Microarchitecture Level Power and Thermal Simulation Considering Temperature Dependent Leakage Model

Weiping Liao Fei Li Lei He

Electrical Engineering Department
University of California at Los Angeles

Abstract

In this paper, we present power models with clock and temperature scaling, and develop the first of its type coupled thermal and power simulation with temperature-dependent leakage power model at micro-architecture level. We show that leakage energy and total energy can be different by up to $2.5X$ and $2X$ for temperatures between 90^oC and 130^oC, respectively. Given such big energy variations, no power model at microarchitecture level is accurate without considering temperature dependent leakage models.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Simulation.

General Terms

Design.

Keywords

Leakage, thermal, simulation.

1. INTRODUCTION

A number of cycle accurate microarchitecture level power simulators [1, 2, 3, 4] have been developed for microarchitecture optimization. However, all these simulators consider only cycle-based power models without clock scaling, i.e., the dynamic and leakage energy is given per clock cycle, and the ratio between them is invariant with respect to the clock rate. This assumption of fixed leakage and dynamic energy ratio is no longer valid with clock scaling because for each clock cycle, the dynamic energy does not depend on clock period but the leakage energy does.

Temperature scaling is another important factor in power modeling. There is some limited study on the thermal modeling. [5] evaluates the thermal impact on future nanometer VLSI design from technology scaling point of view. [6] discusses the thermal performance for Intel's processors with emphasis on the package material and cooling mechanism. TEM^2P^2EST [3] is the first microarchitecture simulator with built-in thermal model. However, leakage power models in [1, 2, 3, 4] are independent of temperature, and may lead to severe estimation error given that leakage power is an exponential function of temperature.

In this paper, we present power models with clock and temperature scaling, and develop the coupled thermal and power simulator at microarchitecture level. With this simulator, we are able to accurately simulate the inter-dependence between the power and temperature, obtain accurate power and thermal profile, and evaluate microarchitecture level power and thermal management techniques. The rest of this paper is organized as follows: In Section 2, we propose our power models. In Section 3, we present our thermal model. In Section 4, we present the experiments on thermal-sensitive energy simulations as well as the impact of clock gating. To the best of our knowledge, this is the first paper considering the inter-dependence between clock rates, system energy and temperatures at microarchitecture level.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ISLPED'03, August 25–27, 2003, Seoul, Korea.
Copyright 2003 ACM 1-58113-682-X/03/0008 ...$5.00.

2. LEAKAGE POWER CALCULATION

We define three power states same as [4]: (i) *active mode*, where a circuit performs an operation 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). (ii) *standby mode*, where a circuit is idle but ready to execute an operation, and dissipates only leakage power(P_s). (iii) *inactive mode*, where a circuit is deactivated by power gating or other leakage reduction techniques, and dissipates a reduced leakage power defined as inactive power (P_i). A circuit in the inactive mode needs non-negligible amount of time to wake up and then perform an useful operation.

In cycle accurate simulations, power is defined as the energy per clock cycle. Therefore, P_d is equal to $\frac{1}{2}f_sCV^2$ where C is the switching capacitance, V is the supply voltage and f_s is the switching factor per clock cycle. In essence, P_d is the energy to finish a fixed number of operations during one cycle and is assumed to be independent of the clock rate in this paper. Consistently, P_s is defined as $P_{so}*t$ where P_{so} is leakage power per second and t is the clock period. Same as P_s, P_i is proportional to the clock period.

2.1 Related Work on Leakage Modeling

Microarchitecture level power simulators [2, 3, 4] calculate leakage power by assuming a ratio between P_s and P_d. For example, the ratio for logic circuits in [4] is decided by SPICE simulations on typical circuits. Furthermore, [4] also

presented formula-based models for P_s and P_i in memory based units. The models in [2, 3, 4] are efficient for microarchitecture level simulations. But the ratio is independent of clock and temperature, and therefore it is not accurate.

Another high-level leakage power model is proposed in [7] with the following simple equation:

$$P_{so} = V_{dd} \cdot N_{FET} \cdot k_{design} \cdot \hat{I}_{leakage} \quad (1)$$

where V_{dd} is the supply voltage, N_{FET} is the number of transistors, k_{design} is a design dependent parameter, and $\hat{I}_{leakage}$ is a technology dependent parameter. Both k_{design} and $\hat{I}_{leakage}$ are different for different circuit types. However, there is no well-defined method to decide k_{design} and $\hat{I}_{leakage}$. More importantly, (1) does not consider temperature scaling.

2.2 New Leakage Model for Logic

In this paper, we propose a new leakage power model for logic circuits. As shown in (2), for a given circuit, the leakage power can be calculated as the product of gate number (N_{gate}) and the average leakage current per gate (I_{avg}).

$$P_{so} = N_{gate} * I_{avg} * V_{dd} \quad (2)$$

I_{avg} can be calculated by computing the average leakage current per gate for given n circuits based on the gate-level estimation results. Because leakage current depends on different input vectors [8], we apply Genetic Algorithm for a few typical circuits, and obtain the I_{avg} for both maximum and minimum leakage currents. Figure 1 shows such I_{avg} calculation with respect to the number of circuits for both maximum and minimum leakage currents. It is easy to see that after the number of circuits exceeds 20, the value of I_{avg} becomes very stable. Formula similar to (2) has been proposed in [9] which can consider statistic impact of the stack effect. However, no explicit method is proposed in [9] to calculate I_{avg}.

Also as shown in Figure 1, the average difference between maximum and minimum I_{avg} is about $1.6X$. To consider the worst-case leakage current, we always use the maximum leakage current in the rest of the paper. Furthermore, we propose the following temperature scaling for I_{avg} considering the exponential relationship between the leakage power and temperature:

$$I_{avg}(T) = I_s * exp\left(-\frac{\alpha}{T+273-\beta}\right) \quad (3)$$

where I_s is a constant value. The coefficients α and β are decided by circuit designs. Values for α and β as well as validation of (3) will be presented in Section 2.4.

2.3 New Leakage Model for Memory Based Units

Memory based units such as caches and register files are usually modeled by SRAM arrays. The formula-based leakage power model has been proposed in [4]. We propose the following temperature scaling based on the leakage model from [4].

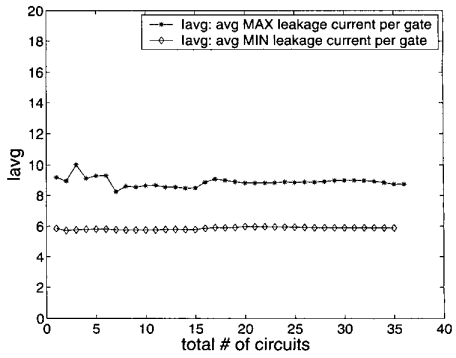

Figure 1: I_{avg} calculation.

$$P_{so} = P_{circuits} + P_{cells} \quad (4)$$

$$P_{circuits} = (X * words + Y * word\_size) * exp\left(-\frac{\alpha}{T+273-\beta}\right) \quad (5)$$

$$P_{cells} = (Z * words * word\_size) * exp\left(-\frac{\gamma}{T+273-\delta}\right) \quad (6)$$

where P_{cells} is the leakage power dissipated by SRAM memory cells and $P_{circuits}$ is the power generated by the circuits such as wordline drivers and precharge transistors, etc. P_{cells} is proportional to the number of SRAM memory cells. α and β in (5) are the same as those in (3), while X, Y, Z, γ and δ in (5) and (6) are coefficients decided by circuit designs. Values for X, Y, Z, γ and δ as well as validation of (5) and (6) will be presented in Section 2.4.

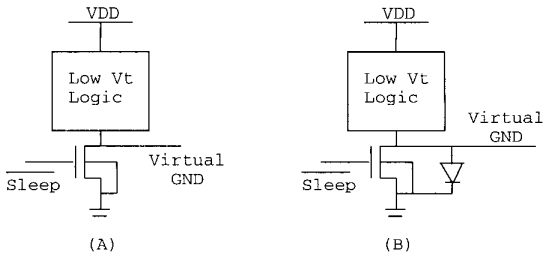

Figure 2: Power gating techniques: (a) MTCMOS and (b) VRC.

2.4 Consideration of Leakage Reduction

For both logic circuits and memory based unit, we can easily extend our leakage power model for leakage reduction techniques. For example, power gating techniques such as MTCMOS and VRC are used in [4]. Figure 2 shows the schematics for MTCMOS and VRC. In MTCMOS, a sleep transistor is inserted between the circuits and GND. When the sleep transistor is turned off, there is no power supply to the logic circuits. In VRC, a diode is inserted parallel with

the sleep transistor and maintains the voltage level that is needed to keep the logic states in the circuits. VRC should be used for memory based units such as caches and TLBs for data retention. In these cases, the leakage power formulas still have the same format as we shown above, although the coefficients are different.

We collect the power consumption for different types of circuits at a few temperature levels by SPICE simulations. We then obtain the coefficients in (3) - (6) by curve fitting. Table 1 summarizes the coefficients for ITRS 100nm technology we used, and Table 2 compares our high-level leakage power estimation for logic circuits and SRAM arrays with SPICE simulations in ITRS 100nm technology. The difference between our formulas and SPICE simulation is less than 6% without power gating, and is less than 15% with power gating. Note that with the presence of sleep transistors, the leakage power value is smaller due to the stack effect. For example, as shown in Table 2, 13.88% difference of P_{so} for a 128x32 SRAM array with power gating is about 0.035 mW, which is equal to 11.5% of P_{so} without power gating.

		With power gating	Without power gating
Logic circuits	X	2.63e-6	3.89e-4
	Y	2.49e-6	1.24e-3
	α	606.53	3040.11
	β	192.02	23.37
Memory based units	Z	8.91e-5	4.33e-4
	γ	2273.27	3168.35
	δ	3168.35	21.34

Table 1: Coefficients in (3) - (6) for 100nm technology, where MTCMOS and VRC are the power gating techniques for logic and SRAM arrays, respectively.

circuit	T (°C)	I_{avg} or P_{so} formula	I_{avg} or P_{so} SPICE	abs. err. %
adder	100	0.0249	0.0238	4.62
multiplier	100	0.0228	0.0217	5.07
SRAM 128x32	110	0.298	0.304	1.97
SRAM 512x32	110	1.163	1.145	1.57
SRAM(VRC) 128x32	110	0.287	0.252	13.88
SRAM(VRC) 512x32	110	1.137	1.082	5.08

Table 2: Comparison of I_{avg} for logic circuits and P_{so} for SRAM power model between our formula and SPICE simulation. The SRAM arrays are represented as "row number" x "column number". The units for I_{avg} and P_{so} are uA and mW, respectively.

3. TEMPERATURE CALCULATION

We develop the thermal model based on conventional heat transfer theory [10]. The stable temperature in our thermal model can be calculated according to (7):

$$T = T_a + R_t * P \quad (7)$$

where T is the stable temperature, T_a is the ambient temperature, P is the power consumption, A is the area, and R_t is the thermal resistance, which indicates the ability to remove heat to the ambient under the steady-state condition.

The dissipated power does not convert to temperature immediately because of the slow material response. Therefore, an exponential response is expected for the transient temperature. In our thermal model, we calculate the temperature change for any period from time t_1 to t_2 by (8) and (9):

$$\Delta T_+ = (T_{max} - T_{t1}) * exp\left(-\frac{t_2 - t_1}{\tau_{heat}}\right) \quad (8)$$

$$\Delta T_- = (T_{t1} - T_a) * exp\left(-\frac{t_2 - t_1}{\tau_{cool}}\right) \quad (9)$$

where $\Delta T_+ (\Delta T_-)$ are the increment(decrement) of the temperature from time t_1 to t_2, T_{max} is the maximum silicon temperature that the package supports, and τ_{heat} and τ_{cool} are heating and cooling time constants. The decision to increase or decrease the temperature at time t_2 is made by the following criterion:

Suppose the average power between t_1 and t_2 is P_{avg}. If $T_a + R_t * P_{avg} > T_{t_1}$, then the temperature increases; otherwise, the temperature decreases.

In our thermal model, we have two different modes with different granularities to calculate the temperature: (i) *individual mode*. We assume that there is no horizontal heat transfer between components, and calculate a temperature for each individual component. In general, the horizontal heat reduces the temperature gaps between components. So the individual mode essentially gives the upper bound of temperature gaps. (ii) *universal mode*, which is similar to the thermal model in TEM^2P^2EST [3]. We assume the whole processor as a single component with a uniform thermal characteristic and temperature. The universal mode gives the lower bound of the highest on-chip temperature.

4. EXPERIMENT RESULTS

Although our power and thermal models are applicable to any architecture, we study VLIW architecture in this paper. We integrate our thermal and power model into the PowerImpact [4] toolset. Instead of fixing the absolute thermal resistance value, we use the relative thermal resistance in our experiments. First, we select the thermal resistance of one integer unit as the unit thermal resistance, and further define the integer unit's area as the unit area. The thermal resistances for all other components are inversely proportional to their areas. Table 3 presents the micro-architecture configuration of the VLIW processors we study, and Table 4 summarizes the power consumption, the relative thermal resistances and the relative areas for all components in our system. We set the thermal time constants as $\tau_{heat} = \tau_{cool} = 100us$, which are independent of component area.

4.1 Chip Temperature

In our experiments, we update temperatures after each time step t_s. We then update the power value with respect to new temperature for each t_s. Smaller t_s gives a more accurate transient temperature analysis, e.g., $t_s = 1$ cycle represents the cycle accurate temperature calculation. Figure 3 plots the transient temperature calculated under different t_s shown as the percentages of the thermal time

Component	Configuration			
Decode (MTCMOS)	6-issue width			
BTB	512 entries 4-way associative, Two-level predictor			
Register file	128 integer and 128 floating-point registers with 64-bit data width			
Memory	page size 4096 bytes, latency 30 cycles			
Memory Bus	8 bytes/cycle			
ALU (MTCMOS)	Number	Latency		
Integer	4	1 cycle for add, 2 cycles for multiply and 15 cycles for division		
Floating-point	2	2 cycles for add/multiply, 15 cycles for division		
Cache (VRC)	Size	Block size	Associativity	Policy
L1 Instruction	32 KB	32 bytes	2	LRU
L1 Data	32 KB	32 bytes	4	LRU
L2	512 KB	64 bytes	8	LRU

Table 3: System configuration for experiments.

Component	P_a	P_s	P_i	R_t	*Area*
BTB	10.38	2.2385	0.0093	2048.2	0.49
I-L1 Cache	90.28	28.18	0.027	277.9	4.28
D-L1 Cache	90.78	28.28	0.027	275.4	4.30
L2 Cache	666.53	437.07	0.287	30.1	31.61
Registers	4.99	1.65	0.004	5012.4	0.24
Decode	14.06	0.025	0.00032	1178.22	0.667
Integer ALU	21.09	0.036	0.0005	1185.48	1.0
FPU	42.18	0.073	0.001	592.74	2.0

Table 4: Power consumption (in pJ/cycle), relative thermal resistance R_t and relative areas for all components. We assume 125°C and 1GHz clock rate. The decode, integer ALU and FPU are only one unit among total six, four and two units.

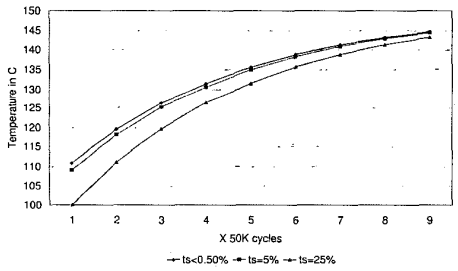

Figure 3: Temperature curve obtained by the universal mode for different time step t_s. The clock frequency is 2GHz. 0.5%, 5% and 25% of thermal time constant corresponds to 1000, 10000 and 50000 cycles, respectively. The benchmark is *equake*.

constant, where 0.5% of thermal time constants is equal to 1000 clock cycles for a 2GHz clock. When $t_s \leq 1000$ cycles (i.e. 0.5% of thermal constants), the temperatures are identical to those with $t_s = 1$ cycle. Observable difference appears when t_s is increased to 5% of the thermal constants and significant error is induced when $t_s = 25\%$ of the thermal constants. Furthermore, Table 5 presents the running time normalized with respect to that without temperature calculation. By setting t_s to more than 100 cycles, we can reduce the running time by more than 5 times compared to $t_s = 1$ cycle, and achieve virtually the same computation efficiency as the power simulation without temperature calculation. Since 0.5% of thermal constants are always more than 100 cycles for the clocks we study, and lead to negligible error on temperature calculation compared with the cycle accurate temperature calculation, we only update temperatures and power values after every period of 0.5% of the thermal time constants in the rest of the paper.

t_s (cycle)	N.T.	1	10	100	500	1000
Running time	1.0	5.66	1.49	1.05	1.03	1.01

Table 5: Normalized running time for different temperature updating period. The *N.T.* means we do not have to update temperature and power during the whole simulation.

4.2 Energy Consumption

Figure 4 shows the experimental results for total energy consumption with different clocks. We assume there is no throttling, i.e., P_a is dissipated in every cycle. We study two cases: one assumes a fixed temperatures, and another considers energy consumption with temperature dependence in both individual mode and universal mode. From Figure 4 we can see that by changing the temperature from 90°C to 130°C, the total leakage energy can be changed by a factor of $2.5X$, and the total energy is changed by up to 30%. Figure 4 clearly shows that any study regarding to leakage energy is not accurate if the thermal issue is not considered. To consider temperature using methods in [4, 11], we may assume a fixed temperature appropriate for the processor and the environment, and then use circuit-extracted leakage values for that temperature. As shown in Figure 4, how to decide the appropriate temperature is of paramount importance for accurate energy estimation, and it is an open problem in the literature. Given the dependence between temperature and leakage energy, our work actually presents an approach to select the appropriate temperature point. Note that we assume that dynamic energy is independent of clock. This assumption will be revisited in Section 5.

Figure 4 also shows that the total leakage energy is re-

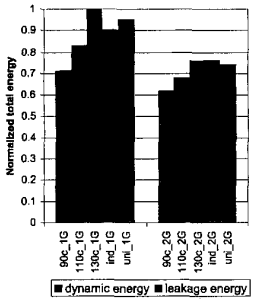

Figure 4: Energy consumption without any throttling. We study fixed temperatures of 90°C, 110°C and 130°C, as well as the case with temperature dynamically updated. The prefix of "ind" and "uni" stand for the individual mode and universal mode, respectively. The benchmark is *equake*.

duced as the clock increases from 1GHz to 2GHz, due to the reduced execution time for the given amount of work. This result implies that total leakage energy can be reduced by increasing system clock. However, the maximum temperature and maximum temperature gap constraints prevent us from increasing clock rate indefinitely. In our experiments, we assume the maximum allowable temperature is 130°C and the maximum temperature gap among components is 40°C. We use the individual mode to calculate the maximum temperature and the maximum temperature gap, where the maximum temperature is set as the largest temperature among all temperature. Table 6 shows the maximum system temperature and the maximum temperature gap without any throttling. We can see that the maximum clock with thermal constraints is about 1GHz when there is no throttling.

Benchmark	Clock (Hz)	500M	1G	1.5G	2G
equake	Max T	118.59	128.07	137.71	149.75
	Max Gap	5.96	9.86	14.85	21.92
go	Max T	118.58	128.06	137.6	147.52
	Max Gap	3.76	6.8	14.51	22.1
ijpeg	Max T	118.6	128.07	137.7	144.67
	Max Gap	3.76	6.8	14.5	17.04

Table 6: Maximum temperatures (*Max T*) and temperature gaps (*Max Gap*) among components for different clocks without any throttling. The unit for temperatures is °C.

4.3 Impact of Clock Gating

Clock gating [12] is effective to reduce dynamic power by turning off the clock signal for idle components. It also reduces total leakage energy by lowering the temperature. In this section, we consider the ideal clock gating, i.e. no overhead to turn on and off the clock for a component, and present a quantitative study on the impact of clock gating.

In [4], the instructions are always assigned to preferred functional units first. This method makes the preferred units much busier, and therefore have much higher temperatures

Benchmark	Clock (Hz)	500M	1G	1.5G	2G	2.5G
equake	Max T	111.4-114.8	113.5-125	115.6-135.5	117.5-146.3	119.3-154.5
	Max Gap	12.99	21.50	30.39	39.78	42.20
go	Max T	110.8-112.8	112.4-119.2	114-123.9	115.6-128.7	117.1-133.6
	Max Gap	11.84	17	21.47	25.3	29.1
ijpeg	Max T	111.7-112.5	114.2-121.1	116.8-129.2	119.3-136.6	121.5-142.1
	Max Gap	11.8	19	25.47	30.67	34.2

Table 7: Maximum temperatures and temperature gaps among components for different clocks under clock gating. The maximum temperatures are shown as a range where the lower bound and the upper bound are given by the universal mode and individual mode, respectively.

than others. To reduce the temperature gaps, we propose to evenly distribute instructions to functional units. Our experiments show that temperature gaps between integer units can be virtually eliminated by this scheduling.

Table 7 presents the maximum temperatures and maximum temperature gaps with clock gating for different clocks. Compared with Table 6, the maximum temperatures with clock gating are reduced due to energy reduction but the maximum temperature gaps are increased due to dynamic throttling.

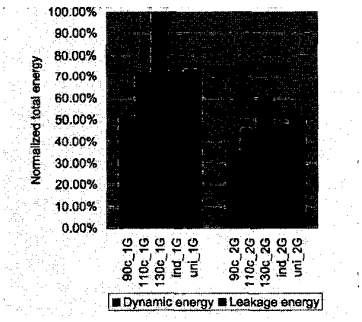

Figure 5: Energy consumption with clock gating. The conditions are the same as those in Figure 4.

4.3.1 Energy

Figure 5 shows the total energy consumption with clock gating for different clock. When the temperature change from 90°C to 130°C, the total energy is increased by up to a factor of $2X$ due to that the dominate leakage energy changes dramatically. From Figure 5 we can also see that leakage energy with temperature dependence in both individual mode and universal mode is close to that with fixed temperature of 110°C, while in Figure 4, the corresponding leakage energy with temperature dependence is close to that with fixed temperature of 130°C. The leakage energy with clock gating in Figure 5 is smaller than that without clock

gating in Figure 4 and the overall temperature profile with clock gating is also lower as shown in Table 7. All these results again show that any study regarding to leakage energy should consider the thermal issues for accuracy.

4.3.2 Maximum Clock

Clock gating has no direct effect on leakage energy reduction. However, as clock gating reduces the dynamic power consumption and the system temperatures, the temperature dependent leakage power is reduced. Furthermore, the maximum clock can be increase under the same thermal constraints. Faster clock rates reduce the execution time, and therefore reduce the total leakage energy.

We use the upper bound of the maximum temperature in Table 7 as the maximum temperature. By satisfying the same maximum temperature and maximum temperature gap constraints as those in Section 4.2, Table 8 summarizes the max clocks and total leakage energy under clock gating. It is shown that we can increase the maximum clock by up to a factor of $2X$, and reduce the total energy to as low as 51.94%. In other words, the total energy can be reduced by up to 48.06%.

Benchmark	Max Clock	Leakage energy
equake	1GHz	100 %
go	2GHz	51.94%
ijpeg	1.5GHz	69.89%

Table 8: Maximum clock and corresponding normalized total energy under clock gating. The total leakage energy without clock gating under 1GHz clock is assumed as 100%.

5. CONCLUSIONS AND DISCUSSIONS

Considering cycle accurate simulation, we have presented dynamic and leakage power models with clock and temperature scaling, and developed the coupled thermal and power simulation at the microarchitecture level. With this simulator, we have shown that the leakage energy and total energy can be different by up to $2.5X$ and $2X$ for different temperatures, respectively. Hence, microarchitecture level power simulation is hardly accurate without considering temperature dependent leakage model. We have also discussed temperature scaling of the reduced leakage power in the power-gated circuit. Such scaling model can be applied to microarchitecture level power gating presented in [4, 13].

As pointed out by the reviewers, a per-component thermal model similar to our individual mode was developed in [14] and was used for dynamic thermal management. The temperature calculation in [14] is essentially same as (8) and (9) in this paper. Furthermore, it was shown in [14] that the horizontal heat transfer is negligible, same as assumed in our individual mode. However, leakage power was not taken into account in [14].

In this paper we assume that the dynamic power, i.e., the switching energy per cycle is independent of the clock rate. To increase the clock rate, the designer may have to increase the supply voltage to meet the delay constraints determined by the clock rate. Dynamic voltage scaling (DVS) [15] has been proposed to consider the relationship between supply voltage and clock rate. Because the dynamic power depends on the supply voltage, our future work will consider not only leakage power scaling but also dynamic power scaling with respect to clock. We do not consider interconnect power explicitly and assume fixed floorplan in this paper. In the future, we intend to study microarchitecture power/thermal management with simultaneous interconnect power estimation and floorplanning optimization.

6. REFERENCES

[1] W.Ye, N.Vijaykrishnan, M.Kandemir, and M.J.Irwin, "The design and use of simplepower: a cycle-accurate energy estimation tool," in *DAC*, 2000.

[2] D.Brooks, V.Tiwari, and M.Martonosi, "Wattch: A framework for architectural-level power analysis optimization," in *ISCA*, 2000.

[3] A. Dhodapkar, C. Lim, and G. Cai, "Tem2p2est: A thermal enabled multi-model power/performance estimator," in *Workshop on Power Aware Computer Systems*, Nov 2000.

[4] W. Liao, J. M. Basile, and L. He, "Leakage power modeling and reduction with data retention," in *ICCAD 02*, Nov 2002.

[5] D. Sylvester and H. Kaul, "Future performance challenges in nanometer design," in *DAC*, 2001.

[6] R. Viswanath, V. Wakharkar, A. Watew, and V. Lbonheur, "Thermal performance challenges from silicon to systems," *Intel Technology Journal*, vol. 3, 2000.

[7] J. Butts and G. Sohi, "A static power model for architects," in *Proc. of MICRO33*, December 2000.

[8] D. Duarte, Y. Tsai, N. Vijaykrishnan, and M. J. Irwin, "Evaluating run-time techniques for leakage power reduction techniques," in *ASP-DAC*, 2002.

[9] W. Jiang, V. Tiwari, E. de la Iglesia, and A. Sinha, "Topological analysis for leakage prediction on digital circuits," in *Proceedings of the 7th Asia and South Pacific Design Automation Conference, joint with the 15th International Conference on VLSI Design*, 2002.

[10] J. V. D. Vegte., *Feedback Control System, 3rd Edition.* Prentice Hall, 1994.

[11] H. Hanson, M. Hrishikesh, V. Agarwal, S. Keckler, and D. Burger, "Static energy reduction techniques for microprocessor caches," in *Proceedings of the International Conference on Computer Design*, 2001.

[12] V. Tiwari, D. Singh, S. Rajgopal, and G. Mehta, "Reducing power in high-performance microprocessors," in *DAC*, 1998.

[13] L. Li, I. Kadayif, Y.-F. Tsai, N. Vijaykrishnan, M. Kandemir, M. J. Irwin, and A. Sivasubramaniam, "Leakage energy management in cache hierarchies," in *Proceedings of International Conference on Parallel Architectures and Compilation Techniques*, 2002.

[14] T. A. K. Skadron and M. Stan, "Control-theoretic techniques and thermal-rc modeling for accurate and localized dynamic thermal management," in *Proceedings of the Eighth International Symposium on High-Performance Computer Architercture*, 2002.

[15] T. Burd, T. Pering, A. Stratakos, and R. Bordersen, "A dynamic voltage-scaled microprocessor system," in *2000 IEEE International Solid-State Circuits Conference Digest of Technical Papers*, Feb 2000.