Journal of Systems Architecture 61 (2015) 321–340

Contents lists available at ScienceDirect

Journal of Systems Architecture

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

Intra-task device scheduling for real-time embedded systems

Muhammad Ali Awan *, Stefan M. Petters

CISTER/INESC-TEC, ISEP, Polytechnic Institute of Porto, Rua Dr. António Bernardino de Almeida 431, 4200-072 Porto, Portugal

ARTICLE INFO

Article history:
Received 18 December 2012
Received in revised form 22 June 2015
Accepted 1 July 2015
Available online 6 July 2015

Keywords:
Device power management
Multiple sleep state devices
Multi-state devices
Real-time energy management
Intra-task device scheduling

ABSTRACT

An ever increasing need for extra functionality in a single embedded system demands for extra Input/Output (I/O) devices, which are usually connected externally and are expensive in terms of energy consumption. To reduce their energy consumption, these devices are equipped with power saving mechanisms. While I/O device scheduling for real-time (RT) systems with such power saving features has been studied in the past, the use of energy resources by these scheduling algorithms may be improved.

Technology enhancements in the semiconductor industry have allowed the hardware vendors to reduce the device transition and energy overheads. The decrease in overhead of sleep transitions has opened new opportunities to further reduce the device energy consumption. In this research effort, we propose an intra-task device scheduling algorithm for real-time systems that wakes up a device on demand and reduces its active time while ensuring system schedulability. This intra-task device scheduling algorithm is extended for devices with multiple sleep states to further minimise the overall device energy consumption of the system. The proposed algorithms have less complexity when compared to the conservative inter-task device scheduling algorithms. The system model used relaxes some of the assumptions commonly made in the state-of-the-art that restrict their practical relevance. Apart from the aforementioned advantages, the proposed algorithms are shown to demonstrate the substantial energy savings.

1. Introduction

Real-time embedded systems have to perform a set of functions while adhering to additional timing constraints. These systems interact with their environment and hence use I/O devices. Typical domains in which such systems are deployed includes avionics, automotive electronics, mobile phones and control systems. Besides the timing constraints, many RT systems have limited or intermittent power supply. Therefore, energy efficiency is another important aspect that needs to be considered in the design process of such systems.

The currently observed trend in the increased number of I/O devices can be attributed to the integration of previously isolated functionalities on a single chip. The energy consumption of CPUs have decreased considerably in modern embedded systems, while on the other hand, I/O devices are more power hungry and relative to CPUs consume large portion of the system's energy [1]. Therefore, I/O devices are of particular concern in mobile systems and provide opportunities to reduce the energy consumption of the system. Nowadays, they are often equipped with (a) power saving state(s) to minimise the overall energy consumption. Similar to CPUs, energy saving is achieved by turning-off certain parts of the device. For example, a hard-disk in an idle mode can be spun-down to reduce its energy consumption. A device can only operate in an active mode, and its transition into and out of a sleep state incurs both time and energy overheads. For instance, a hard-disk can only read/write in its active mode and it requires extra energy/time to spin-up from its power saving state. Concerning the schedulability, the request instant and access interval of the device usually cannot be determined beforehand. In order to guarantee the temporal correctness of such RT system, the device transition delay to bring the device up from a sleep state needs to be taken into account.

Considering the variability of the device usage instant and access interval, traditional device-scheduling algorithms made a safe but pessimistic decision that an active state of all the devices used by a task are ensured throughout its execution. This category of device scheduling is known as an inter-task device scheduling. However, in most cases, I/O devices are used for a very short duration. For instance, consider any image processing application on embedded platforms (face/thumb recognition) that reads a image and afterwards, the majority of processing is performed to extract the required features from the sampled data. Therefore, inter-task device scheduling wastes energy saving opportunities. In contrast to this, intra-task device scheduling makes a device wake-up call

* Corresponding author.
E-mail addresses: muaan@isep.ipp.pt (M.A. Awan), smp@isep.ipp.pt (S.M. Petters).

http://dx.doi.org/10.1016/j.sysarc.2015.07.001
1383-7621/

when it is requested by a task. To the best of our knowledge, online intra-task device scheduling for hard RT systems has not yet been explored beyond the direct precursor of this paper [2]. To ensure temporal correctness, the system has to compensate for the transition delays of a device requested on demand, as the corresponding task's execution is suspended until its associated device reaches an active state. Technological advancements have decreased the overheads associated with a device by several orders of magnitude, e.g. a solid state storage device has an extremely low overhead when compared to the conventional storages disks. Similarly, other devices are also optimised to reduce their overheads. Hence, it is worthwhile to explore this paradigm of intra-task device scheduling for RT device scheduling.

This article extends our previous work [2] of online intra-task device scheduling for hard RT systems considering a sporadic task model, to a model of a generic device with multiple sleep states. The proposed algorithms utilise the online and offline available spare processing capacities of the system to reduce the device energy consumption by allowing them to wake-up on demand such that it's always available during task execution. The contributions of this paper are the following. (1) We compute the spare capacity (slack) in the schedule to compensate for the on demand device transitions. A limit on the static slack is determined offline. Additionally, we propose an energy efficient online slack management algorithm to accumulate slack generated by the difference of worst-case execution time (WCET) and the actual execution time. (2) Based initially on a single sleep state per each device, an intra-task device scheduling algorithm is proposed that utilises the collated slack in the schedule and makes a device wake-up call on demand to enhance the energy performance. (3) This is complemented by an online device budget reclamation algorithm which recovers unused time allocations of devices in the system. This algorithm enhances the efficiency of the proposed intra-task device scheduling algorithm and allows to allocate a recovered slack resource to additional devices. (4) An energy density function is developed to analyse the effect of the different sleep states of a device on the overall device energy consumption of the system. This information in turn can be used to prioritise the devices and their sleep states competing for the different slack resources to compensate for their transitions. (5) The single sleep state assumption is eventually relaxed and we present three different algorithms for a generic power model, in which each device assumes more than one sleep states. (i) The first algorithm has no online overhead when compared to online algorithms, as the slack resources are allocated to the devices offline to compensate for their transition overheads. (ii) The second algorithm uses the online intra-task device scheduling algorithm. It prunes the inefficient sleep states and selects the single most efficient sleep state for each device offline. (iii) The third algorithm prioritises the sleep states offline but selects the most energy saving sleep state online based on the given slack resource capacity at that moment. (6) The complexity of the proposed algorithms is compared with those in the state-of-the-art. (7) Finally, extensive simulations results are presented to demonstrate the effectiveness and scalability of our algorithms. The contributions 3, 4, 5, 6 and 7 are over and beyond the original work [2].

The rest of the paper is organised as follows. The next section discusses the related work and is followed by our system model. In Section 4, we revisit the spare slack resources available in the schedule and propose different techniques to collate them. The online intra-task device scheduling algorithm "Static Slack Container algorithm" (SSC) is proposed in Section 5 followed by our device budget reclamation mechanism given in Section 6. Section 7 extends SSC for the multiple sleep state devices. The complexity of state-of-the-art with our algorithms is compared in Section 8. Finally, we present the evaluation and conclude with future directions.

2. Related work

Initially device scheduling in the context of RT systems has been explored by Swaminathan et al. [3]. They proposed an offline method for dynamic I/O power management with hard RT constraints. Their low energy device scheduler (LEDES) is based on look-ahead information about the tasks future arrival-pattern to decide on the shut-down of devices. Later on, multi-state constrained low-energy scheduler (MUSCLES), an extension of LEDES for the multiple sleep state devices was proposed by Swaminathan and Chakrabarty [4]. MUSCLES generates the sequence of power states for every device given the precomputed task schedule with a per task device usage list. The LEDES and MUSCLES algorithms assume fixed tasks releases, which limits its applicability/extension to a sporadic task model and/or to a task model that allows variable task's execution time. Our proposed algorithms relax these assumptions.

The same authors also developed energy optimal device scheduler (EDS) [5]. EDS computes a schedule tree for all possible scheduled combination, and prune it based on the temporal and energy constraints. Due to high spatial requirement and temporal complexity of EDS, they provide a maximum device overlap heuristic (MDO) which clusters the requests of the same device to prolong the idle intervals. The maximum device overlap heuristic is adapted from the work of Lu et al. [6] that was proposed for best-effort systems. Both MDO and EDS are based on an inter-task scheduling mechanism. They are computationally expensive and are of limited utility for sporadic task models as they assume a priori information of a task's release pattern.

A procrastination based I/O device scheduling algorithm is proposed by Cheng and Goddard [7]. The basic idea is to prolong the device's sleep interval by procrastination of the task's execution that requires this device. This method assumes inter-task device scheduling and has high online overhead. However, it can be applied to a sporadic task model with tasks having varying execution times. Later, Devadas and Aydin [8] proposed the device power management algorithm for static priority systems through device forbidden regions. The device forbidden regions enforces idle intervals in the schedule to prolong the sleep interval of devices. To preserve the schedulability, the bounds on the explicit idle intervals are computed using time bound analysis [9]. This algorithm also exploits inter-task device scheduling.

Chu et al. [10] proposed a composite low-power scheduling framework called COLORS. COLORS is a Dynamic Voltage Scaling (DVS) assisted I/O device scheduling algorithm for periodic hard RT systems. They assume devices access intervals and their usage times are known a priori. The execution of the task is divided into computation and peripheral intervals. It uses both static and dynamic slack to extend the computation interval of a task by running it at low frequency to prolong the device shut-down time. A simplistic power-model and a priori device usage information makes it useful only for predictive systems. A similar slot-based algorithm was proposed by Kim and Ha [11]. However, they did not consider the transition overheads. A genetic algorithm customised for the device power management is proposed by Tian and Arslan [12] for periodic RT systems. This algorithm assumes jobs execute for their WCET and try to find the near-optimal solution with the provided set of jobs and devices.

The low-power quasi-dynamic scheduling (LQS) proposed by Hsiung and Kao [13] determines the feasible schedule to reduce the device power consumption. The system is modelled with power-aware real-time petri-nets (PARTPN). LQS uses the