Exploring the Effects of On-Chip Thermal Variation on High-Performance Multicore Architectures

CHEN-YONG CHER and EREN KURSUN, IBM Thomas J. Watson Research Center

Inherent temperature variation among cores in a multicore architecture can be caused by a number of factors including process variation, cooling and packaging imperfections, and even placement of the chip in the module. Current dynamic thermal management techniques assume identical heating profiles for homogeneous multicore architectures. Our experimental results indicate that inherent thermal variation is very common in existing multicores. While most multicore chips accommodate multiple thermal sensors, the dynamic power/thermal management schemes are oblivious of the inherent heating tendencies. Hence, in the case of variation, the chip faces repetitive hotspots running on such cores. In this article, we propose a technique that leverages the on-chip sensor infrastructure as well as the capabilities of power/thermal management to effectively reduce the heating and minimize local hotspots. This technique can be used in existing multicore chips as long as the thermal sensor data can be made transparent to the power/thermal management at the software layer. According to our experimental analysis on test-chips, 5°C peak temperature reduction can be achieved with no performance degradation, hence the inherent energy efficiency of the chip can be improved without any performance or cost penalty.

Categories and Subject Descriptors: C.4 [**Performance of systems**]: *Reliability, availability, and service ability*; D.4.1 [**Operating System**]: Process Management—*Scheduling*; D.2.8 [**Software Engineering**]: Metrics (D.4.8)—*Process Metrics*

General Terms: Design

Additional Key Words and Phrases: Temperature management, multicore architectures, thermal variation, temperature-aware scheduling, thermal imaging, thermal efficiency

ACM Reference Format:
Cher, C.-Y. and Kursun, E. 2011. Exploring the effects of on-chip thermal variation on high-performance multicore architectures. ACM Trans. Architec. Code Optim. 8, 1, Article 2 (April 2011), 22 pages.
DOI = 10.1145/1952998.1953000 http://doi.acm.org/10.1145/1952998.1953000

1. INTRODUCTION

Multicore architectures have become defacto in recent years for various market segments ranging from embedded processors to high end servers. While benefits of homogeneity in multicores have been well studied for widespread acceptance in the market; in reality, the individual cores in the so-called homogeneous multicore architectures exhibit significant differences. The causes and amplitudes of on-chip variation are very diverse—including, but not limited to, lithography-induced variations, imperfections in the chemical/mechanical processing stages and packaging/cooling imperfections.

Among these factors, variation in parameters such as V_{th}, L_{eff}, t_{ox}, and pattern density has been increasing steadily with each generation. As a result, individual cores, functional units and even macros on the same chip differ in terms of performance,

Author's address: C.-Y. Cher and E. Kursun, email: {chenyong, ekursun}@us.ibm.com.

DOI 10.1145/1952998.1953000 http://doi.acm.org/10.1145/1952998.1953000

peak clock frequency, power, and thermal profiles (even with perfect packaging/cooling assumptions). While functional implications of such on-chip variation have been subject to many research studies, research on power/thermal effects is limited. This is mostly due to the common belief that such power/thermal variation would be minimal or nonexistent at the core or functional unit level.

To the contrary, thermal imaging results indicate that significant thermal variation is typical at core and unit levels in the current processors across different vendor's chips and technologies. However, the existing thermal management solutions are mostly oblivious to the underlying variations on the chip. Furthermore, unlike the heating on ideal chips, such inherent thermal variation is more difficult to plan for and evaluate during the design and planning stages.

Variation interferes with the effectiveness of thermal management schemes, such as distorting the accuracy of task profiling. Current thermal management techniques do not differentiate whether the heating is caused by task characteristics or by the underlying hardware tendencies; hot jobs can be incorrectly profiled as cold and cold jobs as hot. Hence, more frequent thermal emergencies occur, resulting in performance degradation. On-chip heating is very much intertwined with power dissipation, as the temperatures cause increase in leakage power dissipation and vice versa. The resulting increases in thermal profile have an immediate effect on the energy efficiency of the chip as well.

In this study we investigate the effects of variability on the existing process technologies by real-life analysis/measurement of test chips along with product-level simulations. We make the following observations and contributions:

—cross-correlation of infrared thermal imaging and sensor measurements to quantify and confirm the core-to-core and unit-to-unit thermal variation on test chips;
—analysis of the correlation between workload characteristics and chip heating behavior, using performance counters and thermal sensors; To our best knowledge, this is the first high fidelity study of temperature and performance counter correlation in both single-core and multicore processors;
—a novel variation assessment technique that incorporates a sensor-based characterization stage to generate a high-level chip variation map; The chip variation map is then used by the dynamic power/thermal management of the multicore architecture to improve energy and thermal efficiency;
—enhancing the temperature estimation accuracy by leveraging the chip variation profile map, which is critical for proactive thermal management;
—thermal analysis of the proposed variation assessment scheme through multicore thermal simulations (experimental data indicates that the variation-awareness improves the resulting thermal profile in multicore settings);
—analysis of temperature and performance counters for mixed-workload multithreading runs;
—effects of spatial heat dissipation in the multicore setting with intercore variations.

It is important to note that, even though systematic variations may contribute to the total on-chip variation of chip temperatures, the variation profile is still unique to each individual chip. Therefore it is essential that an effective solution tunes to the individual characteristics of the underlying hardware. Even though the manufacturers can provide such information, we show that run-time assessment using on-chip sensor infrastructure can substitute for manufacturers' guidelines. As a result, this technique can be easily applied at large scale, such as data center level. In a data center application various chips from different manufacturers can be handled using the same

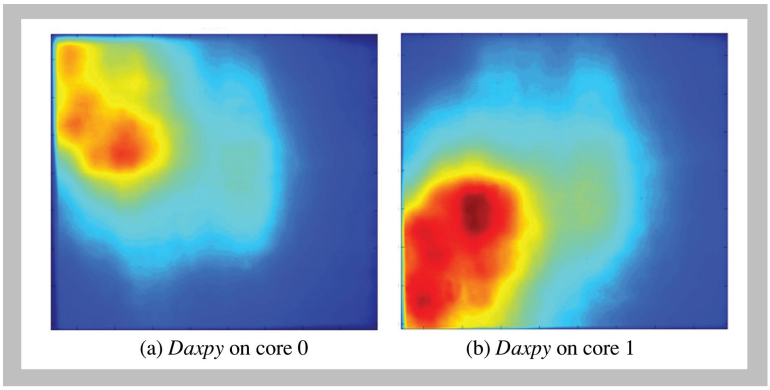

Fig. 1. Measured thermal variation on special test chip.

software technique without depending on manufacturer guidelines, which may or may not exist.

Variation-aware thermal management enables dynamic power/thermal management to adapt to the underlying chip characteristics with no additional performance degradation. Since the system is aware of the existing imperfections on the chip itself, the efficiency is improved. The technique is also effective in dealing with a wide range of thermal variation sources, including process variation inherent to the chip itself. It is capable of alleviating packaging/cooling/placement-induced variations that may occur at the client site, independent of the sources.

Experimental analysis of the test chip shows that the variation-aware thermal management improves the effectiveness of activity migration and thermal-aware scheduling. This, in fact, indicates that the power and thermal efficiency of the chip can be improved by software level management. (Temperature-leakage dependency provides leakage power improvement in the resulting configuration as well, which becomes prominent at high temperatures). It's important to note that the main focus of this study is the power/thermal variations on the chip.

The functional/timing problems due to core-to-core variation can be addressed by setting the clock frequency (and supply voltage) per core, or according to the slowest core on the chip (details of which are beyond the scope of this study). However, power/thermal differences among cores are largely unaddressed by binning and can be exacerbated by voltage frequency adjustments that utilize on-chip variations. It is also important to note that on-chip sensors do not guarantee the efficiency of dynamic power/thermal management under process variation without variation-aware actuation.

Figure 1 illustrates the temperature differences between two cores running the same benchmark, Daxpy, on the same test chip. (Further details of the experiments/setup are discussed in Section 4). Before going into the detailed characterization of on-chip thermal variations, we make three general observations to motivate variation-aware thermal management on chips with intracore variation.

Observation 1. In a case with inherent thermal variation (as in Figure 1)—thermal task profiling is distorted.

In a case like Figure 1, current dynamic power/thermal management schemes are not capable of distinguishing between the heating caused by the inherent tendencies of cores or the characteristics of the task running on them. Therefore even cool jobs running on hot cores may be profiled incorrectly as hot, based on purely reading the thermal sensor data. Similarly, hot jobs that are assigned to cooler cores may be profiled

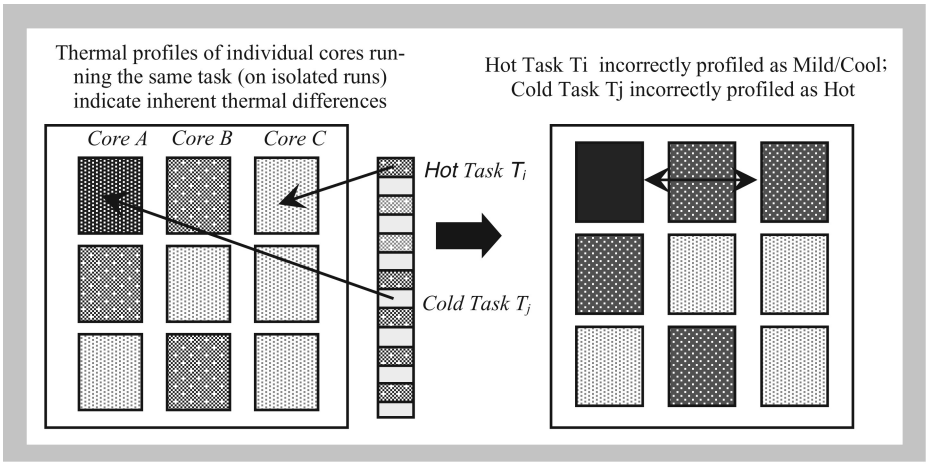

Fig. 2. Effect of thermal variation on temperature-aware task scheduling and migration.

as moderately cool. As a result, assuming that the hardware is uniform and using incorrect profiling data power/thermal management will be ineffective in assigning the tasks. Figure 2 shows an illustration of a 9-core case where Core A is inherently hotter than Core C: a hot Task Ti scheduled on Core C will appear to be cooler than a much cooler Task Tj that is scheduled on the inherently hot Core A. Hence tasks Ti and Tj will be incorrectly profiled and even an optimal temperature-aware scheduling algorithm will not be able to avoid hot spots.

Observation 2. The cores with higher leakage power dissipation have an inherent tendency to heat up more frequently than the others.

This is valid even when all the cores start at the same ambient temperature. Since traditional dynamic thermal management schemes don't differentiate the rates with which cores heat up throttling is commonly used to reduce the resulting heating. Hence, the number and frequency of such unplanned hot spots can cause performance degradation.

Observation 3. Resulting temperature increases translate to higher leakage power at normal server operating temperatures, which causes increased power/cooling/ maintenance costs for the data center.

Power/cooling costs constitute about 40% of the overall data center cost, and as high as 60% of the running cost according to data from US Department of Energy [USDOE2008; Chong2008]. Due to the exponential dependence between on-chip temperature and leakage power, even a few degrees of temperature reduction is likely to translate to observable reduction in the data center running costs. In addition, such power dissipation is not dedicated to useful computation, since it is caused by lack of proper information made available for dynamic power thermal management (DPTM).

As a result, variation awareness is needed for effective management of on-chip temperatures in a multicore architecture. In the next section we discuss details of variation assessment schemes using existing thermal sensor infrastructure and propose variation-aware thermal management at the system software level.

2. THERMAL MANAGEMENT ON MULTICORES WITH INHERENT VARIATION

The variation-aware thermal management technique starts with a high-level characterization of the chip by collecting sensor data during isolated runs over individual

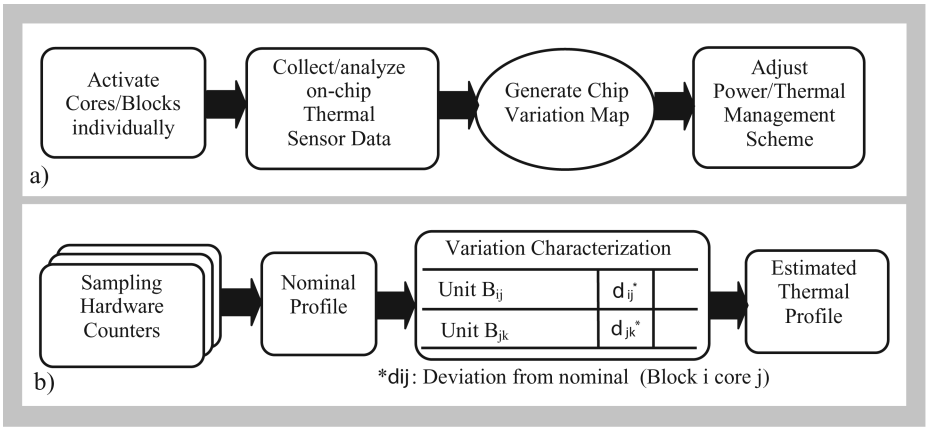

Fig. 3. (a) Generating a variation map for individual cores; (b) Using the variation map for accurate run-time thermal estimation.

cores and architectural units (as shown in Figures 3a and 3b. While it is not possible to completely isolate the computation to a single architectural block, special benchmarks that stress individual architectural units are used for block characterization. The characterization collects thermal sensitivity data for each core on its architecture units as well as for neighboring cores and units.

The data collected during the characterization phase provides the differences among blocks. This data is analyzed to generate a variation map. The deviation of the block temperature from chip average and from blocks with identical functionality is calculated for different starting temperatures and workloads. The multiple data points indicate the leakage differences of the core/block relative to the other cores/blocks.

Each temperature reading is compared to the corresponding hardware counter, which indicates whether the heating is inherent to the block or is caused by the utilization. The block temperatures are ranked in terms of criticality, such that high temperature blocks are identified properly. These three components (temperature deviation, activity counter comparison, and block criticality) are then compiled to represent the variation coefficient of each block. Similarly, a coefficient is assigned per core.

It is important to collect thermal sensor readings in isolation (for individual cores/functional units) during the profiling phase over separate runs in order to isolate the characteristics and filter out the thermal spills from neighboring blocks. This is a key requirement to accurately profile the chip; special benchmarks that stress different units in each core can be used for this stage.

While various on-chip sensors such as temperature sensors and critical path monitors can be used for this profiling phase, we focus on thermal sensors due to wider availability. The variation map is maintained by the system software with power/thermal management schemes. The table can also be maintained in hardware as long as the information is made transparent to the system software and dynamic power/thermal management schemes.

The on-chip variation information can be acquired in a number of ways, including performance measurements per core, temperature measurements, or manufacturer guidelines. Ideally it is best to collect all data as well as other test results during wafer and module testing at the manufacturer's site. However, our experimental results indicate that using even one of these techniques can improve the efficiency. Furthermore, we are not currently aware of any manufacturer providing such information.

Another important point to note is that the variability profile is different for each individual chip. Even though a number of factors such as systematic variations could contribute to similarity among various chips the resulting chips would still appear slightly different when all causes are factored in. Hence, it would be effective to incorporate a universal and self-adjusting scheme capable of addressing all possible cases. As a result, we focus on assessment of variation by the on-chip sensors (chosen for wide-spread availability) as well as utilizing this information at the system software level. At this point it is also important to note that thermal sensor accuracy is another important factor that contributes to the effectiveness of the technique. However, since we correlate the hardware counter data and various on-chip thermal sensor readings, we factor out the minor differences that may be caused by imperfections in the sensor accuracy.

Current microprocessor architectures incorporate a number of on-chip thermal sensors with high accuracy. Hence the proposed technique can easily be used by various thermal management software schemes running on existing multicore architectures with on-chip thermal sensors as well. Variation-awareness can be incorporated into a wide range of power and thermal management schemes.

We illustrate the potential benefits through two dynamic thermal management (DTM) schemes on a real test chip: (1) *Core hopping (Core-level activity migration)* balances the on-chip profile by moving the computation from hotter cores to cooler ones, (2) *Thermal-aware task scheduling,* which uses thermal task profiling to effectively assign tasks on the chip. (3) We also demonstrate thermal-aware scheduling on a multicore architecture model with inherent thermal variations. We show that by scheduling tasks according to the characteristics of the underlying variations we can effectively balance out the thermal profile and reduce the leakage power dissipation at no cost. By adding the chip characterization stage we enable the core hopping and task assignment schemes to compensate for the inherent differences between the cores/units, by utilizing the cooler cores more heavily than their hotter counterparts. Similarly, thermal-aware task scheduling uses the on-chip variation map to accurately profile the threads and tasks to effectively manage the hardware resources to meet software requirements.

3. METHODOLOGY

For the first part of the characterization analysis we have conducted experiments on a real test chip running at 1.5GHz and 1.05V, running Bare Metal Linux [Venton et al. 2005]. There is a 1.44MB L2 cache shared by both cores and 2GB main memory. While it is possible to run at higher supply voltages and clock frequencies, for which the thermal benefits of the proposed schemes are more pronounced, we used more conservative operating points due to the experimental setup limitations. Hence, the experimental results should be interpreted as conservative indicators of potentially higher improvement in terms of temperature. On-chip temperature sensor data was sampled by Linux at each scheduling tick (10ms). Benchmarks from SPEC2000 and SPEC2006 suites were used for the experimental analysis including: integer *Perlbench, Mcf, Hmmer, Libquantum,* and floating point benchmarks: *Milc, Gromacs*, *Namd*, and *Lucas*.

Figure 4 shows the experimental setup used for the thermal analysis. The on-chip sensors are calibrated by comparing the infrared images [Choi et al. 2007]. This kind of calibration is not required for the proposed scheme in general, but it was employed for experimental analysis purposes. (Most manufacturers precalibrate the sensors before the chips are shipped.) The setup in Figure 4 was employed to generate the thermal images presented in the experimental results section.

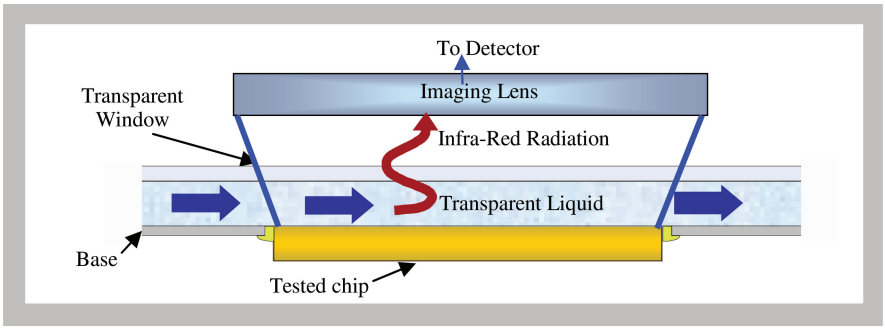

Fig. 4. Experimental setup for real-time thermal imaging.

The imaging setup required replacing the traditional cooling solution with a transparent liquid, to enable the infrared camera take real-life measurements of the underlying chip. The heat sink was replaced during the calibration and imaging phase with a liquid heat sink to enable the infrared imaging analysis. Later, the heat sink was reattached and the runs were repeated to ensure consistency. It is also important to note that variation characteristics are unique to the individual chips and the presented profile is not likely to represent other chips, hence a static solution cannot be applicable. However, the proposed technique can be applied to any multicore architecture; it's capable of handling the unique characteristics of any chip with variations.

In the second part of experimental analysis we used ANSYS®based[1] thermal models in conjunction with internal IBM thermal simulation infrastructure, which include a complete model of the package, front and back-end, as well as the cooling solution based on internal product models. The baseline case peak and average temperature numbers are calibrated against product data.

The power maps used for experimental analysis are the idealized power maps for the test chip taken from the design stage simulations (as illustrated in Figure 1). In the variation simulations package and chip-level variation was injected into the thermal model in the form of TIM conductivity variation, and cooling solution imperfections (up to 20% variation was assumed for the parametric variation case), for which we partitioned the chip into different areas and varied the silicon and packaging parameters of the individual areas to mimic on-chip variations. We believe that the source of variations is not purely process-induced, but includes a significant systematic variation factor such as packaging and chemical-mechanical polishing imperfections (which simulator infrastructures targeting random variation cannot capture). The proposed technique can alleviate variation regardless of the source.

4. EXPERIMENTAL ANALYSIS AND RESULTS

In this section we present measured data from live measurements on a test chip as discussed earlier, as well as simulation results using the thermal modeling infrastructure discussed in Section 3. We start with high-level chip characterization, which is used to generate the variation map. Then, we present improved core hopping and task scheduling enabled by this map. Please note that the measured temperatures are dependent on the characteristics of the experimental setup and corresponding cooling solution; these absolute temperature values should not be interpreted as typical operating temperatures.

[1]ANSYS 11.0. http://www.ansys.com/produts/multiphysics/products.asp

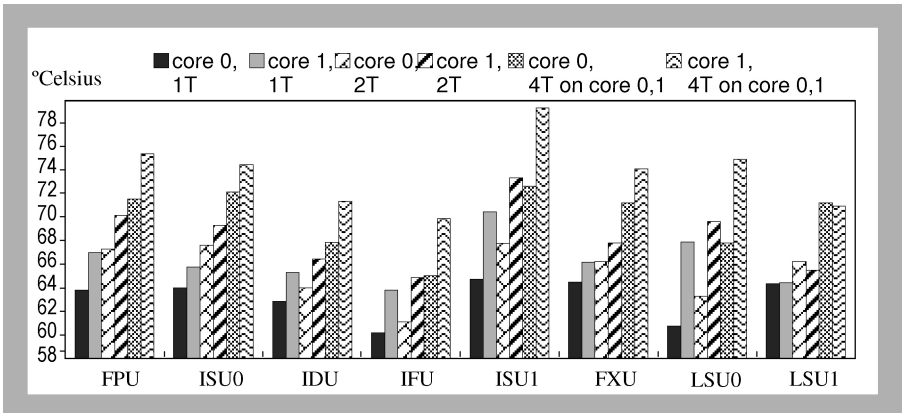

Fig. 5. Peak temperature values for blocks in core 0 and core 1 for SPEC2006 (FPU: Floating Point Unit, ISU: Instruction Scheduler, IFU: Instruction Fetch Unit, FXU: Fixed Point Unit, IDU: Instruction Dispatch Unit, LSU: Load Store Unit).

4.1. Core-to-Core and Within-Core Thermal Variation

In this section we quantify the core-to-core and unit-to-unit temperature variation on the test-chip while cross checking the thermal imaging and temperature sensor data. While the absolute temperature differences among units were different from core-to-core, the general characteristics were consistent among all the test cases we experimented on. Figure 5 displays the measured variation between the two cores on the test chip, running 1–4 threads in single-threaded and simultaneous multithreading (SMT) modes. Core 1 is almost always hotter than core 0 (except for the LSU1, which we will discuss later). The temperature difference between the cores was as high as 6^oC for ISU1 and 7.5^oC for LSU0 during the experiment runs.

ISU1 was the hotspot over all the runs, with peak temperature around 80^oC (when both cores are active in SMT2 mode). Notice that the presented results include isolated core runs used for the variation map as well as multithreaded runs for general characterization. Figure 5 also reveals that thermal profiles within the two cores are not identical. For instance, load store unit, LSU0, is one of the hot spots on core 1; whereas it is relatively cooler on core 0. On the other hand LSU1 has the opposite thermal profile, an elevated temperature for core 0 compared to core 1. Similar thermal differences are observable on other blocks. This clearly illustrates that some of the existing thermal variation on-chip cannot be predicted at design time, hence a run-time technique that relies on measured data from chip sensor infrastructure can effectively alleviate the implications of such variation.

In the test chip layout, the two cores are placed next to each other, where the integer units and load-store structures are immediately in proximity to the neighbor core. It is interesting to note that the LSU0 and LSU1 show significant thermal variation even though they are placed at the center of the chip next to each other. On the opposite sides, the floating point, IFU, and IDU blocks do not see the thermal impact of the other core due to their placement at opposite sides of the chip, which is consistent with the experimental measurements.

Figure 6 shows that the temperature differences are consistent over all the experimented benchmarks from the SPEC2006 suite. On average, the peak core temperature is 4^oC higher on core 1. Simultaneous multithreading causes further increase in the on-chip temperatures. The existing hotspots with high utilization become even more prominent under SMT (such as ISU1, FPU etc.), due to the differences in utilization.

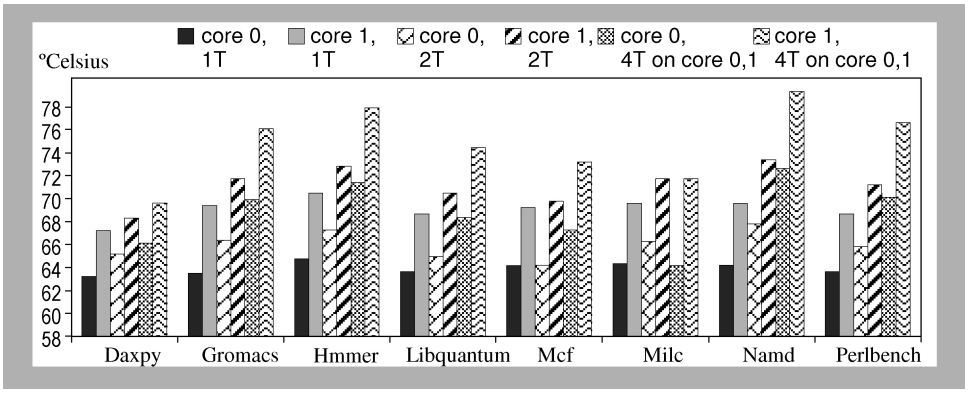

Fig. 6. Peak core temperatures for different SPEC2006 benchmarks.

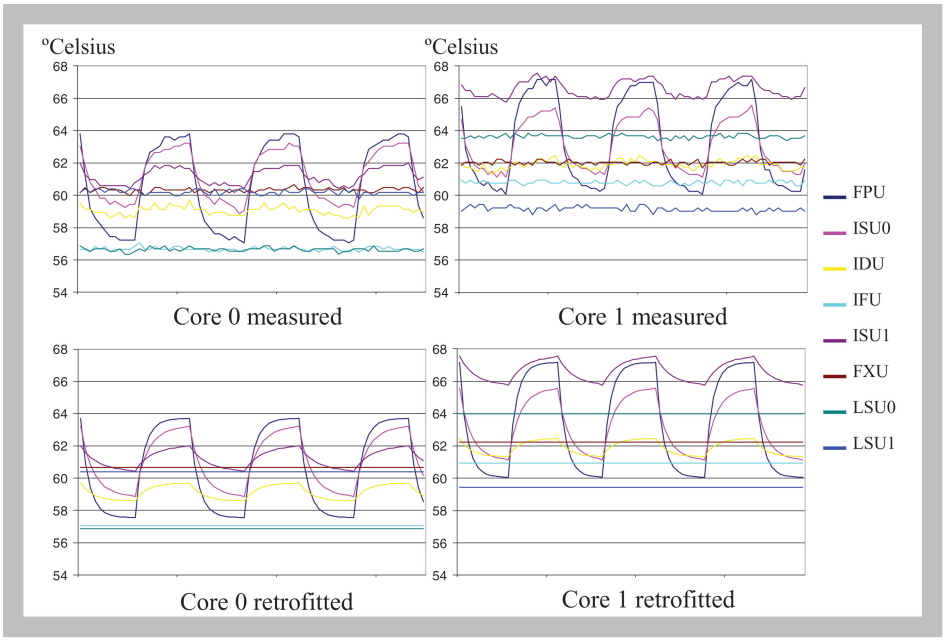

Fig. 7. Heating and cooling thermal characteristics of dual-threaded 100ms core-hopping Daxpy.

To further study the temporal effects of core-to-core variation, we implement core-hopping in BML with a granularity of 100ms. The top two graphs in Figure 7 show the resulting measured thermal sensor data on both cores, clearly showing that the core-to-core variation is consistent over time. In order to characterize the variation, we model the variations with simple exponential equations. The bottom two graphs in Figure 7 show the resulting curves generated by two retrofitted functions characterized by three variables, A, B, and C for each core, as shown in Table I.

Table I. The Values of A, B and C in the Retrofitted Curves

		FPU	ISU0	IDU	IFU	ISU1	FXU	LSU0	LSU1
Core 0	A	63.7	63.2	59.7	57.0	62.0	60.7	56.9	60.4
	B	57.6	58.9	58.6	57.0	60.4	60.7	56.9	60.4
	C	16.7	25.0	25.0	33.3	33.3	33.3	33.3	33.3
		FPU	ISU0	IDU	IFU	ISU1	FXU	LSU0	LSU1
Core 1	A	67.2	65.6	62.5	60.9	67.6	62.2	64.0	59.4
	B	60.1	61.1	61.3	60.9	65.8	62.2	64.0	59.4
	C	16.7	25.0	25.0	33.3	33.3	33.3	33.3	33.3

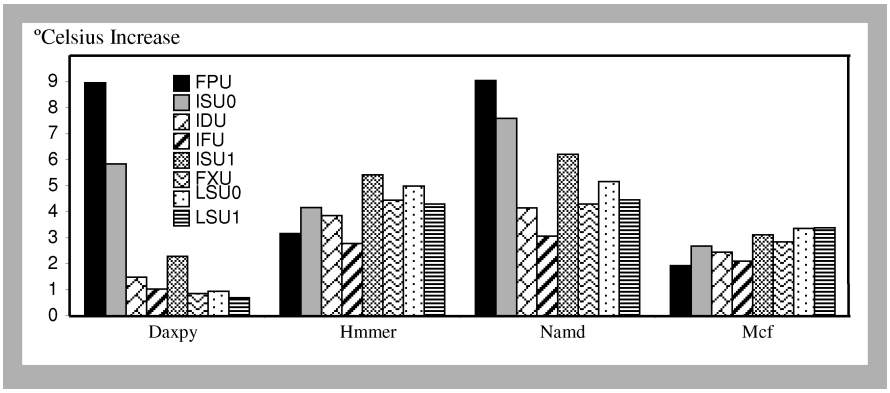

Fig. 8. Increase in the block temperatures for benchmarks in SMT2 mode (y-axis displays the temperature increase in °C).

$$\text{Temperature (rising)} = \text{A} - |\text{A} - \text{B}| * \text{exponent}(-\text{t/C}), \text{where t is number of cycles.}$$

$$\text{Temperature (falling)} = \text{B} + |\text{A} - \text{B}| * \text{exponent}(-\text{t/C}), \text{ where t is number of cycles.}$$

4.2. Performance Counter-Thermal Sensor Correlation

We analyze the correlation between the performance counters and the temperature profile in this section. Figure 8 shows the temperature increase per unit in SMT2 mode compared to Linux idle loop, indicating the correlation between unit temperatures and workload demands. From Figure 8, Namd and Daxpy heat the FPU because of high utilization of the unit.

Neither Hmmer nor Mcf utilizes the FPU, yet FPU temperatures are different because of chip-wide heating. Therefore, we conclude that instructions per cycle and unit activities, together with our variability assessment scheme should be taken into account for temperature estimations. The corresponding hardware counter values are shown in Table II–indicating the correlation between the hardware counter readings and the corresponding heating.

Table III provides essential information needed to generate the variation characterization map as explained in Section 2. It shows that temperature delta between the two cores, core 0 and core 1, is consistent for different benchmarks and for different SMT modes. We observe that the temperature delta was up to four degree Celsius in single-threaded mode. The figure also shows that the temperature variation can occur regardless of the spatial nature of heating, which indicates that there is inherent temperature variation on the chip beyond lateral heat dissipation of the neighbors.

Figure 9 shows the peak temperature over time (1ms interval) for single-threaded and multithreaded runs over core 0 and core 1. The temperature delta across different units is higher for core 1. The peak temperature delta between the two cores is also observable; the differences become more prominent when going from single-thread to

Table II. Benchmark Performance Counter Statistics: (IPC = Instructions per Cycle, FPC = FP Ops per Cycle)

	One Thread		Two Threads			One Thread		Two Threads	
	IPC	*FPC*	*IPC*	*FPC*		*IPC*	*FPC*	*IPC*	*FPC*
Daxpy	1.41	1.33	2.12	1.99	Perlbench	0.85	0.00	1.44	0.00
Namd	0.95	0.35	1.47	0.54	Hmmer	0.93	0.00	1.30	0.00
Gromacs	0.99	0.24	1.38	0.34	Libquantum	0.81	0.00	1.28	0.00
Milc	0.34	0.10	0.50	0.15	Mcf	0.32	0.00	0.54	0.00

Table III. The Steady-State Temperatures when Running Single-Thread and Multithreading on Core 0 and Core 1

Daxpy	Core 0 1T	Core 1 1T	Core 0 2T	Core 1 2T	Perl-bench	Core 0 1T	Core 1 1T	Core 0 2T	Core 1 2T
FPU	62.7	66.0	64.5	67.7	FPU	58.3	60.7	59.2	62.0
ISU0	62.2	64.3	63.5	65.7	ISU0	61.2	62.6	62.5	64.2
IDU	59.7	62.5	59.5	62.4	IDU	61.4	63.7	62.1	64.6
IFU	57.1	61.4	56.7	60.9	IFU	58.7	62.2	59.1	62.9
ISU1	60.7	66.3	61.8	67.4	ISU1	63.1	68.1	65.0	70.5
FXU	59.9	61.7	60.2	61.9	FXU	62.6	63.7	63.8	65.5
LSU0	56.3	63.4	56.6	63.7	LSU0	59.0	65.5	60.7	67.2
LSU1	59.9	58.8	60.1	59.1	LSU1	62.3	60.7	63.5	62.4
Namd	Core 0 1T	Core 1 1T	Core 0 2T	Core 1 2T	Hmmer	Core 0 1T	Core 1 1T	Core 0 2T	Core 1 2T
FPU	62.7	66.0	65.9	69.0	FPU	59.1	62.1	60.3	63.4
ISU0	63.4	65.1	66.5	68.3	ISU0	62.1	63.8	63.9	65.6
IDU	61.5	64.0	63.0	65.7	IDU	62.3	64.8	63.1	65.7
IFU	59.0	62.9	60.3	64.1	IFU	59.5	63.3	60.2	64.1
ISU1	63.6	69.0	66.8	72.4	ISU1	64.3	69.7	66.4	72.1
FXU	62.9	64.7	65.0	67.0	FXU	64.0	65.6	65.5	67.4
LSU0	59.8	66.4	62.3	68.8	LSU0	60.4	67.1	62.2	68.9
LSU1	62.9	61.7	65.2	64.2	LSU1	63.5	62.3	65.2	64.2
Gromacs	Core 0 1T	Core 1 1T	Core 0 2T	Core 1 2T	Lib-quantum	Core 0 1T	Core 1 1T	Core 0 2T	Core 1 2T
FPU	61.4	64.7	63.7	66.8	FPU	58.2	60.8	58.6	61.6
ISU0	62.3	64.5	64.9	66.9	ISU0	61.2	62.5	61.8	63.7
IDU	61.1	64.0	62.4	65.0	IDU	61.3	63.6	61.5	64.1
IFU	58.4	62.8	59.6	63.7	IFU	58.6	62.1	58.6	62.5
ISU1	62.9	68.8	65.6	71.2	ISU1	63.0	68.1	64.2	69.7
FXU	62.3	64.2	64.0	65.9	FXU	62.7	64.1	63.4	65.3
LSU0	59.0	66.1	61.2	67.9	LSU0	58.7	65.3	59.6	66.4
LSU1	62.3	61.5	64.3	63.1	LSU1	62.3	60.6	63.1	61.8
Milc	Core 0 1T	Core 1 1T	Core 0 2T	Core 1 2T	Mcf	Core 0 1T	Core 1 1T	Core 0 2T	Core 1 2T
FPU	61.1	64.9	63.3	66.6	FPU	58.2	61.1	58.1	59.3
ISU0	62.0	64.5	64.4	66.2	ISU0	61.2	62.6	61.0	60.3
IDU	60.2	63.6	61.7	64.6	IDU	60.9	63.4	60.7	61.1
IFU	58.3	62.9	59.8	63.8	IFU	58.8	62.5	58.6	60.4
ISU1	62.5	68.5	65.0	70.5	ISU1	63.3	68.5	63.2	65.3
FXU	62.0	64.4	63.9	65.8	FXU	63.0	64.5	62.8	61.6
LSU0	59.2	66.6	61.6	68.4	LSU0	59.7	66.4	59.8	63.3
LSU1	62.8	62.6	65.3	64.5	LSU1	63.3	62.0	63.3	58.8

two threads. It is interesting to note that the benchmarks do not show much variation in unit temperatures over time.

4.3. The Effects of Mixed Multithreading Workloads

In this section, we study the effect of mixed workloads in multithreading on the same cores. We construct mixed workload combinations that span a range of cases including high-IPC/low-IPC, Hot/Cold and INT/FP combinations. Table IV lists the workload performance counter statistics of each thread when running on the same core in SMT2

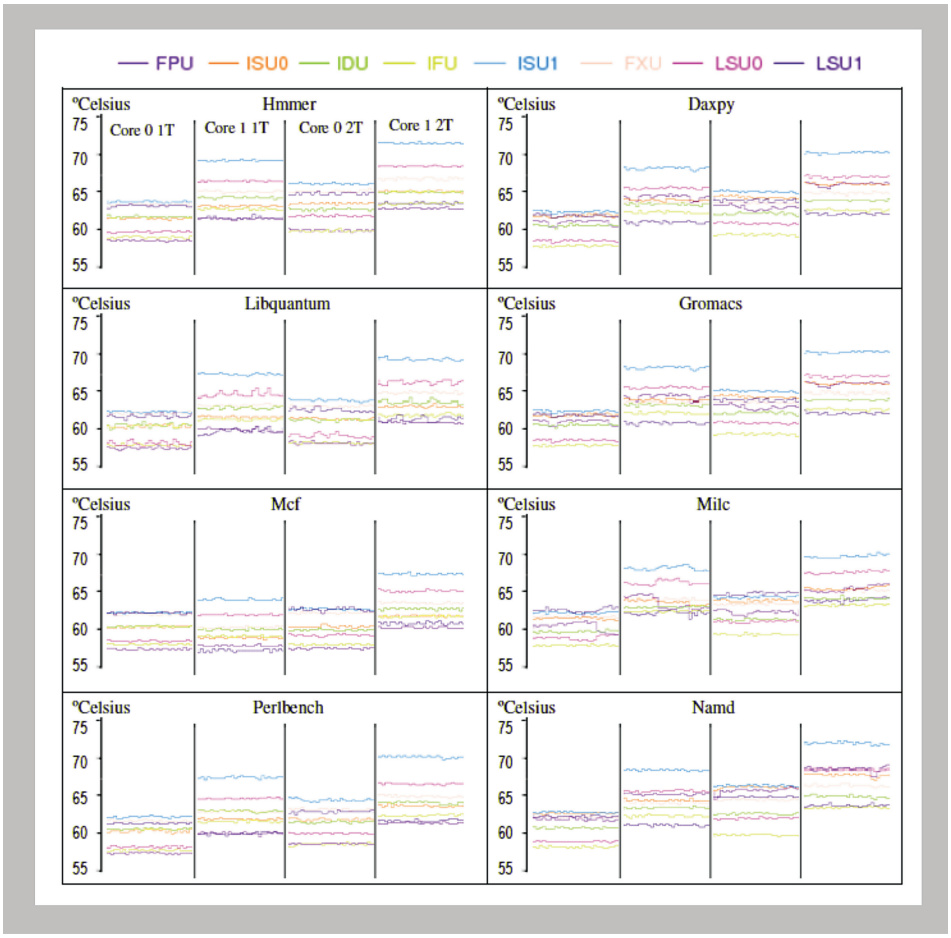

Fig. 9. The temperature effects over time of single-thread and running multithreading on core 0 and core 1.

Table IV. Benchmark Performance Counter Statistics for Mixed Multi-Threading Workloads

		Per-thread IPC	Per-thread FPC	Total IPC	Total FPC		Per-thread IPC	Per-thread FPC	Total IPC	Total FPC
Thread 0	Namd	0.87	0.32	1.43	0.32	Namd	0.74	0.27	1.00	0.27
Thread 1	Hmmer	0.56	0.00			Mcf	0.26	0.00		
Thread 0	Milc	0.29	0.08	0.99	0.34	Milc	0.31	0.09	0.90	0.09
Thread 1	Namd	0.70	0.25			Hmmer	0.59	0.00		
Thread 0	Milc	0.27	0.08	0.50	0.08	Mcf	0.27	0.00	0.84	0.00
Thread 1	Mcf	0.23	0.00			Hmmer	0.57	0.00		

mode. The FPC/IPC comparison clearly indicates the nature of workload in terms of floating point/integer computations. The temperature profiles of these workloads are shown in Figure 10 with corresponding hot spots in floating point and fixed point units.

Until this point, the analysis was focused on the overall heating on the functional units while stressing with SPEC2006 benchmarks. Next, we focus on the time-varying nature of the heating on the chip: showing how the on-chip temperatures go through different phases that correlate with the hardware counter readings, which indicate elevated activity. Figure 11 demonstrates the correlation between Milc and Lucas, as

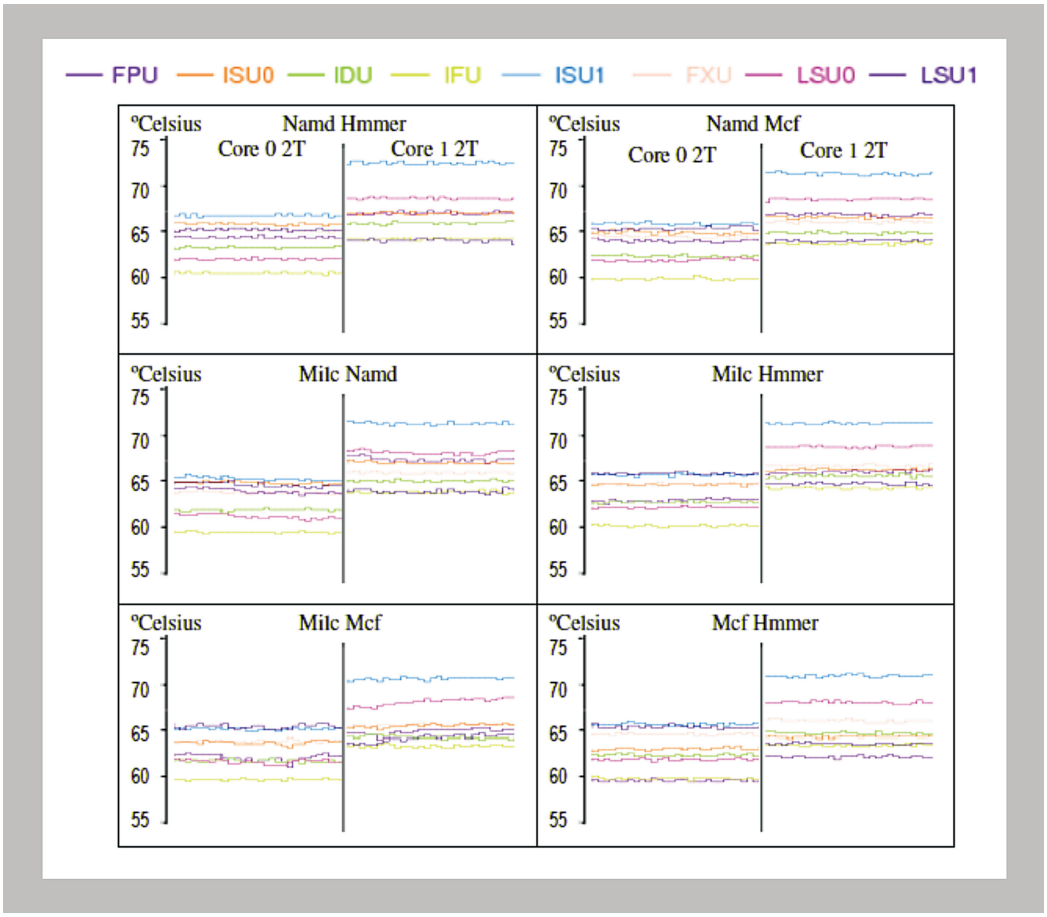

Fig. 10. The temperature effects over time of mixed multithreading workload on core 0 (left) and core 1 (right).

well as their phase behavior. The top part of the figure shows the thermal sensor readings; the bottom shows the performance counters normalized to the number of cycles. Notice that the activity counter peaks are followed by the thermal peaks with a delay which is due to the thermal time constant; the same effect is observable for cooling period.

Another point to note is the difference between Milc and Lucas temperature profiles. The length and intensity of the high-activity period in Lucas cause the temperatures to rise considerably in the FPU, whereas the rapid switches between high and low activity modes do not translate to on immediate temperature difference in Milc. This clearly illustrates the low-pass filter character of temperature on the underlying hardware activity.

Figure 12 shows the correlation between the hardware counters and on-chip temperatures for FPU. The highlighter sampling interval of 50 msec at the bottom left gives the one-to-one correlation between the counters and on-chip temperature readings. The correlation for the lower and higher values is not as clear as the 50msec case.

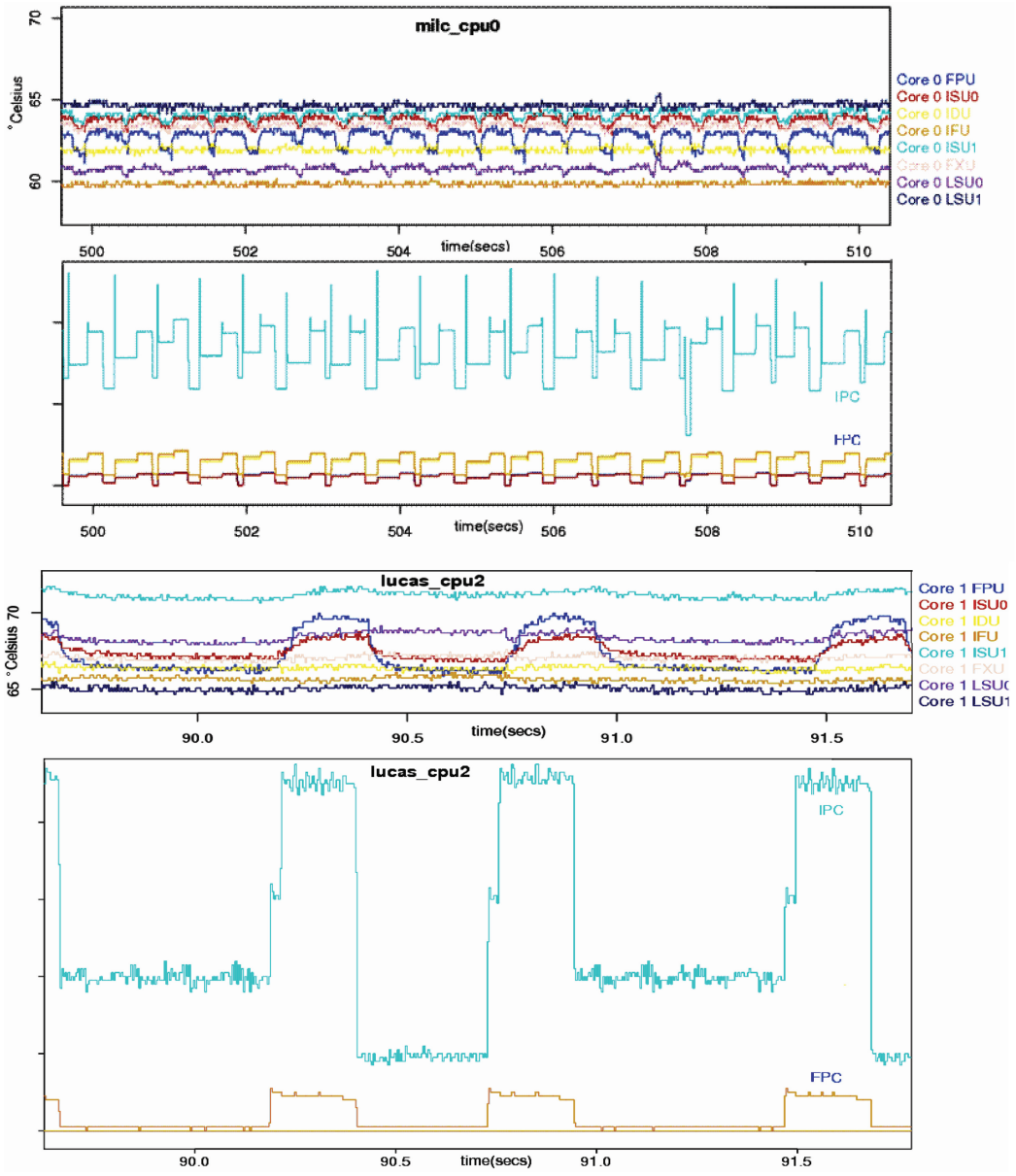

Fig. 11. Thermal sensor and activity counters (FPC) and IPC for Milc. Thermal sensor and activity counters (FPC) and IPC for Lucas.

4.4. Effects of Core-to-Core Variation on Spatial Heating

To study the effects of spatial heating, we pin Daxpy to always run on one core and compare the temperature of the other core. Figure 13 shows the thermal-spatial effects of how running Daxpy on one core heats up the other. In Figure 13, three properties are observed: (1) The curves of the idle cores are quite similar. This is because the two cores are arranged as flipped images of each other, as shown in Figure 1. (2) FXU,

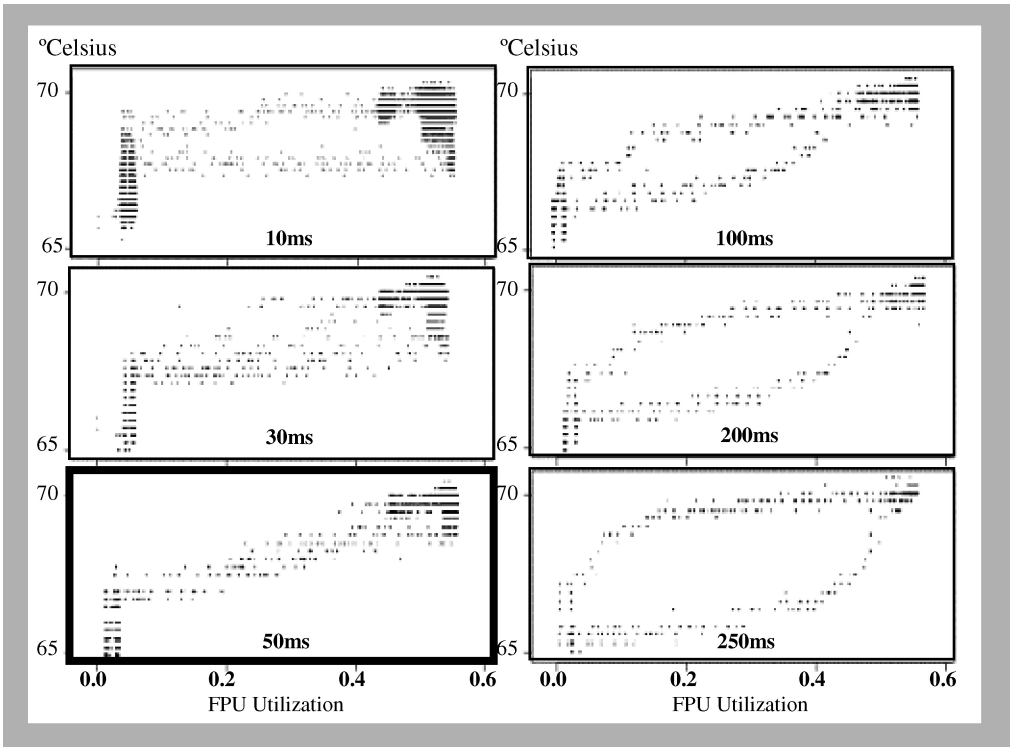

Fig. 12. Hardware counter-temperature correlation for FPU for different sampling intervals for Lucas: y-axis: Celsius, x-axis: FPU utilization for 10–250 msec range.

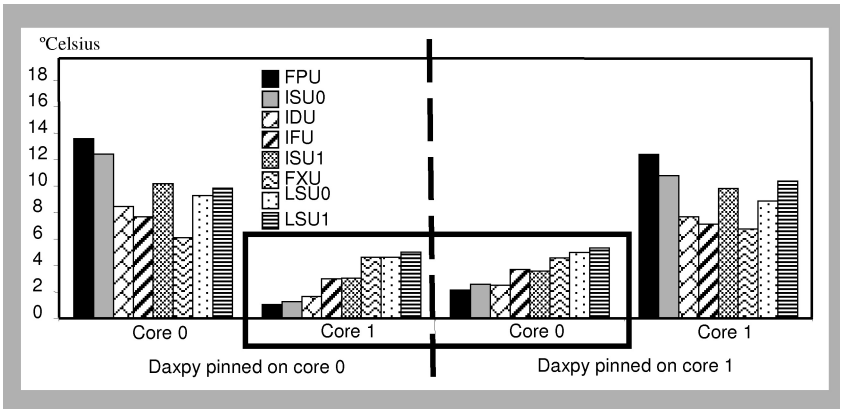

Fig. 13. Effects of spatial heating—temperature profiles of units on cores 0 and 1, with Daxpy pinned on core 0 and core 1.

LSU0, and LSU1 on the idle cores are heated more because they are spatially closer to the busy core; FPU, which is furthest from the other core is least affected. (3) Core 0 is spatially heated slightly more because core 1 is inherently hotter when running Daxpy. Although the variation in spatial heating is minor, the variation map can also capture this spatial thermal effect.

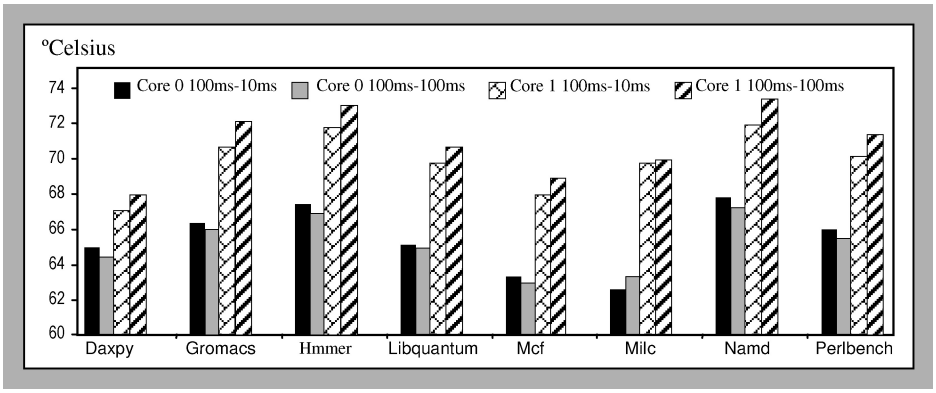

Fig. 14. Peak temperatures over SPEC2006.

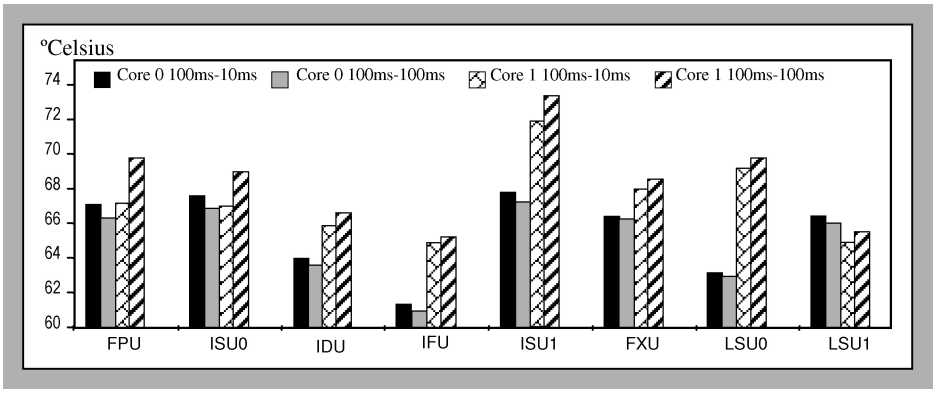

Fig. 15. Peak block temperatures for core0 and core1 in SMT2 mode.

4.5. Variation-Aware Activity Migration

We experimented with preemptive activity migration to demonstrate the potential improvement with variability awareness. Even though reactive activity migration is equally applicable, our goal is to avoid the heating by effectively distributing the computation between two cores. The core hopping interval is adjusted to adapt to the variation between the intrinsic thermal profiles of the two cores.

Figures 14 and 15 illustrate the effects of core hopping on peak block temperatures within the core over various benchmarks. Equal interval activity migration is compared to variability-aware activity migration (at the bottom) for Namd in Figure 16 By choosing the core hopping intervals to be shorter on the hotter core, 1, the corresponding hotspots are reduced (100ms/Core0—10ms/Core1), whereas core 0 does not heat up significantly due to the higher utilization.

Figure 16 illustrates the heating behavior in time. In the top part of the figure the '100ms–100ms' symmetrical core hopping interval causes core 1 to heat up, compared to the '100ms–10ms' case, where the cooler core, 0, is utilized more intensely to improve the thermal profile.

The effects of symmetrical and asymmetrical intervals are displayed for core 1 running Daxpy, as shown in Figure 17 shows the resulting thermal image of the variation-aware asymmetric activity migration, for which the on-chip temperatures are observably reduced (compared to Figure 1).

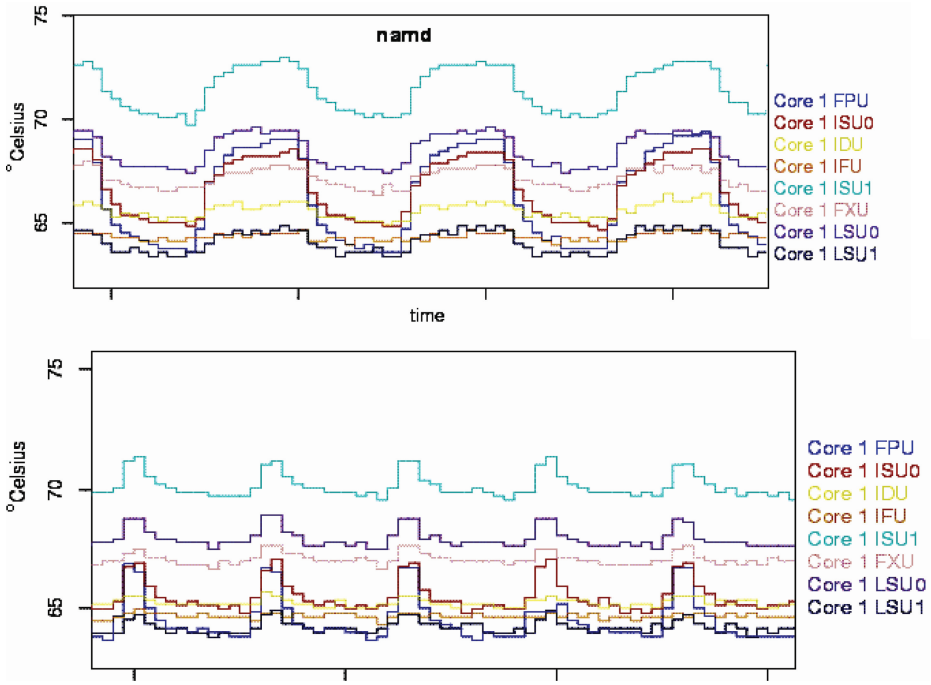

Fig. 16. Effect of activity migration intervals on block temperature (Hotspot ISU - in light blue; block temperatures color coded).

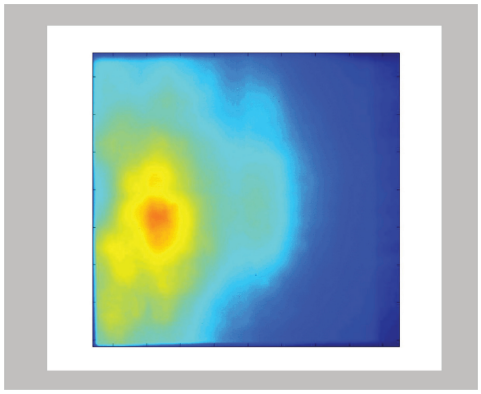

Fig. 17. Infrared thermal image (thermal map) for Daxpy with proposed variation-aware activity migration.

4.6. Variation-Aware Task Scheduling

To further demonstrate the advantages, we implemented the proposed variation-awareness in thermal-aware task scheduling. In this experiment, we ran workloads that utilize all four hardware threads on the chip. Figure 19 shows the resulting temperatures over idle loop, where the reduction in maximum chip temperature through variation-aware scheduling of hot threads from Spec2006 (e.g. *Namd, Gromacs, Milc)* and cold threads (e.g. *Hmmer, Mcf)* are illustrated—as previously measured and characterized in Figure 6. Without the assessment scheme, an oblivious task scheduler can assign the hot threads to the hotter core, resulting in higher peak temperature.

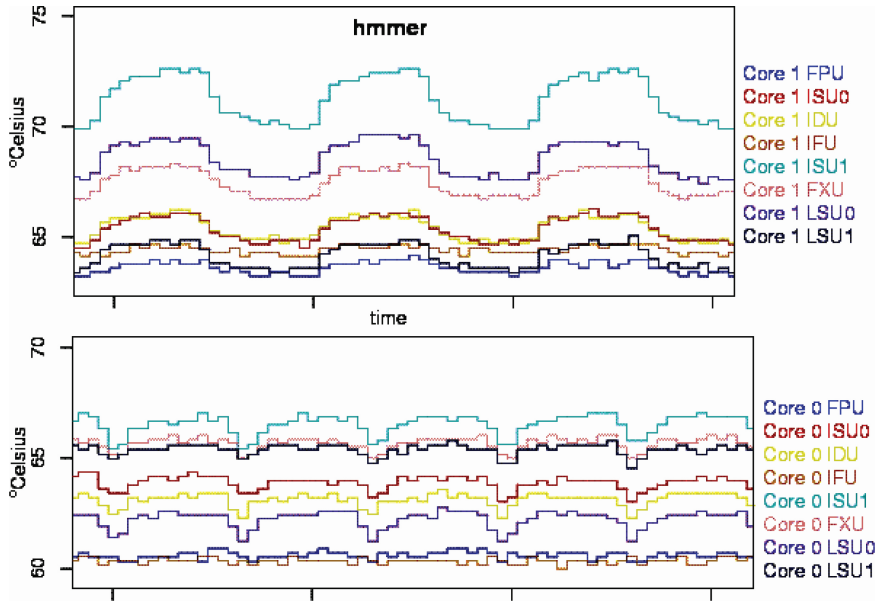

Fig. 18. Temperature profile for Hmmer 100-100ms intervals, 100-10ms migration intervals.

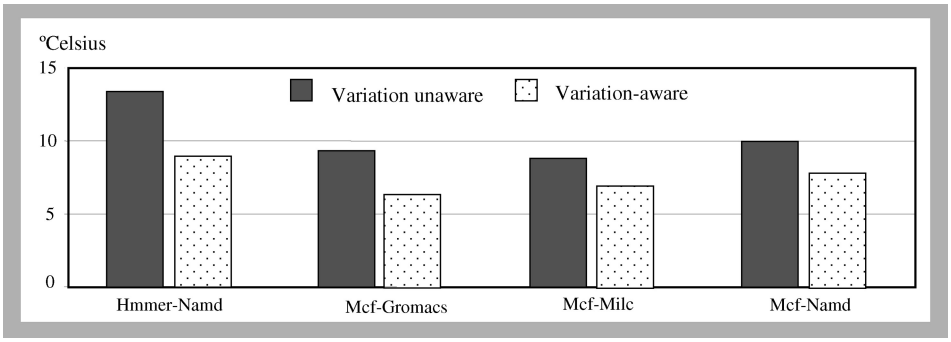

Fig. 19. Peak temperatures (relative to idle loop) for alternative scheduling schemes: variation-unaware and variation-aware.

The Hmmer-Namd experiment shows that by using the proposed scheme the peak temperatures can be reduced by 4.5°C, while overall the experiments show 1.8°C–4.5°C reduction in peak temperatures. The two cores have the same IPC at the specified clock frequency. Therefore there is no performance cost for reducing the peak temperatures with variation-awareness. The experiment shows that by assessing a chip's hardware characteristics, an OS or workload scheduler can achieve reduction in on-chip temperatures through intelligent distribution of the workloads on the multicore architecture.

4.7. Simulated Thermal Variation in Multicore Case

Figure 20 shows the thermal modeling of the preceding case extended to 4 cores. In order to mimic on-chip variation we simulated the 4 cores by injecting proportionate variation in the thermal conductivity of the interface material (Core 1/top left (10%), Core 2/top right (0%), Core3/bottom left (20%), Core 4/bottom right (10%)). Please note that the thermal variation can be caused by various reasons—such parametric

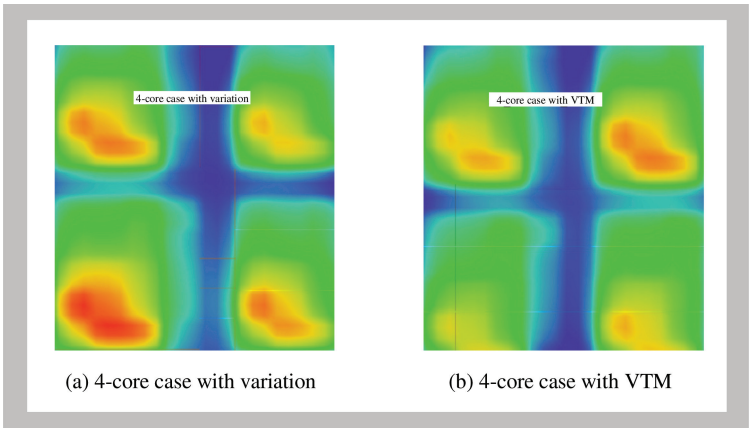

Fig. 20. (a) 4 core with variation: (b) With variation aware task allocation.

variation is only one example of the resulting differences in heating profiles—similar variation profiles could be simulated with different parameters as well. We used synthetic benchmarks for scheduling, where the starting power maps were based on Daxpy as mentioned in the preceding, with ±20% power induced in addition to the base power in steps of 10%. In all cases the additional power, was applied uniformly to the power map.

In part (a) Daxpy is scheduled on the cores; due to the induced variation the cores, exhibit slightly different thermal profiles while even running the same task. An identical power map was applied to all 4 cores; the resulting thermal differences are due to the variation in the hardware model. This phase indicates the inherent thermal differences between the cores (the core at the bottom left heats up more than the rest of the cores). After generating the variation map, the task scheduling intelligently runs lower power task on bottom left core with the highest peak temperatures according to the variation map and migrates the higher power tasks from the task queue to the top right core, which sees additional benefits as a result of this. The lowest power task is scheduled for Core 3(bottom left)—and the highest power task for Core 2(top right). The peak temperature is reduced by 4^oC in this particular example. The resulting thermal profile shows the reduction in the hotspot area in the bottom left core. While the variation can be caused by number of factors including process variation, imperfection in the cooling solution, thermal interface materials and such, the variation-aware thermal management targets all such cases by generating the resulting variation map that includes the contributions from all such sources.

5. RELATED WORK

While there is an extensive amount of work on dealing with the impact of on-chip variability on microprocessor design as well as the thermal management of multi-core architectures, variation implications on thermal management was unexplored in the test-chip/silicon characterization context to the best of our knowledge. With the increases in total power dissipation many-core architectures are challenged by the thermal design and need more sophisticated thermal management algorithms [Huang et al. 2008].

Monchiero et al. [2008]; Donald [2006]; Chapparo et al. [2004, 2007]; Humenay et al. [2007]; and Gargand Marculescu [2008] provide an in-depth look at the power/thermal tradeoff for multi-core architectures in such contexts. While most the thermal management is still handled by the on-chip hardware controllers–along with the power

management, the relatively longer time constants for temperature (compared to much faster power fluctuations) provide an opportunity to manage at operating system scheduling tick-level; the system software can effectively regulate the temperatures with each scheduling assignment [Choi et al. 2007]. It has been shown by a number of studies that task combination and assignment impact both performance and the resulting temperatures [Tiwari et al. 2007; Chung and Skadton 2008].

Intelligent operating system scheduling can also improve the energy efficiency [Devuyst et al. 2006], as well as thermal profile of the chip. Gunther et al. [2007], Gomma et al. [2005], Marculescu and Talpes [2005], Winter and Albonesi [2008], Coskun et al. [2008], and Stavrou and Trancoso [2007] have looked at the potential benefits of temperature-aware scheduling for multiprocessor architectures, showing significant reduction in the resulting temperatures with temperature-aware scheduling. The aforementioned techniques are variation-unaware in managing the on-chip temperatures. The on-chip thermal profile can be improved—the proposed technique is orthogonal to the prior art.

Microarchitecture design for variability has been studied by various researchers within the context of the performance and functional implications on such variation [Bernstein et al. 2006; Liang and Brooks 2006; Das et al. 2007; Herbert and Marculescu 2009; Romanescu et al. 2009; Teodorescu et al. 2007; Teodorescu and Torrellas 2008; Soundararajan et al. 2008]. The functional/timing problems due to core-to-core variation can be addressed by setting the clock frequency and supply voltage per core, or according to the slowest core on the chip [Sarangi et al. 2008; Humenay 2007]. Marculescu and Garg [2008] proposed techniques to handle process variation through voltage and frequency adjustments. Unit and macrolevel solutions were also proposed for timing and performance aspects of variation problems, but do not target the thermal implications of variation in the multicore context.

6. CONCLUSIONS

Temperature variation is caused by a range of reasons including process variation in the form of increased leakage power in some cores, packaging imperfections, thermal grease or interface material imperfections, as well as cooling solution or other manufacturing complications. However, current chip-level power/thermal management is oblivious of such thermal variation; so it's not able to deal with the implications effectively. As a result, the frequency of thermal emergencies increases and the total leakage power is elevated. We propose variation-aware thermal management techniques enabled by a chip variation assessment scheme using on-chip thermal sensors. The thermal sensor data are processed to construct a one-time chip variation map to improve the effectiveness of dynamic power/thermal management techniques. The resulting on-chip variation map is made available to the system software, which can then effectively manage the inherent heating tendencies of the units/cores on the chip. We show through real test-chip measurements that this technique can provide up to 4.5^{o}C reduction in the peak temperatures without any performance loss. The thermal modeling further indicates that the scheme can be effective in dealing with an increased number of cores in the multicore architecture. As a result of the reduction in the peak temperatures, energy efficiency is also improved, which can be beneficial for the existing multicore architectures in use today.

REFERENCES

Bernstein, K., Frank, D. J., Gattiker, A. E., Haensch, W., Ji, B. L., Nassif, S. R., Nowak, E. J., Pearson, D. J., and Rohrer, N. J. 2006. High-performance CMOS variability in the 65-nm regime and beyond. *IBM J. Res. Devel. 50,* 4/5, 433–449.

CHAPARRO, P., GONZALEZ, J., MAGKLIS, G., CAI, Q., AND GONZALEZ, A. 2007. Understanding the thermal implications of multi-core architectures. *IEEE Trans. Parall. Distrib. Syst. 18,* 8,1055–1065.

CHAPARRO, P., GONZALEZ, J., AND GONZALEZ, A. 2004. Thermal-aware clustered microarchitecture. In *Proceedings of the IEEE International Conference on Computer Design,* 48–53.

CHOI, J., CHER, C.Y., FRANKE, H., HAMANN, H., WEGER, A., AND BOSE, P. 2007. Thermal-aware task scheduling at the systems software level. In *Proceedings of the International Symposium on Low Power Electronics Design,* 213–218.

CHONG, F. 2008. Towards more sustainable computer design. In *Proceedings of the IEEE International Conference on Computer Design.*

CHUNG, S. W. AND SKADRON, K. 2008. A novel software solution for localized thermal problems. In *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing and Applications*, 63–74.

COSKUN, A. K., SIMUNIC ROSING, T., WHISNANT, K. A., AND GROSS, K. C. 2008. Static and dynamic temperature-aware scheduling for multiprocessor SoCs. *IEEE Trans. VLSI Syst. 16,* 9.

DAS, A., OZDMIR, S., MEMIK, G., ZAMBRENO, J., AND CHOUDHARY, A. 2007. Mitigating the effects of process variations: Architectural approaches for improving batch performance. *In Proceedings of the International Symposium on Computer Architecture,* 129–136.

DE VUSYT, M., KUMAT, R., AND TULLSEN, D. 2006. Exploiting unbalanced thread scheduling for energy and performance on a CMP of SMT processors. In *Proceedings of the Parallel and Distributed Processing Symposium.*

DONALD, J. AND MARTONOSI, M. 2006. Power efficiency for variation-tolerant multi-core processors. *In Proceedings of the International Symposium on Low Power Electronics Design,* 304–309.

DONALD, J. AND MARTONOSI, M. 2006. Techniques for multi-core thermal management: Classification and new exploration. In *Proceedings of International Symposium on Computer Architecture,* 78–88.

GARG, S. AND MARCULESCU, D. 2008. System level throughput analysis for process variation adaptive multiple voltage-frequency island designs. *ACM Trans. Des. Auto. Electron. Syst. 13,*4,1–25.

GHIASI, S., KELLER, T., AND RAWSON, F. 2005. Scheduling for heterogeneous processors in server systems. In *Proceedings of the Conference on Computing Frontiers,* 199–210.

GOMAA, M., POWELL, M. D., AND VIJAYKUMAR, T. N. 2004. Heat and run: Leveraging SMT and CMP to manage power density through the operating system. In *Proceedings of Architectural Support for Programming Languages and Operating Systems,* 260–270.

GUNTHER, S., BINNS, F., CARMEAN, D., AND HALL, J. 2001. Managing the impact of increasing microprocessor power consumption. *Intel Technol. J. 5.*

HANSON, H., KECKLER, S., RAJAMANI, K., GHIASI, S., RAWSON, F., AND RUBIO, J. 2007. Power, performance and thermal management of high-performance systems. In *Proceedings of International Parallel and Distributed Processing Symposium,*1–8.

HERBERT S. AND MARCULESCU, D. 2009. Mitigating the impact of variability on chip-multiprocessor power and performance. *IEEE Trans. VLSI Syst. 17,* 10,1520–1533.

HUANG, W., STAN, M., SANKARANARAYANAN, K., RIBANDO, R., AND SKADRON, K. 2008. Many-core design from a thermal perspective. In *Proceedings of IEEE/ACM Design Automation Conference,* 746–749.

HUMENAY, E., TARJAN, D., AND SKADRON, K. 2007. Impact of process variation on multi-core performance symmetry. In *Proceedings of the Design Automation and Test Conference in Europe,* 1653–1658.

LIANG, X. AND BROOKS, D. 2006. Microarchitecture parameter selection to optimize system performance under process variation, In *Proceedings of the International Conference on Computer- Aided Design,* 429–436.

MARCULESCU, D. AND TALPES, E. 2005. Variability and energy awareness: A microarchitecture-level perspective. In *Proceedings of the Design Automation Conference,* 11–16.

MARCULESCU, D. AND GARG, S. 2008. Process-driven variability analysis for single and multiple voltage-frequency island, latency-constrained systems. *IEEE Trans. Comput.-Aid. Design Integr. Circuits Syst. 27,* 5,893–905.

MONCHIERO, M., CANAL, R., AND GONZALEZ, A. 2008. Power/performance/thermal design-space exploration for multi-core architectures. *IEEE Trans. Parall. Distrib. Syst. 19,* 5, 666–681.

ROMANESCU, B., BAUER, M. E., OZEV, S., AND SORIN, D. 2008. Reducing the Iimpact of intra-core process variability with criticality-based resource allocation and prefetching. In *Proceedings of the International Conference on Computing Frontiers,* 129–138.

SARANGI, S., GRESKAMP, B., TIWARI, A., AND TORRELLAS, J. 2008. EVAL: Utilizing processors with variation-induced timing errors. In *Proceedings of the 41st International Symposium on Microarchitecture (MICRO).*

SOUNDARARAJAN, N., YANAMANDRA, A., NICOPOULOS, C., NARAYANAN, V., SIVASUBRAMANIAM, A., AND IRWIN, M. J. 2008. Analysis and solutions to issue queue process variation. In *Proceedings of Dependable Systems and Networks (DCCS),* 11–21.

STAVROU, K. AND TRANCOSO, P. 2007. Temperature-aware scheduling for future chip multiprocessors. *EURASIP, J. Embe. Systems,* Article ID 48926.

TEODORESCU, R. AND TORRELLAS, J. 2008. Variation-aware application scheduling and power management for chip multiprocessors. In *Proceedings of the 35th Annual International Symposium on Computer Architecture (ISCA).*

TEODORESCU, R., NAKANO, J., TIWARI, A., AND TORRELLAS, J. 2007. Mitigating parameter variation with dynamic fine-grain body biasing. In *Proceedings of the 40th International Symposium on Microarchitecture (MICRO).*

TIWARI, A., SARANGI, S. R., AND TORELLAS, J. 2007. ReCycle: Pipeline adaptation to tolerate process variation. In *Proceedings of the International Symposium on Computer Architecture*, 323–334.

USDOE. 2008. United States Department of Energy, Energy Efficiency and Renewable Energy, Data Center Energy Efficiency Program.

WINTER, J. AND ALBONESI, D. 2008. Addressing thermal non-uniformity in SMT workloads. *Trans. Arch. Code Optim. 5,*1.

VENTON, T., MILLER, M., KALLA, R., AND BLANCHARD, A. 2005. A Linux-based tool for hardware bring up, Linux development and manufacturing. *IBM Syst. J. 44,* 2, 319–329.

Received September 2009; revised April 2010, August 2010; accepted October 2010