Task-Level Dynamic Voltage Scaling for Embedded System Design: Recent Theoretical Results

Taewhan Kim
School of Electrical Engineering and Computer Science
Seoul National University, Seoul, Korea
tkim@ssl.snu.ac.kr

Received 7 June 2010; Accepted 23 July 2010

It is generally accepted that dynamic voltage scaling (DVS) is one of the most effective techniques of energy minimization for real-time applications in embedded system design. The effectiveness comes from the fact that the amount of energy consumption is quadractically proportional to the voltage applied to the processor. The penalty is the execution delay, which is linearly and inversely proportional to the voltage. According to the granularity of tasks to which voltage scaling is applied, the DVS problem is divided into two subproblems: inter-task DVS problem, in which the determination of the voltage is carried out on a task-by-task basis and the voltage assigned to the task is unchanged during the whole execution of the task, and intra-task DVS problem, in which the operating voltage of a task is dynamically adjusted according to the execution behavior to reflect the changes of the required number of cycles to finish the task before the deadline. Frequent voltage transitions may cause an adverse effect on energy minimization due to the increase of the overhead of transition time and energy. In addition, DVS needs to be carefully applied so that the dynamically varying chip temperature should not exceed a certain threshold because a drastic increase of chip temperature is highly likely to cause system function failure. This paper reviews representative works on the theoretical solutions to DVS problems regarding inter-task DVS, intra-task DVS, voltage transition, and thermal-aware DVS.

Categories and Subject Descriptors: Design **[Special-purpose and Application-based Systems]**:

General Terms: Design Methodology

Additional Key Words and Phrases: Dynamic Voltage Scaling, Power Management, Task Scheduling

1. INTRODUCTION

Over the past decades there have been enormous efforts to minimize the energy consumption of CMOS circuit systems. Dynamic voltage scaling (DVS), involving

dynamic adjustments of the supply voltage and the corresponding operating clock frequency, has emerged as one of the most effective energy minimization techniques. A one to one correspondence between the supply voltage and the clock frequency in CMOS circuits imposes an inherent constraint to DVS techniques to ensure that voltage adjustments do not violate the target system's timing constraint.

Many previous works have focused on hard real-time systems with multiple tasks. Their primary concern is to assign a proper operating voltage to each task while satisfying each task's timing constraint. In these techniques, determination of the voltage is carried out on a task-by-task basis and the voltage assigned to the task is unchanged during the whole execution of the task, which is referred to as *inter-task voltage scheduling*. Yao et al. [Yao et al. 1995] proposed an optimal inter-task voltage scheduling algorithm for independent tasks in which a task is characterized by its arrival time, deadline, and required CPU cycles. The proposed scheduling technique computes the speed of execution at any given time (and thus automatically determines each task's starting and ending times) so that the total energy consumption is minimized. Although they formulated the problem without the constraint that the task should be assigned to a single operating voltage, by the convexity of the power function each task is given only one 'middle' voltage that is proved to be optimal. This is because of the underlying assumption that the speed of any specific time is constant, which is not true since the required number of processor cycles, on which the calculation of the speed may vary depending on the behavior of the task. That means, the proposed inter-task scheduling technique is optimal only if each task execution follows the worst-case execution path. Leaving the same assumption untouched, many works in the literature have tried to formulate new inter-task scheduling problems considering other issues. Some instances of such issues include tasks with dependency relations [Schmitz et al. 2002; Zhang et al. 2002; Varatkar and Marculescu 2003; Gorji-Ara et al. 2004; Andrei et al. 2004], discretely variable voltage processors [Kwon and Kim 2005; Gorji-Ara et al. 2004], multi-processor environments [Schmitz et al. 2002; Zhang et al. 2002; Varatkar and Marculescu 2003; Gorji-Ara et al. 2004; Andrei et al. 2004], voltage transition overheads [Andrei et al. 2004]. It has been reported [Shin and Kim 2006] that the *voltage transition overheads* are not trivial in terms of energy and delay, and should be taken into account in DVS as well.

On the other hand, a number of studies of other direction (e.g., [Shin et al. 2001; Seo et al. 2004; Seo et al. 2006]) have added a new dimension to the voltage scheduling problem, by considering energy saving opportunities within the task boundaries. In their approach, the operating voltage of the task is dynamically adjusted according to the execution behavior to accurately reflect the changes of the required number of cycles to finish the task before the deadline, which is referred to as *intra-task voltage scheduling*. Shin et al. [Shin et al. 2001] proposed a remaining worst-case path-based algorithm which achieves the best granularity by executing the basic blocks with possibly different operating voltages. To obtain tight operating points that lead to a minimum energy consumption, the algorithm updates the remaining path length as soon as the execution deviates from the previous remaining worst-case path. More recently, a profile-based optimal intra-task voltage scheduling technique was presented in [Seo et al. 2004; Seo et al. 2006]. It shows the best energy

savings by incorporating the task's execution profile into the calculation of the operating voltages. This algorithm is proved to be optimal in that when the task is executed repeatedly or periodically, it achieves a minimum average energy consumption. A recent work [Seo et al. 2005] attempts to solve the inter-task and intra-task DVS problems simultaneously so as not to miss the energy saving opportunity at each granularity of inter-task and intra-task DVS.

Recently, as the power density on a processor increases very rapidly, managing or controlling thermal profiles become another hot issue in DVS. Thus, in addition to minimizing energy consumption by DVS, the peak temperature caused by the task execution should be controlled via the execution scheduling of tasks and DVS to the task. The problems addressed are divided into a number of cases: DVS for minimizing total execution time under peak temperature constraint [Zhang and Chatha 2007]; task scheduling and DVS for minimizing peak temperature under deadline constraint [Jayaseelan and Mitra 2008]; distributing idle times between the executions of tasks with DVS for minimizing dynamic and leakage energy including the temperature induced leakage under deadline constraint [Bao et al. 2010]; thermal-constrained energy optimization using DVS and energy-constrained thermal optimization using DVS under deadline constraint in multiprocessor systems [Liu et al. 2007].

In this paper, we survey and describe, in a theoretical aspect, state-of-art techniques of dynamic voltage scaling (DVS) problems, which include: (1) inter-task DVS problem, (2) intra-task DVS problem, (3) integrated inter-task and intra-task DVS problem, (4) transition-aware DVS problem, and (5) thermal-aware DVS problem. (The preliminary version of this work can be found in [Kim 2006].) The scope of this survey is confined to single processor DVS.

2. INTRA-TASK DVS TECHNIQUES

The amount of energy dissipation for the execution of a task is

$$E \propto V_{DD}^2 \times N_{tot} \tag{1}$$

where N_{tot} is the total number of instruction cycles executed for a task. Thus, the *intra-task voltage scheduling* problem is to assign a proper voltage to each basic block of the task so that the energy consumption in Eq. (1) is minimized.

The relationship between clock frequency and voltage in CMOS circuits is

$$f_{CLK} \propto (V_{DD} - V_T)^{\alpha} / V_{DD} \tag{2}$$

where V_T is the threshold voltage and α is the velocity saturation index. If the value of V_T is small enough, the expression is reduced to $f_{CLK} \propto V_{DD}^{\alpha-1}$.

Since the clock frequency determines the voltage, the scheduling problem can be stated as:

(Intra-Task DVS Problem) *The intra-task voltage scheduling problem for a task's CFG (control flow graph) is to determine the clock frequency for each node (i.e., basic block) of the CFG so that the total energy by the task is minimized while satisfying the timing constraint of the task.*

The key problem to solve is to determine what clock frequency should be set to the

entry point of each basic block so that the overall energy consumption of the task is minimized. Existing intra-task DVS techniques can be classified according to the way of determining the lowest clock frequency to be set at the entry point of each basic block.

1. (*RWCEP based DVS*): This technique [Shin et al. 2001] uses the lowest clock frequency by which the remaining WCEP (worst case execution path) can be completed within the deadline of the task.
2. (*RACEP based DVS*): This technique [Shin and Kim 2001] uses the lowest clock frequency by which the remaining ACEP (average case execution path) can be completed within the deadline of the task.
3. (*ROCEP based DVS*): This technique [Seo et al. 2006] uses the lowest clock frequency by which the remaining OCEP (energy-optimal case execution path) can be completed within the deadline of the task.

For example, consider a simple hard real-time task with deadline of 100 ms and three basic blocks. Its control flow graph is shown in Figure 1(a) where the number inside each node indicates the number of execution cycles of the block and the number assigned to each arc indicates the probability that the control flow follows the edge.

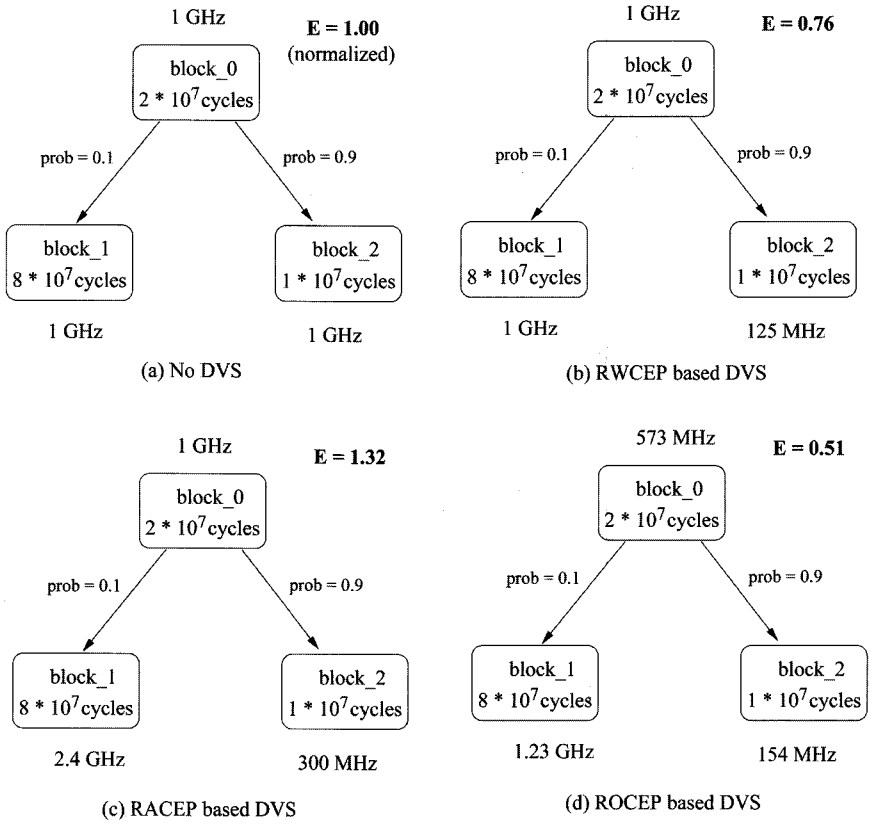


Figure 1. Example showing the calculations of clock frequency in basic block_0.

If DVS technique is not used, the speed for the task should be set tightly to (length of the critical path)/(remaining time to deadline)=[(2+8)10^7 cycles]/(100·10^{-3} s)=1 GHz. (See Figure 1(a).) If DVS technique follows the worst case execution path, the speed of block_2 will be set to (length of the (RWCEP from block_2)/(remaining time to deadline)=[1·10^7 cycles]/(80·10^{-3} s)=125 MHz. (See Figure 1(b).) On the other hand, if DVS technique follows the average case execution path, the speed is set to (length of the (RACEP from block_0)/(remaining time to deadline)=[(2+1)·10^7 cycles]/(100·10^{-3} s) =300 MHz. (See Figure 1(c).) Finally, if DVS follows energy-optimal execution path, the speed to be set in block_0 is calculated based on the probabilities of its succeeding basic blocks. Here, the speed is 573 MHz. See Figure 1(d).) In summary, we can see that the ROCEP based DVS technique outperforms the others because the clock speed used at each basic block always leads to the total energy consumption which is optimal on the average.

The work in [Seo et al. 2006] contains the detailed procedure of energy-optimal speed calculation of basic blocks. Here, we give a summary of the speed calculation. A task τ is represented with its CFG $G_\tau=(V, A)$, where V is the set of basic blocks in the task and A is the set of directed edges which impose precedence relations between basic blocks. (For example, see Figure 2(a).) The set of immediate successor basic blocks of any $b_i \in V$ is denoted by $succ(b_i)$. Each basic block b_i is annotated with its non-zero number of execution cycles n_i and each arc (b_i, b_j) is given a probability p_j that the execution follows the arc.

Given a task's CFG and its execution profile that offers the probabilities, we execute each basic block b_i at a speed of δ_i/(*remaining time*) and adjust the supply voltage accordingly, where δ_i is defined as:

$$\delta_i=\begin{cases} n_i, & \text{if } succ(b_i)=0 \\ n_i+\sqrt[3]{\sum_{\forall b_j \in succ(b_i)} p_j \delta_j^3}, & \text{otherwise} \end{cases} \quad (3)$$

The corresponding average energy consumption is proved to be optimal and expressed as [Seo et al. 2006]:

$$E_{intra}=\left(C\cdot\frac{\delta_0}{(relative)\ deadline}\right)^2\cdot\delta_0 \quad (4)$$

where C is a system-dependent constant, and δ_0 is the δ value of the top basic block b_0 and called *energy-optimal path length* of the task. One interesting interpretation of Eq. (4) is that it can be considered as the energy consumed in the execution of δ_0 cycles at a speed of δ_0/*deadline*. Note that δ_0/*deadline* is the initial speed of the task.

For example, consider a real-time task τ_{simple} shown in Figure 2(a). Figure 2(b) shows the procedure of calculating each δ_i value according to Eq. (3). Once we have performed the calculation, the operating speed of basic block b_i is simply obtained by dividing δ_i by the remaining time to deadline. For example, suppose that *deadline*=10 (unit time) and the execution follows the path (b_0, b_2, b_3, b_5, b_6, b_8). Then, the corresponding speed/ending time changes as follows:

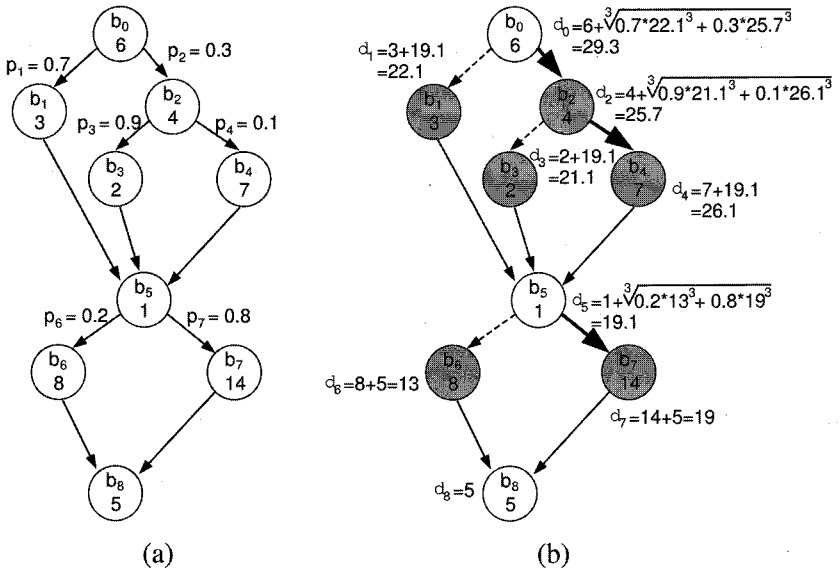


Figure 2. (a) CFG of a task τ_{simple}; (b) calculation of δ values.

Speed: $2.93 > 3.24 > 3.14 > 3.14 > 2.26 > 2.26$
Ending Time: $2.05 > 3.29 > 3.92 > 4.24 > 7.78 > 10$

In Figure 2(b), the thick, dotted, and regular arrows respectively indicate the increase, decrease and no change of the processor speed, and the basic blocks with changed operating speed are marked with gray-color.

3. INTER-TASK DVS

The most DVS works belong to the inter-task DVS. The inter-task DVS problem can be stated as:

(Inter-Task DVS Problem) *Given an instance of tasks with deadlines and voltages, find a feasible task-level schedule and task-level voltage allocation to tasks that minimizes the total energy consumption while satisfying the deadline constraints of tasks.*

Table I summarizes the current status of the energy-optimal works for the constrained cases of inter-task DVS problems. Note that in addition to the voltage, the amount of energy consumption is affected by the switched capacitance of the task. The value of capacitance is determined according to the execution characteristics of the task: If the task requires hardware components with high switched capacitance, such as multipliers, for execution, the capacitance value will be large, and vice versa. Consequently, to reduce the total energy consumed by tasks, it is desirable to execute the tasks with low switched capacitance using high supply voltages while the tasks with high switched capacitance using low supply voltages. The unsolved case is that of continuous voltages with nonuniform capacitance of tasks. The work by Kwon et al.

Table I. Summary of intra-task DVS techniques.

Voltage	Tasks		optimal?/ref.
cont. voltage	single task		Yes/trivial
	multiple tasks	uniform cap.	Yes/Yao et al. [Yao et al. 1995]
		nonuniform cap.	unknown
disc. voltage	single task		Yes/Ishihara et al. [Ishihara and Yasuura 1998]
	multiple tasks	uniform cap.	Yes/Kwon et al. [Kwon and Kim 2005]
		nonuniform cap.	Yes/Kwon et al. [Kwon and Kim 2005]

[Kwon and Kim 2005] for uniform capacitance case actually makes use of the optimal-work by Ishihara et al. [Ishihara and Yasuura 1998] and Yao et al. [Yao et al. 1995]. The work by Kwon et al. [Kwon and Kim 2005] for nonuniform capacitance case uses a linear programming (LP) formulation. Here, we show the algorithm of Kwon et al. [Kwon and Kim 2005] for uniform capacitance case.

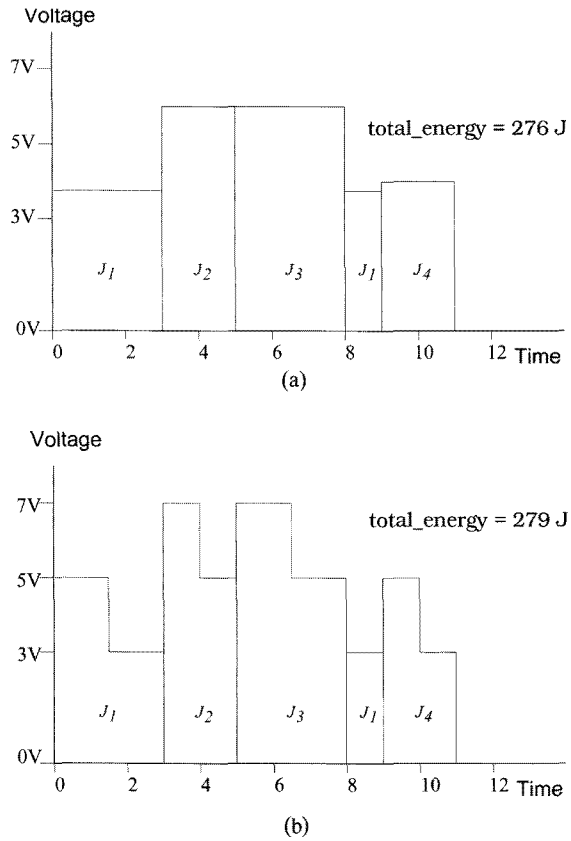


Figure 3. An example illustrating the transformation of continuously variable voltage allocation into discontinuously variable voltage allocation. (a) A continuously variable voltage allocation for tasks; (b) a discontinuously variable voltage allocation derived from (a).

The procedure starts from the results of the possibly invalid voltage allocation with the feasible task schedule obtained from Yao et al.'s algorithm [Yao et al. 1995], and transforms it into that of valid voltage allocation with a feasible schedule. More precisely, the schedule of tasks during transformation, but change the voltages so that they are all valid. Then, the question is what and how the valid supply voltages are selected and used. A valid voltage for each (scheduled) task is determined by performing the following three steps: (Step 1: *Merge time intervals*) All the scheduled time intervals that were allotted to execute the task are merged into one; (Step 2: *Voltage reallocation*) The invalid supply voltage is replaced with a set of valid voltages. (Step 3: *Split time interval*) The merged time interval is then split into the original time intervals.

For example, suppose that we have three voltages 7.0V, 5.0V, and 3.0V available for use and their corresponding clock speeds are 70 MHZ, 50 MHZ, and 30 MHZ, respectively. Then, for each scheduled task with the ideal voltage in Figure 3(a), the three steps are applied. Figure 4 shows the results of three steps for task J_1. Initially, J_1 is scheduled to be executed in two time intervals [0,3] and [8,9] with the voltage being 3.75V, as shown in Figure 4(a). (According to the results in [Yao et al. 1995] each task always uses the same voltage.) Consequently, in Step 1 the time intervals are merged into [0,4] as shown in Figure 4(b). The supply voltage in Step 2 is then

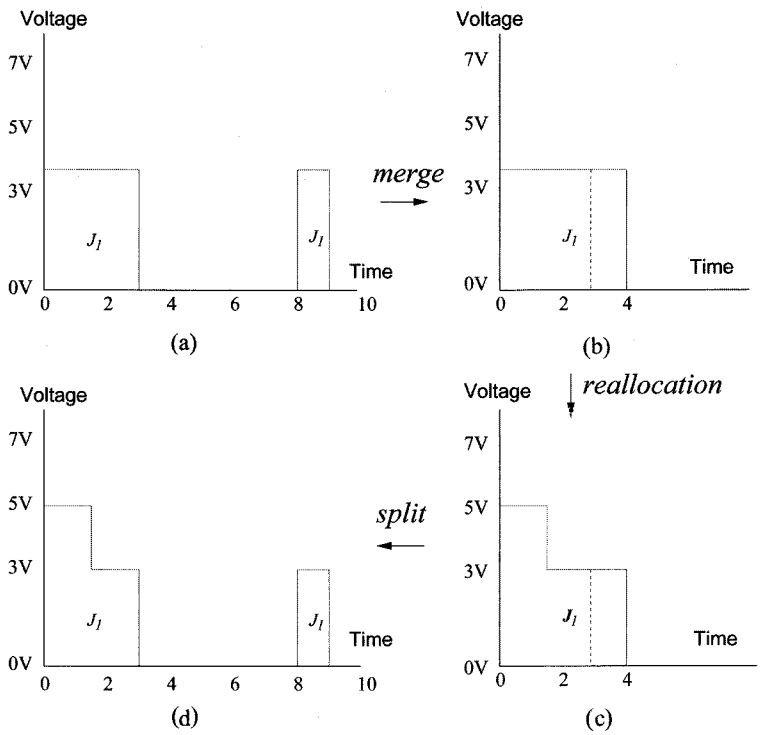


Figure 4. The three steps of voltage allocation procedure [Kwon and Kim 2005] for task J_1 in Figure 3. (a) An initial schedule in Figure 3(a); (b) the result after merging time intervals; (c) the result after voltage reallocation; (d) the result after splitting the time interval.

updated. To do this, we make use of Ishihara and Yasuura's results [Ishihara and Yasuura 1998]: For a given ideal (optimal) voltage for a task, the valid (optimal) voltage allocation is to use the two immediately neighboring valid voltages to the ideal voltage. (For details on how to find the time point at which the clock speed changes, see [Ishihara and Yasuura 1998].) Figure 4(c) shows the result of voltage reallocation where two voltages 3.0V and 5.0V are used because the ideal voltage (=3.75V) is in between 3.0V and 5.0V, and no other valid voltages are in the interval. Finally, in Step 3 we restore the time intervals while preserving the voltage reallocation obtained in Step 2, as shown in Figure 4(d). By repeating these three steps for J_2, J_3, and J_4 in Figure 3(a), a voltage allocation for all tasks with a feasible schedule is obtained, as shown in Figure 3(b). Note that because we used only a number of discrete voltages, the energy consumption, which is 279J, increases from that in Figure 3(a), which is 276J. However, according to [Kwon and Kim 2005], the amount of the increase is minimal.

4. INTEGRATION OF INTRA-TASK DVS

The combined problem of the inter- and intra-task DVS problems can be described as:

(Combined DVS Problem) *Given an instance of tasks with deadlines and voltages, find a feasible task-level schedule and task-level voltage allocation to tasks that minimizes the total energy consumption while satisfying the deadline constraints of tasks.*

There are two optimal results in the literature related to the combined DVS problem: (i) an optimal inter-task DVS scheme [Yao et al. 1995] that determines the operating voltage of each task assuming the worst-case execution path and (ii) an optimal intra-task DVS scheme [Seo et al. 2006] that determines the operating voltage of each basic block in a single task. From the analysis of the procedures of (i) and (ii), the work in [Seo et al. 2005] found that an energy-optimal integration of (i) and (ii) is possible with a slight modification of the procedures. The proposed integrated DVS approach is a two-step method:

(1) Statically determine energy-optimal starting and ending times (s_i and e_i) for each task τ_i.
(2) Execute τ_i within $[s_i, e_i]$ while varying the processor speed according to the voltage scales obtained by an existing optimal intra-task DVS scheme.

The key concern is to develop a new inter-task scheduling algorithm that finds starting and ending times (s_i and e_i) for each task, which leads to a minimum value of total energy consumption when an optimal intra-task scheme is applied to the tasks. For the further details on the optimal algorithm, it can be referred to the work in [Seo et al. 2005].

5. TRANSITION-AWARE DVS

During the execution of tasks, three types of transition overhead are encountered for each voltage transition: *transition cycle*, *transition interval* and *transition energy*.

A *transition cycle* (denoted as Δn) is the number of instruction cycles of the transition code itself. We assume that the energy for executing the transition instruction varies depending on the voltages in transition.

A *transition interval* (denoted as Δt) is the time taken during the voltage transition, which is not constant. Note that in the current commercial designs, the Phase-Locked Loop that is used to set the clock frequency requires a fixed amount of time to lock on a new frequency. This locking time is known to be independent of the source and target frequencies, and is typically much smaller than the time it takes for the voltage to change [Analog-Device 2010]. Therefore, it is desirable to assume that the transition interval Δt is not a fixed value. A reasonable assumption for the variable voltage processor is that the transition interval is proportional to the difference between the starting and ending transition voltages [Texas-Instruments 2010].

A *transition energy* (denoted as ΔE) is the amount of energy consumed during the transition interval by the systems. The value of ΔE may vary depending on the starting and ending voltage levels. The DVS problem with the additional consideration of transition energy is even more difficult to solve because the generalized model of the transition energy for various systems/processors is hard to obtain and even a simplified version of the problem with the assumption that the transition energy is a constant looks a non-trivial task to solve. Let $\Delta E_{(v_i \to v_j)}$ denote the energy dissipated by the transition of voltage from v_i to v_j. We assume that the processor has a set $\mathcal{V}$, called *voltage set*, of voltages available to use, i.e., $\mathcal{V}=\{v_1, v_2, \cdots, v_M\}$ $(v_1 < v_2 < \cdots < v_M)$. In addition, we assume that the values of $\Delta E_{(v_i \to v_j)}$, v_i, $v_j \in \mathcal{V}$ have already been given, and the values are stored in a table, called *transition-energy table.* Furthermore, we reasonably assume that $\Delta E_{(v_i \to v_j)} < \Delta E_{(v_i \to v_{j'})}$ if $|v_i - v_j| < |v_i - v_{j'}|$.

(Transition energy aware DVS problem) *Given a fixed schedule of tasks, and voltage set $\mathcal{V}$ and the associated transition-energy table, assign the voltages to the tasks so that the total energy consumption for the execution of tasks, together with the energy consumed by the voltage transitions, is minimized.*

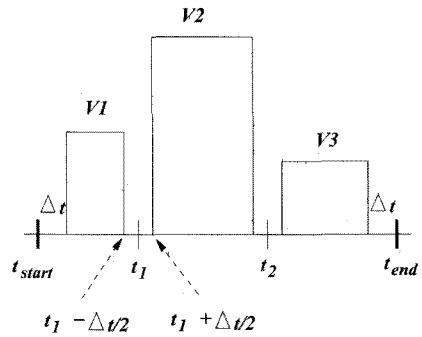
Unfortunately, most of the DVS methods never take into account the minimization of the amount of energy consumed during the voltage transition. To our knowledge, Mochocki, Hu, and Quan [Mochocki et al. 2002] and Shin and Kim [Shin and Kim 2006] addressed the DVS problem with the consideration of transition overheads and discrete voltages. The work in [Mochocki et al. 2002] put the primary emphasis on the consideration of transition intervals by modifying the optimal scheduling algorithm in [Yao et al. 1995] together with a simple treatment on both the transition energy and discrete voltages. On the other hand, the work in [Shin and Kim 2006] attempted the limitation of the work in [Mochocki et al. 2002] in that it tries to solve the problem optimally for a constrained case. We review the work by [Shin and Kim 2006] here. The key contribution of the work is the network formulation of the problem. Here, we show the procedure of single task case only. It was naturally extended to the cases of multiple tasks.

Suppose an instance of TE-VA has a task τ_1 with the schedule interval $[t_{start}, t_{end}]$ and R cycles to be executed, If the overheads of voltage transition are not taken into account, an optimal result can be obtained by using the voltage assignment technique

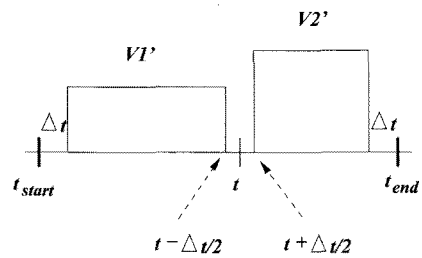
in [Ishihara and Yasuura 1998], i.e., the optimal voltage assignment is to use the two voltages in $\mathcal{V}$ that are the immediate neighbors to the (ideal) voltage corresponding to the lowest possible clock speed which results in an execution of task τ_1, exactly starting at time t_{start} and ending at time t_{end}; (The ideal voltage can be obtained according to the voltage and delay (clock speed) relation [Ishihara and Yasuura 1998].

Lemma 1 *For an instance of the transition-aware DVS problem with a single task, an energy-optimal voltage assignment uses at most two voltages for the execution of task.* (The proof described below becomes the foundation of the proposed network formulation.)

proof. Suppose there is an optimal voltage assignment $\mathcal{VA}$ which uses more than two voltages for the execution of the task during the scheduled interval of $[t_{start}, t_{end}]$. Let $v_1, \cdots, v_K$ $(K > 2)$ be the sequence of voltages applied to the task, starting from t_{start} to t_{end}, by the optimal voltage assignment, and let $[t_{start}+\Delta t, t_2-\Delta t/2]$, $[t_2+\Delta t/2, t_3-\Delta t/2]$, $\cdots$, $[\Delta t/2+t_r, t_{end}-\Delta t]$ be the corresponding execution intervals. (See Figure 5(a) for an example.) Note that the length of the actual execution interval, not including the transition interval, is $T_L=t_{end}-t_{start}-(r+1)\Delta t$. Since the voltages before and after the execution of the task are 0V, transition interval Δt is required from 0V to v_1 at the beginning (i.e., $[t_{start}+\Delta t, t_2-\Delta t/2]$) and from v_r to 0V at the end (i.e., $[\Delta t/2+t_r, t_{end}-\Delta t]$) Also, a transition interval is required between the two consecutive execution intervals



(a) An example of more than two voltage assignment for a task execution



(b) An example of two voltage assignment for the same task in (a)
(v1' and v2' are adjacent voltages)

Figure 5. Transition intervals and actual execution intervals for an example of voltage assignment using more than two voltages and voltage assignment using two voltages.

(e.g., Δt around time t_i in $[t_{i-1}+\Delta t/2,\ t_i-\Delta t/2]$ and $[t_i+\Delta t/2,\ t_{i+1}-\Delta t/2]$).

Now, consider another voltage assignment $\mathcal{VA}'$ which uses two voltages for the actual execution length of $\mathcal{TL}'=t_{end}-t_{start}-3\Delta t$. (See Figure 5(b).) Let v'_1 and v'_2 be the two voltages which lead to a minimum total energy consumption except the transition energy. (Note that the two voltages can be obtained by applying the technique in [Ishihara and Yasuura 1998] to the task of R cycles with an execution interval of length T_L.) Then, we want to compare the amount of energy consumptions excluding the transition energy used in $\mathcal{VA}$ and $\mathcal{VA}'$, and compare the amounts of transition energy used in $\mathcal{VA}$ and $\mathcal{VA}'$.

Obviously, we can see that $\max\{v_1, \cdots, v_r\} \geq \max\{v'_1, v'_2\}$ since the execution interval for $\mathcal{VA}$ is shorter than that for $\mathcal{VA}'$ due to more transition intervals in $\mathcal{VA}$ than that of $\mathcal{VA}'$. This means that the transition energy from 0V at the start to eventually $\max\{v_1, \cdots, v_r\}$ is greater than that from 0V to $\max\{v'_1, v'_2\}$ by the assumption $\Delta E_{(v_i \to v_j)} < \Delta E_{(v_i \to v_{j'})}$ if $|v_i - v_j| < |v_i - v_{j'}|$. Thus, the total transition energy for $\mathcal{VA}$ is greater than that for $\mathcal{VA}'$. On the other hand, since from the fact that v'_1 and v'_2 are the voltages of optimal voltage assignment with time length of T'_L, and $T'_L < T_L$, the total energy consumption without transition energy for $\mathcal{VA}'$ is less than that for $\mathcal{VA}$. Thus, the assumption that $\mathcal{VA}$ is optimal is false.

From Lemma 1, the solution space for a single task can be confined to the solutions which only use at most two voltages in $\mathcal{V}$. The remaining issue is to give, for an given execution interval for the task, a technique of finding the two (optimal) voltages and their execution intervals. According to [Shin and Kim 2006]: A network $N(V, A)$ is constructed for a task with R instruction cycles, execution interval $[t_{start}, t_{end}]$, voltage set $\mathcal{V}$ and transition energy table where V is the set of nodes and A is the set of arcs between two nodes. Note that Lemma 1 tells there is always an optimal voltage assignment which uses two voltages only. (See the upper figure in Figure 6.) If we know the two voltages used, then the two actual execution intervals can be computed accordingly using the speed and voltage relation [Ishihara and Yasuura 1998; Kwon and Kim 2005]. For each of the two execution intervals, $|\mathcal{V}|$ nodes are arranged vertically, each node representing a unique voltage in $\mathcal{V}$. Then two additional nodes are included in V. One is *start-node*, placing at the front of $N(V, A)$ and the other is *end-node* at the end. (See the lower figure in Figure 6.) We insert arcs from the nodes in the first column to those in the second column, from the start-node to the nodes in the first column, and from the nodes in the second column to the end-node. (See the lower figure in Figure 6.) We then assign weight to each arc. The weight of an arc from the start-node to a node labelled as v_i in the first column indicates $\Delta E_{(0V \to v_i)}$, and the weight of an arc from a node labelled as v_i to the end-node indicates $\Delta E_{(v_i \to 0V)}$. The weight of an arc from a node labelled as v_i to a node labelled as v_j represents the total sum of the (minimal) energy consumed by the execution of the task (i.e., $E1(v_1)+E2(v_2)$ in Figure 6) and the transition energy $\Delta E_{v_i \to v_j 1}$.

Then, from the network $N(V, A)$, a shortest path can be found from the start-node to the end-node. The cost of the shortest path is exactly equivalent to the total amount of energy consumption including the transition energy for the execution of the task. the procedure [Shin and Kim 2006] performs the two steps: (Step 1) Construct

network $N(V, A)$ for the task; (Step 2) Find a shortest path (SP) on $N(V, A)$.

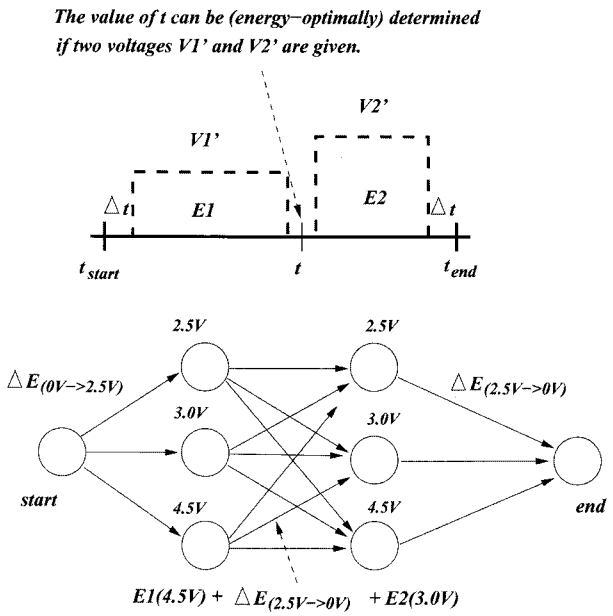


Figure 6. The construction of network for modeling an instance of the TE-VA problem with a single task.

6. TEMPERATURE-AWARE DVS

As the power density on a chip increases rapidly, controlling or minimizing the increase of temperature is one of the most important design factors to be considered in the course of task execution, because it increases circuit delay, in particular spatially unbalanced delays on a chip, causing system function failure. Furthermore, the leakage power increases exponential as the temperature increases.

(Processor thermal model) Most of the temperature-aware DVS works (e.g., [Liu et al. 2007; Jayaseelan and Mitra 2008; Zhang and Chatha 2007; Bao et al. 2010] use, as the thermal model, the lumped RC model similar to that in [Skadron et al. 2002] to capture the heat transfer phenomena as shown in Figure 7. (For the additional models, see the references [Rao et al. 2006; Martin et al. 2002; Pillai and Shin 2001; Xie et al. 2005].) In the figure, T, C, and R, represent the processor's die temperature, the thermal capacitance of the die, and the thermal resistance, respectively. T_{amb} indicates the ambient temperature and $P(t)$ the power consumption of the processor at current time t. For the given values of T_{amb}, C, and R, the relation between die temperature T and power consumption $P(t)$ can be expressed as:

$$R \cdot C \cdot \frac{dT}{dt} + T - R \cdot P = T_{amb}. \tag{5}$$

In the following, we introduce two fundamental problems on the temperature-aware

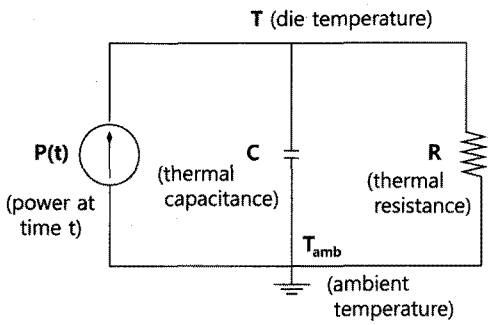


Figure 7. Processor heat transfer model.

voltage scaling and their solutions.

(Temperature-constrained DVS problem for performance optimization) *Given an execution schedule of tasks and peak temperature constraint T_{max}, determine the voltages to be applied to tasks, if needed, inserting sleep time intervals between task executions so that the completion time of all tasks is minimized while satisfying the peak temperature constraint.*

One notable work on the temperature-constrained DVS problem is done by [Zhang and Chatha 2007] in which it showed the NP-hardness of the problem when a set of discrete voltages is assumed to be used. The optimal formulation is followed by introducing two types of variables defined below.

A. Suppose there are n tasks $\tau_1, \tau_2, \cdots, \tau_n$ that should be executed in that order. Each task can be assigned to a voltage among m discrete voltages $v_1, \cdots, v_m$. Let $x_{i,j}$ ($1 \leq i \leq n$, $1 \leq j \leq m$) denote a variable having a value of 0 or 1 such that $x_{i,j}=1$ if τ_i is executed with v_j, and $x_{i,j}=0$ if τ_i is executed with a voltage other than v_j. Thus, when $exe\_t(\tau_i, v_j)$ denotes the time spent on executing task τ_i on v_j, the total execution time spent on executing all tasks is $\Sigma_{i=1,\cdots,n}\ \Sigma_{j=1,\cdots,m}\ x_{i,j} \cdot exe\_t(\tau_i, v_j)$.
B. In addition, there are total of $n+1$ idle intervals, one for each of two consecutive task executions, one before τ_1, and one after τ_n. The time length of each idle interval will be upper-bounded by t_{max_sleep}: the value of t_{max_sleep} is the time that is minimally required to cool down T_{max} to T_{amb}. We discretize $[0, t_{max\_sleep}]$, so that q sub-intervals of equal length are formed i.e., $[t_0, t_1, t_2, \cdots, t_i, \cdots, t_q]$ where $t_i = i \cdot (t_{max\_sleep}/q)$, $i=0, \cdots, q$. (q is a user defined value.) Now, let variable $y_{i,j}$ ($1 \leq i \leq n$, $1 \leq j \leq q$) be such that $y_{i,j}=1$ if the time spent on idle state right after τ_i is t_j. Then, the total idle time spent during the execution of tasks is $\Sigma_{i=1,\cdots,n+1}\ \Sigma_{j=0,\cdots,q}\ y_{i,j} \cdot t_j$.

Thus, the formulation is expressed as:

$$minimize\ D = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{i,j} \cdot exe\_t(\tau_i, v_j) + \sum_{i=1}^{n} \sum_{j=0}^{q} y_{i,j} \cdot t_j \tag{6}$$

such that

$$\sum_{j=1}^{m} x_{i,j}=1,\ \sum_{j=0}^{q} y_{i,j}=1,\ \forall i \in \{1,\cdots,n\}; \tag{7}$$

$$R \cdot C \cdot \frac{dT}{dt} + T - R \cdot P = T_{amb}; \tag{8}$$

$$T \le T_{\max}; \; x_{i,j}=\{0, 1\}; \; y_{i,j}=\{0, 1\}; \tag{9}$$

$$T(t) \le T_{\max}, \; 0 \le t \le D; \; T(t=0)=T(t=D) \tag{10}$$

in which the last constraint ensures that the temperature at the beginning of the execution of tasks must be the same at the time when the execution of all task is completed to support the periodic execution of task set.

The problem can also be optimally formulated using the dynamic programming (DP) approach that runs in pseudo-polynomial time [Zhang and Chatha 2007]. Let $T_1(\tau_i, D)$ denote the minimum temperature at D which is the time when τ_i just finishes the execution, and let $T_2(\tau_i, D)$ denote the minimum temperature at D which is the time just when τ_{i+1} starts execution. Then, the optimal D^* is determined by the smallest value of D such that $T_2(\tau_n, D) \le T_{\max}$. The recurrence relation can be given as:

$$T_1(\tau_i, D)=min_{j=1, \cdots, m}\{T=T_2(\tau_{i-1}, D-exe\_t(\tau_i, v_j))+\Delta T(exe\_t(\tau_i, v_j)) \mid T \le T_{\max}\};$$
$$T_2(\tau_i, D)=min_{j=0, \cdots, q}\{T=T_1(\tau_i, D-t_j)+\Delta T(t_j, v_{sleep}) \mid T \le T_{\max}\}$$

where v_{sleep} is the voltage at idle state. (The derivation of termination cases can be easily obtained.)

The time complexity of the DP algorithm is polynomial in terms of n, q, and D_{UB} where D_{UB} is a large number that surely exceeds D^*. The work in [Zhang and Chatha 2007] also proposes a fully polynomial time approximation scheme (FPTAS).

(Time constrained DVS problem for peak temperature minimization) *Given a set of tasks and time constraint D_{max}, determine the schedule of tasks and voltages to be applied to tasks so that the peak temperature is minimized while satisfying the time constraint.*

The work in [Jayaseelan and Mitra 2008] showed that the problem is NP-hard even when the voltage to each task is given and proposed a greedy heuristic which

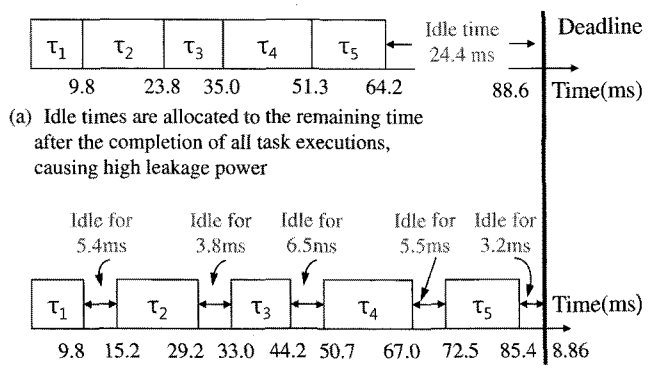
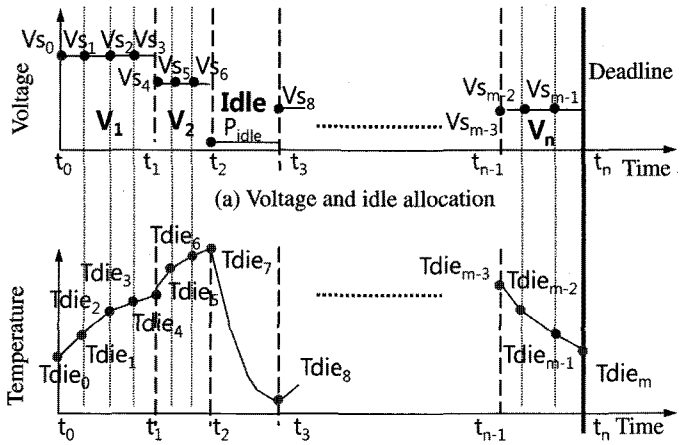


Figure 8. Two different idle time allocations. Distributing idle times, as shown in (b), between task executions can reduce the temperature induced leakage power.



(b) Temperature profile corresponding to (a)

Figure 9. The changes of temperature as the processor alternates task execution with DVS and idle state, like that in Figure 8(b). Task executions heat up the die while idle states cool down the die.

attempts to schedule tasks such that a cold task is put right after a hot task and a hot task right after a cold task. The heuristic is further exploited repeatedly to gradually find the voltages to tasks so that the peak temperature is minimized.

For a give task sequence, the voltage assignment under $D_{\max}$ constraint for minimizing peak temperature can be formulated in pseudo-polynomial time [Jayaseelan and Mitra 2008], which is similar to the DP algorithm shown earlier.

Unfortunately, the work does not address the allocation of idle states between task executions. The work in [Bao et al. 2010] proposes a solution to the idle allocation problem in which the objective is to minimize the total power including the temperature induced leakage power while meeting $D_{\max}$ constraint. For example, Figure 8 ([Bao et al. 2010]) shows two different idle allocations for a sequence of five tasks τ_1, $\cdots$, τ_5 with predefined DVS. Clearly, Figure 8(b) will lead to less power consumption than that in Figure 8(a). Figure 9 ([Bao et al. 2010]) shows how the temperature changes as the voltages and idle times are allocated in a sequence like that in Figure 8(b).

7. CONCLUSIONS

In this paper, we described the current status of the research works on dynamic voltage scaling (DVS) in view of the optimality of energy minimization. The DVS problems we covered in the paper were (1) inter-task DVS problem, (2) intra-task DVS problem, (3) integrated inter-task and intra-task DVS problem, (4) transition-aware DVS problem, and (5) thermal-aware DVS problem. It should be mentioned that, except the techniques described in the paper, there are many other effective DVS techniques which are targeted under other DVS constraints or environments such as fixed priority scheduling, jitter constraint, soft deadline, and multiprocessor systems.

ACKNOWLEDGEMENT

This work was supported by Basic Science Research Program through National Research Foundation (NRF) grant funded by the Korea Ministry of Education, Science and Technology (No. 2009-0091236).

REFERENCES

ANALOG-DEVICE. 2010. Analog dialogue. http://www.analog.com/library/analogDialogue/archives/30-3/single_chip.html.

ANDREI, A., SCHMITZ, M., ELES, P., PENG, Z., AND AL-HASHIMI, B. M. 2004. Overhead-conscious voltage selection for dynamic and leakage energy reduction of time-constrained systems. In *Proceedings of Design Automation and Test in Europe.* IEEE, 518–523.

BAO, M., ANDREI, A., ELES, P., AND PENG, Z. 2010. Temperature-aware idle time distribution for energy optimization with dynamic voltage scaling. In *Proceedings of Design Automation and Test in European.* IEEE.

GORJI-ARA, B., CHOU, P., BAGHERZADEH, N., RESHADI, M., AND JENSEN, D. 2004. Fast and efficient voltage scheduling by evolutionary slack distribution. In *Proceedings of Asia-South Pacific Design Automation Conference.* IEEE, 659–662.

ISHIHARA, T. AND YASUURA, H. 1998. Voltage scheduling problem for dynamically variable voltage processors. In *Proceedings of International Symposium on Low-Power Electronics and Design.* ACM, 197–202.

JAYASEELAN, R. AND MITRA, T. 2008. Temperature aware task sequencing and voltage scaling. In *Proceedings of International Conference on Computer-Aided Design.* IEEE, 618–623.

KIM, T. 2006. Application driven low-power techniques using dynamic voltage scaling. In *Proceedings of International Conference on Embedded and Real-Time Computing Systems and Application.* IEEE, 199–206.

KWON, W. C. AND KIM, T. 2005. Optimal voltage allocation techniques for dynamically variable voltage processors. *ACM Transactions on Embedded Computing Systems 4,* 1 (Feb.), 211–230.

LIU, Y., YANG, H., DICK, R., WANG, H., AND SHANG, L. 2007. Thermal vs energy optimization for dvfsenabled processors in embedded systems. In *Proceedings of International Symposium on Quality Electronic Designs.* IEEE, 204–209.

MARTIN, S., FLAUTNER, K., MUDGE, T., AND BLAAUW, D. 2002. Combined dynamic voltage scaling and adaptive body biasing for low power microprocessor under dynamic workloads. In *Proceedings of International Conference on Computer Aided Design.* IEEE, 721–725.

MOCHOCKI, B., HU, X. S., AND QUAN, G. 2002. A realistic variable voltage scheduling model for real-time applications. In *Proceedings of International Conference on Computer-Aided Design.* ACM/IEEE, 726–731.

PILLAI, P. AND SHIN, K. G. 2001. Real-time dynamic voltage scaling for low-power embedded operating systems. In *Proceedings of Symposium on Operating Systems Principles.* ACM, 89–102.

RAO, R., VRUDHULA, S., CHAKRABARTI, C., AND CHANG, N. 2006. An optimal analytical processor speed control with thermal constraint. In *Proceedings of International Conference on Computer Aided Design.* IEEE, 292–297.

SCHMITZ, M. T., AL-HASHIMI, B. M., AND ELES, P. 2002. Energy-efficient mapping and scheduling for dvs enabled distributed embedded systems. In *Proceedings of Design Automation and Test in Europe.* IEEE, 514–521.

SEO, J., KIM, T., AND CHUNG, K. 2004. Profile-based optimal intra-task voltage scheduling for hard real-time applications. In *Proceedings of Design Automation Conference.* ACM/IEEE, 87–92.

SEO, J., KIM, T., AND DUTT, N. D. 2005. Optimal integration of intra and inter task dynamic voltage scaling for hard real-time applications. In *Proceedings of International Conference on Computer-Aided Design.* ACM/IEEE, 450–455.

SEO, J., KIM, T., AND LEE, J. 2006. Optimal intra-task dynamic voltage scaling and its practical

extensions. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 25*, 1 (Jan.), 47–57.

SHIN, D. AND KIM, J. 2001. A profile-based energy-efficient intra-task voltage scheduling algorithm for hard real-time applications. In *Proceedings of International Symposium on Low-Power Electronics and Design*. ACM, 271–274.

SHIN, D., KIM, J., AND LEE, S. 2001. Intra-task voltage scheduling for low-energy hard real-time applications. *IEEE Design and Test of Computers 18*, 2 (Mar.), 20–30.

SHIN, J. AND KIM, T. 2006. Technique for transition energy-aware dynamic voltage assignment. *IEEE Transactions on Integrated Circuits and Systems II 53*, 9 (Sept.), 956–960.

SKADRON, K., ABDELZAHER, T., AND STAN, M. R. 2002. Control-theoretic techniques and thermal remodeling for accurate and localized dynamic thermal management. In *Proceedings of International Symposium on High Performance Computer Architecture*. IEEE, 17–28.

TEXAS-INSTRUMENTS. 2010. Power management. http://focus.ti.com/analog/docs.

VARATKAR, G. AND MARCULESCU, R. 2003. Communication-aware task scheduling and voltage selection for total systems energy minimization. In *Proceedings of International Conference on Computer-Aided Design*. ACM/IEEE, 510–515.

XIE, F., MARTONOSI, M., AND MALIK, S. 2005. Bounds on power saving using runtime dynamic voltage scaling: an exact algorithm and linear-time heuristic approximation. In *Proceedings of International Symposium on Low Power Electron Design*. IEEE, 287–292.

YAO, F., DEMERS, A., AND SHENKER, S. 1995. A scheduling model for reduced cpu energy. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*. ACM, 374–377.

ZHANG, S. AND CHATHA, K. S. 2007. Approximation algorithm for the temperature-aware scheduling problem. In *Proceedings of International Conference on Computer-Aided Design*. IEEE, 281–288.

ZHANG, Y., HU, X., AND CHEN, D. Z. 2002. Task scheduling and voltage selection for energy minimization. In *Proceedings of Design Automation Conference*. ACM/IEEE, 183–188.

Taewhan Kim received the B.S. degree in computer science and statistics and the M.S. degree in computer science from Seoul National University, Korea in 1985 and 1987, respectively, and received the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign in 1993. Currently, he is a professor at the school of electrical engineering and computer science, Seoul National University, Seoul, Korea. His research interests are in the area of embedded hardware/software, computer-aided design of integrated circuits, and combinatorial optimizations.