Investigating the Effect of Voltage-Switching on Low-Energy Task Scheduling in Hard Real-Time Systems[1]

Vishnu Swaminathan and Krishnendu Chakrabarty
Dept. of Electrical & Computer Engineering, Duke University
Durham, NC 27708, USA
{vishnus,krish}@ee.duke.edu

Abstract

We investigate the effect of voltage-switching on task execution times and energy consumption for dual-speed hard real-time systems, and present a new approach for scheduling workloads containing periodic tasks. Our method minimizes the total energy consumed by the task set and guarantees that the deadline for every task is met. We present a mixed-integer linear programming model for the NP-complete scheduling problem and solve it for moderate-sized problem instances using a public-domain solver. For larger task sets, we present a novel extended-low-energy earliest-deadline-first (E-LEDF) scheduling algorithm and apply it to two real-life task sets. Our results show that energy can be conserved in embedded real-time systems using energy-aware task scheduling. We also show that switching times have a significant effect on the energy consumed in hard real-time systems.

1 Introduction

In embedded systems with variable-speed processors, the operating system (OS) can reduce energy consumption by scheduling tasks appropriately. For real-time systems, optimal preemptive off-line scheduling algorithms have been developed [16]. Heuristics for off-line scheduling of non-preemptive real-time tasks were presented in [4]. On-line and off-line scheduling algorithms for the preemptive task model have also been developed [6, 11]. From the hardware perspective, researchers have developed efficient DC-DC switching converters that allow the supply voltage to be rapidly changed under external control [10]. Although the scheduling methods cited above are very efficient, most of them make the assumption that the CPU can operate at several different voltage levels (and hence different clock frequencies) which can be varied continuously. In addition, a number of these methods are aimed at the synthesis of low-power designs and they do not address energy minimization during field operation.

In this paper, we present an on-line scheduling algorithm for real-time systems that attempts to minimize the energy consumed by a periodic task set while also considering the voltage switching times and energies. The algorithm is based on the well-known earliest-deadline-first (EDF) algorithm [9, 7]. We consider a practical scenario where a single CPU executes a set of periodic non-preemptable tasks. The voltage, and consequently the clock speed, of the CPU may be switched between two or more values dynamically at run-time through OS system calls. This option is available in most modern computers, which provide at least two different operating speeds [5]. Unlike previous approaches, e.g. [4], we make the realistic assumption that voltage switching takes time and consumes energy. During this switching period, the CPU cannot execute any tasks. We attempt to find the voltage at which each task must be executed such that the energy consumed by the entire set of periodic tasks is minimized and generate a schedule for the task set such that the release time requirements are satisfied and the deadlines for every task is met, while also accounting for the effect of the voltage switching times on the energy consumption of the task set.

2 Preliminaries

In this section, we present our notation and the underlying assumptions. We are given a set of n periodic tasks $R = \{r_1, r_2, \ldots, r_n\}$. Each task $r_i \in R$ has the following parameters: a *release* (or arrival) time a_i, a deadline d_i, a length l_i (represented in number of instruction cycles), and a period p_i.

Each task r_i is released at time $t = a_i$. We assume, without loss of generality, that all tasks have identical periods. All tasks must complete execution before their deadlines.

We assume that the CPU can operate at one of two voltages: V_1 or V_2. Depending on the voltage level, the CPU speed may take on two values: s_1 or s_2. The model can be

Report Documentation Page

1. REPORT DATE **2005**	2. REPORT TYPE	3. DATES COVERED **-**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Investigating the Effect of Voltage-Switching on Low-Energy Task Scheduling in Hard Real-Time Systems	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Defense Advanced Research projects Agency,3701 North Fairfax Drive,Arlington,VA,22203-1714**	8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**4**	

easily extended to handle more than two voltages (speeds). The supply voltage to the CPU is under OS control and the OS may dynamically switch the voltage during run-time. We restrict ourselves to two speeds out of practical considerations. CPU speeds are specified in terms of the number of instructions executed per second. Each task r_i may be executed at a voltage v_i, $v_i \in \{V_1, V_2\}$, and correspondingly, at a speed x_i, $x_i \in s_1, s_2$. The voltage (speed) may be switched to a different value between tasks. The system uses C units of energy and t_s units of time in performing a voltage switch.

It is well-known that power consumption in CMOS circuits has a quadratic dependence on the CPU voltage. Hence, energy E_i consumed by task r_i of length l_i is given by $E_i \propto v_i^2 l_i$, and the total energy consumed for a given task set is proportional to $\sum_{i=1}^{n} v_i^2 l_i$. In the subsequent dicussion, we measure energy consumption in units of $\sum_{i=1}^{n} v_i^2 l_i$.

Although the scheduling problem is NP-complete, it can be solved exactly for moderate-sized instances using mixed-integer linear programming (MILP). In the following section, we present an MILP model for the low-energy scheduling problem with switching times and energies.

3 Mixed-integer linear programming model

The goal of MILP is to minimize a linear objective function on a set of integer and/or real variables, while satisfying a set of linear constraints.

The optimization problem we address is to minimize the total energy consumed by the set of n tasks by optimally determining their start times $t_1, t_2, \ldots, t_n$, their voltages and corresponding execution speeds.

The following constraints need to be modeled: (i) CPU speeds are limited to one of two values—s_1 or s_2, (ii) the deadline for each task must be met, (iii) real-time tasks are non-preemptable, (iv) a task may start only after it has been released, and (v) tasks immediately follow each other.

We observed in Section 2 that the energy consumed by task r_i is $E_i \propto v_i^2 l_i$. Moreover, with each task r_i is an associated switching cost C_i if a voltage switch was performed immediately before the task began execution. Hence, to minimize energy, we need to minimize the term $\sum (l_i v_i^2 + C_i)$, and this is now our objective function.

We assume a linear relationship between the operating voltage v of the processor and its execution speed x. This generally holds true for processors designed using CMOS technology [13]. Hence, $v_i = \dfrac{x_i}{\alpha}$.

First, we introduce binary variables a_i and b_i to restrict x_i, the execution speed of task i, to be either s_1 or s_2. The following constraints relate a_i and b_i to x_i: $x_i = a_i s_1 + b_i s_2$, and $a_i + b_i = 1$.

Since x_i may be written in terms of s_1 and s_2, and v_i^2 is

proportional to x_i^2, we have: $v_i^2 = \frac{x_i^2}{\alpha^2} = a_i \cdot \frac{s_1^2}{\alpha^2} + b_i \cdot \frac{s_2^2}{\alpha^2}$.

We next model the requirement that all task deadlines must be met. This is ensured by introducing completion time variables $c_i = t_i + \frac{l_i}{x_i} + t_s sw_i$, where t_s is the time taken to perform a voltage switch, and sw_i is a binary variable whose value is 1 if a switch occurred immediately before task r_i. We hence obtain the following inequality: $c_i \leq d_i$. The above inequalities are non-linear. The non-linearities may be eliminated using standard techniques [15].

Non-concurrency of tasks may be modeled with the following inequalities:

$$t_i - t_j - \frac{l_j}{x_j} \geq 0 \ \text{or} \ t_j - t_i - \frac{l_i}{x_i} \geq 0, \ \ 1 \leq i, j \leq n, i \neq j.$$

This represents the fact that when task r_i has been scheduled to start execution at time t_i, no other task r_j can start execution until r_i has completed.

The above inequality is non-linear too. This may be also be linearized using standard linearization techniques. These techniques are not explained here due to space limitations.

Finally, we model the fact that a task r_i starts as soon as a previous task r_j ends. This is done by introducing integer variables $zero_i$ and β_{i1} through β_{in}.

$$zero_i(\beta_{i1}(t_i - c_1) + \beta_{i2}(t_i - c_2) + \ldots + \beta_{in}(t_i - c_n)) = 0.$$

The final MILP model is shown in Figure 1. Although the MILP model is too computationally-intensive to be used for large task sets, it is helpful in determining a lower bound on the amount of energy consumed by a given task set. In the MILP formulation described above, a priori knowledge of the release times has been inherently assumed. We observe that energy can be minimized to a greater extent in the off-line case than in the on-line one. This justifies use of MILP as a comparison tool for providing lower bounds on energy consumption. In the next section, we provide a description of our heuristic and our experimental results.

4 The E-LEDF heuristic

Although MILP is a useful and optimal solution method for small problem instances, it cannot be used for large test cases. In order to solve large problem instances, we have developed a heuristic algorithm to generate near-optimal solutions in polynomial time.

The extended-low-energy earliest deadline first heuristic, or simply E-LEDF, is an extension of the well-known earliest deadline first (EDF) algorithm. The operation of E-LEDF is described in Figure 2.

To illustrate the effectiveness of E-LEDF, we also developed a simple **low-energy earliest deadline first** algorithm (LEDF) [14] that does not take into account the effect of voltage switching. LEDF generated near-optimal schedules

Minimize the cost function $\sum l_i v_i^2 + C_i s w_i$ subject to :

1. $v_i^2 = a_i \cdot \frac{s_1^2}{\alpha^2} + b_i \cdot \frac{s_2^2}{\alpha^2}, \ 1 \leq i \leq n,$
2. $a_i + b_i = 1, \ 1 \leq i \leq n,$
3. $\delta_{ij1} + \delta_{ij2} = 1, \ 1 \leq i, j \leq n, i \neq j,$
4. $c_i = t_i + u_i + t_s s w_i, \ 1 \leq i, j \leq n, i \neq j,$
5. $c_i \leq d_i, \ 1 \leq i \leq n,$
6. $x_i = a_i s_1 + b_i s_2, \ 1 \leq i \leq n,$
7. $v_i - l_i a_i \leq 0, \ 1 \leq i \leq n,$
8. $-u_i + v_i \leq 0, \ 1 \leq i \leq n,$
9. $u_i - v_i + \overline{l_i} a_i \leq l_i, \ 1 \leq i \leq n,$
10. $w_i - l_i b_i \leq 0, \ 1 \leq i \leq n,$
11. $-u_i + w_i \leq 0, \ 1 \leq i \leq n,$
12. $u_i - w_i + \overline{l_i} b_i \leq l_i, \ 1 \leq i \leq n,$
13. $f_{ij1i} - f_{ij1j} - e_{ij1j} - ds_{ij1j} + f_{ij2j} - f_{ij2i} - e_{ij2i} - ds_{ij2j} \geq 0, \ 1 \leq i, j \leq n, i \neq j,$
14. $\gamma_{1ij} + \gamma_{2ij} = 1, \ 1 \leq i, j \leq n, i \neq j,$
15. $f_{ij1i} - d_i \delta_{ij1} \leq 0, \ 1 \leq i \leq n,$
16. $-t_i + f_{ij1i} \leq 0, \ 1 \leq i \leq n,$
17. $t_i - f_{ij1i} + d_i \delta_{ij1} \leq d_i, \ 1 \leq i \leq n,$
18. $e_{ij1i} - l_i \leq 0, \ 1 \leq i, j \leq n, i \neq j,$
19. $-u_i + e_{ij1i} \leq 0, \ 1 \leq i, j \leq n, i \neq j,$
20. $u_i - e_{ij1i} + \overline{l_i} \delta_{ij1} \leq l_i, \ 1 \leq i, j \leq n, i \neq j,$
21. $f_{ij2i} - d_i \delta_{ij2} \leq 0, \ 1 \leq i \leq n,$
22. $-t_i + f_{ij2i} \leq 0, \ 1 \leq i \leq n,$
23. $t_i - f_{ij2i} + d_i \delta_{ij2} \leq d_i, \ 1 \leq i \leq n,$
24. $e_{ij2i} - l_i \leq 0, \ 1 \leq i, j \leq n, i \neq j,$
25. $-u_i + e_{ij2i} \leq 0, \ 1 \leq i, j \leq n, i \neq j,$ and
26. $u_i - e_{ij2i} + \overline{l_i} \delta_{ij2} \leq \overline{l_i}, \ 1 \leq i, j \leq n, i \neq j.$
27. $s w_i = \sum_j f o l_{ij} (a_i \overline{b_j} + a_j b_i), \ 1 \leq i, j \leq n, i \ neqj,$
28. $f o l_{ij} a b s_{ij} - f o l_{ij} \leq a b s_{ij} - 1, \ 1 \leq i, j \leq n, i \neq j,$
29. $\sum_j f o l_{ij} \leq 1, \ 1 \leq i, j \leq n, i \neq j,$
30. $a b s_{ij} = \gamma_{1ij} (t_i - c_j) + \gamma_{2ij} (c_j - t_i),$
31. $f o l_{ij} a b s_{ij} - f o l_{ij} \leq a b s_{ij}, \ 1 \leq i, j \leq n, i \ neqj,$
32. $f o l_{ij} a b s_{ij} = 0, \ 1 \leq i, j \leq n, i \neq j,$
33. $z e r o_i (\beta_{i1} (t_i - c_1) + \beta_{i2} (t_i - c_2) + \ldots + \beta_{in} (t_i - c_n)) = 0, \ 1 \leq i, j \leq n, i \neq j,$
34. $\beta_{i1} + \beta_{i2} + \ldots \beta_{in} = 1, \ 1 \leq i \leq n,$
35. $z e r o_i t_i + z e r o_i \geq t_i, \ 1 \leq i \leq n,$

Figure 1. Mixed-integer linear programming model.

for a number of task sets; hence it was used for comparison purposes.

We ran LEDF and E-LEDF on task sets comprising thirteen to seventeen tasks [14]. We show a plot of the energy consumed for 3 different scheduling strategies in Figure 3.

We observe that the energies consumed in schedules generated by E-LEDF are greater than that of the MILP and LEDF schedules. Our results show that voltage-switching times and voltage switching energies play a significant role in the energy consumption of a real-time system, with the energy rising by as much as 16%.

We next consider two real-life task sets comprising 24 and 39 tasks, respectively. These task sets were used in the development of an application-specific integrated circuit for an avionics application [12] and in an embedded signal processing application for an anti-submarine warfare (ASW) system [1]. For the set of 24 tasks, we assume the processor speeds to be 100 MIPS and 260 MIPS, with corresponding voltages 1.2 V and 3.3 V respectively. In the case of the set of 39 tasks, we have assumed that the processor is capable of operating at 100 MHz and that the processor

Procedure E-LEDF()
begin
Repeat forever
Remember speed at which previous task was scheduled
if *tasks are waiting to be scheduled in ready list* **then**
  Sort deadlines in ascending order; Select task τ_i with earliest deadline
  if *this is the first task to be scheduled* **then**
    If $t_{low} + t_s \leq d_i$, schedule task at lower voltage (speed); **continue**
    If $t_{hi} + t_s \leq d_i$, schedule task at higher voltage (speed); **continue**
    Call exception handler!!
  else
    if *previous speed = high speed* **then**
      Compute E_{hi} and E_{low}
      if *task is not schedulable* **then**
        Call exception handler!!
      else
        if *$t_{low} + t_s \leq d_i$*
          if *$E_{low} \leq E_{hi}$* **then**
            If $t_{low} + t_s \leq d_i$, schedule task at low speed
          else
            If $t_{high} + t_s \leq d_i$, schedule at high speed
        else
          Schedule task at high speed
    if *previous speed = low speed* **then**
      Compute E_{hi} and E_{low}
      if *task is not schedulable* **then**
        Call exception handler!!
      else
        if *$t_{hi} + t_s \leq d_i$* **then**
          if *$E_{hi} \leq E_{low}$* **then**
            Schedule task at high speed
          else
            If $t_{low} + t_s \leq d_i$, schedule task at low speed
end

Figure 2. Pseudocode for the E-LEDF algorithm.

speeds are dynamically switchable. We have also assumed that the CPU operates at 200 MHz at 3.3 V and at 100 MHz at 1.65 V. Furthermore, in order to emphasize energy minimization, the deadlines we have used are tighter than the actual deadlines for the actual task set. The minimum and maximum voltage-switching times t_s were chosen to be 1 unit and 5 units, respectively, and the minimum and maximum voltage-switching energies E_s were chosen to be 10 units and 200 units, respectively [10]. These numbers have been chosen because the DC-DC switching converter described in [10] has a switching time of less than 6 units. The LEDF and E-LEDF results are shown in Table 1.

Task set	Configuration	LEDF	E-LEDF	% increase
	$t_s = 5, v_s = 200$		213376.26	6.99 %
24 tasks	$t_s = 5, v_s = 10$	199418.76	212236.26	6.42 %
(Avionics)	$t_s = 1, v_s = 200$		201771.65	1.17 %
	$t_s = 1, v_s = 10$		200631.65	0.60 %
	$t_s = 5, v_s = 200$		380317.625	22.90 %
39 tasks	$t_s = 5, v_s = 10$	309338.65	378987.625	22.50 %
(ASW)	$t_s = 1, v_s = 200$		345221.25	11.59 %
	$t_s = 1, v_s = 10$		342751.25	10.80 %

Table 1. Results for two real-life task sets.

We see from Table 1 that voltage switching energies do not have as significant an effect as voltage switching times on the energy consumption of the task set. This is due to the fact that voltage switching energies do not affect the
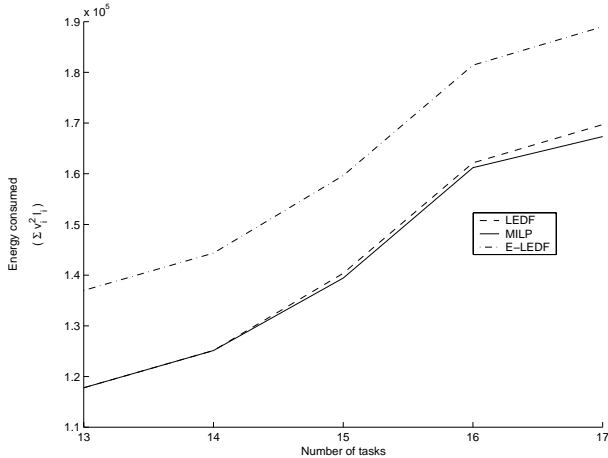
Figure 3. Energy consumptions of E-LEDF, LEDF, and MILP.

speed at which a task is executed. Any increase in energy is merely because of the increased energy consumption while switching. On the other hand, a change in voltage switching time increases energy consumption by as much as 6 % in the 24-task taskset and 12 % in the 39-task taskset. The reason for this is that the effective execution time of a task is increased by time t_s and any task that could run at a lower speed with a lower t_s now has to run at a higher speed in order to meet its deadline. This results in more tasks executing at a higher speed, and consequently a higher voltage, resulting in increased energy consumption.

The MILP model took prohibitively large amounts of time for scheduling task sets consisting of more than four tasks running on a dual-processor Sun Ultra-1 with a 256 MB memory capacity. For a five task data set, MILP took over a day to generate the optimal solution. On the other hand, E-LEDF took under a second to generate a near-optimal solution for the thirty-nine task example.

5 Conclusions

As embedded systems become more prolific, energy consumption is becoming an increasingly important design issue. Due to increased transistor counts, heat generation in embedded systems and system-on-a-chip designs is on the rise. This adversely affects both the reliability of the system, as well as its availability. Hence, the need for algorithms that attempt to minimize energy usage both at the system synthesis/design level, as well as the run-time/operating system level are being increasingly felt. In this paper, we have taken the latter approach to energy minimization. We have provided an algorithm that generates near-optimal solutions for task sets of varying size in polynomial time. We proved that the minimum-energy scheduling problem is NP-complete, and further showed that mixed-integer linear programming (MILP) is a tool that may be used for exact solution of moderate-sized problems. Finally, we applied an efficient heuristic algorithm to several representative real-time task sets, and showed that our algorithm provides near-optimal solutions.

References

[1] J. Anderson, University of North Carolina at Chapel Hill, private communication, March 2000.

[2] M. Berkelaar. *lpsolve*, version 3.0. Eindhoven University of Technology, Design Automation Section, Eindhoven, The Netherlands. *ftp://ftp.ics.ele.tue.nl/pub/lp_solve*

[3] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness,* Freeman and Co., New York, NY, 1979.

[4] I. Hong, D. Kirovski, G. Qu, M. Potkonjak and M. Srivastava, "Power optimization of variable-voltage core-based systems", *Proc. Design Automation Conf.,* pp. 176–181, 1998.

[5] Intel Pentium III Processor Brief, *http://www.intel.com*

[6] T. Ishihara and H. Yasuura, "Voltage scheduling problem for dynamically variable voltage processors", *Proc. Intl. Symp. on Low-Power Electronics Design,* pp. 197–202, 1998

[7] K. Jeffay, D. F. Stanat and C. U. Martel, "On non-preemptive scheduling of periodic and sporadic tasks with varying execution priority," *Proc. IEEE Real-Time Systems Symp.,* pp. 129–139, December 1991.

[8] E. L. Lawler and C. U. Martel, "Scheduling periodically occurring tasks on multiple processors", *Information Processing Letters,* vol. 12, no. 1, pp. 9–12, 1981.

[9] C. L. Liu and J. Layland, "Scheduling algorithms for multiprogramming in a hard real-time environment", *Journal of the ACM,* vol. 20, pp. 46–61, 1973.

[10] W. Namgoong, M. Yu and T. Meng, "A high-efficiency variable voltage CMOS dynamic DC-DC switching regulator," *IEEE Intl. Solid-State Circuits Conf.,* pp. 380–381, 1997.

[11] T. Okuma, T. Ishihara, and H. Yasuura, "Real-time task scheduling for a variable voltage processor," *Proc. 12th Intl. Symp. on System Synthesis,* pp. 24–29, 1999.

[12] J. H. Wensley, K. N. Levitt, M. W. Green, J. Goldberg and P. G. Neumann, " Design of a fault tolerant airborne digital computer. Volume 1: Architecture," Final Report, Stanford Research Institute, Menlo Park, CA, October 1973.

[13] J. M. Rabaey and M. Pedram, *Low Power Design Methodologies,* Kluwer Academic Publishers, Norwell, MA, 1996.

[14] V. Swaminathan and K. Chakrabarty, "Real-time task scheduling for energy-aware embedded systems", *to appear in Proc. IEEE Real-Time Systems Symp. (Work-in-Progress Session),* November 2000.

[15] H. P. Williams, *Model Building in Mathematical Programming,* 2nd ed., John Wiley, New York, NY, 1985.

[16] F. Yao, A. Demers and S. Shenker, "A scheduling model for reduced CPU energy", *Proc. IEEE Annual Foundations of Computer Science,* pp. 374–382, 1995.