Coordinated Management: Power, Performance, Energy, and Temperature

Heather Hanson* Stephen W. Keckler

Computer Architecture and Technology Laboratory
cart@cs.utexas.edu - www.cs.utexas.edu/users/cart

Department of Computer Sciences
*Department of Electrical and Computer Engineering
The University of Texas at Austin

IBM Technical Contact : Rob Bell, Jr.

Abstract

We are developing a next-generation power manager that supports high performance within the constraints of limited power, energy, and temperature levels. To date, we have developed simulation infrastructure to model a technology-scaled version of an Alpha 21364 processor and have added three independent management techniques to the simulator model. One technique dynamically scales frequency and voltage settings. Another technique, pipeline throttling, restricts the rate of integer instruction issue. The third technique reduces leakage power in caches while retaining memory contents. We are currently using the simulation infrastructure to gauge the effect of management techniques. This report provides an overview of our research project and summarizes the current status of this work.

1 Introduction

Effective power management is crucial to sustain future generations of high-performance chips as the power liability per transistor grows and Moore's Law integration trends offer more transistors per die each generation. We are developing a next-generation processor manager that coordinates a multitude of management options to ensure safe operation while enabling high throughput and performance.

1.1 Coordinated Management

The coordinated manager's central component is a multi-criteria optimization algorithm that sorts priorities and balances conflicting goals for performance, power, energy, and temperature. The manager will select a goal, such as "maximum performance within fixed temperature and energy limits" and enable a coherent set of management mechanisms to achieve the goal. The closed-loop feedback system between the manager and on-die sensors and performance counters will provide continual updates on system status, allowing the manager to intelligently tune its directives to achieve the desired response. The manager will track system behavior and shift goal objectives in synchrony with changing application demands and energy resources.

A hierarchy of intelligence gathering and processing components in the manager will distribute decisions according to required response time: quick response for phenomena with shorter time constants, such as current spikes in the power distribution network, and longer intervals between decisions for slow-moving trends like gradual chip warming. With coordinated information from multiple sources and a goal-driven algorithm, the manager can adapt to the system environment and push the operating conditions to the edge of acceptable limits [5].

1.2 Infrastructure

The initial phase of the research project focused on building simulation infrastructure that models and monitors processor behavior under a range of management conditions. We extended our existing simulation infrastructure reported in [5] to include temperature dependence and a static power model and upgraded the cache system to include a large on-die L2 cache, with a power model based on the Alpha 21364. We built three independent management techniques into the simulator model. First, dynamic frequency and voltage scaling controls activity throughout the chip. A second technique, pipeline throttling, re-

stricts the rate of integer instruction issue, causing localized changes in integer execution units as well as downstream effects throughout the pipeline. The third technique, a cache sleep mode, reduces leakage power in caches while retaining memory contents. The cache sleep mode targets static power in the large portion of the die devoted to SRAM structures.

This document summarizes our current status in the infrastructure development for a next-generation power manager. Section 2 describes our simulation infrastructure in detail and Section 3 presents preliminary experimental data. Section 4 concludes this status report with a summary of completed research and our plans for future work.

2 Simulation Infrastructure

We have developed simulation infrastructure to estimate performance, power, energy, and temperature throughout program execution in order to evaluate the effects of management control.

2.1 Processor Model

The base simulator for this research project is `sim-alpha`, a validated microarchitectural performance simulator developed in our research group [3] that models behavior of the Alpha 21264 processor. We have extended the simulator with additional components to measure power and temperature. In previous work [7], we added the Wattch [2] dynamic power estimator to the sim-alpha simulator. We augmented the power models to match the Alpha 21264's components for separate load and store queues, 64-bit data path, I/O pins, a system interface, and low-power ALU result buses. For this project, we extended the model to include a large level-2 cache based on the Alpha 21364 processor [4].

For temperature estimates, we had incorporated the HotSpot [10] temperature estimator, which models thermal resistance and capacitance as electrical resistance and capacitance in a three-dimensional model of the chip. As the Wattch-based power estimator calculates power during a simulation, the HotSpot module reads the power consumed over a specified time period (10,000 cycles) to calculate the temperature for each unit in the chip's floorplan. We monitor each unit's temperatures to find the "hotspots" during program execution. It is possible that the hotspots are overestimated in this simulator version as a result of temperature rising too quickly. We are in the process of confirming the accuracy of coefficients and time constants for this system.

We built a static power model based on the 2003 ITRS projected gate and subthreshold leakage current for high-performance devices [6]. The static power model incorporates two types of transistors, a high-leakage/high-performance transistor and a low-leakage/lower-performance device. The high-leakage value corresponds to the *hp90* high-performance device specified in the ITRS 2003 roadmap and the lower leakage value is a fraction of the high leakage mode. Configuration parameters determine the percent of high/low leakage transistors in caches and logic units and can be varied to create a wide range of leakage conditions. The leakage power per microarchitectural unit is based on the unit's area and an estimate of transistor density, and is scaled with temperature to reflect greater leakage current at higher operating temperatures.

2.2 Simulated Processor Configuration

For baseline simulations, the simulator is configured to model a contemporary high-performance processor. Table 1 lists simulation parameters for the baseline configuration. The microarchitectural components are based on the Alpha 21364 processor, with modifications to the memory hierarchy. The Alpha 21364 was designed to use RAMBUS memory; the simulator uses a standard DRAM model. The simulated L2 cache is 8-way set associative cache with 1MB capacity, rather than the 1.75MB 7-way cache design of the Alpha L2 cache. Both the L2 and memory adaptations fit trends for high-performance processors in 90nm technology [8] [1].

Technology parameters are derived from the ITRS *hp90* high-performance 90nm node [6]. We linearly scaled HotSpot's 21364 floorplan based on 130nm technology to produce a 90nm equivalent chip.

2.3 Management Policies

We incorporated three management techniques into the simulator, as listed in Table 2. Simulation configuration flags enable and disable each technique for the duration of a simulation. When enabled, each technique is triggered independently by events during program execution.

First, dynamic voltage and frequency scaling (DVFS) alters the supply voltage and operating frequency, which directly controls power and energy consumption throughout the chip. The voltage and frequency settings in the model are independent, though the manager model currently sets voltage-frequency pairs together for simpler control. The DVFS manager raises or lowers the voltage and frequency when maximum and minimum temperature thresholds are breached.

The second management technique is a form of pipeline throttling in which the integer issue stage maximum rate varies between 1 and 4 instructions per cycle. When the issue rate is throttled below 3 instructions per cycle, one integer subcluster is effectively turned off with power ac-

Table 1. Baseline Simulator Configuration

Parameter	Value	Notes
V_{DD}	1.2 volts	
Clock rate	4 GHz	
Leakage current–high leakage	6.6e-6 W/μ m	alternate high-leakage scenario: 3e-6 W/μ m
Leakage current–low leakage	6.6e-7 W/μ m	10% of high leakage
# integer clusters	2	each cluster: 2 ALU units, 1 register file
# floating-point clusters	1	2 ALU units, 1 register file
front-side (system) bus	400 MHz, 64 bits	6 GB/sec bandwidth
main memory	DDR2 400 MHz	reference [8]

Table 2. Management Policies

Technique	Action	Simulator Settings
DVFS	modulate frequency and voltage settings (select from V,f pairs)	{600MHz, 0.6V} {1GHz, 0.8V} {2GHz, 1.0V} {4GHz, 1.2V} {6GHz, 1.6V}
pipeline scaling	turn on/off portions of integer pipeline	1, 2, 3, or 4 integer operations
cache sleep	automatically transition idle cache lines into sleep mode	- idle time before sleep mode - time to wake up sleeping lines

counting to eliminate clock and leakage power, though the microarchitectural model continues to use pre-assigned subclusters for instruction execution. The pipeline throttling manager raises or lowers the issue stage processing rate when minimum or maximum temperature thresholds have been exceeded.

The DVFS and pipeline width techniques use the same temperature thresholds as trigger points, with varying effects. The DVFS technique controls dynamic and static power for the full chip, while the pipeline throttling policy directly affects the integer execution subclusters and indirectly influences the remainder of the pipeline due to changing activity rates in the integer units.

The third technique, a cache sleep mode, controls the amount of leakage current in a cache. When the mode is enabled, it automatically transitions idle cache lines to a low-leakage mode that preserves cell contents. When a sleeping cache line is needed, the cells are restored to an active state after a wakeup transition time. The length of idle time and the wakeup time are determined by simulation configuration parameters for each cache.

We created a reporting mechanism to the simulator to record the current settings for voltage, cache sleep mode, pipeline width, power, energy, temperature, and performance during simulation. The statistics are recorded every 10,000 cycles and printed in the simulation's output file upon program completion.

3 Simulation Results

Figures 1 and 2 illustrate system behavior throughout program execution. The plots show the results of a 100 million instruction EIO (external input/output) trace simulation of the Spec2000 integer benchmark `gzip`. The traces are composed of program segments identified with SimPoint [9] to create a reasonably sized input set that represents benchmark behavior. EIO traces encapsulate all communication with the operating system in the trace file, creating identical conditions for each simulation. Differences in simulation output will be the result of simulator behavior, not due to variable responses to operating system calls.

The strip charts show the following measurements as a function of time in msec (from top to bottom):

1. performance in terms of the total instructions completed and instructions per second
2. maximum hotspot temperature on die
3. accumulated energy throughout simulation

4. average power during 10,000-cycle epochs
5. supply voltage setting
6. pipeline width setting (maximum number of integer instructions issued per cycle)
7. percentage of L2 cache lines in sleep mode

3.1 Baseline Case

Preliminary data for the baseline experiment of the system with no power management are shown in Figure 1. The supply voltage is set to 1.2 volts and the pipe width is fixed at 4 integer instructions. Cache sleep mode is disabled; the SRAM cells are composed of high-VT, low-leakage transistors but the cache does not benefit from the additional leakage reduction of sleep mode. Throughout execution, instructions commit at a steady rate, average power oscillates near 50 Watts, total energy accumulates correspondingly, and the maximum temperature gradually increases.

3.2 Management-Enabled Case

The second case enables all three management techniques independently. Voltage and frequency levels and the pipeline width are controlled by thermal monitoring units that are set to the same trigger points. The controllers will lower the settings if the maximum temperature exceeds 355 K (82 C) to allow the chip to cool, then increase the settings if the temperatures drop below 345 K (72 C) to spur performance. With the narrow temperature range, DVFS and pipeline width settings will continuously adjust with the intent to let the simulator run at the best possible performance levels within a safe thermal margin such that hotspots on die never exceed 85 C.

The sleep mode controller does not use temperature levels; the sleep mode policy automatically transitions cache lines into a low-leakage mode after 1,000 idle cycles, and charges an extra 25 nsec time penalty to wake up sleeping lines before use.

The first portion of the management-enabled experiment is shown in Figure 2. In this simulation, very few lines are awake at any point in time and the L2 cache maintains a minimal-leakage state throughout the execution time, regardless of the other settings. The pipe width and voltage-frequency settings adjust as the temperature fluctuates between the minimum and maximum thresholds. Every DVFS change causes a 1 μs stall, temporarily suspending pipeline operation for the voltage change. The voltage and frequency levels start with the same initial conditions as the baseline case, then ramp up while the chip is below the minimum temperature threshold. After the hotspots have exceeded the maximum threshold, the voltage-frequency settings step back down a maximum rate of one setting change per millisecond, until the chip has sufficiently cooled and the cycle begins again. The pipe width controller is set to step between 2 and 4 in this configuration. Figure 2 illustrates the cyclic nature of the temperature fluctuations and corresponding management setting adjustments. The average power graph reflects the periodic nature of operation between setting changes. The rate at which instructions commit is also periodic, although overall lower than the baseline case due to the lower voltage settings. The rate of instructions committed per second declines with the DVFS setting.

The management-enabled system does prevent temperature violations but dramatically extends the execution time, completing 100 million instructions in about 55 msec. In contrast, the baseline scenario successfully completed the program segment in about 17 msec.

3.3 Management Control

Neither the baseline nor the independently-enabled configuration produces a satisfactory balance of performance, power, energy, and temperature. At one extreme, the uncontrolled temperature could present reliability problems; at the other extreme, the excessive measures to ensure safe temperature could cripple performance, causing a server to violate quality-of-service agreements or a cutting-edge desktop computer to perform worse than its predecessors.

Clearly, there is room for improvement in the control and judicious application of the management techniques. One solution is to empirically determine the best fixed management "knob" settings for a given program. For the `gzip` benchmark, we found the fixed configuration of 2GHz frequency, 1.0 volt supply voltage, pipe width of 2, and the L2 cache sleep mode enabled provided the minimum execution time without violating a maximum-allowable temperature limit of 358 K (85 C). Figure 3 shows the simulation results for temperature, total completed instructions throughout execution, and the rate of instructions committed per second. The program completed in about 48 msec.

If all program behavior were constant like the `gzip` benchmark and known *a priori*, the best static settings could provide effective system control. We expect that a dynamic management algorithm could further improve the control. A dynamic, online algorithm could converge to the best static settings for constant behavior, and for programs with time-varying behavior, a dynamic algorithm should be able adapt to find the best settings for each distinct epoch within the execution.

We have simulated many permutations of management techniques enabled to trigger automatically, as well as voltage-frequency and pipe width settings fixed throughout program execution. By exploring behavior across a spectrum of management choices, we can gather information on

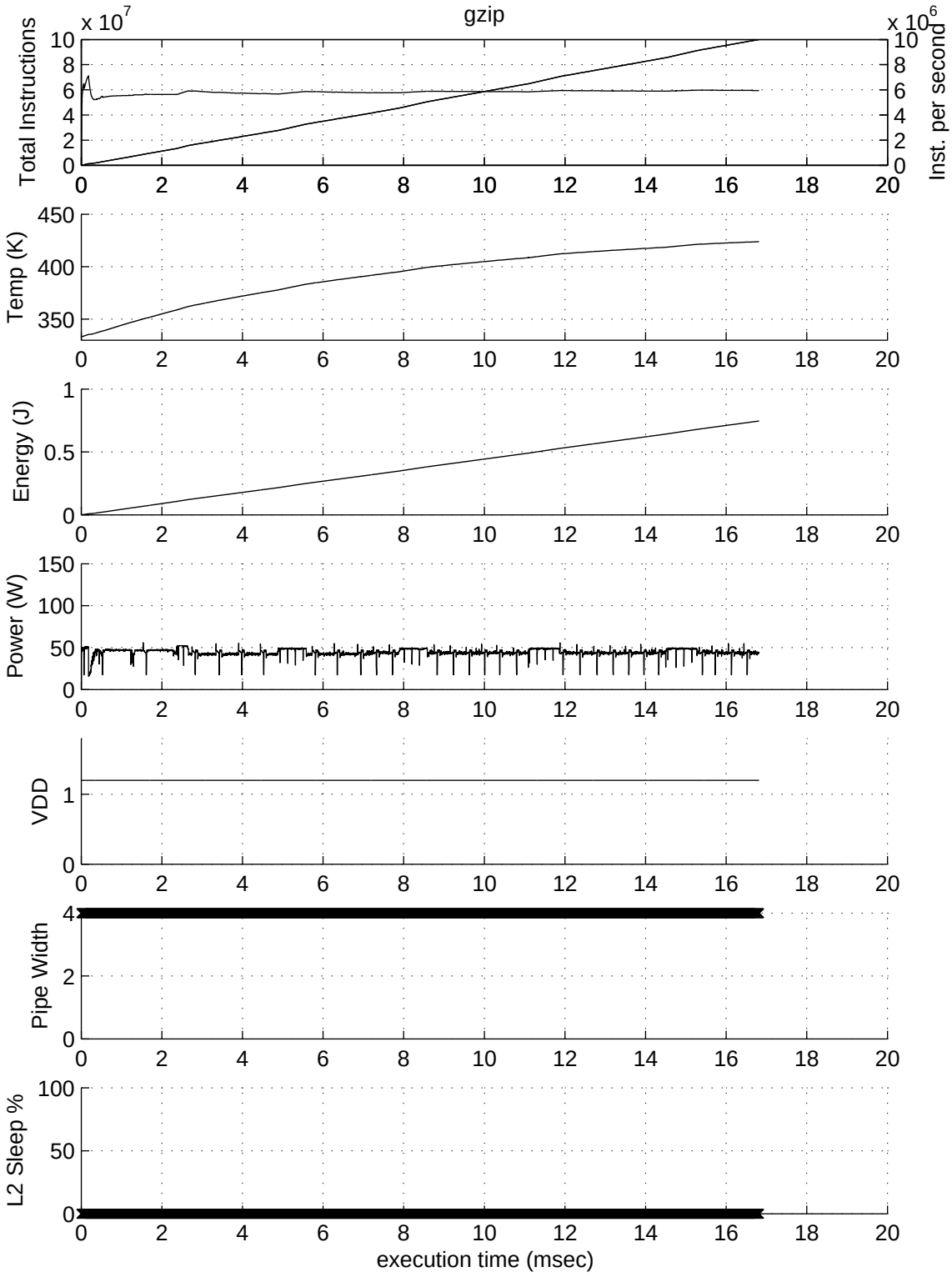

Figure 1. Baseline Configuration

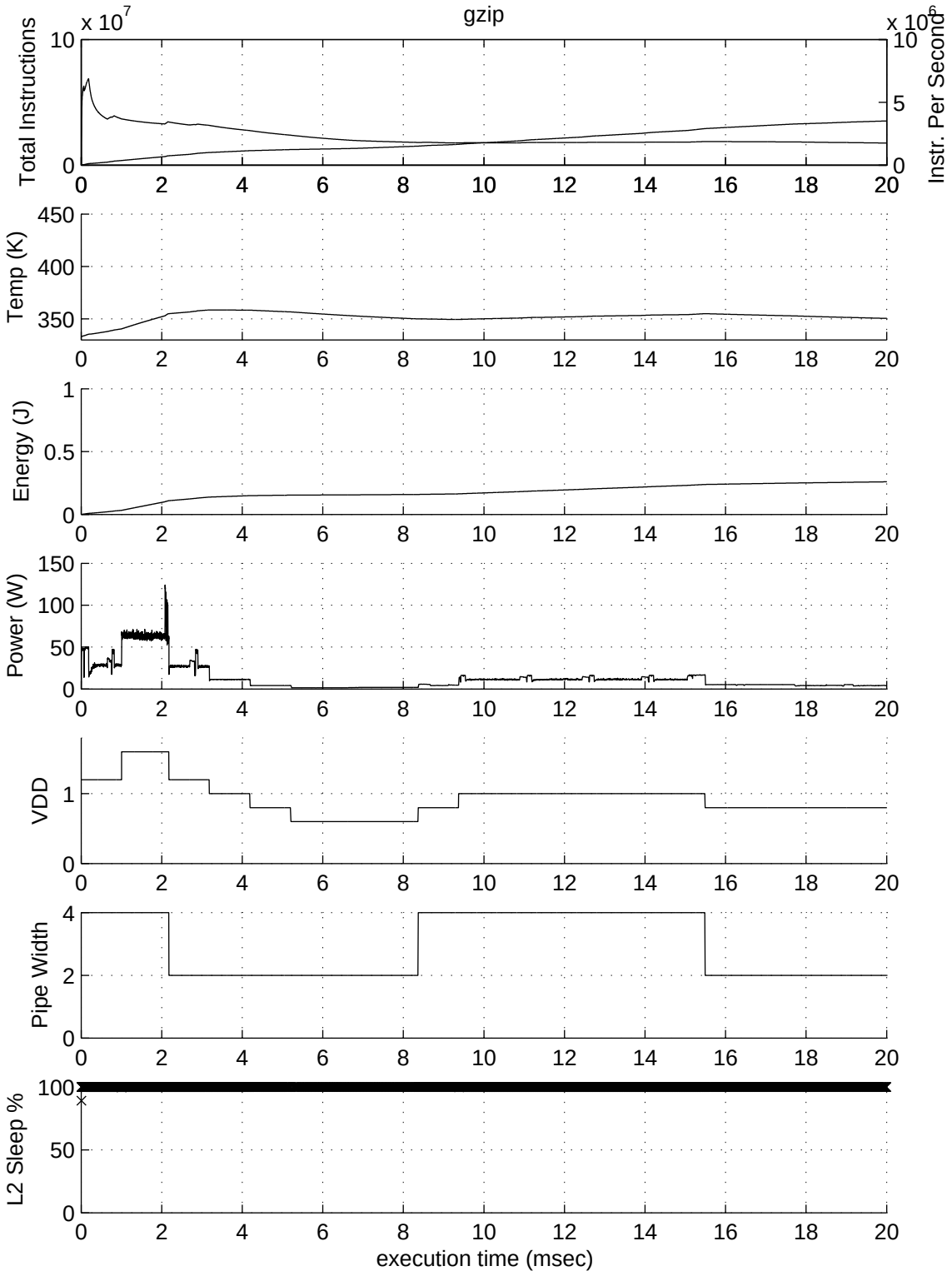

Figure 2. Independently Controlled Management Techniques

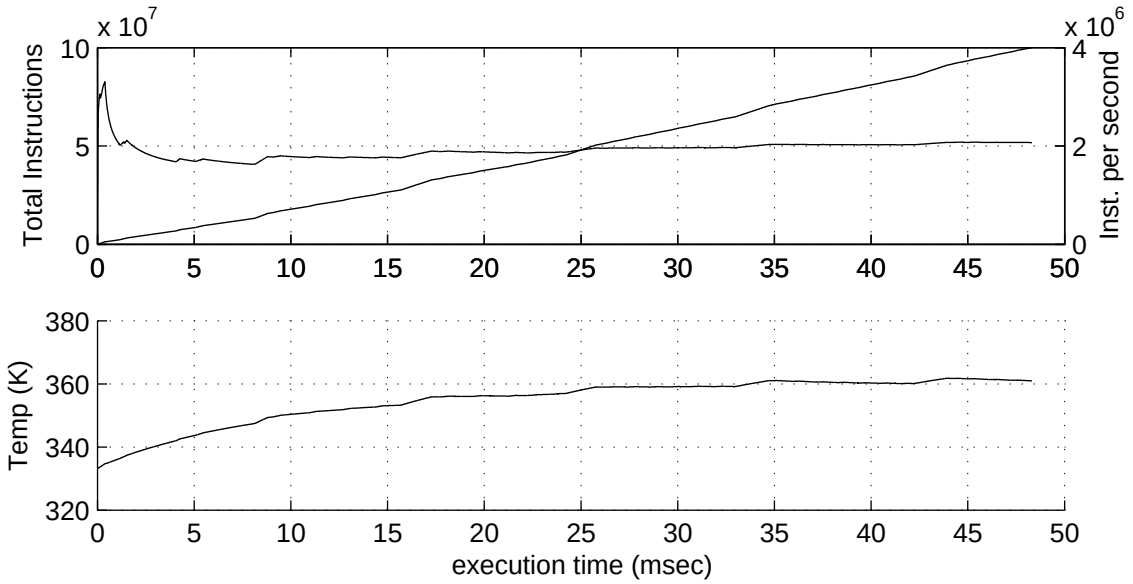

Figure 3. Best Fixed Setting

the system response for a wide range of conditions. We will then use the database of simulation data to build a table of cost functions and range of effectiveness for each technique. The cost functions will include performance penalties, such as additional cache latency due to sleep mode or a pipeline stall to change voltage levels, as well as energy, power, and temperature costs associated with applying the technique. Static algorithms for the coordinated manager design will use the simulation database as an off-line method to choose knob settings; more sophisticated dynamic algorithms will use the database to determine initial conditions and continue to update the cost function table during operation.

4 Conclusion

We developed microarchitectural simulation infrastructure to estimate performance, power, energy, and temperature throughout program execution. The processor model is based on a 90nm version of the Alpha 21364 and is equipped with three independent management techniques:

- dynamic frequency and voltage scaling (DVFS) changes the operating speed and total power for the full chip
- pipeline throttling restricts the rate of integer instruction issue to reduce dynamic power, and also eliminates leakage power by disabling a subcluster of the chip's datapath for narrow pipe-width settings
- cache sleep mode reduces leakage power in caches while preserving data

This report shows simulations the `gzip` benchmark to illustrate scenarios of the processor operating with no management, uncoordinated dynamic management, and empirically selected fixed management settings.

The goal of our current research phase is to show the opportunity for a next-generation power manager in a high-performance processor. By finding an optimal, or near-optimal, sequence of management settings for execution of each benchmark in the suite, we can determine an upper bound for "perfect" management. We expect the bound to be significantly better than un-coordinated management techniques working independently. As we develop the coordinated manager's algorithms, we will evaluate their effectiveness in comparison with both the upper bound and contemporary management approaches.

References

[1] Intel Xeon processor. Intel Corporation.

[2] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: a framework for architectural-level power analysis and optimizations. In *Proceedings of the 27th Annual Symposium on Computer Architecture (ISCA)*, pages 83–94, 2000.

[3] R. Desikan, D. Burger, and S. W. Keckler. Measuring experimental error in microprocessor simulation. In *Proceedings of the 28th Annual Symposium on Computer Architecture*, pages 266–277, 2001.

[4] J. Grodstein, R. Rayess, T. Truex, L. Shattuck, D. Bailey, D. Bertucci, G. Bischoff, D. Dever, M. Gowan, R. Lane, B. Lilly, K. Nagalla, R. Shah, E. Shriver, S. Yin, and S. Morton. Power and CAD considerations for the 1.75mbyte,

1.2ghz L2 cache on the Alpha 21364 CPU. In *Proceedings of the 12th ACM Great Lakes Symposium on VLSI*, pages 1–6, 2002.

[5] H. Hanson, S. W. Keckler, and D. Burger. Coordinated power, energy, and temperature management for high-performance processors. In *Proceedings of the 5th Annual ACAS Conference*, 2004.

[6] International technology roadmap for semiconductors (ITRS), 2003 edition.

[7] K. Natarajan, H. Hanson, S. W. Keckler, C. R. Moore, and D. Burger. Microprocessor pipeline energy analysis. In *ISLPED*, pages 282–287, 2003.

[8] R. Radhakrishnan, R. Ali, G. Kochhar, K. Chadalavada, R. Rajagopalan, J. Hsieh, and O. Celebioglu. Performance characterization of BLAST on Intel Xeon and Itanium2 processors. In *Proceedings of the IEEE 7th Annual Workshop on Workload Characterization (WWC-7)*, pages 81–88, 2004.

[9] T. Sherwood, E. Perelman, G. Hamerly, and B. Calder. Automatically characterizing large scale program behavior. In *Tenth International Conference on Architectural Support for Programming Languages and Operating Systems*, October 2002.

[10] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. Temperature-aware microarchitecture. In *Proceedings of the 30th International Symposium on Computer Architecture*, pages 2–13, 2003.