

Power Aware Scheduling for Real-Time Systems with (m, k) -Guarantee

Gang Quan, Linwei Niu, James P. Davis
Department of CSE
University of South Carolina
Columbia, SC 29208
{gquan, niul, jimdavis@cse.sc.edu}

Abstract

Energy consumption and Quality of Service (QoS) are two of the primary concerns in design of today's pervasive computing systems. In this paper, we study the problem of minimizing the energy consumption while ensuring the designated QoS requirements for a real-time system on a two-level variable voltage processor. In our approach, the real-time tasks are scheduled according to the earliest deadline first scheme, and the QoS requirements are quantified with so-called (m, k) -constraints, which require that at least m number of any k consecutive jobs of a task meet their deadlines. In this paper, we develop a static scheduling strategy to guarantee the (m, k) -constraints while optimizing the energy. Extensive experiments, both on synthesized systems and real applications, have been conducted to demonstrate the effectiveness of our approach.

1 Introduction

Power aware computing has come to be recognized as a critical enabling technology in the design of real-time embedded systems for use in the pervasive computing applications. Today's embedded systems continue to evolve towards higher levels of performance requirements to provide rich features for a variety of dedicated and extensible applications. Such high performance requirements are inevitably accompanied by the high level of energy consumption required to run the pervasive computing platform. On the other hand, to extend the battery life and thus the mission cycles for systems, a low power/energy consumption is highly desirable. If a system is to be both energy efficient and has high performance available on-demand, it has to be a *power aware* system—a system that is scalable, adaptive, and energy efficient where the system performance can be guaranteed and the energy consumption can be optimized.

In recent years, there has been increasing interest in applying scheduling techniques to conserve energy in real-

time applications. Many dynamic voltage scaling (DVS) techniques, e.g., [2, 7, 17, 25], have been proposed which take advantage of the power-manageable capability of the modern processors to reduce the energy consumption. They differ from the implementation strategies and real-time system characteristics, such as static vs. dynamic, fixed priority vs. dynamic priority assignment, preemptive vs. nonpreemptive effects. One common feature of these techniques is that they assume that all the task instances must meet their deadlines. While significant energy can be saved using these techniques, they become inefficient or inadequate when quality of service (QoS) requirements are imposed for the applications.

QoS requirements dictate under what conditions the system will provide its service the real-time tasks executed on the embedded processor. These requirements can be captured by different QoS metrics such as response delay, sample loss rate, among others. It is not difficult to see that the QoS requirements are closely related to the energy consumption. It is our belief QoS requirement and power/energy consumption must be studied cooperatively in real-time scheduling in order to provide a reliable, robust, and energy efficient solution for the real-time applications.

Recently, we have seen some research on using the DVS technique for real-time systems with QoS constraints. For example, Weiser et. al. [23] presented several interval-based scheduling techniques in which the processor speed in each interval is set dynamically according to the required QoS levels and the prediction of the processor usage in that interval. This prediction-and-speed-setting framework has been adopted in the later work, e.g. [4, 16, 11, 20]. Aydin et al. [2] and Rusu et. al. [21] treated this problem as a reward-based scheduling problem, and the best scheduling strategy is the one that can maximize the reward—in terms of energy saving and QoS levels—from executing the tasks. To improve the overall performance of the system while conserving energy, Ma et al. [13] used the criticality to prioritize the scheduling of more critical tasks over non-critical tasks in the systems. These best-effort techniques are use-

ful for applications such as real-time database and internet server, where there is no much known knowledge about the system characteristics and the QoS requirements are weakly defined. However, they cannot be applied for systems, such as real-time control and teleconference, in which the QoS has to be ensured to satisfy the users' requirements.

In response to this, some other research strives to enforce the QoS guarantee when developing the DVS techniques. For example, Qiu *et al.* presented a technique [18] to statistically guarantee the QoS for a real-time embedded system. They used the *generalized stochastic petri nets* to capture the probabilistic nature of the real-time embedded system, which is later transformed to a *Markov decision* model, based on which a linear programming formulation is constructed with the energy consumption as the objective and QoS requirements (such as delay, jitter, and loss rate) as the constraints. Yuan *et al.* [27] proposed to incorporate the stochastic analysis into DVS for soft deadline system to provide the statistic QoS guarantee. They also presented several resource conservation techniques [26, 28] when more deterministic information about the applications is available. With the QoS requirements formulated as the overall deadline missing rate that is tolerable by the users, Hua *et al.* [6] introduced several techniques to exploit the related processor slack time to save energy.

These approaches ensure the quality of service in a probabilistic manner which can be problematic for some real-time applications. For example, many real-time applications such as the navigation control in the unmanned aircraft can tolerate occasional deadline misses of the real-time tasks, and the information carried by these tasks can be estimated using interpolation techniques as long as the interval between deadline misses is large enough. Unfortunately, even a very low overall miss rate cannot rule out the possibility that a large number of deadline misses occurred in such a short period of time that the data cannot be successfully restored. This may cause the loss of important data or even the wrong operation of the system.

For the sake of designing highly reliable, robust, and yet energy efficient embedded system, we believe that it is necessary not only to guarantee the overall the QoS statistically, but also in such a way that a lower bounded and predictable QoS can be ensured at any specified time interval. Therefore, in this paper, we adopt the so called (m, k) -model [5] in which an (m, k) ($0 < m \leq k$) constraint is associated with each task requiring requiring that m out of any k consecutive job instances of the task must meet their deadlines.

Since not all executing jobs are required to meet specific deadlines according to the (m, k) -model, some "redundant" jobs can thus be discarded to save on the energy expenditure. In this paper, the "redundant" jobs are discarded according to techniques discussed in the literature [19], a technique originally applied to the problem of dealing with

cases of task overloading in real-time systems. We argue that this techniques can well serve our dual goals of improving the energy-saving performance, in addition to providing a guarantee of QoS formulated as (m, k) -constraints. For the remaining jobs, we propose a technique that is necessary and sufficient for their schedulability. Based on this capability, some of the remaining jobs can be executed at a low voltage level to save energy as long as they can finish before the deadlines. Moreover, this technique can be readily incorporated with the known dynamic scheduling techniques, such as [17, 2, 8] to further improve the energy efficiency. The significance of our approach is that, to the best of our knowledge, this is the first approach that can efficiently save on energy consumption while also providing a *deterministic* QoS guarantee for a real-time system.

This paper is organized as follows. Section 2 formally introduces our problem. Section 3 present several of our observations regarding to the identification of the "redundant" jobs. Section 4 presents a necessary and sufficient feasibility condition, with which a static DVS strategy is proposed in section 5. The effectiveness and energy efficiency of our approach are demonstrated using simulation results in Section 6. Section 7 concludes the paper.

2 System models

The real-time system considered in this paper contains n independent periodic tasks, $\mathcal{T} = \{\tau_0, \tau_1, \dots, \tau_{n-1}\}$, scheduled by an earliest deadline first (EDF) policy [10]. Each task contains an infinite sequence of periodically arrived instances with each of which called a *job*. Task τ_i is characterized using five parameters, i.e., $(T_i, D_i, C_i, m_i, k_i)$. T_i , D_i , and C_i represent the period, the deadline for each job instance, and the worst case execution time for τ_i , respectively. A pair of integers, i.e., (m_i, k_i) ($0 < m_i \leq k_i$), represent the QoS requirement for τ_i , requiring that, among any consecutive k_i jobs of τ_i , at least m_i must meet their prescribed completion deadlines.

The variable voltage processors used in our approach can run in one of two modes: *high voltage mode* and *low voltage mode*. In high voltage mode, the processor requires a high supply voltage (V_H), running at a faster clock rate. In low voltage mode, the voltage (V_L) supplied to the processor circuitry is lower and, thus, the speed at which the processor operates is slower. Commercial processors such as the ARM7D [12] and Motorola PowerPC 860 [14] have this type of operating characteristic relative to voltage scaling. We assume that C_i is the worst case execution time for task τ_i in high voltage mode. Therefore, if $\alpha = \frac{V_H}{V_L}$, the worst case execution time for τ_i is αC_i if τ_i is executed in low voltage mode.

Given the above abstraction and set of assumptions, we formulate our model as follows:

Problem 1 Given system $\mathcal{T} = \{\tau_0, \tau_1, \dots, \tau_{n-1}\}$, $\tau_i = (T_i, D_i, C_i, m_i, k_i)$, $i = 0, \dots, (n-1)$, schedule \mathcal{T} with EDF on a two-mode variable voltage processor such that all (m, k) -constraints are guaranteed and the energy consumption is minimized.

Since not all jobs scheduled are required to meet their deadlines due to (m, k) -constraints, there are opportunities to save energy by running as many jobs as possible at low voltage, such that just enough jobs meet their deadlines. The question, however, is how to minimize energy consumption while also providing a guarantee that no less than m_i of any k_i consecutive jobs from task τ_i meet their deadlines. We explore the theory and formulate an answer to this question in the following sections.

3 Meeting the (m, k) -constraints

There is much work, such as [3, 5, 24], that has investigated using the best-effort strategy to deal with the (m, k) -constraints. A common feature of these approach is that they dynamically promote the execution of a task to a higher priority if the task is proximally “closer” to violating the (m, k) -constraints. While these approaches help to improve the opportunity for tasks to meet their (m, k) constraints, they cannot ensure the (m, k) -constraints for the system be met. Also, precious energy resource is consumed to execute the jobs that miss their deadlines which is of little benefit to the overall performance of the embedded systems.

Another approach is to statically partition the jobs to be either mandatory or optional, and the (m, k) -constraints are ensured as long as all the mandatory jobs meet their deadline. Ramanathan [19] proposes such an approach which is of particular interest to us. Specifically, the job τ_{ij} , *i.e.* the j th job of task τ_i , is determined to be mandatory if

$$j = \lfloor \lfloor j \times \frac{m_i}{k_i} \rfloor \times \frac{k_i}{m_i} \rfloor, \quad j = 0, 1, 2, \dots, \quad (1)$$

and it is optional otherwise. For example, let τ_i have (m, k) constraint as $(3, 7)$. Then τ_{i2} is mandatory since $2 = \lfloor \lfloor 2 \times \frac{3}{7} \rfloor \times \frac{7}{3} \rfloor$, and τ_{i3} is optional since $3 \neq \lfloor \lfloor 3 \times \frac{3}{7} \rfloor \times \frac{7}{3} \rfloor$.

While this strategy is initially proposed for improving the stability performance under overloading situation for real-time control, several of our observations show it has great potential to save energy while guaranteeing the (m, k) -constraints. We summarize our observations in the following lemma (The proofs of the lemmas and theorems in this paper are omitted due to the page limit).

Lemma 1 Let the mandatory jobs for task τ_i with (m, k) constraint (m_i, k_i) be determined by equation (1). Then (i) for any k_i consecutive jobs of τ_i , there are exactly m_i mandatory jobs; (ii) within any p_i ($p_i > 0$) consecutive jobs

of τ_i , the difference of the numbers of mandatory jobs is no more than 1.

Lemma 1 implies that, using the strategy as shown in (1), a minimal set of mandatory jobs are determined, no more and no less. Removing any job from this set will violate the (m, k) -constraints, and any other jobs added to this set will be redundant in terms of (m, k) -guarantee. Therefore, the energy is only consumed for the necessary jobs, not for jobs that are nonessential for the (m, k) -constraints. Moreover, according to Lemma 1, formula (1) helps to spread out the required jobs *evenly* in each task along the time to avoid the “burst” workload for the processor and thus have great energy-saving potential, since the power consumption is a convex function of the processor speed, and dramatic variation of the processor speed tends to increase the energy consumption sharply.

With the mandatory job sets determined according to equation (1), the solution for Problem 1 then becomes how to schedule these jobs on a two-mode processor to save energy. Recall that any mandatory job missing its deadline will cause the violation of (m, k) -constraint. Therefore, to ensure the (m, k) -constraints, an accurate schedulability analysis is crucial since inaccurate prediction may either cause the violation of the system requirements, or lead to a pessimistic design which can seriously degrade the energy performance.

4 Schedulability analysis for the mandatory job set

In this section, we investigate the schedulability condition for the mandatory job set determined with equation (1). There are two benefits to conduct the schedulability analysis. First, it helps to predict if a real-time system with (m, k) -constraints using the static partitioning technique as shown in equation (1) is schedulable or not. Second, it also helps to put the execution of some mandatory jobs in the low voltage mode to save the energy while meeting their deadlines, and hence, the (m, k) -constraints.

Since the mandatory jobs are scheduled according to EDF, it is desirable to use the well-known Liu&Layland bound ($U \leq 1$) [10] to predict the schedulability. However, $U \leq 1$ is only the necessary and sufficient condition for task set with tasks’ deadlines equal their periods which is not the case for these mandatory jobs. An alternative way is to apply a more general EDF-schedulability condition, proposed by Zheng *et. al.* [29] and Liebeherr *et. al.* [9], as shown in Theorem 1.

Theorem 1 Real-time system \mathcal{T} is EDF-schedulable iff

$$\forall t > 0, \sum_i W_i(0, t) \leq t, \quad (2)$$

where $W_i(0, t)$ is the total workload from the jobs of τ_i that arrive before t and must finish by t .

The close-form necessary and sufficient condition presented in Theorem 1 can be used predict the EDF-schedulability for task system with arbitrary job arrivals and deadlines. Unfortunately it cannot be readily applied to check the feasibility for a mandatory job set, since it requires checking an infinite number of points ($t > 0$) to guarantee the schedulability. It is highly desirable that the feasibility of the system can be predicted by checking with a limited and small number of points. In what follows, we introduce another necessary and sufficient condition to precisely predict the schedulability requiring only limited number of computations. Before we present the new condition, we first introduce the following definition.

Definition 1 Let $w(t)$ represent the workload from the jobs that arrives in $[0, t]$ but not finished before t . A busy interval, i.e. $[t_s, t_e]$, is the interval such that $w(t_s^-) = w(t_e^+) = 0$ and $w(t) > 0$ for $t_s \leq t < t_e$.

The following theorem allows one to predict the schedulability for a mandatory job set by checking only a limited number of points.

Theorem 2 Let system $\mathcal{T} = \{\tau_0, \tau_1, \dots, \tau_{n-1}\}$, where $\tau_i = \{T_i, D_i, C_i, m_i, k_i\}$, and \mathcal{R} be the mandatory job set generated from \mathcal{T} according to equation (1). \mathcal{R} is schedulable with EDF iff all the mandatory jobs within the first busy interval can meet their deadlines.

The ending point of the first busy interval can be easily found by observing that it must be the smallest t such that the accumulated mandatory workload within interval $[0, t]$ equals t , that is,

$$\sum_i W_i(t) = \sum_i \left\lceil \frac{m_i}{k_i} \left\lceil \frac{t}{T_i} \right\rceil \right\rceil C_i = t. \quad (3)$$

One can easily compute the ending point for the first busy interval by using fixed-point iteration on formula (3) with the initial value t set to 0^+ . The fixed-point iteration will converge rapidly as long as the task set is schedulable. When the system is overloaded, the busy period can be very long. However, we can always set the upper bound of the busy interval length to be the least common multiple of the tasks' periods. Since if the busy interval length is longer than the least common multiple of the task periods, it implies that not all the mandatory workload within the first least common multiple of the task periods can finish in time. Thus, the task set cannot be schedulable.

5 DVS scheduling for the mandatory job set

After the mandatory job set is determined to be schedulable according to Theorem 2, one can take the advantage of

the variable voltage feature of the processor and assign as many mandatory jobs as possible to the low voltage mode as long as all the mandatory jobs are schedulable. However, care must be taken to assign a job with the low voltage mode, since a job may miss its deadline or cause other jobs to miss their deadlines when executed at a lower speed. In what follows, based on Theorem 2, we propose an algorithm (as shown in Algorithm 1) to find the schedule that can satisfied the (m, k) -constraints while minimizing the energy consumption.

Algorithm 1 A static algorithm that can guarantee the (m, k) -constraints while minimizing the energy

```

1: Input: The period task set  $\mathcal{T}$  and the two voltage values  $V_H$  and  $V_L$ 
2: Output: The minimal energy consumption ( $E_{min}$ ) and the voltage mode assignment  $Q_{min}$ .
3: Order the tasks according to their non-increasing values of  $\frac{m_i C_i}{k_i T_i}$ ,  $i = 0, \dots, n - 1$ ;
4:  $\alpha = \frac{V_H}{V_L}$ ,  $E_{min} = -1$ ,  $Q = \emptyset$ ;
5:  $U =$  all the tasks with the high voltage mode;
6:  $Q_{min} = U$ ;
7:  $(E_{min}, Q_{min}) = \text{Search}(\alpha, Q, U, Q_{min}, E_{min})$ 
8: Return  $E_{min}$  and  $Q_{min}$ ;
9: FUNCTION Search( $\alpha, Q, U, Q_{min}, E_{min}$ )
10: while  $U \neq \emptyset$  do
11:    $\tau =$  the head of  $U$ ;
12:   Set the low voltage mode for  $\tau$  and insert it into  $Q$ ;
13:   Check the feasibility of the task set  $Q \cup U$  according to Theorem 2;
14:   if schedulable then
15:     Compute the energy consumption  $E$ ;
16:     if  $E < E_{min}$  then
17:        $E_{min} = E$ ,  $Q_{min} = Q \cup U$ ;
18:     end if
19:      $(E_{min}, Q_{min}) = \text{Search}(\alpha, Q, U, E_{min}, Q_{min})$ ;
20:   else
21:     Set the voltage mode for  $\tau$  to be high;
22:      $(E_{min}, Q_{min}) = \text{Search}(\alpha, Q, U, E_{min}, Q_{min})$ ;
23:   end if
24: end while

```

Algorithm 1 applies the branch-and-bound strategy that is commonly used to solve the optimization problem. To improve its efficiency, the task set is initially ordered according to the values of $\frac{m_i C_i}{k_i T_i}$, $i = 0, \dots, n - 1$ (line 1), since the higher the value, the higher the possibility that the task set becomes unschedulable when the execution of the corresponding task is slowed down. Note that, this strategy used in Algorithm 1 in principle can be applied for the processor with multiple level of voltages. However, since the branch-and-bound is essentially an exhaustive search strategy, the computation cost will increase exponentially with

the number of voltage levels.

After the voltage levels are statically determined for each task by Algorithm 1, we can further improve the energy performance by incorporating the dynamic techniques such as the ones in [2, 8]. In next section, we use experiments to evaluate the energy performance of our approach.

6 Experimental Results

In this section, we use experiments to demonstrate the effectiveness of our approach to schedule the real-time systems while guaranteeing the QoS requirements in terms of (m, k) -constraints. As no other previous work, from our best knowledge, can minimize the energy savings while and guarantee the (m, k) -constraints, we compare our approach with two simple and intuitive approaches that may achieve the same goal. One approach uses the shut-down-when-idle strategy, and the other one uses the dynamic slack reclaiming technique [2, 8], which is also incorporated in our approach for a fair comparison. For brevity, we use MK_{LP} , MK_{SD} , and MK_{DYN} to represent our approach, the one with shut-down-when-idle, and the one using dynamic slack reclaiming technique, respectively.

Our experiments contain two sets of test cases. The first set contains randomly generated real-time task sets and uses an ideal two-mode variable processor. For the second set of the tests, we use the practical applications reported in the literature and adopt the hardware specifications of a real two-mode processor, *i.e.* Motorola PowerPC 860 [14].

For the randomly generated task sets, the periods are randomly selected from interval 1000,5000 and the deadlines are assumed to be equal to their periods. The worst case execution time (WCET) of a task at the high voltage mode is uniformly distributed from 1 to its deadline, and the actual execution time of a job is randomly picked from $0.5WCET$, WCET. The m_i and k_i for the (m, k) -constraints are also randomly generated such that k_i is uniformly distributed between 2 to 12, and $m_i < k_i$. In our experiment, we investigate the energy saving performance by different approaches under different task utilizations, since different task utilizations can result in significant differences for energy savings. Specifically, we assign the task sets into 10 groups according to their (m, k) utilization, *i.e.* $U_m = \sum_i \frac{m_i C_i}{k_i T_i}$. To reduce the statistical errors, we require that each group contain at least 20 schedulable task sets, or until at least 5000 task sets with the corresponding (m, k) utilization have been generated. For the processor used for the randomly generated task sets, we assume it is a two-mode variable voltage processor with $\alpha = 2$.

We also test our conclusions in a more practical environment. In the second set of experiments, we use three real-world applications: web phone [22], CNC (Computerized Numerical Control) machine controller [15], and INS (In-

ertial Navigation System) [1], as our task systems. For the (m, k) -constraints, we still resort to the extensive randomly generated samples as we do for the randomly generated task sets to provide us with some meaningful insights. The processor used in this set of experiments is adopted from the product specifications of a practical two-mode variable voltage processor, *i.e.* Motorola PowerPC 860. Specifically, it can be run at 50Mhz on 3.3 V, or 25Mhz at 2.4V. 20 sets of (m, k) -constraints are randomly generated for each applications. The average energy consumption for each approach in each test is collected. All results are normalized against the ones by MK_{SD} and filled into Table 1.

(m, k)	Avg. Energy			Improvement	
	MK_{SD}	MK_{DYN}	MK_{LP}	MK_{SD}	MK_{DYN}
Util					
0.0-0.1	100	98.2	44.9	55.1%	53.3%
0.1-0.2	100	98.1	43.4	56.6%	55.7%
0.2-0.3	100	96.7	53.4	46.6%	44.8%
0.3-0.4	100	92.8	52.2	47.8%	43.8%
0.4-0.5	100	91.9	64.3	35.7%	30.0%
0.5-0.6	100	89.7	65.8	34.2%	26.6%
0.6-0.7	100	85.2	75.4	24.6%	11.5%
0.7-0.8	100	89.8	89.0	11.0%	0.89%
0.8-0.9	100	83.3	83.3	16.7%	0.00%
0.9-1.0	100	80.2	80.2	19.8%	0.00%
web phone	100	99.1	90.1	19.9%	9.08%
CNC	100	92.1	71.0	29%	22.9%
INS	100	96.0	87.0	13%	9.38%

Table 1. Average energy consumption and corresponding energy saving improvement of MK_{LP} over MK_{SD} and MK_{DYN} for randomly generated task sets with the ideal dual mode processor, and that for three real applications (web phone [22], CNC [15], and INS [1]) with the practical two-mode processor (Motorola PowerPC 860 [14]).

From Table 1, one can readily see that our approach have much better energy saving performance than the other two. Compared with the simply shut-down-when-idle strategy, our approach can save the average energy consumption up to around 56%. This is because that, our approach can assign some mandatory jobs that initially need to be executed in the high voltage mode to the low voltage mode and still guarantee their deadlines, and hence, the (m, k) -requirements. The more mandatory jobs can be assigned to the low voltage mode, the higher the energy saving improvement. Compared with the dynamic slack reclaiming, our proposed technique has significant energy saving improvement when the system utilization is relative small. As the system utilization increases, the energy sav-

ing improvement is decreasing. For example, from Table 1, the energy saving improvement can be as high as around 56% when the system (m, k) -utilization is among 0.1 to 0.2, while the improvement becomes around 11% for system with (m, k) -utilization among 0.6 to 0.7. The reason is that, when the utilization is lower, the probability for statically assign the execution of mandatory jobs is higher. Therefore, our approach can have a much higher energy saving improvement than using the dynamic slack reclaiming technique alone. If the system utilization is very high, statically assigning a mandatory job to the low voltage mode will potentially cause it or other mandatory jobs to miss the deadlines, and thus the energy improvement is not as significant as that for the systems with low utilizations.

The above analysis also conform to the experimental results in a more practical environment as shown in the last three rows of Table 1. Our approach can lead to much less energy consumption, *i.e.* 13% to 29%, than the power down strategy. Compared with the dynamic slack reclaiming technique, our approach can save more energy, *i.e.* from 9.08% to 22.9%. Overall, the experimental results based on both the randomly generated systems as well as the practical applications have clearly demonstrate the effectiveness of our approach in saving energy.

7 Conclusions and future work

Energy consumption and QoS guarantee are two of the most critical factors for the successful design of pervasive real-time computing system. We propose in this paper a static scheduling technique that can reduce energy consumption while also ensuring QoS requirements deterministically on a two mode variable voltage processor. Our technique can be readily incorporated with existing dynamic scheduling techniques to further save the energy. Experiments with parameters drawn from both synthesized systems as well as practical applications have shown the effectiveness of our approach. The future work of this research includes the study of schedulability for the dynamically determined mandatory jobs and also a less computation intensive algorithm for the multiple level voltage processor.

References

- [1] A. Burns, K. Tindell, and A. Wellings. Effective analysis for engineering real-time fixed priority schedulers. *IEEE Transactions on Software Engineering*, 21:920–934, May 1995.
- [2] H. Aydin, R. Melhem, D. Mosse, and P. Alvarez. Determining optimal processor speeds for periodic real-time tasks with different power characteristics. In *ECRTS01*, June 2001.
- [3] G. Bernat and A. Burns. Combining (n,m) -hard deadlines and dual priority scheduling. In *RTSS*, Dec 1997.
- [4] K. Govil, E. Chan, and H. Wasserman. Comparing algorithms for dynamic speed-setting of a low-power cpu. In *MOBICOM*, pages 13–25, November 1995.

- [5] M. Hamdaoui and P. Ramanathan. A dynamic priority assignment technique for streams with (m,k) -firm deadlines. *IEEE Transactions on Computers*, 44:1443–1451, Dec 1995.
- [6] S. Hua, G. Qu, and S. Bhattacharyya. Energy reduction techniques for multimedia applications with tolerance to deadline misses. *DAC*, pages 131–136, 2003.
- [7] T. Ishihara and H. Yasuura. Voltage scheduling problem for dynamically variable voltage processors. In *ISLPED*, pages 197–202, August 1998.
- [8] W. Kim, J. Kim, and S.L.Min. A dynamic voltage scaling algorithm for dynamic-priority hard real-time systems using slack analysis. *DATE*, 2002.
- [9] J. Liebeherr, D.W.Wrege, and D. Ferrari. Exact admission control for networks with a bounded delay service. *IEEE/ACM Transactions on Networking*, 4:885–901, 1996.
- [10] C. L. Liu and J. W. Layland. Scheduling algorithms for multiprogramming in a hard real-time environment. *Journal of the ACM*, 17(2):46–61, 1973.
- [11] J. R. Lorch and A. J. Smith. Improving dynamic voltage scaling algorithms with PACE. In *SIGMETRICS/Performance*, pages 50–61, 2001.
- [12] A. R. M. Ltd. Introduction to thumb. version 2.0. *ARM DVI-0001A*, 1995.
- [13] T. Ma and K. Shin. A user-customizable energyadaptive combined static/dynamic scheduler for mobile applications. *RTSS*, pages 227–236, 2000.
- [14] MPC860. Mpc860 powerpc hardware specifications. *MPC860EC/D*, Motorola 1998.
- [15] N.Kim, M. Ryu, S. Hong, M. Saksena, C. Choi, and H. Shin. Visual assessment of a real-time system design: a case study on a cnc controller. In *RTSS*, Dec 1996.
- [16] T. Pering, T. Burd, and R. Brodersen. The simulation and evaluation of dynamic voltage scaling algorithms. *ISLPED*, pages 76–81, 1998.
- [17] P. Pillai and K. G. Shin. Real-time dynamic voltage scaling for low-power embedded operating systems. In *SOSP*, 2001.
- [18] Q. Qiu, Q. Wu, and M. Pedram. Dynamic power management in a mobile multimedia system with guaranteed quality-of-service. In *DAC*, pages 834–839, 2001.
- [19] P. Ramanathan. Overload management in real-time control applications using (m,k) -firm guarantee. *IEEE Trans. on Paral. and Dist. Sys.*, 10(6):549–559, Jun 1999.
- [20] K. F. S. Reinhardt and T. Mudge. Automatic performance-setting for dynamic voltage scaling. *MOBICOM'01*, 2001.
- [21] C. Rusu, R. Melhem, and d. Mosse. Maximizing the system value while satisfying time and energy constraints. *IBM Journal on Res. & Dev.*, 47(5/6):689–701, Sep/Nov 2003.
- [22] D. Shin, J. Kim, and S. Lee. Intra-task voltage scheduling for low-energy hard real-time applications. *IEEE Design and Test of Computers*, 18(2), March-April 2001.
- [23] M. Weiser, B. Welch, A. Demers, and S. Shenker. Scheduling for reduced cpu energy. In *Proceedings of USENIX Symposium on Operating System Design and Implementation*, pages 13–23, 1994.
- [24] R. West and K. Schwan. Dynamic window-constrained scheduling for multimedia applications. In *ICMCS*, 1999.
- [25] F. Yao, A. Demers, and S. Shenker. A scheduling model for reduced cpu energy. In *IEEE Annual Found. of Comp. Sci.*, pages 374–382, 1995.
- [26] W. Yuan and K. Nahrstedt. Integration of dynamic voltage scaling and soft real-time scheduling for open mobile systems. In *NOSS-DAV*, 2002.
- [27] W. Yuan and K. Nahrstedt. Energy-efficient soft real-time cpu scheduling for mobile multimedia systems. In *SOSP'2003*, 2003.
- [28] W. Yuan and K. Nahrstedt. Recalendar: Calendaring and scheduling applications with cpu and energy resource guarantees for mobile devices. In *PerCom*, 2003.
- [29] Q. Zheng and K. G. Shin. On the ability of establishing real-time channels in point-to-point packet-switched networks. *IEEE Trans. on Comm.*, 42(2/3/4):1096–1105, 1994.