Accepted Manuscript

Slack clustering for scheduling frame-based tasks on multicore embedded systems

Farimah Poursafaei, Mostafa Bazzaz, Morteza Mohajjel Kafshdooz, Alireza Ejlali

PII: S0026-2692(17)30895-9

DOI: 10.1016/j.mejo.2018.09.002

Reference: MEJ 4413

To appear in: *Microelectronics Journal*

Received Date: 18 November 2017

Revised Date: 6 August 2018

Accepted Date: 10 September 2018

Please cite this article as: F. Poursafaei, M. Bazzaz, M.M. Kafshdooz, A. Ejlali, Slack clustering for scheduling frame-based tasks on multicore embedded systems, *Microelectronics Journal* (2018), doi: https://doi.org/10.1016/j.mejo.2018.09.002.

Slack Clustering for Scheduling Frame-Based Tasks on Multicore Embedded Systems

Farimah Poursafaei, Mostafa Bazzaz, Morteza Mohajjel Kafshdooz, and Alireza Ejlali

Abstract—Adopting multicore platforms is a general trend in real-time embedded systems. However, integrating tasks with different real-time constraints into a single platform presents new design challenges. While it must be guaranteed that hard real-time tasks are able to meet their deadline even in worst case scenarios, firm real-time tasks should be scheduled in a way to achieve high system utilization in order to provide better quality of service. In this paper, we propose a scheduling scheme for frame-based tasks on real-time multicore embedded systems which is able to guarantee the schedulability of the hard real-time tasks, while it improves the number of executed firm real-time tasks. Considering intrinsic variation of execution time of tasks and with the help of a clustering algorithm, hard real-time tasks get scheduled in a way that their expected slack time could be exploited more efficiently for the execution of firm tasks. The extensive simulations show that our proposed scheme can improve the system utility up to 60 percent compared to a state-of-the-art scheduling technique.

Index Terms—Execution time variation, frame-based tasks, hard real-time, firm real-time, clustering.

1 Introduction

Real-time embedded systems exhibit an interesting trend to integrate many different tasks onto fewer hardware platforms. This trend is a result of the intrinsic constraints of these systems in terms of cost, weight, size, and power. Therefore, in many cases multiple tasks with different importance, safety, or real-time requirements are executing on a single platform in modern real-time systems [1], [2], [3], [4], [5], [6]. The examples of such systems where tasks with different importance levels are integrated on the same platform are prevalent in the domain of avionics and automotive systems [7]. In these systems, two important objectives often exist: (i) satisfying conservative assumptions for the *hard* real-time applications, and (ii) achieving high resource utilization when *firm* real-time applications are running and the actual behavior does not follow the pessimistic expectations that were satisfied during design time analyses [8].

Contrary to hard real-time applications in which missing the deadline results in catastrophic consequences, firm real-time applications have flexible timing requirements where missing the deadline usually leads to degradation of performance or quality of service [9]. Even though the design of hard real-time system is based on worst-case scenario, using these assumptions would be inappropriate for firm real-time systems and may result in resource underutilization. The system utilization can be viewed as the sum of utility values which depend on the completion time of each task. For a hard task where the deadline miss may jeopardize the behavior of the whole system, the utility value is assumed to be minus infinity after the deadline. On the other hand, a firm real-time task can be considered zero utility value if completed after the deadline, since its late completion does not jeopardize the system [9]. Therefore, in systems with hard and firm tasks, not only the completion of hard tasks before their deadlines must be guaranteed in order to prevent catastrophic consequences, but also is it very desirable to accomplish as many firm tasks as possible. To satisfy the critical requirements of hard tasks, the systems are completely and pessimistically validated at design time, however they face severe underutilization at run time. For example, next generation unmanned aerial vehicles have workloads that consist of tasks from different criticality levels. On one hand, the workload consists of tasks that adjust flight surfaces or are responsible to answer to threats. These tasks are *safety-critical*. On the other hand, there exist tasks responsible for decision making and communication which are identified as *mission-critical* tasks. Moreover, some tasks are specified as *best-effort* that do not have strict timing requirements but need high computational capacity in oder to perform surveillance and route mapping [10].

Timing predictability is a very important factor for hard real-time tasks. A task's upper bound on execution time, which is known as *Worst-Case Execution Time (WCET)*, must be carefully estimated to guarantee the satisfaction of timing constraints [11]. It is generally observed that the pessimistic WCET estimation required for satisfaction of stringent timing constraints of hard tasks causes system underutilization. Hence, the high criticality requirements of hard real-time tasks can be alleviated for better utilization when executing firm real-time tasks [1], [4], [12]. It should be noted that the actual execution time of a task is usually much less than its WCET, and only in very rare situation the execution of a task matches its WCET. This is because in WCET estimation a more pessimistic approach is used in order to increase the reliability of the system [13]. Typically, a certain variation of execution time is shown by each task which depends on input data or environmental behavior [11]. The set of all execution times of each task can be demonstrated by a function, where the minimum and maximum value are called

F. Poursafaei, M. Bazzaz, M. M. Kafshdooz, and A. Ejlali are with the Department of Computer Engineering, Sharif University of Technology, Tehran, Iran. Email:{poursafaei, bazzaz, mohajjel}@ce.sharif.edu, ejlali@sharif.edu