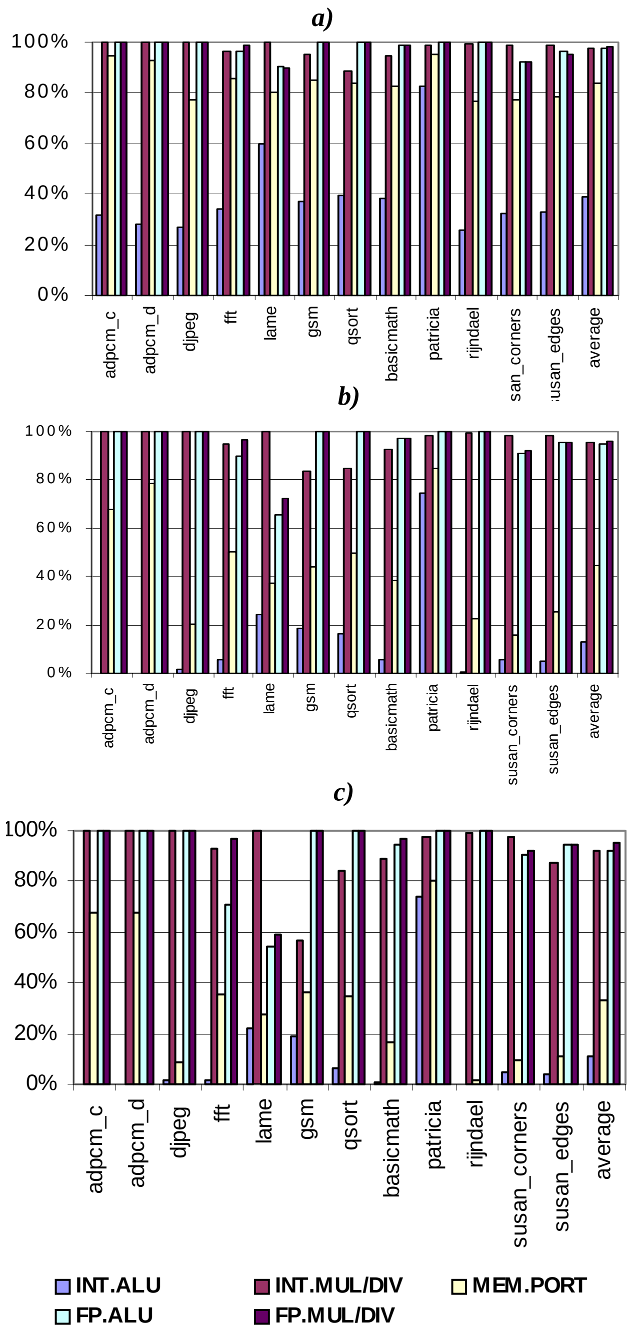

Figure 3: Leakage power reduction achieved by power gating for a) zero timing overhead, b)10 cycle timing overhead and c)20 cycle timing overhead.

Table 1: Configuration of the processor model

Issue Width	In-Order:2
Functional Units	1 I-ALU, 1 F-ALU, 1 I-MUL/DIV, 1 F-MUL/DIV
BTB	128 entries
Branch Predictor	Bimodal, 128 entries
Main Memory	Infinite, 32 cycles
Inst/Data TLB	32 entries, fully associative
L1 - Instruction/Data Caches	32K, 32-way SA, 32-byte blocks, 1 cycle
L2 Cache	None
Load/Store queue	8 entries
Register Update Unit	8 entries

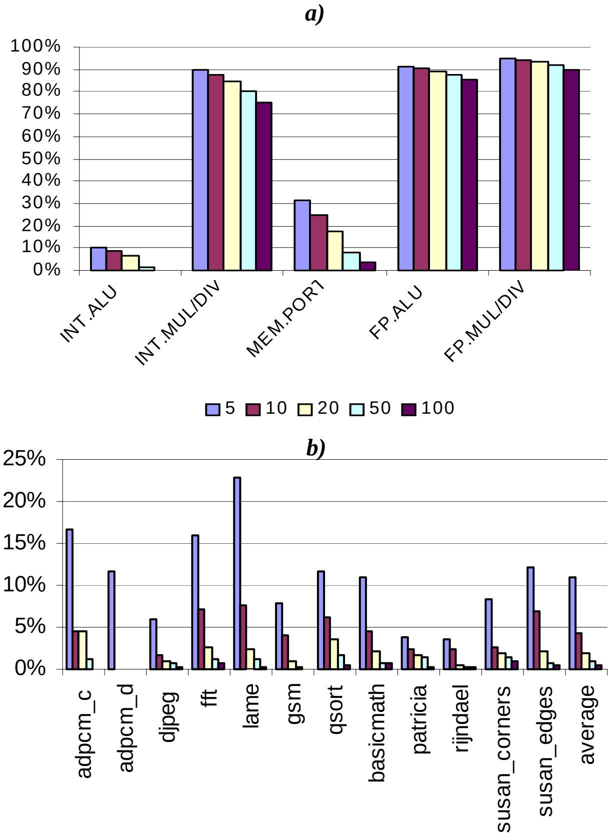

Figure 4: a) Average leakage power savings achieved by power gating for different IDT values for ASP=5 and SAP=3. Higher is better. b) Performance cost associated with power gating for different IDT values for ASP=5 and SAP=3. Lower is better.

In 4(a) bars from left to right report average percentage of cycles each execution unit is gated for the benchmarks studied here for IDT values 5, 10, 20, 50 and 100.

As presented in figure 4(a) some execution units appear to be more sensitive to changes in IDT. This is particularly true for integer ALU units and memory ports. Integer multiplier/divider, floating point ALU and floating point multiplier/divider show less sensitivity to IDT changes as their average idle cycles remain above 70%, 80% and 90% respectively across all the reported IDT values.

In 4(b) we report performance cost for the benchmarks studied here for different IDT values. Average performance slowdown is 10.9%, 4.1%, 1.9%, 0.9% and 0.3% for IDT values 5, 10, 20, 50 and 100 respectively.

We conclude from figure 4 that for the benchmarks studied here, IDT values below 20 are probably too costly from the performance point of view.

In figure 5 we report how changes in ASP impact power gating. As explained earlier, ASP is the time required to turn off the power supply for an execution unit. ASP depends on the circuit parameters and may be different from one design to another. In 5(a) we report average leakage power savings for different ASP values for each execution unit. We assume that IDT and SAP are 20 and 3 respectively. Bars from left to right report for ASP values of 0, 5, 10 and 15 respectively. While a zero ASP represents how well power gating performs if we cut the power supply at the earliest possible, an ASP of 15 represents a scenario where power gating comes with a long turn off transition period. As presented in 5(a), as expected, power savings are higher for lower ASP values. However, changes in ASP do not have a major impact on power savings.

In 5(b) we report how different ASP values impact performance cost for the benchmarks studied here. ASP changes do not impact performance in a major way. Average performance loss remains below 2% for all ASP values reported in figure 5(b).

We conclude from figure 5 that ASP changes occurring within practical limitations do not impact power gating dramatically.

In figure 6 we report how changes in SAP impact power gating. Note that SAP is the time required to reactivate an execution unit by turning on the power supply. Similar to ASP, SAP depends on the circuit parameters and may change from one design to another. In 6(a) we report average leakage power savings for different SAP values for each execution unit. We assume that IDT and ASP are 20 and 5 respectively. Bars from left to right report for SAP values of 1, 2, 3 and 4 respectively. Note that SAP is the time gap between when we realize that we need an execution unit and the time it is reactivated. As reported in 6(a), power savings are slightly higher for lower SAP values.

In 6(b) we report how SAP changes impact performance cost. As reported performance cost is sensitive to SAP. As expected the longer it takes to reactivate a gated execution unit the higher the performance penalty. Average performance cost is 0.5%, 1.2%, 1.9% and 2.7% for different SAP values.

We conclude from figure 6 that while SAP changes do not impact power gating's leakage power reduction dramatically, they could impact performance cost seriously.

5. CONCLUSION

In this paper we analyzed how power gating could be exploited in embedded processors to reduce leakage power. In particular we investigated how variations in timing overhead and design parameters impact performance and leakage power savings. Our study shows that for a processor with a configuration similar to Intel's XScale and for the representative subset of MiBench benchmarks studied here, using a simple time-based power gating techniques can reduce leakage power dramatically while maintaining performance. Moreover we show that changes in the idle detection threshold and the sleep to transition timing overhead can impact power gating considerably. We also show that power gating seems to be not sensitive to the active to sleep transition timing overhead.

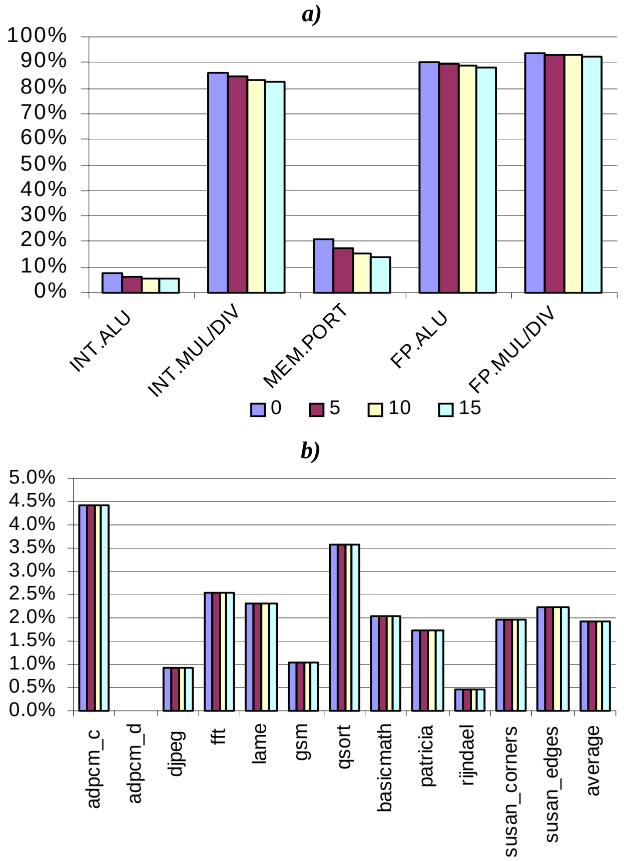

Figure 5: a) Average leakage power savings achieved by power gating for different ASP values for IDT=20 and SAP=3. Higher is better. b) Performance cost associated with power gating for different ASP values for IDT=20 and SAP=3. Lower is better.

6. ACKNOWLEDGMENTS

This work was supported by the Natural Sciences and Engineering Research Council of Canada, Discovery Grants Program and the Canada Foundation of Innovation, New Opportunities Fund.

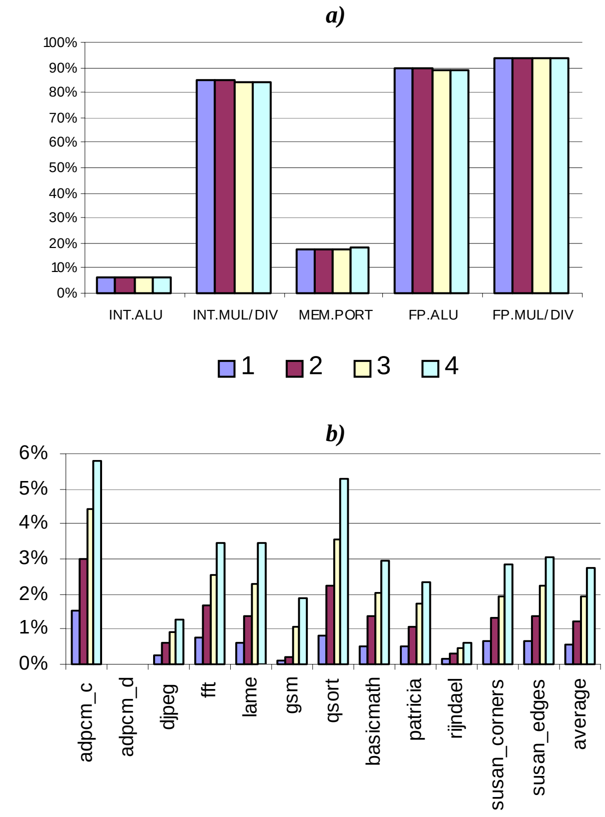

Figure 6: a) Average leakage power savings achieved by power gating for different SAP values for IDT=20 and ASP=5. Higher is better. b) Performance cost associated with power gating for different SAP values for IDT=20 and ASP=5. Lower is better.

References

[1]Borkar, S., Design Challenges of Technology Scaling. IEEE Micro 1999.

[2]Hu, Z., Buyuktosunoglu, A., Srinivasan, V., Zyuban, V., et al, Jacobson, H. and Bose, P., Microarchitectural Techniques for Power Gating of Execution Units. ISLPED'04.

[3]Powell, M., Yang, S., Falsafi, B., Roy, K. and Vijaykumar, T., Gated-Vdd: A Circuit Technique to Reduce Leakage in Deep-Submicron Cache Memories. In proceedings of ISLPED 2000.

[4]Kao, J. and Chandrakasan, A. Dual-Threshold Voltage Techniques for Low-Power Digital Circuits. IEEE Journal of Solid State Circuits 35 2000.

[5]Johnson, M., Somasekhar, D., Cheiou, L. and Roy, K., Leakage Control With Efficient Use of Transistor Stacks in Single Threshold CMOS. IEEE Transactions on VLSI Systems 10 2002.

[6]Durate, D., Tsai, Y. F., Vijaykrishnan, N. and Irwin, M. J., Evaluating Run-Time Techniques for Leakage Power Reduction. In proceedings of ASPDAC 2002.

[7]Burger, D., Austin, T. M. and Bennett, S., Evaluating Future Microprocessors: The SimpleScalar Tool Set. Technical Report CS-TR-96-1308, University of Wisconsin-Madison, July 1996.

[8]Guthaus, M., Ringenberg, J., Ernst, D., Austin, T., Mudge, T. and Brown, R., MiBench: A Free, Commercially Representative Embedded Benchmark Suite, IEEE 4th Annual Workshop on Workload Characterization (WWC-4), December 2001.

[9]Butts, J. A. and Sohi, G. S, A static power model for architects. In Proceedings of the 33rd Annual IEEE/ACM International Symposium on Microarchitecture, pages 191--201, Dec. 2000.

[10]Mamidipaka, M., Khouri, K., Dutt, N. and Abadir, M., Leakage Power Estimation in SRAMs, CECS Technical Report #03-32 Center for Embedded Computer Systems, University of California Irvine, 2003.

[11]Tsividis, Y. P., Operation and Modeling of the MOS Transistor. McGraw-Hill Book Company, 1988.

[12]Unsal, O. S. and Koren, I., System-Level Power-Aware Design Techniques in real-Time Systems, In proceedings of the IEEE, VOL. 91, NO. 7, July 2003.

[13]Jejurikar, R. and Gupta. R., Dynamic Voltage Scaling for Systemwide Energy Minimization in Real-Time Embedded Systems. In proceedings of ISLPED 2004.