Adaptive Thermal Management for High-Performance Microprocessors

David Brooks and Margaret Martonosi
Dept. of Electrical Engineering
Princeton University
{dbrooks,mrm}@ee.princeton.edu

Abstract

With the increasing clock rate and transistor count of today's microprocessors, power dissipation is becoming a critical component of system design complexity. Thermal and power-delivery issues related to the maximum power dissipation are becoming especially critical for high-performance computing systems.

In this work, we investigate dynamic thermal management as a technique to control CPU power dissipation. With the increasing usage of clock gating techniques, the average power dissipation typically seen by common applications is becoming much less than the chip's rated maximum power dissipation. However, system designers still must design thermal heat sinks to withstand the worst-case scenario. We show that with appropriate dynamic thermal management, the CPU can be designed for a much lower maximum power rating with minimal performance impact for typical applications.

1 Introduction

Today's highest performance microprocessors contain on the order of one hundred million transistors with this number continuing to grow with Moore's Law. The majority of the transistors on these chips are designed to extract ILP and to reduce memory access times for a range of common applications. As we move towards billion transistor microprocessors, the growing power budgets of these chips must be addressed at all levels of the design cycle.

The system complexity associated with increased power dissipation can be divided into two main areas. First, there is the cost and complexity of designing thermal packaging which can adequately cool the processor. It is estimated that after exceeding 35-40W, additional power dissipation increases the total cost per CPU chip by more than \$1/W [8]. The second major source of design complexity involves power delivery, specifically the on-chip decoupling capacitances required by the power distribution network.

One important aspect of the complexity of these cooling techniques is that they must be designed to withstand the *maximum* possible power dissipation of the microprocessor, even if these cases rarely occur in typical applications. For example, while the Alpha 21264 processor is rated as having a maximum power dissipation of 95W, the average power dissipation was found to be only 72W [3]. This disparity between the maximum possible power dissipation and the typical average power dissipation observed led us to consider dynamic thermal management techniques which seek to ensure that the processor does not reach these maximum power dissipation levels. That is, we seek to explore scenarios where the cooling apparatus is designed for a wattage less than the true maximum power, and dynamic CPU approaches guarantee that this designed-for level is never exceeded during a program run.

The G3 and G4 PowerPC microprocessors include a Thermal Assist Unit (TAU) providing dynamic thermal management [6, 7]. The TAU provides programmable thermal thresholds and generates an interrupt if the temperature crosses the threshold. In the PowerPC designs, the TAU invokes instruction cache throttling as a means to lower the processor's overall junction temperature when a thermal emergency occurs. With this technique, the instruction flow between the instruction cache and the instruction buffer is decreased to reduce the overall instruction execution rate and hence the system's overall power dissipation.

In this work, we consider speculation control as a method for dynamic thermal management. We compare this to restricting the decode bandwidth of the machine which is similar to the I-cache throttling of the PPC. For many benchmarks, speculation control leads to a smaller performance degradation with similar reductions in chip temperature.

1.1 Motivations for Dynamic Thermal Management

Dynamic thermal management is particularly appealing for portable and embedded systems where heat sinks and fans add size and weight in addition to cost and battery life. In fact, two of the most notable commercial applications of dynamic thermal management, the TAU in the G3 microprocessor by Motorola and the LongRun technology in Transmeta's Crusoe chip were both targeted for the portable computing market [6, 9].

Dynamic thermal management can also play a key role in the highest-end microprocessors where maximum power is especially important. System designers are often forced to overdesign for worst-case maximum power which significantly increases the system cost and complexity. Two of the major challenges related to maximum power are the thermal issues related to packaging and power-delivery issues related to the large cycle-to-cycle current swings. In fact, the two problems can be thought of as being somewhat similar; the major difference is the time quantum of interest.

With thermal issues, power spikes that last on the order of milli-seconds (10k-1000k cycles) will cause the chip packaging to heat up and require complex cooling systems to ensure system reliability. Dynamic thermal management based on junction temperature feedback could be used to smooth out these power spikes to ensure that the chip temperature never exceeds a pre-set level in order to ensure system reliability.

The challenges in the power-delivery problem are related to much shorter power spikes that last on the or-

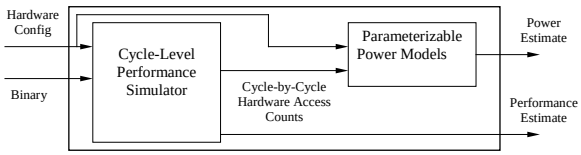

Figure 1: Overall Structure of Wattch.

der of nano- or micro-seconds (10s or 100s of cycles). Many of these short-term power spikes are directly related to the growing usage of aggressive clock gating for reducing average power within the microprocessor. Short-term power spikes can be smoothed out by slowly turning on and off clock gated units [5].

In this work we introduce the number of thermal emergencies as a metric and explore a new thermal management technique based on speculation control. We also describe a methodology for identifying candidates for dynamic thermal management techniques by investigating the correlation between IPC, power dissipation, and various architectural statistics for a variety of applications. We hope to gain the greatest leverage by looking for levers that reduce power more than they reduce performance.

In this work we primarily focus on dynamic thermal management in the context of reducing thermal issues in high-end microprocessors. In future work, we hope to address the related issue of the power-delivery problem.

Section 2 provides a description of our performance and power modeling methodology. Section 3 provides analysis for the performance and power effects of applying dynamic thermal management. Section 4 describes the methodology for identifying thermal management techniques based on correlation statistics. In Section 5 we discuss future possibilities for research in the area of adaptive thermal management and provide conclusions.

2 Methodology

We have developed an architectural-level power modeling tool called Wattch [1]. Wattch provides power modeling extensions to the SimpleScalar architecture simulator [2]. Wattch's power modeling infrastructure is based on parameterized power models of common structures present in modern superscalar microprocessors. Per-cycle power estimates are generated by scaling these power models with hardware access counts and activity factors from the performance simulator. Figure 1 illustrates the overall structure of Wattch and the interface between the performance simulator and the power models.

2.1 Simulation Model Parameters

Unless stated otherwise, our results in this paper model a processor with the configuration parameters shown in Table 1. For technology parameters, we use the process parameters for a .35um process at 600MHz. We use Wattch's aggressive clock gating style (linear power scaling with number of active ports on any particular unit) for all results.

Parameter	Value
Processor Core	
RUU size	80 instructions
LSQ size	40 instructions
Fetch Queue Size	8 instructions
Fetch width	4 instructions/cycle
Decode width	4 instructions/cycle
Issue width	4 instructions/cycle (out-of-order)
Commit width	4 instructions/cycle (in-order)
Functional Units	4 Integer ALUs
	1 integer multiply/divide
	1 FP add, 1 FP multiply
	1 FP divide/sqrt
Branch Prediction	
Branch Predictor	Combined, Bimodal 4K table
	2-Level 1K table, 10bit history
	4K chooser, 1024-entry, 2-way
RAS	32-entry
Mispredict penalty	7 cycles
Memory Hierarchy	
L1 D-cache	64K, 2-way (LRU)
	32B blocks, 1 cycle latency
L1 I-cache	64K, 2-way (LRU)
	32B blocks, 1 cycle latency
L2	Unified, 2M, 4-way (LRU)
	32B blocks, 12-cycle latency
Memory	100 cycles
TLBs	128 entry, fully associative
	30-cycle miss latency

Table 1: Baseline Configuration of Simulated Processor

2.2 Benchmark Applications

We evaluate our ideas on programs from the SPECint95 and SPECfp95 benchmark suites. SPEC95 programs are representative of a wide mix of current integer and floating-point codes. We have compiled the benchmarks for the Alpha instruction set using the Compaq Alpha *cc* compiler with the following optimization options as specified by the SPEC Makefile: -migrate -std1 -O5 -ifo -non_shared. For each program, we simulate 200M instructions.

2.3 Power vs. Temperature

Wattch provides per-cycle power estimates, but one challenge in this research has been translating these power estimates into chip-level temperature variations. The most accurate approach would be to develop a model for the chip packaging and heat sink in a microprocessor. We are currently investigating such models, but for this work our approach has been more abstract. We use the average power over a suitably large chunk of cycles (10k, 100k, and 1M) as a proxy for temperature.

3 Dynamic Thermal Management: Speculation Control

Dynamic thermal management uses thermal feedback from an on-chip temperature sensor to adjust the performance of the processor to dynamically reduce the power consumption. The goal of designing a good dynamic thermal management scheme should be to reduce power with as small a performance loss as possible. Clock frequency scaling essentially trades a linear performance

loss for a linear power savings. Obviously many other schemes could be used to dynamically reduce power consumption, and in fact two such schemes already exist in commercial microprocessors.

The PowerPC G3 microprocessor uses instruction cache throttling to restrict the flow of instructions to the processor core [7]. This scheme relies on clock gating to reduce power dissipation as the flow of instructions is restricted. As motivation for selecting I-cache throttling instead of clock frequency scaling, the authors cite the difficulty in implementing dynamic clock control for the on-chip PLL as well as the fact the chip's L2 cache interface operates at a different clock rate from the chip's core.

Transmeta's LongRun technology performs dynamic clock frequency scaling along with dynamic voltage scaling to reduce power dissipation when necessary [9]. Obviously this requires detailed timing analysis and careful attention to circuit design choices.

In this section, we consider a dynamic thermal management technique based on speculation control. This technique is similar to Manne's work on speculative pipeline gating based on branch confidence estimation [4]. However, instead of basing the speculation control on branch confidence as in [4], we arbitrarily restrict the amount of speculation in the pipeline whenever a thermal trigger level is reached.

The mechanism for speculation control is fairly trivial to implement. A counter of the number of unresolved branches active in the processor pipeline is incremented whenever a branch is decoded. If it exceeds a limit, the decode stage stalls until a live unresolved branch has been resolved. The infrastructure for this mechanism is most likely already in place in most processors which limit the number of branches in the pipeline due to the need to restrict the additional state required for each active branch.

The proposed modification would involve reducing the maximum allowed number of unresolved branches when the processor junction temperature exceeds a certain thermal trigger value based on the available heat-sink. Both the trigger and the number of unresolved branches could be programmable, allowing system designers to specify thermal management levels based on the amount of heat-sink technology in the system. For example, more expensive high-end server systems could have higher trigger limits and allow more unresolved branches, while cheaper low-end desktop systems would have lower trigger limits corresponding to their smaller heat-sinks.

The goal of dynamic thermal management is to reduce the number of cycles in which the processor's temperature exceeds a certain value. In our simulation environment, we say that a "thermal emergency" occurs if the moving average of full chip power dissipation for the past 10,000 cycles exceeds the pre-set thermal emergency wattage value.

In deciding what values to use for a pre-set thermal emergency value, we gathered the data shown in Figure 2. For each cycle in the simulation, the average of the power dissipation for the past 10,000 cycles is generated. We will refer to this as the 10k moving average of power dissipation. We use power integrated over time as an abstract proxy for temperature. The data in Figure 2 shows the average (for each cycle of the entire simulation) of the 10k moving average of power dissipation. Thus, this roughly compares to the processor's average temperature for the entire benchmark run. The average 10k moving average of power dissipation for the benchmarks was 24.3W with a standard deviation of 2.41W.

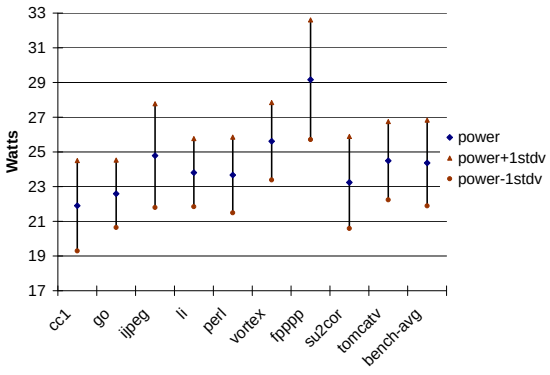

Figure 2: Average of the 10k moving average and 1 standard deviation range.

We present analysis for the case where the thermal management is triggered when the 10k moving average exceeds 24W and a full-fledged thermal emergency is considered to occur when the 10k moving average exceeds 25W. These value were chosen to correspond to the high-range of the benchmarks, but future work will investigate the effects of different trigger and emergency points. We have also neglected *compress* and *m88ksim* from the analysis because neither application had any cycles exceeding the chosen emergency point.

As a point of reference, we have also recorded the maximum power dissipated by the processor on any given cycle for the benchmarks. For almost all of the benchmarks, this maximum power was roughly 2x the average power, at 45.8W.

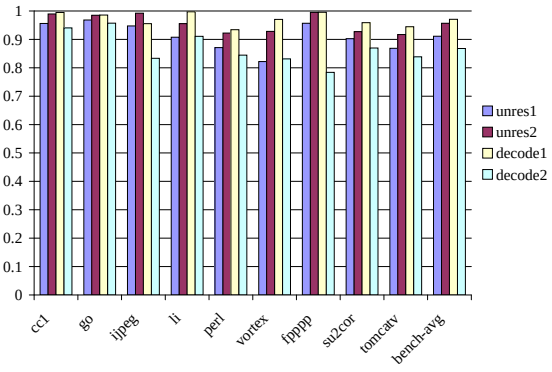

Figure 3: Reduction in IPC compared to baseline.

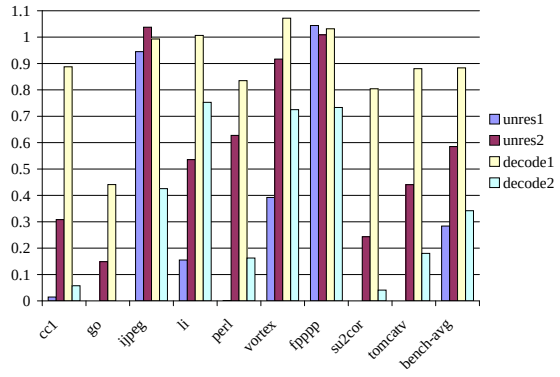

Figure 4: Reduction in thermal emergencies compared to baseline.

Figures 3 and 4 show the overall program IPC reduction and the reduction in thermal emergency cycles from the baseline for a selection of the benchmarks. Here we consider two different thermal management strategies. The first is speculation control and the *unres1* and *unres2* bars corresponding to the cases where the pipeline is restricted to 1 and 2 unresolved branches respectively. For comparison, we also consider pinching the pipeline at the decode stage, which is similar to the I-cache throttling utilized in the PPC. The *decode1* and *decode2* bars correspond to the cases where the decode bandwidth is reduced by 1 instruction and 2 instructions per cycle respectively.

The results show that speculation control is useful for some benchmarks. For example, by limiting the number of outstanding branches to 1 in *cc1*, the percent of cycles that had thermal emergencies drops from 24% with no thermal management to around 0.3% with thermal management. This reduction is achieved with only a 4% performance degradation. A 1% performance loss is incurred by limiting the number of unresolved branches to 2, and the number of cycles with thermal emergencies is reduced to 6.9%. On the other hand, reducing the decode bandwidth by 2 incurred a 6% performance degradation in order to reduce the number of thermal emergencies to 1.2% of the total cycles.

Other benchmarks, especially *li*, *su2cor*, and *tomcatv*, performed better with speculation control than with decode bandwidth reduction. Across the benchmarks, 1-unresolved branch speculation control reduced the number of cycles in thermal emergency to 28% of the baseline with a 9% performance degradation. Restricting the decode bandwidth to 2, reduced the number of cycles in thermal emergency to 34% of the baseline with a 14% performance loss. However, there are clearly some benchmarks such as *ijpeg* and *fpppp* where speculation control is not as successful as pinching the decode bandwidth.

In order to reduce the number of thermal emergencies to zero for all benchmarks, the trigger could be lowered, the thermal emergency limit could be raised, or more aggressive thermal management could be invoked if needed. We plan to study the range of trigger and emergency values in more detail in the future. We are also interested in studying combinations of the techniques and methods for dynamically selecting management techniques based on other processor statistics such as branch prediction rate or branch confidence.

4 Towards a More Algorithmic Approach: Correlation Between Power and Processor Statistics

In this work we have compared the benefits of dynamic thermal management via speculation control to pinching the decode pipeline. In considering other schemes for thermal management, we would like to develop a more systematic approach to identifying potential techniques. We have performed simulations using Wattch to correlate power dissipation with other processor statistics such as branch direction and address prediction accuracy, data and instruction cache hit rates, execution bandwidth, and IPC. We use this method to isolate certain processor statistics which track more closely with power than with IPC. That is, we wish to find levers that reduce power with a less-then-proportional reduction in performance.

We collected the average power and performance statistical data for fixed chunks of 10,000 cycles. These statistics were then correlated with each other after the simulation completed. Figure 5 shows the correlation between processor power dissipation and branch prediction accuracy, cache hit rates, execution bandwidth (comm + mis-spec insns/cycle), and IPC. As expected, power correlates very strongly with execution bandwidth and IPC for both integer and floating point applications. Branch direction and address prediction accuracy, secondary indicators of application performance, also correlate with power but to a lesser degree. Data and instruction-cache hit rates correlate slightly more than branch predictor accuracy with power. There are some obvious outliers to the general trend: *ijpeg* has negative correlation between power and branch predictor accuracy and *li* has negative correlation between data cache hit rate and power. The unexpected correlation data is most likely due to the fact that the application's bottleneck is not related to the statistics, so changes in that statistic have a small impact on the overall application behavior. The correlation data between IPC and the processor statistics for the benchmarks is shown in Figure 6.

Figure 7 shows the difference in the power correlation and IPC correlation data. These charts attempt to highlight applications and statistics which are more strongly correlated with power than with IPC. This may reveal strategies that would be most useful for dynamic thermal management. From the data it is clear that for the most part, the statistics correlate more with IPC than with power. For example, for all of the benchmarks, branch predictor accuracy correlated more with IPC than with power. However, execution bandwidth correlates more with power than with IPC for five of the benchmarks. This lends support to our decision to use speculation control as a means of dynamic thermal management. We plan future work that will broaden the types of management handlers we investigate with correlation analysis.

5 Conclusion

Dynamic thermal management holds promise for simplifying computer system design by reducing the max-

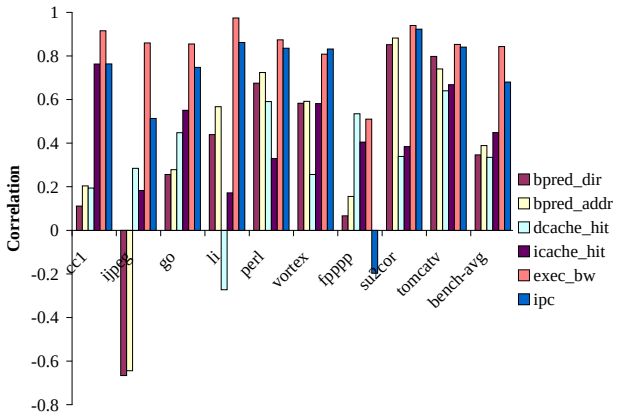

Figure 5: Correlation between power and several performance statistics.

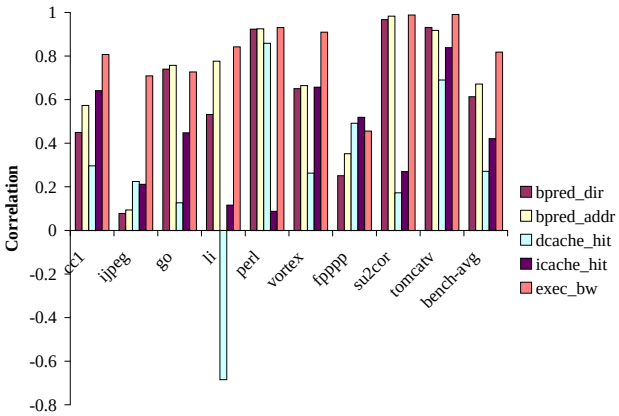

Figure 6: Correlation between IPC and several performance statistics.

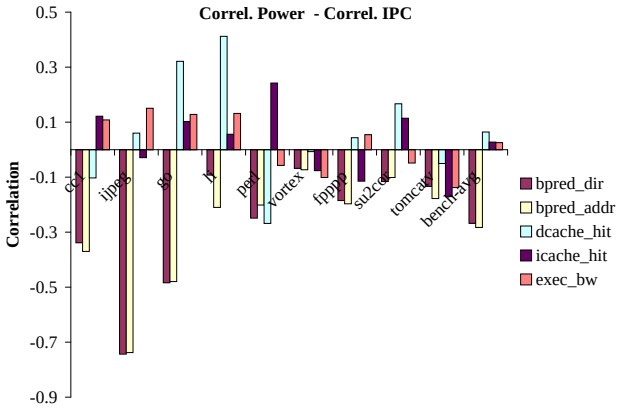

Figure 7: Difference between power and IPC Correlation.

imum power dissipation ratings to numbers that are much closer to the average power ratings. We have investigated the correlation of power and IPC with branch predictor accuracy, cache hit rates, and execution bandwidth. Speculation control has been identified as a useful thermal management technique for several benchmarks.

In the future, we feel that a selection of multiple techniques, perhaps invoked in combination or on an application specific basis, seems like a useful strategy for a hardware-based dynamic thermal management scheme. In micro-architectures with clustering or simultaneous multi-threading, clusters or threads could be disabled in response to thermal emergencies. Branch and value predictors could be disabled for applications that see only incremental benefits from these hardware structures.

A variety of software based dynamic thermal management schemes should also be considered at the compiler and operating system levels. A power aware compiler, for example, could generate multiple versions of a code sequences which are selected based on the current thermal conditions. At the operating system, the scheduling routine could interleave lower power processes or threads with higher power ones. Overall, systemic approaches to thermal and power management are an open research area, of which the work described here offers some initial steps.

Acknowledgments

This work has been supported by research funding from the National Science Foundation and Intel Corp. In addition, Brooks currently receives support from a National Science Foundation Graduate Fellowship and a Princeton University Gordon Wu Fellowship.

References

[1] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: A framework for architectural-level power analysis and optimizations. In *Proceedings of the 27th Annual International Symposium on Computer Architecture*, 2000.

[2] D. Burger and T. M. Austin. The SimpleScalar Tool Set, Version 2.0. *Computer Architecture News*, pages 13–25, June 1997.

[3] M. Gowan, L. Biro, and D. Jackson. Power considerations in the design of the Alpha 21264 microprocessor. In *35th Design Automation Conference*, 1998.

[4] D. Grunwald, A. Klauser, S. Manne, and A. Pleszkun. Confidence estimation for speculation control. In *Proc. of the 25th Int'l Symp. on Computer Architecture*, pages 122–31, June 1998.

[5] M. Pant, P. Pant, D. Wills, and V. Tiwari. An architectural solution for the inductive noise problem due to clock-gating. In *Proc. of Int'l Symposium on Low-Power Electronics and Design*, August 1999.

[6] P. Reed et al. 250 MHz 5W RISC microprocessor with on-chip L2 cache controller. *Digest of Technical Papers - IEEE International Solid-State Circuits Conference*, 40:412, 1997.

[7] H. Sanchez et al. Thermal management system for high performance powerpc microprocessors. *Digest of Papers - COMPCON - IEEE Computer Society International Conference*, page 325, 1997.

[8] V. Tiwari et al. Reducing power in high-performance microprocessors. In *35th Design Automation Conference*, 1998.

[9] Transmeta Corp. The Technology Behind the Crusoe Processor Whitepaper, 2000.