TEAPC: Temperature Adaptive Computing in a Real PC

Augustus K. Uht and Richard J. Vaccaro
University of Rhode Island
Microarchitecture Research Institute
Department of Electrical and Computer Engineering
4 East Alumni Ave.
Kingston, RI 02864, USA
{uht@ele.uri.edu, vaccaro@ele.uri.edu}

Abstract

TEAPC is an IBM/Intel-standard PC realization of a CPU temperature-adaptive feedback-control system. The control system adjusts the CPU frequency and/or voltage to maintain a constant set-point temperature. TEAPC dynamically adapts to changing CPU computation loads, as well as any other system phenomenon that affects the CPU temperature. For example, with the specific TEAPC hardware, should the CPU cooling fan fail, the control system detects the increase in CPU temperature and automatically reduces the CPU frequency and voltage to keep the CPU from overheating. In such a circumstance the system continues to operate. All of the adaptation is done dynamically, at runtime, on an unmodified standard operating system (Windows 2000) with a purely software-implemented feedback-control system.

1. Introduction

With today's high-end processors dissipating 10's of Watts, the CPU may become overheated. Overheating leads to reduced reliability at the least, and CPU damage and system failure at the extreme. Overheating is not often a show-stopping problem in desktop systems (Intel Prescott CPU's excepted). However, it is of great concern in commercial systems such as server farms. By adjusting the temperature of such systems, or maintaining them at suitably low temperatures, reliability should increase markedly. Also, power consumption should decrease, and system efficiency should increase. System temperature control begins with the control of individual processors.

Adaptive computing systems achieve the above characteristics, and can enhance performance [11, 12]. They "optimize" their operation for changes in the environment (e.g., temperature), operating conditions (e.g., supply voltage) and manufacturing conditions (e.g., production run quality).

This work was partly presented in a prior, private venue [13].

Herein we present an adaptive prototype, TEAPC, based on the standard IBM/Intel PC architecture and realizing *underclocking*, that is, operating the system greatly below its specified frequency. We have implemented the entire TEAPC control system in software, in a Windows application. It runs in a normal application mode on Windows 2000, multitasking normally with other applications. CPU frequency and core voltage are changed dynamically, based on the output of a feedback control system using the CPU chip temperature as its input. The control loop and underclocking models should be usable for any PC to be built, as well as many that already exist.

This paper's contributions are as follows: Demonstration of temperature control of a COTS (Commercial Off-The-Shelf) based PC via a modern feedback control system; On-demand reliability and performance control; Reduced power consumption; Fine-grain frequency (about 0.2% increments) and voltage transitions, allowing a continuously active CPU (no lost cycles); Adaptation to the current CPU computational load and other conditions; and Underclocking for disaster tolerance. No operating system or hardware modifications are needed.

In the rest of the paper, Section 2 discusses prior work. Section 3 describes the TEAPC feedback control system. In Section 4 TEAPC's implementation is discussed, along with the experimental methodology. Section 5 presents the experiments, data, and analyses. We conclude in Section 6.

2. Prior Work

Performance-enhancing [7] and power-reducing [4] adaptive computing has only appeared in the last decade or so. Voltage- and frequency scaling are the usual methods employed. A survey is in [1]. Many thermal approaches and applications are presented in [9], including those for large multiprocessor systems. Most frequency-scaling is coarse-grained [3, 5, 6].

Skadron *et al* [8] first considered the use of formal feedback control theory as applied to the on-chip control of chip temperature of a microprocessor.

In [10] chip temperature was regulated without formal control theory since the relevant thermal delays and changes in frequency and temperature were small.

Operating-system based approaches to thermal control [5, 15] propose to coarsely throttle the frequency, e.g., HLT or full speed, based on either task-level activity or CPU-counter based activity, in one case using open-loop temperature control [15]. Therefore, undesirable OS modifications are necessitated, a CPU could be shut-down for many seconds negatively-impacting critical applications, and a CPU could overheat due to inaccurate temperature estimation. The peak performance is unchanged, remaining at the worst-case design point.

Systems such as SpeedStep [3] coarsely adjust both frequency and voltage to keep temperature within bounds, but CPU dead-times of many microseconds are required for the new settings to stabilize.

3. The Feedback Control System

The feedback control system was designed using formal control system theory to achieve a fast but stable frequency response to thermal changes. The basic control input is a temperature setting, Tset, used by the control system to maintain a CPU temperature of Tset degrees C. The single sensor input is the temperature reading obtained via the CPU's embedded temperature-sensing thermal diode. (The diode is part of the CPU chip itself.) (Multiple sensors were used initially, but their control system requirements were excessive for this stage of our work.) Tset can be made equal to a temperature corresponding to a point above the maximum CPU frequency under full load, so as to maximize performance, or, more typically, Tset is kept somewhat lower to save power or increase the reliability of the CPU. Tset can be any value as long as it is below the maximum temperature spec for the CPU. Although our studies essentially kept Tset constant during an experiment, it may easily be dynamically modified for more system flexibility.

The feedback control system was designed with state-of-the-art discrete-time techniques as an integral control system using state-space methods [2, 14]. The system was modeled from measured open-loop input/output data of TEAPC using the System Identification Toolbox (from the Mathworks). The Toolbox returned a second-order model with two real-valued poles (i.e. two time constants). This model was approximated with a first-order system model by keeping only the dominant time constant.

We made this approximation for two reasons. First, the dominant time constant was likely a reliable indicator of the behavior of the system under a range of operating conditions. Second, for a first-order system, there is only one state variable, which can be taken as the system output. The control system also contains an integrator (accumulator for clock frequency increments) and thus, the overall system is second order with both state variables measured.

The two gains for a full-state feedback system are K_{FA} and K_G, shown in Figure 1. In effect, the K_{FA} feedback block magnifies the transient behavior of the input data (temperature) to speed up changes in the output (frequency) so as to decrease the response time of the overall system. The gains were calculated to give a 10-second settling time to a step change in Tset.

As in any integral control system, care must be taken whenever there is saturation (e.g. clock frequency at maximum or minimum limits). We deal with this as follows: whenever a calculated frequency increment would change the CPU frequency to a value beyond its limits, we put the frequency at its limit and turn off the integral action (that is, we do not accumulate the frequency increment). If the value of Tset is not attainable, the clock frequency will be pegged at one of its limits (upper or lower).

The resulting control system has excellent response characteristics, having a frequency settling time of only about 10 seconds for a Tset change equivalent to a change in frequency from 2.5 GHz to 3.5 GHz of the CPU under full load.

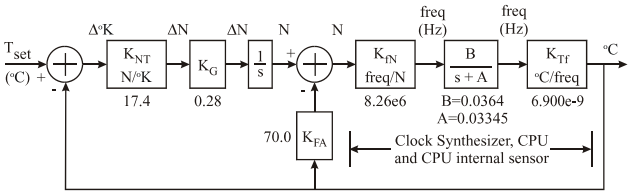

Figure 1. TEAPC feedback control system.

4. Prototype and Experimental Setup

The internal block diagram of TEAPC is shown in Figure 2, with its major relevant components listed in Table 1. Figure 3 is a photograph of TEAPC, including the display.

To show the wide applicability of the results, we made TEAPC's foundation a standard IBM/Intel PC architecture built primarily out of both hardware and software COTS parts. Therefore, the results of this work can easily be applied to commercial PC's during their design, construction and/or testing, and in some cases even after their manufacture.

We wished to be conservative, realizing a practical system, not a limit study. Thus, TEAPC was built out of harder-to-control high-end parts (at the time): a 3.0 GHz Intel Pentium 4 microprocessor with an 800 MHz bus and an Intel 875P chipset (a "chipset" is the glue logic that connects the CPU to the main memory and I/O devices; in Figure 2 it is the

Northbridge/Southbridge pair of chips). The main memory is high speed DDR (Dual Data Rate) dynamic RAM. (Note: from now on, 'frequency' refers to the CPU's frequency.) The frequency indirectly controls the Northbridge's and memory's frequencies. No other sub-systems' frequencies or voltages were varied.

The new TEAPC components consist solely of software; existing hardware is used. The new `teapc` program, for TEAPC control, is written in C and C++. It is a normal Windows user application, constructed with the Win32 standard API (Applications Program Interface). `teapc` is 800 Kbytes long, including the control system. Typically, `teapc` uses less than 1% of the CPU's time, not including display overhead; little or no performance is lost.

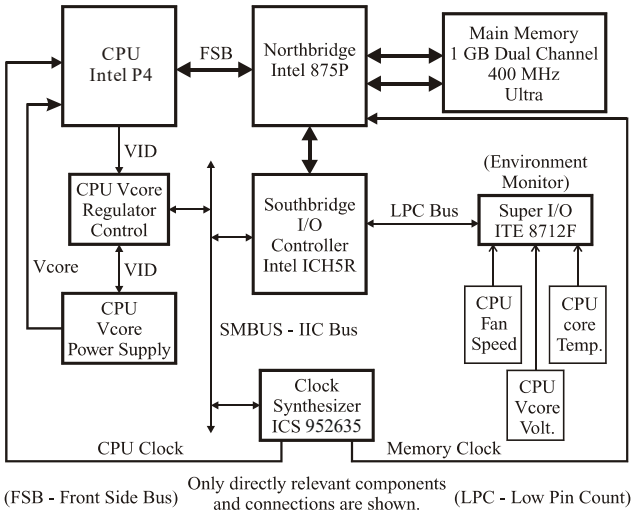

Figure 2. Major relevant motherboard structures used in TEAPC.

Table 1. Major TEAPC components.

PC Component	Manufacturer	Part Number/Description
Motherboard	Gigabyte	GA-8KNXP (Rev. 2); w/DPS regulator
CPU	Intel	P4 3.0 GHz, 800 MHz bus
Chipset	Intel	875P, ICH5R
Clock Synthesizer	ICS	ICS952635
Super I/O (Environment Monitor)	ITE	IT8712F V0.6
CPU Voltage Regulator Control	ITE	IT8206R V0.1
Main Memory	Ultra	U10-5903R; 2 x 512 MB; 400 MHz DDR, Dual Ch. (Operated at 320 MHz.)
Operating System	Microsoft	Windows 2000 SP4, HT disabled
Disk System – RAID 0+1	ITE	GigaRAID IT8212F
Disks	Maxtor	4 x 6E040L0, 40 GB, 133MHz IDE

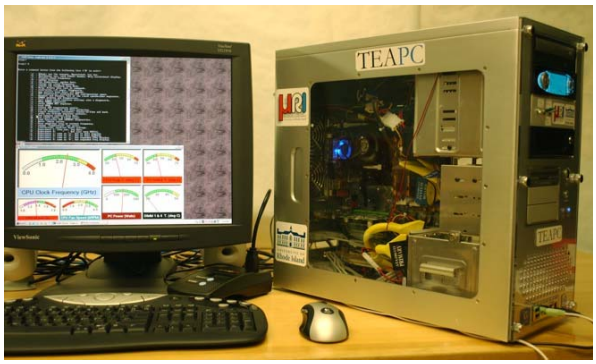

Figure 3. TEAPC prototype, with instrumentation shown on the display.

Referring to Figure 2, `teapc` reads the CPU's core temperature, core voltage, and fan speed from the Super I/O chip. The program reads and sets the CPU frequency by accessing the Clock Synthesizer, and reads and sets the CPU core voltage via the Vcore Regulator Controller. The Synthesizer and Regulator are accessed with the two-wire SMBUS through the Southbridge. `teapc` accesses all components, including the chipset, via the x86 I/O address space.

(In industry, sensor reading and device control are being standardized in the Common Information Model {CIM}. However, manufacturers are given wide latitude to implement their own proprietary specifications, keeping the data inaccessible to third parties. Therefore, these parties resort to reading part numbers off of the chips on the motherboard, finding the parts' datasheets, and writing their own device drivers. While painful, this is certainly doable, and is not a problem for the PC manufacturers themselves.)

In operation, `teapc` updates the feedback control loop every second. The control loop is represented in the program by a list of the non-CPU blocks shown in Figure 1. Every update the program starts with the input data (the running-averaged CPU temperature, the latter measured once a second), and re-calculates the control block values around the loop. The new CPU frequency is the main output; the CPU core voltage is sometimes linked to the frequency. For safety, the maximum and minimum possible output frequencies are hard limits; when reached, we say the frequency has "pegged" (as in an old analog meter). In all of the experiments, the frequency's change with respect to time was solely determined by the control system's dynamics. Being a normal application, `teapc`'s operation is often pre-empted by higher-priority programs. Such pauses are not an issue since the thermal dynamics are so slow (seconds).

A simple on/off switch was added solely for the disaster tolerance experiment. Also, an external power meter measured changes in total PC power consumption, again solely for the experiments.

5. Experiments

These studies characterized TEAPC's operation and showed its achievement of the project's goals. Only the CPU frequency was directly varied by the control system. In some cases the CPU core voltage was varied as a square-root function of the frequency, in order to potentially simplify future control system analysis and design (the temperature is thus a function of the square of the frequency, not the cube).

In general, the main results in Figure 4 and Figure 5 show that although there is some oscillation in the dependent variables, it is not great. The CPU temperature may still oscillate, even with a constant CPU frequency; this is mainly due to load changes in the CPU-100%-loading burn-in program (SiSoft's Sandra). Prior to obtaining these results, the system's temporal characteristics were briefly examined.

5.1. Step Response to Frequency Change

Table 2 shows TEAPC's CPU temperature's non-linear reaction speed to changes in CPU frequency. Overall, the CPU neither heats up or cools very quickly. The settling time varies substantially with load and direction of frequency change. It is easy to heat up the CPU, but takes considerably longer to cool it, regardless of the CPU's load. This is intuitive, as it is usually the case in thermodynamics that forced addition or removal of energy to a system will heat up or cool the system, respectively, faster than with a passive transfer mechanism. In TEAPC's case the increasing frequency directly increases the CPU's power consumption and temperature, whereas removal of the resulting heat depends on the thermal resistance and capacitance of the passive cooling system.

The response times approximately doubled from a full load condition to an unloaded condition. This may arise from the internal CPU power control system, which shuts down major sections of the CPU when the sections are not used. Hence, in a lightly loaded system a given increase in frequency increases power consumption less than a fully loaded system; in the latter, the entire chip is affected by the frequency change, decreasing the response time.

5.2. Basic Operation

Figure 4 and Figure 5 show the basic operation of TEAPC and its control system. The latter senses changes in the CPU temperature and adjusts the frequency and/or voltage to counter the changes. This results in a relatively stable CPU operating temperature, centered around the Tset value.

Table 2. Step response of CPU temperature to frequency changes, under differing loads.

Run ID	CPU Utilization	Frequency Transition (GHz)	Start Temp. (deg. C.)	End Temp. (deg. C.)	Settling Time (2%) (sec.)
91	100%	3.5 to 2.5	59.3	53.7	90
92	100%	2.5 to 3.5	52.5	58.9	60
93	~5%	3.5 to 2.5	53.1	48.0	170
94	~5%	2.5 to 3.5	48.0	52.9	130

The power's staircase shape comes from the time-varying execution profile of the CPU-loading program. Overclocking headroom is exploited.

TEAPC dynamically and automatically adapts to computation load changes. In Figure 4 the system is initially stable at the upper peg point of 3.5 GHz, and under no load. At the point shown, the CPU-burn program is started, putting the CPU under full load. The control system senses the rapid increase in CPU temperature, and immediately lowers the frequency. Within about 50 seconds TEAPC's frequency drops to 2.65 GHz, on average, while the CPU temperature stays relatively constant. The power consumption stays about the same. In this case, the CPU core voltage was held constant at its high value, 1.5125 V.

We performed the same experiment, but with the voltage a function of frequency as described earlier; see Figure 5. The lower voltage and a constant temperature allow a higher final operating frequency, about 3.0 GHz, with no change in power consumption. Thus, the voltage-to-frequency linking results in higher performance for the same power consumption.

5.3. Disaster Toleration

In this experiment, the CPU's cooling fan is turned off while the CPU is under full load and at its "maximum" frequency. The fan's airflow through a passive heatsink lowers the overall thermal resistance, thus increasing the heatsink's dissipation abilities. Hence, stopping the fan is a disastrous condition, and an excellent test of TEAPC's adaptation abilities.

Figure 6 shows TEAPC's response first to fan 'off' and then fan 'on' conditions. Initially, the CPU ran at its "maximum" frequency of 3.50 GHz and a corresponding core voltage of 1.5125 V., and while under full load. The CPU's core voltage was linked to the CPU frequency. As is shown in the figure, shortly after the CPU's fan was turned off the CPU temperature began to rise, causing the control system to lower the frequency, and thus also its core voltage.

Since the CPU's temperature never dropped below the Tset value of 56 degrees C., the control system dropped the CPU's frequency all the way down to its "minimum" value of 1.1 GHz and the corresponding minimum core voltage of 1.0875 V.

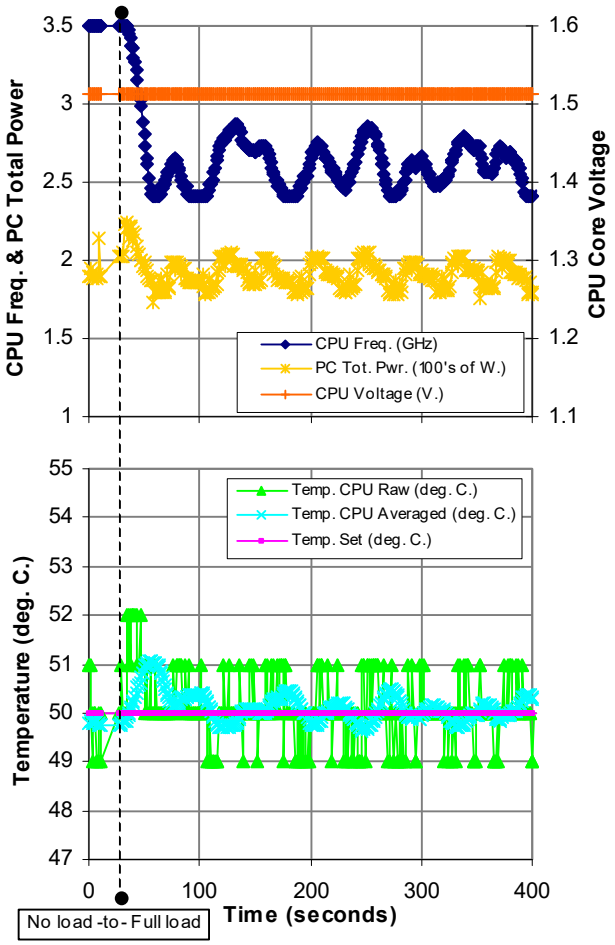

Figure 4. Basic operation and load adaptation test. Vcore NOT linked to CPU frequency.

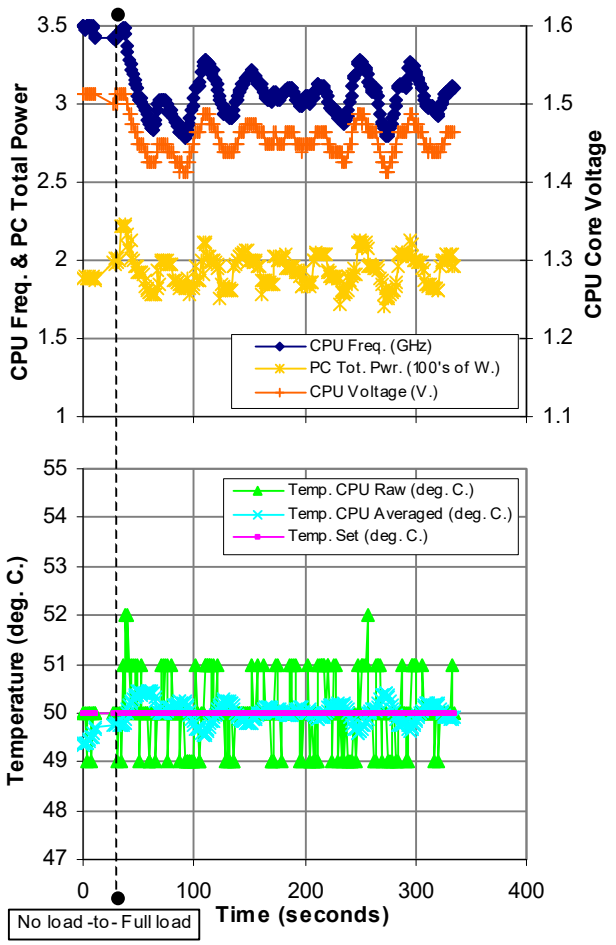

Figure 5. Basic operation and load adaptation test. Vcore IS linked to CPU frequency.

Other applications were run at the same time to test the CPU's capabilities at such extreme conditions; the programs were: Sandra, IE 6 (including video clips), and PowerPoint. TEAPC was still functional at the low frequency and voltage; no OS or other crashes occurred. Thus, disaster tolerance is achieved.

Also, the power savings at the low frequency and voltage settings were substantial. The overall PC power decreased from about 218 W. down to 132 W., under full load, a power savings of about 40%.

The temperature dropped slightly as the frequency and voltage decreased, then rose slightly to a steady 59° C. (The CPU's specified limit is 70° C.) On-chip temperature transients can be fast: 50° C./sec. in a Pentium 4, but this is a result of normal operation and does not require the control system to react as quickly.

With the fan back on, the control system sensed the drop in the CPU's temperature and increased the CPU's frequency. Thus, TEAPC always adapts to the existing conditions, taking advantage of favorable ones as well. Disaster recovery is also achieved.

6. Summary

TEAPC demonstrates the numerous and deep possibilities inherent in modern PC's and microprocessors when advantage is taken of low-level inputs and outputs, and, most especially, when a well-designed feedback-control system is used.

TEAPC maintains a constant temperature while adapting to varying environmental and loading conditions. Thus, with high ambient temperatures TEAPC lowers the frequency (and possibly voltage) to keep the CPU temperature within specifications. Performance is limited, but the system still functions. Conversely, with low ambient temperatures the more favorable conditions can give improved performance. TEAPC also has disaster tolerance, and low-power/high-reliability operation. We feel TEAPC could open the way for much more versatile and cost-saving PCs, in many cases those that already exist.

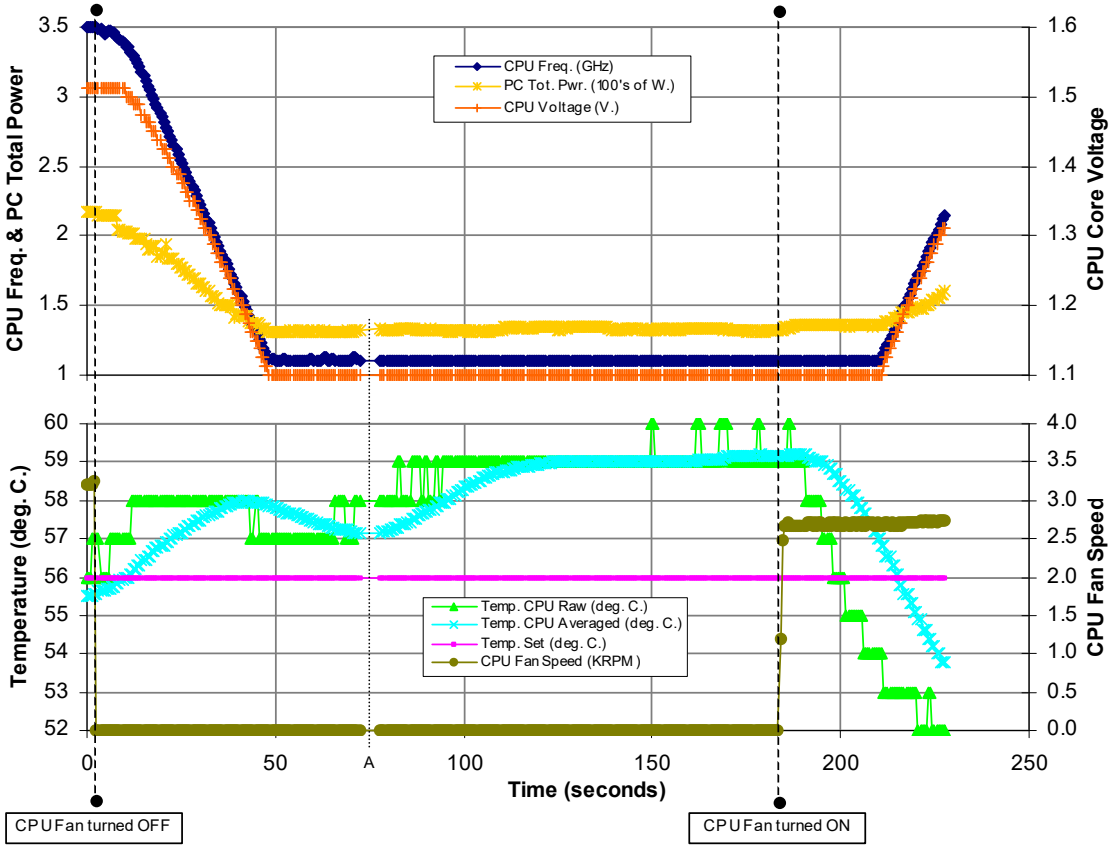

Figure 6. Disaster tolerance and recovery: CPU fan turned off then back on; system under full load. TEAPC remains functional at the low frequency and core voltage, even with the fan off, adapting to take the best advantage of existing conditions. (`teapc` is briefly idle around point 'A.')

References

[1] D. Brooks and M. Martonosi, "Dynamic Thermal Management for High-Performance Microprocessors," in Proceedings of the Seventh International Symposium on High-Performance Computer Architecture (HPCA'01). Nuevo Leone, Mexico: IEEE, January 20-24, 2001, pp. 171-184.

[2] G. F. Franklin, M. L. Workman, and D. Powell, Digital Control of Dynamic Systems, 3rd ed: Prentice-Hall, 1997.

[3] Intel Staff, "Get a Notebook That Enables Extended Battery Life," Intel Corporation, 2003.

[4] T. Kuroda, K.Suzuki, S. Mita, T. Fujita, F.Yamane, F. Sano, A. Chiba, Y. Watanabe, K. Matsuda, T. Maeda, T. Sakurai, and T. Furuyama, "Variable Supply-Voltage Scheme for Low-Power High-Speed CMOS Digital Design," IEEE Journal of Solid-State Circuits, vol. 33, no. 3, pp. 454-462, March 1998.

[5] E. Rohou and M. D. Smith, "Dynamically Managing Processor Temperature and Power," in Proceedings of the 2nd Workshop on Feedback-Directed Optimazation, November 1999.

[6] H. Sanchez, B. Kuttanna, T. Olson, M. Alexander, G. Gerosa, R. Philip, and J. Alvarez, "Thermal Management System for High Performance PowerPCTM Microprocessors," in Proceedings of COMPCON 97. San Jose, CA, USA: IEEE, February 23-26, 1997, pp. 325-330.

[7] A. E. Sjogren and C. J. Myers, "Interfacing Synchronous and Asynchronous Modules Within a High-Speed Pipeline," in Proceedings of the 17th Conference on Advanced Research in VLSI (ARVLSI '97), 1997, pp. 47-61.

[8] K. Skadron, T. Abdelzaher, and M. R. Stan, "Control-Theoretic Techniques and Thermal-RC Modeling for Accurate and Localized Dynamic Thermal Management," in Proc. of the 2002 International Symposium on High-Performance Computer Architecture. Cambridge, MA, USA: IEEE, 2002.

[9] K. Skadron and M. Stan, "First Workshop on Temperature-Aware Computer Systems (TACS-1)," in 2004 International Symposium on Computer Architecture (ISCA) Munich, Germany, June 2004.

[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. San Diego, CA, USA: IEEE and ACM, June 2003.

[11] A. K. Uht, "Going Beyond Worst-Case Specs with TEAtime," Computer, vol. 37, no. 3, pp. 51-56, March 2004.

[12] A. K. Uht, "Uniprocessor Performance Enhancement through Adaptive Clock Frequency Control," IEEE Transactions on Computers, vol. 54, no. 2, pp. 132-140, February 2005.

[13] A. K. Uht and R. J. Vaccaro, "TEAPC: Adaptive Computing and Underclocking in a Real PC," in Proceedings of the First IBM P=ac2 Conference. Yorktown Heights, NY, USA: IBM T.J. Watson Research Center, October 6-8, 2004, pp. 45-54.

[14] R. J. Vaccaro, Digital Control: A State-Space Approach: McGraw-Hill, 1995.

[15] A. Weissel and F. Bellosa, "Dynamic Thermal Management for Distributed Systems," in Proceedings of the First Workshop on Temperature-Aware Computer Systems (TACS), at ISCA 2004. Munich, Germany, June 2004.