Feasibility Analysis for Temperature-Constraint Hard Real-Time Periodic Tasks

Gang Quan, *Member, IEEE*, and Vivek Chaturvedi, *Student Member, IEEE*

***Abstract*—While the dynamic thermal management problem is closely related to the dynamic power management problem, it has its own distinct features. In this paper, we study the feasibility checking problem for real-time periodic task sets under the peak temperature constraint. We show that the traditional scheduling approach, i.e. to repeat the schedule that is feasible through the range of one hyperperiod, does not apply any more. We then present new necessary and sufficient conditions to check the feasibility of real-time schedules. We further incorporate the close relationship of leakage, temperature, and supply voltage into our feasibility analysis, and develop more elaborated feasibility conditions. Our experiments, based on technical parameters derived from a processor using the 65 nm IC technology, demonstrate the effectiveness of our feasibility conditions and, at the same time, highlight the fact that a power/thermal-aware computing technique becomes ineffective at the submicron scale if the inter dependency of leakage, temperature, and supply voltage is not properly addressed.**

***Index Terms*—Feasibility analysis, leakage, power aware, real-time systems, temperature, thermal management.**

I. Introduction

CATERING to society's rapidly growing appetite of computing power, the processor's performance is expected to continuously grow dramatically in the future [1]. According to Borkar *et al.* [2] from Intel, there are more than 10 billion transistors integrated into a single 300 mm^2 die today, and the number is growing rapidly toward 100 B by the middle of the next decade. The power consumed by these transistors is also tremendous, reaching 300 W in early next decade. The exponentially increased power consumption has posted significant challenges not only on how to provide sufficient power source to an electronic system but also on how to dissipate the heat generated by the system.

The thermal issues have become increasingly prominent as power consumption continues to grow. The high chip temperature not only increases package/cooling cost (estimated at 1–3 dollar per watt [3]) but also adversely affects the reliability and performance of the computing systems, and can even cause a system to fail catastrophically. Even if a processor is not totally failed at the high temperature, a small increase in the temperature (e.g., 10 °C) can result in significant reduction (50%) in its life span [4]. From the perspective of real-time systems, when the chip temperature exceeds certain limit, the self-protection controls in many new generation processors may be invoked automatically by reducing the performance and thus cause tasks to miss deadlines.

As a closely related problem, power management problem for real-time systems have been researched extensively in the past decade [5], crossing different abstraction levels and platforms. At first sight, since the higher power consumption usually compounds with the higher energy consumption and high temperature, it seems intuitive that power-aware scheduling techniques can be readily applied for the purpose of thermal-aware computing. However, the thermal management problem has its unique characteristics which are quite different from the power management problem [6]–[8].

In this paper, we study the problem on how to guarantee the feasibility of periodic tasks under the maximal temperature constraint. Traditionally, one common strategy is to check if each task instances of the task set can meet their deadlines within the first hyperperiod, i.e., the least common multiple (LCM) of the task periods. However, when we consider the maximal temperature constraints, this strategy does not apply anymore. As shown in [8] and [9], as well as later in this paper, a schedule for a periodic task set that can satisfy both the timing and the maximal temperature constraint within the first hyperperiod is not necessarily feasible later in the schedule. Therefore, new techniques need to be developed for checking the schedulability of real-time periodic task sets under the maximal temperature constraint.

When developing the power-aware or thermal-aware techniques in the deep submicron domain, the leakage plays a critical role, not only because the power and thermal issues become more prominent, but also because the leakage power consumption is becoming more and more significant. Liao *et al.* [10] have shown that the leakage power consumption can be 2–3 times higher than the dynamic power consumption for processors using the 65 nm technology. Furthermore, there is a strong relationship among the leakage, temperature, and supply voltage [10] in submicron circuits. When changing the temperature from 65 °C to 110 °C, Liao *et al.* [10] have shown that the leakage power can increase as much as 38%. High power consumption leads to high chip temperature, and high chip temperature, in turn, increases the leakage power, and thus the overall power consumption dramatically. Evidently, a power/thermal aware technique become less effective if this temperature/leakage feedback loop is not addressed properly.

To incorporate the leakage, temperature, and supply voltage relationship into the system-level thermal analysis and tech-

Manuscript received October 31, 2009; revised April 02, 2010 and May 26, 2010; accepted May 26, 2010. Date of publication June 14, 2010; date of current version August 06, 2010. This work was supported in part by the National Science Foundatin (NSF) under Project CNS-0545913, Project CNS-0917021, and Project CNS-CNS-0969013. Paper no. TII-09-10-0303.

The authors are with the Department of Electrical and Computer Engineering, Florida International University, Miami, FL 33174 USA (e-mail: gang.quan@fiu.edu; vchat001@fiu.edu).

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

Digital Object Identifier 10.1109/TII.2010.2052057

nique development is a challenging task. First, the circuit level analysis results [10], [11] indicate that the leakage power has a very complex relationship with both temperatures and supply voltages. Second, the temperature changes with the processor power consumption in a nonlinear manner. In this paper, we developed a novel power model that can capture the interaction between the temperature and leakage with high accuracy. Yet the model is also simple enough for ease of system level analysis. Based on this model, we developed three conditions to check the feasibility of real-time tasks under the peak temperature requirement. We conducted experiments based on technical parameters derived from a processor using the 65 nm technology. The results clearly demonstrate that the feasibility analysis without appropriately addressing the interaction of temperature, leakage, and supply voltage can deviate far away from the actual results.

The rest of this paper is organized as follows. We introduce the related work in Section II. Section III discusses the system models and formulates our problem formally. In Section IV, we study the unique characteristics of feasibility analysis problem for periodic tasks under maximal temperature constraints. In Section V, we incorporate the leakage/temperature dependency into our feasibility analysis and introduce several feasibility checking methods. We present our experimental results in Section VI and draws conclusions in Section VII.

II. Related Work

There have been increasing number of research results published on thermal aware real-time scheduling, crossing both single and multiple processor platforms (e.g., [6], [8], and [12]–[14]). In this paper, we focus on the feasibility checking problem for real-time tasks running on a single processor.

Several researches (e.g., [6] and [12]) try to identify the upper bound of the maximal temperature when executing real-time tasks on a single processor. These techniques cannot guarantee that real-time tasks can still meet deadlines when the maximal temperature is given. Some others (e.g., [6] and [15]–[18]) intend to minimize the peak temperature or guarantee the given maximal temperature constraints when scheduling a job set or a single copy of a task graph. For example, Bansal *et al.* [6] introduced an offline technique to minimize the energy consumption for a *job set*, and Chantem *et al.* [16] proposed an MILP-based solution to minimize the peak temperature when executing a task graph. While it is a common practice to repeat a real time schedule developed for jobs within the first hyperperiod of a periodic task set, as noted by Quan *et al.* [8] and Chen *et al.* [9], this approach is not applicable anymore if the temperature constraint is taken into consideration.

For periodic task sets, Wang *et al.* [19], [20] considered the problem of using two processor speeds to schedule a hard real-time task set. A processor runs at the highest possible speed until the temperature reaches the temperature threshold. Then the processor is set to run with the "equilibrium speed" at which the processor enters the equilibrium state with its temperature unchanged. This approach does not take the advantages that many modern processors support more than two levels of running speeds. In addition, it is not always possible that the "equilibrium speed" is exactly one of the available processor speeds. Zhang *et al.* [13] proposed to guarantee the temperature feasibility of a periodic system by forcing the temperature at the end of its first hyperperiod to be equal or less than the starting temperature. However, as shown later in this paper, this constraint can be overly pessimistic. In addition, none of these researches has taken the temperature/leakage dependency into consideration.

Researchers have already studied in depth the complex relationship between the leakage and temperature at the circuit and micro architecture level [10], [11], where the leakage current can be formulated as

$$I_{\text{leak}} = I_s \cdot \left(\mathcal{A} \cdot T^2 \cdot e^{((\alpha \cdot V_{dd} + \beta)/T)} + \mathcal{B} \cdot e^{(\gamma \cdot V_{dd} + \delta)} \right) \quad (1)$$

where I_s is the leakage current at certain reference temperature and supply voltage, T is the operating temperature, V_{dd} is the supply voltage, $\mathcal{A}$, $\mathcal{B}$, α, β, γ, δ are empirically determined technology constants. A temperature modeling tool called "HotSpot" [21] was developed base on this model, which can be effectively used to simulate and study the processor thermal phenomena at the architecture level. However, due to the nonlinear and high-order magnitude terms in (1), such a model or tool is too complex to be used for our feasibility analysis and scheduling technique development.

A few recent papers incorporate the temperature/leakage dependency into the energy- or thermal-aware scheduling. He *et al.* [22] and Yuan *et al.* [23] studied how to reduce the leakage power at the system level. Yuan *et al.* [24] introduced an offline and an online scheduling algorithm that take into account the leakage/temperature interactions when scheduling a set of soft real-time jobs. This approach cannot guarantee that real-time periodic tasks can meet deadlines under the given maximal temperature. A number of other approaches formulate the temperature-constrained problem as a convex optimization problem (e.g., [7] and [25]). The leakage/temperature relationship can thus be formulated as one of the constraints. The problem is that the computational complexity for the convex optimization problem is very high. Therefore, these approaches can only work at system level when the design solution space is small.

Liu *et al.* showed that using linear models is an effective way for accurate leakage estimation over the operating temperature ranges in real ICs [26]. A number of researches (such as [9] and [27]–[29]) simplify the leakage/temperature relationship based on this idea. Specifically, Chen *et al.* [9] and Chantem *et al.* [29] adopted a simple temperature/leakage dependency model that assumes the leakage power changes linearly *only* with temperature. As can be seen from (1), leakage varies not only with temperature but also supply voltage as well. In Section VI, we use experiments to study the accuracy of this model and its impacts to the schedulability analysis results.

III. Preliminary

The real-time system considered in this paper contains n independent periodic tasks, $\mathcal{T} = \{\tau_0, \tau_1, \cdots, \tau_{n-1}\}$. Task τ_i is characterized using three parameters, i.e., $\tau_i = (p_i, d_i, c_i)$. p_i, d_i ($d_i \leq p_i$), and c_i represent the period, the deadline and the worst case execution time for τ_i, respectively.

We use the lumped RC model similar to Shadorn *et al.* [30] to capture the thermal phenomena of the processor. Specifically, assuming a fixed ambient temperature (T_{amb}),[1] let $T(t)$ denote the temperature at time t. Then, we have

$$RC\frac{dT(t)}{dt} + T(t) - RP(t) = T_{\text{amb}} \tag{2}$$

where $P(t)$ denotes the power consumption (in Watt) at time t, and R, C denote the thermal resistance (in J/°C) and thermal capacitance (in Watt/°C). If we replace $T(t) - T_{\text{amb}}$ with $T(t)$ in (2), which is equivalent to scale T such that T_{amb} is zero, then we have

$$\frac{dT(t)}{dt} = aP(t) - bT(t) \tag{3}$$

where $a = 1/C$ and $b = 1/RC$. For the rest of this paper, we assume that the initial temperature of the processor equals the ambient temperature.

We assume the processor can run in different modes, with each mode being characterized by a pair of parameters (v_i, f_i), where v_i is the supply voltage and f_i is the working frequency in mode i. Even though the circuit delay (i.e., t_d) changes with the temperature dynamically, as shown in (4) [10],

$$f_i = \frac{1}{t_d} \propto \frac{(V_i - v_t)^\mu}{V_i T^\eta} \tag{4}$$

where v_t is the threshold voltage, and μ and η are technology-related constants, we assume that the processor working frequency in each mode is unique, and is the one that can accommodate circuit delay caused by the peak temperature (i.e., by assigning the peak temperature in (4)) across the chip. Let $f_{\max}$ be the largest f_i among different modes. We can normalize the processor working frequency with $f_{\max}$ and get the normalized processor speed for each mode. In what follows, unless otherwise specified, we use the term processor speed or working frequency interchangeably.

The power consumption (P) of the processor consists of two parts: the dynamic power (P_{dyn}) and the leakage power (P_{leak}).

$$P = P_{\text{dyn}} + P_{\text{leak}}. \tag{5}$$

The dynamic power consumption is independent of the temperature and can be formulated as $P_{\text{dyn}} \propto v_k^\xi$ with $\xi > 1$ [31]. For simplicity, we choose $\xi = 3$. The leakage power is sensitive to the temperature and can be estimated using the following formula:

$$P_{\text{leak}} = N_{\text{gate}} \cdot I_{\text{leak}} \cdot V_{dd} \tag{6}$$

where N_{gate} is the total number of gates, V_{dd} is the supply voltage and I_{leak} can be determined by (1).

Varying processor supply voltage and working frequency is one of the most effective ways to manage the power consumption dynamically. We call a schedule that dictates how to vary the processor supply voltage and working frequency as the *speed schedule*, which is formally defined as follows.

Definition 1: Given periodic task set $\mathcal{T}$, let L be the LCM of the periods, i.e.,$p_0, p_1, \cdots, p_{n-1}$. The speed schedule $\hat{S}(t)$ is defined as a sequence of $\langle [st_i, ed_i], mode_i \rangle$, where:

- $[st_i, ed_i]$ is an interval in which the processor runs in $mode_i$;
- $\bigcup_i [st_i, ed_i] = [0, L]$;
- $[st_i, ed_i] \bigcap [st_j, ed_j] = \emptyset$ if $i \neq j$.

With the thermal and processor models introduced as above, our problem can be formulated as follows.

Problem 1: Given

- a hard real-time task set $\mathcal{T} = \{\tau_0, \tau_1, \cdots, \tau_{n-1}\}$;
- a variable voltage processor that can run in m different modes, i.e., (v_i, f_i), $i = 0, \cdots, m-1$;
- the maximal allowable temperature $T_{\max}$;
- and a speed schedule $\hat{S}(t)$ with l intervals, i.e., $\langle [t_i, t_{i+1}], mode_i \rangle$, $i = 0, 1, \cdots, l-1$;

determine if $\mathcal{T}$ can meet the required deadlines using $\hat{S}(t)$ with the temperature stays below $T_{\max}$ all the time.

IV. The Leakage Oblivious Feasibility Analysis

In this, we study Problem 1 assuming that the leakage power is negligible. Under this assumption, the overall power consumption is therefore independent of temperature. Through this study, we intend to gain some valuable insights on how to deal with the maximal temperature constraints in the feasibility analysis for periodic task systems. We then incorporate the leakage/temperature dependency and develop several more elaborated feasibility conditions.

One common practice to ensure the feasibility of a periodic real-time task set is to construct a feasible schedule with interval $[0, L]$, where L represents the hyperperiod, i.e., the LCM of task periods. As long as the tasks are feasible in $[0, L]$, by replicating the schedule, the timing feasibility of the real-time system is guaranteed. However, when the execution of the real-time tasks are further constrained by a maximal temperature, is this approach still feasible?

We first introduce Theorem 1 which helps to answer this question.

Theorem 1: Given periodic task set $\mathcal{T}$, let:

- L be the LCM of the periods, i.e., $P_0, P_1, \cdots, P_{n-1}$;
- $\hat{S}(t)$ be the speed schedule within interval $[0, L]$ that can guarantee the deadlines of $\mathcal{T}$ under the maximal temperature constraints $T_{\max}$ with the initial temperature $T(0)$.

Then, when repeating $\hat{S}(t)$ later in the schedule, all task deadlines can be guaranteed under initial temperature $T(0)$ if $T(L) \leq T(0)$.

Proof: For interval $[t_0, t_1]$, let the temperature at $t = t_0$ be $T(t_0)$. Assuming there is no leakage power, based on (5), we can simplify the overall power consumption formulation as $P = v_k^3$, where v_k is the supply voltage when the processor is running in mode k. By solving (3), we have

$$T(t_1) = \int_{t_0}^{t_1} av^3(\tau) e^{-b(\tau - t_0)} d\tau + T(t_0) e^{-b(t_1 - t_0)}. \tag{7}$$

[1]The highest possible temperature can be used here for the safe design if the ambient temperature is variable.

So, we have

$$T(L) = \int_0^L av^3(\tau)e^{-b\tau}d\tau + T(0)e^{-bL}. \quad (8)$$

If we repeat $\hat{S}(t)$ for interval $[L, 2L]$, we have

$$T(2L) = \int_L^{2L} av^3(\tau)e^{-b(\tau-L)}d\tau + T(L)e^{-bL} \quad (9)$$

$$= \int_0^L av^3(\tau - L)e^{-b\tau}d\tau + T(L)e^{-bL} \quad (10)$$

Note that $v^3(\tau) = v^3(\tau - L)$. Thus, we have

$$T(2L) - T(L) = (T(L) - T(0))\, e^{-bL}. \quad (11)$$

So, for the $(k+1)$th LCM interval, we have

$$T((k+1)L) - T(kL) = (T(L) - T(0))\, e^{-kbL}. \quad (12)$$

Therefore, when $T(L) < T(0)$, the temperatures at $t = 0, L, 2L, \cdots$ will be monotonically decreasing. This ensures that if the maximal temperature constraint is not violated within $[0, L]$, it will not be violated within interval $[L, 2L], [2L, 3L], \cdots$. Therefore, under this scenario, $\hat{S}(t)$ must be globally schedulable. ■

Theorem 1 states that as long as the temperature at the ending point of a schedule is no more than the initial temperature at $t = 0$, repeating the schedule that is feasible during the first LCM interval is safe to guarantee the temperature and timing constraint. The question is then what if $T(L) > T(0)$. We present another theorem for this case. We use the same notation as that in Theorem 1.

Theorem 2: If $T(L) > T(0)$, when repeating $\hat{S}(t)$, all task deadlines can be guaranteed with initial temperature $T(0)$ if and only if

$$T(t_m) \le T_{\max} - \frac{T(L) - T(0)}{1 - e^{-bL}} e^{-bt_m} \quad (13)$$

for all $t_m \in [0, L]$.

Proof: From (12), when $T(L) > T(0)$, the temperatures at $t = 0, L, 2L, \cdots$ will be monotonically increasing. Also, $T(L) - T(0), T(2L) - T(L), T(3L) - T(2L), \cdots, T((k+1)L, kL)$ forms a geometric series and we have

$$T((k+1)L) = T(0) + \frac{(T(L) - T(0))\,(1 - e^{-kbL})}{1 - e^{-bL}}. \quad (14)$$

As $k \to \infty$, we have

$$\lim_{k\to\infty} T(kL) = T(0) + \frac{(T(L) - T(0))}{1 - e^{-bL}}. \quad (15)$$

Let $t_m \in [0, L]$ and $t_{m'} \in [kL, (k+1)L]$, where $t'_m = t_m + kL$. Based on (7), we have

$$T(t_{m'}) = \int_{kL}^{(k+1)L} av^3(\tau)e^{-b(\tau-kL)}d\tau + T(kL)e^{-b(t_{m'}-kL)} \quad (16)$$

and

$$T(t_m) = \int_0^L av^3(\tau)e^{-b\tau}d\tau + T(0)e^{-bt_m}. \quad (17)$$

Since $t_m = t_{m'} - kL$, we have

$$T(t_{m'}) = T(t_m) + c\frac{(T(kL) - T(0))}{e^{bt_m}}. \quad (18)$$

Since

$$\lim_{k\to\infty} T(kL) = T(0) + \frac{T(L) - T(0)}{1 - e^{-bL}} \quad (19)$$

so, $T(t_{m'}) \le T_{\max}$ if and only if

$$T(t_m) \le T_{\max} - \frac{T(L) - T(0)}{1 - e^{-bL}} e^{-bt_m}. \quad (20)$$

■

Theorems 1 and 2 provide the necessary and sufficient condition to predict if a schedule feasible within the first LCM is globally feasible. On the other hand, Theorem 2 also implies that not all schedules are feasible under the maximal temperature constraint even if they can guarantee the deadlines and maintain the maximal temperature below $T_{\max}$ during the first LCM, i.e., $[0, L]$. This is another example that the temperature-constrained real-time scheduling problem has its unique characteristics, compared with the corresponding power-aware scheduling problems.

V. The Leakage Conscious Feasibility Analysis

The results in previous section reveal some interesting and important characteristics in feasibility analysis for periodic tasks under the maximal temperature constraint, with the leakage power consumption ignored. However, the leakage power consumption is too significant to be ignored in the deep submicron domain, as stated before. In this section, we take the leakage power consumption into account and conduct a more sophisticated study on feasibility analysis.

A. Simplifying the Leakage/Temperature Dependency

When taking the leakage into account, one of the biggest challenges is to deal with the complex behavior of the leakage current, as formulated in (1). While (1) can capture accurately the characteristics of leakage current, the high-order and nonlinear terms make it prohibitive for our real-time feasibility analysis. Liu *et al.* [26] found that using linear approximation method to model the leakage/temperature dependence can maintain reasonable accuracy, i.e., with error within 1% using the piecewise linear function or less than 5.5% using single linear function, but the leakage model is significantly simplified. Based on this idea, we define the leakage power for the processor running in mode k as

$$P_{\text{leak}}(k) = (C_0(k) + C_1(k)T) \cdot v_k \quad (21)$$

where $C_0(k)$ and $C_1(k)$ are constants that depend on the running mode, i.e., k. In what follows, we omit variable k for sake of conciseness.

The overall power consumption in mode k is thus given by the following formula:

$$P(k) = (C_0 + C_1 T) \cdot v_k + C_2 v_k^3. \tag{22}$$

C_0, C_1, and C_2 can be determined in practice once the practical power consumptions at different temperatures are profiled. In Section VI, we show an example and study the accuracy of this model using technical parameters drawn from a processor using the 65 nm technology.

With the simplified leakage power formulation, we are now able to formulate the temperature dynamics in a closed form. Note that, when the processor runs in mode k, by combining (3) and (22), we have temperature variations as follows:

$$\frac{dT(t)}{dt} = A(k) - B(k)T(t) \tag{23}$$

where

$$A(k) = a\left(C_0 v_k + C_2 v_k^3\right) \tag{24}$$

$$B(k) = (b - aC_1 v_k). \tag{25}$$

If we run processor in mode k during interval $[t_1, t_2]$, with temperature at $t = t_1$ be $T(t_1)$, by solving (23), we can thus get the temperature at $t = t_2$ as

$$T(t_2) = \frac{A(k)}{B(k)} + \left(T(t_1) - \frac{A(k)}{B(k)}\right) e^{-B(k)(t_2 - t_1)}. \tag{26}$$

Equations (21)to (26) form the basis of feasibility analysis with leakage/temperature interplay taken into account. In what follows, we introduce several feasibility conditions developed based on this leakage power consumption model.

B. Checking the Temperature at the End of First Hyperperiod

The reason that a periodic task set feasible within the first hyperperiod is not necessarily feasible later in its life time is that the temperature at the end of a hyperperiod may be higher than that at the beginning of the hyperperiod. If this is case, starting at a new hyperperiod, the processor will run at a higher initial temperature and continue to reach an even higher temperature at the end of this hyperperiod. As this process continues, the temperature may eventually exceed the peak temperature. Conversely, from Theorem 1, as long as we can ensure that the temperature within the first hyperperiod is not higher than $T_{\max}$, and as long as the ending temperature is not higher than the initial temperature, we can determine that a schedule must be feasible under the given maximal temperature constraint. However, assuming the initial temperature be the ambient temperature, unless some aggressive cooling strategies are applied, the temperature of a processor will always increase when executing tasks. Therefore, the applicability of Theorem 1 is very limited. Next, we introduce two other theorems that can effectively deal with the case when $T(L) > T(0)$.

C. Checking the Temperature Safe Modes

Recall that, in Section III, the processor can work in different modes, each of which is associated with a distinct pair of supply voltage and working frequency. For some of the modes, no matter how long the processor runs in that mode, the maximal temperature will never exceed the given $T_{\max}$. We call these processor modes as the *safe modes*.

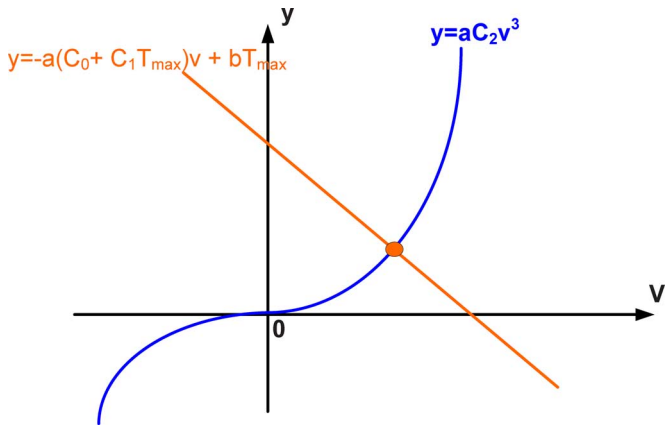

Fig. 1. Since the slope for the linear function $y = -a((C_0 + C_1 T_{\max})) \cdot v + bT_{\max}$ is less than zero, there is only one cross point for function $y = -a((C_0 + C_1 T_{\max})) \cdot V + bT_{\max}$ and function $y = aC_2 V^3$. So equation (28) has only one *real* root. [28]

To determine if a processor mode, i.e., mode k, is safe, we can set

$$\left.\frac{dT(t)}{dt}\right|_{T(t)=T_{\max}} = 0. \tag{27}$$

Based on (3) and (22), we have

$$a\left((C_0 + C_1 T_{\max}) \cdot v + C_2 v^3\right) - bT_{\max} = 0. \tag{28}$$

Note that (28) is the classic *depressed cubic equation* [32]. In addition, if we transform (28) slightly, we have

$$aC_2 v^3 = -a\left((C_0 + C_1 T_{\max})\right) \cdot v + bT_{\max}. \tag{29}$$

From Section V-A, it is not difficult to see that $aC_2 > 0$, and $a((C_0 + C_1 T_{\max})) > 0$. Therefore, as illustrated in Fig. 1, (28) has only one single *real* root for v, which can be solved analytically [33]. We call the solution to (28) as the *equilibrium voltage*. Note that different processor running modes may have different equilibrium supply voltages since C_0 and C_1 in (29) are different in different modes.

Formally, we have the following lemma to determine whether or not a processor running mode is a safe mode.

Lemma 1: Let v_e be the equilibrium voltage (i.e., the solution to (28)) for processor's mode k (i.e., (v_k, f_k)). Then, this mode is a *safe mode* if $v_e \geq v_k$.

Proof: We prove it by contradiction. Assume that at time $t \leq t_0$, we have $T(t) = T_{\max}$ but at $t_0 + \triangle t$, we have $T(t_0 + \triangle t) > T_{\max}$. So, we must have

$$\left.\frac{dT(t)}{dt}\right|_{t=t_0} = \left.\frac{dT(t)}{dt}\right|_{T(t)=T_{\max}} > 0. \tag{30}$$

On the other hand, since $v_e \geq v_k$, from (28), we have

$$\begin{aligned}\left.\frac{dT(t)}{dt}\right|_{t=t_0} &= a\left((C_0 + C_1 T_{\max}) \cdot v_k + C_2 v_k^3\right) - bT_{\max} \\ &\leq a\left((C_0 + C_1 T_{\max}) \cdot v_e + C_2 v_e^3\right) - bT_{\max} \\ &= 0\end{aligned} \tag{31}$$

which contradicts (30). ■

Based on Lemma 1, we can formulate our second feasibility checking method in the following theorem.

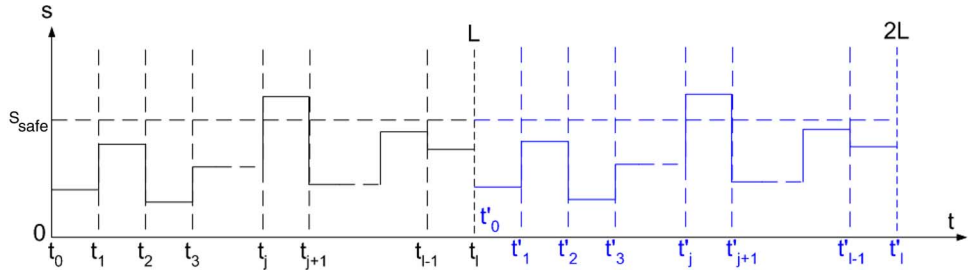

Fig. 2. A speed schedule within 2 hyperperiods.

Theorem 3: Let $\hat{S}(t)$ be the speed schedule within interval $[0, L]$ that can guarantee the deadlines of $\mathcal{T}$ under the maximal temperature constraint $T_{\max}$, and let $s_{\max}$ be the maximal speed in $\hat{S}(t)$. Also, let s_{mf} be the highest speed among the processor safe modes. Then, if $s_{\max} \leq s_{mf}$, when repeating $\hat{S}(t)$ later in the schedule, the temperature will never exceed $T_{\max}$.

Theorem 3 can be easily proved following the similar proof as that for Lemma 1. Note that, as long as the maximal temperature $T_{\max}$ and the processor is given, the highest speed (s_{mf}) among the processor safe modes is well determined. Also, it is much less costly to get the maximal speed $(s_{\max})$ in a schedule (such as those generated by the approach in [34]) rather than to get the entire speed schedule for a periodic task set. Therefore, this approach can be effectively used for the purpose of design space exploration.

Even though the feasibility condition formulated in Theorem 3 can be used for cases when the ending temperature of the first hyperperiod is higher than the initial temperature, this feasibility condition is still only a sufficient condition. In other words, when the maximal processor speed is higher than the maximal safe speed of the processor, the schedule may still be feasible under the maximal temperature. In what follows, we introduce the third feasibility condition, which is also a stronger condition and can be used to check the schedulability for these cases.

D. The Necessary and Sufficient Condition

To guarantee the maximal temperature constraint, we need to make sure that this constraint is not violated at the end point of each hyperperiod and anywhere inside the hyperperiod. It helps then to identify the possible locations within the hyperperiod that this constraint may be violated. In what follows, we first introduce the term, *island interval*.

Definition 2: An interval $[t_i, t_j]$ in $\hat{S}(t)$ is called an *island interval* if the processor needs to run in a nonsafe processor mode within this interval, or a *non-island interval* otherwise.

For an island interval, we have the following observation.

Lemma 2: Let $[t_1, t_2]$ be an island interval. If for any $t \in [t_1, t_2]$, we have $T(t) \leq T_{\max}$, then we have $T(t) \leq T(t_2)$.

Proof: For any $t \in [t_1, t_2]$, since the processor must be running at a nonsafe mode at t and $T(t) \leq T_{\max}$, we have

$$\left.\frac{dT(t)}{dt}\right|_{t\in[t_1,t_2]} > 0. \tag{32}$$

Therefore, $T(t)$ monotonically increases with t. So $T(t_2)$ must be the highest within the interval.

According to Lemma 2, the highest temperature for an island interval always occurs at its end, given that the maximal temperature constraint is not violated. Therefore, to verify if the temperature constraint is violated within a given hyperperiod, we only need to check temperatures at the ends of all island intervals, plus the one at the end of the hyperperiod.

Our goal is to make sure that the temperature constraint is not violated during the entire life cycle when executing a periodic task set. Exhaustively checking temperature constraint for all hyperperiods is apparently impossible. In addition, it is not adequate to draw a conclusion that a schedule is feasible under the maximal temperature constraint simply because the temperature constraint is not violated within the first hyperperiod. So, if we want to check temperatures only at the first hyperperiod, additional constraints must be imposed. The following theorem provides such "additional" constraints.

Theorem 4: Let the ith interval in $\hat{S}(t)$ be $[t_i, t_{i+1}]$ and let its processor mode be k. Define A_i, B_i such that

$$A_i = A(k) = a\left(C_0 v_k + C_2 v_k^3\right) \tag{33}$$
$$B_i = B(k) = (b - aC_1 v_k). \tag{34}$$

Let $[t_{(j-1)}, t_j]$ be an arbitrary island interval in $\hat{S}(t)$, and let

$$K_j = \exp\left(-B_0(t_1 - t_0) - \cdots - B_j\left(t_j - t_{(j-1)}\right)\right) \tag{35}$$
$$K = \exp\left(-B_0(t_1 - t_0) - \cdots - B_l\left(t_l - t_{(l-1)}\right)\right) \tag{36}$$

where $t_l = L$. Then repeating $\hat{S}(t)$ later in the schedule, the temperature will never exceed $T_{\max}$ *iff* the following conditions hold:

- $0 \leq K < 1$;
- $T(L) \leq T_{\max}(1 - K)$;
- $T(t_j) \leq T_{\max} - \frac{T(L)}{1-K} K_j$.

Proof: See Fig. 2. Let starting points for intervals in $\hat{S}(t)$ be $t_0, t_1, \cdots, t_{(l-1)}$, respectively. After repeating $\hat{S}(t)$, let the corresponding points during the second hyperperiod be $t'_0, t'_1, \cdots, t'_{(l-1)}$, correspondingly. Note that $t_0 = 0$, $t'_0 = t_l = L$ and $t'_l = 2L$.

According to (26), we have

$$T(t_1) = \frac{A_0}{B_0} + \left(T(t_0) - \frac{A_0}{B_0}\right) e^{-B_0(t_1 - t_0)}$$

and

$$T\left(t'_1\right) = \frac{A_0}{B_0} + \left(T(t'_0) - \frac{A_0}{B_0}\right) e^{-B_0(t'_1 - t'_0)}.$$

Since $(t'_1 - t'_0) = (t_1 - t_0)$, we have

$$T(t'_1) - T(t_1) = (T(t'_0) - T(t_0))\, e^{-B_0(t_1 - t_0)}. \tag{37}$$

Similarly, we have

$$T(t'_2) - T(t_2) = (T(t'_0) - T(t_0))\, e^{-B_0(t_1 - t_0) - B_1(t_2 - t_1)}.$$
$$\cdots$$

Therefore, we have

$$\begin{aligned} T(2L) - T(L) &= T(t'_l) - T(t_l) \\ &= (T(L) - T(0))\, e^{\sum_{i=1}^{l} -B_{i-1}(t_i - t_{i-1})} \\ &= (T(L) - T(0))\, K. \end{aligned} \tag{38}$$

In the same way, we can see that

$$T(3L) - T(2L) = (T(2L) - T(L)) \tag{39}$$
$$T(4L) - T(3L) = (T(3L) - T(2L))\, K$$
$$\cdots \tag{40}$$

Therefore, $T(L) - T(0), T(2L) - T(L), T(3L) - T(2L), \cdots, T(qL) - T((q-1)L)$ form a geometric series, and we have

$$T(qL) = T(0) + \frac{(T(L) - T(0))\,(1 - K^q)}{1 - K}. \tag{41}$$

Since $0 \le K < 1$, as $q \to \infty$, we have

$$\lim_{q \to \infty} T(qL) = T(0) + \frac{(T(L) - T(0))}{1 - K}. \tag{42}$$

After T_0 is calibrated to 0, $T(qL) \le T_{\max}$ if and only if

$$T(L) \le T_{\max}(1 - K). \tag{43}$$

We now need to make sure that the maximal temperature constraint is not violated in any island interval. Let $[t_{j-1}, t_j]$ be an arbitrary island interval in $\hat{S}(t)$. Follow the same procedure as stated above, we have

$$\begin{aligned} T(L + t_j) - T(t_j) &= (T(L) - T(0))\, e^{\sum_{i=1}^{j} -B_{i-1}(t_i - t_{i-1})} \\ &= (T(L) - T(0))\, K_j. \end{aligned}$$

Similarly, we have

$$\begin{aligned} T(2L + t_j) - T(L + t_j) &= (T(2L) - T(L))\, K_j \\ T(3L + t_j) - T(2L + t_j) &= (T(3L) - T(2L))\, K_j \\ &\cdots \\ T(qL + t_j) - T((q-1)L + t_j) &= (T(qL) \\ &\quad - T((q-1)L))\, K_j. \end{aligned}$$

Add all above questions together, we have

$$T(qL + t_j) - T(t_j) = (T(qL) - T(0))\, K_j. \tag{44}$$

Similarly, since $0 \le K < 1$, as $q \to \infty$, with (41), we can get

$$T(qL + t_j) = T(t_j) + \frac{(T(L) - T(0))}{1 - K} K_j. \tag{45}$$

So, after T_0 is calibrated to 0, $T(qL + t_j) \le T_{\max}$ if and only if

$$T(t_j) \le T_{\max} - \frac{T(L)}{1 - K} K_j. \tag{46}$$

■

Note that, after the speed schedule $\hat{S}(t)$ is defined, K and K_j are well defined. We then can check temperatures at the end of first hyperperiod as well as those at the end of island intervals. $\hat{S}(t)$ is a feasible schedule if the three conditions in Theorem 4 hold.

E. Further Discussions

From Theorem 4 and its proof, we have a number of interesting observations. Corollary 1, for example, is a straightforward conclusion from proof of Theorem 4.

Corollary 1: If $K > 1$ and $T(L) > T(0)$, the processor temperature will run away and reach infinity.

Corollary 1 can be easily proved from (41). When $K > 1$

$$\begin{aligned} \lim_{q \to \infty} T(qL) &= \lim_{q \to \infty} \left(T(0) + \frac{(T(L) - T(0))\,(1 - K^q)}{1 - K} \right) \\ &= \infty. \end{aligned}$$

This implies that the heat generated by the processor exceeds its cooling capability, and the temperature continues to rise until the system breaks down.

On the other hand, when $0 \le K < 1$, the processor temperature will eventually enter a *stable status* if the system does not break down before that. The *stable status* is formally defined as follows.

Definition 3: Assume a processor is running a periodic schedule $\hat{S}(t)$ with period L, the processor temperature is called to be in a *stable status* if for a given threshold, i.e., $0 < \varepsilon \ll 1$

$$|T((i+1)L) - T(iL)| < \varepsilon \tag{47}$$

where $i \ge 0$, $i \in Z$.

When the processor temperature enters the stable status, the temperature profile does not change much from one hyperperiod to another hyperperiod. Also, from the proof of Theorem 4, the temperature when the processor is in its stable status can be analytically formulated as follows.

Lemma 3: Assume a processor is running a periodic speed schedule $\hat{S}(t)$ with period L. Let $T(0)$ and $T(L)$ be temperatures at $t = 0$ and L, respectively. Then, when the processor temperature reaches its stable status, the temperature at the starting (or ending) point of a period, denoted as $T(L')$, can be formulated as

$$T(L') = T(0) + \frac{(T(L) - T(0))}{1 - K}. \tag{48}$$

Note that similar lemmas can also be developed to calculate the temperature at each specific scheduling point, based on (45) in the proof of Theorem 4.

In summary, we introduce three feasibility testing methods (Sections V-B, –Section V-D) to verify if a feasible schedule

developed within the first hyperperiod is globally feasible or not under the given maximal temperature constraint. The first two, which are based on Theorems 1 and 3, are sufficient conditions. The third one, based on Theorem 4, is a more elaborated necessary and sufficient condition. All three methods take the leakage/temperature dependency into account based on the processor power model formulated in (22). In what follows, we use experiments to further study their effectiveness.

VI. Experiment

The feasibility conditions introduced in the previous section are established based on the leakage power model proposed in Section III, or more specifically, formulated by (22). Therefore, to study the performance of the feasibility conditions, we need to first validate the leakage model. It is difficult to analyze the accuracy of the leakage model analytically, since the constants C_0 and C_1 are obtained through polynomial approximation methods rather than from some analytical formulas. In this section, we validate the leakage model through experiments. We then examine the performance of different schedulability conditions presented before. We also studied what impacts different leakage models may have in schedulability analysis.

A. Leakage Model Validation

We constructed our processor model based on the work by Liao *et al.* [10] for a processor using 65 nm technology. We assumed that the processor can run in six different active modes, with the corresponding supply voltages as 0.85 V, 0.9 V, 0.95 V, 1.0 V, 1.05 V, and 1.1 V. The processor can also be shut down and consumes no energy. For each mode, we set the frequency of the processor in each mode such that it can accommodate the longest delay at the highest temperature (110 °C) based on (4), with $\mu = 1.19$, $\eta = 1.2$, and $v_t = 0.3$ [10]. For the thermal constants, we selected $Rth = 0.8\text{K/W}$, $Cth = 340\text{J/K}$ [3], and the ambient temperature was set to 25 °C.

We studied four different leakage models shown below:

- $Const_A$ The leakage power is a constant that is independent of both temperature and supply voltage. This is the leakage power model used by many previous researches (such as [8]). The constant is determined by the average leakage power consumption at the ambient temperature.
- $Const_H$ The leakage power is also a constant. But the constant is determined by the average leakage power consumption at the highest temperature.
- LK_T The leakage power changes only with the temperature but not the supply voltage. This model is adopted in previous work such as [9], [27], and [29].
- $LK_{T\&V}$ The leakage power varies with both the temperature and supply voltage. This is the model proposed in Section III.

We used the analytical formula, i.e., (1), to compute the actual leakage power for temperature from 40 °C to 110 °C with step size of 10 °C. The total number of gate, i.e., N_{gate} in (6), was set to be 10^6. The dynamic power consumption was determined based on the experimental results reported in [10] on benchmark *gcc*. The corresponding power consumption results were then used to determine the constants in the leakage models.

TABLE I
Processor Parameters and Constants for Model $LK_{T\&V}$

$V_{dd}(V)$	C_0	C_1	C_2	Frequency
0.00	0.0	0.0	0.0	0.0
0.85	7.3249	0.1666	15.0	0.8010
0.90	8.6126	0.1754	15.0	0.8291
0.95	10.238	0.1846	15.0	0.8553
1.00	12.315	0.1942	15.0	0.8797
1.05	14.998	0.2043	15.0	0.9027
1.10	18.497	0.2149	15.0	1.0

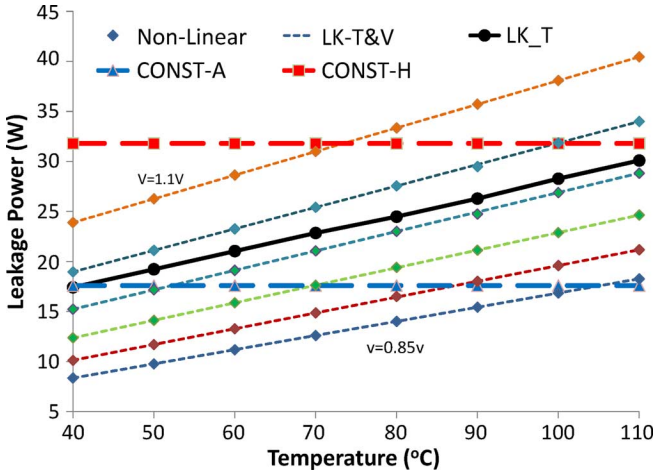

Fig. 3. The leakage power consumptions based on different leakage models.

Specifically, the power consumptions obtained above were used to determined the curve fitting constants C_0, C_1 and C_2 for Model $LK_{T\&V}$, which are listed in Table I. The leakage power consumptions for $Const_A$ and $Const_H$ were defined as the average results at the ambient temperature ($T = 25$ °C) and the highest temperature ($T = 110$ °C), respectively. For Model LK_T, the average leakage power consumption at each temperature was used to derive the corresponding linear function.

Fig. 3 plots the leakage power consumptions based on the complex nonlinear model [i.e., (1)] and other four models. As we can see from Fig. 3, with linear approximation, the leakage power consumptions based on Model $LK_{T\&V}$ match very closely to that by a much more complex method [(1)]. As further shown in Fig. 4, the relative error is less than 5% for 0.85 V and less than 3% for 1.1 V. The results clearly show that Model $LK_{T\&V}$ is a leakage model with low complexity and very good accuracy.

On the other hand, however, our experimental results also show that if the supply voltage or temperature dependency are not carefully addressed, the leakage model can lead to estimation results deviated far away from the actual values. From Fig. 3, when the supply voltage is not taken into consideration, the estimation errors by Model LK_T can be as much as 2.08 times higher or 26.7% lower than the actual leakage power consumptions. When also ignoring the leakage/temperature dependency, the estimation errors by Model $Const_A$ and $Const_H$ become even larger, i.e., as much as 3.8 times higher or 56% lower than the actual values. We investigate how significant the leakage power estimation errors may affect schedulability analysis in Section VI-C.

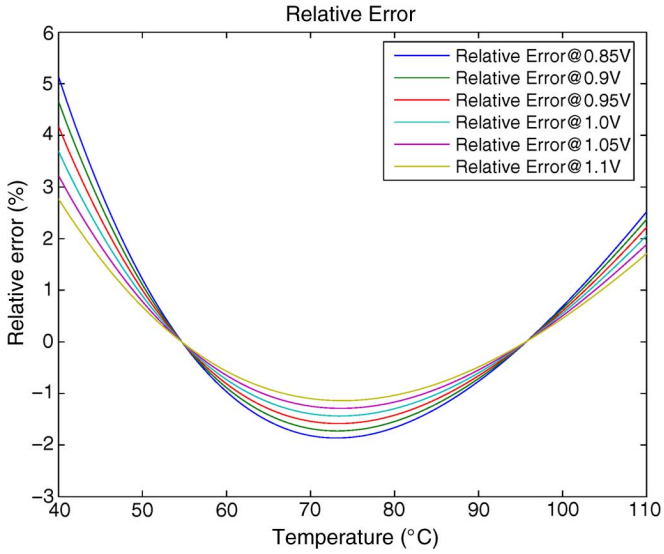

Fig. 4. The relative estimation errors by Model $LK_{T\&V}$.

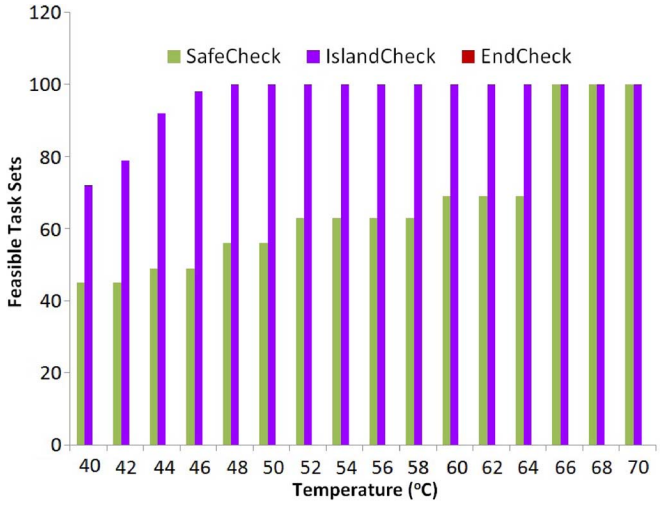

Fig. 5. Success rates under different maximal temperatures (deadline − period ratio = 0.3).

B. The Performance of Feasibility Conditions

We next used the processor model developed above to study the feasibility analysis methods proposed in this paper. Three feasibility checking methods were implemented and investigated. The first one (namely, **EndCheck**), based on Theorem 1 checks if the temperature at the end of the hyperperiod is no more than the initial temperature. The second one (namely, **SafeCheck**) applies Theorem 3 and uses the processor safe speed to check the feasibility. The third one (namely, **IslandCheck**) employs Theorem 4, checking temperatures at the ending points of the first hyperperiod and all island intervals in the first hyperperiod.

The real time tasks were randomly generated with periods distributed evenly in range [100, 500] seconds. Deadlines were determined by multiplying periods with a constant, called the *deadline-period ratio*. The execution time for each task was also randomly generated, which is evenly distributed between 1 and its deadline. The feasible speed schedules, generated based on the optimal method to minimize the dynamic energy [34], were used as our test cases. Since the approach in [34] assumes a processor model with continuously variable speed, we always rounded up a processor speed to the next higher available one in our experiments.

In the first set of experiments, we fixed the deadline-period ratio at 0.3 (i.e., $DPratio = 0.3$) and varied the peak temperature constraint from 40 °C to 110 °C, with step size of 1 °C. For each peak temperature, we generated 100 test cases that can satisfy deadlines if the temperature factor is not taken into consideration. Fig. 5 shows the numbers of schedulable task sets using the three feasibility checking methods stated above. Fig. 5 shows clearly the significant impacts of peak temperature requirement to the schedule's feasibility. Note that in Fig. 5, when the peak temperature constraint is higher than 66 °C, all 100 schedules randomly generated as above can satisfy the temperature constraints based on **IslandCheck** and **SafeCheck**. When the peak temperature constraint getting tighter, however, the feasibility drops quickly. In Fig. 5, when the peak temperature is set to 40 °C, about 30% of the original feasible schedules become infeasible. At the same time, we can see that **SafeCheck** is pessimistic in predicting the feasibility for a schedule. This is because a schedule occasionally using a speed higher than the processor safe speed can still reach a temperature lower than the required maximal temperature. In Fig. 5, when the maximal temperature is set to be 58 °C, about 38% of the feasible task sets cannot be properly verified by **SafeCheck**. When the given maximal temperature becomes very high, all processor running modes become safe modes, and thus **SafeCheck** obtains the same results as that by **IslandCheck** in Fig. 5.

Note that in Fig. 5, none of the task sets can be predicted as feasible using **EndCheck**. This is because that, starting from the ambient temperature, the processor temperature will never fall below the ambient temperature after executing tasks, given the thermal settings stated before. Conceivably, if we set the starting temperature to be higher than the ambient temperature, using **EndCheck** may be more effective. Therefore, we conducted the second set of experiments to further compare the performance of **EndCheck** and **IslandCheck**.

In this set of experiments, we fixed the maximal temperature at 50 °C. The test cases were randomly generated as above with the period-deadline ratio varied from 0.1 to 0.9 with step size of 0.2. We also varied the starting temperature of **EndCheck** from 0 (the ambient temperature of 25 °C) to 25 (the maximal temperature of 50 °C). The number of feasible task sets by **EndCheck** is normalized using the number of feasible task sets by **IslandCheck** and plotted in Fig. 6.

As shown in Fig. 6, the success rate by **EndCheck** initially improves with the increase of the starting temperature. This is because that setting the starting temperature higher makes it more likely that the ending temperature at the first hyperperiod can becomes lower than the starting temperature. However, it is interesting to note that as starting temperature continues to increase, the success rate by **EndCheck** starts to drop. This is because that if the starting temperature is set to be too high, the temperature may exceed the temperature constraint before its hyperperiod and thus lead to pessimistic conclusion. Fig. 6 clearly shows that **EndCheck** is sensitive to the choice of the

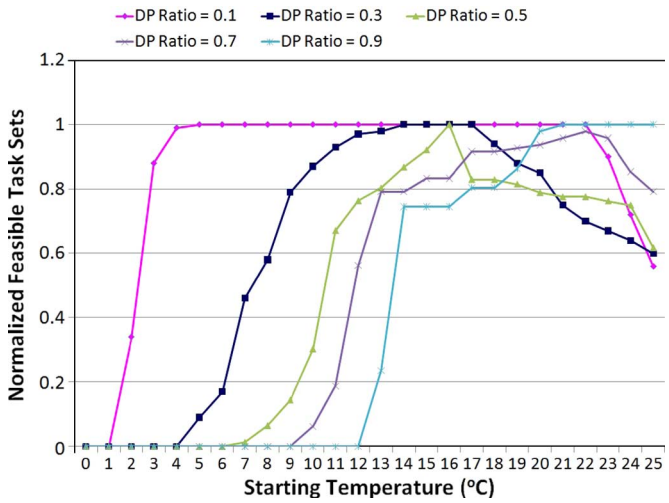

Fig. 6. Success rates by **EndCheck** using different starting temperatures. ($T_{\max} = 50$ °C)

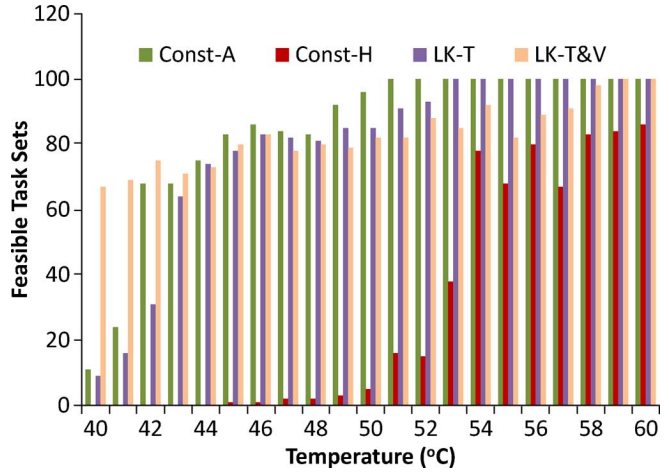

Fig. 7. Feasible task sets based on different leakage models under different temperature (deadline − period ratio = 0.3)

starting temperature and can be very pessimistic in predicting the feasibility of real-time task sets.

C. The Impacts of Different Leakage Models

We further examine how different leakage power models may affect the feasibility analysis results. We used the same test cases generated in Section VI-B and applied **IslandCheck** on all four leakage models introduced in Section VI-A. The numbers of feasible task sets are collected and depicted in Fig. 7.

The results shown in Fig. 7 verify the large discrepancies in terms of task set schedulability caused by the large estimation errors by different leakage models. When assuming the leakage power being a constant at the highest temperature, the leakage model $Const_H$ can lead to a feasibility analysis that is extremely pessimistic. Note that in Fig. 7 none of the task sets is predicted as schedulable according to $Const_H$ at temperature of 44 °C. According to leakage model $LK_{T\&V}$, however, 74% of the task sets are in fact schedulable.

When assuming the leakage power consumption at the ambient temperature, the feasibility analysis based on leakage model $Const_A$ can be both pessimistic or optimistic. Note that we defined the constant leakage power consumption in $Const_A$ using the average results under different supply voltages at the ambient temperature. This leakage power consumption can thus be over estimated when the actual supply voltage is low or under estimated when the actual supply voltages is high. When the maximal temperature constraint is low, the schedules that employ low processor speed are more likely to satisfy the temperature constraints. As a result, the feasibility analysis results based on $Const_A$ tends to be pessimistic for these test cases. In Fig. 7, at 40 °C, 82% of the feasible task sets cannot be correctly predicted based on leakage model $Const_A$. As temperature increases, the feasibility analysis results become more and more optimistic. At temperature 49 °C, at least 22% task sets that are predicted as feasible according to $Const_A$ are in fact infeasible according to results based on the leakage model $LK_{T\&V}$.

Even though the leakage/temperature dependency is considered in model LK_T, large estimation errors still exist since the leakage power varies not only with temperature but also supply voltage. As a result, similar to leakage model $Const_A$, the feasibility analysis based on leakage model LK_T can be overly pessimistic or overly optimistic. At 40 °C, as many as 86% of the feasible tasks cannot be properly predicted based on model LK_T, and as many as 15% of the feasible task sets at 53 °C in fact cannot satisfy the temperature constraint. These results clearly demonstrate that the feasibility analysis without appropriately accounting for the leakage power with temperature and supply voltage can deviate far away from the actual results.

VII. Conclusion

As semiconductor technology continues to evolve, temperature-aware computing plays an increasingly critical role in computing system design, and the power-aware design techniques alone are inadequate to effectively address the temperature-related issues. Further, as the transistor size continue to shrink, the leakage in the CMOS circuit and its interaction with temperature and supply voltage become a too significant factor to be ignored at the system level design.

In this paper, we study the feasibility checking problem for real-time periodic task sets under the peak temperature constraint. We show that the traditional scheduling approach, i.e., to repeat the schedule that is feasible through the range of one hyper-period, does not apply any more. We propose a novel leakage mode with leakage/temperature dependency taken into consideration. This model leverages the circuit and micro-architecture level leakage model for ease of system level analysis. Based on the proposed power model, we present three feasibility analysis techniques to determine if a period task set can meet deadlines under a given maximal temperature constraint. Our experimental results, based on technical parameters derived from a processor using 65 nm technology, show that our leakage current model have a relative error less than 5%. In addition, our experimental results on the feasibility analysis demonstrate the effectiveness of our methods and clearly highlight the importance to deal with the impacts of leakage/temperature relationship.

REFERENCES

[1] *International Technology Roadmap for Semiconductors (ITRS)*. Austin, TX: International SEMATECH, 2005 ed. [Online]. Available: http://public.itrs.net/,

[2] S. Borkar, "Thousand core chips: A technology perspective," in *Proc. DAC*, 2007, pp. 746–749.

[3] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, "Temperature-aware microarchitecture," in *Proc. ICSA*, 2003, pp. 2–13.

[4] L.-T. Yeh and R. C. Chu, *Thermal Management of Microelectronic Equipment: Heat Transfer Theory, Analysis Methods, and Design Practices*. New York: ASME Press, 2002.

[5] O. S. Unsal and I. Koren, "System-level power-aware design techniques in real-time systems," *Proc. IEEE*, vol. 91, no. 7, pp. 1–15, Jul. 2003.

[6] N. Bansal, T. Kimbrel, and K. Pruhs, "Speed scaling to manage energy and temperature," *J. ACM*, vol. 54, no. 1, pp. 1–39, 2007.

[7] Y. Liu, H. Yang, R. P. Dick, H. Wang, and L. Shang, "Thermal vs energy optimization for DVFS-enabled processors in embedded systems," in *Proc. ISQED*, 2007, pp. 204–209.

[8] G. Quan, Y. Zhang, W. Wiles, and P. Pei, "Guaranteed scheduling for repetitive hard real-time tasks under the maximal temperature constraint," in *Proc. ISSS+CODES*, 2008, pp. 267–272.

[9] J.-J. Chen, S. Wang, and L. Thiele, "Proactive speed scheduling for real-time tasks under thermal constraints," in *Proc. RTAS*, 2009, pp. 141–150.

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

[11] Y. Zhang, D. Parikh, K. Sankaranarayanan, K. Skadron, and M. Stan, "Hotleakage: A temperature-aware model of subthreshold and gate leakage for architects," Dept. Comput. Sci., Univ. Virginia , Charlottesville, VA, Tech. Rep., 2003.

[12] J. Chen, C. Hung, and T. Kuo, "On the minimization of the instantaneous temperature for periodic real-time tasks," in *Proc. RTAS*, 2007, pp. 236–248.

[13] S. Zhang and K. S. Chatha, "Approximation algorithm for the temperature-aware scheduling problem," in *Proc. ICCAD*, 2007, pp. 281–288.

[14] X. Zhou, Y. Xu, Y. Du, Y. Zhang, and J. Yang, "Thermal management for 3d processors via task scheduling," in *Proc. ICPP*, 2008, pp. 115–122.

[15] A. Cohen, L. Finkelstein, A. Mendelson, R. Ronen, and D. Rudoy, "On estimating optimal performance of CPU dynamic thermal management," *IEEE Computer Architecture Lett.*, vol. 2, no. 1, pp. 6–9, 2003.

[16] T. Chantem, R. P. Dick, and X. S. Hu, "Temperature-aware scheduling and assignment for hard real-time applications on MPSoCs," in *Proc. DATE*, 2008, pp. 288–293.

[17] R. Rao, S. Vrudhula, C. Chakrabarti, and N. Chang, "An optimal analytical solution for processor speed control with thermal constraints," in *Proc. ISLPED*, 2006, pp. 292–297.

[18] J. Yang, X. Zhou, M. Chrobak, Y. Zhang, and L. Jin, "Dynamic thermal management through task scheduling," in *Proc. ISPASS*, 2008, pp. 191–201.

[19] S. Wang and R. Bettati, "Reactive speed control in temperature-constrained real-time systems," in *Proc. ECRTS*, 2006, pp. 161–170.

[20] S. Wang and R. Bettati, "Delay analysis in temperature-constrained hard real-time systems with general task arrivals," in *Proc. RTSS*, 2006, pp. 323–334.

[21] "Hotspot 4.2 temperature modeling tool," Univ. Virgina. Charlottesville, VA. [Online]. Available: http://lava.cs.virginia.edu/HotSpot, 2009

[22] L. He, W. Liao, and M. R. Stan, "System level leakage reduction considering the interdependence of temperature and leakage," in *Proc. DAC*, 2004, pp. 12–17.

[23] L. Yuan, S. Leventhal, and G. Qu, "Temperature-aware leakage minimization technique for real-time systems," in *Proc. ICCAD*, 2006, pp. 761–764.

[24] L. Yuan and G. Qu, "Alt-dvs: Dynamic voltage scaling with awareness of leakage and temperature for real-time systems," in *Proc. NASA/ESA Conf. Adaptive Hardware and Syst.*, 2007, pp. 660–670.

[25] S. Murali, A. Mutapcic, D. Atienza, R. Gupta, S. Boyd, and G. D. Micheli, "Temperature-aware processor frequency assignment for MPSoCs using convex optimization," in *Proc. CODES+ISSS*, 2007, pp. 111–116.

[26] Y. Liu, R. P. Dick, L. Shang, and H. Yang, "Accurate temperature-dependent integrated circuit leakage power estimation is easy," in *Proc. DATE*, 2007, pp. 1526–1531.

[27] N. Fisher, J.-J. Chen, S. Wang, and L. Thiele, "Thermal-aware global real-time scheduling on multicore systems," in *Proc. RTAS*, 2009, pp. 131–140.

[28] G. Quan and Y. Zhang, "Leakage aware feasibility analysis for temperature-constrained hard real-time periodic tasks," in *Proc. ECRTS*, 2009, pp. 207–216.

[29] T. Chantem, X. S. Hu, and R. Dick, "Online work maximization under a peak temperature constraint," in *Proc. ISLPED*, 2009, pp. 105–110.

[30] K. Skadron, T. Abdelzaher, and M. R. Stan, "Control-theoretic techniques and thermal-RC modeling for accurate and localized dynamic thermal management," in *Proc. HPCA*, 2002, p. 17.

[31] J. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*. Englewood Cliffs, NJ: Prentice-Hall, 2003.

[32] P. J. Nahin, *The Story of* $\sqrt{-1}$. Boston: Princeton University Press, 1998.

[33] Wikipedia, Cubic Function, 2008. [Online]. Available: http://en.wikipedia.org/wiki/Cubic_equation

[34] F. Yao, A. Demers, and S. Shenker, "A scheduling model for reduced CPU energy," in *Proc. FOCS*, 1995, pp. 374–382.

Gang Quan (S'01–M'02) received the B.S. degree from the Tsinghua University, Beijing, China, the M.S. degree from the Chinese Academy of Sciences, Beijing, and the Ph.D. degree from the University of Notre Dame, Notre Dame, IN.

He is currently an Associate Professor with the Electrical and Computer Engineering Department, Florida International University, Miami. His research interests includes real-time system, power/thermal aware design, embedded system design, advanced computer architecture and reconfigurable computing. Prof. Quan received the NSF CAREER award in 2006.

Vivek Chatuvedi (S'10) received the M.S. degree from the Department of Electrical Engineering, Syracuse University, Syracuse, NY, in 2008. He is currently working towards the Ph.D. degree at the Electrical and Computer Engineering Department, Florida International University, Miami.

During his studies at Syracuse University, he also worked as an Engineering Intern with the Micro-Electronics Group in SUN Microsystems, Burlington, MA. His research interests include real-time systems, operating systems, scheduling techniques, and computer architecture.