IPP HURRAY!

www.hurray.isep.ipp.pt

Technical Report

Comparing the Schedulers and Power Saving Strategies with SPARTS

Muhammad Ali Awan

Borislav Nikolic

Stefan M. Petters

HURRAY-TR-111104

Version:

Date: 11/23/2011

Comparing the Schedulers and Power Saving Strategies with SPARTS

Muhammad Ali Awan, Borislav Nikolic, Stefan M. Petters

IPP-HURRAY!

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8340509

E-mail: maan@isep.ipp.pt, borni@isep.ipp.pt, smp@isep.ipp.pt

http://www.hurray.isep.ipp.pt

Abstract

We have developed SPARTS, a simulator of a generic embedded real-time device. It is designed to be extensible to accommodate different task properties, scheduling algorithms and/or hardware models for the wide variety of applications. SPARTS was developed to help the community investigate the behaviour of the real-time embedded systems and to quantify the associated constraints/overheads.

Comparing the Schedulers and Power Saving Strategies with SPARTS

Muhammad Ali Awan Borislav Nikolić Stefan M. Petters
Cister Research Unit, ISEP-IPP, Porto, Portugal
{maan,borni,smp}@isep.ipp.pt

***Abstract*—We have developed SPARTS, a simulator of a generic embedded real-time device. It is designed to be extensible to accommodate different task properties, scheduling algorithms and/or hardware models for the wide variety of applications. SPARTS was developed to help the community investigate the behaviour of the real-time embedded systems and to quantify the associated constraints/overheads.**

I. Introduction

Nowadays, there is an ever increasing demand for embedded systems. These devices interact with their environment and perform computations which may have to satisfy several constraints. Embedded systems with timing requirements are called Real-Time embedded Systems. In addition to timing constraints and function correctness, these devices often have limited or intermittent power supply. Technology enhancements and cost reduction mandated integration of the multiple functionalities into one device. Along with this, the design itself grew more complex. For instance, in modern cars safety critical applications are integrated with comfort functionality.

In order to provide efficient execution of such systems, specific scheduling algorithms are needed. They are very diverse in terms of requirements and produced results. Given those, there is the question of evaluating the scheduling approaches and identifying the trade-offs involved in employing one strategy over another. One approach in tackling this problem is to build a system model. It gives the possibility to hide unnecessary and negligible details, so only aspects of interest need to be implemented and tested. This method is very fast and can provide the comparison between different scheduling approaches or associated power management schemes. We have developed SPARTS (Simulator for Power-Aware Real-Time Systems) that can model aforementioned different system behaviour aspects.

Numerous available tools target different areas of stated problems. However, several limitations are recognised as the greatest drawbacks of their use. Some simulators [1]–[3] utilise an unnecessarily detailed approach in simulation which has a huge impact on the performance. Others lack in expressiveness while defining new functionalities and/or adapting existing ones for personal needs, e.g. [4]. Furthermore, commercial tools such as SymTA/S [5] or RapiTime [6] are for obvious reasons closed systems not accessible to casual users.

This work was supported by the RePoMuC project, ref. FCOMP-01-0124-FEDER-015050, funded by FEDER funds through COMPETE (POFC - Operational Programme 'Thematic Factors of Competitiveness) and by National Funds (PT) through FCT-Portuguese Foundation for Science and Technology, and the RECOMP project, funded through the FCT under grant ref. ARTEMIS/0202/2009, as well as by the ARTEMIS Joint Undertaking, under grant agreement Nr. 10202.

II. SPARTS Overview

SPARTS simulates a generic real-time device and is implemented as a discrete-event execution environment. It provides extensive flexibility in task-set generation for different scenarios and purposes. The task-sets can be used for schedulability tests as well as for simulation purposes. The modular structure of SPARTS allows easy development and integration of new scheduling algorithms for both, single and multi-core systems. The results of the simulations give indications about the performance and various overheads incurred by different scheduling approaches (pre-emptions, energy consumptions, migrations for multi-cores, etc). SPARTS can be extended and adapted to fit the needs of the user in the area of interest.

SPARTS performs the simulation in event-driven manner. Rather than doing a cycle-step execution, SPARTS works by looking backward into the interval between two consecutive job releases and simulates the execution without unnecessary cycle-level granularity. This feature is implemented with a *Timers* mechanism. Timers represent possible interrupts of the execution process, such as, new arrival of a job, expiration of a deadline, completion of an execution etc. The simulation jumps to the point in time when the first timer would fire. At this moment possible actions corresponding to the interrupts and the intervening time like power attribution are serviced.

The architecture of SPARTS is depicted on Figure 1. For easier use and extensibility, we separated the responsibilities by encapsulating limited functionalities within different modules and providing the interfaces for them to communicate. The input parameters are delivered to the Task-Set Generator (TSG), which creates the tasks for a particular simulation run. Generated task-set is then passed to Job Generator (JG) and job instances for the desired simulation time are produced. The Job Sequencer (JS) orders the jobs by their release times and prepares the stream for the execution. Finally, the Execution Environment (EE) executes the stream of jobs and collects required parameters for the reporting tool to do further analyses.

Several performance bottlenecks recognised in related work are addressed in SPARTS. Firstly, the simulation process is driven by discrete events and executes by looking into the past. With this approach we save the computation and yet provide ”correct” execution modelling. This allows to perform the simulations of large task-sets for long periods of time with high temporal efficiency. Additionally, this gives the possibility to manually configure the granularity of the system and adapt it to the particular needs of the user. Secondly, SPARTS breaks the simulation time into smaller pieces, which

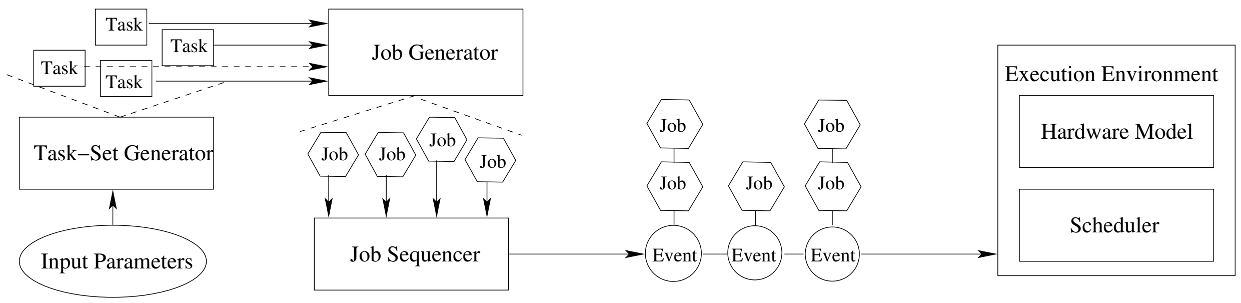

Fig. 1. The Simulator Architecture

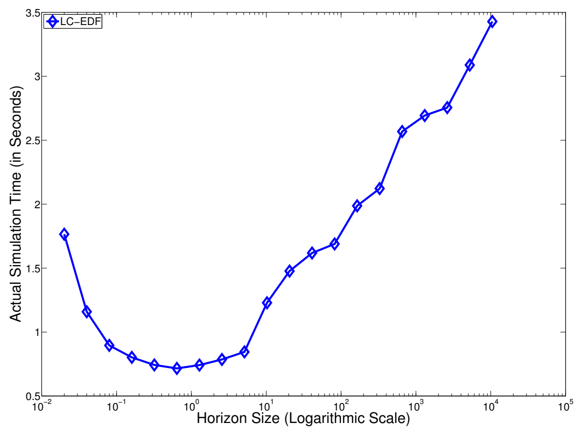

Fig. 2. Horizon size trade-off

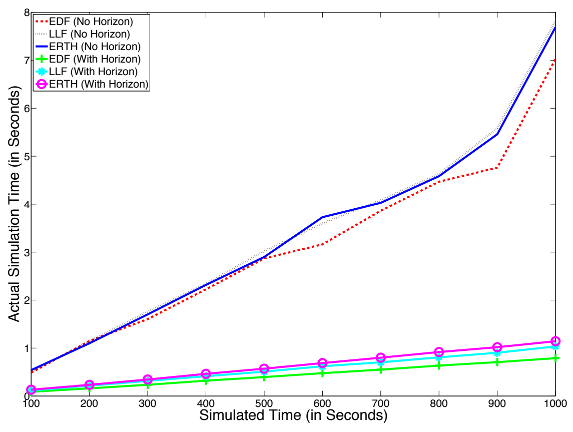

Fig. 3. Simulation time vs actual simulated time

are sequentially, on-demand generated and utilised. By doing this, the costly manipulations of the event queues are reduced and the memory footprint of the simulation is significantly smaller with a positive impact on the caching footprint of the simulator. This partial simulation window is denoted as the *horizon*. It is a configurable parameter and its utilisation gives dramatically better performance than the system which generates the streams for the whole simulated period at once.

In order to check the performance of SPARTS, we performed several experiments in which we varied parameters such as task-set size, total system utilisation and simulated time for different scheduling algorithms. Several scheduling algorithms are implemented and used in the experiments (Earliest Deadline First (EDF), Least Laxity First (LLF) and Enhanced Race-To-Halt [7]). The effectiveness of the *horizon* feature is demonstrated in Figure 2 and Figure 3. Figure 2 illustrates that small time-windows bring unnecessary overheads. The extension of the horizon gives better results, but from one point onwards, the simulation time again starts to increase, due to very large lists and queues as a direct result of the horizon length. We found the value that gives the best results for this particular set-up and we used the same in the Figure 3 and other experiments. Figure 3 shows that the same scheduling algorithms with this feature exhibit linear dependency on the simulated time, while the execution time of the conventional approach grows exponentially.

When compared against reported results of aforementioned simulators, given in respective papers, we found that the ratio between simulated time and actual simulation time of SPARTS is several orders of magnitude greater. The simulator is accesible for download on its website [8]. Guidelines for the implementations of new algorithms and further simulator extensions will be presented during the demonstration session. Also, the pictorial view of the schedules generated for different schedulers will be presented.

III. Conclusions

SPARTS is under active development and is being extended depending on concrete analysis needs. The features presented here are only a small subset and the reader is directed for a more detailed description of the other experiments and the simulator in general to a full paper [9] and to the simulator website [8].

References

[1] S. De Vroey, J. Goossens, and C. Hernalsteen, "A generic simulator of real-time scheduling algorithms," in *29th Simul. Symp. 1996*, pp. 242 –249, Apr 1996. 1

[2] T. Kramp, M. Adrian, and R. Koster, "An open framework for real-time scheduling simulation," in *Int. WS Parall. & Distr. Processing*, pp. 766–772, 2000. 1

[3] R. Urunuela, A. Deplanche, and Y. Trinquet, "Storm a simulation tool for real-time multiprocessor scheduling evaluation," in *Emerging Technologies & Factory Automation (ETFA), 2010 IEEE Conf.*, pp. 1 –8, Sep 2010. 1

[4] F. Singhoff, J. Legrand, L. Nana, and L. Marcé, "Cheddar: a flexible real time scheduling framework," in *ACM SIGAda international conference*, (New York, NY, USA), pp. 1–8, ACM, 2004. 1

[5] "Symta/s, Symtavision GmbH." http://www.symtavision.com/symtas.html. 1

[6] "Rapitime, Rapita Systems Ltd.." http://www.rapitasystems.com/products/RapiTime. 1

[7] M. A. Awan and S. M. Petters, "Enhanced race-to-halt: A leakage-aware energy management approach for dynamic priority systems," in *23rd ECRTS*, 2011. 2

[8] B. Nikolic, M. A. Awan, and S. M. Petters, "Simulator for power aware and real-time systems: Sparts," 2011. http://www.cister.isep.ipp.pt/projects/sparts/. 2

[9] B. Nikolic, M. A. Awan, and S. M. Petters, "SPARTS: Simulator for power aware and real-time systems," in *8th IEEE Int. Conf. Emb. Softw. & Syst.*, (Changsha, China), IEEE, Nov 2011. 2