2016 IEEE International Parallel and Distributed Processing Symposium

TECfan: Coordinating Thermoelectric Cooler, Fan, and DVFS for CMP Energy Optimization

Wenli Zheng, Kai Ma, and Xiaorui Wang
Department of Electrical and Computer Engineering
The Ohio State University, Columbus, OH, USA
Email: {zheng.691, ma.495, wang.3596}@osu.edu

***Abstract*—The cooling needs of modern processors are dominantly provided by cooling fans, which can only offer global cooling capability. Even for cooling down one single hot unit (i.e., local hot spot), the fan needs to run at a high speed level, which consumes a lot of cooling power. Fortunately, emerging technologies, such as thermoelectric cooler (TEC), offer effective local cooling, which can be integrated with the fan to improve the overall cooling efficiency. However, relying on only TEC and fan may not optimize the total energy consumption of a chip multiprocessor (CMP), because the CMP core power states impact both computing and cooling power consumption. Therefore, for optimizing CMP energy, it is necessary to intelligently manage the processor power states and coordinate it with the cooling system.**

In this paper, we propose TECfan, a hierarchical runtime optimization framework that integrates TEC, fan, and DVFS for the overall energy efficiency of CMP. TECfan coordinates TEC and fan for efficient cooling, and also exploits DVFS to adapt the computing power consumption and execution time. Specifically, we first formulate CMP energy optimization with temperature constraint as a nonlinear optimization problem. Since there are no known polynomial-time algorithms for such a problem, solving it online is prohibitive. Hence, a novel heuristic algorithm is designed to solve it with acceptable time overheads. Our experiment results show that TECfan leads to 29% less energy consumption for medium workload compared to a state-of-the-art solution and 27%less overall compared to fan-based cooling.

I. Introduction

The semiconductor industry has observed the end of classic Dennard scaling [1] in recent years. As the transistor integration outpaces supply voltage scaling, the power density of microprocessor chips increases rapidly [2], which exacerbates the thermal issues on processors. Traditionally, processor cooling relies on cooling fans that are driven by motors with feedback controllers, such that the fan speed is adjusted by on-board firmware or management algorithms running on the CPU [2]. Unfortunately, this solution has the following two major limitations. First, linearly increasing the fan speed introduces a cubic increase of the fan power consumption [3]. For example, the fan system on an IBM P670 server, when running at its highest speed, consumes approximately 51% of the total server power [4]. Second, the fan system can only provide global cooling, which cannot efficiently address local hot spots on a chip multiprocessor (CMP) [5]. For example, among all the cores, if only some of them become hot, the fan has to work at a high speed to lower the temperatures of all cores, resulting in unnecessarily low cooling energy efficiency. One of the key reasons for high cooling power is that those management policies were mainly designed to address overheating without sufficient consideration of energy efficiency. Therefore, novel cooling solutions must be designed to improve the energy efficiency by selectively cooling down only the local hot spots.

Thermoelectric cooler (TEC) is a new type of film material that can actively pumps heat from one side to the other side. Multiple TECs can be deployed on top of each processor core to dissipate heat to the heat spreader. Compared with conventional fan-based cooling, TEC-based cooling has a promising potential to manage the local hot spots [6]. However, previous studies usually assume a fixed fan speed when designing TEC management strategies [7], without exploring any coordination between TECs and fan. Since a cooling fan with adjustable speed can provide flexible global cooling and TECs excel in removing local hot spots, intelligently coordinating the two can improve the overall energy efficiency of the CMP cooling subsystem. While only a very recent study [8] has proposed to integrate TECs and fan, their solution has to solve a non-convex nonlinear optimization problem. The high time overhead of such a solution makes it difficult to perform online cooling management, especially for future CMPs with many cores. Therefore, practical solutions with low overheads must be proposed to coordinate fan and TEC online.

More importantly, reducing only the cooling power consumption with the integration of TEC and fan may not lead to overall CMP energy optimization, which is the ultimate goal of improving CMP energy efficiency. The key reason is that there exists a strong correlation between the computing power consumption of a CMP and its cooling demand. For example, active power reduction approaches, like dynamic voltage and frequency scaling (DVFS), can be used to mitigate local hot spots without demanding for more cooling power, at the cost of possible performance degradation. A salient feature of DVFS is that a CMP can gain a cubic dynamic power reduction with only a linear core frequency (and so performance) degradation. Therefore, with careful designs, the integration of DVFS with cooling energy optimization can allow a significant reduction of CMP energy consumption, only at the cost of negligible performance degradation. Unfortunately, existing work, such as [8], focuses only on cooling power reduction without exploring this important direction.

Based on the above observations, this paper presents TECfan, a hierarchical runtime optimization framework that coordinates the TEC, fan, and DVFS for optimized energy efficiency of CMP. In sharp contrast to most existing work that manages TEC and fan in an isolated manner, TECfan features a hierarchical solution. It uses TECs on each core to handle local hot spots for temperature balancing among different cores, such that the fan (for global cooling) no longer needs to be set at a high speed to cool down local hot spots. As a result, the overall CMP cooling efficiency can be significantly improved. Furthermore, TECfan coordinates cooling management with DVFS to achieve minimized CMP energy (for both cooling and computation) by considering the impacts of DVFS on computing power consumption, cooling need and execution time. TECfan formulates the coordination of TEC, fan, and DVFS as a multi-variable energy optimization problem with temperature constraints. Since the search space of the optimization problem is prohibitively large, we propose a novel and efficient heuristic algorithm to solve this problem with low time overhead and negligible performance degradation. Our results show that TECfan leads to 29% less energy consumption for medium workload compared to a state-of-the-art solution and 27%less overall compared to fan-based cooling. TECfan is also shown to achieve comparable results with the oracle solution that exhaustively searches the entire solution space. Specifically, this paper makes the following major contributions:

- While previous work addresses TEC, fan, and DVFS in an isolated manner, we propose TECfan, a hierarchical runtime optimization framework that explores the interplay among the three knobs for significantly improved energy efficiency.
- We analytically formulate the management of the three knobs as a multi-variable energy optimization problem with temperature constraints, whose solution can minimize the overall CMP energy consumption.
- To reduce the time overhead of TECfan, we propose a novel heuristic algorithm to solve the problem with negligible performance degradation. Our extensive results show that TECfan can save 29% energy compared to a state-of-the-art solution.

The rest of this paper is organized as follows. Section II highlights the differences between this paper and related work. Section III sketches the system design. Section IV introduces our simulation setups and the implementation details of our solution. Section V presents the evaluation results. Section VI concludes the paper.

II. Related Work

Thermoelectric cooler (TEC) has been studied for on-chip cooling for several years. Chowdhury et al. [6] have shown a prototype chip of thin-film TEC and presented key parameters. Gupta et al. [9] have examined the transient temperature behaviors of TECs. Those studies focus on physical parameter characterization. At the architecture level, Long et al. [10] have studied the optimal amount of TECs and their locations on chips. Chaparro et al. [5] have conducted several case studies on how to manage the on/off state of TEC devices. Biswas et al [7] have shown that using TEC to improve cooling efficiency can reduce cooling cost in data centers. Ikeda et al. [11] have prototyped a power-efficient low-noise cooling system with TEC devices. Rho et al. [12] have studied using TEC and DVFS to co-optimize cooling energy on a 3D cache-stacked system. Although those studies all explicitly or implicitly assume the existence of fan in the cooling subsystem, they do not explore the interaction between TEC (local cooling) and adjustable fan (global cooling) to optimize their total energy consumption.

Some previous work has considered integrating TEC with fan. For example, Dousti et al. have proposed some strategies to improve the power efficiency of TEC [13], [14] and the *OFTEC* scheme to optimize the total cooling power consumed by TEC and fan [8]. Compared to *OFTEC*, TECfan optimizes the total CMP energy consumption (for both cooling and computation) by coordinating DVFS with TEC and fan, since DVFS affects the computing power consumption, cooling need and execution time of workload. Jayakumar et al. [15] integrates all the three knobs to improve the computing performance at the cost of additional cooling power. In contrast, TECfan focuses on improving energy efficiency with negligible performance degradation. To the best of our knowledge, TECfan is the first study that coordinates the TEC, fan, and DVFS to improve energy efficiency within temperature constraints.

There have been several studies on joint co-optimization of cooling and computing power. Shin et al. [2] have discussed the co-optimization of CPU and fan. Compared with discussing the trade-off between the fan and the CPU, TECfan addresses a more challenging multi-variable optimization problem. Ayoub et al. [4] have introduced a cooling-aware task scheduling. Indrani et al [16] have examined the thermal interaction between CPU and GPU when they are deployed on the same die. However, those studies neglect the local cooling capacity that TEC can offer.

III. System Design

In this section, we present the system design of TECfan, which is illustrated in Figure 1. The TEC devices are embedded between the heat spreader and the processor chip in the thermal interface material layer, and usually there are multiple TECs deployed on each core. We assume that the through-silicon via (TSV) [17], an extensively studied

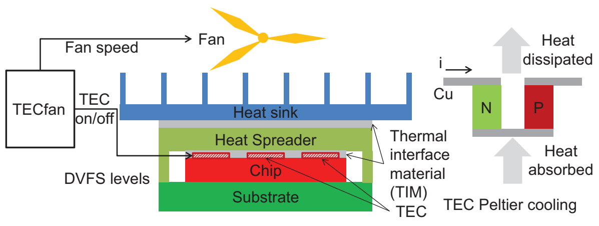


Figure 1. The side view of the target chip packaging and TEC (thermoelectric cooler) cooling effect. TECs are embedded between the heat spreader and the processor chip in the Thermal Interface Material layer.

technique in 3D stacking, is used to connect the TEC and the on-chip power delivery network. Although it is feasible to adjust the efficacy of a TEC by manipulating its current, this method requires dedicated on-chip voltage regulator (VR) that can be costly to implement. Thus we assume using power transistors to control the on/off state of TECs. By turning on TEC, the heat can be pumped from the chip side to the heat sink side. In this way, we enable fine-grain local cooling that can be specified for each core. We integrate the TEC operation with the adaptation of fan, which enables global cooling, to improve the overall cooling efficiency. In addition, we also exploit DVFS to coordinate the computing and cooling power. By adjusting the TEC on/off state, per-core DVFS level and fan speed, we minimize the energy consumption of the CMP system, with the constraint that the peak temperature is always below a safe threshold.

In the following sections, we first introduce the models used to estimate the thermal, power, and performance of the target system. We then use those models to formulate the energy minimization as a multi-variable optimization problem. Finally, we present our heuristic algorithm to solve the problem with low computational overhead and negligible performance degradation.

A. Thermal Models

We adopt a widely-used steady state thermal model [10] for multiple components on a chip:

$$\widehat{G}(k)\widehat{Ts}(k) = \widehat{P}(k) \tag{1}$$

where $\widehat{Ts}(k) = [Ts_1(k), Ts_2(k), \cdots, Ts_N(k)]^T$ is the steady state temperature vector of all the components in the k^{th} time interval; $\widehat{P}(k) = [P_1(k), P_2(k), \cdots, P_N(k)]^T$ is the power vector of all the components; $\widehat{G}(k)$ is the thermal conductance matrix among the components:

$$\widehat{G}(k) = \begin{bmatrix} g_{11}(k) & \cdots & g_{1N}(k) \\ \vdots & \ddots & \vdots \\ g_{N1}(k) & \cdots & g_{NN}(k) \end{bmatrix} \tag{2}$$

where g_{ij} is the thermal conductance between node i and j (g_{ij} is 0 if node i and j are not adjacent). Usually, the parameters needed to define $\widehat{G}(k)$ can be extracted from a full-blown SPICE model, which is available at the stage of packaging design. Even such a model is not available, we can still use HotSpot [18] or other micro-architectural thermal models to estimate $\widehat{G}(k)$. Since $\widehat{G}(k)$ is impacted by fan and TECs, we can dynamically adjust it by changing the fan speed and the on/off state of TECs. In addition to $\widehat{G}(k)$, we can also adjust $\widehat{P}(k)$ through DVFS. The fan power and air flow rate (used to calculate the impact of fan on $\widehat{G}(k)$) can be derived from a fan datasheet [19]. The TEC power and the impact of TECs on $\widehat{G}(k)$ can also be derived from the product datasheet, and in this paper we use the models and parameters from [10].

Please note that Equation (1) only presents the steady state temperature model. To estimate the transient temperature behavior, we use the well-known RC thermal model [20]:

$$\frac{dT(k)}{dt} = \frac{1}{C_{th}} * P(k) - \frac{1}{R_{th}C_{th}}(T(k) - T_\alpha) \tag{3}$$

where $T(k)$ is the processor temperature, T_α is the ambient temperature, R_{th} is the thermal resistance, C_{th} is the heat capacity, and $P(k)$ is the power feeding into the RC model. By solving Equation (3), we have:

$$T(k) = (1-\beta) * Ts + \beta * T(k_0), \beta = e^{-\frac{k-k_0}{R_{th} * C_{th}}} \tag{4}$$

where Ts is the steady state temperature and $T(k_0)$ is the initial temperature, based on which the transient temperature $T(k)$ can be interpolated. By discretizing Equation (4), we obtain the difference equation:

$$T(k) = (1-\beta) * Ts + \beta * T(k-1), \beta = e^{-\frac{\Delta k}{R_{th} * C_{th}}} \tag{5}$$

where Δk is the length of time interval. By solving Equation (1), we can get $\widehat{Ts}(k)$. Then we plug $\widehat{Ts}(k)$ into Equation (5) to estimate the transient temperature results of our actuators (i.e., TEC on/off states, fan speed, and per-core DVFS).

B. Power and Performance Models

The power consumption of a processor component (e.g., adder, multiplier or instruction cache) m contains two parts, the leakage power and the dynamic power, i.e., P_m = P_{leak_m} + P_{dyn_m}. The leakage power can be calculated as:

$$P_{leak_m}(k) = (P_{TDP_{leak}} + \alpha * (T_m(k-1) - T_{TDP})) * \frac{A_m}{A_{chip}} \tag{6}$$

where $P_{TDP_{leak}}$ is the leakage part of the Thermal Design Power (TDP), which is determined based on the temperature limit T_{TDP}. These two can be derived from the datasheet and used as constants in our on-line estimation. A_m is the area of component m and A_{chip} is the total chip area. Using a linear temperature model to estimate leakage power within a limited temperature range has been shown to be reasonably accurate in [2], [21]. The dynamic power is calculated as:

$$P_{dyn_m}(k) = P_{dyn_m}(k-1) * \frac{F_m(k)}{F_m(k-1)} * (\frac{Vdd_m(k)}{Vdd_m(k-1)})^2 \tag{7}$$

where $F_m(k)$ and $Vdd_m(k)$ are the frequency and voltage of component m in the k^{th} time interval, respectively. We calculate the dynamic power ($P_{dyn_m}(k)$) based on its value in the previous time interval ($P_{dyn_m}(k-1)$), and the latter can be precisely estimated by monitoring only six performance metrics as proposed in [22]. By adding the power consumptions of all the components of a core, we can get the total power of the core P_{core_n}.

The power consumption of the entire target system is the summation of core power, TEC power and fan power, i.e.,

$$P_{chip} = \sum_{n=0}^{N-1} P_{core_n} + \sum_{l=0}^{L-1} P_{TEC_l} + P_{fan} \tag{8}$$

with N cores and L pieces of TEC devices in the packaging. The TEC power can be calculated as:

$$P_{TEC} = r * I^2 + \alpha I \Delta\theta \tag{9}$$

where I is the current applied on the TEC; r and α are the material parameters; $\Delta\theta$ is the temperature difference between the two sides of the TEC device. Since applying more than 8A has been identified as dangerous to introduce overheating [10], we conservatively assume the current is 6A. The fan power is determined by the fan speed, and the relationship between them can be obtained from the datasheet or modeled by profiling. For example, the datasheets of some processor fans list the power consumption with respect to each speed level [19].

We use the instructions per second (IPS) as the metric to quantify the computational performance of CMP. The total IPS of the chip IPS_{chip} can be calculated as the summation of the IPS of all the cores:

$$IPS_{chip}(k) = \sum_{i=0}^{N-1} IPS_n(k) \tag{10}$$

$$IPS_n(k) = IPS_n(k-1) * \frac{F_n(k)}{F_n(k-1)} \tag{11}$$

The IPS of each core IPS_n is derived based on the frequency scaling and the IPS in the previous time interval (which is commonly measured system statistics).

C. Problem Formulation

Our goal is to minimize the per-instruction CMP energy consumption, including both the cooling and computational part, with peak temperature constraint. Our energy model for the chip is based on the power model:

$$En_{chip} = P_{chip} * Time_{chip} \tag{12}$$

where En_{chip} is the energy consumption of entire chip; $Time_{chip}$ is the execution time of workload, which models the performance of CMP and is inversely proportional to the execution speed IPS_{chip}. Therefore, the per-instruction energy consumption is EPI_{chip} = P_{chip} / IPS_{chip}. Therefore, the formal formulation of the problem is:

$$min\{EPI_{chip}(k) = \frac{(\sum_{n=0}^{N-1} P_{core_n} + \sum_{l=0}^{L-1} P_{TEC_l} + P_{fan})}{\sum_{0}^{N-1}(IPS_n(k-1) * \frac{F_n(k)}{F_n(k-1)})}\} \tag{13}$$

subject to

$$max\{\widehat{T}(k)\} \le T_{th} \tag{14}$$

With TECfan, EPI_{chip} can be minimized by adjusting TEC states (impact $\widehat{G}(k)$ and P_{TEC_l}), fan speed (impact $\widehat{G}(k)$, and per-core DVFS (impact $\widehat{P}(k)$, P_{core_n}, and F_n) and P_{fan}). $\widehat{G}(k)$ can then impact the constrained variable $\widehat{T}(k)$, i.e., the transient temperature of processor components.

D. Heuristic Solution

To solve the problem formulated in Section III-C, in general, exhaustive search can be used to find the optimal solution, which is unfortunately prohibitive for on-line management due to the large computational time overhead. Therefore, we develop a multi-step down-hill heuristic algorithm to simplify solving the problem based on the following observations: 1) The three different knobs, i.e., TEC on/off state modulation, per-core DVFS and fan speed adjustment, have different impacts on the chip temperature. The TEC modulation impacts the heat dissipation capacity of each core; per-core DVFS impacts the local power consumption and hence heat generation; the fan speed adjustment impacts the global thermal characteristics. 2) The amount of time needed for each knob to take effect is different. The fan cooling effect takes place through the heat sink and heat spreader, whose heat capacity is normally hundreds of Joule per Kelvin (several seconds to stable). In contrast, the effect of TEC and DVFS engages much faster. The cooling effect of TEC device (i.e., Peltier effect) takes up to 20μs [9] to engage. The overhead of per-core DVFS is decided by the on-chip voltage regulator, which is reported to be at the order of tens of nanoseconds [23].

Those two key observations inspire us to develop the following two-level hierarchical algorithm for *TECfan*. At the lower level, we use per-core DVFS and TEC device on/off state modulation to save energy and control temperature. These operations can be conducted with a fine time scale, e.g., 2ms, since the TEC modulation overhead is 20μs, and the DVFS overhead is 100ns [24]. The key idea is to use the thermal, power, and performance models presented in Sections III-A and III-B to estimate the temperature and per-instruction energy consumption in the next time interval if certain adjustment is made. We then select the adjustment that has the least energy consumption within the temperature constraint. At the higher level (with a relatively coarse time scale, e.g., a few seconds), we use the fan speed modulation to control temperature and save energy.

In detail, our heuristic algorithm works as follows. In each lower-level time interval, we assume the fan speed is fixed and use Equations (1) and (5) to estimate the temperature in the next interval if the DVFS levels and/or TEC on/off states are adjusted. As shown in Figure 2, if some components have higher temperature than the threshold (i.e., $max\{\widehat{T}(k)\} > T_{th}$), the algorithm enters the *hot iteration*. To reduce the temperature, we can turn on the TEC devices or decrease the DVFS levels. Since we want to avoid significant performance degradation, our algorithm starts with turning on TEC devices. Specifically, we estimate the thermal impact of turning on the TEC on top of the hottest spot. If the estimated temperature is still higher than the threshold, we

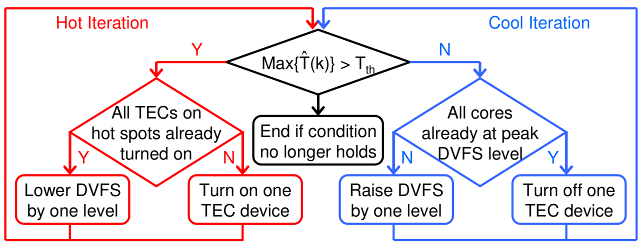

Hot Iteration
Cool Iteration
Y
Max{$\hat{T}$(k)} > T_{th}
N
All TECs on hot spots already turned on
End if condition no longer holds
All cores already at peak DVFS level
Y
N
N
Y
Lower DVFS by one level
Turn on one TEC device
Raise DVFS by one level
Turn off one TEC device

Figure 2. Multi-step down-hill heuristic algorithm flow chart. Based on the thermal, power, performance models, *TECfan* estimates the next step energy consumption if certain adjustment is made; it then selects the adjustment that has the smallest energy consumption within the temperature constraint. The hot or cool iteration ends when the condition $max\{\widehat{T}(k)\} > T_{th}$ changes from true to false or from false to true, respectively.

repeat the hot iteration until the estimated temperature is lower than the threshold. If all the TECs on top of the hot spots (where the component temperature is higher than the threshold) are already turned on but there still exist hot spots, we start decreasing the DVFS levels. We select the DVFS adjustment that offers the smallest per-instruction energy consumption to estimate the temperature in the next time interval. If the temperature is still higher than the threshold, we keep decreasing the DVFS levels until the estimated temperature is lower than the threshold. By first considering turning on TECs and then lowering the DVFS levels, we minimize the use of throttling to reduce temperature. In addition, since we select the DVFS levels that result in the lowest energy consumption, the CMP energy is reduced.

On the other hand, if there are no hot spots, the algorithm enters the *cool iteration*, which is also illustrated in Figure 2. We evaluate the opportunity to change the DVFS levels or turn off the TEC device to save energy. First, we calculate the energy consumption in the next interval if we increase the DVFS level of a core by one step. After getting the results for all the cores, we select the configuration that can save the most energy. We then estimate the temperature. If there still exist no hot spots, we use the new configuration as the starting point of the next iteration, and repeat this process until hot spots occur. If all the cores are already at the highest DVFS level, we calculate the energy consumption in the next interval if we turn off the TEC on top of the coolest component. Similarly, we select the configuration that can save the most energy as well, and then update the temperature estimation. If there are still no hot spots, we use the new configuration as the starting point of the next iteration until there is no opportunity to conserve energy consumption by turning off the TEC without violating the temperature constraint.

We define this temperature estimation, comparison, and energy estimation process as one iteration, and there can be multiple iterations for each time interval (i.e., control period). When the condition $max\{\widehat{T}(k)\} > T_{th}$ changes from true to false in a hot iteration, or from false to true in a cool iteration, we end the iterations, and apply the configurations of TECs and DVFS that minimize the per-instruction energy consumption and meet the temperature constraint to the chip.

In each higher-level time interval, we adjust the fan speed based on the total power and peak temperature of the chip, like the current industry practice on cooling system. We use the average power of the last interval as the power reading, and the average TEC on/off state in the last interval as the TEC state for temperature estimation, which means we can have intermediate state besides the on/off states. Then we estimate the temperature by using Equations (1) and (5). If hot spots exist, we increase the fan speed level until the hot spots disappear. If there are no hot spots, we evaluate the opportunity to slow down the fan to save energy, and decrease the fan speed until the hot spots occur.

E. Hardware Cost

We estimate the hardware cost of our proposed solution TECfan in this section. The major hardware overhead is due to the temperature estimation with Equations (1) and (5). Since the thermal impact only takes place on adjacent components, $\widehat{G}$ is by-nature a band matrix, which means the matrix has nonzero elements only along the main diagonal and some additional minor diagonals (typically only one) on either side of the main diagonal. Band matrix and vector multiplication can be implemented in a systolic array [25], which is fast and space-efficient. Since the inter-core thermal impact is limited in tile-structured many-core architectures, we only evaluate the temperature of one core each time. Note that we use the thermal conductance matrix $\widehat{G}$ to be consistent with our evaluation framework HotSpot simulator, because HotSpot uses thermal conductance matrix. In real hardware implementation, in addition to using the thermal conductance matrix, we can also select using the thermal resistance matrix. In that case, the calculation to derive thermal conductance from thermal resistance could be bypassed and only the matrix-vector-multiplication is needed, which means less hardware overhead.

We analyze the power and area cost of an aggressive design, in which the systolic array gives the temperature results of one core in one cycle, where we need $K \times M$ fixed-point multiplication (M is the number of components in one core, K is the number of components that have thermal impact). For temperature and energy comparison, 8-bit encoding is sufficient. Bitirgen et al. [26] have estimated that a 16-bit fixed-point multiplier yields an area of 0.057mm^2 with 65nm process technology. For a typical 200mm^2 die, the hardware overhead is only 0.03%. To approximate the power consumption by the fixed-point multiplier circuits in the hardware, we adopt the power density of IBM POWER6's FPU [27], which is 0.56W/mm^2 at 100% utilization with nominal voltage and frequency values (1.1V and 4 GHz). Hence the extra power consumed by the fixed-point multiplier is only 0.03W. In our design, we evaluate 18

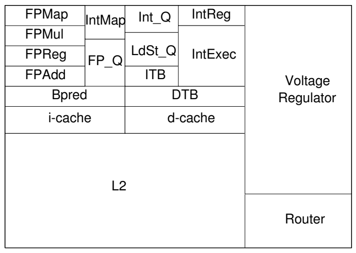


Figure 3. The overview of a core tile (2.6mm×3.6mm), which is half the size of the dual-core tile on Intel SCC. The chip floor plan: 10.4mm×14.4mm, a 4×4 core tile array.

processor components and assume only adjacent components have thermal interaction. We use $M \times K = 18 \times 3 = 54$ eight-bit fixed-point multiplications in order to evaluate the temperature of one core in one cycle, which adds only less than 1.7% extra area and power to the target CMP system. The other computation of TECfan can time-share the calculation unit of the temperature estimation part. Therefore, TECfan is an affordable solution in terms of hardware costs.

Note that TECfan does not rely on per-core DVFS. We use per-core DVFS only to show that even with per-core DVFS enabled, TECfan is still able to handle the large search space that is introduced by this extra knob. TECfan can be integrated with chip-level DVFS seamlessly.

IV. Experiment Setup

In this section, we introduce our experiment environment.

A. Processor Model

We model a 16-core processor based on the floor plan of Intel single-chip cloud (SCC) processor [28], because Intel published detailed sizing and power parameters [28] that are unavailable for other chips. The overview of one core tile is shown in Figure 3. The placement and relative size of each component of the core are based on the Alpha 21264 architecture because it makes this very detailed information available to the public. Each core has a private 256KB, 16-way L2 cache and each cache line has 64Bytes.

The size of voltage regulator (VR) is estimated based on the measurement of delivered power, which is 0.5W/mm^2 on a prototype on-chip regulator [23]. The peak power of one core is 1.8W, resulting in a 3.6mm^2 VR. Considering the core size is only 4mm^2, the VR has too much area overhead. Therefore, we use a quasi-parallel VR [29]. The key idea is that we connect the off-chip regulator and on-chip regulator in parallel. We rely on off-chip converter to deliver most of the output power, while using a low-power on-chip buck type converter to achieve voltage regulation. In the proposed off-/on-chip hybrid VR design, for 1.8W peak power, only 0.9W needs to be provided by on-chip VR at the steady state. However, to be conservative, we budget the on-chip VR with 2.2mm^2, resulting in 1.1W peak power delivery capability. Thus the size of VR has been reduced by 40% in our proposed scheme. Although the on-chip VR still occupies a large on-chip area, this 24% area overhead (i.e., 2.2mm^2 on a 3.6×2.6 core tile) could be justified by the dark silicon effect [24] (e.g., 20% area is dark).

B. Simulators

The SESC [30] simulator is used to evaluate the processor performance and the SPLASH-2 benchmarks are used as the workload. SESC has been integrated with Wattch [31] and CACTI [32] to estimate the power of each on-chip component. We calibrate the peak power estimated by Wattch to the SCC measurement results [28], and use a second-order polynomial model [21] to estimate the leakage power (also calibrate it to the SCC leakage power measurement [28]). We assign the chip leakage power to each component in proportion to the area of each component.

We use HotSpot 5.02 [18] to estimate the temperature. Please note that HotSpot 5.02 only integrates temperature-leakage loop in the steady temperature routine. Hence we modify the transient temperature calculation routine to consider temperature-leakage loop at run-time. At the beginning of simulation, we use a default uniform initial temperature. Then from the second time interval, we use the result temperature from the previous interval as the initial temperature of each component, and run HotSpot to estimate the temperature. We repeat this process until the difference between the peak temperatures in two consecutive intervals is less than 0.5°C. With such simulation iteration, our results converge to the stable value.

A different simulation setup has been used in Section V-E for the comparison among TECfan, the *Oracle* solution (described in Section V-A), and a state-of-the-art solution *OFTEC* [8]. The key reason for us to use a different simulation setup is that the time complexity of *Oracle* does not scale with the number of cores to be simulated, because it solves the energy optimization problem in Section III-C by exhaustive search. A similar problem exists for *OFTEC* as well. Therefore, in Section V-E, we simulate a 4-core CMP in this case to run the workload generated based on a trace file of HTTP service from Wikipedia [33]. Since the processor utilization derived from the trace is too low to activate much usage of TECs, we scale up the utilization by a factor of 1.5. We calculate the power consumption according to the model proposed by [34] and the parameters of Core i7-3770K processor [35]. The system performance is derived as a quadratic polynomial function of the frequency, by curving fitting with the testing results of SPECjbb in [36].

C. TEC and Fan Settings

We assume using the same TEC device as [10], which is a 0.5mm×0.5mm film-form material. We embed a 3×3 TEC array on top of each core tile between the heat sink and processor die, to cover the most core area. Each TEC device is controlled independently by a power transistor, which is coupled to the on-chip power delivery circuit by through silicon via (TSV) [37]. Because the cooling effect of TEC (i.e., Peltier effect) takes up to 20μs to engage [9], we calculate the change of thermal conductance 20μs after turning on a TEC. Please note this setting is conservative, and the cooling effect will be better with a shorter delay.

We assume using a speed adjustable fan in the CMP packaging. The available speed levels as well as the fan power and air flow rate at different speed levels are obtained from a commercial fan datasheet [19]. The impact of fan is related to the heat sink, whose thermal constant is in the range of 15-30s [4]. If we simulate the dynamic effect of fan, the simulated CPU time should be at least several minutes. However, for a 16-core processor in SESC (integrated with HotSpot), the chip temperature usually becomes stable in less than 1 second with the operations of TECs and DVFS, during which the fan speed is almost a constant. Therefore, for each benchmark, we run all the studied policies with all possible fan speed levels in multiple tests, and choose the results with the lowest fan speed without violating the temperature threshold.

V. Experiments

In this section, we present our evaluation results.

A. Baseline Policies

We compare *TECfan* with four well-designed baselines in Sections V-C and V-D, and compare *TECfan* with a state-of-the-art scheme and an oracle solution in Section V-E.

Fan-only applies the lowest fan speed level without violating the temperature constraint but conducts no operations of TECs or DVFS. Please note that this is an ideal solution which could not be implemented in the real world, because the cooling effect of adjusted fan speed takes several minutes to engage, but it is difficult to accurately predict the temperature behavior during this amount of time.

Fan+TEC adjusts fan speed and TEC on/off states independently for cooling. It operates the fan speed in the same way as *Fan-only*. For TECs, when the temperature of an on-chip component is higher than the threshold, we turn on the TEC on top of the hot spot; when the temperature of all the components below a TEC is lower than the threshold, we turn off the TEC. We assume temperature sensing is available at all components (also assumed in [10], [5]).

Similarly, *Fan+DVFS* manages fan speed and DVFS independently. The fan is operated in the same way as *Fan-only*, and a classic DVFS-based dynamic thermal management algorithm is adopted. Specifically, we raise the DVFS level of a core when its temperature is below the threshold to improve performance, and lower the DVFS level when the temperature is higher than the threshold.

DVFS+TEC uses all the three knobs (fan speed control, TEC on/off state modulation and DVFS) but manages them independently. The TEC is operated based on the local

Table I
Testing results in the base scenario

Workload	Inputfile	FF Inst	Threads	Inst	Time (ms)	Power (W)	T(°C)
cholesky	tk29.O	200M	16	1 billion	48.0	125.9	90.07
			4	250M	57.2	42.0	74.8
fmm	fmm.in	300M	16	1 billion	59.68	74.9	69.69
			4	250M	72.66	32.5	62.15
volrend	head	300M	16	800M	41.42	85.4	71.79
water	water.in	300M	4	250M	38.1	43.7	68.7
lu	no input	300M	16	400M	20.34	109.9	84.49
			4	100M	19.6	42.1	70.75

temperature as in *Fan+TEC*, and the DVFS is also operated based on the local temperature as in *Fan+DVFS*. Neither of the two is aware of the adjustment of the other. The fan is operated in the same way as *Fan-only*. We compare *TECfan* with the above four baselines to show the importance of coordinating the three knobs.

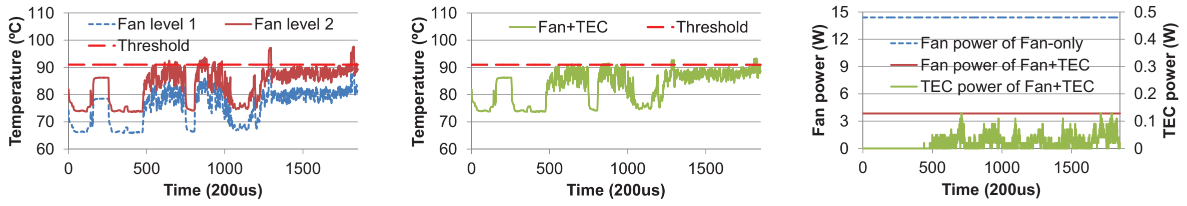
The state-of-the-art scheme *OFTEC* [8] minimizes the total cooling power consumption of TEC and fan, by solving a nonlinear power optimization problem with temperature constraint. The impact of temperature on the leakage power of processor is also considered. Differently, *TECfan* actively adapts the processor power consumption by DVFS for more energy saving, and coordinates DVFS with TEC and fan for more efficient cooling.

Oracle is the optimal solution of the problem in Section III-C. It guarantees the minimized overall CMP energy by exhaustive search, but its time complexity does not scale with the number of cores, which prohibits *Oracle* from practical online management. We compare *TECfan* with it to evaluate the gap between *TECfan* and the optimal solution.

While *OFTEC* in [8] adopts the active-set SQP method to speed up the searching and find a near-optimal solution, here we make *OFTEC* to do exhaustive search like *Oracle* and find the optimal solution, since we do not compare their time overheads in the experiments. Hence the time complexity is $O(2^{NL})$ for *OFTEC* and $O(M^N 2^{NL})$ for *Oracle*, where L, M and N are the numbers of per-core TECs, DVFS levels and cores, respectively. Compared to them, the time complexity of *TECfan* is only $O(NL + N^2M)$, because it can repeat the hot or cool iteration for at most NL times to manipulate all the TECs and NM times to change DVFS levels of all the cores. When changing DVFS levels, it compares the change of each core and select the one with the lowest energy consumption.

B. Importance of Integrating TEC with Fan

In this experiment, we compare the cooling effects (in terms of peak temperature) and cooling costs (in terms of cooling power) of *Fan-only* and *Fan+TEC*, to show the effectiveness of integrating TEC with fan. At first, we generate the results for the base scenario, i.e., running all the cores at the peak DVFS level and the fan at its highest speed, and turning off all the TECs. We run 16 or 4 threads on the 16-core system each time, with the parameters and results



(a) Temperature traces of *Fan-only*. (b) Temperature trace of *Fan+TEC*. (c) Power traces of both.

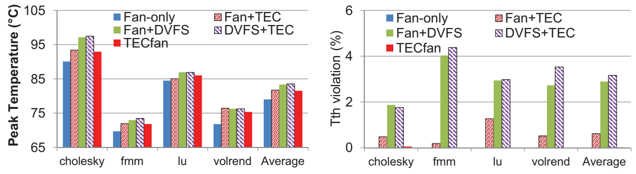
Figure 4. Temperature and cooling power comparison between *Fan-only* and *Fan+TEC*. Using the 1st (highest) fan speed level can achieve much better cooling than using the 2nd fan speed level (shown in (a)). However, using TECs and the 2nd fan speed can achieve almost the same cooling effect as that of the 1st fan speed level (shown in (b)). In addition, the total cooling power of using TECs and the 2nd fan speed is much lower than running fan at 1st speed level (shown in (c)), due to the cubic dependence of fan power consumption on the fan speed [4].

presented in Table I. When we run 16-thread *water* and 4-thread *volrend*, the simulation suspends before reaching the required number of instructions. Therefore, we only report the 4-thread *water* and 16-thread *volrend* cases. We set the peak temperature in the base scenario as the temperature threshold T_{th} in our experiments.

In Figure 4(a), we can see that with the highest fan speed level (i.e., "Fan level 1"), *Fan-only* can successfully keep the peak temperature below the threshold. However, simply setting the fan on the 2nd highest speed level (i.e., "Fan level 2") for *Fan-only* will introduce multiple temperature violations. Figure 4(b) shows that *Fan+TEC* significantly decreases the temperature when the fan is running at the 2nd highest speed level. The temperature is always below the threshold except for two data points. The comparison between Figures 4(a) and (b) shows the effectiveness of using TECs to address hot spots. Figure 4(c) shows the corresponding cooling power in the two cases. The left y-axis is the power scale of the fan part; the right y-axis is the power scale of the TEC part. Since the power of a fan has a cubic relationship with its speed [4], the highest fan speed level consumes much more power (14.4W) than the 2nd highest fan speed level (3.8W). Here the fan power is estimated based on a Dynatron R16 fan datasheet [19] designed for Intel Core i5 packaging. The total power consumed by *Fan+TEC*, i.e., the summation of fan power at the 2nd speed level and the TEC power, is still much lower than the power of *Fan-only* that runs the fan at the 1st speed level, while the two solutions achieve close cooling effects.

C. Cooling Performance

We apply *TECfan* and the baseline policies in the simulation with different workload benchmarks, and compare their cooling effects here. Figure 5(a) shows the peak temperature achieved by each policy. For three out of the four benchmarks, *cholesky*, *fmm* and *lu*, *Fan+TEC* has a lower peak temperature than *Fan+DVFS*, because *Fan+TEC* uses TECs to effectively address the local hot spots. However, for *volrend*, the workload with high power consumption and relatively uniform power density distribution across the chip, *Fan+DVFS* has a lower peak temperature than *Fan+TEC*. Interestingly, we find that in all the tested cases,



(a) Peak temperature (b) T_{th} violation points/total points

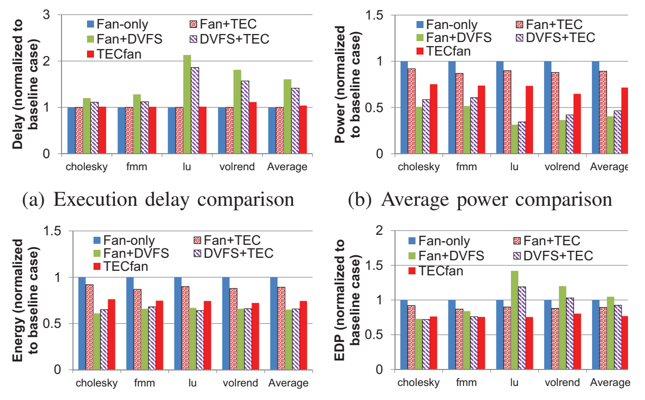
Figure 5. Cooling performance comparison. We set the temperature threshold T_{th} according to Table I. *Fan-only* operates in the same way as the base scenario, and hence has no T_{th} violation.

the simple combination of DVFS and TEC, i.e., *DVFS+TEC*, has a higher peak temperature than that of either *Fan+TEC* or *Fan+DVFS*, because *DVFS+TEC* does not consider the interference between the two knobs. For example, when the current temperature is lower than the threshold, the TEC management algorithm turns off some TECs, while the D-VFS algorithm raises the DVFS level at the same time. As a result, in the next time interval, the temperature can become too high. In contrast, *TECfan* consistently has the lowest temperature in the studied cases because it coordinates all the available knobs to fully take advantage of each.

For Figure 5(b), we set the peak temperature of each workload in the base scenarios as the temperature threshold T_{th}. Due to dynamic management, the peak temperatures of studied policies have occasional temperature violations. Since in the base scenarios the fan is operated at the highest speed level, T_{th} is the best cooling performance that the fan-based cooling can offer (at the cost of large power). Therefore, *Dyanmic-fan* adopts the highest fan speed as well because using any other fan speed will cause many temperature violations as shown in Figure 4(a). *Fan+DVFS* and *DVFS+TEC* have more violations than *Fan+TEC*, because changing the DVFS level by one step has a much greater impact on temperature than turning on/off one TEC device. *TECfan* coordinates all the three knobs to achieve a small violation ($< 0.5\%$) for all the benchmarks, e.g., when the DVFS level is raised by one step, *TECfan* can turn on some TECs simultaneously to mitigate the temperature change.

D. Energy Efficiency and System Performance

In this section, we compare the execution delay, average power, energy and Energy Delay Product (EDP) [38] of



(a) Execution delay comparison

(b) Average power comparison

(c) Total energy comparison

(d) Energy delay product comparison

Figure 6. Execution performance comparisons. Since *TECfan* reduces the power without sacrificing much performance, it achieves the lowest EDP.

different policies. Figure 6(a) compares the delay. We define the delay as the execution time of each case normalized to that in the base scenario, and a shorter delay means better performance. Since *Fan+TEC* and *Fan-only* do not adjust the DVFS level, they have the same execution time as the base scenario. By coordinating all the available knobs and choosing the best actuation to enforce, *TECfan* has only 4% longer delay. Relying on throttling the processor to cut power dissipation to reduce temperature, *Fan+DVFS* has 60% longer delay. *DVFS+TEC* leverages some advantages of TEC's local cooling capability to reduce the engagement of throttling the processor, but still suffers from the inconsistency of DVFS and TEC. Therefore, *DVFS+TEC* has longer delay than *TECfan* but shorter delay than *Fan+DVFS*.

Figure 6(b) compares the power in each case. By using TECs for local cooling, *Fan+TEC* can run the fan at the 2nd speed level with the same cooling effect as in the baseline case (with the 1st fan speed level), but reduces the total power by 9% on average. Since *Fan+DVFS* lowers the DVFS level to cut the dynamic power and reduce heat dissipation, it significantly reduces the power by 57%. Compared to *Fan+DVFS*, *DVFS+TEC* reduces the employment of DVFS at the cost of TEC power. Therefore, *DVFS+TEC* consumes slightly higher power than *Fan+DVFS*. *TECfan* coordinates all the available knobs for optimized energy and its heuristic algorithm gives priority to performance. Therefore, *TECfan* rarely lowers the DVFS level and hence consumes higher power than *DVFS+TEC*.

Figure 6(c) presents the energy consumption of each case. We add all the products of power readings and time interval in the trace file of one execution as the total energy. The reported numbers in Figure 6(c) is normalized to the base scenario. Having the same execution time but a smaller cooling requirement, *Fan+TEC* reduce the energy by 9%. Due to the cubic dynamic power reduction of DVFS at linear performance degradation cost, *Fan+DVFS* buys more power saving than compromises performance, and achieves 34% energy saving. *DVFS+TEC* also adopts aggressive DVFS throttling and achieves 32% energy saving. Although *TECfan* gives priority to performance, it still saves 27% energy. To correct the energy bias of DVFS, we also present the energy delay product [38] results in Figure 6(d), and *TECfan* shows clear advantages. In contrast, the policies highly relying on DVFS lose their advantages, and *Fan+DVFS* even has a worse EDP than the base scenario.

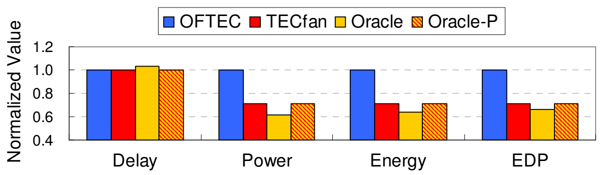


Figure 7. Comparison among *TECfan*, *OFTEC* and *Oracle*. *Oracle-P* is *Oracle* without performance degradation.

E. Comparison with OFTEC and Oracle

We now compare *TECfan* with two more baselines introduced in Section V-A, *OFTEC* and *Oracle*, with the 4-core simulation setup introduced in Section IV-B. The setup has a smaller scale because of the high time complexity of *OFTEC* and *Oracle*, because they both perform exhaustive search to find the optimal solution. Each simulation runs for 10 minutes, such that the impacts of fan speed on CMP temperatures are stabilized. We cut the first 40 minutes from the 7-day workload trace, divide it into four 10-minute pieces, and run each 10-minute trace in one core. The average CPU utilization is 48.6%.

Figure 7 presents the results normalized to *OFTEC* results. *TECfan* and *Oracle* achieve much lower power and energy consumption than *OFTEC*, because of using DVFS. *TECfan* can select appropriate DVFS levels to reduce 29% energy consumption without degrading the performance. *Oracle* saves energy more aggressively with the lowest DVFS levels and causes 3% longer delays on average. To fairly compare *TECfan* and *Oracle*, we add a constraint to the optimization problem, which allows *Oracle* to have the exactly same performance degradation with *TECfan*. We call this solution *Oracle-P*. As shown in Figure 7, the power, energy, and EDP results of *TECfan* are approximately the same with those of *Oracle-P*, which demonstrates that the CMP energy efficiency achieved by *TECfan* is almost close to the optimal. *OFTEC* has the smallest energy savings due to the fact it relies only on TEC and fan without adapting DVFS. It is important to note that both *Oracle* and *OFTEC* are impractical due to their high time complexity.

VI. Conclusions

As an emerging technology, TECs offer effective local cooling, which complements the global cooling of fans to improve the overall cooling energy efficiency. However, relying on only TEC and fan may not optimize the total energy consumption of a chip multiprocessor (CMP), because the CMP core power states impact both computing and cooling power consumption. In this paper, we have presented TECfan, a hierarchical framework that integrates TEC, fan, and DVFS for the overall energy efficiency of CMP. Specifically, we first formulate CMP energy optimization with temperature constraint as a nonlinear optimization problem. Since it requires a prohibitive long time to be solved online, a novel heuristic algorithm is designed to solve it with acceptable time overheads. Our extensive experiment results show that TECfan leads to 29% less energy consumption for medium workload compared to a state-of-the-art solution and 27%less overall compared to fan-based cooling.

ACKNOWLEDGEMENT

We would like to thank Prof. Radu Teodorescu at The Ohio State University for discussions in the early stage of this project. This work is funded, in part, by NSF under grants CCF-1143605, CNS-1218154, and CNS-1143607 (NSF CAREER Award).

REFERENCES

[1] H. Esmaeilzadeh, E. Blem, R. S. Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *ISCA*, 2011.

[2] D. Shin, S. W. Chung, E.-Y. Chung, and N. Chang, "Energy-optimal dynamic thermal management: Computation and cooling power co-optimization," *Industrial Informatics, IEEE Transactions on*, vol. 6, no. 3, 2010.

[3] M. Patterson, "The effect of data center temperature on energy efficiency," in *ITHERM*, 2008.

[4] R. Ayoub and T. Rosing, "Cool and save: Cooling aware dynamic workload scheduling in multi-socket cpu systems," in *ASP-DAC*, 2010.

[5] P. Chaparro, J. González, Q. Cai, and G. Chrysler, "Dynamic thermal management using thin-film thermoelectric cooling," in *ISLPED*, 2009.

[6] I. Chowdhury, R. Prasher, K. Lofgreen, G. Chrysler, S. Narasimhan, R. Mahajan, D. Koester, R. Alley, and R. Venkatasubramanian, "On-chip cooling by superlattice-based thin-film thermoelectrics," *Nature NanoTech.*, vol. 4, 2009.

[7] S. Biswas, M. Tiwari, T. Sherwood, L. Theogarajan, and F. T. Chong, "Fighting fire with fire: Modeling the data center scale effects of targeted superlattice thermal management," in *ISCA*, 2011.

[8] M. J. Dousti and M. Pedram, "Power-aware deployment and control of forced-convection and thermoelectric coolers," in *DAC*, 2014.

[9] M. P. Gupta, M.-H. Sayer, S. Mukhopadhyay, and S. Kumar, "Ultrathin thermoelectric devices for on-chip peltier cooling," *Components, Packaging and Manufacturing Technology, IEEE Transactions on*, vol. 1, no. 9, 2011.

[10] J. Long and S. O. Memik, "A framework for optimizing thermoelectric active cooling systems," in *DAC*, 2010.

[11] M. Ikeda, T. Nakamura, Y. Kimura, H. Noda, I. Sauciuc, and H. Erturk, "Thermal performance of thermoelectric cooler (tec) integrated heat sink and optimizing structure for low acoustic noise / power consumption," in *Semiconductor Thermal Measurement and Management Symposium*, 2006.

[12] S. Rho, K. Kang, and C.-M. Kyung, "Energy minimization of 3d cache-stacked processor based on thin-film thermoelectric coolers," in *MWSCAS*, 2011.

[13] M. J. Dousti and M. Pedram, "Platform-dependent, leakage-aware control of the driving current of embedded thermoelectric coolers," in *ISLPED*, 2013.

[14] ——, "Power-efficient control of thermoelectric coolers considering distributed hot spots," in *DATE*, 2015.

[15] S. Jayakumar and S. Reda, "Making sense of thermoelectrics for processor thermal management and energy harvesting," in *ISLPED*, 2015.

[16] I. Paul, S. Manne, M. Arora, L. Bircher, and S. Yalamanchili, "Cooperative boosting: Needy versus greedy power management," in *ISCA*, 2013.

[17] J. Mulder and P. Dillon, "Through silicon via technology status," in *ETW*, 2012.

[18] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, "Hotspot: Thermal modeling for CMOS VLSI systems," *TCPM*, 2005.

[19] Dynatron, "Dynatron new product SPEC sheet R16," http://www.dynatron-corp.com.

[20] K. Skadron, T. Abdelzaher, and M. R. Stan, "Control-theoretic techniques and thermal-RC modeling for accurate and localized dynamic thermal management," in *HPCA*, 2002.

[21] H. Su, F. Liu, A. Devgan, E. Acar, and S. Nassif, "Full chip leakage-estimation considering power supply and temperature variations," in *ISLPED*, 2003.

[22] M. D. Powell, A. Biswas, J. S. Emer, S. S. Mukherjee, B. R. Sheikh, and S. Yardi, "Camp: A technique to estimate per-structure power at run-time using a few simple parameters," in *HPCA*, 2009.

[23] W. Kim, D. Brooks, and G.-Y. Wei, "A fully-integrated 3-level dc/dc converter for nanosecond-scale dvfs," *JSSC*, 2012.

[24] G. Yan, M. Seeman, S. Sanders, V. Sathe, S. Naffziger, and E. Alon, "Agileregulator: A hybrid voltage regulator scheme redeeming dark silicon for power efficiency in a multicore architecture," in *HPCA*, 2012.

[25] E. I. Milovanović, M. P. Bekakos, and I. Ž. Milovanović, "Synthesis of space optimal systolic arrays for band matrix-vector multiplication," *JC*, 2008.

[26] R. Bitirgen, E. Ipek, and J. Martinez, "Coordinated management of multiple interacting resources in chip multiprocessors: A machine learning approach," in *MICRO*, 2008.

[27] B. Curran, B. McCredie, L. Sigal, E. Schwarz, B. Fleischer, Y.-H. Chan, D. Webber, M. Vaden, and A. Goyal, "4GHz+ low-latency fixed-point and binary floating-point execution units for the power6 processor," in *ISSCC*, 2006.

[28] J. Howard, S. Dighe, S. Vangal, G. Ruhl, N. Borkar, and S. Jain, "A 48-core ia-32 processor in 45 nm cmos using on-die message-passing and dvfs for performance and power scaling," *JSSC*, vol. 46, no. 1, 2011.

[29] J. Sun, M. Xu, D. Reusch, and F. Lee, "High efficiency quasi-parallel voltage regulators," in *APEC*, 2008.

[30] J. Renau, "SESC simulator," January 2005, http://sesc.sourceforge.net.

[31] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural-level power analysis and optimizations," in *ISCA*, 2000.

[32] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, "CACTI 6.0: A tool to model large caches," HP Laboratories, Tech. Rep., 2009.

[33] G. Urdaneta, G. Pierre, and M. Steen, "Wikipedia workload analysis for decentralized hosting," *Elsevier Computer Networks*, 2009.

[34] T. Horvath and K. Skadron, "Multi-mode energy management for multi-tier server clusters," in *PACT*, 2008.

[35] "Intel Core i7-3770K," http://ark.intel.com/products/65523.

[36] X. Zhang, K. Shen, S. Dwarkadas, and R. Zhong, "An evaluation of per-chip nonuniform frequency scaling on multicores," in *USENIX ATC*, 2010.

[37] P. Singh, R. Sankar, X. Hu, W. Xie, A. Sarkar, and T. Thomas, "Power delivery network design and optimization for 3d stacked die designs," in *3DIC*, 2010.

[38] R. Gonzalez and M. Horowitz, "Energy dissipation in general purpose microprocessors," *JSSC*, vol. 31, 1996.