2009 Third International Conference on Sensor Technologies and Applications

Scheduling real-time of the synchronous hybrid tasks under energy constraint

Akli Abbas, Hamid Hentous
Computer Science Department, M.P. School
BP 17, Bordj El Bahri, Algiers, Algeria
e-mail: {akliabbas, hentoush}@yahoo.fr

Tayeb Kenaza
Artificial Intelligence Laboratory, M.P. School
BP 17, Bordj El Bahri, Algiers, Algeria
e-mail: ken_tayeb@yahoo.fr

Abstract— **In this paper, we are interested by the integration of precedence and resources sharing constraints in the real time scheduling of periodic and aperiodic tasks. We also treat the problem of dynamic voltage scaling of the processor in the same context. Many dynamic voltage scaling algorithms have been proposed in the literature. However, these algorithms do not consider the constraints of precedence and shared resources on the scheduling of periodic and aperiodic tasks. In this case, we propose two algorithms with the constraints cited above. Experimental results show that the proposed algorithms reduce the energy consumption under earliest deadline first scheduling policy and the stack resource policy.**

Keywords- Real time scheduling; embedded system; resources precedence and constraints; periodic and aperiodic tasks; power reduction.

I. INTRODUCTION

This paper treat the problem of scheduling real time of the periodic and aperiodic tasks synchronous, under constraint of energy in the embedded systems. These systems often require periodic hard real-time tasks and aperiodic tasks with soft deadlines. However, we must decrease their response times, without compromising the execution of the periodic tasks. Moreover, such systems generally need tasks which are dependent and collaborate to realize the awaited objectives from these systems. This collaboration is done by the data exchange and/or the resource sharing. However, it collaboration generates constraints of precedence and resource sharing, which we must take into account in the analysis of the scheduling of the tasks. These same systems are characterized by their autonomous functioning, whose energy supply is ensured by batteries. So the reduction of the power consumption became crucial metric of optimization in the design and the realization of such systems. The objective, in this case, is not only to determine the order of execution of the tasks under time constraints and synchronization, but also to fix the frequency of the processor and the supply voltage.

The real-time system considered is composed by periodic and aperiodic tasks. The tasks are scheduled in a system with a single processor that supports variable frequency and voltage levels. These tasks are synchronized by the precedence induced by the communications and/or by the access shared resources in a mutually exclusive manner by the use of semaphores. In this paper, we call a system with periodic and aperiodic tasks a hybrid task system.

This article is organized as follows: In section 2, we summarize the related works. The concepts relating to scheduling of the tasks, the constraints of synchronization and variable speed processors will be detailed in section 3. Our contribution concerning the scheduling of the synchronous hybrid tasks will be discussed in section 4. In section 5 we give two DVS algorithms to compute static slowdown factors in the presence of task synchronization to minimize the energy consumption of the system. The experimental results will be discussed in section 6. We will finish this article by a conclusion in section 7.

II. RELATED WORKS

In the literature, many algorithms were proposed in order to treat the problem of real-time scheduling of the tasks. For example, Liu et al. [1] have proposed the algorithm Earliest Deadline First (EDF) with aim of scheduling the independent periodic tasks, in [2] the author proposes an approach to treat the precedence constraint under EDF policy. As well as the constraints of precedence, the use of resources generates difficulties related to the protection of the access to the resources. To palliate these problems, resource access protocols were proposed. We mention for example the Stack Resource Policy (SRP) proposed by Baker in [3]. For the scheduling of the aperiodic tasks, two solutions are used; that is to say, by a treatment of background or the use of server task. Among this last solution, we evoke the total bandwidth server (TBS) [4]. The TBS solution is combined with protocol SRP in [5, 6] in order to schedule the hybrid tasks which access to critical resources. The DVS is one of the most effective approaches in reducing the power consumption of real-time systems. When the required performance of the target system is lower than the maximum performance, supply voltage can be dynamically reduced to the lowest possible extent that ensures a proper operation of the system. Recently, many voltage scheduling algorithms have been proposed for hard real-time systems. Thus, when EDF policy is used and in the case of attribution speed to jobs tasks, Yao et al. proposed a polynomial algorithm presented in [7], to enable the scheduling of periodic tasks in order to minimize the power consumption. In the case of the attribution of a single speed to all the tasks, Aydin et al. [8] gave an approach based on the necessary and sufficient condition suggested in [1]. When the tasks share critical resources, Jejurikar et al. [9] presented an algorithm to calculate the slowdown factors of the processor. The authors

978-0-7695-3669-9/09 $25.00 © 2009 IEEE
DOI 10.1109/SENSORCOMM.2009.122

use the analytical condition of scheduling proposed in [3]. Shin et al. [10] suggested an algorithm to computation slowdown factors speed of the processor for a system with periodic and aperiodic tasks. Thus, many algorithms were developed for DVS approach to the real-time systems to scheduling the periodic tasks with different constraints. However, these algorithms do not consider the constraints of sharing critical resources and precedence in scheduling of hybrid tasks.

III. PRELIMINARIES

This section describes the system model and variable speed processors.

A. System Model

We consider a single processor system compound by a set of periodic tasks noted $T=\{t_1\ldots,t_n\}$ and by a set of aperiodic tasks noted $T_a= \{t_{a1}\ldots,t_{am}\}$. The periodic tasks are modeled by the 4-tuple $t_i=\{r_{0,i}, C_i, D_i, P_i\}$ such as $r_{0,i}$ is the first request time of the task t_i, C_i is the worst case execution time (WCET), D_i is the relative deadline and P_i is the period of the task with $D_i \leq P_i$. Its k^{th} absolute deadline is $d_{i,k} = r_{k,i} + D_i$. Each invocation of the task is called a job and the k^{th} job of task t_i is denoted as $t_{i,k}$. A periodic task set is said to be schedulable if all jobs meet their deadline. The processor utilization by periodic tasks set, $U = \sum_{i=1}^{n} C_i/P_i \leq 1$ is a necessary condition for the feasibility of any schedule. As for the aperiodic tasks, they are modeled by a 1-tuple $t_{ai}=\{C_{ai}\}$, where C_{ai} is the execution time. An aperiodic task set is specified by the mean arrival rate λ and the mean service rate μ. The system has a set of shared resources which are accessed by the periodic and aperiodic tasks in a mutually exclusive manner by using semaphores. When a task has been granted access to a shared resource, it is said to be executing in its critical section. The k^{th} critical section of task t_i or t_{ai} which uses a R_k resource is represented as $\beta_{i,k}$. We say a task is blocked if the task has to wait for a lower priority task to release a shared resource and the task holding the resource is called the blocking task. Note that the amount of time a task t_i is blocked is referred to as the task blocking time noted B_i. With the specified task information and a given resource access protocol, the maximum blocking time for a task can be computed. The majority of real-time applications require communications between the periodic tasks, which introduce precedence constraints. Thus, we say that there is a precedence constraint between the task t_i and the task t_j or t_i precedes t_j, noted ($t_i \rightarrow t_j$), if t_j must await the end execution of t_i to begin its own execution. We assume that the periodic tasks have an atomic form (i.e. normal form), in such way that the waiting of messages by a task are in beginning, and the emission of messages are at the end of the task. We assume also that the precedence constraints between tasks are simple (i.e. $t_i \rightarrow t_j \Rightarrow P_i = P_j$).

B. Variable speed processors

A wide range of processors support variable voltage and frequency levels. Voltage and frequency levels are tightly coupled. So the important point to note is when we perform a slowdown, we change both the frequency and voltage of the processor. We assume that the speed can be varied continuously from S_{min} to the maximum supported speed [9]. We normalize the speed to the maximum speed to have a continuous operating range of $[S_{min}, 1]$, where $S_{min} = f_{min}/f_{max}$ with the minimum and maximum frequencies represented by f_{min} and f_{max} respectively.

IV. SCHEDULING OF THE SYNCHRONOUS HYBRID TASKS

We consider in this section the problem of the scheduling of the synchronous hybrid tasks. The objective is to have an order of execution of the tasks which guarantee the respect of deadline, the use of the critical resources in mutual exclusion and the respect of the precedence constraints by the periodic tasks, and to allow, as far as possible, to have a better response time by aperiodic tasks that are also using the critical resources in mutual exclusion.

A. Precedence constraints

We exploit the approach which was proposed in [2] in order to consider the precedence constraints that exist between periodic tasks during the use of EDF policy. This policy is based on assignments of priority following the temporal parameters of tasks. Thus, the idea is to assign to a task, which must precede another, a priority lower than the task preceding it. This approach suggests the modification of the request time and deadline so that the precedence constraints are implicitly respected. These modifications are operated as follows:

- Computation of request time:

$$r_{0,i}^{*} = max\left\{r_{0,i}, max_{tj \rightarrow ti}\left\{r_{0,j}^{*} + C_j\right\}\right\} \quad (1)$$

  From the tasks without predecessors and while going down to the tasks of which all the predecessors were treated.

- Computation of deadline:

$$D_i^{*} = min\left\{D_i, min_{ti \rightarrow tj}\left\{D_j^{*} - C_j\right\}\right\} \quad (2)$$

  From the tasks without successors and while going up to the tasks of which all the successors were treated.

We note that these modifications are operated in off-line i.e. before the effective scheduling of the tasks. Moreover, we must consider in the continuation that the deadline of a task can be lower than its period, and requests times are different following the modifications operated previously.

B. Precedence and resource constraints

We choose to use the SRP protocol [3] to manage resource access in mutual exclusion and, in addition, to compute the maximum blocking time for a task, with taking into account the constraints precedence. This protocol is based on some principles, which are summarized as follows:

- In addition to its priority, each task is been allotting statically (off-line), a parameter π called preemption level. These levels are assigned in a way inversely proportional to the relative deadline.

- Each resource is been assigning dynamically a ceiling value noted C_{Rk}, determined by the maximum value of the preemption levels of the active tasks needing more than the unit available resource R_k.

$$C_{RK} = max_i\{\pi_i/t_i \ \ can\ be\ blocked\ in\ R_k\} \quad (3)$$

- The system ceiling noted Π, is the maximum value of the ceilings resources in use.
- A task t_i, can preempt task t_j if the following conditions are verified:
  a)- Its absolute deadline d_i is lower than that of t_j.
  b)- The preemption level of t_i is higher than of t_j.
  c)- Its preemption level is higher than the system ceiling.

Baker established in [2] a sufficient condition to verify scheduling of the periodic tasks as follows:

$$\forall i=1...n; \quad \sum_{j=1}^{i} \frac{C_j}{D_j} + \frac{B_i}{D_i} \leq 1 \quad (4)$$

B_i represents the maximum blocking time of the task t_i, it is equal to the biggest duration of the critical section of the periodic tasks, which have a lower preemption level than that of tasks t_i and it uses a resource having the ceiling value higher or equal to the preemption level of t_i.

To consider the precedence constraints, we suggest that the tasks having lower preemption level considered are those not on relations, directly or indirectly, by constraints of precedence with the task t_i. This duration is given by the following formula:

$$B_i = max_{j,k}\{\beta_{j,k}/\pi_i > \pi_j \wedge C_{Rk} \geq \pi_i \wedge t_j \notin \{succ\ of\ t_i\}\} \quad (5)$$

Where $\beta_{j,k}$ represents the critical section of task t_j, which have a lower preemption level π_j than that of tasks t_i and which use a resource having the ceiling C_{Rk} value higher or equal to the preemption level of t_i. The task t_j is considered if it is not included in the set of predecessors or successors of t_i.

C. Aperiodic tasks

When the system is composed by periodic and aperiodic tasks which share critical resources, we consider the two following approaches:

1) First approach: We consider that the aperiodic tasks are managed by a server task which is the higher priority periodic task and its execution time is equal to the sum of the execution times of the aperiodic tasks. With each activations of this server, it treats the aperiodic tasks blocked in the queue, according to policy FIFO (first in first out), until exhaustion of its execution time.

To analyze the scheduling of the synchronous hybrid tasks, we used sufficient condition of scheduling established by Baker in [3], which we adapted as follows:

$$\forall i=1...n+1; \sum_{j=1}^{i} C_j/D_j + B_i/D_i \leq 1 \quad (6)$$

Where, $n+1$ represents the number of periodic tasks with the task server. When B_i is calculated referring to (5).

2) Second approach: The second approach of scheduling of the aperiodic tasks is based on the idea developed by Caccamo et al. in [5]. The authors suggested assigning at each aperiodic request a deadline and a preemption level so that they can be scheduled by SRP protocol with the use of the total bandwidth server noted TBS. The attribution of deadline is done as follows:

$$d_k = max(r_k, d_{k-1}) + \frac{C_{ak}}{U_s} \quad (7)$$

Where r_k is the invocation date of the k^{th} job task aperiodic, C_{ak} its execution time, d_{k-1} the deadline of the precedent job ($d_0 = 0$) and U_s is the load of the server used. This load admits for value, $U_s = 1-U$. This attribution of deadline respects the fact that the use of the processor by the aperiodic ones never exceeds the value of the server.

As for the preemption level, it is calculated as follows:

$$\pi^* = \frac{U_s}{C_{ak}} \quad (8)$$

For the analysis of the scheduling of the synchronous hybrid tasks, we consider the sufficient condition which was proposed by Lipari et al. in [6] such as the tasks are in ascending order of their preemption levels; the analytical condition is as follows:

$$U+U_s \leq 1 \quad (9)$$

$$\forall i, 1 \leq i \leq n, \qquad D_i \geq \sum_{j=1}^{i} \left\{ \left\lfloor \frac{(D_i - D_j)}{P_j} \right\rfloor + 1 \right\} C_j + max\{0, B_i - 1\} + S_{(i)} D_i U_s \quad (10)$$

With $S_{(i)}$ is the selection function which takes as values:

$$S_{(i)} \begin{cases} 0 \ if\ \pi_i \geq \pi^* \\ 1 \ if\ \pi_i < \pi^* \end{cases} \quad (11)$$

Where π^* represents the maximum preemption level of the aperiodic tasks, π_i is the preemption level of the periodic task t_i. D_i is the deadline of the task t_i, D_j and P_j are respectively the deadline and the period of the periodic task t_j having a preemption level lower than that of t_i. U is the processor utilization by periodic tasks and U_s is the load of the server used. B_i is calculated referring to (5).

V. Computation of Task Slowdown Factors

In this section, we compute task slowdown factors in the presence of hybrid tasks synchronization. We present two algorithms which are used to compute slowdown factors according to the two approaches presented previously.

A. Algorithm I

Shin et al. [10] gave an algorithm to solve the problem of the determination of the task slowdown factors, to reduce the power consumption during the scheduling of the periodic

and aperiodic tasks. This algorithm determines two slowdown factors to determine the speed of execution of the tasks. The first relates to the periodic tasks noted S_p, when the other noted S_s relates to the aperiodic tasks. However, authors do not consider constraints of resources and precedences.

For the determination of the slowdown factors, we based on the model presented in the second approach. We formulate the problem as follows:

Given: U, U_s, η, ρ, $\{t_1 \ldots, t_n\}$ of n periodic tasks with their characteristics (C_i, P_i, B_i)

Find: S_s, S_p which minimize energy E;

$$E = S_p^2 * \eta + S_s^2 * \rho \quad (12)$$

Subject to:

$$U/S_p + U_s/S_s = 1 \quad (13)$$

$$\forall i,\ 1 \leq i \leq n,\ D_i \geq \sum_{j=1}^{i} \left\{ \left\lfloor \frac{D_i - D_j}{P_j} \right\rfloor + 1 \right\} \frac{C_j}{S_p} + max\left\{0,\ \frac{B_i}{S_p} - 1\right\} + S_{(i)} D_i \frac{U_s}{S_s} \quad (14)$$

$$\text{With } 0 < S_s \text{ and } S_p < 1 \quad (15)$$

Where refer to (9), η is the average workload ratio of periodic tasks ($\eta = U/n$), ρ is the average workload ratio of aperiodic tasks ($\rho = \lambda / \mu$) and S_p and S_s are the two slowdown factors.

We give the solution to this problem by using Lagrange transform as follows:

$$S_p = U + U_s \sqrt[3]{\frac{\rho}{U_s * \eta}} \ ; \ S_p = U_s + U \sqrt[3]{\frac{U_s * \eta}{\rho}} \quad (16)$$

B. Algorithm II

In this paragraph, we give an algorithm which calculates the slowdown factor noted S_f when the first approach is used. The slowdown factor is calculated so that the following analytical condition is verified:

$$\forall i,\ i = 1 \ldots n+1;\ \sum_{j=1}^{i} C_j / (D_j * S_f) + B_i / (D_i * S_f) \quad (17)$$

Algorithm II Computation of task slowdown factor S_f

```
1.  T = {t1,…,tn} Given n tasks in non-increasing order of
    their relative deadline with their characteristics (Ci, Di,
    Bi)
2.  X ← 0, Y← 0, i←1; V←0 {initialization}
3.  while (i≤n) do
4.      j ←1;
5.      while (j ≤ i) do
6.          X+= Cj / Dj ;
7.          j← j+1 ;
8.      end while
9.      Y← Bi / Di ;
10.     if (X+Y >1) then
11.         Exit() ;
12.     end if
13.     V =max(V, X+Y);
14.     X ←0 ;
15.     i← i+1 ;
16. end while
17. Sf ← V
```

VI. Experimental Results

This section describes simulations and the experimental results carried out in order to evaluate the two approaches described previously and to compare between the two algorithms for the dynamic voltage scaling (DVS) developed in the framework of our study. Overall the simulation results were promising.

A. Environment and context of simulation

We developed a simulator program with VC++, in which we implemented the EDF dynamic policy, SRP protocol and algorithms that we had developed. To carry out simulations, we randomly generated some Task-sets. For that, we generated according to the uniform law characteristics concerning the periodic tasks. Periods (P_i) are generated in the interval [50, 80], the execution times (C_i) in the interval [4, 8].

Resources R_i from 0 to 2 and the position of the critical section is randomly generated. The length of the critical section ($\beta_{i,k}$) of the task t_i that access to R_k is generated in the interval [1, C_i]. The precedences of the task t_i with other tasks (from 0 to *n-1*).

For the aperiodic tasks, the interval of time between two arrivals is Poisson process with parameter $\lambda = 0.01$ and the execution time is exponential process with parameter $\mu = 0.25$. Such as $1/\lambda$ is the average duration of the inter-arrivals, $1/\mu$ is the average duration of service. Theoretically response time of the aperiodic tasks is calculated by Markov model M/M/1 which is $(\mu - \lambda)^{-1} = 4.16$.

B. Results

We randomly generated four sets from 1 to 4 periodic tasks with one aperiodic task in each set according to the context specified previously then these sets were used to test the two approaches and two algorithms.

We note that in the case of the first approach, we generate also a task server having a high priority (having the smallest deadline) and in the case of the second approach, we assign to TBS a load equal to 0.6. The results of simulations are given by the following tables and figure which allow us to discuss some of them.

TABLE I. SLOWDOWN FACTORS BY ALGORITHM I (U_s=0.6)

Set	Algorithm I						
	Before			*slowdown factors*		*After*	
	U_s	U	Tr_1	S_s	S_p	Tr_2	*E. profit*
1	0.6	0.097	4	0.710	0.627	5.637	0.303
2	0.6	0.175	4	0.792	0.723	5.060	0.225
3	0.6	0.290	4.018	0.928	0.820	4.322	0.110
4	0.6	0.349	4.022	0.982	0.897	4.090	0.051

TABLE II. SLOWDOWN FACTORS BY ALGORITHM II

Set	Algorithm II				
	Before		*slowdown factors*	*After*	
	U	Tr_1	S_f	Tr_2	*E. profit*
1	0.097	25.47	0.18	47	0.82
2	0.175	29.7	0.26	40.1	0.75
3	0.290	29.72	0.37	36.4	0.63
4	0.349	29.72	0.43	34.6	0.57

We notice that algorithm I (second approach) gives the best average response time noted T_r and an energy profit in raised compared to those of algorithm II (first approach). That is due to the fact that we allotted a raised load to the server TBS (U_s=0.6).

However, when we allot a weak load to TBS (U_s =0.05), as shown by table 3, we note that the average response time and the profit in energy is less significant than using the second algorithm (first approach). This situation can occur if the load induced by the periodic tasks is significant.

TABLE III. SLOWDOWN FACTORS BY ALGORITHM I (US=0.05)

Set	Algorithm I						
	Before			*slowdown factors*		*After*	
	U_s	U	Tr_1	S_s	S_p	Tr_2	*E. profit*
1	0.05	0.097	10	0.1	0.2	71.2	0.853
2	0.05	0.175	5.59	0.13	0.28	53.9	0.775
3	0.05	0.29	6.46	0.19	0.39	45.2	0.66
4	0.05	0.349	7.35	0.22	0.45	42.3	0.601

The following table and graph show the variation of the average response time and the energy profit according to the load allotted to the server TBS during the scheduling of four periodic and one aperiodic tasks in the case of using algorithm I (second approach).

TABLE IV. VARIATION OF THE AVERAGE RESPONSE TIME AND THE PROFIT ACCORDING TO THE VARIATION OF US

U_s	Tr_1	UC_1	S_s	S_p	Tr_2	UC_2	E. profit
0.05	7.35	0.35	0.22	0.45	42.31	0.78	0.60
0.10	4.02	0.38	0.31	0.52	13.09	0.69	0.55
0.15	4.02	0.38	0.39	0.57	10.28	0.62	0.50
0.20	4.02	0.38	0.46	0.61	8.66	0.64	0.45
0.25	4.02	0.38	0.54	0.66	7.52	0.59	0.40
0.30	4.02	0.38	0.60	0.69	6.66	0.55	0.35
0.35	4.02	0.38	0.67	0.73	6.01	0.52	0.30
0.40	4.02	0.38	0.73	0.77	5.48	0.50	0.25

With an aim of showing the variation of average response time and energy profit according to a load allotted to server TBS, we normalized the average response times by the maximum value, the result is shown by the following graph.

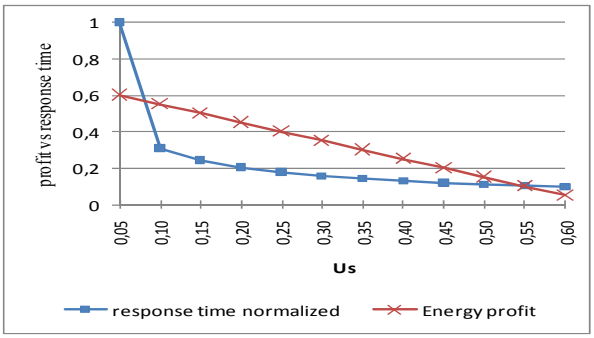

Figure 1. Variation of the average response time and the profit according to the variation of U_s

VII. CONCLUSION

In this article, we proposed two algorithms for the dynamic voltage scaling (DVS) in a context of realtime scheduling of synchronous hybrid tasks. Constraints of sharing resource and precedence were studied in a context of minimization of energy. We showed that the two algorithms gave convincing results in terms of simulations. According to the experimentations done we can saw that our approaches are promising.

REFERENCES

[1] C.L. Liu and J.W. Layland. "Scheduling algorithms for multiprogramming in real-time environment". Journal of the ACM, 1973, pp. 20(1): 46–61.

[2] J. Blazewicz "Scheduling dependent tasks with different arrival times to meet deadlines" in E. Gelembre, H. Beiher Modelling and performance evaluation of computer systeme Nort-holland, amsterdem, 1976.

[3] T.P. Baker. "Stack-based scheduling of real-time processes". The Journal of Real-Time Systems, 1991, pp. 3 :67–99.

[4] M. Spuri and G.C. Buttazzo, “Efficient Aperiodic Service under Earliest Deadline Scheduling", Proc.of the IEEE Real-Time Systems Symposium, San Juan, Portorico, December 1994.

[5] M. Caccamo, G. Lipari, and G. Buttazzo, “Sharing resources among periodic and aperiodic tasks with dynamic deadlines”, Proceedings of the IEEE Real-Time Systems Symposium, Phoenix, Arizona, 1999, pp. 284-293.

[6] G. Lipari and G. Buttazzo, “Schedulability analysis of periodic and aperiodic tasks with resource constraints”, Journal of Systems Architecture, 2000, Vol. 46, No. 4, pp. 327-338.

[7] F. Yao, A. J. Demers, and S. Shenker, “A Scheduling Model for Reduced CPU Energy”, in IEEE Symposium on Foundations of Computer Science, 1995, pp. 374-382.

[8] H. Aydin, R. Melhem, D. Mossé and P. Mejia-Alvarez, “Determining optimal processor speeds for periodic real-time tasks with different power characteristics“ Euromicro Conference on Real-Time Systems, 2001, pp. 225-232.

[9] R. Jejurikar and R. Gupta, “Energy Aware Task Scheduling with Task Synchronization for Embedded Real Time Systems “, Proc. Int’l Conf. Compilers, Architecture and Synthesis for Embedded Systems, 2002.

[10] D. Shin and J. Kim, "Dynamic Voltage Scaling of Periodic and Aperiodic Tasks in Priority-Driven Systems," in Proc. ASPDAC'03, Jan. 2004, pp. 653-658.