Microprocessors and Microsystems 37 (2013) 99–112

Contents lists available at SciVerse ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

Energy-aware preemptive scheduling algorithm for sporadic tasks on DVS platform

Jing Mei [a], Kenli Li [a,*], Jingtong Hu [b], Shu Yin [a], Edwin H.-M. Sha [b,c]

[a] College of Information Science and Engineering, Hunan University, National Supercomputing Center in Changsha, HN 410082, China
[b] Department of Computer Science, University of Texas at Dallas, Richardson, TX 75080, USA
[c] College of Computer Science, Chongqing University, Chongqing, China

ARTICLE INFO

Article history:
Available online 11 December 2012

Keywords:
Preemptive scheduling
Sporadic tasks
Embedded systems
DVS
Energy conservation
Real-time
Uniprocessor

ABSTRACT

Dynamic Voltage Scaling (DVS) is a key technique for embedded systems to exploit multiple voltage and frequency levels to reduce energy consumption and to extend battery life. There are many DVS-based algorithms proposed for periodic and aperiodic task models. However, there are few algorithms that support the sporadic task model. Moreover, existing algorithms that support the sporadic model lack of functionalities in terms of energy-saving. In this paper, we propose a novel energy-aware scheduling algorithm named Cycle Conserve Dynamic Voltage Scaling for Sporadic Tasks (CC-DVSST) algorithm which is an improvement to DVSST [1]. There is a large amount of time slack in the DVSST scheduling due to the significant difference between the actual execution time and the worst-case scenario. Introducing DVS with EDF, CC-DVSST scales down the voltage of a processor when tasks are completed earlier than they are expected, so that the slack time can be reused for other tasks, hence saving energy. Experimental results show that CC-DVSST can reduce the total amount of energy consumption up to 46% compared to DVSST while retaining the quality of service by meeting the deadlines.

1. Introduction

Along with the development of embedded technologies, embedded equipments, such as MP3 players, GPS receivers, and smart phones, are becoming more and more complicated. A device with a same size consumes more energy than before due to its powerful functions. However, the improvement of battery technologies cannot meet the increasing demands of functionality. Therefore, reducing energy consumption has become a very important factor in the design and the implementation of embedded real-time systems. Dynamic Voltage Scaling (DVS) is a well-known technique to reduce energy consumption and to extend battery life for embedded systems.

Although there are a large number of studies on DVS-based real-time scheduling for energy conservation [1–14], most existing studies are proposed for periodic and aperiodic task models, yet few studies adopt the sporadic task model. Sporadic tasks are a kind of tasks that recur at random instants and represent an important part in real-time system applications.

Existing researches on sporadic tasks neither achieve optimal energy saving nor guarantee the deadlines. In this paper, we propose a novel algorithm CC-DVSST for sporadic task model. This algorithm combines DVS technique and preemptive EDF scheduling for real-time systems. Experimental results prove that CC-DVSST can achieve a better energy saving while meeting deadlines.

In real-time systems, each task is assigned with a time constraint, named *deadline* meaning that each task must be completed before its deadline, otherwise, unexpected errors could occur, which would lead to complete failure of the systems. In traditional scheduling, tasks are executed at full processor voltage to make sure that all tasks are completed before their deadlines; however, working at maximum voltage wastes processing resources because the processor becomes idle when tasks are completed earlier than their expected finish time. According to the relationship between energy consumption (E) and processor voltage (V) $E \propto V^2$ [15], we infer that energy consumption is the greatest at maximum voltage; hence, energy consumption can be reduced by lowering processor voltage. Chip makers such as Intel and AMD already produced commercial processors with the DVS capability which can dynamically scale voltage to reduce power consumption. However, as processor clock frequency is proportional to processor voltage, the clock frequency decreases hence the execution time of each clock cycles increases when the processor voltage decreases. Consequently, a critical problem of an embedded system to be considered is how to reduce energy consumption while the system can still meet deadlines.

The worst-case execution time (WCET) of tasks is used as *a priori* information to guarantee the schedulability of a task set in traditional static real-time scheduling methods; however, the actual execution time (AET) is not always equal to WCET. Experiments have shown a wide variation between the longest and the shortest

* Corresponding author.
E-mail address: jt_lkl@hnu.cn (K. Li).

0141-9331/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.micpro.2012.11.002

execution time for many existing applications; therefore, those schedules generated on the basis of WCET produce a large amount of slack time. This slack time can be exploited to lower the processor voltage so as to reduce the energy consumption.

In this paper, we present the *Cycle-Conserving Dynamic Voltage Scaling Algorithm for Sporadic Tasks* (CC-DVSST). It takes advantage of the difference between AET and WCET to scale the processor voltage in order to make the actual utilization of the given task set equal to 1 during arbitrary time intervals. Liu et al. [16] proposed the schedulability criterion of EDF scheduling: $U \leqslant 1$, where U is the sum of task utilization. In this paper, we prove that the schedulability criterion still holds when it is applied to CC-DVSST. Our contributions in this paper are as follows:

- We propose a novel CC-DVSST algorithm, which takes advantage of the difference between the actual execution time and the worst-case execution time to scale the processor voltage.
- We prove that the sporadic tasks scheduled by CC-DVSST can meet all of their deadlines.

The rest of the paper is organized as follows: In Section 2 related works about energy conservation are presented. In Section 3 we define the problem and describe the power model used in this paper. Section 4 introduces the CC-DVSST algorithm and provides an example to illustrate CC-DVSST in detail. In Section 5 we prove the schedulability of the CC-DVSST algorithm. Section 6 presents experimental results. In Section 7, practical implementation results are given. In Section 8, we discuss the overhead of voltage-switching. At last, we conclude our paper in Section 9.

2. Related work

Researches and development efforts on energy conservation have been proposed for real-time systems [1–4,8,17,9,18,11,13,7,16,19,6,5,20–22], which lays in three main topics: periodic task [23,24,17], aperiodic task[25], and sporadic task [1,18]. The major technique used for energy saving is the Dynamic Voltage Scaling (DVS). DVS means that processor voltage is dynamically scaled according to the demand of task execution. In most applications, AET of a task is less than its WCET. Hence, DVS technique can exploit the slack time to lower voltage so as to reduce energy consumption.

The major focus of existing researches focus on periodic task model under Rate Monotonic Scheduling (RMS) or Earliest Deadline First (EDF) scheduling. A general approach is to search CPU slack time based on WCET of task sets, and assign the slack time to tasks. Mosse et al. [21] proposed several algorithms about how to assign slack time including SPM algorithm and DPM-series algorithms. The SPM algorithm introduces a conservative estimate about slack time based on WCET, and it scales the same voltage for all tasks; while Mosse et al. [21] designed three kinds of DPM algorithms, naming DPM-P, DPM-G and DPM-S. They assign slack time to unfinished tasks using different strategies. These two algorithms produce suboptimal solutions. However, they cannot meet deadlines. Pallai et al. [5] proposed a Cycle-Conserving EDF (CCEDF) algorithm, which calculates the actual utilization after tasks completed and it scales voltage according to the actual utilization. According to the schedulability criterion of EDF algorithm, CCEDF can meet the deadlines for all tasks. Lee et al. [3] proposed an On-Line DVS (OLDVS) algorithm which scales voltage properly according to the difference between the usable time and the actual required time. The two latter algorithms are effective for periodic tasks, but they cannot obtain good performance when scheduling sporadic tasks.

Another popular research topic is scheduling and energy saving of real-time aperiodic tasks. Yao et al. [22] provided a preemptive offline real-time scaling algorithm for independent aperiodic tasks, which assumes that the task information has been known before tasks are released. Yao et al. [22] also proposed an Average Rate heuristic (AVR) algorithm which is an online scheduling algorithm. However, authors put efforts on energy savings without analyzing the schedulality of AVR. Mao et al. [8] proposed a Critical Task Decomposition (CTD) algorithm for aperiodic tasks, but its time complexity, which is $O(n^2)$, limits its usage in real-time environments.

Recently, researches on energy conservation of sporadic tasks have been presented [1,6,9,18]. In general, sporadic algorithms are more complicated than periodic algorithms, since periodic tasks release intervals are regular while sporadic tasks are not. In other words, more efforts needs to be paid on sporadic tasks scheduling, as more possible scenarios must be considered. Qadi et al. [1] presented a DVSST algorithm to schedule sporadic hard real-time tasks in conjunction with a preemptive EDF scheduling. The DVSST algorithm keeps trace of the total utilization U of all active sporadic tasks. When a sporadic task is released or the task's deadline arrives, the total utilization will be updated. In addition, the processor frequency is scaled according to the value of U. However, DVSST cannot lead to an optimal energy conservation for sporadic tasks, because that the DVSST algorithm updates U based on WCET, and makes no use of slack time produced by the variable execution time. Zhong et al. [9] proposed a Time-Variant Voltage Scaling (TV-DVS) algorithm, which is more energy efficient than that of DVSST; however, TV-DVS cannot meet the deadlines of tasks.

In this paper, we propose a novel algorithm CC-DVSST, which is an improvement of DVSST by making a full use of the difference between AET and WCET, and is able to scale to lower frequencies than DVSST. Moreover, CC-DVSST can be applied to both sporadic and periodic tasks. When CC-DVSST is applied to periodic tasks, it operates in a similar manner to a well-known periodic task scheduling algorithm proposed by Pallai et al. [5] named CCEDF; however, the CC-DVSST algorithm outperforms CCEDF in terms of sporadic tasks scheduling. The major reason is that the CC-DVSST algorithm scales voltage down when one period of each task finishes while CCEDF does not. Thus CC-DVSST reduces energy consumption more than CCEDF. We only compare the performance of our CC-DVSST algorithm with existing sporadic task scheduling algorithms due to the similarity manner between CC-DVSST and CCEDF for periodic tasks.

3. Models

In this section, we present the task set model and the power model discussed in this paper.

3.1. Task set model

We assume that the task set is sporadic and independent meaning tasks that recur at random instants. Let $T = \{T_1, T_2, \ldots, T_n\}$ be a set of sporadic tasks running on a simple real-time DVS enabled and uniprocessor system. Each task is represented by a three tuple $T_i = C_i, D_i, P_i$, where

- C_i is the worst-case execution clock cycles for task T_i. In this paper, we define that each clock cycle costs one unit time at the maximum processor voltage, hence C_i equals to the worst-case execution time (WCET).
- D_i is the relative deadline of the task T_i.
- P_i is the minimum interval between two executions of the task T_i. Each execution of a task is called a job noted J. For example, the job $J_{i,j}$ presents the jth execution of the task T_i.