Thermal vs Energy Optimization for DVFS-enabled Processors in Embedded Systems

Yongpan Liu* Huazhong Yang* Robert P. Dick† Hui Wang * Li Shang‡
ypliu99@mails.tsinghua.edu.cn dickrp@eecs.northwestern.edu lshang@ee.queensu.ca
* Electronic Engineering Dept. Tsinghua University Beijing, 100084, China
† EECS Dept. Northwestern University Evanston, IL 60208, U.S.A.
‡ ECE Dept. Queen's University Kingston, ON K7L 3N6, Canada

***Abstract*— In the past, dynamic voltage and frequency scaling (DVFS) has been widely used for power and energy optimization in embedded system design. As thermal issues become increasingly prominent, we propose design-time thermal optimization techniques for embedded systems. By carefully planning DVFS at design time, our techniques proactively optimize system thermal profile, prevent run-time thermal emergencies, minimize cooling costs, and optimize system performance. To the best of our knowledge, this is the first work addressing embedded system design-time thermal optimization using DVFS. We formulate minimization of application peak temperature in the presence of real-time constraints as a nonlinear programming problem. This provides a powerful framework for system designers to determine a proper thermal solution and provide a lower bound on the minimum temperature achievable by DVFS. Furthermore, we examine the differences between optimal energy solutions and optimal peak temperature solutions. Experimental results indicate that optimizing energy consumption can lead to unnecessarily high temperature. Finally, we propose a thermal-constrained energy optimization procedure to minimize system energy consumption under a constraint on peak temperature.**

I. Introduction

Multiprocessor or multi-core architectures are popular in complex embedded systems, which range from mobile consumer electronics to high-performance game consoles. With the technology evolution, the demand for increased performance and reduced size leads to increasing power density and temperature [1]. Chip temperature has significant impact on performance, reliability, power consumption, as well as cooling and packaging costs. Thermal-aware design is difficult. Designing a chip and package for the worst-case power consumption scenario may be prohibitively expensive. Therefore, most real cooling solutions are designed for the Thermal Design Power (TDP), which is usually less than 15% of the worst case power consumption [2]. Thanks to material heat capacity, the processor can safely consume more power than the TDP for a brief period of time. However, if the TDP is exceeded for an extended period of time, the chip temperature may reach a dangerous level, triggering a sensor-driven hardware mechanism to reduce power consumption. These techniques are known as Dynamic Thermal Management (DTM), which mainly consists of fetch toggling [3], DVFS [4], and activity migration [5]. In the future, the discrepancy between TDP and the worst case value will increase and DTM will be widely used.

Although DTM techniques can bound chip temperature at the cost of some run-time performance degradation, there is not yet a clear way to choose a proper TDP for an embedded system at design time, even though application execution patterns and real-time constraints are already known. The TDP of a DVFS-enabled processor for worst-case general-purpose applications may be too pessimistic for a specific embedded system design, leading to unnecessarily-high cooling costs. If the thermal solution is fixed, design-time thermal optimization can be used to reduce the operating temperature. Most failure processes, e.g., electro-migration, thermal cycling, time dependent dielectric breakdown, and stress migration depend exponentially on temperature [6]. Hence, embedded system reliability can also benefit from reduced operating temperature. Run-time DTM and design time optimization techniques should be combined to optimize embedded system performance and reliability.

This work was supported in part by the NSFC under awards 90207001 and 60506010; in part by the 863 Program under award 2006AA01Z224; in part by the NSF under award CNS-0347941; and in part by NSERC Discovery Grant 388694-01.

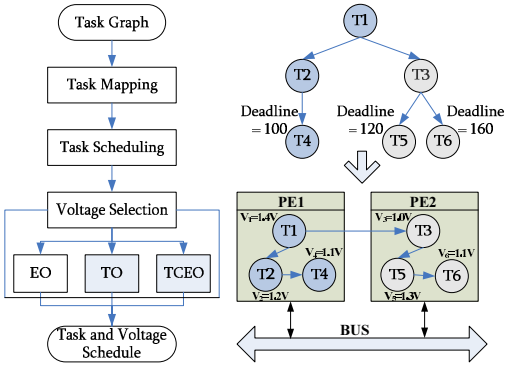

Fig. 1. Voltage selection formulation.

There is a large body of work on using DVFS in single and multiple processor systems to minimize energy consumption [7]–[9]. However, previous work adopted temperature-independent power models, which will result in large leakage energy estimation errors in future deep submicron processors. Minimizing energy is the primary objective in previous work. Several temperature-dependent leakage power and thermal modeling approaches have been proposed at the micro-architecture level [10], [11]. However, neither thermal optimization issues nor voltage selection under real-time constraints were considered. Recently, Hung et al. proposed a thermal-aware embedded system synthesis framework by task mapping and scheduling [12]. Paci et al. demonstrated that temperature-aware design is not critical for ultra low power (less than 3 W) multiprocessor systems-on-a-chip [13] although it is important for systems with high power consumptions.

We consider the problem of task voltage selection under real-time constraints with a number of optimizing objectives: energy optimization (EO), thermal optimization (TO), and thermal constrained energy optimization (TCEO). Our optimization framework is shown

Proceedings of the 8th International Symposium on Quality Electronic Design (ISQED'07)
0-7695-2795-7/07 $20.00 © 2007 IEEE

in Figure 1. To our best knowledge, this is the first article to present a design-time optimization technique for real-time embedded systems that makes use of dynamic voltage and frequency scaling (DVFS) to minimize peak temperature. It should be noted that our method is general and can be used for high-performance as well as low-power systems. The problem can be solved by nonlinear programming [14]. The solution to this problem provides a lower bound on the minimum peak temperature that can be achieved by DVFS. Hence, it provides a reference for designers when determining a temperature constraint for use in optimization, e.g., temperature constrained energy minimization. We compare the proposed method with traditional energy optimization method and show that the results differ, i.e., the optimal energy solution does not generally have optimal temperature. However, the optimal temperature solution can be achieved with little energy overhead. When both energy and thermal metrics are considered in specific embedded systems, designers must choose a tradeoff between them. Finally, we present a temperature-constrained energy optimization formulation that may be used to avoid unnecessarily high temperature resulting from traditional energy optimizing procedures.

The paper is organized as follows. First, thermal-conscious system-level models are described in Section II. Thermal, energy, and temperature-constrained energy optimizing problems are formulated in Section III. Finally, experimental results are reported in Section IV.

II. Modeling Methodology

This section presents a modeling methodology for system-level energy and temperature analysis.

II.A. System Model

Multiprocessor system models can be placed in two categories: hardware and software. In this paper, hardware models specify multiprocessor systems consisting of various processing elements (PEs) such as microprocessors, DSPs, field programmable gate arrays (FPGAs), and application-specific integrated circuits (ASICs). PEs may support DVFS and are typically connected by communication units (CUs). When PEs exchange data, the CU in use consumes energy and introduces delay. We denote the set of PE with $\mathbf{\Psi}$ and the set of CUs with $\mathbf{\Gamma}$. We assume that a PE consists of m functional units, $FU_j,\ j \in 1, 2, \cdots, m$. The switched capacitance and leakage current for a functional unit in the presence of a particular input pattern, voltage, and temperature are c_{fu_j} and i_{fu_j}. $\forall P_i \in \mathbf{\Psi}$, six PE-related parameters are known:

- $f_{\max}$ is the peak operating frequency,
- $V_{\max}$ is the maximal operating voltage,
- $V_{\min}$ is the minimal operating voltage,
- $\mathbf{C}_{PE} = [c_{fu_1}, \cdots, c_{fu_m}]^{\mathrm{T}}$ is the capacitance vector,
- $\mathbf{I}_{PE} = [i_{fu_1}, \cdots, i_{fu_m}]^{\mathrm{T}}$ is the leakage current vector, and
- $\mathbf{R}_{PE}$ is the thermal resistance matrix.

In software models, real-time applications on multiprocessor systems are often represented by task graphs, $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, which are directed acyclic graphs composed of sets of N vertices $\mathbf{V} = (v_1, v_2, \ldots v_N)$ and sets of edges, $\mathbf{E} = \{(v_i, v_j) | \text{an edge exists between } v_i \text{ and } v_j\}$. As Figure 1 shows, each vertex, v_i, denotes a specific task, and each edge, (v_i, v_j), represents a precedence relationship between two tasks. A deadline, dl^i, is associated with each leaf node, v_i. We denote the set or all leaf nodes as $\mathbf{\Delta}$. The communication between tasks is modeled as a special task executed by a CU. Communication volumes are provided by the system-level synthesis framework. It should be noted that another type of precedence relationship may be introduced by task mapping and scheduling. Those constraints are recorded in the edge set, $\mathbf{E}'$:

$$\mathbf{E}' = \left\{ (v_i, v_j) \;\middle|\; \begin{array}{l} v_i \text{ and } v_j \text{ use the same PE or} \\ v_i \text{ and } v_j \text{ use the same CU} \end{array} \right\} \quad (1)$$

The set of all edges is denoted with $\mathbf{\Upsilon} = \mathbf{E} \cup \mathbf{E}'$. A functional unit's switching probability for a given task is θ_{fu_j}, which indicates the average percentage of transistors switching per cycle. The effect of the input vector effect on leakage current is represented by ω_{fu_j}. $\forall v_i \in \mathbf{V}$, three technology parameters are defined:

- EC is the task duration in cycles,
- $\mathbf{\Theta} = [\theta_{fu_1}, \cdots, \theta_{fu_m}]^{\mathrm{T}}$ is the switching probability vector, and
- $\mathbf{\Omega} = [\omega_{fu_1}, \cdots, \omega_{fu_m}]^{\mathrm{T}}$ is the leakage factor vector.

TABLE I
Leakage Model Parameters for Logic and Memory Circuits

Benchmark	A ($\times 10^{-4}$)	α	β	$-\gamma$	B ($\times 10^{-4}$)	μ
c5315	5.406	1127.0	1669.5	2223.7	6.597	5.691
c6288	5.447	1122.7	1670.1	2222.9	6.770	5.692
c7552	5.467	1122.5	1671.0	2223.8	6.769	5.692
16Kx32	2.867	1177.4	1593.1	2162.7	20.037	5.687
64Kx32	2.835	1177.6	1592.0	2161.5	20.325	5.687
2Mx32	2.824	1177.7	1591.6	2161.2	20.422	5.687

II.B. Power and Delay Model

In this section we derive the temperature dependent power model for DVFS-enabled PEs. The functional unit FU_j's dynamic power consumption, p_{dfu_j}, can be calculated using the following formula [7]:

$$p_{dfu_j} = \theta_{fu_j} c_{fu_j} V_{dd}^2 f \quad (2)$$

where f is the processor operating frequency and V_{dd} is the supply voltage.

In the near future, subthreshold leakage and gate leakage will be the dominant types of leakage current [1]. The fundamental leakage current formulas for CMOS devices [10], [15], [16] can be used to derive an expression for functional unit leakage power:

$$p_{lfu_j} = V_{dd} \omega_{fu_j} i_{fu_j} (AT^2 e^{\frac{\alpha V_{dd} + \beta V_{bs} + \gamma}{T}} + B e^{\mu V_{dd}}) \quad (3)$$

where $A, B, \alpha, \beta, \gamma$, and μ are curve-fitting constants that depend on circuit type, process, and design. Using HSPICE, we simulate leakage currents for combinational logic circuits [17] and SRAM [18] benchmarks under different supply voltages, bias voltages, and temperatures in order to extract the leakage constants, which are shown in Table I. Temperature ranges from 25 °C to 110 °C; supply voltage ranges from 0.9 V to 1.4 V; and body bias voltage ranges from 0.0 V to -0.4 V. Using Equation 3, the average and worst-case leakage modeling errors are 1.3% and 6%, respectively, i.e., the leakage power for each functional unit can be accurately estimated. Experimental results show that leakage constants for different circuits synthesized using the same standard-cell library are quite similar. Therefore, we can use general constants to predict leakage for a particular library and process.

The dependence of a circuit's delay t, and thus operating frequency

f, upon supply voltage is determined using the following formula [7]:

$$f^{-1} = t = \frac{K}{(V_{dd} - V_{th})^{\sigma}} \tag{4}$$

where the constant K is decided by logic depth and process and σ is a measure of velocity saturation.

II.C. Thermal Model

According to heat transfer theory, heat flow can be modeled as follows:

$$\mathbf{C}\frac{\mathrm{d}\mathbf{T}(t)}{\mathrm{d}t} = \mathbf{A}\mathbf{T}(t) - \mathbf{p}\mathbf{U}(t) \tag{5}$$

where

- $\mathbf{C}$ is an $n \times n$ diagonal thermal capacitance matrix,
- $\mathbf{A}$ is an $n \times n$ thermal conductance matrix,
- $\mathbf{T}(t) = [T_1 - T_A, T_2 - T_A, \cdots, T_n - T_A]^{\mathrm{T}}$ is the temperature vector in which T_A is the ambient temperature,
- $\mathbf{p} = [p_1, p_2, \cdots, p_n]^{\mathrm{T}}$ is the power vector, and
- $\mathbf{U}(t)$ is a step function.

Note that solving this equation is analogous to RC circuit analysis. In steady-state thermal analysis, one estimates the thermal profile as time proceeds to infinity. Therefore, we can denote $\lim_{t\to\inf} \mathbf{T}(t)$ as $\mathbf{T}$, allowing Equation 5 to be simplified as follows:

$$\mathbf{p} = \mathbf{A} \times \mathbf{T} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \times \mathbf{T}$$

We denote the inversion of thermal conductance matrix $\mathbf{A}$ as $\mathbf{R}$, the thermal resistance matrix. In multiprocessor systems, each PE may have a different cooling configuration and there are no direct heat transfer channels among PEs. The thermal resistance matrix $\mathbf{R}_{PE}$ for each PE can be extracted automatically [19], [20]. In system-level optimization, estimation accuracy and speed are both essential. In the proposed approach, one thermal element is used for each functional unit. Therefore, it is straightforward to obtain the power vector $\mathbf{p} = [(p_{lfu_1} + p_{dfu_1}), \cdots, (p_{lfu_n} + p_{dfu_n})]^T$. However, if more accurate thermal profiles are needed, fine-grained thermal analysis can be used [19], [20].

II.D. Iterative Modeling

Given initial dynamic and static power consumptions, each processor converges to a steady-state temperature under specific cooling and leakage model conditions. The temperature of a PE varies greatly depending on the currently-executing task. Therefore, traditional approaches lead to large estimation errors, which may result in sub-optimal voltage selection. We propose an accurate iterative algorithm to calculate the steady-state temperature and power consumption. Lines 2–11 of Algorithm 1 show the iterative power and temperature calculation algorithm. Lines 3–6 calculate the dynamic and leakage power for each functional unit. Lines 7–10 do thermal analysis and update the steady-state temperature vector. The constant, η, is a user-defined integer specifying iteration count. In our experiments, the processor power vector $\mathbf{p}^{\eta}$ and steady-state temperature vector $\mathbf{T}^{\eta}$ converged with less than 0.1% iteration-to-iteration variation after four iterations.

Algorithm 1 power_thermal_iter($\mathbf{p}^0$, $\mathbf{T}^0$)

1: Set initial temperature vector $\mathbf{T}^0$
2: **for** $j = 0$ to η **do**
3: **for** each functional unit FU_n in a PE **do**
4: $p^j_{dfu_n} = calculate\_dyn(\theta_{fu_n}, c_{fu_n}, V_{dd})$
5: $p^j_{lfu_n} = calculate\_leak(\omega_{fu_n}, i_{fu_n}, V_{dd}, V_{bs}, T^j_{fu_n})$
6: **end for**
7: **for** each functional unit FU_n in a PE **do**
8: $T^{j+1}_{fu_n} = calculate\_temp(\mathbf{p}^j, \mathbf{R}_{PE})$
9: Update temperature for functional unit FU_n
10: **end for**
11: **end for**
12: Output temperature $\mathbf{T}^{\eta}$ and power consumption $\mathbf{p}^{\eta}$

III. Formulation

This section describes three voltage selection formulations based on thermal-conscious system models.

III.A. Problem Formulation

We have developed a system-level synthesis infrastructure [21] and use it to generate assignments and schedules. In this section we will focus on the voltage selection problem. The supply voltage vector, $\mathbf{v}$, for the task set should be optimized. Hence, the voltage selection problem can be expressed as follows:

$$\begin{array}{ll} \text{Minimize} & E_{tot}(\mathbf{v}) \text{ or } T_{\max}(\mathbf{v}) \\ \text{subject to} & f_{power,delay,temp}(\mathbf{v}) = 0 \\ \text{and} & g_{timing,temp}(\mathbf{v}) \leq 0 \end{array}$$

The objective is to minimize either total energy, $E_{tot}(\mathbf{v})$, or maximum temperature, $T_{\max}(\mathbf{v})$, during the execution of a task set on the embedded system, subject to constraints on other functions of voltages $\mathbf{v}$. The non-linear equality constraints are derived from the power, thermal, and delay models in Section II. The inequality constraints describe the linear real-time constraints and an optional temperature constraint. Therefore, the problem may be formulated as a nonlinear programming problem and solved by either interior-point or active-set methods [14].

III.B. Thermal Optimization (TO)

The formulation of the temperature optimization problem under real-time constraints follows:

$$\begin{array}{ll} \text{Minimize} & \max(T^{i(\eta)}) \\ \text{c1} & exec^i = EC^i/f \\ \text{c2} & \mathbf{p}^{\eta}, \mathbf{T}^{\eta} = power\_thermal\_iter(\mathbf{p}^0, \mathbf{T}^0) \\ \text{c3} & start^i + exec^i \leq start^j,\ \exists (v_i, v_j) \in \boldsymbol{\Upsilon} \\ \text{c4} & start^i + exec^i \leq dl^i,\ \exists v_i \in \boldsymbol{\Delta} \\ \text{c5} & start^i \geq 0 \\ \text{c6} & V^{min}_{dd} \leq V^i_{dd} \leq V^{max}_{dd} \end{array}$$

The optimization variables for this problem are the task execution times $exec^i$, the task start times $start^i$, and the operating voltages V^i_{dd}. The optimization objective is to minimize the maximum temperature of all tasks. For each task, constraint *c1* is based on the delay model, which indicates the relationship between execution time and supply voltage. Constraint *c2* describes the iterative relationship between power and temperature. It is determined with Algorithm 1. Constraint *c3* gives the task set's precedence relationships, which are introduced by data dependencies and resource conflicts. Constraint

c4 enforces task deadlines. Constraint *c5* and *c6* bound the ranges of start times and voltages.

III.C. Energy Optimization (EO)

In this problem variant, the goal is to minimize the total execution energy of all tasks by controlling processor voltages subject to the same constraints (c1–c6) described in the previous section.

Minimize	$\sum_{i=1}^{\|V\|} E_{tot}^{i(\eta)}$
c1–c6	same as TO in Section III-B

III.D. Thermal-constrained Energy Optimization (TCEO)

We formulate the temperature-constrained energy optimization problem under real-time constraints as follows:

Minimize	$\sum_{i=1}^{\|V\|} E_{tot}^{i(\eta)}$
c1-c6	same as TO in Section III-B
c7	$\mathbf{T}^{\eta} \leq \mathbf{T}_{thres}$

In this formulation, temperature is used as constraint *c7* in the nonlinear programming procedure thereby permitting energy consumption to be minimized while guaranteeing that the threshold temperature is not violated.

IV. Experiments

In this section, we first explain our experimental setups and then compare the results of embedded system energy optimization and thermal optimization. Finally, results for temperature-constrained energy optimization are reported.

IV.A. Experimental Setup

A system-level synthesis framework [21] was developed for task mapping and scheduling of embedded systems. In this work, the maximum operating voltage and frequency are used. Starting from these performance-optimized solutions, our nonlinear programming procedures choose appropriate voltage settings for each task to optimize energy or temperature metrics under real-time constraints. The software model is represented by a task graph, which indicates the data dependencies and real-time constraints of tasks. We use TGFF [22] to generate task graph sets. All benchmarks are solved on a 1.4 Ghz Centrino™laptop with 768 MB RAM running Linux. In our experiments, the largest problem can be solved in less than 600 s.

Processor dynamic and leakage power values are taken from a product datasheet [23]. The technology constants for power models are extracted based on 65 nm predictive technology model (PTM) [24]. The leakage ratio is approximately 0.3 for a 65 nm process [7]. Different cooling conditions are modeled by the corresponding thermal resistances [25], which range from 0.4 °C/W to 1.4 °C/W. The thermal resistance matrix, $\mathbf{R}_{PE}$, is extracted using thermal analysis tools [19]. According to our HSPICE simulation, changes in input patterns may result in up to $3\times$ change to leakage current. Therefore, we set switching probability θ_{fu_j} and leakage impact factor ω_{fu_j} for random benchmark to real numbers having uniform distributions in the ranges $(0, 1)$ and $(0, 3)$, respectively. It should be noted that our formulation is general and independent of the processor model described in Table II. The processor model used here is most appropriate for high power density cases, for which thermal-aware design is important.

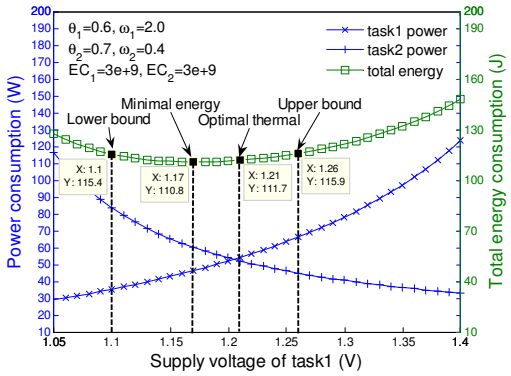

Fig. 2. Energy and power of task1 and task2 as functions of supply voltage.

TABLE II
Experimental Setup

Variable	Value	Variable	Value	Variable	Value
Voltage	0.9–1.4 (V)	C_{eff}	15 (nF)	K	0.344×10^{-9}
V_{th}	0.244 (V)	f_{max}	3.46 (Ghz)	i_{PE}	2.599 (A)
σ	1.2	R	0.8 (°C/W)	DL	2 (s)
θ	0.6/0.7	ω	2.0/0.4	EC	3/3 $\times 10^9$

IV.B. Energy vs Temperature Optimization

In this section, we will first show that the results of energy optimization (EO) and temperature optimization (TO) are inconsistent using a two-task benchmark. Random large-scale benchmarks are optimally solved to show the generality of the proposed technique.

Considering a special benchmark consisted of two tasks, task1 and task2, executed in sequence. Task2 has deadline DL. Table II indicates the execution cycles EC, switching probability θ, and leakage factor ω for each task. The energy and power consumption of each task is given in the following formulas, which are used iteratively until convergence.

$$E(V) = P(V) \times EC/f \tag{6}$$

$$P(V) = \theta C_{eff} V^2 f + V\omega i(AT^2 e^{\frac{\alpha V + \beta V_{bs} + \gamma}{T}} + Be^{\mu V}) \tag{7}$$

Based on the real-time deadline, we can obtain the optimal voltage relationship between task1 and task2.

$$\frac{EC_1 K}{(V_1 - V_{th})^{\sigma}} + \frac{EC_2 K}{(V_2 - V_{th})^{\sigma}} = DL \tag{8}$$

Formulas 6–8 are used to calculate the total energy and power curves of task1 and task2 as functions of the supply voltage of task1 v_1, as shown in Figure 2. The total energy curve is fairly flat for task1 supply voltages ranging from 1.1 V to 1.26 V, the lower and upper voltage bounds shown in the figure. In this range, the total energy is within 5% of optimality. In Table III, energy consumption, peak temperature, peak power, and energy overhead are listed for the voltages in this range.

As we can see from Figure 2 and Table III the voltages resulting in minimal energy and minimal peak temperature differ. Explicitly optimizing peak temperature reduced it by 6 °C with 0.8% energy overhead.

Figure 3 illustrates temperature inconsistency and energy overhead as functions of thermal resistance between the silicon active layer and the ambient environment. The inconsistencies increase with resistance to ambient. The largest temperature difference (up to 15 °C)

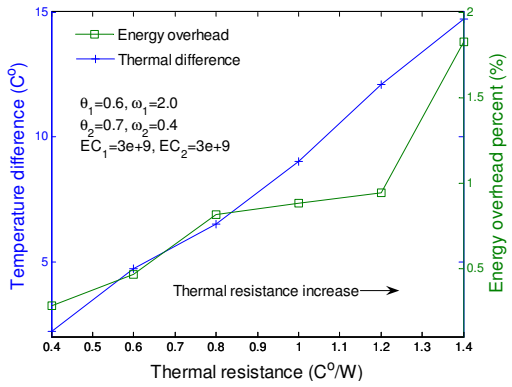

Fig. 3. Energy and thermal optimization inconsistency as a function of thermal resistance to ambient.

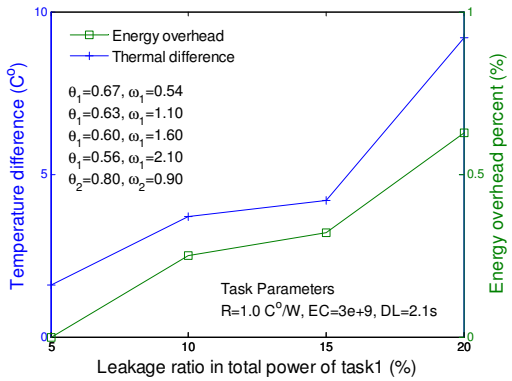

Fig. 4. Energy and thermal optimization inconsistency as functions of leakage ratio.

TABLE III
EXPERIMENTAL RESULTS

Optimization objective	Energy (J)	T_{peak} (K)	P_{peak} (W)	Overhead (%)
Thermal optimal	111.7	355	52.1	0.8
Energy minimal	110.8	361	60.2	0.0
Lower bound	115.4	380	83.5	4.2
Upper bound	115.9	366	66.2	4.6

is observed in the poorest cooling conditions (1.4 °C/W) with 1.8% energy overhead; the same power difference between optimal peak temperature and optimal energy voltage settings can result in higher temperature differences under poor cooling conditions.

Figure 4 illustrates the relationship between inconsistency in temperature and energy optimization and task power profile. The parameters of Task2 are held constant and the parameters of task1 are adjusted to change relative contribution of leakage to the total power consumption. In the high leakage case, the largest temperature difference (up to 9.2 °C) is observed for the case with 0.6% energy overhead. This can be explained by the fact that higher leakage makes the power curve change more dramatically as a function of supply voltage due to stronger power–temperature dependence. Therefore, the peak power difference and temperature difference become larger in the high leakage ratio case.

Table III indicates that a voltage assignment consuming nearly optimal energy can lead to an unnecessarily-high peak temperature. For example, the lower bound case can result in up to a 25 °C increase in temperature. However, the energy overhead for that case is only 4.2%. It is well known that the optimal energy-efficient voltage selection problem in real processors with discrete voltage levels is NP-hard [9]. Therefore, heuristics are used. However, existing techniques only optimize energy consumption without considering temperature. Therefore, they can produce solutions with poor thermal characteristics, especially for embedded systems with high power density or poor cooling.

The two-task benchmark demonstrates that the results of thermal optimization and energy optimization are inconsistent and these inconsistencies depend on both the thermal resistance to the ambient and the leakage ratio. We will now determine whether these observations hold for large benchmarks. Each graph contains between 2 to 66 tasks, which are mapped on a dual-processor platform. The processor model and task parameters are decided based on the method described in Section IV-A.

TABLE IV
ENERGY CONSUMPTION UNDER DIFFERENT VS APPROACHES NODES

Benchmark	Energy EO (J)	Energy TO (J)	Energy Overhead (%)	T_{max} EO (K)	T_{max} TO (K)	T_{max} Reduction (K)
TG1	408	416	2.0	389	375	14
TG2	243655	267737	9.0	365	361	4
TG3	279389	311949	10.4	398	390	8
TG4	747530	845558	11.6	397	379	18.0
Average			8.3			11

As indicated in Table IV, there is a 4–18 °C temperature difference and a 2%–11.6% energy difference between optimal energy and optimal temperature solutions; our observations for the two-task case hold for these large benchmarks. The variation in difference among benchmarks is due to the randomly-selected task parameters, which affect the total power consumption and leakage ratio.

We can conclude that optimizing energy without temperature constraints may result in unnecessarily high temperatures, cooling costs, and reliability problems. Peak temperature should be explicitly constrained or optimized when cooling cost or temperature are important. The simultaneous optimization of energy and temperature based on a temperature-aware power model will be increasingly useful in the future as power density increases.

IV.C. Tradeoff Between Energy and Temperature

In this section, we describe the results of our temperature-constrained energy optimization method. When temperature-constrained energy optimization is used, it is important to choose an appropriate temperature bound. If the bound is too loose, the system may operate at an unnecessary-high temperature. If it is too tight, it may not be possible to find a feasible solution or the energy consumption might be increased. In Figure 5 the results of optimizing energy, optimizing temperature, and optimizing energy under temperature constraints for benchmark TG1 are shown. The minimal energy point is determined by the energy optimization procedure. This provides a lower-bound on the energy consumption for use in temperature-constrained energy optimization. The lower bound on temperature point is determined by the temperature optimization procedure. Different solutions between the optimal energy and optimal temperature solutions are produced by providing different temperature bounds to the temperature-constrained energy optimization procedure. If 381 K is the temperature threshold, the optimal energy value 409.1 J, which implies a 8 °C temperature reduction and 0.27% energy overhead

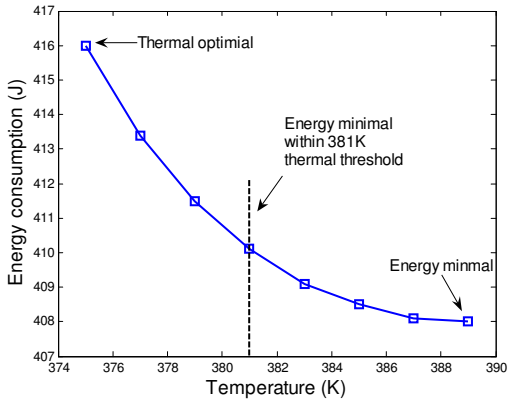

Fig. 5. Energy vs. peak temperature tradeoffs for the benchmark TG1.

compared with the minimal energy point. Energy increases when peak temperature is reduced. Therefore, the energy optimization and peak temperature optimization results provide useful bounds on the tradeoffs available between energy and temperature.

V. Conclusions

Design-time thermal optimization is an effective method of reducing the cooling costs and improving the reliability of embedded systems. This paper has proposed a novel design-time thermal optimization framework based on a temperature-aware power model. Experimental results show that the results of the proposed technique are inconsistent with traditional energy optimization. We observed average 11 °C temperature reduction with 8.3% energy overhead. This underscores the importance of integrating temperature constraints into energy optimization algorithms. Finally, we proposed a design-time procedure to optimize energy under a constraint on peak temperature.

References

[1] "International Technology Roadmap for Semiconductors," 2005, http:// public. itrs. net.

[2] "Thermal design power," http://www.cpuheat.wz.cz/html/ IntelPowerConsumption.htm.

[3] D. Brooks and M. Martonosi, "Dynamic thermal management for high-performance microprocessors," in *Proc. Int. Symp. on High Performance Computer Architecture*, Jan. 2001, pp. 171–182.

[4] K. Skadron, "Hybrid architectural dynamic thermal management," in *Proc. Design, Automation & Test in Europe Conf.*, mar. 2004.

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

[6] J. Srinivasan, et al., "Exploiting structural duplication for lifetime reliability enhancement," in *Proc. Int. Symp. Computer Architecture*, June 2005, pp. 520–531.

[7] S. M. Martin, et al., "Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads," in *Proc. Int. Conf. Computer-Aided Design*, Nov. 2002, pp. 721–725.

[8] R. Jejurikar, C. Pereira, and R. Gupta, "Leakage aware dynamic voltage scaling for real-time embedded systems," in *Proc. Design Automation Conf.*, June 2004, pp. 275–280.

[9] A. Andrei, et al., "Overhead-conscious voltage selection for dynamic and leakage energy reduction of time-constrained systems," in *IEE Proceedings*, Jan. 2005, pp. 28–38.

[10] W. P. Liao, L. He, and K. M. Lepak, "Temperature and supply voltage aware performance and power modeling at microarchitecture level," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 7, pp. 1042–1053, July 2005.

[11] Y. Zhang, et al., "HotLeakage: A temperature-aware model of subthreshold and gate leakage for architects," Univ. of Virginia, Tech. Rep., May 2003, CS-2003-05.

[12] W.-L. Hung, et al., "Thermal-aware task allocation and scheduling for embedded systems," in *Proc. Design, Automation & Test in Europe Conf.*, mar. 2005.

[13] G. Paci, et al., "Exploring "temperature-aware design" in low-power mpsocs," in *Proc. Design, Automation & Test in Europe Conf.*, Mar. 2006.

[14] W. L. Winston and M. Venkataramanan, *Introduction to Mathematical Programming: Applications and Algorithms*. 4 edition, Duxbury Press, 2002.

[15] A. Chandrakasan, W. Bowhill, and F. Fox, *Design of High-Performance Microprocessor Circuits*. IEEE Press, 2001.

[16] K. M. Cao, et al., "BSIM4 gate leakage model including source-drain partition," in *IEDM Technology Dig.*, Dec. 2000, pp. 815–818.

[17] "ISCAS85 benchmarks suite," http:// www. visc. vt. edu/ mh-siao/ iscas85. html.

[18] F. Zhang, "System-level leakage power modeling methodology," Dept. of Electronics Engg., Tsinghua University," Bachelor's Degree Thesis, July 2006.

[19] K. Skadron, et al., "Temperature-aware microarchitecture," in *Proc. Int. Symp. Computer Architecture*, June 2003, pp. 2–13.

[20] Y. Yang, et al., "Adaptive chip-package thermal analysis for synthesis and design," in *Proc. Design, Automation, and Test in Europe*, Mar. 2006, pp. 844–849.

[21] Y. P. Liu, et al., "Combining genetic algorithms based task mapping and optimal voltage selection for energy efficient distributed system synthesis," in *Proc. Int. Conf. Communications, Circuits and Systems*, June 2006, pp. 2074–2078.

[22] R. P. Dick, D. L. Rhodes, and W. Wolf, "TGFF: Task graphs for free," in *Proc. Int. Wkshp. Hardware/Software Co-Design*, Mar. 1998, pp. 97–101.

[23] "Intel®pentium™processor extreme edition 955 on 65 nm process," 2006, datasheet.

[24] W. Zhao and Y. Cao, "New generation of predictive technology model for sub-45nm design exploration," in *Proc. Int. Symp. Quality of Electronic Design*, Mar. 2006, pp. 585–590.

[25] R. Viswanath, et al., "Thermal performance challenges from silicon to systems," Intel Corporation, Tech. Rep., 2000, Intel Technology Journal Q3.