

A programmable thermal management interface circuit for PowerPC systems

Herming Chiueh^{a,b,*}, Jeffrey Draper^a, John Choma Jr.^b

^aInformation Sciences Institute, University of Southern California, Marina del Rey, CA 90292, USA

^bDepartment of Electrical Engineering, University of Southern California, Los Angeles, CA 90089, USA

Abstract

A programmable thermal management interface circuit for PowerPC systems has been designed, implemented, and tested for the Integrated Thermal Management (ITEM) System [1]. Instead of worst-case design, the ITEM system approach is to target nominal power dissipation and have the system actively monitor its thermal activity and control cooling mechanisms to ensure operation within specification. Using a suitable combination of hardware and software, the interface design yields intricate control and optimal management with little system overhead and minimum hardware requirements, as well as provides the flexibility to support different management algorithms. This interface circuit was fabricated in the HP 0.5 μm single-poly 3-metal process through MOSIS. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords: Thermal management; Interface circuitry; Temperature monitoring

1. Introduction

Increases in circuit density and clock speed in modern computer systems have brought thermal issues into the spotlight of VLSI design. Local overheating [2,3] in one part of a high-density circuit, such as CPUs and high-speed data routing circuits, can cause a whole system to crash. Besides the use of heat sinks and other heat dissipation mechanisms, early detection of overheating and proper handling of such an event is becoming an essential capability to avoid system failure [4]. The ACPI (Advanced Configuration and Power Interface) specification [5] was developed to provide a standardized approach to configuring the hardware, systems, and software necessary for power and thermal management within personal computer systems.

Temperature sensors and system monitors are a core part of any reliable thermal management system. However, current desktop implementations of the ACPI standard require an external temperature sensor, which suffers a time delay in temperature reading due to the thermal constant between the integrated circuit being monitored and the external sensor. Furthermore, in most ACPI hardware implementations, the embedded system monitor is hardwired, and thus inflexible, and uses a simplified control algorithm which prohibits opti-

mal management; on the other hand, pure software implementations require an active daemon in the operating system, which decreases system performance [5].

In this paper, we present an implementation of a Thermal Management Interface Circuit (TMIC), a subcomponent of a thermal management chip (TMC) to be used in PowerPC systems. Using a suitable combination of hardware and software, such a system can reduce operating system overhead while achieving extra temperature control, as well as provide the flexibility to support different management algorithms. Implementing the ACPI protocol is easy using this design since the TMIC provides programmable threshold-generated interrupts and a memory-mapped interface. This design yields intricate control and optimal management with less system overhead and minimum hardware requirements.

In Section 2, the design specifications and architecture of the TMIC are addressed and justified. In Section 3, implementation flow, tools, simulation and test results are presented. Integration and functionality in our final system are also addressed. Implementations of ACPI and other thermal management protocols using this design are discussed in Section 4.

2. Specification and architecture design

The TMC was developed for use in an embedded multi-computer system with an integrated hierarchical

* Corresponding author. Tel.: +1-310-822-1511; fax: +1-310-823-6714.

E-mail addresses: chiueh@isi.edu (H. Chiueh), chiueh@usc.edu (H. Chiueh), draper@isi.edu (J. Draper), johnc@usc.edu (J. Choma Jr.).

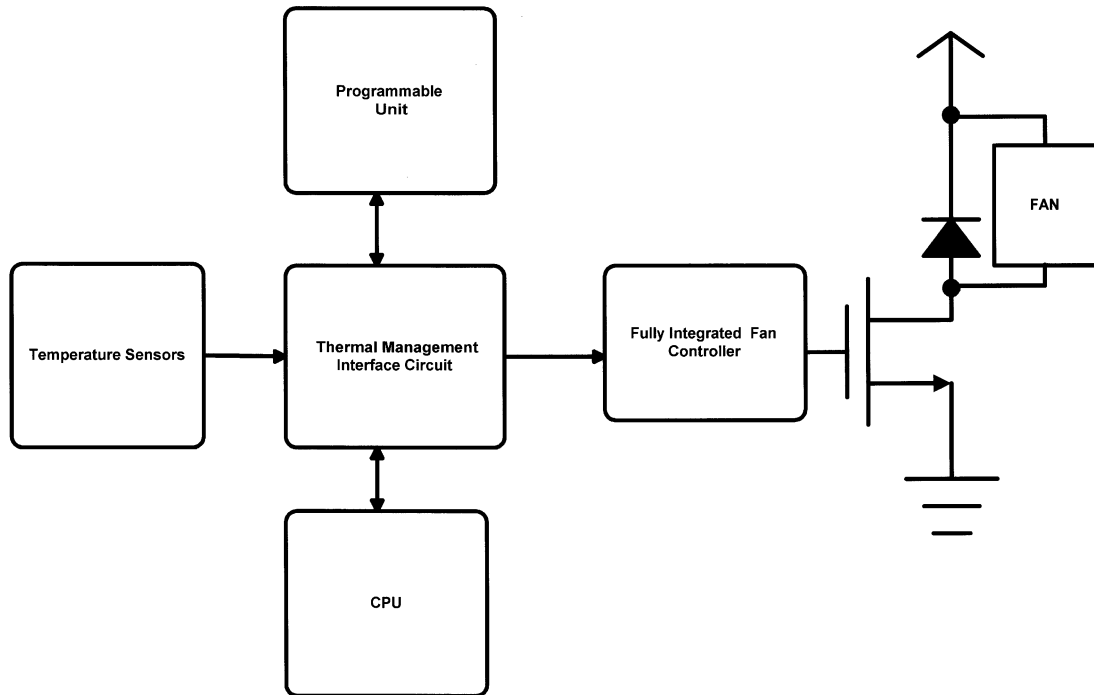


Fig. 1. Block diagram of a thermal management system for advanced computer systems.

thermal management scheme [1]. This system design supports temperature reading and thermal control activity to be performed locally at a node but also provides for global control capability at the system host as well. Each node of this Integrated Thermal Management (ITEM) System contains a PowerPC 604 CPU [6], an Enhanced Router Interface (ERIF) [7], a number of DRAM devices, and a TMC. The TMC device contains an on-chip temperature sensor [8] with an integrated A/D converter [9], and a TMIC [10]. The ERIF device is a custom component that contains a network router, network interface, PPC604 bus controller, and a DRAM controller [7]. Additionally, the ERIF contains 3 embedded ring oscillators to serve as temperature indicators. These temperature sensors as well as the analog temperature sensor in the TMC are based on designs that have been optimized through previous research [3,7–9,12] and are not the focus of this paper.

The TMIC is the configuration and interface portion of the TMC. It contains configuration registers to allow system software to set the sampling rate for reading the temperature and a threshold value for which an interrupt is generated, as well as PowerPC 604 bus interface circuitry to communicate with a node processor. Fig. 1 shows a block diagram of a thermal management system for advanced computer systems. Fig. 2 shows a block diagram of the TMIC. The detailed function of each portion is described below.

- PowerPC 604 interface: supports the bus arbitration policy and four-clock burst read and write mode [6] for Power PC 604 processors. It translates the CPU address and controls signals into corresponding internal control

signals in order to access (read or write) the configuration, sample, and threshold registers and monitor (read) the sampled temperature value.

- Configuration register: 4-bit register that contains 2 temperature sensor selection bits, one interrupt enable bit, and a threshold flag bit (read-only), which indicates the sampled temperature has exceeded the specified threshold temperature. Bit assignments are optimized to reduce the number of clock cycles needed for checking the threshold flag. The sensor selection bits are used to specify which temperature sensor is currently being sampled (the embedded one on the TMC or one of the 3 external ring oscillator temperature sensors in the ERIF). The combination of the interrupt enable bit and threshold flag allows the system programmer to choose to implement polling-based or interrupt-driven (or some combination) temperature monitoring.
- Sampling register: integer value that specifies how many clock cycles elapse between temperature samplings. This value is compared with the value of a continuously incrementing counter. When the two values match, the current temperature sensor value is latched into the temperature register, and the counter is reset to 0. The value of the sampling register may be very different for each sensor or thermal management algorithm requirement.
- Threshold register: integer value that specifies a threshold temperature. When the temperature register value exceeds this threshold value, the threshold flag is set, and an interrupt is generated if the interrupt enable flag is asserted.
- Temperature register: 8-bit register that stores a value

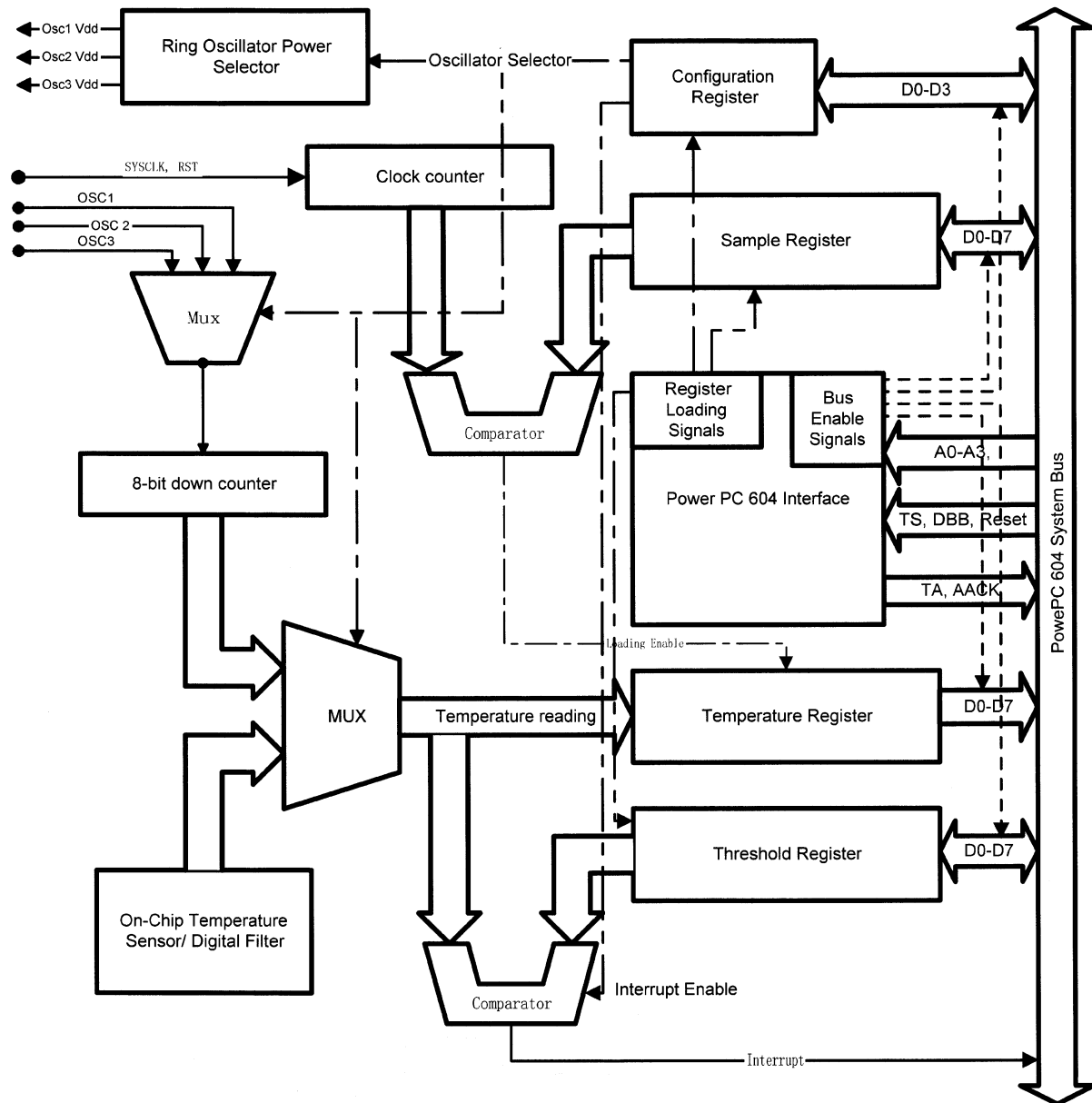


Fig. 2. Block diagram of temperature monitoring interface circuit.

from the currently selected temperature sensor. The temperature register is updated at the periodic rate specified by the sampling register.

- **Interrupt generator:** when the temperature reading exceeds the value of the threshold register, an interrupt is generated if the interrupt enable in the configuration register is asserted. The capability to disable interrupts provides the option for the system design to implement an active or passive thermal control algorithm.

As mentioned above, any one of three ring oscillators on the ERIF chip or the on-chip temperature sensor can be monitored to provide the flexibility of measuring temperature from different locations in the system. For the ring oscillator temperature sensors, the TMIC contains an 8-bit

down counter that is used as a frequency counter, which is calibrated by SPICE simulations and measurements. Depending on the value of the sensor selection bits in the configuration register, either this 8-bit counter value or the 8-bit digital filter output of the on-chip sensor will be stored in the temperature register. The temperature sensor selector also controls the supply power of ring oscillators. By disabling unused temperature sensors, extraneous heat and noise sources may be eliminated.

The TMIC registers occupy two cache lines in the node memory map, one for the temperature register, the other one for the read/write of configuration, sample, and threshold registers. The architecture parameters, such as register bit widths, are based on simulations, measurement, and previous research [2,8,9,12]. Also, careful address and bit

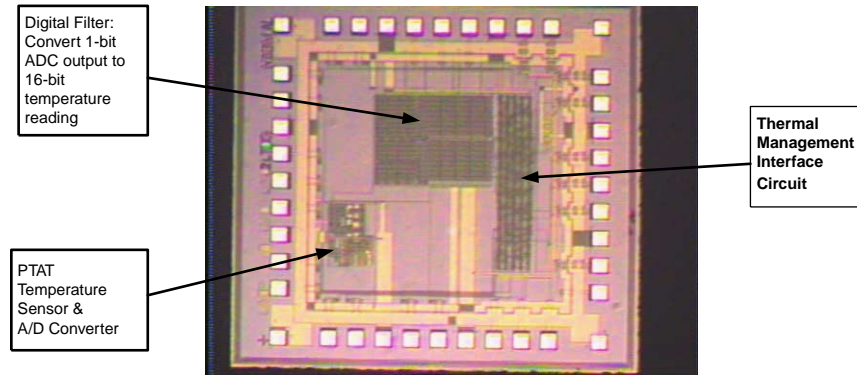


Fig. 3. Microphotograph of TMC.

assignments have minimized and optimized the TMC pin counts as well as the use of system resources by allowing only burst-mode memory accesses.

3. Circuit implementation

This chip was fabricated on an HP 0.5 μm single-poly 3-metal process through MOSIS. The die microphotograph is shown in Fig. 3. The temperature sensor [8], digital filter [12] and TMIC are indicated on the photo. The TMIC occupies 231.3 μm × 1094.4 μm of the area and contains 3293 transistors.

The layout of the TMIC was generated by Powerview schematic capture tools [13] and Lager synthesis tools [14] using standard cells developed for the ERIF chip. These standard cells have been modified previously to fit sub-micron processes. Functionality of the TMIC was verified by Powerview simulations using Lager standard cell VHDL models and Berkeley IRSIM [15] at the transistor

switch level. Both simulations indicate this chip was fully functional at the system clock requirement of 50 MHz.

An initial lot of 5 TMC die were packaged in 40-pin DIPs for low-cost functionality testing. Upon successful results from this test, the remaining TMC die were packaged in 40-pin LCC packages for inclusion in the ITEM system. A final system node board is shown in Fig. 4. The TMC works perfectly at the targeted system speed of 50 MHz.

4. Thermal management system implementation

Different approaches for a thermal management system can be easily implemented with the TMIC device, since the TMIC provides flexible ways for systems to read the temperature, set the threshold value for interrupt generation, and measure temperature values from different sensors. In this section, we illustrate how the TMIC can be used to implement an ACPI-compliant protocol [5] as an example for other thermal management algorithms.

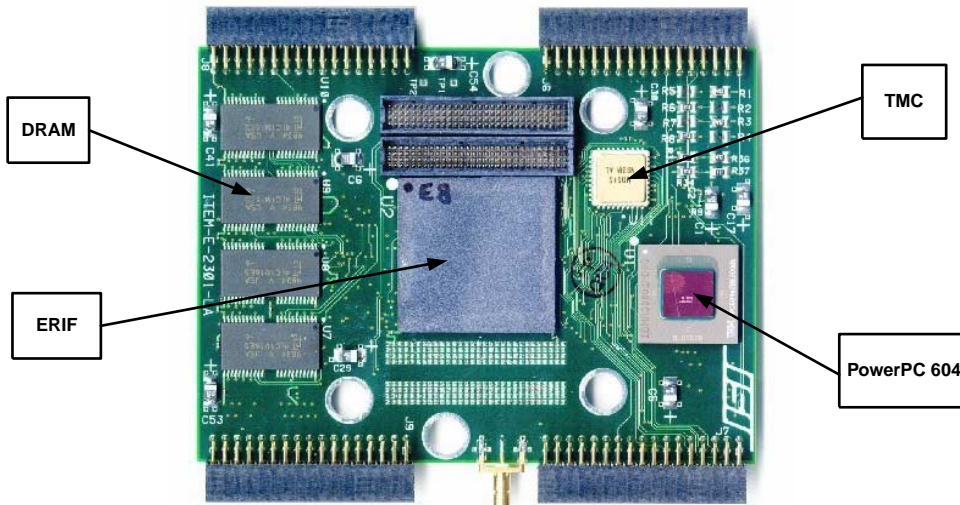


Fig. 4. Item system node board.

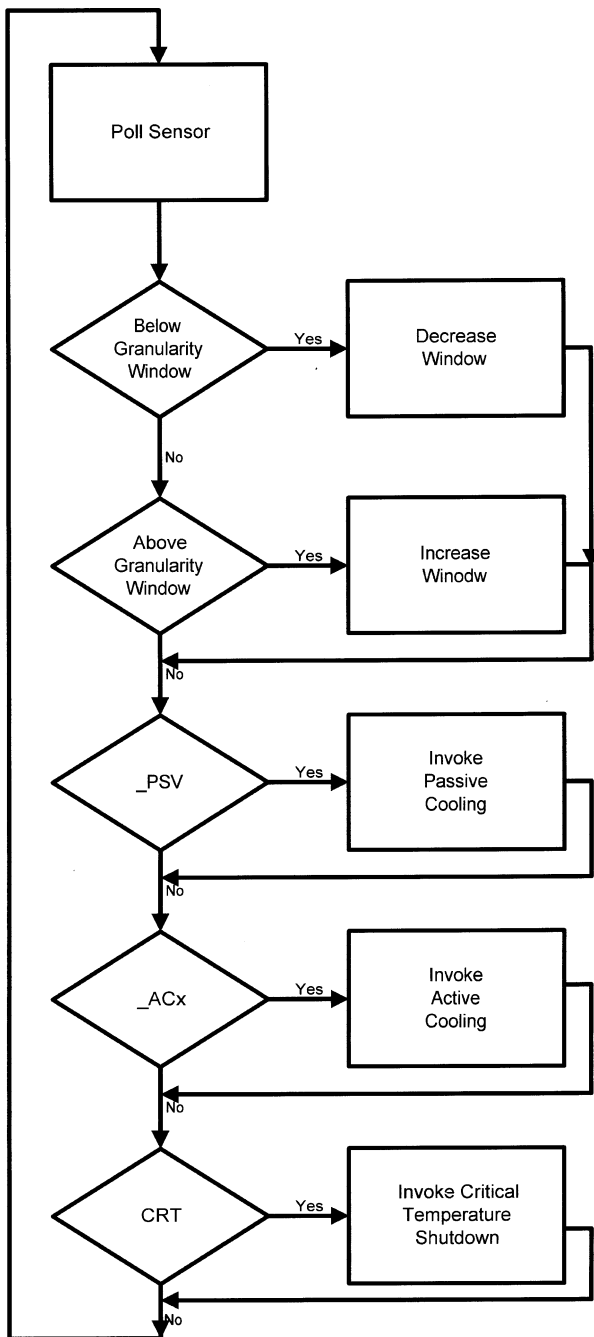


Fig. 5. Typical flow chart of ACPI standard.

A typical flow chart for an ACPI implementation [5] is shown in Fig. 5. The system acquires a temperature and first determines if it is within the current granularity window, repositioning the window as needed. If the sampled temperature has exceeded the Passive Cooling (PSV), Active Cooling (ACX), or Critical Temperature (CRT) thresholds, corresponding defined actions, such as reduce CPU clock, activate fan, and system shutdown, will take place. Most software approaches require a continuous loop, while hardware approaches use interrupts. The advantage of the hardware approach is that processor time is used

for thermal management only when certain situations require it.

With the use of the TMIC, all system actions needed to implement the ACPI protocol are triggered by the actions of acquiring the temperature and generating an interrupt when the sampled temperature exceeds the threshold. These two actions can be used to implement the ACPI protocol as shown in Fig. 6. Here, there is no concept of a granularity window—all temperature values are of importance. The threshold register is initialized to the PSV value in the first step. If an interrupt is generated, it indicates that the PSV has been exceeded, so the CPU performs a predefined action for this threshold and resets the threshold register to the ACX value. This time, when an interrupt is generated, it indicates the ACX value has been exceeded, so the CPU performs a more severe predefined action and now resets the threshold register to the CRT value. Since the TMIC continuously samples the temperature and compares it against a programmable threshold, CPU resources are required only when significant events have occurred. In contrast, software implementations require CPU resources even under default circumstances because they rely on polling to sample temperature values and then must perform computation to determine if a significant event has occurred. A further danger with software implementations is that significant events, such as temperature spikes, may be easily missed if the polling period is not sufficiently small. Compared to other hardware implementations, the TMIC reduces the number of different interrupts and requires the same amount of software cooperation. Furthermore, unlike most hardware implementations, the TMIC provides the ability for the system to actively acquire temperature readings in addition to passively waiting for critical situations.

The TMIC can be used to implement but is not limited to the ACPI protocol. For instance, the temperature threshold can be set to any number of values to represent any number of critical situations. Fuzzy logic control and other algorithms requiring more levels of alerts can be applied. Also with the capability of actively acquiring temperature measures at any time, the CPU can verify a desired temperature response when it executes a cooling action. With this feedback, actions like increase/reduce FAN speed and clock rates can be applied for more complex management algorithms.

5. Conclusion

A TMIC for PowerPC systems was designed and implemented. This paper presents the architecture, design flow and applications for this circuit. The designed architecture yields a balance of hardware and software that is required for the hierarchical thermal management scheme of the ITEM embedded multi-computer system. The TMIC may be used to implement the industry-standard ACPI protocol. However, its elegant, flexible design enables it to implement more complex thermal management algorithms as well. The

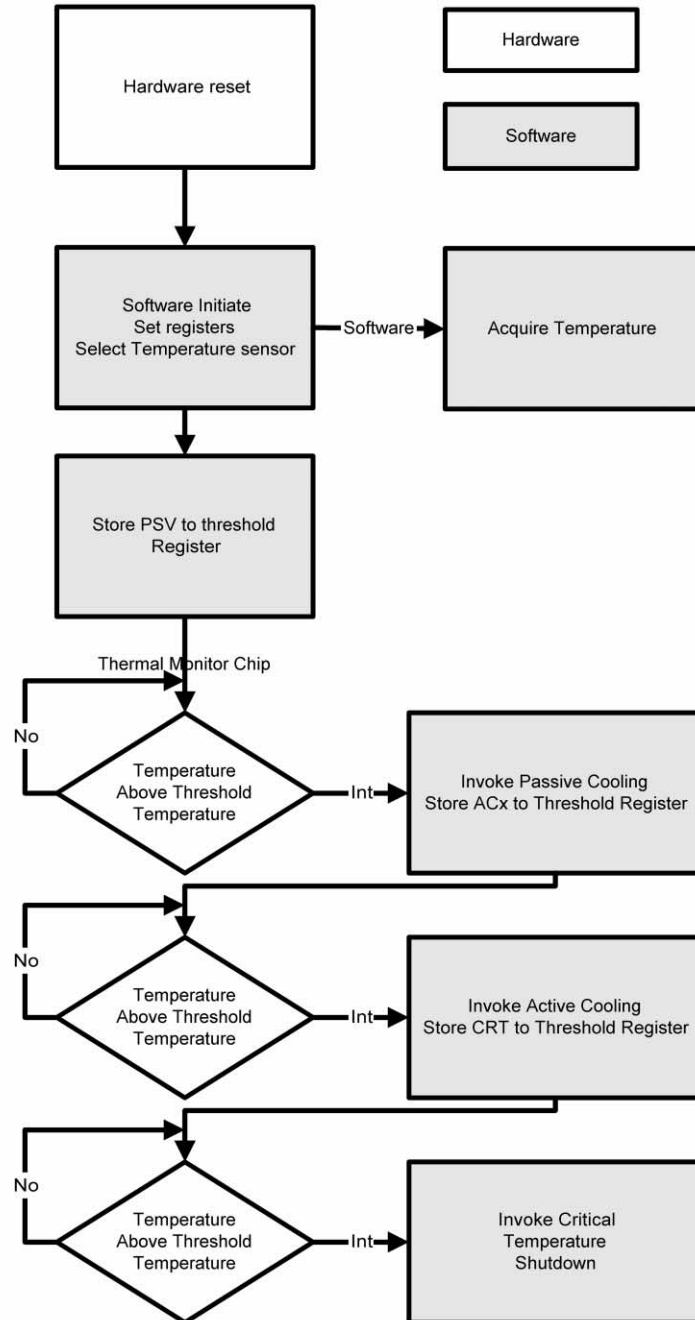


Fig. 6. ACPI implementation by TMC.

proposed architecture combined with the fully integrated fan controller [11] used in this system, make it possible to integrate the whole thermal management system inside the CPU with minimum external components and minimum physical size.

Acknowledgements

The authors would like to acknowledge the support of DARPA (Contract No. DABT63-95-0136).

References

- [1] J. Draper, J. Block, J. Koller, C. Steele, Thermal management in embedded systems using MEMS, Proceedings of the Lecture Notes in Computer Science 1388 (IPPS/SPDP'98 Workshops Proceedings) 1998, pp. 900–901.
- [2] H. Chiueh, J. Draper, L. Luh, and J. Choma Jr., A thermal evaluation of integrated circuits: on-chip offset temperature measurement and modeling, Proceedings of the Second International Workshop on Design of Mixed-Mode Integrated Circuits and Applications, Guanajuato, Mexico, 1998, pp. 109–113.
- [3] H. Chiueh, J. Draper, L. Luh, and J. Choma Jr., A novel model for on-chip heat dissipation, Proceedings of the 1998 IEEE Asia-Pacific

- Conference on Circuit and Systems, Chiangmai, Thailand, 1998, pp. 779–782.
- [4] V. Szekeley, M. Rencz, B. Courtois, Thermal testing methods to increase system reliability, Proceedings of the 13th IEEE SEMI-THERM Symposium, Austin, Texas, 1997, pp. 210–217.
- [5] J. Steele, ACPI thermal sensing and control in the PC, Proceedings of Wescon'98, Anaheim, California, 1998, pp. 169–182.
- [6] Motorola, PowerPC 604 RISC Microprocessor User's Manual, 1994.
- [7] C.S. Steele, J. Draper, J. Koller, C. LaCour, A bus-efficient low-latency network interface for the PDSS multicomputer, Proceedings of the International Symposium on High Performance Distributed Computing, 1997, pp. 213–222.
- [8] L. Luh, J. Choma Jr., J. Draper, H. Chiueh, A high-speed CMOS on-chip temperature sensor, Proceedings of the European Solid-State Circuits Conference (ESSCIRC'99), 1999, pp. 290–293.
- [9] L. Luh, J. Choma Jr., J. Draper, Feed-forward gain compensation for CMOS continuous-time sigma–delta modulators, Proceedings of the IEEE International Conference on Electronics, Circuits and System, 1999.
- [10] H. Chiueh, J. Draper, J. Choma Jr., Implementation of a temperature monitor interface circuit for PowerPC systems, presented at The 43rd Midwest Symposium on Circuits and Systems, 2000.
- [11] H. Chiueh, L. Luh, J. Draper, J. Choma Jr., A Novel fully integrated fan controller for advanced computer system, Proceedings of the Southwest Symposium on Mixed-Signal Design, San Diego, California, 2000, pp. 191–194.
- [12] Luh, J. Choma, Jr., J. Draper, H. Chiueh, A high-speed digital comb filter for sigma–delta analog-to-digital conversion, Proceedings of the IEEE Midwest Symoisum on Circuits and Systems, 1999.
- [13] Viewlogic Inc., "Powerview", <http://www.viewlogic.com>.
- [14] R.W. Brodersen, Anatomy of a Silicon Compiler, 1992.
- [15] A. Salz, M. Horowitz, 'IRSIM: an incremental MOS switch-level simulator, Proceedings of the 26th Annual. ACM/IEEE Design Automation Conference, 1989, pp. 173–178.