Optimal two-level speed assignment for real-time systems

Enrico Bini*

Scuola Superiore Sant'Anna
Via G. Moruzzi 1, 56124, Pisa, Italy
E-mail: e.bini@sssup.it
*Corresponding author

Claudio Scordino

Evidence Srl,
Via Carducci 64/A, 56010 Pisa, Italy
E-mail: claudio@evidence.eu.com

Abstract: Reducing energy consumption is one of the main concerns in the design and implementation of embedded real-time systems. For this reason, the current generation of processors allows to vary voltage and operating frequency to balance computational speed and energy consumption. This technique is called dynamic voltage scaling (DVS).

When applying DVS to hard real-time systems, it is important to provide the worst-case computational requirement; otherwise the timing constraints may be violated. However, the probability of a task executing for its worst-case execution time is very low.

In this paper, we show how to exploit probabilistic information about the execution time of a task in order to reduce the energy consumed by the processor. Optimal speed assignments and transition points are found using a very general model for the processor. The model accounts for the processor idle power and time/energy overheads due to frequency transitions. We also show how these results apply to some significant cases.

Keywords: dynamic voltage scaling; DVS; real-time; power-aware computing; probabilistic execution time.

Reference to this paper should be made as follows: Bini, E. and Scordino, C. (2009) 'Optimal two-level speed assignment for real-time systems', *Int. J. Embedded Systems*, Vol. 4, No. 2, pp.101–111.

Biographical notes: Enrico Bini is an Assistant Professor at the Scuola Superiore Sant'Anna, Pisa, Italy. He completed his PhD at the same institution in 2004. His interests include real-time scheduling, control systems and optimisation methods.

Claudio Scordino is a PhD student and a Teaching Assistant at the University of Pisa collaborating with Scuola Superiore Sant'Anna. His research activities include operating systems, real-time scheduling, energy saving and embedded devices.

1 Introduction

The number of embedded systems operated by batteries is increasing in different application domains, from personal digital assistants (PDAs) to autonomous robots, smart phones and sensor networks. Reducing the energy consumed by these systems has become a key design issue, as they can only operate on the limited battery supply. Battery lifetime is a critical design parameter, and it affects directly the size and the weight of the system.

In recent years, the problem of energy reduction arose in the real-time servers' area as well (Rusu et al., 2006). As processors become more and more powerful, their energy consumption increases correspondingly, and it becomes a problem to dissipate the heat produced (Lefurgy et al., 2003; Bianchini and Rajamony, 2004). In fact, the increase of the computational power in current digital systems is mostly obtained by increasing the clock frequency, leading to a higher energy demand (Pouwelse et al., 2001; Ishihara and Yasuura, 1998; Lorch and Smith, 2001; Rusu et al., 2006) and a greater heat generated. Conventional computers are currently air-cooled, and manufacturers are facing the problem of building powerful systems without introducing additional techniques such as liquid cooling (Lefurgy et al., 2003). Not surprisingly, a significant portion of the consumed energy is due to cooling devices, which may consume up to 50% of the total energy (Lefurgy et al., 2003). For example, a 10 kW rack consumes about 10 MWh a month (including cooling), which represents at least 10% of the operation cost (Barroso et al., 2003), with

this trend likely to increase in the future. Reducing the energy consumed by the computing components would impact on the energy consumed by the cooling devices and, eventually, on the cost of the system.

To prolong the lifetime of all these systems, the energy consumption should be reduced to an absolute minimum through energy-aware techniques. For this reason, the current generation of processors (Intel Corp., 2004a, 2004b; MPC5200; Crusoe, 2003; Intel Centrino) allow the operating system to vary dynamically the operating frequency to balance computational speed versus energy consumption. This technique is called dynamic voltage scaling (DVS), and it is used by many energy-aware scheduling policies (Scordino and Lipari, 2004; Pillai and Shin, 2001; Aydin et al., 2004; Qadi et al., 2003; Zhu and Mueller, 2004). However, a weakness of most approaches is due to the set of assumptions, often not realistic, which are made to simplify the solution. Besides ignoring the energy consumed by the processor when it is idle, these methods often neglect also the delay due to a frequency transition, preventing thus the application of research results to real-world systems. In some approaches (Lee and Sakurai, 2000; Mochocki et al., 2002), such a delay has been considered in the processor model, but only dynamic techniques aimed at reducing the slack time have been developed. For this reason, we have formulated a general model for the processor (Bini et al., 2005; Scordino and Bini, 2005), which takes into account both time and energy overheads due to frequency transition.

In real-time systems, any energy-aware policy acting on the processor speed must take timing constraints into account, to guarantee the timely execution of the computational activities. For safety reasons, hard real-time systems are typically designed to handle peak loads. However, peak load conditions rarely happen in practice, and the system resources are underutilised most of the time. For example, server loads often vary significantly depending on the time of the day or other external factors. Researchers at IBM showed that the average processor utilisation of real servers is between 10% and 50% of their peak capacity (Bohrer et al., 2002). Thus, much of the server capacity remains unused during normal operations (Rusu et al., 2006). These issues are even more critical in embedded systems (Xu et al., 2005), where the peak power has an important impact on the size and the cost of the system. Some studies have observed (Wegener and Mueller, 2001) that the actual execution times of tasks in real-world embedded systems can vary up to 80% with respect to their measured worst case execution cycles (WCECs). This suggests that a striking energy reduction can be achieved by enriching DVS policies with more detailed information on the required workload.

Recently, the discipline of probabilistic timing analysis has significantly advanced (Burns et al., 2003; Ermedahl et al., 2003), and today there exist tools that can provide the probability density function (PDF) of task's execution times (Bernat et al., 2003). Basically, these tools partition the task code into basic blocks, which are sequential instructions between two consecutive conditional branches. The duration of each block depends on the processor status (e.g., caches, pipeline stages, out-of-order execution, etc.) and it can be modelled by a random variable. The PDF of the whole task can be extracted by combining the information of every block. This information can then be exploited to reduce the energy consumption. Lorch and Smith (2001) proposed the processor acceleration to conserve energy (PACE) model that increases speed as the task progresses in continuous speed processors. However, the model is not general: it considers only a well-defined power function (i.e., energy per cycle proportional to the speed square) and it does not account for the overhead during frequency transition. Xu et al. (2004) extended the model to the case of discrete speeds and general power functions. Furthermore, they took into account idle power and speed change overhead. However, none of the above papers dealt with optimal speed transition instants, and it is not clear how an optimal sequence of transition points can be found. In the original paper, the authors proposed a heuristic to select a 'good' sequence of transition points, but only justified it under the condition of continuous speed (Lorch and Smith, 2001). In the second paper, the problem of optimal transition points is said to be 'still an open problem beyond the scope of the paper' (Xu et al., 2004). An attempt to consider stochastic information in energy reduction problems has been made also by Gruian (2002). However, that study addresses the case with no transition overheads and with a specific power function, as well.

In this paper, we integrate the concept of probabilistic execution time within the framework of energy minimisation, providing the basis of a new challenging approach. We show how probabilistic information about task execution times can be exploited to reduce the energy consumed by the processor without violating the timing constraints. Optimal speed assignments and transition points are found using a very general model that accounts for processor idle power and time/energy overheads due to frequency transitions. We also show how these results can be applied to some significant examples.

2 Energy management scheme

We focus on the problem of reducing the energy consumed by a task τ on a variable speed processor. Some existing energy-aware algorithms (Zhu and Mueller, 2004; Lorch and Smith, 2001; Xu et al., 2004) have been developed starting from this simple scheme, since it constitutes a good starting point for more complex analysis.

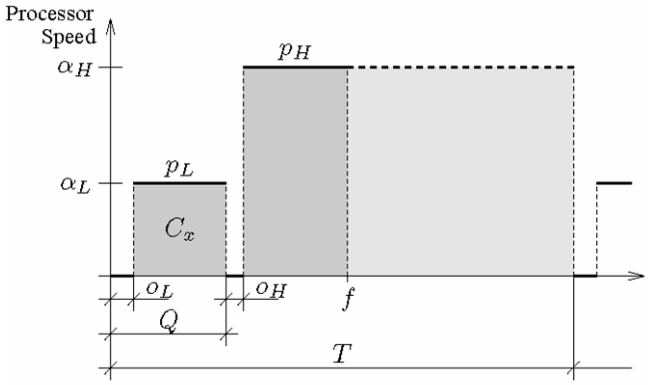
In our model, the task τ has a period and a deadline both equal to T. The number of processor cycles required by the task is modelled by a random variable whose PDF is $f_C(c)$. The maximum possible number of cycles needed by τ is $C_{\max}$. Since τ is a hard real-time task, $C_{\max}$ cycles must be available in $[0,T]$ in any case.

If the number of required cycles in $[0,T]$ is known in advance, it has been shown (Ishihara and Yasuura, 1998;

Pouwelse et al., 2001) that using a constant speed during task execution minimises the energy consumed assuming a continuous speed processor. In fact, the convexity of the power/speed curve implies that maintaining a constant speed α is better than switching between two different speeds. When the processor offers a limited set of speeds, using the two speeds which are the closest to the optimal (ideal) speed minimises the energy consumption (Ishihara and Yasuura, 1998).

In our scheme, it is not possible to compute the optimal constant speed α because the actual number of cycles required by the current instance of τ is not known in advance. In this case, a common technique (Aydin et al., 2004; Zhu and Mueller, 2004; Pillai and Shin, 2001; Lorch and Smith, 2001; Xu et al., 2004) is based upon the idea of deferring some work, expecting that the current instance of τ will request much less than its WCEC $C_{\max}$. This technique typically splits the task execution into two parts, as shown in Figure 1. In the first part, the processor runs at a lower speed α_L to reduce the energy consumed in the average case. In the second part, instead, the processor runs at a higher speed α_H in order to provide up to $C_{\max}$ cycles even in the worst case. The idea is that, if a task tends to use much less than its WCEC, the second part, which consumes more energy, may never be needed.

Figure 1 The energy management scheme



The idea of deferring work has been widely used in the literature to create several energy-aware algorithms. For instance, Pillai and Shin (2001) proposed the RTDVS-look ahead algorithm, which defers as much work as possible, setting the processor frequency to the minimum value which ensures that all future deadlines will be met. This technique has also been used by Aydin et al. (2004) in the 'aggressive' version of the DRA algorithm. This algorithm speculatively assumes that current and future instances of the task will most probably present a computational demand lower than the worst case. Hence, it tries to reduce the speed of the running task by deferring all the work above a certain threshold, set according to the average workload. A similar approach has been applied to EDF by Zhu and Mueller (2004): each task's instance is divided into two portions and the goal is to provide the average number of cycles C_{avg} within the first portion. All these techniques follow intuitive ideas [such as providing the average execution cycles in the first part (Zhu and Mueller, 2004)] to simplify the solution, and do not study analytically the problem of finding optimal values for speed assignments and transition instants.

2.1 Processor model

In CMOS circuits, the energy spent in dynamic switching dominates the energy consumed by leakage currents, and the dynamic portion of energy consumption is modelled by well-known polynomial formulas (Chandrakasan and Brodersen, 1995; Hong et al., 1998). However, as the integration technology advances, it is expected that the leakage will significantly affect, if not dominate, the overall energy consumption in integrated circuits (ICs) (for semiconductors; Rabaey et al., 2002; Gruian, 2002). Very recently, some work addressed the issue of scheduling a real-time application while reducing the leakage power as well (Quan et al., 2004). Also, an important fraction of the consumed energy depends on the memory. It has been shown (Pouwelse et al., 2001) that at low frequencies, the energy consumption is dominated by the memory, whereas at high frequencies it is dominated by the processor core.

All these remarks have led us to formulate a general model for the processor energy consumption. The processor is characterised by a set $\mathcal{M} = \{\Lambda_1, \Lambda_2, \Lambda_3, ...\}$ of operating modes.

Each mode $\Lambda_k = (\alpha_k, p_k, o_k, e_k)$ is described by four parameters:

- α_k is the processor speed in mode Λ_k and it is measured as number of cycles per second
- p_k is the power consumed in mode Λ_k when running at speed α_k, measured in watts
- o_k is the time overhead needed to enter mode Λ_k and it is expressed in seconds
- e_k is the energy overhead involved in the transition to the mode Λ_k, expressed in Joule.

For the sake of simplicity, we assume that o_k and e_k depend only on mode Λ_k and neither on the mode the processor was operating before, nor the processor status. Also, we consider only efficient speeds meaning that:

$$\forall \Lambda_i, \Lambda_j \quad \alpha_i \le \alpha_j \Rightarrow \frac{p_i}{\alpha_i} \le \frac{p_j}{\alpha_j}. \tag{1}$$

In fact, if this condition is not true for some Λ_i and Λ_j, then the mode Λ_j would be always more convenient than the mode Λ_i. DVS architectures may also have inefficient operating frequencies, that can be easily removed from the set of available frequencies (Saewong and Rajkumar, 2003; PARTS).

Notice that this model is very general, since it is suitable for both continuous and discrete speed processors. If the

processor can vary its speed continuously, then the set $\mathcal{M}$ is composed by infinite modes; on the other hand, if the processor has only discrete operating modes then the set $\mathcal{M}$ will be finite.

Finally, we suppose that the processor has one *idle operating mode* denoted by Λ_I. The processor enters idle mode Λ_I when all the computation required by the task is completed. When running in idle mode, the processor does not provide any useful computation (i.e., $\alpha_I = 0$). Notice that existing processors have several idle modes presenting different features. Taking into account only one idle mode Λ_I, however, constitutes a good starting point for considering more complex processors.

3 Optimal speed assignment

A speed assignment is optimal when it minimises the average energy consumed. Given the PDF of the task computation time $f_C(c)$, the expectation of the energy consumption E^{avg} can be computed as $\int_{-\infty}^{+\infty} E(c) f_C(c)\,\mathrm{d}c$, where $E(c)$ denotes the energy consumed when the task executes for c cycles. The optimal values of the parameters occur where the partial derivatives of E^{avg} are equal to zero.

3.1 Average energy consumption

We now compute the average energy consumption based on the probabilistic information of the task execution time. The optimal values for speed assignments and transition instants will be computed in Section 3.2 based on this result.

The energy consumed by the processor can be split into two separate components: the *active energy* E_A, consumed when executing the task τ, and the *idle energy* E_I, consumed when the task has terminated and the processor has entered mode Λ_I. Since these two terms must be added, they can be considered separately. First, let us compute the value of the active energy.

Let α_L and α_H be the lower and the higher processor speeds respectively. The period of the scheme is T. The number of processor cycles required by the task τ in each period is modelled by a random variable whose PDF is $f_C(c)$ and the maximum number of cycles is $C_{\max}$. This amount of cycles must be guaranteed in each period because the task is subject to a hard real-time constraint. Our goal is to find the optimal values for the two speed levels α_L and α_H and the instant Q when the transition should occur.

Let C_x be the number of cycles provided while running at α_L, as shown in Figure 1. Let c be the actual number of cycles required by the current instance of τ and f the finishing time of the task. We distinguish two different cases:

1 if $c \le C_x$, then the task terminates before the speed switch, and we expect $f \le Q$

2 otherwise, if $c > C_x$, then we need to run at speed α_H to provide the required cycles and we expect $f > Q + o_H$.

We consider the two cases separately.

In the first case $(c \le C_x)$, the finishing time is

$$f = o_L + \frac{c}{\alpha_L} \tag{2}$$

and the active energy consumed in one period T is

$$E_A = e_L + p_L\left(f - o_L\right) = e_L + \frac{p_L}{\alpha_L} c. \tag{3}$$

On the other hand, when $C_x < c \le C_{\max}$, we have

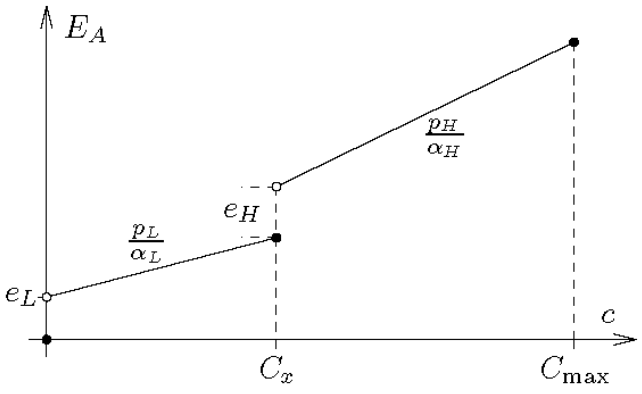
$$f = Q + o_H + \frac{c - C_x}{\alpha_H} \tag{4}$$

and the energy consumption is

$$E_A = e_L + \frac{p_L}{\alpha_L} C_x + e_H + \frac{p_H}{\alpha_H}\left(c - C_x\right). \tag{5}$$

The energy consumption as function of the number of cycles c is shown in Figure 2. Notice that due to the assumption of equation (1), the slope $\frac{p_H}{\alpha_H}$ is greater than $\frac{p_L}{\alpha_L}$.

Figure 2 The active energy E_A vs. c



Equations (3) and (5) provide the active energy E_A consumed when the number of cycles is c. Since the number of cycles is a random variable with PDF $f_C(c)$, then the energy consumed is a random variable as well. The expectation E_A^{avg} of the random variable E_A is

$$E_A^{\text{avg}} = \int_0^{C_x} E_A f_C(c)\,\mathrm{d}c + \int_{C_x}^{C_{\max}} E_A f_C(c)\,\mathrm{d}c$$
$$= e_L + e_H\left(1 - F_C(C_x)\right) + \frac{p_H}{\alpha_H} C_{\text{avg}}$$
$$- \left(\frac{p_H}{\alpha_H} - \frac{p_L}{\alpha_L}\right)\left(G_C(C_x) + C_x\left(1 - F_C(C_x)\right)\right)$$

where we set

$$F_C(x) = \int_0^x f_C(c)\,\mathrm{d}c \qquad G_C(x) = \int_0^x c\, f_C(c)\,\mathrm{d}c.$$

For compactness, if we also set

$$\gamma(x) = G_C(x) + x\left(1 - F_C(x)\right), \tag{6}$$

the average active energy E_A^{avg} consumed in a period can be written as

$$E_A^{\text{avg}} = e_L + e_H\left(1 - F_C(C_x)\right) + \frac{p_L}{\alpha L}\gamma(C_x) + \frac{p_H}{\alpha_H}\left(C_{\text{avg}} - \gamma(C_x)\right) \tag{7}$$

Notice that $G_C(C_{\max})$ is equal to C_{avg} by definition. For this reason, we always have $0 \le \gamma(x) \le C_{\text{avg}}$ for all x.

Accounting for the idle power

Equation (7) takes only into account the energy consumed when the processor is running the task. We now evaluate the contribution E_I to the energy consumed by the processor after the task has terminated. This contribution is

$$E_I = e_I + p_I\left(T - f - o_I\right)$$

where e_I and o_I are respectively the overheads of energy and time to enter the idle mode and f is the finishing time of the task.

From the previous equations (2) and (4), which established the relationship between the required number of cycles and the finishing time f we can express E_I as a function of the number of cycles c. Hence, we have

$$E_I = \begin{cases} e_I + p_I\left(T - o_L - \dfrac{c}{\alpha_L} - o_I\right) & \text{if } c \le C_x \\ e_I + p_I\left(T - o_L - \dfrac{C_x}{\alpha_L} - o_H - \dfrac{c - C_x}{\alpha_H} - o_I\right) & \text{if } c > C_x \end{cases} \tag{8}$$

As done for the active power consumption, we calculate similarly the average consumption of idle energy by integrating equation (8) over the PDF $f_C(c)$ of the number of cycles.

$$E_I^{\text{avg}} = e_I + p_I\left(T - o_L - o_I - \int_0^{C_x} \frac{c}{\alpha_L} f_C(c)\,\mathrm{d}c - \int_{C_x}^{C_{\max}} \left(\frac{C_x}{\alpha_L} + o_H + \frac{c - C_x}{\alpha_H}\right) f_C(c)\,\mathrm{d}c\right)$$
$$= e_I + p_I\left(T - o_L - o_I - o_H\left(1 - F_C(C_x)\right) - \frac{C_{\text{avg}}}{\alpha_H} - \left(\frac{1}{\alpha_L} - \frac{1}{\alpha_H}\right)\gamma(C_x)\right)$$

where we used the definition of $\gamma(x)$ given in equation (6). We can write it in a more compact form as

$$E_I^{\text{avg}} = e_I + p_I\left(T - o_L - o_I - o_H\left(1 - F_C(C_x)\right) - \frac{C_{\text{avg}} - \gamma(C_x)}{\alpha_H} - \frac{\gamma(C_x)}{\alpha_L}\right) \tag{9}$$

Finally, if we add equations (7) and (9), we find that

$$E_I^{\text{avg}} = e_L + e_I + p_I\left(T - o_L - o_I\right) + \left(e_H - p_I o_H\right)\left(1 - F_C(C_x)\right) + \frac{p_L - p_I}{\alpha_L}\gamma(C_x) + \frac{p_H - p_I}{\alpha_H}\left(C_{\text{avg}} - \gamma(C_x)\right) \tag{10}$$

Equation (10), which extends equation (7) to the case of idle power, is a new result in the literature. It expresses the average energy consumption as function of the probability density of the task execution cycles, while taking also into account the idle power and the overheads.

Now an essential remark is in order. Equation (10) can be obtained from equation (7) by the following substitution

$$\tilde{e}_L = e_L + e_I + p_I\left(T - o_L - o_I\right) \qquad \tilde{e}_H = e_H - p_I o_H$$
$$\tilde{p}_L = p_L - p_I \qquad \tilde{p}_H = p_H - p_I$$

Hence, we can say that the 'idle power can be taken into account by a simple adjustment of the operating modes'. For this reason, unless specified differently, in the rest of the paper, we will consider only equation (7).

3.2 Minimum energy consumption

Equation (7) is valid for processors with discrete as well as continuous operating modes. Taking into account discrete operating modes requires the evaluation of the energy consumed by all the mode pairs. The problem then becomes to find the optimal pair with the lowest total energy consumption. The complexity of this evaluation process is $O(m^2)$, where m is the number of available efficient operating modes.

For processors with continuous operating modes, instead, it is possible to analytically find the optimal values for speed assignments and transition instant. Many significant contributions in the literature (Aydin et al., 2004; Pillai and Shin, 2001; Zhu and Mueller, 2004; Scordino and Lipari, 2004) still assume a continuous speed because if the processor speed levels are very close to each other, then this

approximation is very close to reality. Obviously, if the optimal speed is not available, it has to be approximated with the closest discrete speed higher than the optimal one. In this case, there is an increase of energy consumption, called *energy quantisation error*, which was studied by Saewong and Rajkumar (2003).

In the continuous model, we assume that:

- all operating modes Λ_k require the same time overhead o and energy overhead e. Formally, we have that $\forall k \; e_k = e, \quad o_k = 0$
- the speed α varies within $[0, \alpha_{\max}]$, where $\alpha_{\max}$ is the maximum speed allowed by the processor
- the power consumption at speed α is modelled by the function $p(\alpha)$. Typically, the power function $p(\alpha)$ is a polynomial (Chandrakasan and Brodersen, 1995). However, as stated earlier, it is expected that the power/speed relationship may differ from the ideal polynomial function (Gruian, 2002). For this reason, we model this relationship by a generic function $p(\alpha)$.

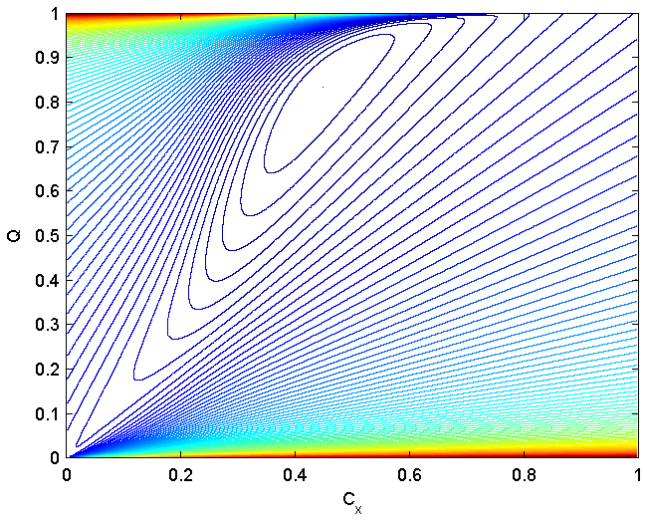
Different from the case of discrete operating modes, in the case of continuous operating modes, we can find the conditions for minimum energy consumption (i.e., optimal transition instant and speed levels) starting from equation (7).

The speeds (α_L, α_H) can be expressed as function of C_x and Q, as follows:

$$\alpha_L = \frac{C_x}{Q-o} \quad \alpha_H = \frac{C_{\max} - C_x}{T-Q-o}. \tag{11}$$

These equalities follow directly from the adopted energy management scheme as shown in Figure 1. In the remainder of the paper, we will develop the energy management scheme using C_x and Q as our free variables.

Figure 3 Level curves of E^{avg} for exponential PDF with $C_{\text{avg}} = 0.2929$ (see online version for colours)



It is very insightful to plot the quantity E^{avg} on a plane (C_x, Q). Figure 3 shows the level curves of the quantity E^{avg} as function of C_x and of Q. In the plot, we assumed an exponential PDF with average value $C_{\text{avg}} = 0.2929$, a period T equal to 1 and a power function $p(\alpha) = k\alpha^3$. The minimum occurs at the centre of the white region for a value of C_x greater than C_{avg}.

The minimum of equation (7) can be found analytically by calculating the partial derivatives of E^{avg} with respect to the variables C_x and Q. Let $p'_H = \frac{\mathrm{d}p}{\mathrm{d}\alpha}(\alpha_H)$ and $p'_L = \frac{\mathrm{d}p}{\mathrm{d}\alpha}(\alpha_L)$. We have:

$$\begin{aligned}\frac{\partial E^{\text{avg}}}{\partial C_x} &= -e f_C(C_x) - \left(p'_H - \frac{p_H}{\alpha_H}\right)\frac{C_{\text{avg}} - \gamma(C_x)}{C_{\max} - C_x} \\ &\quad - \frac{p_H}{\alpha_H}\gamma'(C_x) + \left(p'_L - \frac{p_L}{\alpha_L}\right)\frac{\gamma(C_x)}{C_x} + \frac{p_L}{\alpha_L}\gamma'(C_x)\end{aligned} \tag{12}$$

where we used the property that, from equation (11), it follows:

$$\alpha_L{}' = \frac{\partial \alpha_L}{\partial C_x} = \frac{1}{Q-o} = \frac{\alpha_L}{C_x} \quad \Rightarrow \frac{\alpha_L{}'}{\alpha_L} = \frac{1}{C_x}$$

$$\alpha_H{}' = \frac{\partial \alpha_H}{\partial C_x} = -\frac{\alpha_H}{C_{\max} - C_x} \quad \Rightarrow \frac{\alpha_H{}'}{\alpha_H} = -\frac{1}{C_{\max} - C_x}$$

An equivalent property can also be found for the differentiation with respect to Q. In fact, we have

$$\begin{aligned}\frac{\partial \alpha_L}{\partial Q} &= -\frac{C_x}{Q^2} = -\frac{\alpha_L{}^2}{C_x} \quad \Rightarrow \frac{\alpha_L{}'}{\alpha_L{}^2} = -\frac{1}{C_x} \\ \frac{\partial \alpha_H}{\partial Q} &= \frac{\alpha_H{}^2}{C_{\max} - C_x} \quad \Rightarrow \frac{\alpha_H{}'}{\alpha_H{}^2} = \frac{1}{C_{\max} - C_x}\end{aligned} \tag{13}$$

Now we complete the analysis of the function E^{avg} by computing $\frac{\partial E^{\text{avg}}}{\partial Q}$, which can be greatly simplified, thanks to the property of equation (13). Notice that, differently than in equation (12), in the next equation $\alpha_L{}'$ and $\alpha_H{}'$ denote respectively $\frac{\partial \alpha_L}{\partial Q}$ and $\frac{\partial \alpha_H}{\partial Q}$.

$$\begin{aligned}\frac{\partial E^{\text{avg}}}{\partial Q} &= (p'_H \alpha_H - p_H)\frac{C_{\text{avg}} - \gamma(C_x)}{C_{\max} - C_x} \\ &\quad - (p'_L \alpha_L - p_L)\frac{\gamma(C_x)}{C_x}\end{aligned} \tag{14}$$

Equations (12) and (14) are the components of the gradient ∇E^{avg}. From functional analysis, we know that the minimum satisfies the condition $\nabla E^{\text{avg}} = 0$. Once the

optimal (C_x, Q) is found, then the constraint $\alpha_H \le \alpha_{\max}$ must be checked. In fact, if it is violated, it means that the global minimum would result in a too high value of α_H. It this case, we know from the Kuhn-Tucker conditions that the minimum occurs when $\alpha_H = \alpha_{\max}$, which means that

$$\frac{C_{\max} - C_x}{T - Q - o} = \alpha_{\max} \Rightarrow \alpha_L = \frac{C_x \alpha_{\max}}{\alpha_{\max}(T - 2o) - C_{\max} + C_x} \quad (15)$$

From equation (7), replacing α_H with $\alpha_{\max}$ and α_L with the expression of equation (15), we find E^{avg} as function of the unique variable C_x. The minimal energy solution is found by applying classical techniques of functional analysis of one-variable functions.

4 Examples

After the main equations for the general case are found, we show how they can be applied to find the optimal (C_x, Q) in some common cases.

When considering continuous speed levels, a common assumption are that the relationship between the power consumption p and speed α is

$$p(\alpha) = k\alpha^n$$

for some k, n. The typical value of n is 3. However, we keep the general form as long as the math is tractable. In this hypothesis, the gradient can be simplified as follows:

$$\begin{cases} \dfrac{\partial E^{\text{avg}}}{\partial C_x} = -e f_C(C_x) - k\left(\left((n-1)\dfrac{C_{\text{avg}} - \gamma(C_x)}{C_{\max} - C_x} + \gamma'(C_x)\alpha_H{}^{n-1} - (n-1)\left(\dfrac{\gamma(C_x)}{C_x} + \gamma'(C_x)\right)\alpha_L{}^{n-1}\right)\right. \\ \dfrac{\partial E^{\text{avg}}}{\partial Q} = k(n-1)\left(\alpha_H{}^n \dfrac{C_{\text{avg}} - \gamma(C_x)}{C_{\max} - C_x} - \alpha_L{}^n \dfrac{\gamma(C_x)}{C_x}\right) \end{cases}$$

In order to find the conditions of minimum energy, we have to set both the gradient components equal to zero. The math is greatly simplified by assuming no overhead ($e = 0$ and $o = 0$). If we do so, by setting $\nabla E^{\text{avg}} = 0$, we finally find that the pair (C_x, Q) minimising the average energy E^{avg} must satisfy equation (16).

$$\begin{cases} \dfrac{\left((n-1)\gamma(C_x) + C_x\gamma'(C_x)\right)\left(\dfrac{C_{\max}}{C_x} - 1\right)\left(\dfrac{C_{\text{avg}}}{\gamma(C_x)} - 1\right)}{(n-1)\left(C_{\text{avg}} - \gamma(C_x)\right) + (C_{\max} - C_x)\gamma'(C_x)} = \dfrac{T}{Q} - 1 \\ \left(\dfrac{C_{\max}}{C_x} - 1\right)^{n-1}\left(\dfrac{C_{\text{avg}}}{\gamma(C_x)} - 1\right) = \left(\dfrac{T}{Q} - 1\right)^n \end{cases} \quad (16)$$

Due to lack of space, we do not include all the calculations. Because of their importance, we call equation (16) the *minimum stochastic energy equations*. Once we know n and the probability density $f_C(c)$, equation (16) can be solved to obtain the pair (C_x, Q) that minimises the energy consumption.

4.1 Uniform density

Let us now assume a uniform density between $C_{\min}$ and $C_{\max}$. This means that

$$f_C(c) = \begin{cases} \dfrac{1}{C_{\max} - C_{\min}} & \text{if } C_{\min} \le c \le C_{\max} \\ 0 & \text{otherwise} \end{cases}$$

and also, when $C_{\min} \le c \le C_{\max}$,

$$F_C(c) = \frac{c - C_{\min}}{C_{\max} - C_{\min}} \quad G_C(c) = \frac{c^2 - C_{\min}{}^2}{2(C_{\max} - C_{\min})}.$$

The function $\gamma(c)$, defined in equation (6) and its derivative are

$$\gamma(c) = \frac{-c^2 + 2cC_{\max} - C_{\min}{}^2}{2(C_{\max} - C_{\min})} \quad \gamma'(c) = \frac{C_{\max} - c}{C_{\max} - C_{\min}}$$

In this case, the minimum energy can be simply found by properly substituting $\gamma(C_x)$ and $\gamma'(C_x)$ in the minimum stochastic energy equation (16). To simplify and compact them, it is very convenient to normalise the cycles C_x and $C_{\min}$ with respect to $C_{\max}$. Hence, we set $x = \dfrac{C_x}{C_{\max}}$ and $a = \dfrac{C_{\min}}{C_{\max}}$.

When $n = 2$, after a long algebraic manipulation, we find that the minimum point is the solution of the polynomial

$$3x^2 - (2 + a^2)x^2 + a^2 x + a^4 = 0$$

Since the admissible solution x must be within $[a, 1]$, due to the fact that $C_{\min} \le C_x \le C_{\max}$, we find that the only admissible solution is

$$x_{\text{opt}} = \frac{1 + \sqrt{1 + 3a^2}}{3}. \quad (17)$$

In the same way, if we assume a more realistic power function with $n = 3$, the polynomial to be solved becomes

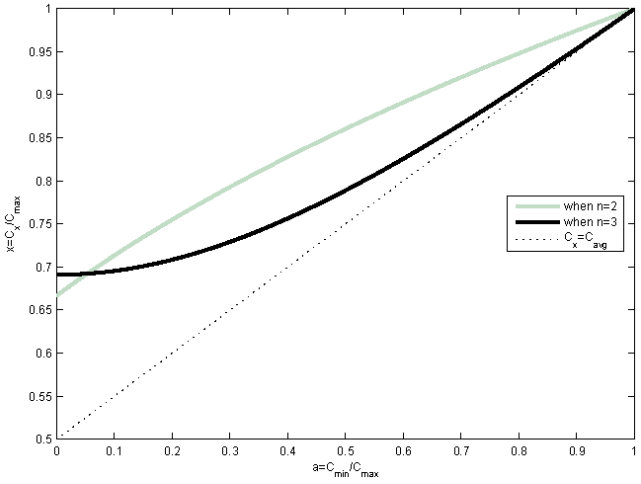
$$\begin{aligned} &4x^5 - (10 + 4a^2)x^4 + (5 + 12a^2)x^3 \\ &-(5a^2 + 2a^4)x^2 - a^4 x + a^6 = 0 \end{aligned} \quad (18)$$

whose only admissible solution in $[a,1]$ is

$$x_{\text{opt}} = \frac{5-\sqrt{5}+\sqrt{2}\sqrt{5\left(3-\sqrt{5}\right)-8\left(1-\sqrt{5}\right)a^2}}{8} \tag{19}$$

These equations show that our result is an improvement with respect to the sub-optimal solution proposed by Zhu and Mueller (2004) (i.e., setting C_x equal to C_{avg}). In fact, from both equations (17) and (19), we see that the 'optimal value is always greater than' C_{avg} (see also Figure 4). Providing C_{avg} cycles at speed α_L would increase the average energy consumed in the period.

Figure 4 The optimal number of cycles (see online version for colours)



4.2 Exponential density

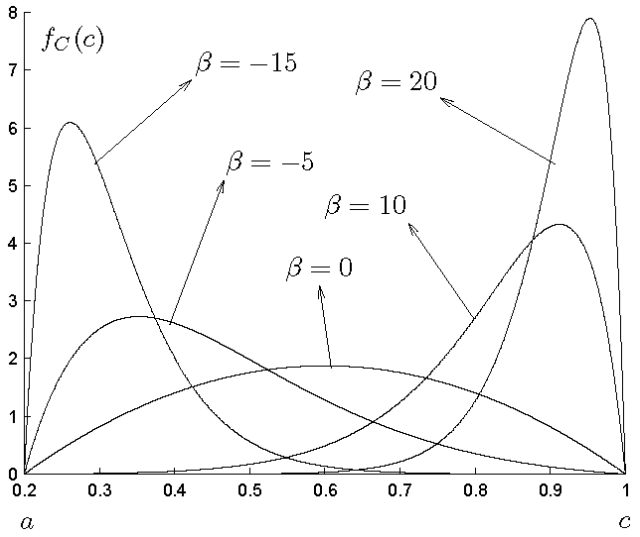
The probability density considered previously is very simple and it allows finding the exact value of the pair (C_x, Q) that minimises the average energy consumption. We consider now a more sophisticated density $f_C(c)$ that better captures the characteristics of real execution times. Without loss of generality, we normalise the number of cycles with respect to $C_{\max}$ so that the possible values of cycles are in $[0,1]$. As before, we set $a = \dfrac{C_{\min}}{C_{\max}}$.

We consider the following exponential PDF:

$$f_C(c) = \begin{cases} \dfrac{1}{K} e^{\beta c}(1-c)(c-a) & \text{if } c \in [a,1] \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

where K is a proper constant such that $\int_a^1 f_C(c)\mathrm{d}c = 1$. The presence of β allows to alter the symmetry of the density. In fact, for negative values of β the density shifts to the left, meaning that values closer to $C_{\min}$ are more likely to happen. On the other hand, positive values of β means that execution cycles closer to $C_{\max}$ occur more frequently. Figure 5 shows the shape of some possible functions.

Figure 5 Exponential PDFs



Unfortunately, when dealing with exponential densities, the minimal energy (C_x, Q) pair can only be found by numerical approximation. We investigated the effect of the PDF asymmetry onto the solution. The result is quite interesting. In Figure 6, we plot the ratios $\dfrac{C_x}{C_{\text{avg}}}$ and $\dfrac{Q}{T}$, assuming $a = \dfrac{C_{\min}}{C_{\max}} = 0.2$. A first result, also noticed for uniform density case, is that the optimal C_x is always greater than C_{avg}. This fact is evidenced by the black curve which is always above 1. We also highlight that for large positive values of β (meaning that values closer to $C_{\max}$ are more likely to occur), C_x approaches to C_{avg}.

Figure 6 The optimal (C_x, Q) pair as a function of the symmetry

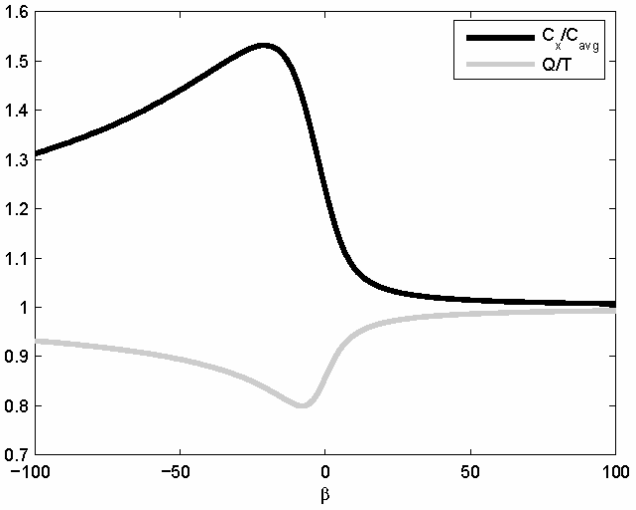
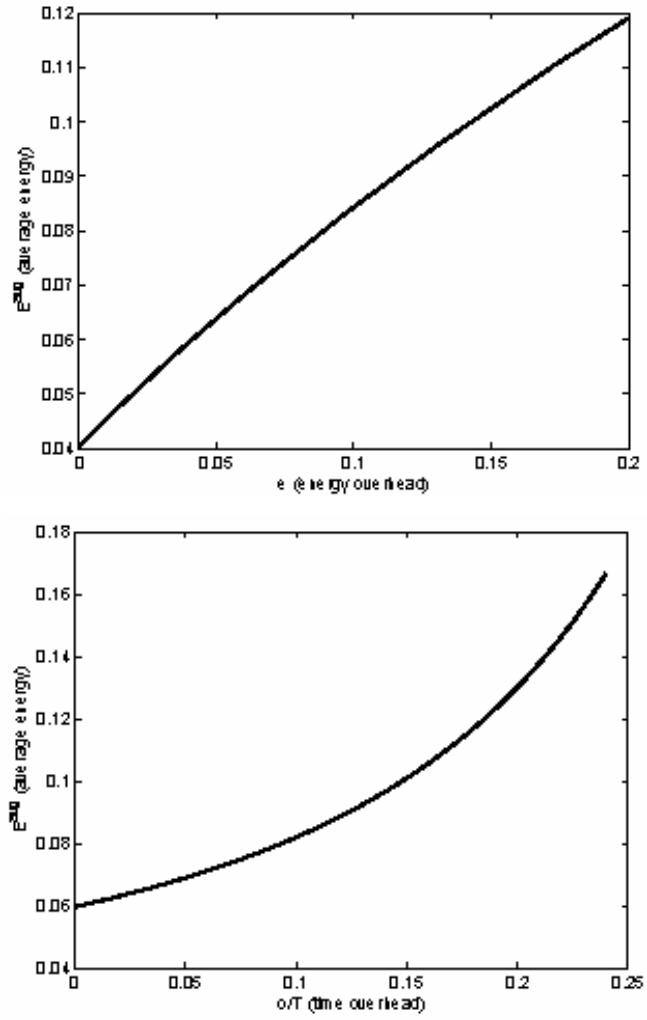


Figure 7 Impact of the overheads



4.3 Impact of the overheads

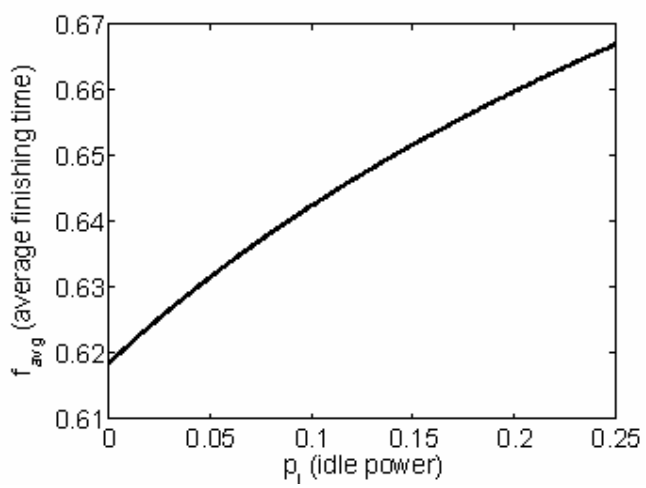
In this experiment, we evaluate the impact of energy overhead e and the time overhead o on the energy consumed. For this purpose, we fix the PDF $f_C(c)$ equal to the exponential density with $\beta = -50$ (please refer to Section 4.2), and the power function $p(\alpha) = \alpha^3$. Then, we vary the time overhead o and the energy overhead e and, for each value, we compute the average energy consumption E^{avg}. The results are shown in Figure 7.

As expected, increasing the overheads results in an increase of the energy consumed.

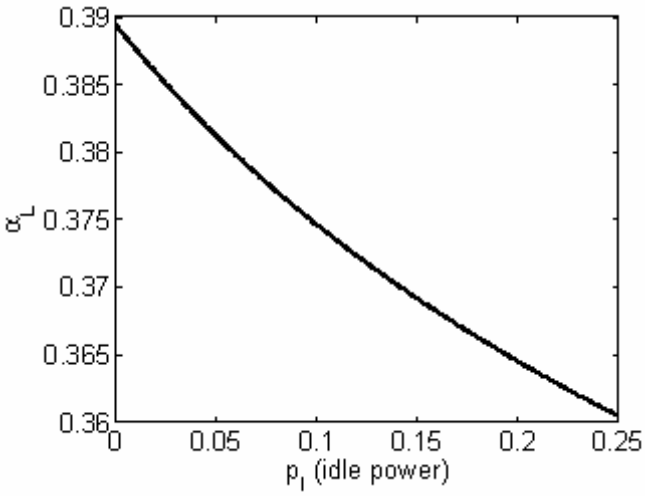
4.4 Idle power

As remarked at the end of Section 3.1, the energy consumed during the idle operating mode Λ_I can be taken into account by adjusting properly the parameters of the two modes Λ_L and Λ_H. In this section, we evaluate the impact of the power p_I consumed during the idle mode on the optimal values of the speed switch. In the experiments, the PDF is set equal to the exponential density and the power function is assumed to be cubic. In Figure 8, we plot the results.

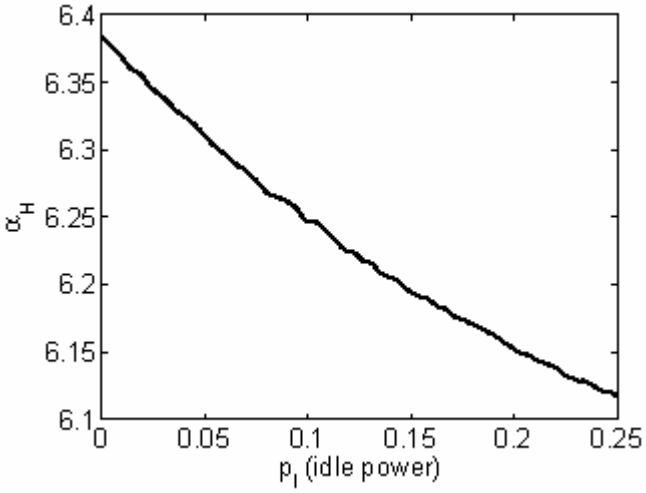
Figure 8 Impact of the idle power on, (a) the finishing time f_{avg} (b) the speed α_L (c) the speed α_H



(a)



(b)



(c)

The increase of the average finishing time f_{avg} shown in Figure 8(a) is justified by the fact that the processor consumes energy during the idle mode Λ_I as well. To avoid a waste of energy, it is preferable to stay in active mode for a longer time by deferring the average finishing time f_{avg}. Therefore, it is possible to lower the speeds α_L and α_H. This insight is confirmed by Figures 8(b) and 8(c).

5 Conclusions

Deferring the workload is an effective technique to reduce the energy consumed by the processor when the actual number of cycles is unknown.

We exploited this technique when proposing a general model for the processor that accounts for the idle power and for both the time and the energy overheads due to frequency transitions. We computed the optimal values for the speed assignments and the transition instant within our energy management scheme. We also studied how the overhead and the idle power affect the optimal values. Very interestingly, we showed that the power consumed during the idle operating mode can be easily taken into account by adjusting the active operating modes of the processor.

Finally, we showed how our results can be applied to some common cases (namely, the uniform and exponential densities).

As future work, we plan to implement the algorithm in the RTSim Simulator (http://www.rtsim.sf.net) to evaluate the improvement over the existing energy aware techniques.

References

Aydin, H., Melhem, R., Mossé, D. and Mejía-Alvarez, P. (2004) 'Power-aware scheduling for periodic real-time tasks', *IEEE Transactions on Computers*, Vol. 53, No. 5, pp.584–600.

Barroso, L.A., Dean, J. and Holzle, U. (2003) 'Web search for a planet: the Google cluster architecture', *IEEE Micro*, Vol. 23, No. 2, pp.22–28.

Bernat, G., Colin, A. and Petters, S.M. (2003) 'pWCET: a tool for probabilistic worst-case execution time analysis of real-time systems', YCS-2003-353, Department of Computer Science, University of York.

Bianchini, R. and Rajamony, R. (2004) 'Power and energy management for server systems', *Computer*, Vol. 37, No. 11, pp.68–74.

Bini, E., Buttazzo, G.C. and Lipari, G. (2005) 'Speed modulation in energy-aware real-time systems', *Proceedings of the 17th Euromicro Conference on Real-Time Systems*, Palma de Mallorca, Spain.

Bohrer, P., Elnozahy, E.N., Keller, T., Kistler, M., Lefurgy, C., McDowell, C. and Rajamony, R. (2002) *The Case for Power Management in Web Servers*, Kluwer Academic Publishers.

Burns, A., Bernat, G. and Broster, I. (2003) 'A probabilistic framework for schedulability analysis', *Proceedings of the EMSOFT*, Philadelphia, PA, USA, pp.1–15.

Chandrakasan, A.P. and Brodersen, R.W. (1995) *Low Power Digital CMOS Design*, Kluwer Academic Publishers, ISBN: 0-7923-9576-X.

Crusoe™ Processor Model TM5800 Version 2.1 Data Book Revision 2.01 (2003) Transmeta Corp., available at http://www.transmeta.com.

Ermedahl, A., Stappert, F. and Engblom, J. (2003) 'Clustered calculation of worst-case execution times', *Proceedings of the 6th International Conference on Compilers, Architectures and Synthesis for Embedded Systems*, San José, CA.

Gruian, F. (2002) 'Energy-centric scheduling for real-time systems', PhD thesis, Department of Computer Science, Lund Institute of Technology, Lund, Sweden.

Hong, I., Qu, G., Potkonjak, M. and Srivastava, M.B. (1998) 'Synthesis techniques for low-power hard real-time systems on variable voltage processors', *Proceedings of the 19th IEEE Real-Time Systems Symposium*, Madrid, Spain, pp.178–187.

Intel Centrino Mobile Technology, Intel Corp., available at http://developer.intel.com/design/mobile/centrinoplatformoverview.htm.

Intel Corp. (2004a) *Enhanced Intel SpeedStep Technology for the Intel Pentium M Processor*.

Intel Corp. (2004b) *Intel PXA27x Processor Family Power Requirements*.

Ishihara, T. and Yasuura, H. (1998) 'Voltage scheduling problem for dynamically variable voltage processors', *Proceedings of the International Symposium on Low Power Electronics and Design*, Monterey, CA, USA, pp.197–202.

Lee, S. and Sakurai, T. (2000) 'Run-time voltage hopping for low-power real-time systems', *Proceedings of the 37th Design Automation Conference*, Los Angeles, CA, USA, pp.806–809.

Lefurgy, C., Rajamani, K., Rawson, F., Felter, W., Kistler, M. and Keller, T.W. (2003) 'Energy management for commercial servers', *IEEE Computer*, Vol. 36, No. 12, pp.39–48.

Lorch, J.R. and Smith, A.J. (2001) 'Improving dynamic voltage scaling algorithms with pace', *ACM SIGMETRICS*, Cambridge, MA, USA, pp.50–61.

Mochocki, B., Hu, X.S. and Quan, G. (2002) 'A realistic variable voltage scheduling model for real-time applications', *Proceedings of the International Conference on Computer Aided Design*, San José, CA, USA, pp.726–731.

MPC5200: 32 Bit Embedded Processor, Motorola, available at http://e-www.motorola.com/webapp/sps/library/prod_lib.jsp.

PARTS, Power efficiency test, available at http://www.cs.pitt.edu/PARTS/demos/efficient.

Pillai, P. and Shin, K.G. (2001) 'Real-time dynamic voltage scaling for low-power embedded operating systems', *Proceedings of the 18th ACM Symposium on Operating System Principles*, Banff, Canada, pp.89–102.

Pouwelse, J., Langendoen, K. and Sips, H. (2001) 'Dynamic voltage scaling on a low-power microprocessor', *Proceedings of the 7th ACM International Conference on Mobile Computing and Networking*, Rome, Italy, pp.251–259.

Qadi, A., Goddard, S. and Farritor, S. (2003) 'A dynamic voltage scaling algorithm for sporadic tasks', *Proceedings of the 24th Real-Time Systems Symposium*, Cancun, Mexico, pp.52–62.

Quan, G., Niu, L., Hu, X.S. and Mochocki, B. (2004) 'Fixed priority based real-time scheduling for reducing energy on variable voltage processors', *Proceedings of the 25th IEEE Real-Time Systems Symposium*, Lisbon, Portugal, pp.309–318.

Rabaey, J.M., Chandrakasan, A. and Nikolic, B. (2002) *Digital Integrated Circuits*, Prentice Hall, 2nd ed., ISBN: 0-13-090996-3.

Rusu, C., Ferreira, A., Scordino, C., Watson, A., Melhem, R. and Mossé, D. (2006) 'Energy-efficient real-time heterogeneous server clusters', *Proceedings of the 12th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'06)*, San José, CA, USA.

Saewong, S. and Rajkumar, R. (2003) 'Practical voltage-scaling for fixed-priority RT-systems', *Proceedings of the 9th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'03)*, Washington, DC, USA, pp.106–114.

Scordino, C. and Bini, E. (2005) 'Optimal speed assignment for probabilistic execution times', *2nd Workshop on Power-Aware Real-Time Computing*, Jersey City, NJ, USA.

Scordino, C. and Lipari, G. (2004) 'Using resource reservation techniques for power-aware scheduling', *Proceedings of the 4th ACM International Conference on Embedded Software*, Pisa, Italy, pp.16–25.

Semiconductors, ITR International Sematech, available at http://public.itrs.net/.

Wegener, J. and Mueller, F. (2001) 'A comparison of static analysis and evolutionary testing for the verification of timing constraints', *Real-Time Systems*, Vol. 21, pp.241–268.

Xu, R., Xi, C., Melhem, R. and Mossé, D. (2004) 'Practical PACE for embedded systems', *4th ACM International Conference on Embedded Software (EMSOFT '04)*, Pisa, Italy.

Xu, R., Zhu, D., Rusu, C., Melhem, R. and Mossé, D. (2005) 'Energy-efficient policies for embedded clusters', *ACM SIGPLAN/SIGBED Conference on Languages, Compilers and Tools for Embedded Systems (LCTES'05)*, Chicago, IL, USA.

Zhu, Y. and Mueller, F. (2004) 'Feedback EDF scheduling exploiting dynamic voltage scaling', *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium*, Toronto, Canada, pp.84–93.

Nomenclature

Symbol	*Explanation*
Λ_k	the kth processor operating mode
α_k	speed when running in Λ_k mode
p_k	power consumption when running in Λ_k mode
o_k	the time overhead to enter the Λ_k mode
e_k	the energy overhead to enter the Λ_k mode
Λ_L	the low speed operating mode
Λ_H	the high speed op. mode, entered after the speed switch
Λ_I	the idle op. mode, entered when the task has finished
T	the task period and deadline
$C_{\max}$	the worst-case execution cycles
C_{avg}	the average execution cycles
C_x	cycles provided in mode Λ_L
Q	instant when switching from Λ_L to Λ_H
$f_C(c)$	the PDF of the execution cycles
$F_C(c)$	the cumulative density function (CDF) of the execution cycles

Nomenclature (continued)

Symbol	*Explanation*
$G_C(c)$	$= \int_0^c x f_C(x)\mathrm{d}x$. A property of $G_C(c)$ is that $G_C(C_{\max}) = C_{\text{avg}}$
$\gamma(x)$	$= G_C(x) + x\left(1 - F_C(x)\right)$
$E_A(c)$	active energy consumed in modes Λ_L, Λ_H, when executing c cycles
$E_{I(c)}$	idle energy consumed in mode Λ_I, when executing c cycles
E_A^{avg}	average active energy
E_I^{avg}	average idle energy
E^{avg}	average total (active + idle) energy