Power Multiplexing for Thermal Field Management in Many-Core Processors

Minki Cho, *Student Member, IEEE*, Chad Kersey, *Student Member, IEEE*, Man Prakash Gupta, Nikhil Sathe, Satish Kumar, Sudhakar Yalamanchili, *Senior Member, IEEE*, and Saibal Mukhopadhyay, *Senior Member, IEEE*

***Abstract*— This paper presents the effect of proactive spatiotemporal power multiplexing on the thermal field in many-core processors. Power multiplexing migrates the locations of active cores within a chip after each fixed time interval, referred to as the migration interval, to redistribute the generated heat and thereby reduce the peak temperature and spatial and temporal nonuniformity in the thermal field. Clock and supply gating is used to minimize the power of the deactivated cores. The control of the migration interval is studied considering a 256-core processor at the predictive 16-nm node to evaluate the conflicting impact of the migration interval on thermal field and system performance.**

***Index Terms*— Dark silicon, many-core processor, performance, power multiplexing, thermal management.**

I. Introduction

A HIGHER on-chip temperature degrades circuit performance, increases system leakage power, degrades circuit reliability, and reduces cooling efficiency (i.e., increases cooling energy) [1]–[10]. Consequently, managing the peak temperature and maximizing the performance under a peak temperature constraint for single or multicore processors have received significant attention over the years. Reducing the power dissipation of the cores using low-power design solutions or dynamic power reduction (e.g., dynamic voltage frequency scaling) reduces heat generation, whereas better cooling solutions increase the heat removal (or outflow) from the system. Both approaches can effectively manage temperature but are limited by advancements in technology (cooling and process technology) and constraints on performance and functional correctness. Specifically, the exponential increase in leakage current and reduced gate overdrive (i.e., supply voltage minus threshold voltage) in deep nanometer nodes limits the ability to reduce power dissipation. These and related technology constraints have led to recent projections of the advent of "dark silicon" [11], where areas of the die will have to be power-gated to satisfy power and thermal constraints. Consequently, we must consider techniques that operate under a fixed power constraint, distribute this power effectively under thermal constraints, and is extensible to dark silicon scenarios.

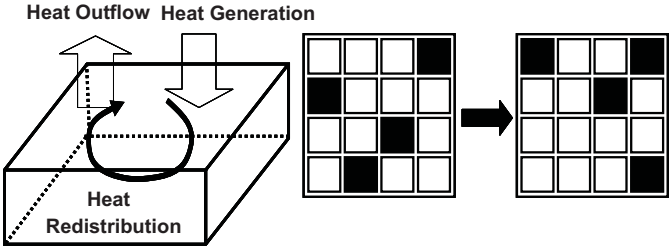


Fig. 1. Basic concept of thermal management with power multiplexing.

From the perspective of thermal management, methods have been explored to manage temperature by optimally redistributing the generated heat over space and/or time [12]–[23]. In principle, this can be achieved by optimally allocating the locations of high-power dissipations in space and modulating these locations over time (Fig. 1). Thermal-aware task assignments and scheduling approaches aim to obtain the optimal spatial redistribution using steady-state temperature redistribution. Further advantage can be obtained by migrating computation from the hotter to the cooler regions over time (temporal heat redistribution) using task, thread, or activity migrations.

The task, thread, or activity migrations are normally invoked in response to a peak temperature violation and to ensure that the peak temperature remains below a predefined threshold. However, with the increasing demand for energy efficiency and reliability, there is a need to consider the thermal behavior of an IC as a primary optimization objective instead of only constraints. For many-core chips, this requires analyzing and controlling the spatiotemporal behavior of the thermal field along with peak temperature. It is expected that the computational load in a many-core processor will not always require all the cores to be simultaneously active. Further, the peak temperature constraint can also limit the number of

Manuscript received May 7, 2012; revised August 23, 2012; accepted August 24, 2012. Date of publication November 29, 2012; date of current version January 4, 2013. This work was supported in part by the Semiconductor Research Corp (#2084.001), Intel, and the IBM Faculty Award. Recommended for publication by Associate Editor A. Bhattacharya upon evaluation of reviewers' comments.

M. Cho, C. Kersey, N. Sathe, S. Yalamanchili, and S. Mukhopadhyay are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: mcho8@gatech.edu; chad.d.kersey@gmail.com; nikhil.sathe@gatech.edu; sudha@ece.gatech.edu; saibal@ece.gatech.edu).

M. P. Gupta and S. Kumar are with the School of Mechanical Engineering, Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: mp.gupta@gatech.edu; satish.kumar@me.gatech.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCPMT.2012.2220546

simultaneously active cores [8], [11]. Under such scenarios, the core-level power and clock gating techniques can be used to reduce the power dissipation of the inactive cores to a very low level [24], [25]. The core-level power control, coupled with spatiotemporal migrations of locations of active cores can be used to realize spatiotemporal power migration in many-core processors. However, activation of different numbers or different locations of active cores can result in different maximum temperatures, locations of hot spots, and spatial variation. Further, with power migration each core experiences dynamic power variations due to sequences of active and inactive states (along with dynamic power control techniques), resulting in temporal variation in temperature or thermal cycles. The spatiotemporal nonuniformity in the thermal field is detrimental to both reliability and cooling efficiency. Large and fast temporal variation in temperature (i.e., faster thermal cycles) is detrimental to circuit reliability [6]. Spatial variations in temperature result in nonuniform delays for different cores and increase the total leakage power [1], [9]. The higher maximum temperature and spatiotemporally nonuniform temperature distribution also increase the cooling energy [10].

This paper presents a detailed analysis of how spatiotemporal power migration using activation and deactivation of cores can be used to minimize the peak temperature and spatiotemporal nonuniformity in a many-core processor under less than 100% utilization. In particular, this paper characterizes the thermal behavior of a predictive many-core processor to show the impact of controlling the migration interval (how fast migrations are performed) on the overall thermal field of a predictive 256-core processor designed at the 16-nm node. With detailed simulation using a distributed RC-based 3-D thermal model, we first show that faster migration helps to minimize the peak temperature and spatiotemporal nonuniformity in the global thermal field. On the other hand, through architectural simulations of application benchmarks we show the implications of faster migration on the overall performance. The combined effect of the two is considered to study the implications of migration interval on the overall system performance considering different migration policies. Since we aim to analyze and manage the thermal field instead of only maximum temperature, the proposed approach is referred to as "thermal field management."

The rest of the paper is organized as follows. Section II presents the related work. Section III presents the principles and attributes of power multiplexing. Section IV discusses the simulation environment. Section V presents the simulation results. Section VI discusses the tradeoffs for the migration interval and Section VII presents the conclusions.

II. Related Work and Contributions

Several architectural policies have been developed over the years to implement the principle of spatial and/or temporal heat redistribution for thermal management in many-core processors. Chaparro *et al.* [3] considered thread migration (TM or core hopping) and DVFS techniques to reduce the maximum temperature and improve performance. Donald *et al.* [13] exploited the distributed nature of multicore processors and classify techniques for multicore thermal management. Ge *et al.* [14] presented a distributed thermal balancing policy to stabilize the operating temperature and improve the performance of many-core systems by distributing task migration with a lightweight agent using decision based on steady-state temperature—not transient or real-time temperature. In [15], heat-and-run performs SMT thread assignment and CMP thread migration to maintain throughput under thermal constraints. Chantem *et al.* [16] presented an integer linear program solver to schedule tasks to cores to reduce overall peak temperature. Heo *et al.* [18] presented activity migration techniques for power density reduction. Ghiasi *et al.* [19] presented migration techniques between asymmetrical dual cores. Coskun *et al.* [1] combined thread migration with scheduling policies to reduce the hot spot in multiprocessing silicon-on-chips (SoCs). They have also proposed online learning [17] and integer linear programming to minimize the frequency of thermal hot spots and gradients [20]. Coskun *et al.* [23] proposed proactive temperature management to estimate future temperature and allocate workload on multicore systems. Yeo *et al.* [21] presented predictive dynamic thermal management with an advanced future temperature prediction model for multicore systems. Chen *et al.* [22] presented proactive process speed scheduling to guarantee real-time deadlines under thermal constraints. Michaud *et al.* [12] explained the need for processor design based on transient temperature as well as steady-state temperature.

The existing works, while providing a variety of techniques for architectural implementations of power migrations, lack a detailed analysis of the effect of the migration interval—a key control parameter—on the effectiveness of spatiotemporal power migration. The analysis of migration interval is critical for proactive/predictive thermal management where the goal is to minimize temperature and control the thermal field instead of only satisfying a peak temperature constraint. This paper contributes to the study of the effect of migration interval on both peak temperature as well as spatial/temporal nonuniformity of the thermal field. The proposed scheme redistributes the active resources available for computation in space and time by using a core-level active–inactive control. The total power delivered to the chip for a given computation load is multiplexed spatiotemporally between different computational resources. The multiplexing is controlled in hardware by turning a core "on" or "off" using clock and supply gating, which (may) force computation migration. This allows us to realize a much smaller migration interval compared to software/OS-controlled spatiotemporal migration [12]. Further, we consider a proactive migration approach to minimize the average chip temperature under all workload conditions thereby improving average performance, leakage, lifetime reliability, and cooling efficiency. We connect the temperature effect to leakage and delay analysis to study the impact of the proposed technique on the system power/performance. The behavior of the thermal system is characterized in the frequency domain to explain the effect of the migration interval and to provide guidance for optimally choosing the migration interval. To facilitate the above optimization, we characterize the effect of the

migration interval on the timing overhead of migration by considering cycle-level architectural simulations and power overhead. Finally, we study the effect of power migration with varying migration intervals considering different migration policies and their impact on the thermal field and their architectural consequences. In summary, this paper provides a comprehensive analysis of spatiotemporal power migration considering the effect of the migration interval.

III. Spatiotemporal Power Multiplexing

The basic thermal management policy considered in this paper is to change the location of the power dissipation (i.e., heat generation) at a fixed time interval referred to as the migration interval or time slice [39]. This is achieved by changing the locations of the active cores after each time slice. The required system throughput (i.e., number of active cores) is maintained during the redistribution. For a given amount of heat generation (i.e., number of active cores) and heat outflow (i.e., cooling method and energy), power migrations continuously redistribute the generated heat in space and time to reduce the maximum temperature and improve thermal uniformity. We consider a coordinated migration of all the active cores at each time slice irrespective of their current temperature instead of migrating individual cores at different time instances depending on their local temperature. This reduces the maximum chip temperature and spatiotemporal nonuniformity under all workload conditions. The multiplexing is controlled in hardware by turning a core on or off using clock and supply gating, as demonstrated in recent many-core chips. The deactivated cores employ clock and supply gating and dissipate negligible power. The recent developments of many-core systems show the feasibility of fine-grain active–inactive control of cores [24]–[29]. The active–inactive core control or core gating can be thought of as extreme case of DVFS where each core has two possible states: nominal voltage/nominal frequency and zero (low) voltage/zero frequency. In principle, power migration can also be applied along with fine-grain DVFS or DFS to both reduce the generated heat (DVFS or DFS) and redistribute the generated heat (power migration). However, we concentrate on core gating-based control because fine-grain DVFS can become less effective in deep nanometer technologies. This is illustrated in Fig. 2, considering circuit simulations of delay-voltage characteristics of an eight-stage FO4 ring oscillator at different predictive technology nodes [30], [31]. For a target frequency reduction, the opportunity for voltage reduction reduces at scaled nodes. For example, the allowable voltage reduction for a 20% frequency reduction target at a 16-nm node is only 5% or ∼40 mV [considering International Technology Roadmap for Semiconductors (ITRS) predictions of ∼0.7 V supply voltage]. Though dynamic frequency scaling alone does not have the same limitations of voltage scaling, it has energy challenges. Reducing frequency reduces only dynamic power, but not dynamic energy. Moreover, as the same operation is performed over a longer time period (as clock cycles are longer), the leakage energy increases. Therefore, the overall energy dissipation may even increase if the clock cycle is simply increased. This is particularly challenging in deep nanometer nodes where leakage contributes to a significant portion of the total energy. Hence, recent high-performance microprocessors have stressed the need for aggressive power gating to reduce leakage [24]–[29]. The active–inactive control needs to be coupled with approaches like core-hopping and thread migration to migrate computation from a deactivated to an activated core.

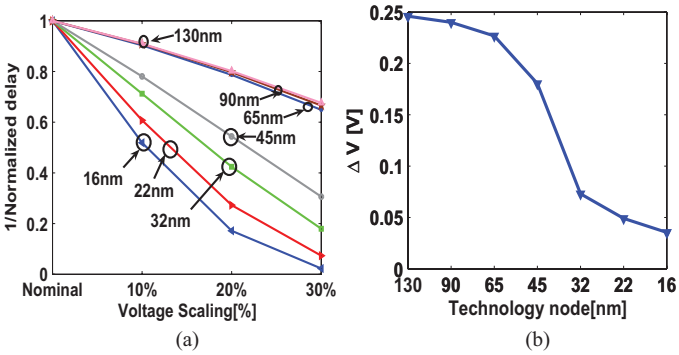
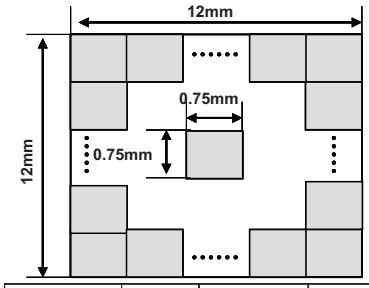


Fig. 2. Limitation of fine control for DVS. (a) Voltage versus frequency. (b) Voltage change for 20% frequency reduction.

IV. Modeling and Simulation Environment

Fig. 3 summarizes the modeling/simulation methodology for generation of the time-varying thermal map using the core-migration-based execution and physical power/heat flow models. The simulations are performed considering a tile-type homogeneous 256 core processor in ITRS predictive 16-nm technology (Fig. 3). Our system model considers each core as having 16 million gates and a local cache, and running at 3 GHz frequency. We have used the technology-driven power estimation tool (Intsim) to estimate the power of each core at 16 nm [32], [33]. The total power is computed by considering the dynamic and leakage power of the logic gates and power dissipation in wires, clock network, registers, and repeaters in the signal/clock interconnects [32], [33]. The cache power is neglected for simplicity. The device dimensions (channel length, oxide thickness, etc.), properties (e.g., on current, off current, threshold voltage, interconnect dielectric constant, etc.), and supply voltage was obtained from ITRS High-Frequency Device Roadmap [31]. The width of the logic gates are optimized to meet the target frequency considering a critical path of 10 NAND2 gates. Our simulation predicts ∼1.5 W power per core, which is reasonable if we extrapolate from each core of 45-nm microprocessors running at 3 GHz (consumes ∼80–100 W) to 16-nm nodes [34]. Assuming a 2× reduction in switching capacitance in successive generations and ∼30% reduction of VDD as predicted by ITRS from 45- to 16-nm node, we obtain ∼5–6 W of power per core. Hence, 1.5 W is an optimistic estimate accounting for 3–4× reduction in complexity. We first verify the effectiveness of power multiplexing with ∼1.5 W core power and later show that the power multiplexing method is more effective for higher core power. The non-uniform power map of the chip is coupled to a full-chip thermal simulator, Hotspot [35], [36] to estimate



	Side (mm)	Thickness (mm)	Thermal Resistivity (m-K/W)	Sp. Heat. Capacity (J/m³K)
Silicon	12	0.2	0.01	1.75×10^6
TIM	12	0.02	0.25	4e6
Heat Spreader	30	1	0.0025	3.55e6
Heat Sink	60	6.9	conv. res. (K/W) = 0.1	

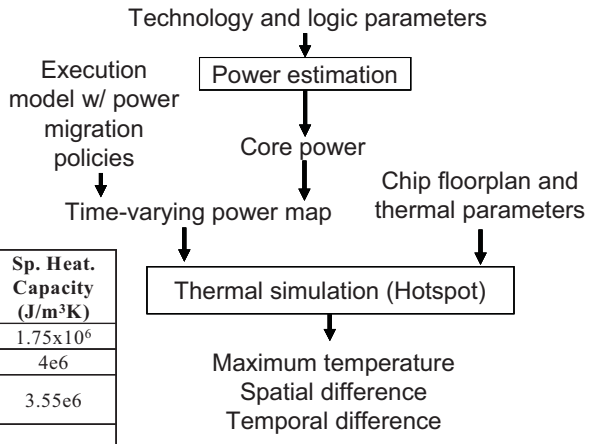


Fig. 3. Many-core chip, and modeling and simulation environment.

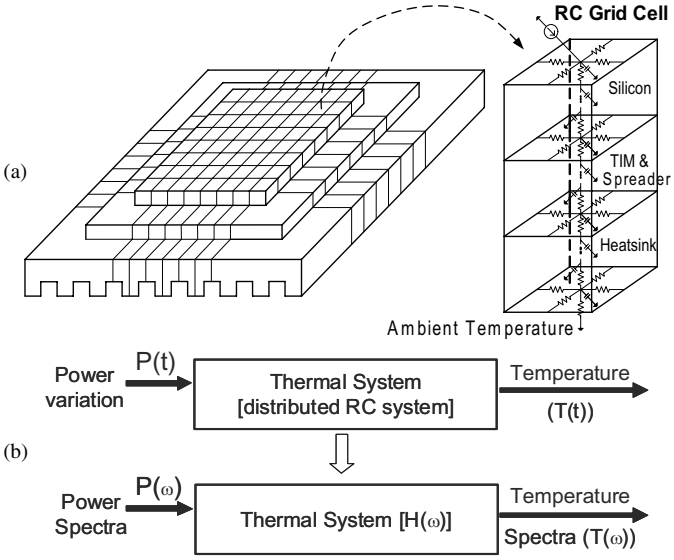


Fig. 4. (a) Distributed RC-based model of a chip-package system for transient thermal analysis. (b) Proposed equivalent frequency domain thermal model.

the thermal field. Hotspot performs thermal simulation using 3-D distributed RC-based models for silicon, thermal interface material (TIM), heat spreader, and heat sink. Although this approach disregards the exact architecture details of the core, it provides insights into the thermal behavior of many-core chips.

V. Thermal Impact of The Migration Interval

We study the effect of the time-slice interval considering varying computational demand and, hence, on core ratio (defined as ratio of the number of on-cores to the total number off-cores). The 25% to 75% variation in the on-core ratio is considered in the simulation. The random migration policy is used for the experiments. In this condition, at the end of each migration interval, the current set of active cores is deactivated and a new random set of active cores is chosen for the next time interval. We compute the following parameters to characterize the thermal field: 1) the maximum chip temperature, i.e., the maximum of the temperature of all the cores at a given time; 2) the spatial difference, i.e., the difference between maximum and minimum on-chip temperature at a given time; and 3) the temporal difference, i.e., the difference between the maximum and minimum temperature for a core over a long time period ($\sim$10 ms) and average of that difference over all cores in the chip.

A. Frequency-Domain Characterization of Thermal System

To understand the implications of migration interval, we have characterized the behavior of the thermal system of a many-core processor in the frequency domain (Fig. 4). This is performed using a small-signal analysis, i.e., by applying sinusoidal power waveforms of different spectral frequencies at one core and measuring the temperature spectra for different cores. This ratio of the spectral response of temperature and power (sinusoidal waveform) is used to compute the frequency-domain characteristics of the thermal system. Fig. 5 shows the gain of the frequency response considering power variations in core D0 and temperature variations in core D0, D1, and D2. Note that cores D1 and D2 represent cores at different distances from the source cores. From both the figures, we observe that the thermal system behaves as a low-pass filter and the gain of the frequency response reduces significantly at higher frequencies. In other words, for the same average power, a higher frequency of power fluctuations will result in a lower increase in temperature. The exact gain depends on the location of the cores, and cores near the edge have a lower gain (i.e., lower temperature) because there is higher cooling efficiency. Moreover, we make another important observation, i.e., the power dissipation of a source core has minimal effect on the temperature of cores even at a reasonably short distance from the source core. This suggests that, if a core is active, the neighboring inactive cores may not have significant rise in the temperature. The effect is further lower if the power at the source core has a

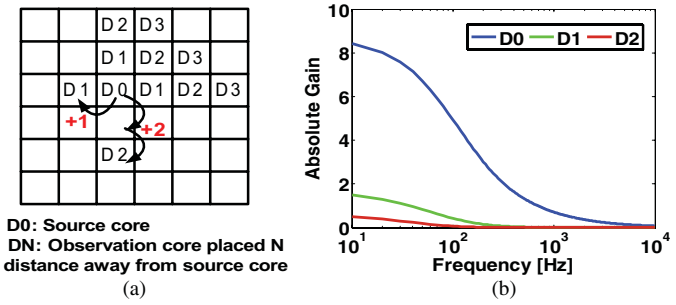


Fig. 5. Filter behavior of thermal system. (a) Distance between source core and observations cores. (b) Observation node.

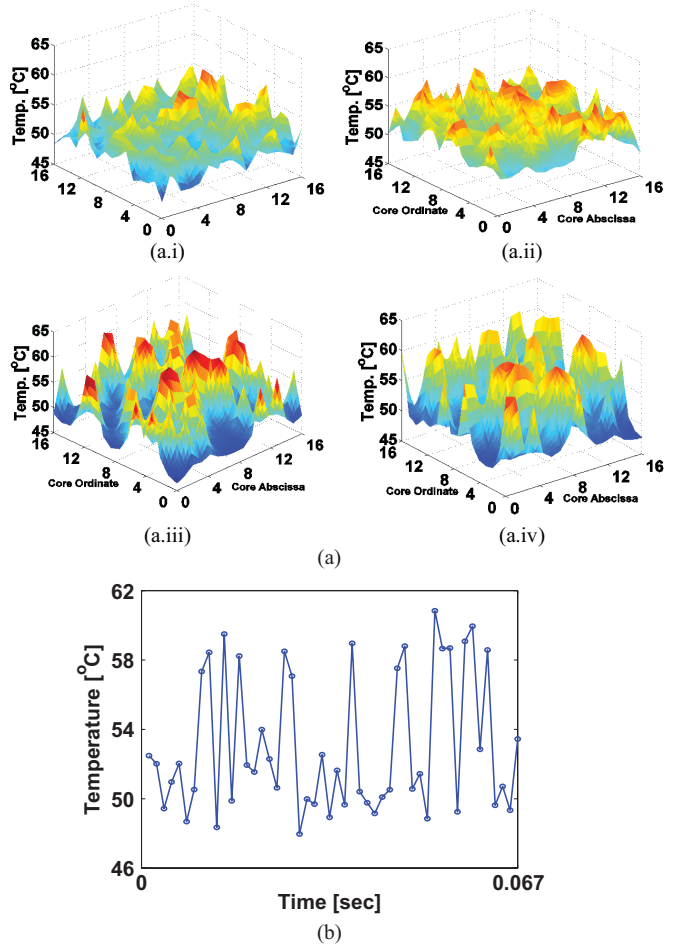


Fig. 6. Thermal behavior of many-core system's nonuniform thermal characteristic with power multiplexing. (a) Spatial maps. (a.i) At $T = 0.165$ s (100 K). (a.ii) At $T = 0.33$ s (100 K). (a.iii) At $T = 0.165$ s (1000 K). (a.iv) At $T = 0.33$ s (1000 K). (b) Temporal variation of the temperature of a core.

higher frequency fluctuation. Note that activation–inactivation of cores using power migration induces fluctuation in the core power. This will result in less increase in the core temperature. We further predict that, because faster migration increases the frequency of power fluctuation of individual cores, it will reduce the increase in temperature as well. Further, the faster power fluctuation of a core will also cause a reduced temperature swing between the on and off period of the core, resulting in a lower temporal nonuniformity in temperature. As the temporal swings are reduced, at any time instant the

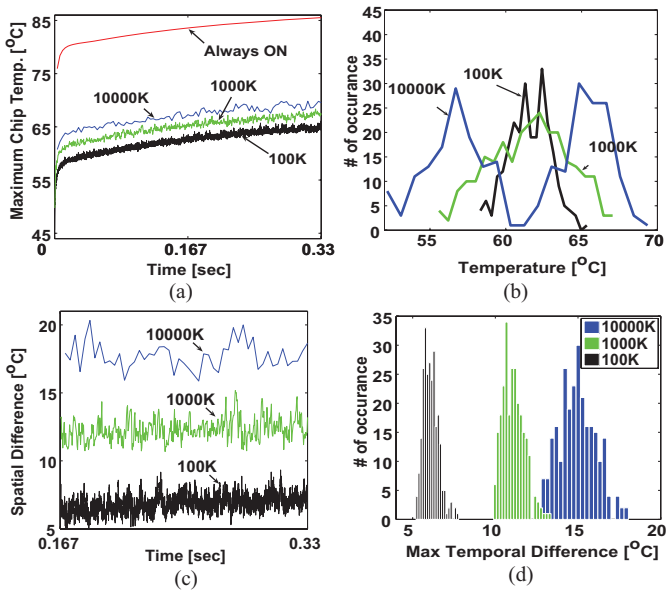


Fig. 7. Effect of time slice on random migration. (a) Max temperature. (b) Core-to-core temperature distribution. (c) Spatial difference. (d) Temporal difference. 100 K refers to 100 000 clock cycles, i.e., 33 μs for a 3-GHz clock.

temperature difference between the active and inactive cores will also be lower. In other words, we also expect to have a lower spatial nonuniformity in the thermal field when the migration interval is decreased.

B. Time-Domain Analysis of the Migration Interval

For a given number of active cores, a larger time slice leads to a higher maximum temperature and higher spatiotemporal nonuniformity. We used the simulation framework and environment to estimate the temperature, as shown in Fig. 3. The random migration policy with the on-core ratio of 50% is considered in Figs. 6 and 7. This is visible in the thermal map at different time instances considering time slice intervals of 100 000 (100 K) and 1 000 000 (1000 K) clock cycles [Fig. 6(a)]. Fig. 6(b) shows the variations in the temperature of a core over time. We can observe the temporal thermal cycles in the core. A smaller time slice shows reduced maximum temperature [Fig. 7(a)]. A faster migration implies that core-to-core temperature variation is much less [Fig. 7(b)]. The spatial difference reduces with a faster time slice [Fig. 7(c)]. This implies that on-chip thermal variations are more uniform at all time instants with a faster migration. Both the average value and core-to-core variations of the temporal difference over all cores reduce significantly with a faster time slice [Fig. 7(d)]. Lower and similar thermal cycles for all cores imply lower and similar reliability degradation for all cores.

The reduction in the maximum temperature, spatial difference, and temporal difference with a lower time slice is effective for different numbers of active cores [Fig. 8(a)–(c)]. However, we observe that a smaller time slice is required for larger on-core ratio to maintain a target maximum temperature, or spatial or temporal difference [Fig. 8(d)–(f)]. The design of the time slice will depend on the thermal properties of the chip, particularly on the lateral resistance (for spatial redistribution) and on the thermal capacity

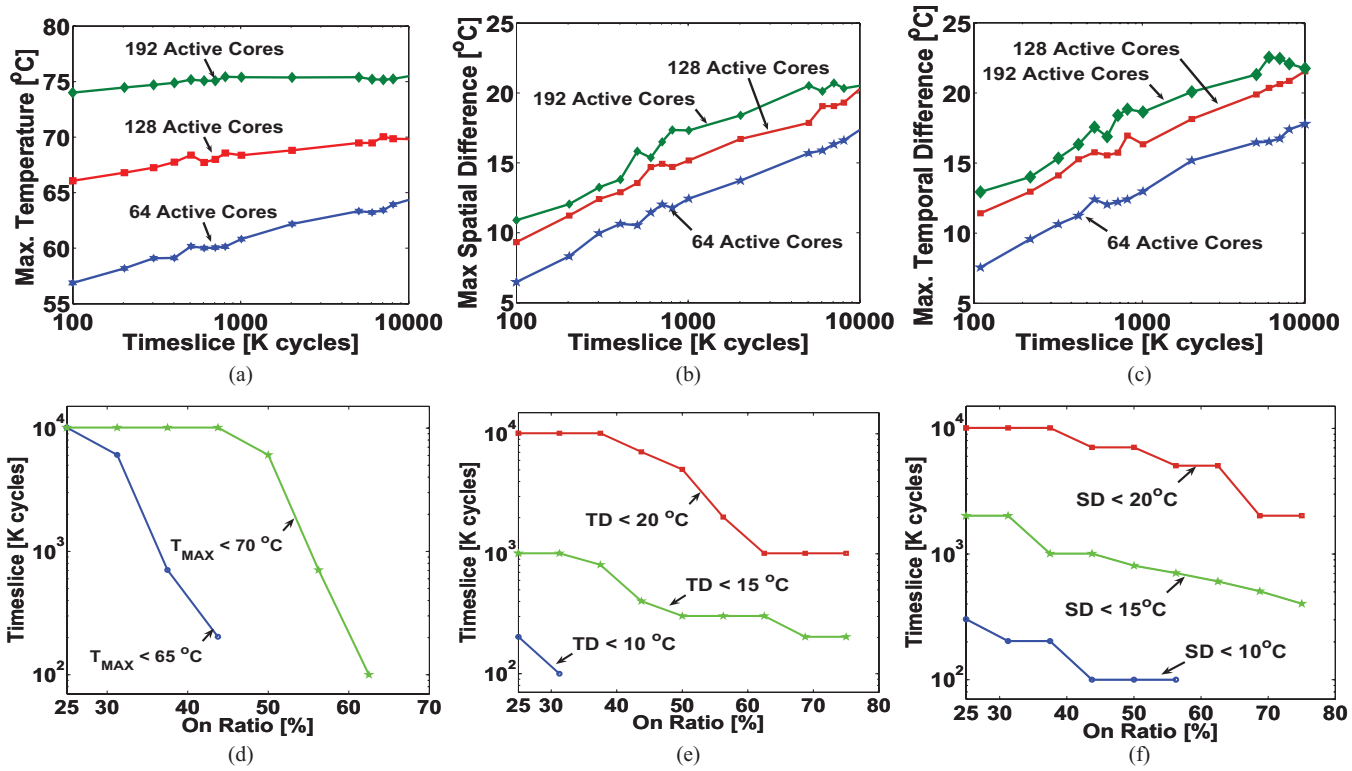


Fig. 8. Effect of time slice on (a) max temperature, (b) max spatial difference, and (c) max TD. Time slice for (d) target max temperature on ratio, (e) target spatial difference in different on ratios, and (f) target temporal difference in different on ratios.

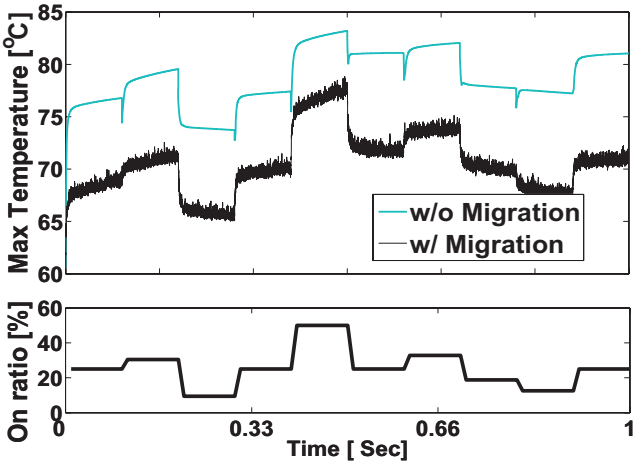


Fig. 9. Chip temperature with run-time variations in active core number.

(for temporal redistribution) of silicon. The time slice can be adapted depending on the computational load or real-time temperature data. Such an adaptive time slice will be addressed in a future paper. Fig. 9 shows the effect of power multiplexing considering run-time variations in the performance demand (i.e., number of active cores). It can be observed that power multiplexing can reduce peak temperature significantly under all demand (i.e., the no. of on-cores) but the effect is more pronounced at low to moderate demand conditions. This suggests that power multiplexing will help in reducing the average temperature of the chip over its lifetime, thereby reducing cooling energy as well as improving reliability.

VI. Electrical Impact of Migration Interval and Analysis of Tradeoffs

Although faster power multiplexing provides a lower maximum temperature and a uniform thermal map, it also requires migration of computation threads, which increases performance overhead. Therefore, the analysis of the tradeoff between the thermal behavior and system performance is critical for the accurate evaluation of power multiplexing.

A. Effect on Core-to-Core Delay and Leakage Variations

The delay and leakage variations are estimated considering an eight-stage FO4 ring oscillator using 16-nm predictive models [30]. Figs. 10(a) and (b) show the histogram of delay and leakage increase (estimated for an inverter) relative to those at 45 °C. We expect faster spatiotemporal power multiplexing to reduce the core-to-core delay and leakage variations. Note that, near the end of a time slice interval, the inactive cores can have much lower temperature for larger migration interval than the inactive cores for faster migration as a larger migration interval provides a longer time for the temperature to decay. Consequently, we observe that, at a higher migration interval, there exist cores with a lower delay and leakage compared to the cores with faster migration. However, the slowest active core determines the operating frequency of the chip and we observe that the maximum core delay is higher for slower migration (higher temperature of active cores). Fig. 10(c) shows the variation in the maximum delay (i.e., delay of the slowest on-core) considering on-chip thermal variation over time. As illustrated in Fig. 10, the overall chip

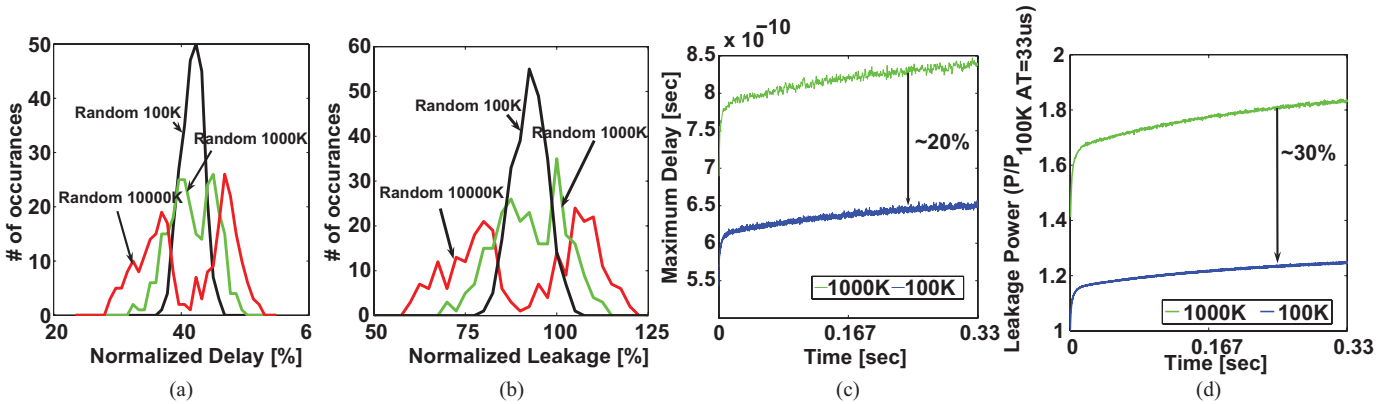


Fig. 10. Effect of power multiplexing on (a) core-to-core delay distribution, (b) core-to-core leakage distribution, (c) chip delay (i.e., maximum core delay), and (d) chip leakage power.

performance for a target on-core ratio improves with faster migration. Fig. 10(d) shows the normalized leakage power for the on-cores over time. As expected, total chip leakage reduces with a faster time slice.

B. Power Multiplexing and System Performance

Power multiplexing has two primary impacts on the execution performance. First, there exists architectural overhead associated with migration of computations. The net effect is a longer execution time for a given thread when migration is performed. If we consider an average number of cycles for each such migration, faster migration will lead to higher number cycles to execute. The second effect of migration on execution performance is a positive one in that a faster migration reduces the temperature, which can help increase the maximum clock frequency. A higher clock frequency improves the overall performance for a constant number of execution cycles. Therefore, we combine the two effects to study the effect of power multiplexing on overall system performance. We define effective performance improvement (PI) for a given migration interval ($T_{\text{migration}}$) as

$$PI\left(@T_{\text{migration}}\right) = \underbrace{\left(\frac{N_{\text{no\_migration}}}{N_{\text{migration}}}\right)}_{N_{\text{norm}}} \times \underbrace{\frac{f_{\text{migration}}}{f_{\text{no\_migration}}}}_{f_{\text{norm}}} \quad (1)$$

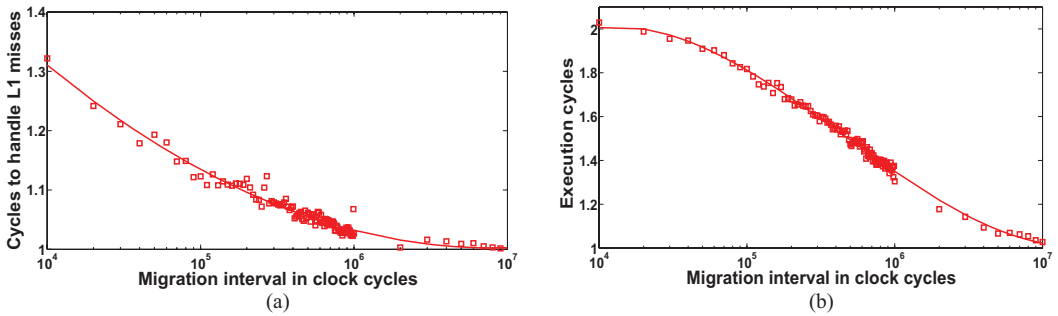
where $f_{\text{no\_migration}}$ is the clock frequency when no migration is performed and $f_{\text{migration}}$ is the higher clock frequency due to lower and more uniform temperature. Similarly, $N_{\text{no\_migration}}$ and $N_{\text{migration}}$ are the number of execution cycles for a given workload without and with migration, respectively. We denote the normalized operating frequency as f_{norm} and the normalized number of execution cycles as N_{norm}. We expect $f_{\text{norm}} > 1$ and increases with faster migration, while $N_{\text{norm}} < 1$ and decreases with faster migration. Note that $PI > 1$ and $PI < 1$ imply better and worse performance compared to no migration, respectively.

C. Estimation of Migration Overhead on Execution Time

For our evaluation of the performance impact of thread migration, we divided the overhead into two components: an application-independent migration delay introduced by the OS and runtime, and an application-dependent overhead that is primarily a consequence of additional cache misses. To estimate the application-independent migration delay introduced by the OS, we performed experimental measurements using current-generation multicore processors. The application-dependent overhead due to cache misses was performed using a many-core simulator. Our measurement of the runtime migration delay assumes that the application: 1) spawns as many threads as there are CPUs; 2) is the only one running on the machine; and 3) does not significantly impact the behavior of the OS and threading library.

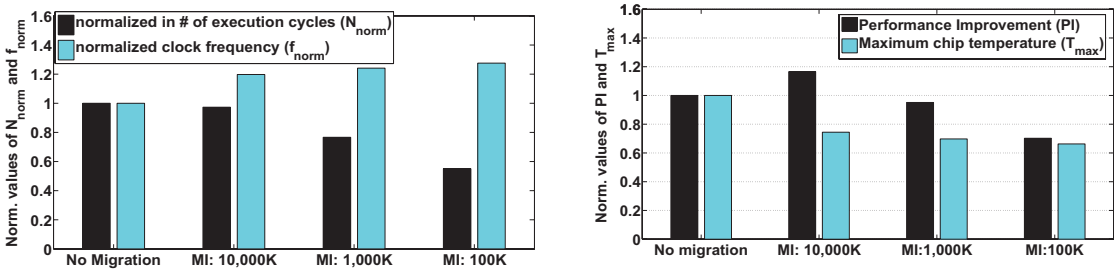
1) Application-Independent Migration Delay Measurement: Our measurement of the application-independent delay was performed on a four-core 2.66-GHz Nehalem desktop. Our microbenchmark spawns one thread per hardware context. Each of these threads reads the system clock, migrates to a new hardware context N times, and reads the system clock again. The total migration time measured in this way is then used to compute a mean migration time. Two versions of this benchmark were created: one using native Linux threads, and another using the QThreads [37] cooperative multithreading library. For comparability, the evaluation on an Intel Westmere platform was run on only one of its sockets. Because of a large number of outliers caused by interference of system processes, the program was run multiple times. The values reported here are the medians. On the quad-core single-threaded Nehalem processor, an OS thread took 9.6 μs to migrate, whereas a QThread took 2.5 μs. On the six-core Westmere with two threads per core, the cost of migrating an OS thread increased to 44.3 μs, while the cost of migrating a QThread dropped to 1.6 μs. Thus we see a sharp reduction from using stock OS mechanisms to a lighter weight explicitly software-managed migration.

2) Application-Dependent Migration Delay Simulation: Evaluation of application-dependent delay due to cache misses was performed on a simulator, so that effects of the operating system could be completely abstracted away. Our experiments made use of the QSim emulator and the Structural Simulation Toolkit [38]. The simulated processor was a 4 × 4 mesh with 8 megabytes of distributed shared L2 and per-core coherent L1 instruction and data caches, 64 kB each. The application running on the processor was a multiprogrammed



(a) (b)

Fig. 11. Evaluation of migration interval. (a) Effect of migration on increased cache misses. (b) Effect of migration on overall execution time. The y-axis values are normalized with respect to the values for no migration case. The data points are the values measured from simulations, and the solid line is the trend line.



(a) (b)

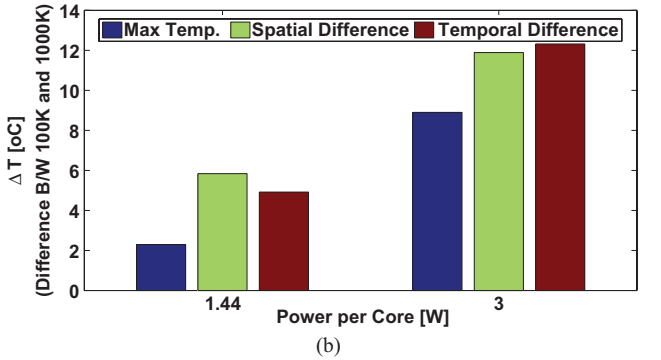
Fig. 12. Tradeoff between migration interval and system performance. (a) Migration interval, normalized execution cycle, and operating frequency. (b) Migration interval and performance improvement considering the combined effect.

workload of diverse serial microbenchmarks; merge sort, bubble sort, sieve, and Fisher–Yates shuffle. For this workload and simulated architecture, the L1 cache's miss rate increased by 32% for a migration every 10 μs, 12% for a migration every 100 μs, and 3% for a migration every 1 ms. Note, for a 1-GHz clock, 10 100, and 1000 μs translate to 10 000, 100 000, and 1 000 000 clock cycles. Fig. 11(a) illustrates the increase in the number of cycles due to L1 misses as the migration frequency is decreased. Note that this is the sum of additional L1 miss cycles across all applications due to migration, normalized to the sum of L1 miss cycles across all applications without migration.

3) Effect on Overall Execution Cycles Due to Migration: The exact impact on execution cycles will depend on the specific target machine and time to access a local versus remote L2 cache bank. In our simulations, an L1 miss incurs a 5-cycle L2 access penalty. Note that, due to the shared nature of the L2 cache, the L2 state of the migrated thread does not change. However, if the reference is to a nonlocal L2 bank, then the miss incurs latency in the network. L1 misses will travel a variable number of network hops depending on where the L2 data is located and how the interconnection network is designed. We have assumed a 5 cycle/hop latency for each miss and an average number of hops (4 in this case). These values correspond to conservative nominal values found in modern systems. Hence the L1 miss penalty will be expected to be less. The penalties due to L1 misses should be added to the Qthreads software cost to obtain the effect on overall migration overhead. Fig. 11(b) shows the impact on overall execution time considering different migration frequencies. Since we have a multiprogrammed workload, this is based on

Timeslice Interval	100K	200K	500K	1000K
Power overhead	0.028	0.014	0.006	0.003

(a)



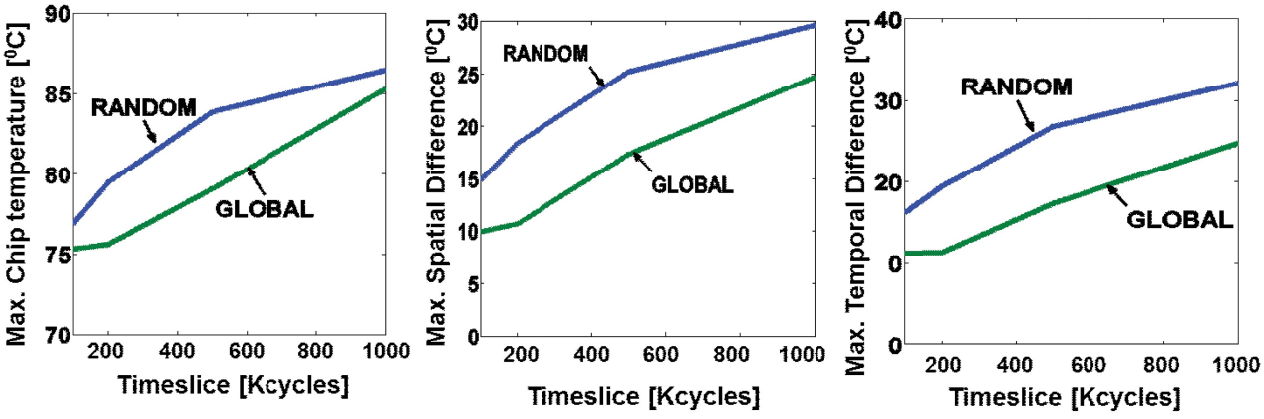
(b)

Fig. 13. Power analysis. (a) Power overhead of migration. (b) Impact of higher power density.

the termination of the last application in the set (rather than the average execution time increase). Therefore the results should be viewed as conservative; i.e., on a per-application basis they are more often likely to be lower. The thread migration overheads presented in this paper are conservative, as the coherence protocol does not employ advanced optimizations such as forwarding to reduce latency of coherence misses.

D. Tradeoff Between Migration Interval and Performance

We now study the effect of migration interval on overall system performance considering both architectural overhead



(a) (b) (c)

Fig. 14. Effect of policies on (a) max temperature, (b) spatial difference, and (c) temporal difference.

and clock frequency improvement. Fig. 12(a) shows that faster migration increases normalized frequency due to lower temperature and, hence, a lower delay. On the other hand, from Figs. 11(b) and 12(a) we observe that normalized number of cycles increases with faster migration. The combined effect is shown in Fig. 12(b). As the migration interval increases from no migration to 1 000 000, we observe an increase in overall performance due to higher clock frequency. However, migrating very fast (∼100 000) starts degrading the overall performance mainly due to unsustainable increase in number of execution cycles. Moreover, the clock frequency increases (i.e., temperature reduces) at a much faster rate between no migration to 10 000 000 migration cycle but migrating faster provides diminishing returns. The optimal choice of the migration interval depends on the thermal properties of the chip and package and architectural properties (i.e., operating system, cache miss overhead, design of the network on the chip, etc.). Note that the preceding analysis was performed using stock operating systems and software that was not designed for systems based on thread migration. When designing anticipating thread migration, many alternatives are possible in mitigating the migration overhead.

E. Power Overhead and Overall Power Saving

Spatiotemporal power multiplexing will require additional transition energy for active–inactive control. We assume that, during transition, all internal nodes make transition (i.e., entire core capacitance switches). The switched capacitance is estimated from the power per core. The transition of the supply network capacitance (estimated based on ITRS roadmap) and 10% additional core-level decoupling capacitance is considered. Additional power due to migration of the flip-flop states is also considered. As power migration is performed faster, power overhead increases [Fig. 13(a)]. However, as the per core switch capacitance is reduced at deep nanometer nodes and migration is performed at relatively low frequency (100 000–1 000 000 times lower than the clock frequency), the power overhead is very low.

We analyze the impact of variation in the active power of a core and non-zero power in the deactivated mode (finite leakage in case of state-preserving supply gating, i.e., voltage is not reduced to "0"). Our analysis shows that these conditions will increase the maximum on-chip temperature, requiring a smaller time slice interval. Also, we study the effect of a varying power density for each core. Since higher power density implies higher heat generation in each active core, a smaller time slice is required for a target temperature. The relative effect of reducing the time slice is more prominent at a higher power per core (Fig. 13).

F. Spatiotemporal Power Multiplexing Policies

Our earlier results assumed a random migration policy to illustrate the impact of migration interval. For a fixed time slice, the effectiveness of the spatiotemporal power multiplexing can be enhanced by properly optimizing the policy for choosing the next set of active cores. To illustrate this, we discuss a partially reactive control approach, referred to as the "global coolest replace," using on-chip real-time temperature data. The objective is to replace the hottest N cores with the coolest N cores. This can help decrease the maximum temperature of the chip, as the cooler cores in the chip are turned on at the end of each time slice. We studied this effect considering 64 active cores and varying the time slice interval. We have considered the higher power density case in these simulations. Our analysis shows that, for a fixed time slice, the global coolest replace results in a much lower maximum temperature, spatial difference, and temporal difference (Fig. 14). From Fig. 5(a), we have observed that the core-to-core coupling reduces rapidly with the number of hops (i.e., the distance between source core and observation core). Hence, we expect that the temperature of an inactive core in the neighborhood of an active core will be close to an inactive core far from the active core, which has been verified by characterizing the global coolest policy. The above analysis points to the fact that it is possible to design a migration policy where the migrations are always restricted to within a neighborhood to reduce the migration cost.

VII. Conclusion

We presented spatiotemporal power multiplexing as an execution principle for many-core microprocessors for managing the thermal field. The proposed method exploits the lateral

heat flow and thermal capacity of materials to redistribute the generated heat in a many-core chip by varying the location of the active cores over time. A faster migration, i.e., a smaller migration interval, can result in lower maximum temperature and better spatiotemporal uniformity of temperature. We analyzed the effects of spatiotemporal uniformity on leakage, operating frequency, and cooling efficiency of many-core chips and showed that, even after considering the energy overhead of core transitions, the proposed method can provide better energy efficiency. We observed that the thermal considerations can force us to rethink compiler and microarchitecture optimizations including the design of the cache hierarchy and consequently merit attention beyond what can be included in this paper. The proposed execution principle for a computer system involving continuous migration of the active on-chip computing resources can help to improve the energy efficiency and reliability and reduce the cooling energy demand for high-performance many-core processors. The concurrent design of architecture and thermal management principles are required to maximize the benefit of power multiplexing while minimizing their performance implications.

ACKNOWLEDGMENT

The authors would like to thank Dr. R. Rao, Dr. P. Bose, and Dr. E. Kursun, IBM T. J. Watson Research Center, Yorktown Heights, NY, and Dr. A. Raychowdhury, Intel Corporation, Santa Clara, CA, for many helpful discussions.

REFERENCES

[1] A. Coskun, T. T. Rosing, K. A. Whisnant, and K. C. Gross, "Static and dynamic temperature-aware scheduling for multiprocessor SoCs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 16, no. 9, pp. 1127–1140, Sep. 2008.

[2] D. Brooks and M. Martonosi, "Dynamic thermal management for high-performance microprocessors," in *Proc. 7th Int. Symp. High-Perform. Comput. Arch.*, 2002, pp. 171–182.

[3] P. Chaparro, J. Gonzalez, G. Magklis, C. Qiong, and A. Gonzalez, "Understanding the thermal implications of multi-core architectures," *IEEE Trans. Parallel Distrib. Syst.*, vol. 18, no. 8, pp. 1055–1065, Aug. 2007.

[4] R. McGowen, C. A. Poirier, C. Bostak, J. Ignowski, M. Millican, W. H. Parks, and S. Naffziger, "Power and temperature control on a 90-nm Itanium family processor," *IEEE J. Solid-State Circuits*, vol. 41, no. 1, pp. 29–237, Jan. 2006.

[5] J. Tschanz, N. S. Kim, S. Dighe, J. Howard, G. Ruhl, S. Vanga, S. Narendra, Y. Hoskote, H. Wilson, C. Lam, M. Shuman, C. Tokunaga, D. Somasekhar, S. Tang, D. Finan, T. Karnik, N. Borkar, N. Kurd, and V. De, "Adaptive frequency and biasing techniques for tolerance to dynamic temperature-voltage variations and aging," in *Proc. Int. Solid-State Circuits Conf.*, 2007, pp. 292–604.

[6] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers, "The case for lifetime reliability-aware microprocessors," in *Proc. 31st Annu. Int. Symp. Comput. Arch.*, Jun. 2004, pp. 276–287.

[7] K. Sankaranarayanan, S. Velusamy, M. Stan, and K. Skadron, "A case for thermal-aware floorplanning at the microarchitectural level," *J. Instruct.-Level Parall.*, vol. 8, pp. 1–16, Oct. 2005.

[8] R. Rao, S. Vrudhula, and C. Chakrabarti, "Throughput of multi-core processors under thermal constraints," in *Proc. Int. Symp. Low Power Electron. Design*, 2007, pp. 201–206.

[9] E. Kursun and C.-Y. Cher, "Temperature variation characterization and thermal management of multicore architectures," *IEEE Micro*, vol. 29, no. 1, pp. 116–126, Jan.–Feb. 2009.

[10] V. Gektin, R. Zhang, M. Vogel, G. Xu, and M. Lee, "Substantiation of numerical analysis methodology for CPU package with non-uniform heat dissipation and heat sink with simplified fin modeling," in *Proc. 9th Conf. Thermal Thermomech. Phenomena Electron. Syst.*, 2004, pp. 537–542.

[11] H. Esmaelizadeh, E. Blem, R. Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *Proc. 38th Annu. Int. Symp. Comput. Arch.*, Jun. 2011, pp. 365–376.

[12] P. Michaud and Y. Sazeides, "Scheduling issues on thermally constrained processors," INRIA, Paris, France, Tech. Rep. RR-6006, Oct. 2006.

[13] J. Donald and M. Martonosi, "Techniques for multicore thermal management: Classification and new exploration," in *Proc. 33rd Int. Symp. Comput. Arch.*, 2006, pp. 78–88.

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

[15] M. Powell, "Heat-and-run: Leveraging SMT and CMP to manage power density through the operating system," in *Proc. 11th Int. Conf. Arch. Support Program. Lang. Operat. Syst.*, 2004, pp. 260–270.

[16] T. Chantem, R. P. Dick, and X. S. Hu, "Temperature-aware scheduling and assignment for hard real-time applications on MPSoCs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 10, pp. 1884–1897, Oct. 2011.

[17] A. Coskun, T. S. Rosing, K. A. Whisnant, and K. C. Gross, "Temperature-aware MPSoC scheduling for reducing hot spots and gradients," in *Proc. Asia South Pacific Design Autom. Conf.*, 2008, pp. 49–54.

[18] S. Heo, K. Barr, and K. Asanovic, "Reducing power density through activity migration," in *Proc. Int. Symp. Low Power Electron. Design*, 2003, pp. 217–222.

[19] S. Ghiasi and D. Grunwald, "Thermal management with asymmetric dual-core designs," Dept. Comput. Sci., Univ. Colorado, Boulder, Tech. Rep. CU-CS-965-03, 2004.

[20] A. Coskun, T. S. Rosing, and K. C. Gross, "Proactive temperature balancing for low cost thermal management in MPSoCs," in *Proc. Int. Conf. Comput.-Aided Design*, 2008, pp. 250–257.

[21] I. Yeo, C. C. Liu, and E. J. Kim, "Predictive dynamic thermal management for multicore systems," in *Proc. Design Autom. Conf.*, 2008, pp. 734–739.

[22] J. Chen, S. Wang, and L. Thiele, "Proactive speed scheduling for real-time tasks under thermal constraints," in *Proc. IEEE Real-Time Embedded Technol. Appl. Symp.*, Apr. 2009, pp. 141–150.

[23] A. Coskun, T. S. Rosing, and K. C. Gross, "Temperature management in multiprocessor SoCs using online learning," in *Proc. Design Autom. Conf.*, 2008, pp. 890–893.

[24] Z. Hu, A. Buyuktosunoglu, V. Srinivasan, V. Zyuban, H. Jacobson, and P. Bose, "Microarchitectural techniques for power gating of execution units," in *Proc. Int. Symp. Low Power Electron. Design*, 2004, pp. 32–37.

[25] J. Clabes, J. Friedrich, M. Sweet, J. DiLullo, S. Chu, D. Plass, J. Dawson, P. Muench, L. Powell, M. Floyd, B. Sinharoy, M. Lee, M. Goulet, J. Wagoner, N. Schwartz, S. Runyon, G. Gorman, P. Restle, R. Kalla, J. McGill, and S. Dodson, "Design and implementation of the POWER_5 TM microprocessor," in *Proc. IEEE Int. Solid-State Circuit Conf.*, Nov. 2004, pp. 1–8.

[26] N. A. Kurd, S. Bhamidipati, C. Mozak, J. L. Miller, T. M. Wilson, M. Nemani, and M. Chowdhury, "Westmere: A family of 32 nm IA processors," in *Proc. IEEE Int. Solid-State Circuit Conf.*, Mar. 2010, pp. 96–97.

[27] S. Sawant, "A 32 nm Westmere-EX Xeon enterprise processor," in *Proc. IEEE Int. Solid-State Circuit Conf.*, Feb. 2011, pp. 74–75.

[28] S. Bell, B. Edwards, J. Amann, R. Conlin, K. Joyce, V. Leung, J. MacKay, M. Reif, J. L. Bao, J. Brown, M. Mattina, C.-C. Miao, C. Ramey, D. Wentzlaff, W. Anderson, E. Berger, N. Fairbanks, D. Khan, F. Montenegro, J. Stickney, and J. Zook, "TILE64 processor: A 64-core SoC with mesh interconnect," in *Proc. IEEE Int. Solid-State Circuit Conf.*, Feb. 2008, pp. 88–598.

[29] S. R. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, V. Erraguntla, C. Roberts, Y. Hoskote, N. Borkar, and S. Borkar, "An 80-tile sub-100-W TeraFLOPS PROCessor in 65-nm CMOS," *IEEE J. Solid-State Circuits*, vol. 43, no. 1, pp. 29–41, Jan. 2008.

[30] *International Technology Roadmap of Semiconductors*. (2008) [Online]. Available: http://www.itrs.com

[31] *Predictive Technology Model*. (2012) [Online]. http://www.eas.asu.edu/~ptm/

[32] D. C. Sekar, A. Naeemi, R. Sarvari, J. A. Davis, and J. D. Meindl, "Intsim: A CAD tool for optimization of multilevel interconnect networks," in *Proc. IEEE Int. Solid-State Circuits Conf.*, Nov. 2007, pp. 560–567.

[33] D. C. Sekar, "Optimal signal, power, clock and thermal interconnect networks for high-performance 2D and 3D integrated circuits," Ph.D. dissertation, Dept. Comput. Sci., Georgia Inst. Technol., Atlanta, 2008.

[34] *Intel Processor Numbers* [Online]. Available: http://www.intel.com/products/processor_number/chart/xeon.htm
[35] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "Hotspot: A compact thermal modeling method for CMOS VLSI systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 5, pp. 501–513, May 2006.
[36] *HotSpot5.0 Temperature Modeling Tool* [Online]. Available: http://lava.cs.virginia.edu/HotSpot/index.htm
[37] K. B. Wheeler, R. C. Murphy, and D. Thain, "Qthreads: An API for programming with millions of lightweight threads," in *Proc. IEEE Int. Symp. Parall. Dist. Process. Conf.*, Apr. 2008. pp. 1–8.
[38] A. F. Rodrigues, K. S. Hemmert, B. W. Barrett, C. Kersey, R. Oldfield, M. Weston, R. Risen, J. Cook, P. Rosenfeld, E. CooperBalls, and B. Jacob, "The structural simulation toolkit," *SIGMETRICS Perform. Eval. Rev.*, vol. 38, no. 4, pp. 37–42, Mar. 2011.
[39] M. Cho, N. Sathe, M. Gupta, S. Kumar, S. Yalamanchili, and S. Mukhopadhyay, "Proactive power migration to reduce maximum value and saptiotemporal non-uniformity of on-chip temperature distribution in homogeneous many-core processors," in *Proc. 26th Annu. IEEE Semicond. Thermal Meas. Manag. Symp.*, Feb. 2010, pp. 180–186.

Minki Cho (S'08) received the B.E. degree in electronics engineering from Sogang University, Seoul, Korea, in 2006, and the M.S. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, in 2009, where he is currently pursuing the Ph.D. degree with the School of Electrical and Computer Engineering.

He was an Intern with the Circuits Research Laboratory, Intel Corporation, Hillsboro, OR, in 2001. His current research interests include self-adaptive circuit analysis and design for thermal management in nanometer technologies, and testing methodologies for 3-D circuits and systems.

Chad Kersey (S'09) is currently pursuing the Ph.D. degree with the Georgia Institute of Technology, Atlanta. His thesis research is focused on the development of fast simulation technologies for microarchitecture research.

Man Prakash Gupta received the B.Tech. and M.Tech. dual degrees in mechanical engineering from the Indian Institute of Technology Kanpur, Kanpur, India, in 2009. He is currently pursuing the Ph.D. degree with the School of Mechanical Engineering, Georgia Institute of Technology, Atlanta.

His current research interests include electronics cooling and electrothermal modeling of carbon nanotube-based composites.

Nikhil Sathe received the B.E. degree in electronics and telecommunication engineering from Pune University, Pune, India, and the M.S. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, in 2006 and 2010, respectively.

He joined Advanced Micro Devices, Fort Collins, CO, where he is currently a Senior Design Engineer.

Satish Kumar received the Ph.D. degree in mechanical engineering from Purdue University, West Lafayette, IN, in 2007.

He joined the George W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, as an Assistant Professor in 2009. His current research interests include thermal management, atomistic transport models for nanostructures, flexible electronics, and thermoelectric coolers.

Dr. Kumar was a recipient of the Purdue Research Foundation Fellowship in 2005.

Sudhakar Yalamanchili (S'79–M'82–SM'91) received the Ph.D. degree in electrical and computer engineering from the University of Texas at Austin, Austin.

He joined the Faculty of Electrical and Computer Engineering, Georgia Institute of Technology (Georgia Tech), Atlanta, in 1989, where he is currently a Joseph M. Pettit Professor of computer engineering. Since 2003, he has been a Co-Director of the NSF Industry University Cooperative Research Center on Experimental Computer Systems, Georgia Tech.

Dr. Yalamanchili was the General Co-Chair of the 2010 IEEE/ACM International Symposium on Microarchitecture (MICRO) and was on the Program Committees for the 2011 International Symposium on Networks on Chip, the IEEE/ACM International Symposium on Microarchitecture, and the Micro Top Picks from Computer Architecture Conferences in 2011.

Saibal Mukhopadhyay (S'99–M'07–SM'11) received the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, in 2006.

He is currently an Associate Professor with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta. His current research interests include analysis and design of low-power and robust circuits in nanometer technologies and 3-D circuits and systems.

Dr. Mukhopadhyay was a recipient of the IBM Faculty Partnership Award in 2009 and 2010, the National Science Foundation CAREER Award in 2011, and the Office of Naval Research Young Investigator Award in 2012.