Journal of Systems Architecture 58 (2012) 387–397

Contents lists available at SciVerse ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

On the fundamentals of leakage aware real-time DVS scheduling for peak temperature minimization

Vivek Chaturvedi [a,*], Huang Huang [a], Shangping Ren [b], Gang Quan [a]

[a] Florida International University, Miami, FL 33174, United States
[b] Illinois Institute of Technology, Chicago, IL 60616, United States

ARTICLE INFO

Article history:
Received 23 January 2012
Received in revised form 15 June 2012
Accepted 17 August 2012
Available online 5 September 2012

Keywords:
Thermal-aware scheduling
Embedded systems
Real-time systems

ABSTRACT

As the consequence of the exponentially increased power density on integrated circuits, thermal issues are becoming critical in design of computing systems. Moreover, as both leakage and thermal issues have become more prominent in the deep sub-micron domain, a power and thermal aware design technique becomes less effective if the leakage/temperature dependency is not appropriately addressed. In this paper, we take into account the dependency among the leakage, the temperature, and the supply voltage in our theoretical analysis and explore the fundamental characteristics on how to employ dynamic voltage scaling (DVS) to reduce the peak operating temperature. We find that, for a specific interval, a real-time schedule using the lowest constant speed is not necessarily the optimal choice any more in minimizing the peak temperature. We identify the scenarios when a schedule using two different speeds can outperform the one using the lowest constant speed. In addition, we find that, when scheduling a periodic task set, the constant speed schedule is still the optimal solution for minimizing the peak temperature when the temperature is at its *stable status*. We formulate our conclusions into several theorems with formal proofs.

1. Introduction

A semiconductor technology continues to scale down, the chip temperature increases dramatically due to the growing power consumption. The escalating heat has directly led to high packaging/cooling costs and also potentially degrades the performance, life span, and reliability of computing systems [1,2]. Left unchecked, the thermal issue will severely handicap the computing systems in the near future. The severity of the problem is further highlighted by the Intel's acknowledgement of hitting a *thermal wall* [3].

For the past several decades, a closely related problem, i.e. the power/energy reduction problem, has been researched extensively. While lowering power/energy consumption does help to reduce the heat generation, the power aware computing problem and the thermal aware computing problem are distinctly different, as exhibited in many previous work (e.g. [4,5]). As a result, the existing power/energy reduction techniques cannot be readily applied to address the thermal issues.

When studying the thermal aware dynamic voltage scaling (DVS) problem, it is imperative to take the dependency among the leakage, the temperature, and the supply voltage into considerations. It is a well known fact [6] that the leakage power increases with the temperature and the supply voltage. Due to down-scaling of semiconductor device technology, the leakage power consumption is becoming comparable or even surpasses the dynamic power consumption [7]. Therefore, a DVS technique cannot be effective without considering the relations among the leakage, the temperature, and the supply voltage.

The goal of this paper is to explore the characteristics and guidelines that can be exploited in the development of effective thermal aware design techniques. Since the thermal management problem is closely related to the power reduction problem, we started our research by investigating how effective the basic principles for energy reduction can be, when applied for dynamic thermal management. We begin with the two well-known principles [8,9], for dynamic energy reduction,

- *Principle 1*: using the lowest constant speed leads to the schedule that consumes the minimum dynamic energy.
- *Principle 2*: if a single lowest constant speed is not available, then using two closest neighboring speeds is the optimal solution in dynamic energy reduction.

The question then becomes: when considering the complex relationship among the leakage power, the temperature, and the supply

* Corresponding author.
E-mail addresses: vchaturv@fiu.edu (V. Chaturvedi), hhuang001@fiu.edu (H. Huang), ren@iit.edu (S. Ren), hhuang001@fiu.edu (G. Quan).

1383-7621/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.sysarc.2012.08.002

voltage, is it still true that a real-time schedule employing the lowest constant speed will lead to the lowest peak temperature within the scheduling interval? We find that, for a specific workload and interval, the schedule that uses the lowest constant speed is not necessarily optimal anymore in reducing the peak temperature. We identify the scenarios when a schedule that uses two different speeds may in fact lead to lower peak temperature. We also find that principles similar to the two listed above do exist to minimize the peak temperature during the temperature *stable status* when scheduling a periodic task. We formulate our observations into several theorems with formal proofs. The significance of this paper is that it uncovers a number of fundamental principles in the development of effective DVS techniques for thermal aware computing.

The rest of the paper is organized as follows. Section 2 introduces the related work. System models used in this paper are defined in Section 3. Section 4 presents our empirical results to motivate our research. Fundamental principles are formulated and proved in Sections 5 and 6. Section 7 concludes the paper.

2. Related work

With the semiconductor technology scaling to the deep sub-micron domain, the power density of the IC chip is increasing exponentially [7]. Reducing energy consumption has long been a major design goal in computing system design. For past several decades, there have been extensive research work published on power/energy reduction by using DVS techniques (e.g. [8–10]). At the same time, the soaring chip temperature is becoming critical as the power density continues to grow. Though the power and the thermal issue are closely related, previous researches have shown clearly that the power/energy aware problem and the thermal aware problem have different characteristics (e.g. [4,5,11,12]). An optimal technique in power/energy reduction is not an optimal technique with regard to the peak temperature minimization. Therefore, many recent researches have been conducted to address the thermal problem at different design abstraction levels, from circuit, logic, architecture, all the way to the system levels (e.g. [1,13,14,5,12,15–17], etc.).

As the related work, there have been many scheduling techniques developed that incorporate the thermal issues into the scheduling decision process (e.g. [18,19,16,15,20] etc.). For example, a number of papers have been published on either minimizing the overall energy consumption (e.g. [18,21,19]) or maximizing the system throughput (e.g. [15,22,20,23]) under the given maximum temperature constraint. Specifically, Bao et al. [18] proposed to distribute the idle interval judiciously when scheduling a task graph such that the temperature of the processor can be effectively "cooled down" and reduce the leakage and overall power consumption. Yang et al. [21] proposed a "pattern-based" scheduling approach to periodically switch the processor between the active and dormant modes and reduce the energy. Huang et al. [19] derived a closed-form energy calculation equation based on which they further proposed an energy minimization scheduling method by extending the concept of m-oscillating approach proposed by Chaturvedi et al. [16]. Zhang et al. [23] developed several algorithms to maximize the throughput of a real-time system by sequencing the execution of a task set consisting of tasks with heterogeneous power/thermal characteristics for processors with and without Dynamic Voltage/Frequency Scaling (DVFS) capability.

Closely related to our work in this paper, there are also many other thermal aware scheduling researches seek to reduce the maximum operating temperature for a computing system. For example, Bansal et al. [5] modeled the cooling behavior of a device using first-order approximation to manage the temperature and energy of the system. Zhang et al. [12] proposed performance optimization by latency minimization under thermal constraints. Chantem et al. [22] proposed a dynamic work maximization technique by using DVFS with non-negligible transition overheads and under the temperature constraint. In [24], Jayaseelan et al. proposed different iterative job sequencing techniques to identify the peak temperature and to reduce the peak temperature of the system. Chaturvedi et al. developed [16] a so-called "m-oscillating" scheduling method to minimize the peak temperature for a periodic task set. In their approach they used two-neighboring speeds to oscillate alternately to reduce the peak temperature of the system. Liu et al. [25] proposed to reduce the temperature of the system by properly sequencing hot and cool jobs and allocating slack time to hot jobs based on the duration of execution. Kumar et al. [17] proposed a stop-n-go approach to reduce the peak temperature for task with data dependencies. They distribute the slack time between jobs such that temperature can be minimized and there is no make-span violation. In [26], Kumar et al. developed a novel online technique that uses shapers to insert idle-time between the task set to reduce the peak temperature of the system. They modeled the task set by a resource-based curve on a single speed processor.

One distinct difference between our research and the existing researches is the way we deal with the leakage/temperature dependency. Some of the existing work totally ignore the leakage power or the leakage/temperature dependency (e.g. [26,17,18,12]). As discussed before, a thermal aware scheduling technique becomes out of sync with the current IC technology in the deep submicron domain without taking the leakage/temperature dependency into considerations. As the leakage power becomes more prominent, the heat generated by the processor can dramatically increase the leakage power and thus the overall power consumption. At the same time, the increased overall power consumption will in turn drive the temperature to an even higher level. Therefore, to build an appropriate model that can effectively handle the sensitive relationship between leakage/temperature, is the key to success when developing thermal aware scheduling techniques.

One way to deal with the leakage/temperature dependency is to incorporate the complex circuit level leakage model into system analysis [27,28]. For instance, He et al. [27] and Yuan et al. [28] studied how to reduce the leakage power at the system level. Yuan et al. [28] introduced an offline and an on-line scheduling algorithm that take into account the leakage/temperature interactions when scheduling a set of soft real-time jobs. However, due to the non-linear and high-order magnitude terms in the model, it can be too complex and cumbersome to be used for more rigorous real-time analysis and scheduling technique development.

Another common approach to address the leakage/temperature dependency that is adopted by existing work (such as [29,22,18,30,23,31,17,26]) is to assume that the *leakage power* changes linearly or quadratically with temperature. Since leakage current changes super linearly with temperature [32], this model works well if the supply voltage do not change. Otherwise, as evidenced by the empirical results in Huang et al. [33], this model can lead to large discrepancies in either power consumption or peak temperature calculation in a DVS system.

In this paper, we use a leakage model that can capture the leakage/temperature/supply voltage dependencies. Quan et al. [34] introduced a leakage/temperature model in which the leakage current changes linearly with the temperature. According to this model, the overall power consumption changes with both temperature and supply voltage. The leakage model we adopted in this paper is a similar to this model but slightly different, as introduced later. In Section 4, we also empirically validate this leakage model.

3. System model definitions

In this section we define the system models used in this paper, which include the task model, the processor model, the power model, and the thermal model.

3.1. Task model

We consider two real-time application scenarios. In the first case, we assume that a number of real-time tasks within total execution cycles of c start at time 0 must be finished within the interval of $[0, p]$. Since all tasks have the same deadline, for simplicity, we assume there is only one task with execution cycle of c and deadline of p. In the second case, we further assume that this task is periodic with period of p.

3.2. Processor model

The processor that we consider 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 changes with the circuit temperature dynamically, as given by Eq. (1) [35],

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

where v_t is the threshold voltage, t_d is the circuit delay, and μ and η are technology-related constants, we assume that the processor working frequency in each mode is fixed, and is the one that can accommodate the peak temperature (i.e. by assigning the peak temperature in Eq. (1) across the entire 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.

3.3. Power model

The power consumption of the processor consists of the dynamic power P_{dyn} and the leakage power P_{leak}. P_{leak} changes with both temperature and supply voltage. Specifically, the leakage current for a single transistor I_{leak} can be formulated as follows [35]:

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

where I_s is the leakage current at certain reference temperature and supply voltage, T is the temperature, $\mathcal{A}, \mathcal{B}, \alpha, \beta, \gamma, \delta$ are empirically determined constants. Liu et al. [32] found that using linear approximation method to model the leakage current/temperature dependence can achieve reasonable accuracy with greatly simplified leakage power model. In our work, we adopt this method and simplify the leakage power model as follows:

$$P_{leak}(k) = C_0(k)\, v_k + C_1 T, \tag{3}$$

where $k = 0, \cdots, m-1$ represents m different processor modes. $C_0(k)$ and C_1 are constants that can be obtained by curve fitting for a particular processor under its operating environment conditions. In Section 4, we use empirical study results to justify the appropriateness of this leakage model.

The dynamic power consumption is independent to temperature and in general can be formulated as $P_{dyn} \propto v_k^2 f_k$ [6]. We assume that v_k varies linearly with f_k [6] and thus we model dynamic power as $P_{dyn} \propto v_k^{\xi}$ with $\xi = 3$. Note that setting ξ to other values (as long as $\xi > 1$) does not affect the theorems and conclusions presented in this paper. Therefore the total power consumption at processor mode k can be formulated as

$$P(k) = C_0(k)\, v_k + C_1 \cdot T + C_2\, v_k^3. \tag{4}$$

3.4. Thermal model

We use the lumped RC model similar to Skadron et al. [36] to capture the thermal phenomena of the processor. Specifically, assuming a fixed ambient temperature (T_{amb}), let $T(t)$ denote the temperature at time t. Then we have

$$RC\frac{dT(t)}{dt} + T(t) - RP(t) = T_{amb}, \tag{5}$$

where $P(t)$ denotes the power consumption (in *Watt*) at time t, and R, C denote the thermal resistance (°C/W), and thermal capacitance (in J/°C). We can then scale T such that T_{amb} is zero and get

$$\frac{dT(t)}{dt} = aP(t) - bT(t), \tag{6}$$

where $a = 1/C$ and $b = 1/RC$.

From Eqs. (4) and (6), when a processor running in mode k for interval $[t_0, t_e]$, let the starting temperature be T_0, then solving Eq. (6), the ending temperature can be formulated as below:

$$T_e = \frac{A(k)}{B} + (T_0 - \frac{A(k)}{B})e^{-B(t_e - t_0)} = G(k) + (T_0 - G(k))e^{-B(t_e - t_0)}. \tag{7}$$

where

$$A(k) = a(C_0(k)\, v_k + C_2\, v_k^3), \tag{8}$$

$$B = b - aC_1, \tag{9}$$

and

$$G(k) = \frac{A(k)}{B}. \tag{10}$$

Eqs. (7)–(10) play a critical role in our analytical analysis. For the sake of simplicity, we use A_k and G_k to denote $A(k)$ and $G(k)$, respectively, if there is no confusion.

4. The empirical studies

Considering that the leakage power changes with both temperature and supply voltage, is the constant speed schedule still the optimal choice in minimizing the peak temperature within a specific interval? Before we draw any conclusions, we first launched a number of empirical studies to obtain some intuitions. We also conducted several experiments to justify our leakage power model as well as to study the thermal characteristics of the processor based on our thermal model. For ease of our presentation, we first define several representative real-time schedules as follows.

Definition 1. The *constant-speed schedule* $\widehat{S}(S_c)$ within an interval $[t_0, t_p]$ is the schedule that employs the lowest constant processor speed S_c to complete the workload within the interval.

Definition 2. A *two-speed schedule* $\widehat{S}(S_1, S_2)$ within an interval $[t_0, t_p]$ is the schedule that uses the two different speeds S_1 and S_2 with at-most two transitions between the speed levels to complete the workload within the interval.

We further define four different types of two-speed schedules.

Definition 3. A *dip schedule* $\widehat{S}(S_1, S_2)$ within an interval $[t_0, t_p]$ is a two-speed schedule that uses S_1 during the interval $[x_1, x_2] (t_0 \leqslant x_1 < x_2 \leqslant t_p)$, and S_2 in the rest of the intervals, with $S_1 < S_2$.

Definition 4. A *hump schedule* $\widehat{S}(S_1, S_2)$ within an interval $[t_0, t_p]$ is a two-speed schedule that uses S_2 during the interval $[x_1, x_2] (t_0 \leqslant x_1 < x_2 \leqslant t_p)$, and S_1 in the rest of the intervals, with $S_1 < S_2$.

Definition 5. A *step-up schedule* $\widehat{S}(S_1, S_2)$ within an interval $[t_0, t_p]$ is a two-speed schedule that uses S_1 during the interval $[t_0, x]$, and uses S_2 in interval $[x, t_p]$ with $S_1 < S_2$.

Definition 6. A *step-down schedule* $\widehat{S}(S_1, S_2)$ within an interval $[t_0, t_p]$ is a two-speed schedule that uses S_2 during the interval $[t_0, x]$, and uses S_1 in the interval $[x, t_p]$ with $S_1 < S_2$.

Fig. 1 shows an example of different speed schedules defined above. Note that, according to Definition 2, once the two speeds and total workload within the interval are defined, the total length of the interval to run processor with each speed is also defined (e.g. t_1 and t_2 in Fig. 1). In what follows, we present several empirical results that help to obtain some intuitions on the applicability—in the context of peak temperature minimization—of the two power reduction principles mentioned above.

Empirical study 1: first, we wanted to verify if two principles listed before are still valid in terms of the peak temperature minimization *for a given interval*. We constructed our processor similar to the one shown in [37,34], based on the 65 nm technology and with the conventional air cooling option of $Rth = 0.8^{\circ}C/W, Cth = 340\ J/^{\circ}C$ [1]. We assumed that the processor can run on four active modes i.e. 0.95, 1.0, 1.05 and 1.10 V, and one shut-down mode. The corresponding frequency was calculated using Eq. (1). The values of the remaining parameters are taken from [34]. The ambient temperature was set to 25 °C and we assume the processor's starting temperature is the same as the ambient temperature.

We selected three available processor speeds with corresponding supply voltages as 0.95 V, 1.0 V, and 1.05 V, respectively. Five different types of schedules, i.e. the constant-speed schedule, the step-down, the step-up schedule, the dip schedule and the hump schedule were constructed that run the same length and complete same workload. For dip and hump schedule, the value of x_1 (see Fig. 1) was randomly selected. We then varied the interval length from 10 to 2000 s to get different schedules. For each test case, the highest temperature within the corresponding interval by each schedule was collected and plotted in Fig. 2(a).

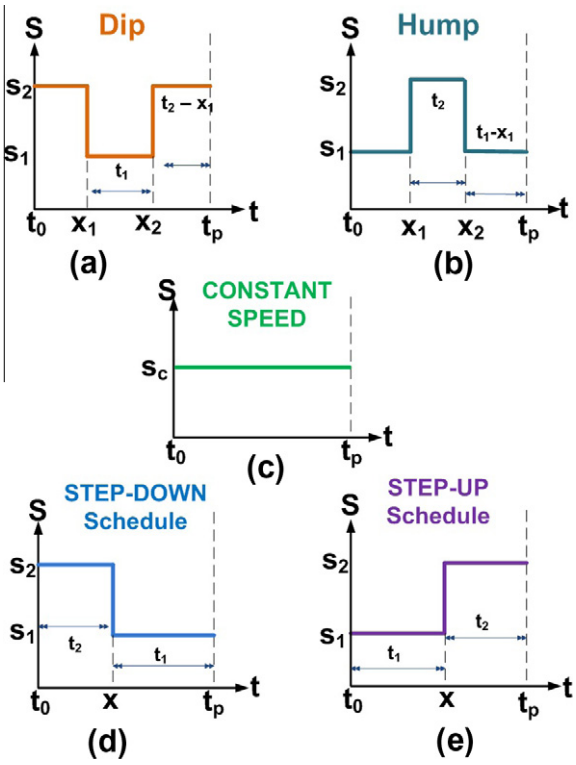

Fig. 1. Different speed schedules: (a) the dip schedule; (b) the hump schedule; (c) the constant schedule; (d) the step-down schedule; (e) the step-up schedule.

As can be seen from Fig. 2(a), while the maximum temperature using the step-up schedule is always higher than that by the constant-speed schedule, the peak temperature by the other two-speed schedules can in fact be lower sometimes. For instance, when the period is equal to 700 s, the peak temperature of the step-down, the constant, the hump, the dip and the step-up speed schedules are 37.84 °C, 43.95 °C, 45.07 °C, 46.5 °C and 47.32 °C, respectively. This result clearly contradicts the conclusion that the constant-speed schedule is the optimal schedule in terms of minimizing the peak temperature within a given interval. In the meantime, we can also observe that the peak temperature by the step-up schedule is indeed consistently higher than that of other types of schedules.

We further studied if using the two closest neighboring speeds is the best choice in terms of peak temperature reduction within a given interval. Two step-down speed schedules *Sa* and *Sb* were constructed. *Sa* uses the speed corresponding to supply voltages 0.95 and 1.05 V for low and high speed, respectively, and *Sb* uses speeds corresponding to 0.95 and 1.10 V. Both *Sa* and *Sb* run the same length and complete same workload. As done before, we then varied the interval length from 10 to 2000 s to get different sched-

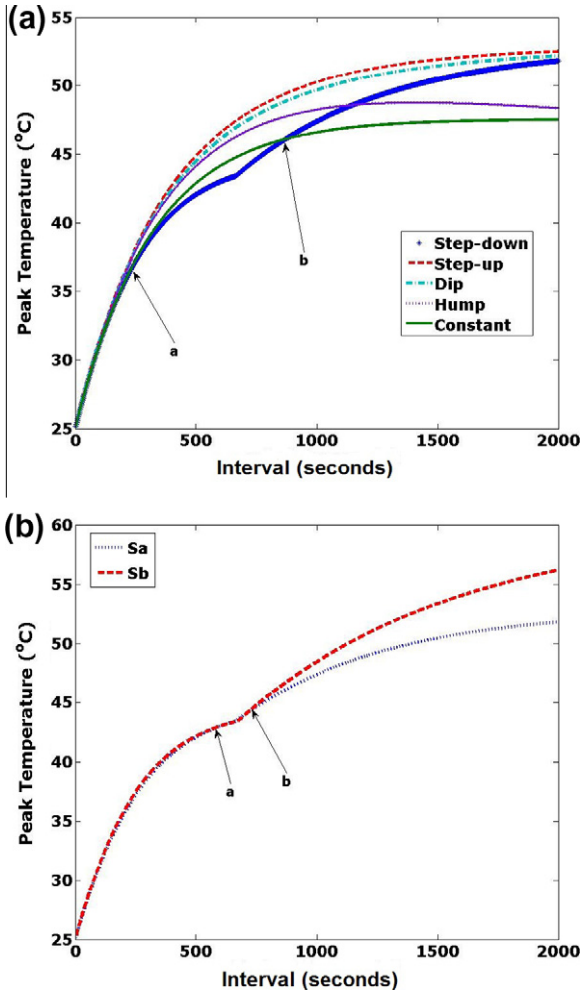

Fig. 2. The peak temperatures by different schedules within a given interval. (a) The peak temperature by the step down schedule is lower than that by the constant schedule for interval length between *a* and *b*. (b) The peak temperature by the schedule (*Sa*) using the neighboring two speeds is higher than the one (*Sb*) using the two non-neighboring speeds for interval length between *a* and *b*.

ules. For each test case, the highest temperature within the corresponding interval by each schedule was collected and plotted in Fig. 2(b). When comparing their peak temperatures, as shown in Fig. 2(b), the step-down schedule *Sa* is not necessarily always better than *Sb*. When the period is set to 700 s, the peak temperature for *Sa* is 43.65 °C, while for *Sb* it is 43.56 °C. This result seems to also imply that the second principle is not valid either in terms of the peak temperature reduction.

Empirical study 2: we next want to verify if the two principles listed before can be used to minimize the peak temperature *when the processor temperature reaches its stable status*.

We constructed the five different schedules, i.e., the constant-speed schedule, the step-down, the dip schedule, the hump schedule and the step-up schedule the same as above, and ran each schedule not one time but periodically until the temperature became stable and do not seem to change anymore. We then varied the periods, collected the maximum temperature for each case and plotted in Fig. 3(a). From this figure, we can clearly see that the constant-speed schedule always lead to the lowest peak temperature in our experiment, and the peak temperatures by remaining two-speed schedules eventually become the same.

In addition, when we constructed the two step-down schedules *Sa* and *Sb* as above and ran them periodically. From Fig. 3(b), we can see that the step-down schedule *Sa* using the two closest neighboring speeds is always better than *Sb*. Our empirical results here seem to suggest that the two principles listed before may be valid in terms of minimizing peak temperature when the processor reaches the stable status.

These empirical results suggest that using a constant speed in a schedule, or using the neighboring speeds when the constant speed is not available, is still the best way to minimize the peak temperature when the temperature reaches its *stable status*. In the next few sections, we formulate these findings into theorems and prove them formally. Before we introduce the theorems and their proofs, we first use empirical results to justify our leakage model and to establish some useful thermal characteristics of our processor model.

Empirical study 3: in Section 3, we have introduced a simplified linear leakage power model to model the relationship of the leakage, the temperature, and the supply voltages. One immediate question is how accurate this model is? In this empirical study, we use an existing processor model drawn from the existing literature [35] to study this problem. The processor model is same as used in Empirical study 1 and 2 with conventional air cooling. However, in this study, we assumed that the processor can run on 15 active modes i.e. 0.60–1.3 V, with step size of 0.05 V and one shut-down mode. The corresponding frequency was calculated using Eq. (1). The values of the remaining empirical and technology parameters are taken from [34,1]. The ambient temperature was set to 25°C and we assume the processor's starting temperature is the same as the ambient temperature. Based on this processor model, we compared several existing leakage models.

- The *Actual* leakage model [35]: $P_{leak}(k) = I_{leak} v_k$ where I_{leak} is defined in Eq. (2).
- The simple *Linear* leakage model (e.g. [29,22,24,23]): $P_{leak}(k) = C_0 + C_1 T$ with C_0, C_1 being constants;
- LK_{TV} model [37]: $P_{leak}(k) = I_{leak}(k) v_k$ and $I_{leak}(k) = C_0 + C_1 T$.
- LK_T model: the model defined in Section 3.

Based on the experimental settings stated above, Fig. 4 depicts the leakage power consumptions under different supply voltages and temperatures according to different leakage models. As we can see from Fig. 4, when the supply voltage varies, the leakage power consumption varies dramatically. For instance, if we consider the *Actual* leakage model at 50 °C, the leakage power becomes almost double when supply level changes from 0.95 to 1.1 V. Therefore, the simple *Linear* leakage model can only be used when the supply voltage cannot be changed. Otherwise, large discrepancies between the actual leakage power consumption (e.g. the results according to the *Actual* leakage model) and the estimated one with this model may occur. On the other hand, we can see that both LT_{TV} and LK_T match the actual leakage power consumption well, with relative errors under 4% in our study. These results clearly show that LK_T model is a leakage model with very good accuracy.[1]

Furthermore, in order to conduct analytical analysis based on temperature dynamics in Eq. (7), it is highly desirable that the characteristics of G_k is known. However, since G_k is determined essentially by the curve-fitting constants C_0 and C_1, it is difficult to analytically study its properties. Therefore we study its attributes empirically.

Fig. 5 plots the characteristics of the function $G(k)$ under different operating conditions i.e. (a) conventional air cooling option, (b) water spray cooling. As illustrated in Figs. 5(a) and (b), we can clearly see that, for both cooling conditions, the function G_k is a po-

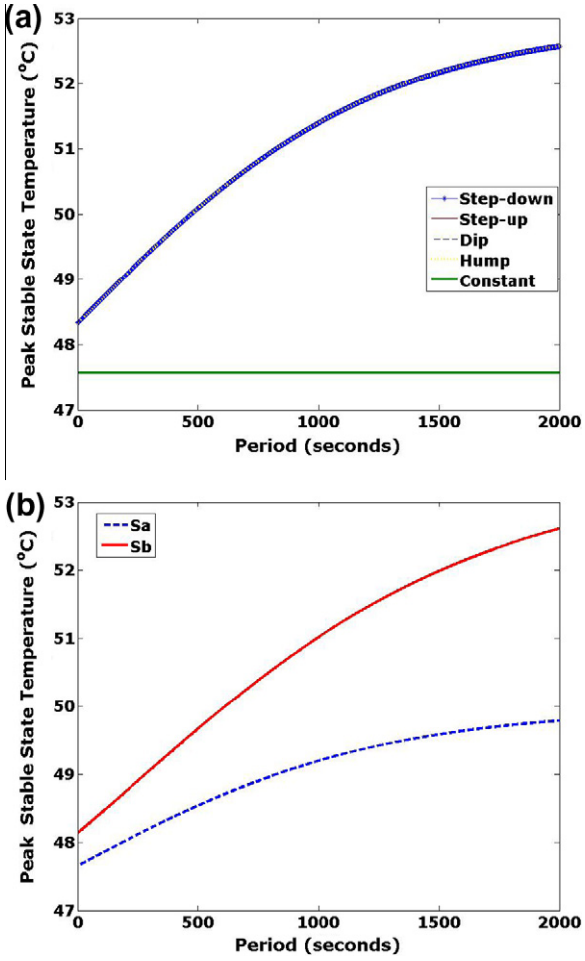

Fig. 3. Peak temperatures at the stable status by different schedules. (a) The peak temperature by the constant speed is consistently lower than others. (b) The peak temperature by the schedule using neighboring speeds is consistently lower than the one using non-neighboring speeds.

[1] According to our empirical results, the LT_{TV} model is more accurate than LK_T model. However, we are not able to formally prove all theorems in this paper based on model LT_{TV}.

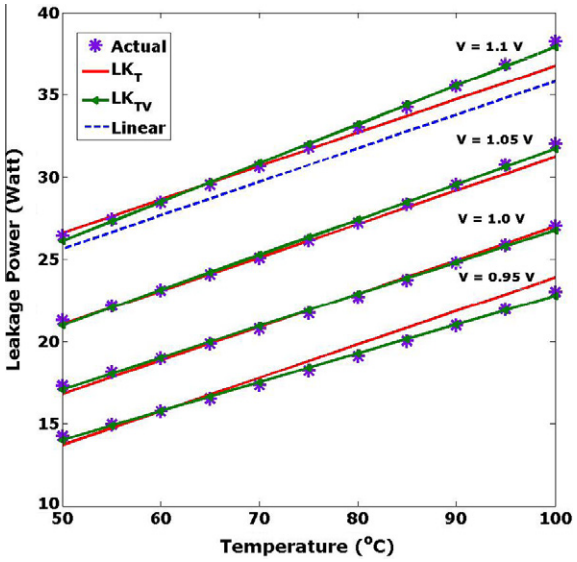

Fig. 4. Leakage power consumptions calculated using different leakage models under different temperatures and supply voltages.

sitive, monotonic increasing, and convex function of the supply voltage. Also, from Eq. (9), we can see that it is necessary that $B > 0$, or the temperature will run away otherwise. Therefore, in what follows, unless otherwise specified, we assume that

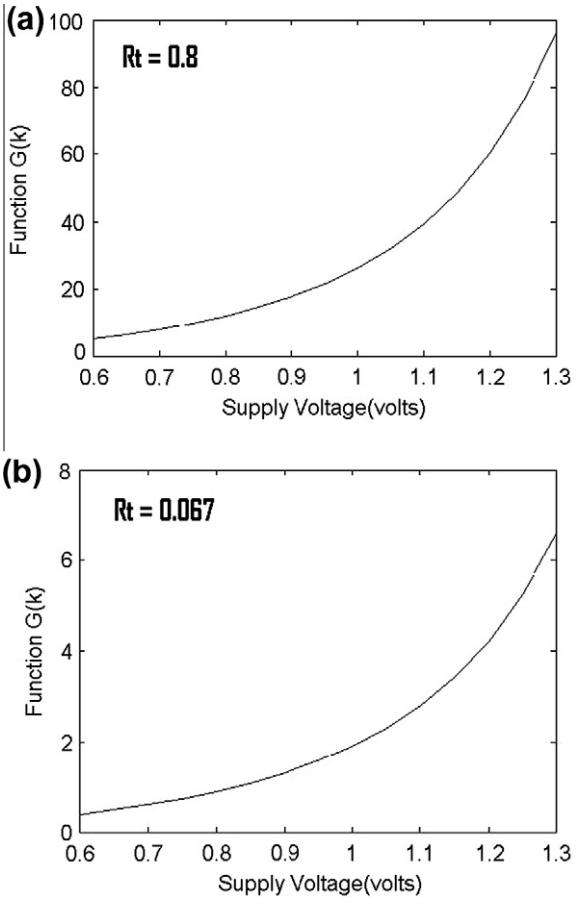

Fig. 5. Function $G(v)$ under different operating conditions (a) Conventional air cooling. (b) Water spray cooling.

- G_k (or $G(v_k)$) is a *positive*, *monotonic increasing*, and *convex function* of k (or v_k), respectively;
- $B > 0$.

Our empirical results presented above reveal some strong and interesting findings. In the following sections, we formulate these findings into theorems and formally prove them.

5. Peak temperature minimization within a specified interval

The empirical study 1 discussed in previous section shows that a constant speed schedule is not the optimal method for the peak temperature reduction within a specified interval. Then the question becomes what the optimal schedule is. To answer this question, in what follows, we formulate several theorems from our empirical study and prove them analytically. These theorems provide us with some insights and guiding principles when developing better DVS schedules for peak temperature minimization within a given interval. Specifically, Theorem 1 characterizes the peak temperature obtained using the step-up schedule within a given interval.

Theorem 1. *Given two processor speeds S_1 and S_2 with $S_1 < S_2$ and a hard real-time job J, the step-up schedule $(\widehat{S}(S1, S2))$ has the highest peak temperature among all two-speed schedules within the same interval if the initial temperature $T_0 < G_1$. Where G_k is defined in Eq. (10).*

Proof. We first compare the peak temperature between the step-up and step-down schedules as shown in Fig. 1(d) and (e). Let T_u be the peak temperature of step-up schedule, which always occurs at the end of the interval [16]. Let t_1 and t_2 be the interval length that the processor runs at speed S_1 and S_2, respectively. From Fig. 1 and based on Eqs. (7)–(10), we have

$$T_u = G_2(1 - e^{-Bt_2}) + G_1(1 - e^{-Bt_1})e^{-Bt_2} + T_0 e^{-B(t_2+t_1)} \tag{11}$$

According to Fig. 1(d), we can see that, when using the step-down schedule, the peak temperature will occur either at point x or at t_p. Therefore to prove this theorem we need to consider two cases:

- *Case 1: the peak temperature of the step-down schedule appears at point t_p.*
  With the starting temperature T_0, the temperature of the step-down schedule T_d at t_p is given by

$$T_d = G_1(1 - e^{-Bt_1}) + G_2(1 - e^{-Bt_2})e^{-Bt_1} + T_0 e^{-B(t_1+t_2)} \tag{12}$$

To show that $T_u > T_d$, by canceling $T_0 e^{-B(t_1+t_2)}$ from both Eqs. (12) and (11), we have

$$G_2(1 - e^{-Bt_2}) + G_1(1 - e^{-Bt_1})e^{-Bt_2} > G_1(1 - e^{-Bt_1}) + G_2(1 - e^{-Bt_2})e^{-Bt_1}. \tag{13}$$

or

$$G_2 > G_1 \tag{14}$$

As G_i is a monotonically increasing function, Eq. (14) is true. Hence, we proved that the step-up schedule results in a higher peak temperature than the step-down schedule when its peak temperature appears at point t_p.

- *Case 2: the peak temperature of step-down schedule T_d appears at point x.*
  The temperature at point x by step-down speed schedule can be formulated as

$$T_d = G_2(1 - e^{-Bt_2}) + T_0 e^{-Bt_2}. \quad (15)$$

To show that $T_u > T_d$, based on Eqs. (15) and (11), we only need to show

$$G_2(1 - e^{-Bt_2}) + G_1(1 - e^{-Bt_1})e^{-Bt_2} + T_0 e^{-B(t_1+t_2)} > G_2(1 - e^{-Bt_2}) + T_0 e^{-Bt_2}. \quad (16)$$

or

$$(G_1 - T_0)e^{-Bt_2} > (G_1 - T_0)e^{-B(t_1+t_2)} \quad (17)$$

As $T_0 < G_1$, Eq. (17) is true. Hence we proved that the step-up schedule always results in higher peak temperature than step-down schedule when its peak temperature appears at point x.

From above, we can conclude that the step-up schedule always result in a higher peak temperature compared to the step-down schedule within a given interval for $T_0 < G_1$.

We now compare the step-up schedule with other types of two-speed schedules. We first compare it with the hump schedule as shown in Fig. 6(a). Let T_x be the temperature at $t = x_1$. Based on Eq. (7), we have

$$T_x = G_1 + (T_0 - G_1)e^{-Bx}. \quad (18)$$

Since $T_0 < G_1$, we have $T_x < G_1$. Moreover, in Fig. 6(a), let $T_u(x_1)$ and $T_f(x_1)$ denote the peak temperature within the interval $[x_1, p]$ by the step-up schedule and the hump schedule, respectively. Since $T_x < G_1$, and from the first part of the proof we know that the step-up schedule always incurs a higher peak temperature than that of a step-down schedule, we immediately prove that $T_f(x) \leqslant T_u(x)$. Similar conclusions can be proved for the dip schedule shown in Fig. 6(b). □

Theorem 1 helps to identify the two-speed schedule that potentially leads to the highest peak temperature. It would be interesting if we can also identify the schedule that potentially leads to the lowest peak temperature. Theorem 2 can be used for this purpose.

Theorem 2. *Given two processor speeds S_1 and S_2 with $S_1 < S_2$ and a hard real-time job J, the step-down schedule $(\widehat{S}(S1, S2))$ has the lowest peak temperature among all two-speed schedules within the same interval if the initial temperature $T_0 < min(G_1, (G_1 - G_2)e^{Bx} + G_2)$, where x is the length of the interval using S_2. Where B and G_k are defined in Eqs. (9) and (10), respectively.*

Proof. Theorem 1 already proves that the peak temperature of the step-up schedule is always higher than the step-down schedule under the same given condition, so we only need to compare the peak temperature between the step-down schedule with that by hump and dip schedules.

Consider Fig. 7(a). Both the step-down schedule and the hump schedule use the same speed between x_2 and p. Also the initial temperature $T_0 < G_1$. Therefore, according to Theorem 1, we conclude that the peak temperature by the step down schedule is lower than that by the hump schedule.

Now consider Fig. 7(b). Let the temperature at $t = x_1$ be T_{x1}. Then based on Eq. (7), we have

$$T_{x1} = G_2 + (T_0 - G_2)e^{-Bx1}. \quad (19)$$

At the same time, since $T_0 < (G_1 - G_2)e^{Bx} + G_2$, we have

$$T_{x1} < G_2 + (G_1 - G_2)e^{B(x-x1)}. \quad (20)$$

Since $G_1 < G_2$ and $e^{B(x-x1)} > 1$, we have

$$T_{x1} < G_2 + (G_1 - G_2) = G_1. \quad (21)$$

Therefore, according to Theorem 1, we conclude that the peak temperature by the step-down schedule is lower than that by the dip schedule. □

Furthermore, our empirical study shows that the conclusion that the constant-speed schedule is the optimal choice in terms of peak temperature minimization within a given interval is not true anymore. Even though our empirical results show that in most cases the constant-speed schedule is a better choice, it can be inferior to a step-down schedule sometimes. In Theorem 3, we formu-

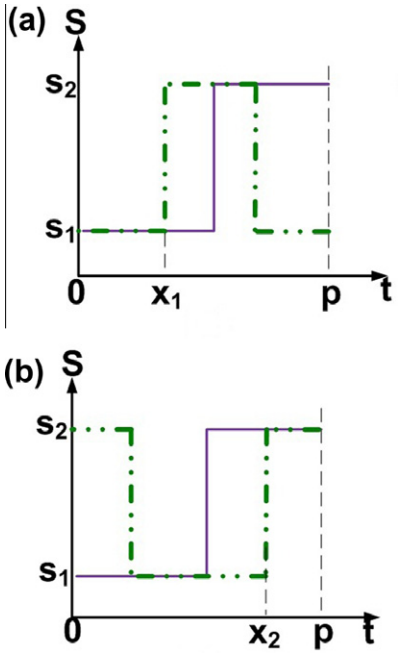

Fig. 6. (a) The peak temperature by the hump and step-up schedule. (b) The peak temperature by the dip and step-up schedule.

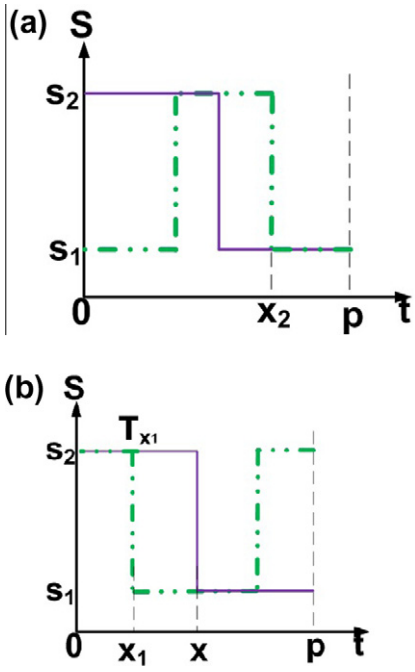

Fig. 7. (a) The peak temperature by the hump and step-down schedule. (b) The peak temperature by the dip and step-down schedule.

late this conclusion and present the conditions when a constant-speed schedule becomes inferior to a step-down schedule in terms of peak temperature reduction.

Theorem 3. *Given a constant-speed schedule $\widehat{S}(S_1)$ and a step-down schedule $\widehat{S}(S_0, S_2)$ for a hard real-time job J. Assuming $T_0 < min(G_1, (G_1 - G_2)e^{Bx} + G_2)$ and $S_0 < S_1 < S_2$. Let $T_m(\widehat{S}(S_1))$ and $T_m(\widehat{S}(S_0, S_2))$ be the peak temperature by $\widehat{S}(S_1)$ and $\widehat{S}(S_0, S_2)$ within the interval [0, p], respectively. Then,*

$$T_m(\widehat{S}(S_1)) > T_m(\widehat{S}(S_0, S_2)) \tag{22}$$

if and only if

- $\frac{1}{B} ln(\frac{G_2 - T_0}{G_2 - G_0}) < x < \frac{1}{B} ln(\frac{G_2 - T_0}{G_2 - G_1(1 - e^{-Bp}) - T_0 e^{-Bp}})$; *or,*
- $x < min(\frac{1}{B} ln(\frac{G_2 - T_0}{G_2 - G_0}), p - \frac{1}{B} ln(\frac{G_2 - G_0}{(G_1 - G_0) + (G_2 - G_1)e^{-Bp}}))$.

where $S_1 p = S_2 x + S_0(p - x)$, and B, G_k are defined in Eq. (8) and (10), respectively.

Proof. From Fig. 8, let T_x and T_p denote the temperatures of the step-down schedule at $t = x$ and $t = p$, respectively. Let T_c be the peak temperature of the constant-speed schedule. Based on Eq. (7), we have

$$T_x = G_2(1 - e^{-Bx}) + T_0 e^{-Bx}. \tag{23}$$

$$T_p = G_0(1 - e^{-B(p-x)}) + G_2(1 - e^{-Bx})e^{-B(p-x)} + T_0 e^{-Bp} \tag{24}$$

$$T_c = G_1(1 - e^{-Bp}) + T_0 e^{-Bp} \tag{25}$$

Note that the peak temperature of the step-down schedule must be either T_x or T_p. Then Eq. (22) becomes true only when

- $T_x < T_c$ when $T_x > T_p$, or
- $T_p < T_c$ when $T_p > T_x$

Replace T_x, T_c and T_p with Eqs. (23)–(25) and solve for x, we can prove the theorem. □

As implied by Theorem 3, neither the constant speed schedule nor any particular two-speed schedule is always the optimal schedule to minimize the peak temperature within a given interval. On the other hand, however, Theorems 2 and 3 help to identify the optimal schedules that can potentially lead to the optimal DVS schedule to minimize the peak temperature within an interval. Both Theorems 2 and 3 target a single real-time job. Since most real-time tasks are repetitive in nature, it is highly desirable we can explore some fundamental principles for the periodic tasks as well.

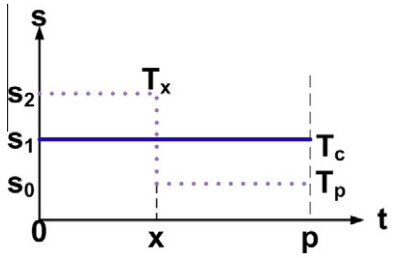

Fig. 8. The constant-speed schedule and a step-down schedule within a given interval.

6. Peak temperature minimization at the stable state

In this section, we extend our research from a single real-time job to a periodic real-time task. When running a real-time task set periodically, unless the processor temperature "runs away" [35], the processor temperature is eventually stabled. The *stable status* is defined as below.

Definition 7. [37]When running a periodic task with period p, the temperature at the processor is called to be *stable* if for a given threshold, i.e. $0 < \varepsilon \ll 1$,

$$|T((n+1)p) - T(np)| < \varepsilon, \tag{26}$$

where $n \geqslant 0, n \in Z$, and $T(t)$ is the temperature at t.

Similarly, we want to investigate the validity of applying *Principle* 1 and 2 in the context of minimizing peak temperature when scheduling a periodic task set.

We first present two theorems that act as the basis in formulating the key principles of peak temperature minimization when the processor temperature becomes stable.

Theorem 4. *Given a hard real-time periodic task τ, the maximum temperature when the processor temperature reaches its stable status does not depend upon the initial temperature.*

Proof. Let us consider a step-down speed schedule shown in Fig. 1, where S_2 and S_1 denotes the high speed and low speed. From Fig. 1, t_1 and t_2 denotes the duration of S_1 and S_2 in the first period.

Based on Eq. (7), the temperature at $t = x$ and $t = t_p$ can be formulated as

$$T_x = G_2 + (T_0 - G_2)e^{-Bt_2}, \quad T_{t_p} = G_1 + (T_x - G_1)e^{-Bt_1}$$

where B and G_k are defined in Eqs. (9) and (10), respectively.

From [37], the maximal temperature at the stable state temperature can be formulated as

$$T_{max} = max(T_x^{\infty}, T_{tp}^{\infty})$$

where,

$$T_x^{\infty} = \frac{G_2(1 - e^{-Bt_2})}{1 - e^{-Bt_2}} = G_2 \tag{27}$$

$$T_{tp}^{\infty} = \frac{G_1(1 - e^{-Bt_1}) + G_2(1 - e^{-Bt_2})e^{-Bt_1}}{1 - e^{-B(t_1 + t_2)}} \tag{28}$$

As can be seen from Eqs. (27) and (28), no matter if the maximal temperature occur at t = x or t=t_p, it does not depend upon the initial temperature T_0. Similar conclusion can be achieved using other speed schedules. □

Based on Theorem 4, we can show that the maximum peak temperatures at the stable state with any periodic two-speed schedule are the same.

Theorem 5. *Given a real-time periodic task τ and two processor speeds, then the maximal temperature at the stable status with any two-speed schedule using the same two speeds are the same.*

Proof. Consider a periodic step-up speed schedule shown in Fig. 9. From [37], we can calculate the stable state temperature or the peak temperature as:

$$T_{max} = \frac{G_2(1 - e^{-Bt_2}) + G_1(1 - e^{-Bt_1})e^{-Bt_2}}{1 - K} \tag{29}$$

where t_1 and t_2 denotes duration of low speed and high speed. B and G_i are defined in Eqs. (9) and (10), respectively and

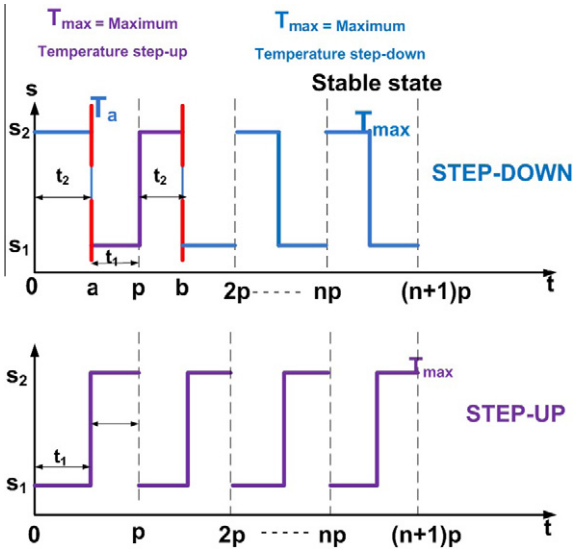

Fig. 9. Stable temperature for step-down and step-up schedule.

$K = e^{-B(t_1+t_2)}$.

From Fig. 9 we can see that the step-down schedule shown is same as the periodic step-up schedule but with an initial temperature T_a. Similarly, all other periodic two-speed schedules can be viewed as the step-up schedule with an initial shift. Therefore their peak temperatures are equivalent to the one with the periodic step-up schedule with a different initial temperature. Since Theorem 4, already proves that the stable temperature does not depend on the starting temperature, the conclusion is proved. □

Based on the conclusions from Theorems 4 and 5, we can now formulate an important theorem for the problem of scheduling hard real-time periodic task, with the goal of peak temperature minimization.

Theorem 6. *Given a real-time periodic task τ, the maximum temperature at the stable state is minimized when running τ using the lowest constant-speed.*

Proof. From Theorem 5, we know that at *stable status* all the two-speed schedules result in the same peak temperature. Therefore, to prove this theorem we will compare constant-speed schedule with any two-speed schedule. Let T_c^∞ and T_u^∞ denote the maximum stable temperature for the constant-speed (S_1) and the step-up schedule ($S_0 < S_2$), respectively (Fig. 10). Without loss of generality, we can assume $p = 1$. Also from the conclusion of Theorem 4, we know the stable temperature does not depend on the initial temperature, therefore we assume the initial temperature to be zero. Then we have

$$T_c^\infty = \frac{G_1(1-e^{-B})}{1-e^{-B}} = G_1 \tag{30}$$

$$T_u^\infty = \frac{G_2(1-e^{-B(1-x)}) + G_0(1-e^{-Bx})e^{-B(1-x)}}{1-e^{-(B(1-x)+Bx)}} \tag{31}$$

To show that $T_c^\infty \leqslant T_u^\infty$, we only need to show that

$$G_1 \leqslant kG_0 + (1-k)G_2, \tag{32}$$

where

$$k = \frac{e^{-B(1-x)} - e^{-B}}{1-e^{-B}}, 1-k = \frac{1-e^{-B(1-x)}}{1-e^{-B}}. \tag{33}$$

Since

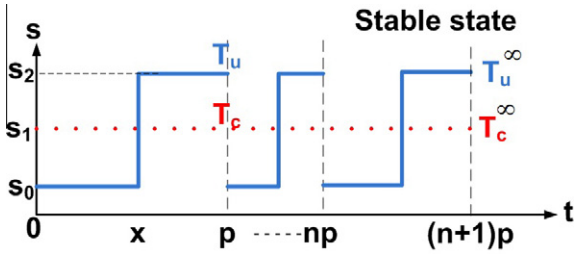

Fig. 10. Stable temperature for step-up and constant-speed schedule.

$$S_1 = S_0x + S_2(1-x), \tag{34}$$

and $B > 0$ and G_i is a convex function, we have

$$G_1 \leqslant xG_0 + (1-x)G_2. \tag{35}$$

Therefore, to show that Eq. (32) holds, we only need to show that

$$xG_0 + (1-x)G_2 \leqslant kG_0 + (1-k)G_2, \tag{36}$$

or

$$(G_0 - G_2)(x-k) \leqslant 0. \tag{37}$$

As $G_0 \leqslant G_2$, we only need to prove that

$$x \geqslant k = 1 - \frac{1-e^{-B(1-x)}}{1-e^{-B}}. \tag{38}$$

Or, equivalently,

$$\frac{1-e^{-B(1-x)}}{1-e^{-B}} \geqslant 1-x. \tag{39}$$

Now consider function

$$F(z) = \frac{1-e^{-Bz}}{1-e^{-B}} - z. \tag{40}$$

with $0 \leqslant z \leqslant 1$. We can readily show that function $F(z)$ is a concave function since $F''(z) < 0$. Note that the curve $F(z)$ passes two points, i.e. $(0,0)$ and $(1,0)$, as $F(0) = 0$ and $F(1) = 0$. Let $H(z)$ be the line that crosses these two points. Since $F(z)$ is concave, we have $F(z) \geqslant H(z) \geqslant 0$ for $0 \leqslant z \leqslant 1$.

We therefore prove that the constant speed schedule always outperforms a step-up periodic schedule in minimizing the peak temperature when the temperature reaches the *stable status*. In Theorem 5, we have already proved that at *stable status* the peak temperature of step-up and any other two-speed are the same. Hence we can immediately conclude that at the stable state the constant-speed outperforms any two-speed schedule in peak temperature reduction. □

Moreover, since there are only a small number of processor speeds available. A constant-speed schedule is not always available and two or more speeds have to be used. In that case, we show that a principle that is similar to *Principle 2* stated previously can also be established.

Theorem 7. *If a two-speed schedule is used for a hard real-time periodic task, then the one that uses the two closest neighboring speeds minimizes the maximum temperature at the stable state.*

Proof. Consider interval $[0,p]$ and step-up schedules $\hat{S}_1(s_1,s_4)$ and $\hat{S}_2(s_2,s_3)$ shown in Fig. 11. Without loss of generality, we assume $s1 \leqslant s2 < s3 \leqslant s4$. Let $T(\widehat{S})$ represent the maximal temperature at the *stable status* with schedule $\widehat{S}$. We want to show that $T(\widehat{S}(s_2,s_3)) \leqslant T(\widehat{S}(s_1,s_4))$. Let the speed change occur at x in $\hat{S}_2(s_2,s_3)$. Consider another schedule $\hat{S}_3(s_2,s_4)$ and let its speed

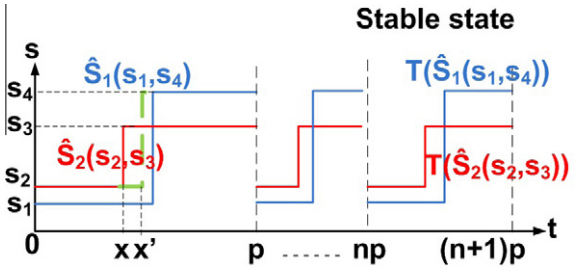

Fig. 11. Step-up schedules $\widehat{S}_2(s_2,s_3)$ and $\widehat{S}_3(s_2,s_4)$ for a real-time periodic task.

change at x'. Then we have $x \leqslant x'$. Note that $\widehat{S}_2$ and $\widehat{S}_3$ complete the same workload within interval $[0,x]$ with the same speed, but $\widehat{S}_2$ uses a constant speed to complete the rest of the interval and $\widehat{S}_3$ uses two different speeds. From Theorem 6, we can immediately conclude that $T(\widehat{S}(s_2,s_3)) \leqslant T(\widehat{S}(s_2,s_4))$. Similarly, we can prove that $T(\widehat{S}(s_2,s_4)) \leqslant T(\widehat{S}(s_1,s_4))$. Therefore, $T(\widehat{S}(s_2,s_3)) \leqslant T(\widehat{S}(s_1,s_4))$. □

Note that, even though Theorems 6 and 7 look very similar to the two basic principles that have been widely used for dynamic energy reduction, it does not necessarily imply that the existing energy reduction techniques can be readily applied for the purpose of peak temperature minimization. On the other hand, Theorem 1 to Theorem 7 present some fundamental guidelines in the development of new DVS schedule techniques that can minimize the peak temperature.

7. Summary

In deep sub-micron domain, the thermal management is becoming a critical issue in design of modern computing systems. Also as both leakage and thermal issues become more prominent, a power and thermal aware design technique becomes less effective if the leakage/temperature dependency is not appropriately addressed.

In this paper, we incorporate the leakage/temperature/supply voltage dependency into the real-time scheduling analysis that aims at minimizing the peak temperature. We show that a constant-speed schedule is not always the optimal schedule in terms of peak temperature minimization for a given interval. We further show that for a given periodic task, the lowest constant-speed is the optimal schedule among all two-speed schedules to minimize the peak temperature at the stable state. If this constant-speed is not available, then the schedule that uses the two closest neighboring speeds is the best choice. These new findings and theorems form the basis for the future study of developing more effective power and thermal aware scheduling techniques for more complicated architectures and real-time systems.

Acknowledgments

We thank anonymous reviewers for their comments and suggestions which contributed significantly to improving quality of this paper. This work is supported in part by NSF under projects CNS-0969013, CNS-0917021, CNS-1018108, CNS 1018731 and CNS CAREER 0746643.

References

[1] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, D. Tarjan, Temperature-aware microarchitecture, ICSA (2003) 2–13.
[2] L.-T. Yeh, R.C. Chu, Thermal Management of Microelectronic Equipment: Heat Transfer Theory, Analysis Methods, and Design Practices, ASME Press, New York, NY, 2002.
[3] J. Markoff, Intel's big shift after hitting technical wall, New York Times (2004).
[4] K. Skadron, M. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, D. Tarjan, Temperature-aware computer systems: opportunities and challenges, IEEE Micro 23 (2003) 52–61.
[5] N. Bansal, T. Kimbrel, K. Pruhs, Speed scaling to manage energy and temperature, Journal of the ACM 54 (2007) 1–39.
[6] J. Rabaey, A. Chandrakasan, B. Nikolic, Digital Integrated Circuits: A Design Perspective, Prentice Hall, 2003.
[7] ITRS, International Technology Roadmap for Semiconductors, 2009 ed., International SEMATECH, Austin, TX. Available at: <http://public.itrs.net/>.
[8] F. Yao, A. Demers, S. Shenker, A scheduling model for reduced cpu energy, FOCS (1995) 374–382.
[9] T. Ishihara, H. Yasuura, Voltage scheduling problem for dynamically variable voltage processors, ISLPED (1998) 197–202.
[10] P. Pillai, K.G. Shin, Real-time dynamic voltage scaling for low-power embedded operating systems, SOSP (2001) 89–102.
[11] G. Quan, Y. Zhang, W. Wiles, P. Pei, Guaranteed scheduling for repetitive hard real-time tasks under the maximal temperature constraint, ISSS+CODES (2008) 267–272.
[12] S. Zhang, K.S. Chatha, Approximation algorithm for the temperature-aware scheduling problem, ICCAD (2007) 281–288.
[13] K. Skadron, M.R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, D. Tarjan, Temperature-aware microarchitecture: modeling and implementation, The ACM Transactions on Architecture and Code Optimization . 1 (2004) 94–125.
[14] Hotspot 4.2 temperature modeling tool, University of Virgina, 2009. Available at: <http://lava.cs.virginia.edu/HotSpot>.
[15] S. Wang, R. Bettati, Reactive speed control in temperature-constrained real-time systems, ECRTS (2006) 161–170.
[16] V. Chaturvedi, H. Huang, G. Quan, Leakage aware scheduling on maximal temperature minimization for periodic hard real-time systems, ICESS (2010) 1802–1809.
[17] P. Kumar, L. Thiele, Thermally optimal stop-go scheduling of task graphs with real-time constraints, ASPDAC (2011) 123–128.
[18] M. Bao, A. Andrei, P. Eles, Z. Peng, Temperature-aware idle time distribution for energy optimization with dynamic voltage scaling, DATE (2010) 21–26.
[19] H. Huang, G. Quan, Leakage aware energy minimization for real-time systems under the maximum temperature constraint, DATE (2011) 1–6.
[20] V. Hanumaiah, S. Vrudhula, K. Chatha, Maximizing performance of thermally constrained multi-core processors by dynamic voltage and frequency control, ICCAD (2009) 310–313.
[21] C. Yang, J. Chen, L. Thiele, T. Kuo, Energy-efficient real-time task scheduling with temperature-dependent leakage, DATE (2010) 9–14.
[22] T. Chantem, X.S. Hu, R. Dick, Online work maximization under a peak temperature constraint, ISLPED (2009) 105–110.
[23] S. Zhang, K.S. Chatha, Thermal aware task sequencing on embedded processors, DAC (2010) 585–590.
[24] R. Jayaseelan, T. Mitra, Temperature aware task sequencing and voltage scaling, ICCAD (2008) 618–623.
[25] S. Liu, M. Qiu, Thermal-aware scheduling for peak temperature reduction with stochastic workloads, RTAS WiP (2010).
[26] P. Kumar, L. Thiele, Cool shapers: shaping real-time tasks for improved thermal guarantees, DAC (2011) 468–473.
[27] L. He, W. Liao, M.R. Stan, System level leakage reduction considering the interdependence of temperature and leakage, DAC (2004) 12–17.
[28] L. Yuan, S. Leventhal, G. Qu, Temperature-aware leakage minimization technique for real-time systems, ICCAD (2006) 761–764.
[29] J.-J. Chen, S. Wang, L. Thiele, Proactive speed scheduling for real-time tasks under thermal constraints, RTAS (2009) 141–150.
[30] A. Andrei, P. Eles, O. Jovanovic, M. Schmitz, J. Ogniewski, Z. Peng, Quasi-static voltage scaling for energy minimization with time constraints, IEEE Transaction on VLSI systems 19 (2011) 10–23.
[31] S. Perathoner, K. Lampka, N. Stoimenov, L. Thiele, J.J. Chen, Combining optimistic and pessimistic dvs scheduling: an adaptive scheme and analysis, ICCAD (2010) 131–139.
[32] Y. Liu, R.P. Dick, L. Shang, H. Yang, Accurate temperature-dependent integrated circuit leakage power estimation is easy, DATE (2007) 1526–1531.
[33] H. Huang, G. Quan, J. Fan, Leakage temperature dependency modeling in system level analysis, ISQED (2010) 447–452.
[34] G. Quan, V. Chaturvedi, Feasibility analysis for temperature-constraint hard real-time periodic tasks, IEEE Transaction on Industrial Informatics 6 (2010) 329–339.
[35] W. Liao, L. He, K. Lepak, Temperature and supply voltage aware performance and power modeling at microarchitecture level, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 24 (2005) 1042–1053.
[36] K. Skadron, T. Abdelzaher, M.R. Stan, Control-theoretic techniques and thermal-RC modeling for accurate and localized dynamic thermal management, HPCA (2002) 17–28.
[37] G. Quan, Y. Zhang, Leakage aware feasibility analysis for temperature-constrained hard real-time periodic tasks, ECRTS (2009) 207–216.

Mr. Vivek Chaturvedi is currently a Ph.D. candidate at the Electrical and Computer Engineering Department, Florida International University, Miami. His research interests include real-time systems, operating systems, scheduling techniques, and computer architecture. Mr. Chaturvedi has also worked as an engineering intern at Sun Microsystems, Burlington, MA. He is a student member of IEEE since 2009.

Mr. Huang Huang is a Ph.D. candidate in Department of Electrical and Computer Engineering, Florida International University. His research interests include real-time system scheduling and power/thermal aware design for VLSI. He received his B.S. in Electrical Engineering from Northwest University, China in 2007. He is a student member of IEEE since 2008.

Dr. Shangping Ren is an associate professor in Department of Computer Science at Illinois Institute of Technology and a senior member of IEEE. Her research interests are in the areas of real-time computing, distributed and cyber-physical systems, defect-tolerant virtualizations for multi-core architectures.

Dr. Gang Quan 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.