TECHNICAL UNIVERSITY OF VALENCIA

Abstract

School of Computer Engineering
Department of Computer Engineering

Doctor of Philosophy
(Computer Engineering)

**Dynamic Power-Aware Techniques for
Real-Time Multicore Embedded Systems**

by José Luis March Cabrelles

The continuous shrink of transistor sizes has allowed more complex and powerful devices to be implemented in the same area, which provides new capabilities and functionalities. However, this complexity increase comes with a considerable rise in power consumption. This situation is critical in portable devices where the energy budget is limited and, hence, battery lifetime defines the usefulness of the system. Therefore, power consumption has become a major concern in the design of real-time multicore embedded systems.

This dissertation proposes several techniques aimed to save energy without sacrifying real-time schedulability in this type of systems. The proposed techniques deal with different main components of the system. In particular, the techniques affect the task partitioner and the scheduler, as well as the memory controller.

Some of the techniques are especially tailored for multicores with shared *Dynamic Voltage and Frequency Scaling* (DVFS) domains. Workload balancing among cores in a given domain has a strong impact on power consumption, since all the cores sharing a DVFS domain must run at the speed required by the most loaded core.

In this thesis, a novel **workload partitioning** algorithm is proposed, namely *Load-bounded Resource Balancing* (*LRB*). The proposal allocates tasks to cores to balance a given resource (processor or memory) consumption among cores, improving real-time schedulability by increasing overlapping between processor and memory. However, distributing tasks in this way regardless the individual core utilizations could lead to unfair load distributions. That is, one of the cores could become much loaded than the others. To avoid this scenario, when a given utilization threshold is exceeded, tasks are assigned to the least loaded core.

Unfortunately, workload partitioning alone is sometimes not able to achieve a good workload balance among cores. Therefore, this work also explores novel **task migration** approaches. Two task migration heuristics are proposed. The first heuristic, referred to as *Single Option Migration* (*SOM*), attempts to perform only one migration when the workload changes to improve utilization balance. Three variants of the *SOM* algorithm have been devised, depending on the point of time the migration attempt is performed: when a task arrives to the system (SOM_{in}), when a task leaves the system (SOM_{out}), and in both cases (SOM_{in-out}). The second heuristic, referred to as *Multiple Option Migration* (*MOM*) explores an additional alternative workload partitioning before performing the migration attempt.

Regarding the memory controller, **memory controller scheduling policies** are devised. Conventional policies used in *Non Real-Time* (NRT) systems are not appropriate for systems providing support for both *Hard Real-Time* (HRT) and *Soft Real-Time* (SRT) tasks. Those policies can introduce variability in the latencies of the memory requests and, hence, cause an HRT deadline miss that could lead to a critical failure of the real-time system. To deal with this drawback, a simple policy, referred to as *HR-first*, which prioritizes requests of HRT tasks, is proposed. In addition, a more advanced approach, namely *ATR-first*, is presented. *ATR-first* prioritizes only those requests of HRT tasks that are necessary to ensure real-time schedulability, improving the *Quality of Service* (QoS) of SRT tasks.

Finally, this thesis also tackles **dynamic execution time estimation**. The accuracy of this estimation is important to avoid deadline misses of HRT tasks but also to increase QoS in SRT systems. Besides, it can also help to improve the schedulability of the systems and reduce power consumption. The *Processor-Memory* (*Proc-Mem*) model, that dynamically predicts the execution time of real-time application for each frequency level, is proposed. This model measures at the first hyperperiod, making use of *Performance Monitoring Counters* (PMCs) at run-time, the portion of time that each core is performing computation (*CPU*), waiting for memory (*MEM*), or both (*OVERLAP*). This information will be used to estimate the execution time at any other working frequency.