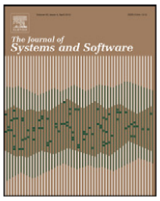
The Journal of Systems and Software 85 (2012) 1386–1399

Contents lists available at SciVerse ScienceDirect

The Journal of Systems and Software

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



Quasi-static fault-tolerant scheduling schemes for energy-efficient hard real-time systems[☆]

Tongquan Wei[a,*,1], Piyush Mishra[b,1], Kaijie Wu[c,1], Junlong Zhou[a]

[a] CS Department of East China Normal University, Shanghai 200241, China
[b] GE Global Research, Niskayuna, NY 12309, USA
[c] ECE Department of University of Illinois, Chicago, IL 60607, USA

ARTICLE INFO

Article history:
Received 13 August 2011
Received in revised form 7 January 2012
Accepted 7 January 2012
Available online 28 January 2012

Keywords:
Energy-efficient dynamic scheduling
Dynamic voltage scaling (DVS)
Fault tolerance
Hard real-time embedded systems

ABSTRACT

This paper investigates fault tolerance and dynamic voltage scaling (DVS) in hard real-time systems. The authors present quasi-static task scheduling algorithms that consist of offline components and online components. The offline components are designed the way they enable the online components to achieve energy savings by using the dynamic slack due to variations in task execution times and uncertainties in fault occurrences. The proposed schemes utilize a fault model that considers the effects of voltage scaling on transient fault rate. Simulation results based on real-life task sets and processor data sheets show that the proposed scheduling schemes achieve energy savings of up to 50% over the state-of-art low-energy offline scheduling techniques and incur negligible runtime overheads. A hard real-time real-life test bed has been developed allowing the validation of the proposed algorithms.

1. Introduction

The number of faults in hardware, particularly the transient faults, has been rising continuously due to the increasing complexity of design, aggressive technology scaling, and extreme operating conditions. For example, the increasing integration level of transistors, reducing feature sizes, and lowering voltage levels are making the integrated circuits highly susceptible to radiation-induced bit-flips. In addition, high-energy particles, such as neutrons from cosmic radiation, are able to introduce transient faults in electronic systems (Normand, 1996). On the other hand, a growing number of complex safety critical applications operate under extreme conditions and demand ultra-reliability and high performance. For example, hard real-time systems deployed in navigation, process control, and system surveillance require the high fault tolerance without sacrificing the feasibility of task sets. The need for reliability is rising even in non-critical applications which are prone to operate in harsher environments but have lower expectancy of failures. Common examples include outdoor sensor networks and massive communication infrastructure deployed in fields which suffer frequent physical abuse and are often exposed to strong radiation.

Numerous fault-tolerance techniques have been proposed for real-time systems (Shin and Lee, 1984; Shin et al., 1987; Kwak et al., 2001; Axer et al., 2011; Huang et al., 2011). One of the typically used fault-tolerance techniques is online concurrent fault detection followed by a hardware-based checkpointing and rollback recovery mechanism. It allows processors to rollback to the previously known valid states to resume normal executions by exploiting the slack time available in task schedules. Traditionally, fault tolerance techniques aim to maximize the fault coverage and minimize the fault detection latency and associated redundancy costs (Pradhan, 1986). The costs are usually measured in terms of hardware, time, or information overhead and are of great significance in real-time embedded systems due to their severe resource constraints.

Owing to the fast-evolving application of real-time systems in battery-powered portable devices, energy has emerged as another important design constraint. Dynamic power management is an active area of research and several techniques have been proposed to minimize energy consumption at the system level (Benini et al., 2000). The energy efficiency is achieved by dynamically reconfiguring active system components and selectively turning off system components when they are idle. Dynamic voltage scaling (DVS) is a widely used system level power management technique that exploits technological advances in power supply circuits to reduce the energy consumption. It reduces the processor power consumption by dynamically scaling down the processor supply voltage at the cost of the increased execution times.

Fault tolerance and energy have been jointly investigated in the literature. On one hand, with the continuous shrinking of the

[☆] The preliminary version of this manuscript appeared in ICCAD, 2006.
[*] Corresponding author.
E-mail address: tqwei@cs.ecnu.edu.cn (T. Wei).
[1] Member, IEEE.

0164-1212/$ – see front matter © 2012 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2012.01.020

feature size and reducing of voltage margins, it is expected that all digital computing systems will be remarkably vulnerable to transient faults (Ernst et al., 2004). Fault-tolerance in real-time systems is typically achieved by using some forms of redundancy. This redundancy causes extra power dissipations and hence necessitates energy efficient schemes for batter-powered real-time systems to reduce heat dissipation and extend operational lifetime. On the other hand, energy management through dynamic voltage scaling has adverse effects on system reliability. Scaling down the supply voltage results in an increase in the rate of transient faults (Zhu et al., 2004). Consequently, both fault-tolerance and energy efficiency have been the primary design goals for real-time systems, integrated in the design process at all levels for joint optimization with the system feasibility.

In the recent past, considerable attention has been paid to exploit the DVS technique to achieve energy savings in the presence of transient faults and a number of excellent energy-efficient fault-tolerance schemes have been designed for real-time embedded systems (Zhu et al., 2004; Zhang et al., 2003; Melhem et al., 2004; Wei et al., 2006; Zhang and Chakrabarty, 2006; Zhao et al., 2009, 2011; Iqbal et al., 2011). In this paper a systematic approach is proposed to derive energy-efficient fault-tolerant task schedules for hard real-time embedded systems by utilizing both the static and dynamic slack in the task schedules. Based on the observation that the probability of the single event upset (SEU)-induced faults remains low in the foreseeable future and fault-free condition will continue to dominate (Reed et al., 2006; Weulersse et al., 2006; Langley et al., 2003), one fundamental innovation has been introduced in the proposed energy-efficient fault-tolerance schemes. That is, unlike the traditional approach that focuses on designing offline energy-efficient fault-tolerance algorithms, the proposed scheme aims to achieve further round of energy savings by designing efficient offline algorithms that enable the adaptation of the offline schedules to the runtime behavior of fault occurrences.

1.1. Related work

Extensive research has been performed to investigate the energy efficiency of real-time systems from both offline and online perspective (Shin and Choi, 1999; Gruian, 2001; Pillai and Shin, 2001; Saewong and Rajkumar, 2003; Krishna and Lee, 2003; Mochocki et al., 2007; Huang et al., 2009, 2011; Perathoner et al., 2010). The power efficient version of fixed-priority preemptive scheduling such as rate monotonic scheduling was explored by Shin and Choi (1999). Power reduction is achieved by exploiting the slack time both inherent in system schedule and due to runtime variations in task execution time. Similarly, Gruian (2001) presented a scheduling policy for hard real-time tasks with fixed priorities assigned in a rate monotonic manner. The offline scheduling uses exact timing analysis to derive multiple voltage scaling factors for each task based on stochastic characteristics of task execution time. The online scheduling policy distributes available slack time on priority basis. Based on the voltage scaling algorithms proposed in Pillai and Shin (2001), four voltage scaling algorithms including Sys-Clock, PM-Clock, and DPM-Clock were proposed in Saewong and Rajkumar (2003) for different hardware which may have high or low voltage scaling overhead and different taskset characteristics. Of these algorithms, Sys-Clock assigns a single frequency to all tasks in a task set, PM-Clock assign multiple frequencies to tasks in a task set, and DPM-Clock dynamically adapts offline task schedule to runtime behaviors of task execution times. Krishna and Lee (2003) described a two phase heuristic for independent and periodic tasks. The heuristic has an offline component computing a voltage schedule based on worst case execution time, and an online component utilizing slack time due to variations in task execution time for further round of energy savings. In Mochocki et al. (2007), both offline and online scheduling schemes were proposed to handle the transition time and energy overhead of DVS processors. The offline scheme generates task schedule during design time based on a prior known task execution time while the online scheme effectively accommodates runtime variations of task execution time to achieve energy savings. Online algorithms were presented in Huang et al. (2009, 2011) to effectively reduce system energy consumption to handle event streams with hard real-time guarantees. The scheduling scheme adaptively controls the power mode of the processor to postpone the processing of arrival events as late as possible. Although energy efficiency in real-time systems were explored from both offline and online aspects in the above literature, fault tolerance which is an another important design constraints were not considered.

Fault-tolerance is another important design constraint in energy efficient real-time systems. Joint optimization of energy and fault-tolerance in real-time embedded systems has attracted considerable attention in the past decade (Zhang et al., 2003; Melhem et al., 2004; Zhang and Chakrabarty, 2006; Zhao et al., 2009, 2011; Iqbal et al., 2011; Wei et al., 2011). Melhem et al. (2004) proposed DVS techniques to exploit slacks in a task schedule to reduce energy consumption while tolerating faults during task execution. A task in the task schedule is assumed to be susceptible to at most one fault occurrence and the processor can scale its frequency in a continuous range. In Zhang et al. (2003) and Zhang and Chakrabarty (2006) a fixed priority offline scheduling scheme was proposed based on the rate monotonic scheduling to tolerate faults in hard real-time systems. Practical design issues such as checkpointing cost and voltage switching overhead are considered. Fault-tolerance scheduling techniques were developed in Zhao et al. (2009, 2011) to minimize the system-level energy consumption while still preserving the systems original reliability. Fault tolerance is achieved by reserving shared recovery blocks that can be used by any task at the runtime. In Iqbal et al. (2011), the authors presented a soft error aware energy efficient scheduling technique for soft real-time systems with stochastic task execution times. The task execution time estimation is modeled as a joint state-space model, the solution of which is found by an online Monte Carlo sampling based recursive technique.

An offline reliability-aware power management scheme is presented in Zhu et al. (2008) for real-time tasks with probabilistic execution times. The scheme puts aside just enough slack to guarantee the required reliability while leaving more slack for energy management to achieve better energy savings. In Pop et al. (2007), the authors addressed the scheduling and voltage scaling for hard real-time applications that have been statically mapped on heterogeneous distributed embedded systems. Tasks in a given task set are assumed to share a common deadline and the effect of voltage scaling on system reliability is taken into account. Shafik et al. (2010) examined the impact of application task mapping on the reliability of MPSoC. The number of transient faults is minimized without compromising the timeliness of the system. All these energy-aware fault-tolerance schemes, however, statically derive offline task schedules to guarantee hard timing constraints, hence are conservative and cannot utilize the dynamic slack due to variations in task execution times and uncertainties in fault occurrences for further energy savings.

Zhang and Chakrabarty (2003) developed an online scheduling algorithm that combines checkpointing with DVS to tolerate faults in real-time uni-processor systems with periodic tasks. However, this scheme cannot handle hard real-time task scheduling. In Izosimov et al. (2008), the authors present an approach to the synthesis of fault-tolerant schedules for embedded applications with soft and hard real-time constraints. A set of task schedules is synthesized offline and, at run time, the scheduler selects the right schedule based on the fault occurrence and the actual task