Overhead-Conscious Voltage Selection for Dynamic and Leakage Energy Reduction of Time-Constrained Systems

Abstract

Dynamic voltage scaling and adaptive body biasing have been shown to reduce dynamic and leakage power consumption effectively. In this paper, we optimally solve the combined supply voltage and body bias selection problem for multi-processor systems with imposed time constraints, explicitly taking into account the transition overheads implied by changing voltage levels. Both energy and time overheads are considered. We investigate the continuous voltage scaling as well as its discrete counterpart, and we prove NP-hardness in the discrete case. Furthermore, the continuous voltage scaling problem is formulated and solved using nonlinear programming with polynomial time complexity, while for the discrete problem we use mixed integer linear programming. Extensive experiments, conducted on several benchmarks and a real-life example, are used to validate the approaches.

1 Introduction

Two system-level approaches that allow an energy/performance trade-off during run-time of the application are dynamic voltage scaling (DVS) [9, 13, 16] and adaptive body biasing (ABB) [10, 13]. While DVS aims to reduce the dynamic power consumption by scaling down operational frequency and circuit supply voltage V_{dd}, ABB is effective in reducing the leakage power by scaling down frequency and increasing the threshold voltage V_{th} through body biasing. Up to date, most research efforts at the system-level were devoted to DVS, since the dynamic power component *had been* dominating. Nonetheless, the trend in deep-submicron CMOS technology to reduce the supply voltage levels and consequently the threshold voltages (in order to maintain peak performance) is resulting in the fact that a substantial portion of the overall power dissipation will be due to leakage currents [3, 10]. This makes the adaptive body biasing approach and its combination with dynamic voltage scaling indispensable for energy-efficient designs in the foreseeable future.

Voltage selection approaches can be broadly classified into on-line and off-line techniques. In the following, we restrict ourselves to the off-line techniques since the presented approaches fall into this category, where the scaled supply voltages are calculated before design time and then applied at run-time according to the pre-calculated voltage schedule.

There has been a considerable amount of work on dynamic voltage scaling. Yao et al. [16] proposed the first DVS approach for single processors systems which can dynamically change the supply voltage over a continuous range. Ishihara and Yasuura [9] modeled the discrete voltage selection problem using an integer linear programming (ILP) formulation. Kwon and Kim [11] proposed a linear programming (LP) solution for the discrete voltage selection problem with uniform and non-uniform switched capacitance. Although this gives the impression that this problem can be solved optimally in polynomial time, we will show in this paper that the discrete voltage selection problem is indeed NP-hard and, hence, no optimal solution can be found in polynomial time, for example using LP. Dynamic voltage scaling has also been successfully applied to heterogeneous distributed systems, in which numerous processing elements interact via a communication infrastructure, mostly using heuristics [7, 12, 19]. Zhang et al. [17] approached continuous supply voltage selection in distributed systems using an ILP formulation. They solved the discrete version through an approximation.

While the approaches mentioned above scale supply voltage V_{dd} only and neglect leakage power consumption, Kim and Roy [10] proposed an adaptive body biasing approach, in their work referred to as dynamic V_{th} scaling, for active leakage power reduction. They demonstrate that the efficiency of ABB will become, with advancing CMOS technology, comparable to DVS. Duarte et al. [6] analyze the effectiveness of supply and threshold voltage selection, and show that simultaneous adjusting both voltages provides the highest savings. Martin et al. [13] presented an approach for combined dynamic voltage scaling and adaptive body biasing. At this point we should emphasize that, as opposed to these three approaches, we investigate in this paper how to select voltages for a set of tasks, possibly with dependencies, which are executed on multiprocessor systems under real-time constraints. Furthermore, as opposed to our work, the techniques mentioned above *neglect* the energy and time overheads imposed by voltage transitions. Noticeable exceptions are [8, 14, 18], yet their algorithms ignore leakage power dissipation and body biasing, and further they do not guarantee optimality. In this work, we consider simultaneous supply voltage selection and body biasing, in order to minimize dynamic as well as leakage energy. In particular, we investigate four different notions of the combined dynamic voltage scaling and adaptive body biasing problem—considering continuous and discrete voltage selection with and without transition overheads. The presented work makes the following contributions:

(a) We consider both supply voltage and body bias voltage selection at the system-level, where several tasks with dependencies execute a time-constrained application on a multiprocessor system.

(b) Four different voltage selection schemes are formulated as nonlinear programming (NLP) and mixed integer linear programming (MILP) problems which can be solved optimally. The formulations are equally applicable to single and multi-processor systems.

(c) We prove that discrete voltage selection with and without the consideration of transition overheads in terms of energy and time is NP-hard, while the continuous voltage selection cases can be solved optimally in polynomial time.

To the authors' knowledge, this is the first work describing how adaptive body biasing and its combination with dynamic voltage scaling can be solved at the system-level for a time-constrained application. We also believe to report for the first time optimal voltage scheduling techniques, including the consideration of transition overheads in energy and delay.

The remainder of this paper is organized as follows: Preliminaries regarding system specification as well as power and delay models are given in Section 2. This is followed by a motivational example in Section 3. The four investigated voltage selection problems are formulated in Section 4. Continuous and discrete voltage selection problems are discussed in Sections 5 and 6, respectively. Experimental results are given in Section 7, and conclusions are drawn in Section 8.

2 Preliminaries

2.1 Architectural Model and System Specification

In this paper we consider embedded systems which are realized as heterogeneous distributed architectures. Such architectures consist of several different processing elements (PEs), such as programmable microprocessors, ASIPs, FPGAs, and ASICs, some of which feature DVS and ABB capability. These computational components communicate via an infrastructure of communication links (CLs), like buses and point-to-point connections. A directed graph $G_A(\mathcal{P},\mathcal{L})$ represents this architecture, in which nodes $\pi \in \mathcal{P}$ denote PEs and edges correspond to CLs. An example architecture is shown in Fig. 1(a). The functionality of dataflow intensive applications, such as voice processing and multimedia, can be captured by task graphs $G_S(\mathcal{T},\mathcal{C})$. Nodes $\tau \in \mathcal{T}$ in these directed acyclic graphs represent computational tasks, while edges $\gamma \in \mathcal{C}$ indicate data dependencies between these tasks (communications). Tasks require a finite number of clock cycles NC to be executed, depending on the PE to which they are mapped. Further, tasks are annotated with deadlines dl that have to be met during application run-time. If two dependent tasks are assigned to different PEs, π_x and π_y with $x \neq y$, then the communication takes place over a CL, involving a certain amount of communication time and power.

We assume that the task graph is mapped and scheduled onto the target architecture, i.e., it is known where and in which order tasks and communications take place. Fig. 1(a) shows an example task graph that has been mapped onto an architecture and Fig. 1(b) depicts a possible execution order. On top of the precedence relations given by data dependencies between tasks, we introduce additional precedence relations $r \in \mathcal{R}$, generated as result of scheduling tasks mapped to the same PE and communications mapped on the same CL. In Fig. 1(c) the dependencies $\mathcal{R}$ are represented as dotted edges. We define the set of all edges as $\mathcal{E} = \mathcal{C} \cup \mathcal{R}$. Further, we define the set $\mathcal{E}^{\bullet} \subseteq \mathcal{E}$ of edges, as follows:

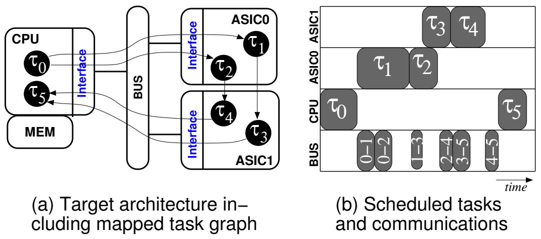

CPU
MEM
Interface
BUS
ASIC0
ASIC1
time
(a) Target architecture including mapped task graph
(b) Scheduled tasks and communications
(c) Scheduled task graph

Figure 1. System models

an edge $(i,j) \in \mathcal{E}^{\bullet}$ if it connects task τ_i with its immediate successor τ_j (according to the schedule), where τ_i and τ_j are mapped on the same PE.

2.2 Power and Delay Models

Digital CMOS circuitry has two major sources of power dissipation: (a) dynamic power P_{dyn}, which is dissipated whenever active computations are carried out (switching of logic states), and (b) leakage power P_{leak} which is consumed whenever the circuit is powered, even if no computations are performed. The dynamic power is expressed by [4, 13],

$$P_{dyn} = C_{eff} \cdot f \cdot V_{dd}^2 \qquad (1)$$

where C_{eff}, f, and V_{dd} denote the effective charged capacitance, operational frequency, and circuit supply voltage, respectively. Although, until recently, the dynamic power dissipation had been dominating, the trend to reduce the overall circuit supply voltage and consequently threshold voltage is rising concerns about the leakage currents—for near future technology ($< 70nm$) it is expected that leakage will account for more than 50% of the total power. The leakage power is given by [13],

$$P_{leak} = L_g \cdot V_{dd} \cdot K_3 \cdot e^{K_4 \cdot V_{dd}} \cdot e^{K_5 \cdot V_{bs}} + |V_{bs}| \cdot I_{Ju} \qquad (2)$$

where V_{bs} is the body bias voltage and I_{Ju} represents the body junction leakage current. The fitting parameters K_3, K_4 and K_5 denote circuit technology dependent constants and L_g reflects the number of gates. For clarity reasons we maintain the same indices as used in [13], where also actual values for these constants are given.

Nevertheless, scaling the supply and the body bias voltage, in order to reduces the power consumption, has a side effect on the circuit delay d [4, 13]:

$$d = L_d \cdot K_6 \cdot \frac{V_{dd}}{(V_{dd} - V_{th})^{\alpha}} \qquad (3)$$

where α reflects the velocity saturation imposed by the used technology (common values $1.4 \leq \alpha \leq 2$), L_d is the logic depth, K_6 is a circuit dependent constant, and V_{th} denotes the threshold voltage. Depending on the supply voltage and the body bias voltage, the threshold voltage is expressed as [13],

$$V_{th} = V_{th1} - K_1 \cdot V_{dd} - K_2 \cdot V_{bs} \qquad (4)$$

where V_{th1}, K_1, and K_2 are fitting constants for a given technology. According to Eq. (3) and Eq. (4), the operational frequency for a certain supply voltage and body bias voltage is derived as [13],

$$f = 1/d = \frac{((1+K_1) \cdot V_{dd} + K_2 \cdot V_{bs} - V_{th1})^{\alpha}}{K_6 \cdot L_d \cdot V_{dd}} \qquad (5)$$

Another important issue, which often is overlooked in voltage scaling approaches, is the consideration of transition overheads, i.e., each time the processor's supply voltage and body bias voltage are altered, the change requires a certain amount of extra energy and time. These energy $\varepsilon_{k,j}$ and delay $\delta_{k,j}$ overheads, when switching from V_{dd_k} to V_{dd_j} and from V_{bs_k} to V_{bs_j}, are given by [13],

$$\varepsilon_{k,j} = C_r \cdot |V_{dd_k} - V_{dd_j}|^2 + C_s \cdot |V_{bs_k} - V_{bs_j}|^2 \qquad (6)$$

$$\delta_{k,j} = \max(p_{Vdd} \cdot |V_{dd_k} - V_{dd_j}|, p_{Vbs} \cdot |V_{bs_k} - V_{bs_j}|) \qquad (7)$$

where C_r denotes power rail capacitance, and C_s the total substrate and well capacitance. Since transition times for V_{dd} and V_{bs} are different, the two constants p_{Vdd} and p_{Vbs} are used to calculate both time overheads independently. Considering that supply and body bias voltage can be scaled in parallel, the transition overhead $\delta_{k,j}$ depends on the maximum time required to reach the new voltage levels.

Voltage scaling is only rewarding if the energy saved through optimized voltages is not outdone by the transition overheads in energy. Furthermore, it is obvious that disregarding transition time overhead can seriously affect the schedulablity of real time systems.

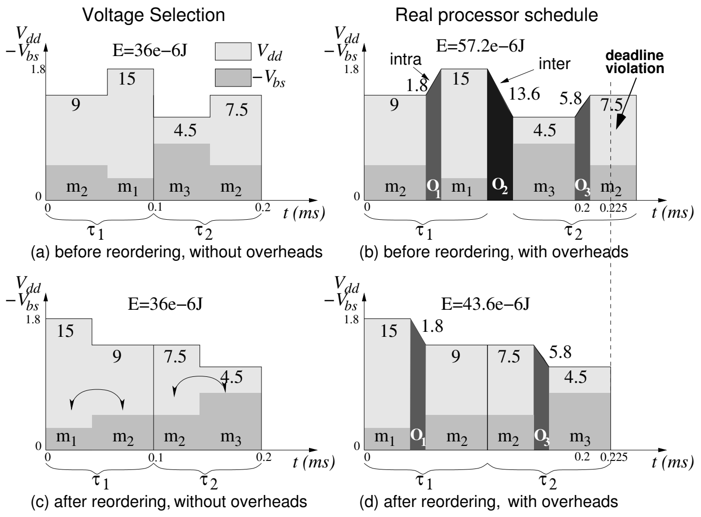


Figure 2. Influence of transition overheads

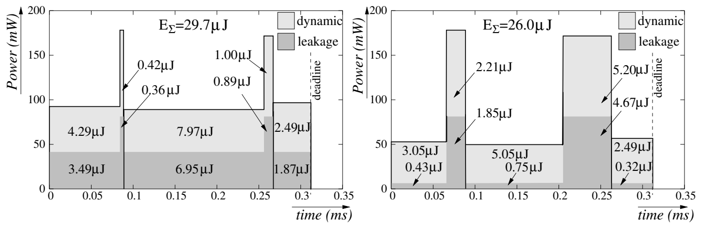
2.3 Mathematical Programming

In this subsection we briefly outline some useful mathematical programming issues, which are relevant for the rest of the paper. Mathematical programming offers methods for solving problems of minimizing or maximizing an objective function $f(x_1,\ldots,x_n)$, with respect to a set of m constraints $g_j(x_1,\ldots,x_n) \leq c_j$ $(j = 1,\ldots,m)$ and bounds for the n variables $(lb_i \leq x_i \leq ub_i,\ i = 1,\ldots,n)$. If both the objective function f and the constraints g_j are linear functions, the problem is called linear programming (LP). Further, if some of the variables are restricted to the integer domain, the problem is called mixed integer linear programming (MILP). If either f or g_j are nonlinear functions, we have a nonlinear programming (NLP) problem. If both f and g_j are convex functions and the variables are ranged over a continuous domain, the problem is called convex nonlinear programming (convex NLP). Solving MILP problems was proved to be NP-complete[1]. For LP as well as for convex NLP efficient algorithms with polynomial complexity are available [15].

3 Motivational Example

To demonstrate the influence of the transition overheads in terms of energy and delay, consider the following motivational example. For clarity reasons we restrict ourselves here to a single processor system that offers three voltage modes, $m_1 = (1.8V, -0.3V)$, $m_2 = (1.5V, -0.45V)$, and $m_3 = (1.2V, -0.8V)$, where $m_z = (V_{dd_z}, V_{bs_z})$. The rail and substrate capacitance are given as $C_r = 10\mu F$ and $C_s = 40\mu F$. The processor needs to execute two consecutive tasks (τ_1 and τ_2) with a deadline of $0.225ms$. Fig. 2(a) shows a possible voltage schedule. As we can observe, each of the two tasks is executed in two different modes: task τ_1 executes first in mode m_2 and then in mode m_1, while task τ_2 is initially executed in mode m_3 and then in mode m_2. The total energy consumption of this schedule is the sum of the energy dissipation in each mode $E = 9 + 15 + 4.5 + 7.5 = 36\mu J$. However, if this voltage schedule is applied to a *real* voltage-scalable processor, the resulting schedule will be influenced by transition overheads, as shown in Fig. 2(b). Here the processor requires a finite time to adapt to the new execution mode. During this adaption no computations can be performed [1, 2], i.e., the task execution is delayed, which, in turn, increases the schedule length such that the imposed deadline is violated. Moreover, transitions do not only require time, they also cause an additional energy dissipation. For instance, in the given schedule, the first transition overhead O_1 from mode m_2 and m_1 requires an energy of $10\mu F \cdot (1.8V - 1.5V)^2 + 40\mu F \cdot (0.3V - 0.45V)^2 = 1.8\mu J$, based on Eq. (6). Similarly, the energy overheads for transitions O_2 and O_3 can be calculated as $13.6\mu J$ and $5.8\mu J$, respectively. The overall energy dissipation of the realistic schedule shown in Fig. 2(b) accumulates to $36 + 1.8 + 13.6 + 5.8 = 57.2\mu J$.

[1]For some subclasses, e.g. convex objectives with linear constraints, there exist polynomial algorithms that solve the MILP formulation.



(a) V_{dd} scaling only

(b) Simultaneous V_{dd} and V_{bs} scaling

Figure 3. Influence of V_{bs} scaling

Let us consider a second possibility of ordering the modes, as given in Fig. 2(c). Compared to the schedule in Fig. 2(a), the mode activation order in Fig. 2(c) has been swapped for both tasks. As long as the transition overheads are neglected, the energy consumption of the two schedules is identical. However, applying the second activation order to a real processor would result in the schedule shown in Fig. 2(d). We can observe that this schedule exhibits only two mode transitions (O_1 and O_3) within the tasks (intra switches), while the switch between the two tasks O_2 (inter switch) has been eliminated. The overall energy consumption has been reduced to $E = 43.6\mu J$, a reduction by 23.8% compared to the schedule given in Fig. 2(b). Further, the elimination of transition O_2 reduces the overall schedule length, such that the imposed deadline is satisfied. With this motivational example we have demonstrated the effects that transition overheads can have on the energy consumption and the timing behavior and the impact of taking them into consideration when elaborating the voltage schedule. However, the approaches presented in this paper do not only achieve energy efficiency by considering transition overheads, but further take into account the simultaneous scaling of the supply voltage V_{dd} and body bias voltage V_{bs}. To illustrate the advantage of this simultaneous scaling over supply voltage scaling only, consider the following example.

Fig. 3 shows two optimal voltage schedules for the same set of three tasks, executing in two possible voltage modes. While the first schedule relies on V_{dd} scaling only, the second schedule corresponds to the simultaneous scaling of V_{dd} and V_{bs}. Please note that the figures depict the dynamic and the leakage power dissipation over time, unlike Fig. 2 which showed V_{dd} and V_{bs} over time. Further, for the simplicity we neglect transition overheads in this example. Further, we consider processor parameters that correspond to CMOS technology ($< 70nm$) which leads to a leakage power consumption close to 50% of the total power consumed. Let us consider the first schedule in which the tasks are executed either at $V_{dd1} = 1.8V$, or $V_{dd2} = 1.5V$. In accordance, the system dissipates either a high amount of dynamic and leakage power, or a low amount, as observable from the figure. We have indicated the individual energy consumed in each of the active modes, separating between dynamic and leakage energy. Correspondingly the total leakage energy and the total dynamic energy of the schedule in Fig. 3(a) are given by $13.56\mu J$ and $16.17\mu J$, respectively. This results in a total energy consumption of $29.73\mu J$.

Consider now the schedule given in Fig. 3(b), where tasks are executed at two different voltage settings for V_{dd} and V_{bs} ($m_1 = (1.8V, 0V)$ and $m_2 = (1.5V, -0.4V)$). There are to main differences to observe. Firstly, the leakage power consumption during mode m_2 is considerably smaller than the leakage power consumptions given in the schedule of Fig. 3(a); this is due to the fact that m_2 reduces the leakage through a body bias voltage of $-0.4V$ (see Eq. (2)). Secondly, the high voltage mode m_1 is active for more time; which can be explained by the fact that scaling V_{bs} during mode m_2 requires the reduction of the operational frequency (see Eq. (5)), hence to meet the system deadline high performance mode m_1 has to compensate for this delay. Nevertheless, the total leakage and dynamic energies results in $8.02\mu J$ and $18.00\mu J$, respectively. Although here the dynamic energy was increased from $16.17\mu J$ to $18.0\mu J$, compared to the first schedule, the leakage was reduced from $13.56\mu J$ to $8.02\mu J$. The overall energy dissipation becomes then $26.02\mu J$, a reduction by 12.5%. This small illustrative examples shows the advantage of simultaneous V_{dd} and V_{bs} scaling compared to V_{dd} scaling only.

4 Problem Formulation

Consider a set of tasks with precedence constraints $\mathcal{T} = \{\tau_i\}$ which have been mapped and scheduled on a set of variable voltage processors. For each task τ_i its deadline dl_i, its number of clock cycles to be executed NC_i and the switched capacitance C_{eff_i} are given. Each processor can vary its supply voltage V_{dd} *and* body bias voltage V_{bs} within certain continuous ranges (for the continuous problem), or, within a set of discrete voltages pairs $m_z = \{(V_{dd_z}, V_{bs_z})\}$ (for the discrete problem). The power dissipations (leakage, dynamic) and the cycle time (processor speed) depend on the selected voltage pair (mode). Tasks are executed cycle by cycle, and each cycle can potentially execute at a different voltage pair, i.e., at a different speed. Our goal is to find voltage pair assignments for each task such that the individual task deadlines are met and the total energy consumption is minimal. Furthermore, whenever the processor has to alter the settings for V_{dd} and/or V_{bs}, a transition overhead in terms of energy and time is required (see Eqs. (6) and (7)).

For reasons of clarity we introduce the following four distinctive problems which will be considered in this paper: (a) Continuous voltage scaling with no consideration of transition overheads (CNOH), (b) continuous voltage scaling with consideration of transition overheads (COH), (c) discrete voltage scaling with no consideration of transition overheads (DNOH), and (d) discrete voltage scaling with consideration of transition overheads (DOH).

5 Optimal Continuous Voltage Selection

In this section we consider that supply and body bias voltage of the processors in the system can be selected within a certain continuous range. We first formulate the problem neglecting the transition overheads (Section 5.1, CNOH) and then extend this formulation to include the overheads in energy and delay (Section 5.2, COH).

5.1 Continuous Voltage Selection without Overheads

We can model the continuous voltage scaling problem excluding the consideration of transition overheads (the CNOH problem), using the following nonlinear problem formulation.

Minimize

$$\sum_{k=1}^{|\mathcal{T}|}\left(\underbrace{NC_k \cdot C_{eff_k} \cdot V_{dd_k}^2}_{E_{dyn_k}} + \underbrace{L_g(K_3 \cdot V_{dd_k} \cdot e^{K_4 \cdot V_{dd_k}} \cdot e^{K_5 \cdot V_{bs_k}} + I_{Ju} \cdot |V_{bs_k}|) \cdot t_k}_{E_{leak_k}}\right) \tag{8}$$

subject to

$$t_k = NC_k \cdot \frac{(K_6 \cdot L_d \cdot V_{dd_k})}{((1+K_1) \cdot V_{dd_k} + K_2 \cdot V_{bs_k} - V_{th_1})^\alpha} \tag{9}$$

$$D_k + t_k \leq D_l \quad \forall (k,l) \in \mathcal{E} \tag{10}$$

$$D_k + t_k \leq dl_k \quad \forall\ \tau_k \text{ that have a deadline} \tag{11}$$

$$D_k \geq 0 \tag{12}$$

$$V_{dd_{min}} \leq V_{dd_k} \leq V_{dd_{max}} \text{ and } V_{bs_{min}} \leq V_{dd_k} \leq V_{bs_{max}} \tag{13}$$

The variables that need to be optimized in this formulation are the task execution times t_k, the task start times D_k as well as the voltages V_{dd_k} and V_{bs_k}. The whole formulation can be explained as follows. The total energy consumption, which is the combination of dynamic and leakage energy, has to be minimized, as in Eq. (8)[2]. The minimization has to comply to the following relations and constraints. The task execution time has to be equivalent to the number of clock cycles of the task multiplied by the circuit delay for a particular V_{dd_k} and V_{bs_k} setting, as expressed by Eq. (9). Given the execution time of the tasks, it becomes possible to express the precedence constraints between tasks (Eq. (10)), i.e., a task τ_l can only start its execution after all its predecessor tasks τ_k have finished their execution ($D_k + t_k$). Predecessors of task τ_l are all tasks τ_k for which there exists an edge $(k,l) \in \mathcal{E}$. Similarly, tasks with deadlines have to be completed ($D_k + t_k$) before their deadlines dl_k are exceeded (Eq. (11)). Task start times have to be positive (Eq. (12))

[2]Please note that *abs* and *max* operations cannot be used directly in mathematical programming, yet there exist standard techniques to overcome this limitation by equivalent formulations [20].

and the imposed voltage ranges should be respected (Eq. (13)). It should be noted that the objective (Eq. (8)) as well as the task execution time (Eq. (9)) are convex functions. Hence, the problem falls into the class of general convex nonlinear optimization problems. As outlined in Section 2.3, such problems can be solved in polynomial time. For clarity reasons, in this paper, we did not include communication issues into the constraints and objective function. Nevertheless, they can be included in a straightforward way, by modeling communication links as non-scalable processors and communications as tasks mapped to such processors [19].

5.2 Continuous Voltage Selection with Overheads

In this section we modify the previous formulation in order to take transition overheads into account (COH problem). The following formulation highlights the modifications.

$$\text{Minimize} \quad \sum_{k=1}^{|\mathcal{T}|}(E_{dyn_k}+E_{leak_k})+\sum_{(k,j)\in\mathcal{E}^\bullet}\varepsilon_{k,j} \tag{14}$$

$$\text{subject to} \quad D_k+t_k+\delta_{k,j}\le D_j \quad \forall(k,j)\in\mathcal{E}^\bullet \tag{15}$$

$$\delta_{k,j}=\max(p_{Vdd}\cdot|V_{dd_k}-V_{dd_j}|,p_{Vbs}\cdot|V_{bs_k}-V_{bs_j}|) \tag{16}$$

As we can see, the objective function Eq. (14) now additionally accounts for the transition overheads in terms of energy. The energy overheads can be calculated according to Eq. (6) for all consecutive tasks τ_k and τ_j on the same processor ($\mathcal{E}^\bullet$ is defined in Section 2.1). However, scaling voltages does not only require energy but it introduces delay overheads as well. These overheads might delay the start times of subsequent tasks. Therefore, we introduce an additional constraint similar to Eq. (10), which states that a task τ_j can only start after the execution of its predecessor τ_k (D_k+t_k) on the same processor and after the new voltage mode is reached ($\delta_{k,j}$). This constraint is given in Eq. (15). The delay penalties $\delta_{k,j}$ are introduced as a set of new variables and are constrained subject to Eq. (16). Similar to the formulation of the CNOH problem, the COH model is a convex nonlinear problem, i.e., it can be solved in polynomial time.

6 Optimal Discrete Voltage Selection

In the previous section, we have shown how continuous voltage scaling can be solved optimally in polynomial time. Voltage scaling was performed for both V_{dd} and V_{bs}. Furthermore, the consideration of transition overheads was introduced into the model. These approaches provide a theoretical lower bound on the possible energy savings. In reality, however, processors are restricted to a discrete set of V_{dd} and V_{bs} voltage pairs. In this section we investigate the discrete voltage selection problem without and with the consideration of overheads. We will also analyze the complexity of the discrete voltage selection problem.

6.1 Problem Complexity

Theorem 1 *The discrete voltage scaling problem is NP-hard.*

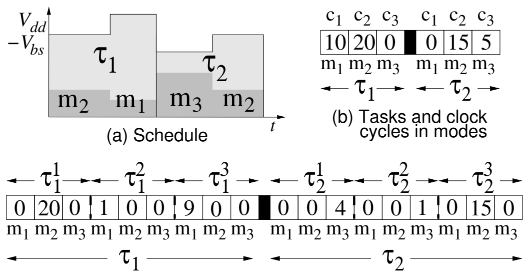
Proof 1 *We proof by restriction. The discrete time-cost tradeoff (DTCT) problem is known to be NP-hard [5]. By restricting the discrete dynamic voltage scaling (DDVS) problem to contain only tasks that require an execution of one clock cycle, it becomes identical to the DTCT problem. Hence, DTCT ∈ DDVS which leads to the conclusion DDVS ∈ NP.* ■

For space reasons, we refer the interested reader to [20], where an exhaustive proof is given. Note, that the problem is NP-hard, even if we restrict it to supply voltage scaling (without adaptive body biasing) and even if transition overheads are neglected. It should be noted that this finding renders the conclusion of [11][3] impossible, which states that the discrete voltage scaling problem (considered in [11] without body biasing and overheads) can be solved optimally in polynomial time.

6.2 Discrete Voltage Selection without Overheads

In the following we will give a mixed integer linear programming (MILP) formulation for the discrete voltage selection problem without overheads (DNOH). We consider that processors can run in different modes $m\in\mathcal{M}$. Each mode m is characterized by a voltage pair

[3]The flaw in [11] lies in the fact that the number of clock cycles spent in a mode is not restricted to be integer.


V_{dd}
$-V_{bs}$
τ_1
τ_2
m_2 m_1 m_3 m_2
t
c_1 c_2 c_3 c_1 c_2 c_3
10 20 0 0 15 5
m_1 m_2 m_3 m_1 m_2 m_3
τ_1 τ_2
τ_1^1 τ_1^2 τ_1^3 τ_2^1 τ_2^2 τ_2^3
0 20 0 1 0 0 9 0 0 0 0 4 0 0 1 0 15 0
m_1 m_2 m_3 m_1 m_2 m_3 m_1 m_2 m_3 m_1 m_2 m_3 m_1 m_2 m_3 m_1 m_2 m_3
τ_1 τ_2

(a) Schedule (b) Tasks and clock cycles in modes (c) Division of task into subtasks and modes

Figure 4. Discrete mode models

(V_{dd_m},V_{bs_m}), which determines the operational frequency f_m, the normalized dynamic power P_{dnom_m}, and the leakage power dissipation P_{leak_m}. The frequency and the leakage power are given by Eqs. (5) and (2), respectively. The normalized dynamic power is given by $P_{dnom_m}=f_m\cdot V_{dd_m}^2$. Accordingly, the dynamic power of a task τ_k operating in mode m is computed as $C_{eff_k}\cdot P_{dnom_m}$. Based to these definitions, the MILP problem is formulated as:

$$\text{Minimize} \quad \sum_{k=1}^{|\mathcal{T}|}\sum_{m\in\mathcal{M}}\left(C_{eff_k}\cdot P_{dnom_m}\cdot t_{k,m}+P_{leak_m}\cdot t_{k,m}\right) \tag{17}$$

$$\text{subject to} \quad D_k+\sum_{m\in\mathcal{M}}t_{k,m} \le dl_k \tag{18}$$

$$D_k+\sum_{m\in\mathcal{M}}t_{k,m} \le D_l \quad \forall(k,l)\in\mathcal{E} \tag{19}$$

$$c_{k,m}=t_{k,m}\cdot f_m \quad \text{and} \quad \sum_{m\in\mathcal{M}}c_{k,m}=NC_k \quad c_{k,m}\in\mathbb{N} \tag{20}$$

$$D_k\ge 0 \quad \text{and} \quad t_{k,m}\ge 0 \tag{21}$$

The total energy consumption, expressed by Eq. (17), is given by two sums. The inner sum indicates the energy dissipated by an individual task τ_k, depending on the time $t_{k,m}$ spent in each mode m. While the outer sum adds up the energy of all tasks. Unlike the continuous voltage scaling case, we do not obtain the voltage V_{dd} and V_{bs} directly, but rather we find out how much time to spend in each of the modes. Therefore, task execution time $t_{k,m}$ and the number of clock cycles $c_{k,m}$ spent within a mode become the variables in the MILP formulation. Of course, the number of clock cycles has to be an integer and hence $c_{k,m}$ is restricted to the integer domain. We exemplify this model graphically in Figures 4(a) and 4(b). The first figure shows the schedule of two tasks executing each at two different voltage settings (two modes out of three possible modes). Task τ_1 executes for 20 clock cycles in mode m_2 and for 10 clock cycles in m_1, while task τ_2 runs for 5 clock cycles in m_3 and 15 clock cycles in m_2. The same is captured in Fig. 4(b) in what we call a mode model. The modes that are not active during a task's runtime have the corresponding time and number of clock cycles 0 (mode m_3 for τ_1 and $m1$ for τ_2). The overall execution time of task τ_k is given as the sum of the times spent in each mode ($\sum_{m\in\mathcal{M}}t_{k,m}$). It should be noted that the model in Fig. 4(b) does not capture the order in which modes are activated, it solely expresses how many clock cycles are spent in each mode. Eq. (18) ensures that all the deadlines are met and Eq. (19) maintains the correct execution order given by the precedence relations. The relation between execution time and number of clock cycles as well as the requirement to execute all clock cycles of a task are expressed in Eq. (20). Additionally, task start times D_k and task execution times have to be equal or larger than zero, as given in Eq. (21).

6.3 Discrete Voltage Selection with Overheads

We now proceed with the incorporation of transition overheads in the MILP formulation given in Section 6.2. Obviously, the order in which the modes are activated has an influence on the transition overheads, as we have already demonstrated in Section 3. Nevertheless, the formulation in Section 6.2 omits information regarding the activation order of modes. For instance, from the Fig. 4(b), we cannot tell if for task τ_1, mode m_1 or m_2 is active first. We introduce the following extensions needed in order to take both delay and energy overheads into account. Given m operational modes, the execution of a single task τ_k can be subdivided into m subtasks $\tau_k^i, i = 1,\ldots,m$. Each subtask is executed in one and only one of the m modes. Subtasks are further subdivided into m slices, each corresponding to a mode. This results in $m \cdot m$ slices for each task. Fig. 4(c) depicts this model, showing that task τ_1 runs first in mode m_2, then in mode m_1, and that τ_2 runs first in mode m_3, then in m_2. This ordering is captured by the subtasks: the first subtask of τ_1 executes 20 clock cycles in mode m_2, the second subtask executes one clock cycle in m_1 and the remaining 9 cycles are executed by the last subtask in mode m_1; τ_2 executes in its first subtask 4 clock cycles in mode m_3, 1 clock cycle is executed during the second subtask in mode m_3, and the last subtask executes 15 clock cycles in the mode m_2. Note, that there is no overhead between subsequent subtasks that are running in the same mode. For instance, the two subtasks τ_1^2 and τ_1^3 run both in mode m_1 and hence there is no switch. In the following, we give the modified MILP formulation:

Minimize

$$\sum_{k=1}^{|\mathcal{T}|}\sum_{s\in\mathcal{M}}\sum_{m\in\mathcal{M}}\left(C_{eff_k}\cdot P_{dnom_m}\cdot t_{k,s,m}+P_{leak_m}\cdot t_{k,s,m}\right)$$
$$+\sum_{k=1}^{|\mathcal{T}|}\sum_{s\in\mathcal{M}}\sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{M}}\left(b_{k,s,i,j}\cdot EP_{i,j}\right) \qquad (22)$$

subject to

$$\delta_k = \sum_{s\in\mathcal{M}^*}\sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{M}} b_{k,s,i,j}\cdot DP_{i,j} \qquad (23)$$

$$\delta_{k,l} = \sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{M}} b_{k,m,i,j}\cdot DP_{i,j} \quad \text{where } (k,l)\in\mathcal{E}^\bullet \qquad (24)$$

$$D_k + \sum_{s\in\mathcal{M}}\sum_{m\in\mathcal{M}} t_{k,s,m} + \delta_k \le dl_k \qquad (25)$$

$$D_k + \sum_{s\in\mathcal{M}}\sum_{m\in\mathcal{M}} t_{k,s,m} + \delta_k + \delta_{pl,l} \le D_l \quad \forall (k,l)\in\mathcal{E}, (pl,l)\in\mathcal{E}^\bullet \qquad (26)$$

$$c_{k,s,i} = t_{k,s,i}\cdot f_i \quad k \text{ in } 1,\ldots,n,\ s \text{ in } 1,\ldots,m,\ i \text{ in } 1,\ldots,m, c\in\mathbb{N} \qquad (27)$$

$$\sum_{s\in\mathcal{M}}\sum_{i\in\mathcal{M}} c_{k,s,i} = NC_k \quad k \text{ in } 1,\ldots,n \qquad (28)$$

In order to capture the energy overheads in the objective function (Eq. (22), we introduce the boolean variables $b_{k,s,i,j}$. In addition, we introduce an energy penalty matrix EP, which contains the energy overheads for all possible mode transitions, i.e., $EP_{i,j}$ denotes the energy overhead necessary to change form mode i to j. These energy overheads are precomputed based on the available modes (voltage pairs) and Eq. (6). The overall energy overhead is given by all intratask and intertask transitions. The intratask and intertask delay overheads, given in Eq. (23) and (24), are calculated based on a delay penalty matrix $DP_{i,j}$, which, similar to the energy penalty matrix, can be precomputed based on the available modes and Eq. (7). For a task τ_k and for each of its subtasks τ_k^s, except the last one, the variable $b_{k,s,i,j} = 1$ if mode i of subtask τ_k^s and mode j of τ_k^{s+1} are both active ($s = 1,\ldots,m-1$, $i, j = 1,\ldots,m$). These are used in order to capture the intratask overheads, as in Eq. (23). For intertask overheads, we are interested in the last mode of task τ_k and the first mode of the subsequent task τ_l (running on the same processor). Therefore, $b_{k,m,i,j} = 1$ if the mode i of the last subtask τ_k^m and the mode j of first subtask τ_l^1 are both active. For the example given in Fig. 4(c), $b_{1,1,2,1}$, $b_{1,2,1,1}$, $b_{1,3,1,3}$, $b_{2,1,3,3}$, $b_{2,2,3,2}$ are all 1 and the rest are 0. Deadlines and precedence relations, taking the delay overheads into account, have to be respected according to Eq. (25) and (26). Here $\sum_{s\in\mathcal{M}}\sum_{m\in\mathcal{M}} t_{k,s,m}$ represents the total execution time of a task τ_k, based on the number of cycles in each of the subtasks and modes. Eq. (27) and (28) are a reformulation of Eq. (20), which expresses the relation between the execution time and the number of clock cycles and the requirement to execute all clock cycles of a task. Due to space limits, we refer the reader to [20], where more details regarding this MILP formulation are given. In particular, we have omitted here details on the computation of the b variables as well as the constraints that make sure that one and only one mode must be used by a subtask.

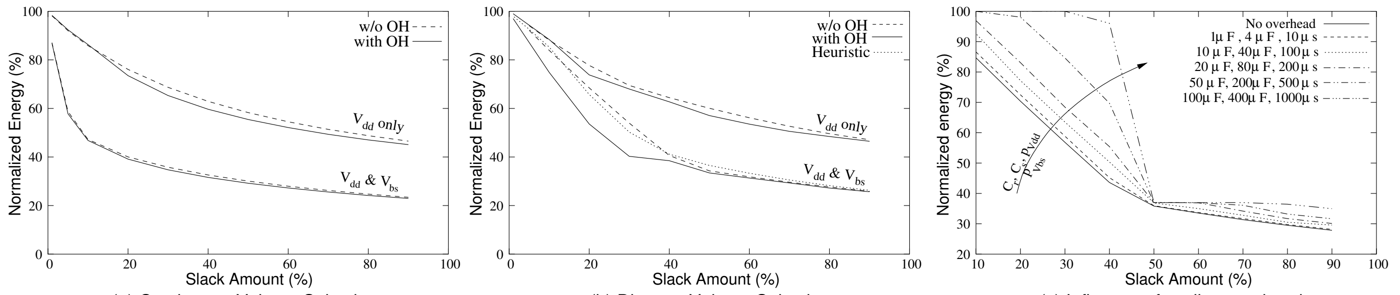
7 Experimental Results

We have conducted a set of experiments using numerous generated benchmarks as well as a real-life GSM voice codec example, in order to demonstrate the applicability of the presented approaches. The automatically generated benchmarks consist of 75 task graphs containing between 10 and 150 tasks, which are mapped and scheduled onto architectures composed of 1 to 3 processors. The technology dependent parameters of these processors were considered to correspond to a CMOS fabrication in 70nm, for which the leakage power represents approximately 50% of the total power consumed. For experimental purpose the amount of deadline slack in each benchmark was varied over a range 0 to 90%, using a 10% increment, resulting in 750 performed evaluations, carried out with the aim to achieve representative average values.

The first set of experiments was conducted in order to demonstrate the achievable energy savings when comparing the classic V_{dd} selection with simultaneous V_{dd} and V_{bs} selection. Fig. 5(a) shows the outcomes for the continuous voltage selection with and without the consideration of transition overheads. The continuous voltage ranges were set to $0.6V \le V_{dd} \le 1.8V$ and $-1V \le V_{bs} \le 0$. The figure shows the percentage of total energy consumed (relative to the baseline energy) as a function of the available slack within the application. As a baseline we consider the energy consumption at the nominal (highest) voltage for V_{dd} and V_{bs}. It is easy to observe the advantage of the combined voltage selection scheme over the classical voltage selection, with a difference of up to 40%. These observations hold with and without the consideration of overheads. Regarding the overhead influence on the overall energy consumption, we can see from the figure that the savings are around 1% for the combined scheme and 2% for the classical voltage selection. These moderate amounts of additional savings have a straightforward explanation: Within the continuous scheme (which from a practical point of view is unrealistic), the voltage differences between tasks are likely to be small, i.e., large overheads are avoided (see Eq. (6) and Eq. (7)).

We have further evaluated the discrete voltage selection scheme. Here the processors could switch between three different voltage settings $(1.8, 0)$, $(1.5, -0.4)$, and $(1.2, -0.6)$ for the combined scheme, and 1.8, 1.5, and 1.2 for the classical V_{dd} selection. The results are given in Fig 5(b). As in the continuous case, we can observe a difference between the classical supply voltage selection and the more efficient combined selection scheme. For low amounts of slack (around 10%), the savings for the combined selection are significantly lower than in the continuous case. The reason for this is that, due to the small slack available, the processors have to run in the highest voltage mode, which does not reduce leakage power. Further, we can see that with increasing slack, the overall energy approaches the theoretical minimum given by the continuous case, since more time is spent in the energy-efficient mode m_3. It is interesting to observe the influence of the transition overheads, in particular when not much system slack is available. In this situation the unnecessary switching between voltages to exploit the "small" amounts of slack causes an increased energy overhead. Consider, for instance, the cases where the combined V_{dd} and V_{bs} selection has been optimized with and without overheads. Between 10% to 40% of slack, the consideration of transition overheads results in improved solutions with up to 12% higher savings. Of course, with increasing slack the number of tasks executed at the lowest voltage setting increases, and hence the number of transitions is decreased. As a result, the influence of the transition overheads reduces. It should be noted that the reported results for the discrete scheme have been evaluated using graphs with at most 80 tasks (without overhead, DNOH) and 30 tasks (with overhead, DOH), since the required optimization times become intractable, as a result of the NP-hardness of the problem (Section 6.1). To overcome this problem we have additionally investigated a voltage selection heuristic. This heuristic uses the voltage schedules derived from the continuous selection (COH). For each selected continuous task voltage, the two surrounding discrete voltage pairs are chosen (similar to the classical approach proposed in [9]). In order to minimize overheads, we perform a simple reordering of mode activations. The results of this simple heuristic follows the discrete voltage selection without overheads, as shown in Fig 5(b). However, due to its relatively reduced polynomial time complexity, it can be applied to large instances of the problem.

In order to further investigate the influence of transition overheads, we have carried out an additional set of experiments in which the amount



(a) Continuous Voltage Selection (b) Discrete Voltage Selection (c) Influence of scaling overheads

Figure 5. Optimization Results

of processors' overheads in energy and delay were varied by adjusting the values for C_r, C_s, p_{Vdd}, and p_{Vbs}. In accordance, we use the discrete voltage selection with consideration of overheads. The results are given in Fig 5(c). As expected, the energy dissipation increases for higher values of the overhead determining parameters. For instance, while a "hypothetical" processor which requires no transition overheads can reduce the energy consumption by 58% if 40% of slack is available, a realistic processor with $C_r = 20\mu F$, $C_s = 80\mu F$, $p_{Vdd} = 200\mu s/V$, and $p_{Vbs} = 200\mu s/V$ achieves only 42%. This highlights the importance to carefully consider the influence of transition overheads.

In addition to the above given benchmark results, we have conducted experiments on a real-life GSM voice codec application, in order to validate the real-world applicability of the presented techniques. Details regarding this application can be found in [19]. The GSM codec consists of 87 tasks and is considered to run on an architecture composed of 3 processing elements with two voltage modes ($(1.8V, -0.1V)$ and $(1.0V, -0.6)$). At the highest voltage mode, the application reveals a deadline slack close to 10%. Switching overheads are characterized by $C_r = 1\mu F$, $C_s = 4\mu F$, $p_{Vdd} = 10\mu s/V$, and$p_{Vbs} = 10\mu s/V$. Since the processing elements can only run at these discrete voltages, we restrict the following discussion to discrete voltage selection. Tab. 1 shows the resulting energy consumptions in terms of dynamic E_{dyn}, leakage E_{leak}, overhead ε, and total E_Σ energy (Columns 2–5). Each line represents

Approach	E_{dyn}	E_{leak}	ε	E_Σ	Reduc. %
Nominal	1.342	0.931	no	2.273	—
DVDDNOH	1.276	0.892	0.051	2.219	2.34
DVDDOH	1.277	0.892	0.005	2.174	4.38
DNOH	1.292	0.625	0.168	2.085	9.91
DOH	1.294	0.626	0.010	1.931	15.18
Heuristic	1.324	0.617	0.112	2.053	9.67

Table 1. Optimization results for voice codec algorithm

a different voltage selection approach. Line 2 (Nominal) is used as a baseline and corresponds to an execution at the nominal voltages. The lines 3 and 4 give the results for the classical V_{dd} selection, without (DVDDNOH) and with (DVDDOH) the consideration of overheads. As we can see, the consideration of overheads achieves higher energy saving (4.38%) than the overhead neglecting optimization (2.34%). Although the dynamic energy is slightly increased when considering the overheads, the total energy is minimized due to the reduction of transition overheads. The results given in lines 5 and 6 correspond to the combined V_{dd} and V_{bs} selection schemes. Again we distinguish between overheads neglecting (DNOH) and overhead considering (DOH) approaches. If the overheads are neglected, the energy consumption can be reduced by 9.91%, yet taking the overheads into account results in an reduction of 15.18%, solely achieved by decreasing the transition overheads. Compared to the classical voltage selection scheme (4.38% savings), the combined selection achieved a further reduction of 10.8%. These experiments underline how the consideration of transition overheads helps in achieving energy-efficient voltage schedules. For comparison, the last line shows the results of the heuristic approach. Although the result does not match the optimal one given in line 6, it should be noted that such heuristic techniques are needed when dealing with problems of larger complexity (increased number of voltage modes and tasks). In the GSM application, although the number of tasks is realistically large, we considered only two voltage modes. Therefore the optimal solutions could be obtained for the DOH problem. Overall, the conducted experiments have demonstrated the advantages of the combined voltage selection over the classical V_{dd} scheme. Furthermore, it was shown that the consideration of transition overheads has a profound impact on the overall achievable energy savings.

8 Conclusions

Energy reduction techniques, such as dynamic voltage scaling and adaptive body biasing can be effectively exploited at the system-level. In this paper, we have investigated different notions of the combined dynamic voltage scaling and adaptive body biasing problem at the system-level. These include the consideration of transition overheads as well as the discretization of the supply and threshold voltage levels. It was demonstrated that nonlinear programming and mixed integer linear programming formulations can be used to solve these problems. Further, the NP-hardness of the discrete voltage scaling case was shown. Several generated benchmark examples as well as a real-life voice codec example were used to show the applicability of the introduced approaches.

References

[1] Intel® XScale^{TM} Core, Developer's Manual, December 2000.

[2] Mobile AMD Athlon^{TM}4, Processor Model 6 CPGA Data Sheet, November 2000. Publication No 24319 Rev E.

[3] S. Borkar. Design Challenges of Technology Scaling. *IEEE Mirco*, pages 23–29, July 1999.

[4] A. P. Chandrakasan and R. W. Brodersen. *Low Power Digital CMOS Design*. Kluwer Academic Publisher, 1995.

[5] P. De, E. Dunne, J. Ghosh, and C. Wells. Complexity of the Discrete Time-Cost Tradeoff problem for Project Networks. *Operations Research*, 45(2):302–306, March 1997.

[6] D. Duarte, N. Vijaykrishnan, M. Irwin, H. Kim, and G. McFarland. Impact of Scaling on The Effectiveness of Dynamic Power Reduction. In *Proc. ICCD*, Sept. 2002.

[7] F. Gruian and K. Kuchcinski. LEneS: Task Scheduling for Low-Energy Systems Using Variable Supply Voltage Processors. In *Proc. ASP-DAC'01*, pages 449–455, Jan 2001.

[8] I. Hong, G. Qu, M. Potkonjak, and M. B. Srivastava. Synthesis Techniques for Low-Power Hard Real-Time Systems on Variable Voltage Processors. In *Proc. Real-Time Systems Symposium*, 1998.

[9] T. Ishihara and H. Yasuura. Voltage Scheduling Problem for Dynamically Variable Voltage Processors. In *Proc. Int. Symp. Low Power Electronics and Design (ISLPED'98)*, pages 197–202, 1998.

[10] C. Kim and K. Roy. Dynamic Vth Scaling Scheme for Active Leakage Power Reduction. In *Proc. Design, Automation and Test in Europe Conf. (DATE02)*, pages 163–167, March 2002.

[11] W. Kwon and T. Kim. Optimal Voltage Allocation Techniques for Dynamically Variable Voltage Processors. In *Proc. IEEE DAC'03*, pages 125–130, June 2003.

[12] J. Luo and N. Jha. Power-profile Driven Variable Voltage Scaling for Heterogeneous Distributed Real-time Embedded Systems. In *Proc. VLSI'03*, 2003.

[13] S. Martin, K. Flautner, T. Mudge, and D. Blaauw. Combined Dynamic Voltage Scaling and Adaptive Body Biasing for Lower Power Microprocessors under Dynamic Workloads. In *Proc. ICCAD-02*, pages 721–725, 2002.

[14] B. Mochocki, X. Hu, and G. Quan. A Realistic Variable Voltage Scheduling Model for Real-Time Applications. In *Proc. ICCAD-02*, pages 726–731, 2002.

[15] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. Studies in Applied Mathematics, 1994.

[16] F. Yao, A. Demers, and S. Shenker. A Scheduling Model for Reduced CPU Energy. *IEEE FOCS*, 1995.

[17] Y. Zhang, X. Hu, and D. Chen. Task Scheduling and Voltage Selection for Energy Minimization. In *Proc. IEEE DAC'02*, June 2002.

[18] Y. Zhang, X. Hu, and D. Chen. Energy Minimization of Real-time Tasks on Variable Voltage Processors with Transition Energy Overhead. In *Proc. ASP-DAC'03*, pages 65–70, 2003.

[19] omitted for blind review.

[20] Technical Report. omitted for blind review. ftp://...