Dynamic Thermal Management for Multi-core Microprocessors Considering Transient Thermal Effects *

Zao Liu †,Tailong Xu ‡, Sheldon X.-D. Tan†, Hai Wang*
† University of California, Riverside, CA 92521
‡ Anhui Univeristy, Hefei, China 230601
*University of Electronic Science and Technology of China, Chengdu, China 610054

Abstract— Dynamic thermal management method is a viable way to effectively mitigate the thermal emergences. In this paper, a new thermal management scheme is proposed to reduce the on-chip temperature variance and the occurrence of hot spots by considering more transient thermal effects. The new method performs the task migrations to reduce the temperature variations across the chip. Instead of intuitively assigning the heavy tasks to the low temperature cores to balance the thermal profile based on steady state thermal analysis, the proposed method applies moment matching based transient thermal analysis techniques for fast thermal estimation and prediction to guide the migration process. We show that by considering the dominant temperature moment component, the resulting algorithm can lead to significant reduction of hot spots without full transient thermal simulation. Our experimental results on a 16 core microprocessor demonstrate that the proposed method can reduce the number of the hot spots by 50% compared to the simple lowest temperature based task scheduling method, leading to more uniform on-chip temperature distribution across the microprocessor cores.

I. Introduction

Excessive on-chip temperature has become a top concern for high-performance microprocessor design as more devices are integrated on a chip. Thermal constraints are the major driving force for wide adoption of multi/many core architectures. As the power density continues to increase, the excessively high temperature spots, called *thermal hot spot* would adversely affect the performance of multi-core microprocessors, causing performance degradation, increased cooling costs, reduced reliability and signal integrity issues [1, 2, 3, 4]. Thus, thermal management techniques that are used to tackle the problem of thermal hot spot continue to be identified by the Semiconductor Industries Association Roadmap [5] as one of the five key challenges during the next decade for achieving the projected performance goals of the industry.

The goal of thermal management is to keep the multi-core microprocessor system working below a safe temperature threshold while maintaining the efficient task execution performance. The most popular methods for runtime thermal management are involved with a dynamic power reduction technique, called *Dynamic Voltage and Frequency Scaling* (DVFS) [1, 6, 7]. These types of method could quadratically reduce the dynamic power through reducing the supply voltage and operating frequency, which translate to reduced data rate. Thus, the DVFS based thermal management method achieves temperature reduction at the expense of reduced performance.

In order to maintain the same performance of the processor system, task scheduling technique is served as an alternative to control the temperature of the cores [8, 9, 3, 10]. The general idea of these methods are migrating the heavy loaded task away from an overheated core to a cooler core. However, the traditional intuitive method like [8, 10], that typically migrating the heaviest task (with largest powers) to the coolest cores, may lead to sub optimal choice on practical platform. The reason is that such decision will make more sense for steady-state thermal effects considering only thermal resistances. However, with transient effects become more significant across a chip, it is more desirable to consider more dynamic thermal effects when we optimize the task migrations to reduce temperature hotspots. However, it is not an easy task to consider the transient effects as full transient numerable thermal analysis for a task period may be required. The simulation will need to be inside the loop of the optimization engine, which can be prohibitively expensive. We notice some early effort to consider this effect has been proposed [3], where the temperature from the neighborhood cores were considered and the temperature incremental factor was used to make task migration decision [3]. However, such neighborhood looking approach, can be viewed as an ad-hoc approach to looking at the transient thermal effects on predicted powers, and this will become more clear in light of the method proposed in this paper.

In this work, we are aimed at developing a task scheduling/migration based thermal management scheme that could effectively reduce the runtime thermal hot spots of multi-core microprocessors.

The new method performs the task migrations to reduce the temperature variations across a chip. Instead of intuitively assigning the heavy tasks to the low temperature cores to balance the thermal profile based on steady state thermal analysis, the proposed method applies moment matching based transient thermal analysis techniques for fast thermal estimation and prediction to guide the migration. By considering the dominant temperature moment component, the resulting algorithm can lead to significant reduction of hot spots.

Comparing with the existing works, the major contributions of the proposed thermal management scheme is summarized as the following: (1) Derivation of a dynamic temperature indicator based on the dominant component of responses temperatures in frequency domain to avoid expensive full transient thermal simulation; (2) Development of a new task scheduling strategy based on the new dynamic temperature indicator. The proposed method essentially considers both the current local temperature and transient effects of past executions of tasks

*This work is supported in part by NSF grant under No. CCF-0902885, in part by Semiconductor Research Corporation(SRC) Grant under No. 2009-TJ-1991.

978-1-4673-3030-5/13/$31.00 ©2013 IEEE

of each core to determine the dominant dynamic temperature response for better task scheduling. Our experimental results on a 16-core microprocessor demonstrate that the proposed method can reduce the number of the hot spots by 50% compared to the simple lowest temperature based task scheduling method, leading to more uniform on-chip temperature distribution across the microprocessor cores.

The rest of the paper is organized as follows. In section II, we present a general outline of the thermal management problem and the motivation to solve it. In section III, we present the proposed method with the algorithm flow. In section IV, the proposed method is evaluated and its significance is discussed. Section V concludes this paper.

II. Task migration based thermal management problem

Thermal management is a broad topic, which consists of many thermal control techniques. In this paper, we focus on the task migrations to reduce the hot spots. Specifically, we assume a multi-core microprocessor consists of N tasks of different load, denoted as $\mathbb{P} = [P_1, P_2, \cdots, P_N]$. M processor cores are involved in executing the tasks. Since the load of the tasks are different, the power intensity varies across all the cores, resulting in non-uniformly distributed temperature profile, which could be possiblely represented in Fig. 1. If some of the cores are always assigned heavy load, it could result in continuous temperature increase up to an alarming level, which would probably lead to performance degradation and reliability issues, even failure of the chip. By properly schedule the tasks among different cores, many hot spots can be avoided without degrading the performance. For instance, we could migrate the heavy task away from the high temperature cores to prevent the emergency of excessively high temperature, and assign it to another core that is less likely to have thermal emergency. In this way, we balanced the workload of all the cores according to their thermal capability, leading to more uniformly distributed temperature profile, and less excessively high temperature cores.

The task migration problem could be formulated into the following description: Given N tasks for M processor cores, find a task scheduling method to minimize the temperature variance, and reduce the number of on-chip hot spots.

One critical part for solving this problem is to identify a suitable core that could take the heavy load task without generating thermal emergency. Traditionally, [8] applies 'heat-and-run' scheme that simply migrates the heaviest load to the coolest core. However, this intuitive choice of assigning heavy tasks to the core with lowest temperature does not work very efficiently to remove the on-chip hot spot because it does not consider the thermal influence from its neighboring cores [3] and the thermal impacts from the previous task executions as will be shown in this paper. In this paper, we propose a new task migration method based on better estimation of the thermal responses of each core to guide the task migration process.

III. The proposed new task migration method

For task migration, ideally, we would like to perform full-blown transient analysis for a given task period using the predicted or estimated power traces and thermal models to fully evaluate the best task assignments at current time and initial temperature conditions. However, such analysis can be prohibitively expensive as numerical integration based simulation is superlinear for the size of the thermal circuits and analysis may be performed many times as the internal loop of the scheduling engine. To mitigate this problem, fast analysis techniques or thermal estimation techniques are required.

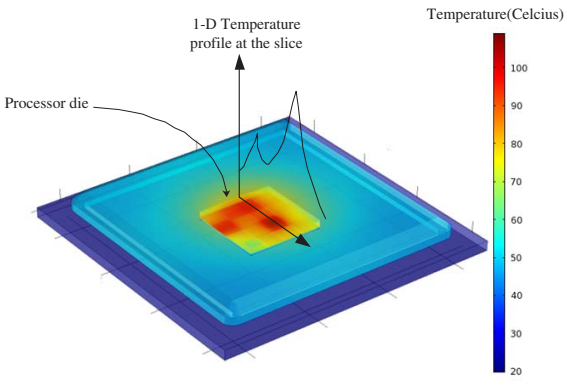

Fig. 1. Temperature profile of a multi-core microprocessor package

A. 0th-moment temperature based indicator

Moment matching based analysis has been used for fast estimation of interconnect delays in the past [11]. In this work, we apply moment matching based analysis for fast transient thermal estimation. Instead of performing the full moment matching based analysis, we would like to derive a temperature indicator to guide the task migration. This scheme is motivated by the observation that the 0th moment component of the power traces of the microprocessor is dominant compared to high frequency components. Fig. 2 shows the Fourier transformation of the power trace from a benchmark, which clearly indicates the dominance of the 0th moment component. As a result, the corresponding 0th moment of the transient temperature responses can be used as a good indicator of the temperature for each core.

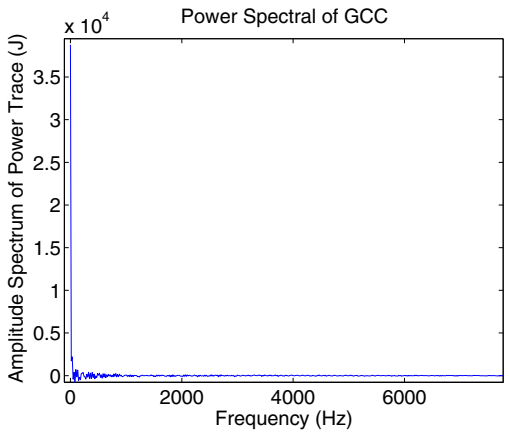

Fig. 2. Discrete Fourier transformation of the power trace for running GCC benchmark

Mathematically, for equivalent thermal circuit using G as the thermal conductance matrix and C as the thermal capacitance matrix, we can apply Modified Nodal Analysis (MNA)

to formulate the thermal circuit

$$GT(t) + C\dot{T}(t) = u(t) \tag{1}$$

where $T(t)$ is the time-domain temperature response, and $u(t)$ is the power trace of the processor.

Assume $U(s)$ as the Laplace transformation of the power trace $u(t)$ of that processor, and $T(s)$ as the corresponding temperature response, in frequency domain, we could have

$$GT(s) + sCT(s) - CT(t_0) = U(s) \tag{2}$$

where $T(t_0)$ is the initial temperature vector of all the nodes at time zero, t_0, or alternatively, the temperature vector at the end of the previous simulation cycle.

To find out the contribution of the transient temperature in terms of frequency moments, we expand $T(s)$ and $U(s)$ using Taylor's series at $s = 0$ to have

$$\begin{aligned} G(T_0 + T_1 s + T_2 s + \cdots) + sC(T_0 + T_1 s + T_2 s + \cdots) \\ -CT(t_0) = U_0 + U_1 s + U_2 s + \cdots \end{aligned} \tag{3}$$

Thus, each temperature moment could be calculated in a recursive way as follows:

$$\begin{aligned} T_0 &= G^{-1}(U_0 + CT(t_0)) \\ T_1 &= G^{-1}(U_1 - CT_0) \\ T_2 &= G^{-1}(U_2 - CT_1) \\ &\vdots \\ T_m &= G^{-1}(U_m - CT_{m-1}) \end{aligned} \tag{4}$$

Since the 0th moment of the power trace U_0 is dominant, the 0th moment component of the transient temperature response T_0 is also dominant. Thus, it makes a good sense to use the 0th moment component of the response temperature (called *0th moment temperature*) as an on-chip temperature indicator. In this way, we could conveniently use (5) to calculate the dominant 0th temperature T_0 given the 0th moment component of power trace U_0 for the future task and the initial temperature $T(t_0)$ at the current time.

$$T_0 = G^{-1}U_0 + G^{-1}CT(t_0) \tag{5}$$

The T_0 is not traditional 0th moment response from the resistor only thermal systems. It actually contains transient effects of the thermal system, which will be explored by the this work. As we can see, the 0th moment temperature consists of two component, $G^{-1}U_0$, actually is the steady state temperature responses considering resistances only of the thermal circuits. The second term, $G^{-1}CT(t_0)$, relates to from the initial temperature condition $T(t_0)$, as well as the thermal capacitance and conductance of the whole system.

B. The proposed task scheduling method

According to (5), we need to consider both terms to determine the dominant temperature responses of the cores in frequency domain. The second term is actually not simple initial temperatures, $x(t_0)$. As a result, we define a new term called *effective initial temperature* in frequency domain as

$$T_{eff} = G^{-1}CT(t_0) \tag{6}$$

T_{eff} has temperature unit in frequency domain. The term $CT(t_0)$ can be viewed as the energy source stored from the previous task executions. Typically the thermal systems obtained from the finite difference method can be remodeled as RC networks where each node has a capacitor connect to ground [12]. The resulting matrix C is typically a diagonal matrix. The thermal capacitance, or heat capacity, indicates the ability of the thermal system to store the thermal energy. Larger C will lead to larger energy and thus higher temperature for the next task cycle or period given the same initial temperature $T(t_0)$. For multi-core systems, core with less thermal capacitance will be favored in this case (given the same initial temperature for all the core). For a two-dimensional chip, those cores typically are located in the corner of the chip. Intuitively, those core will have better cooling capability as they are close to the thermal boundaries of the chip. As a result, they will lead to less temperature for the same initial temperature than the cores in the middle. Notice the G^{-1} is a dense or full matrix. This means the final temperature of specific core (or element in the vector T_{eff}) will also depend on situations of their neighboring cores as well. Given T_{eff}, we can rewrite (5) as

$$T = G^{-1}U_0 + T_{eff} \tag{7}$$

Physically, the first term $G^{-1}U_0$ actually represents the temperature responses due to the (predicted or estimated) power of current task cycle. TheT_{eff} represents the transient effects due to past task executions.

With these definitions, instead of assigning heavy load task to the low temperature core as the traditional intuitive method suggests, we look at T_{eff} instead. Specifically, we first rank T_{eff} first, and then assign the heavy load (with largest U_0) task to the core that has the lowest T_{eff} value. The general scheme is described in Fig. 3, in which the heaviest load task is assigned to the core with lowest T_{eff}, and the second heaviest task is assigned to the core with the second lowest T_{eff}, and so on.

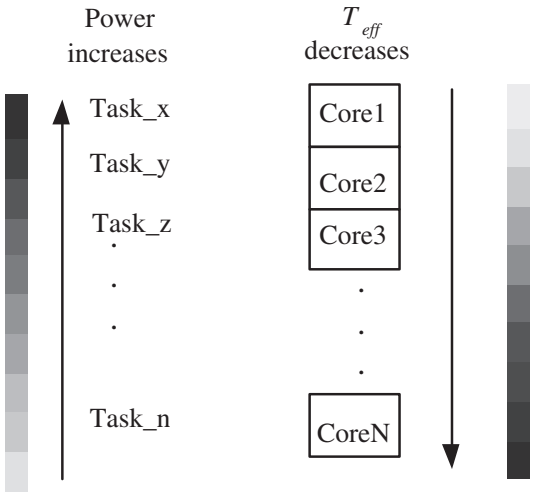

Fig. 3. Proposed new task assignment scheme based on T_{eff}

We remark that ideally we should assign the tasks by ranking the elements in the vector $G^{-1}U_0$ instead of U_0 as the each core's neighbors also contribute to its temperature. But we observe that for multicore system, the temperature of each core is still dominated by its own power. As a result, we found that using U_0 will lead to similar effects of $G^{-1}U_0$. If necessary, the task assignment process in Fig. 3 can be carried out several times until no improvements are founds in terms of computed 0th moments of temperature. Practically, we found that one pass is sufficient enough for most of benchmark cases.

To sum up, by defining T_{eff} and using task scheduling scheme

in Fig. 3, the heat conduction effect due to the processor package design is considered in addition to the local temperature of the core in the thermal management, which allows more effective task scheduling for more balanced temperature profile.

C. The proposed thermal management scheme

Having discussed the concept of 0th moment temperature and task scheduling strategy, it is ready to present the proposed thermal management scheme in this section.

The task scheduling framework is shown in Fig. 4, where the multi-core microprocessor is executing different set of tasks in different execution cycles. Within each execution cycle, a given set of tasks are assigned, and different cores are running different tasks. Similar to [10], we assume that each task occupies an equal slice of execution time. In our approach, we assume that we know the power traces for each task. For practical applications, the power traces or workloads can be obtained from OS or are predicted from the history of the power traces using some estimation methods like time series prediction methods [13]. At the end of each execution cycle, the thermal management scheme uses the final temperature of the processor of the finished cycle to calculate the effective initial temperature T_{eff} of the next execution cycle through (6), then it uses the task scheduling scheme in Fig. 3 to assign the tasks to different cores according to the calculated effective initial temperature T_{eff}. In this way, the tasks are scheduled at the beginning of each new execution cycle to reduce the temperature of hot cores. The algorithm flow of the proposed thermal management scheme is summarized in Fig. 5.

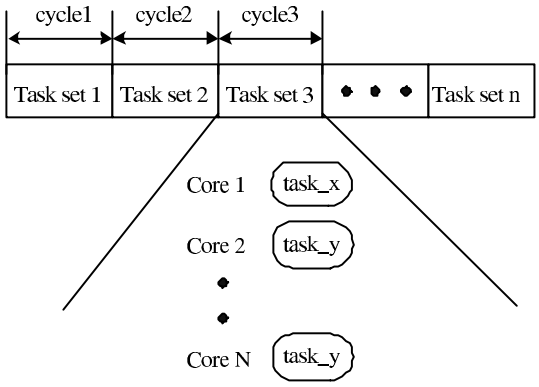

Fig. 4. The thermal scheduling framework used

IV. Experimental results

A. Experiment setups

The thermal management scheme is implemented by Matlab 7.0, and Hotsplot is integrated as a thermal simulator [14, 1]. A 16-core system with 4×4 grids is chosen to evaluate the proposed method. The size of each core is 4 mm $\times$ 4 mm, and thickness of the chip is 0.15 mm. More detailed thermal properties and package structure information is summarized in Table I, where k and c represent the thermal conductivity and specific heat respectively. The heat convection coefficient of the heat sink is set to be 140.4 J/K to model the cooling effect. The thermal sampling interval of Hotspot is set to be 30 μs to preserve simulation accuracy.

Algorithm: Thermal Management

1. Obtain or predict the power traces for different benchmarks.
2. Obtain the frequency domain power spectral for all the power traces.
3. If it is the first execution cycle, use the known initial temperature. Otherwise, use the final temperature of the processor at the end of the previous task execution cycle.
4. Calculate the frequency domain effective initial temperature T_{eff}.
5. Perform task scheduling where task with largest power is assigned with core with lowest T_{eff} and so on as shown in Fig. 3.

Fig. 5. The flow of the proposed thermal management method

TABLE I
Processor package structure and thermal properties

Components	Chip	Heat Spreader	Heat Sink
Thickness(mm)	0.15	1.00	6.90
k ($W/(mK)$)	100.0	400.0	400.0
c ($J/(m^3K)$)	1.75×10^6	3.55×10^6	3.55×10^6

The proposed task scheduling method is validated with dynamic workload from the SPEC 2000 benchmarks shown in Table II. The average power could be used as an indicator of the load of each benchmarks, and the detailed transient power traces could be obtained by using Wattch power analysis tool [15], which is used as the input of the thermal simulator.

We set 80 ^oC as the temperature threshold, above which the hot spot is identified during the transient simulation. In the simulation, for each execution cycle, initially, we randomly assign the SPEC 2000 benchmark tasks to the 16 cores. The thermal management scheme checks the temperature indicator at the end of each execution cycle (2048 time steps) to make task scheduling decision. The new task scheduling decision made by thermal management overwrites the initial random choices to optimize the distribution of the temperature of all the cores. The following metrics is used to evaluate the performance of the proposed thermal management method.
(1) The number of hot spots encountered during the transient simulation given the same total work load: we expect the proposed method could effectively reduce the number of hot spots count in simulation.
(2) On-chip temperature variances: we expect the proposed method makes the distribution of the on-chip temperature more uniform, so that the temperature of the hot cores is effectively reduced.

B. Task execution with the proposed thermal management scheme

In this subsection, the simulation results are discussed and evaluated. In comparison, we also simulate the thermal response of the same chip with two other schemes:
(1) Random scheduling: no thermal management method is applied; at the beginning of each execution cycle as shown in

TABLE II
SPEC 2000 BENCHMARKS USED FOR THE VALIDATION

Benchmarks	BZIP	GZIP	MCF	GCC	SWIM	MGRID	GALGEL
Task IDs	1	2	3	4	5	6	7
Avg. Power(W)	24.62	27.56	35.12	37.87	42.12	44.62	46.33

Fig. 4, a new set of the tasks are randomly assigned to different cores.
(2) Simple lowest temperature scheduling: at the beginning of each execution cycle, this simple method checks the temperature of the cores and assigns the heavy task to the low temperature cores to reduce the hot spots.

Fig. 8 show the temperature variances for the proposed method, the simple method and random scheduling method for the given time. From this figure, we could clearly observe that: (a) The thermal management schemes (both the simple one and the proposed one) could reduce the variance of the on-chip temperature. (b) The proposed method could more effectively reduce the on-chip temperature variance comparing with the simple approach that simply assigns heavy task to low temperature cores with out considering heat removal effect of the package thermal conductance matrix G.

Fig. 6 shows the architecture of 16-core microprocessor, in which 'C1', 'C2', ... ,'C16' are the index of its cores. Fig. 7 shows task assignments into different cores. The upper matrix represent the task assignment using random method. The column or x axis is for different cores. The row or y axis is the time instance. The boxed column is the position for the corner cores. Larger task id represents heavier task (with larger power consumption as shown in Table II). For instance, the 5 in the first column and first row means the task 5 is assigned to the first core at time instance 1. From this figure, we can see that the heavier tasks were indeed assigned to the corners, which has less thermal capacitance and better cooling capability.

C13	C14	C15	C16
C9	C10	C11	C12
C5	C6	C7	C8
C1	C2	C3	C4

Fig. 6. Architecture of 16-core microprocessor die

Task before scheduling (random)

C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
5	2	6	2	3	4	2	3	2	7	3	2	5	2	6	6
5	1	3	5	1	5	1	6	1	4	2	4	5	2	3	7
5	5	5	5	7	4	1	5	5	3	5	5	4	6	6	3
7	4	3	4	5	6	1	6	1	5	2	3	1	4	6	5
2	4	6	7	4	4	3	7	4	6	1	6	3	7	3	4
5	5	6	2	5	7	4	7	4	4	6	3	3	3	2	6

Task after scheduling (By proposed method)

C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
7	2	5	6	3	2	2	5	2	2	2	4	6	3	3	6
6	4	3	5	5	1	1	2	5	1	1	2	7	4	3	5
6	5	5	6	4	1	3	5	5	3	4	5	5	5	5	7
6	5	5	7	4	2	1	5	4	1	1	4	6	3	3	6
7	4	3	6	4	2	1	4	4	3	3	4	7	6	6	7
7	5	6	7	5	3	3	5	4	2	2	4	6	3	4	6

Fig. 7. Task assignments on different cores before (random) and after the proposed scheduling method. Boxed columns are tasks executed on corner cores. The larger task id represents heavier task.

reduce the temperature of the core by assigning a proper task to it based on the value of the calculated effective initial temperature X_{eff}.

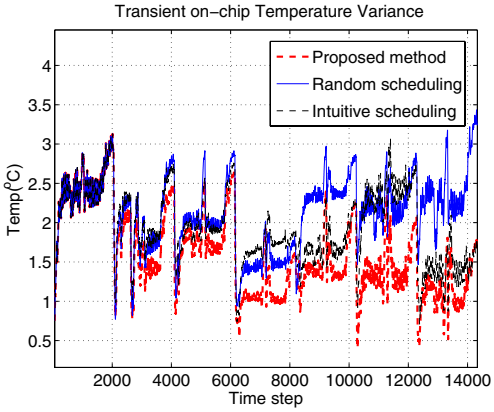

Fig. 8. Transient temperature variance among the three scheduling method

Fig. 9 compares the temperature distributions across all the 16 cores of the processor at a given time step. It clearly confirms that the proposed thermal management scheme leads to more uniform temperature distribution across all the 16 cores, and the temperature of the hottest core is effectively reduced, comparing with the case when employing the simple scheduling method.

In Fig. 10, we compare the transient temperature responses of one microprocessor core under different task scheduling methods. We could clearly observe that the simple scheduling method is sub-optimal and sometimes even increase temperature of the core to a higher degree because its thermal management decision is unaware of the heat removal effect of that core. The proposed method, on the other hand, could always effectively

Table. III summarizes the number of encountered hot spots (Temperature above 80 ^oC) during the transient simulation. The proposed thermal management effectively reduced the temperature of the hot cores, so that the occurrence of the hot spot is significantly reduced (50 % less occurrence comparing with the simple scheduling method). The reduced number of hotspot count clearly shows that the processor cores are less frequently operating with the temperature above 80 ^oC during transient simulation time, which is the desired purpose of the on-chip thermal management scheme.

Hence, by comparing with the simple scheduling method used before, our proposed method indeed lead to more significant on-chip hot spot removal and temperature reduction

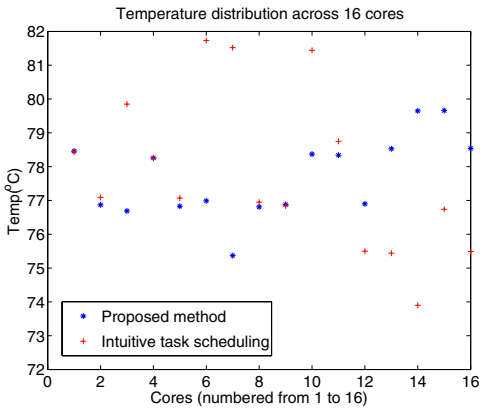

Fig. 9. Temperature distribution across the cores (time step 11600)

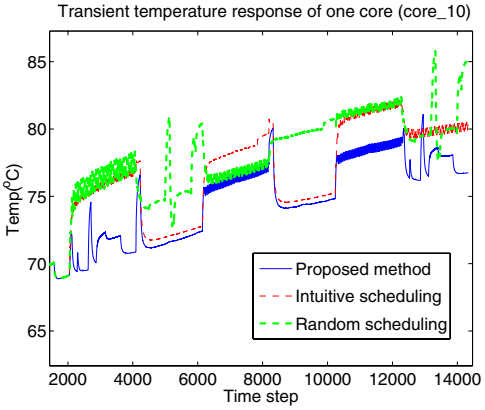

Fig. 10. Comparison of transient temperature responses using different task scheduling schemes

for hot cores. More uniform temperature profile of the working multi-core microprocessor could be achieved by using our thermal management scheme.

TABLE III
STATISTIC OF HOT SPOT OCCURRENCE DURING TRANSIENT SIMULATION

Scheduling Methods	Proposed	simple	Random
Hot spot count	9305	18705	21908

V. CONCLUSION

In this paper, we have proposed a new dynamic thermal management to reduce the on-chip temperature variance and the occurrence of hot spots by considering more transient thermal effects. The new method performs the task migrations to reduce the temperature variations across the chip. The proposed method applies moment matching based transient thermal analysis techniques for fast thermal estimation and prediction to guide the migration. We showed that by considering the dominant temperature moment component, the resulting algorithm can lead to significant reduction of hot spots without full transient thermal simulation. Our experimental results on a 16-core microprocessor demonstrated that the proposed method could more effectively reduce the number of the hot spots during the runtime comparing with the traditional intuitive approach, leading to more uniform on-chip temperature distribution across the microprocessor cores.

VI. REFERENCES

[1] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-aware microarchitecture," in *Proc. Int. Symp. on Computer Architecture (ISCA)*, pp. 2–13, 2003.

[2] M. Hertl, D. Weidmann, and A. Ngai, "An advanced Reliability Improvement and Failure Analysis Approach to Thermal Stress Issues in IC Packages," in *35th International Symposium for Testing and Failure Analysis, ISTFA*, pp. 28–32, 2009.

[3] G. Liu, M. Fan, and G. Quan, "Neigbor-Aware Dynamic Thermal Management for Multi-core Platform," in *Proc. European Design and Test Conf. (DATE)*, pp. 187–192, 2012.

[4] A. Chakraborty, K. Duraisami, A. Sathanur, P. Sithambaram, A. Macii, E. Macii, and M. Poncino, "Implementation of a thermal management unit for canceling temperature-dependent clock skew variations," *Integration, the VLSI Journal*, vol. 41, no. 1, pp. 2–8, 2008.

[5] "International technology roadmap for semiconductors (ITRS), 2011," 2011. `http://public.itrs.net`.

[6] R. Mukherjee and O. S. Memik, "Physical aware frequency selection for dynamic thermal management in multi-core systems," in *Proc. Int. Conf. on Computer Aided Design (ICCAD)*, pp. 547–552, 2006.

[7] S. Herbert and D. Marculescu, "Analysis of dynaic voltage/frequency scaling in chip-multiprocessors," in *Proc. Int. Symp. on Low Power Electronics and Design (ISLPED)*, pp. 38–43, 2007.

[8] M. Powell, M. Gomaa, and T. N. Vijaykumar, "Heat-and-run: leveraging smt and cmp to manage power density through the operating systems," in *ACM SIGPLAN NOTICES*, vol. 39, pp. 260–270, 2004.

[9] I. Yeo, C. C. Liu, and E. J. Kim, "Predictive dynamic thermal management for multicore systems," in *Proc. Design Automation Conf. (DAC)*, DAC '08, (New York, NY, USA), pp. 734–739, ACM, 2008.

[10] Y. Ge, P. Malani, and Q. Qiu, "Distributed task migration for thermal management in many-core systems," in *Proc. Design Automation Conf. (DAC)*, pp. 579–584, 2010.

[11] L. T. Pillage, R. A. Rohrer, and C. Visweswariah, *Electronic Circuit and System Simulation Methods*. New York: McGraw-Hill, 1994.

[12] Y.-K. Cheng, C.-H. Tsai, C.-C. Teng, and S.-M. Kang, *Electrothermal Analysis of VLSI Systems*. Kluwer Academic Publishers, 2000.

[13] G. Box, G. M. Jenkins, and G. C. Reinsel, *Time Series Analysis: Forecasting and Control*. Prentice-Hall, 3rd ed., 1994.

[14] W. Huang, M. Stan, K. Skadron, K. Sankaranarayanan, S. Ghosh, and S. Velusamy, "Compact thermal modeling for temperature-aware design," in *Proc. Design Automation Conf. (DAC)*, pp. 878–883, 2004.

[15] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural-level power analysis and optimizations," in *Proc. Int. Symp. on Computer Architecture (ISCA)*, pp. 83–94, 2000.