Scheduling and Power Management Facilities in Linux/RTAI for Real-time Systems powered by Ambient Energy Source

Maryline Chetto, Audrey Queudet

RenPar'20 / SympA'14 / **CFSE 8**
Saint-Malo, France, du 10 au 13 mai 2011

Scheduling and Power Management Facilities in Linux/RTAI for Real-time Systems powered by Ambient Energy Source

Maryline Chetto and Audrey Queudet*

IRCCyN - UMR CNRS 6597
IUT de Nantes, 2 Avenue du Professeur Rouxel
44475 Carquefou - France
maryline.chetto@univ-nantes.fr

*LINA - UMR CNRS 6241
2, rue de la Houssinière - BP 92208
44322 Nantes cedex 03
audrey.queudet@univ-nantes.fr

Abstract
In this paper, we present a component-based development project dedicated to real-time embedded harvesting systems. Energy harvesting consists in converting the ambient energy into electricity to power electronic devices, making them self-sufficient. Instead of minimizing its energy consumption, such a system must operate in a so-called energy neutral mode by consuming only as much energy as harvested in the environment. Consequently, our objective is to develop and integrate power management and scheduling facilities specifically adapted to energy harvesting embedded systems with real-time constraints. For example, these system level facilities have to make on-line decisions such as : when should the system use energy (i.e. active state) and when should it recharge the energy storage (i.e. idle state) ? We are providing these facilities as open-source software components. Adding them to the existing open-source CLEOPATRE library based on Linux/RTAI will participate to the evolution of the open-source real-time technology.

Keywords : RTAI, Linux-based operating systems, power aware scheduling, energy harvesting, component-based development.

1. Introduction

Scavenging energy from available sources offers the potential to power applications indefinitely without wires and extend the operating times of battery-powered systems. Ambient energy sources include light, heat differentials, mechanical vibration, transmitted RF signals, or any source that can produce an electrical charge through a transducer (photovoltaic cell , piezoelectric element...). There are many existing energy harvesting deployments all around us and most of them are real-time ones. In a real time system, performance is generally measured by the deadline miss rate. Lower is this rate, better is the Quality of Service (QoS) delivered by the system.

The autonomous nature of operation makes it imperative that the system learns its own energy environment and adapts its power consumption accordingly. Indeed, an important issue in such Real-Time Harvesting (RTH) systems is the way they control their energy consumption, which directly affects their performance, as well as their possibility to provide a continuous service, as depicted in Figure 1.

A good policy for power management and scheduling will have to make best effort decisions to adapt the quality of service according to the available energy profile. As the stored energy and the energy harvested from the environment are time-varying and thus very unstable, performance of a RTH system will heavily depend on its operating system and more precisely on all mechanisms for dynamically managing the processor activity and then maintain an acceptable quality of service even through a degraded mode.

The paper is organized as follows : Section 2 describes the characteristics of real-time energy harvesting systems. In section 3, we present a brief state of the art about both real-time scheduling and RTOS under energy constraints. A new scheduling framework is introduced in section 4. We describe an open-source library, Cleopatre, which will integrate energy management software components. Section 6 concludes the paper.

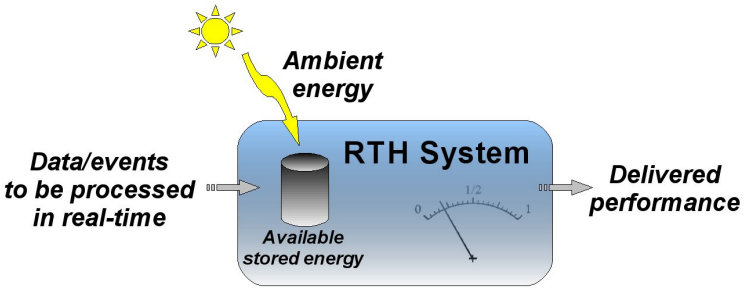

FIG. 1 – A typical RTH system

2. Specificities of energy harvesting computers

2.1. Motivations

Most prior wireless monitoring systems in last decades have relied on continuous power supplied by batteries such as lithium-ion cells. Their disadvantage is that they become depleted, must be periodically replaced or recharged and consequently place hard restrictions on products' usability, lifetime, and cost of ownership. Moreover, while processing power roughly doubles ever two years, battery technology advances at a much more sluggish pace. Even if it is possible to increase their energy density by tenfold within a few years, we must still consider practical safety concerns. First, given improper use, batteries with extremely high energy densities can become dangerous, explosive devices. Second, in many embedded applications, battery replacement is impractical or has high labor costs associated with maintenance. Harvesting energy in surrounding environment to power embedded systems for the lifespan appears to be the alternative to conventional batteries. In that context, super-capacitors or dynamically rechargeable batteries can be used to store the ambient energy.

New applications of energy harvesting technology for embedded systems are beginning to drive economic development in many sectors, as seen in the newly created on line journal about energy harvesting products (http ://www.energyharvestingjournal.com). Given the limitations of battery power, there are commercial, scientific and military needs for systems which can operate effectively using alternative renewable power sources. It concerns as well the high technology sectors as the general public products in which wireless sensor networks are used in a variety of applications, such as environmental applications (forest fire and flood detection, monitoring of drinking water and level air pollution), military applications (battlefield surveillance, reconnaissance of enemy forces), health applications (tele-monitoring of human physiological data, tracking and monitoring of doctors and patients), home applications (intrusion detection), or commercial applications (monitoring of product quality, climate control in large buildings). While some of these applications are marginal today, they will become commonplace one day and will contribute to improve the quality of living while reducing the overall health care cost. But in order to make this technology affordable to the general population, the researchers have to develop efficient devices built with inexpensive off-the-shelf components for harvesting and storing environmental energy, for sensing, processing and communicating data. Low-cost, autonomous sensor networks will enrich our lives by providing valuable data about the status of our environment (including human body), and will do so with no reoccurring cost or impact on the environment.

2.2. Real-time scheduling and energy harvesting

Typical embedded real-time applications require periodic activities (in general sensing, actuating and communicating) that have to be cyclically executed at fixed rates and within specific deadlines. To cope with the timing constraints, the computing system must be provided with a scheduler in charge of dispatching and executing the activities (called tasks) at the good times. A lot of work has been done for the well known Rate Monotonic (RM) and Earliest Deadline First (EDF) scheduling strategies [8]. These ones are commonly implemented in commercialized and free real-time operating systems including Linux/RTAI, see [7] for a survey on real time uniprocessor scheduling. However, under unpredictable

conditions (i.e. workloads), their performance may be poor. As an example, EDF algorithm is one of the most widely used online scheduling policies and it is proven to be optimal in deterministic environments only. Hence, the same poor behavior can be expected under fluctuating energy approvisioning. In brief, despite the significant body of results, specificities of energy harvesting has not been supported by these real-time schedulers.
Researchers have been motivated to design specific power management capabilities for RTH systems from less than 10 years only. The crucial issue is to find both power management and scheduling mechanisms that can adapt dynamically the activity of the embedded system according to the available energy. Up to now, when designing a real-time embedded system, the unique concern has been time, leaving energy efficiency as a hopeful consequence of empiric decisions. In the other hand, when designing an energy harvesting system, the unique concern has been energy efficiency in terms of harvesting and storing, abstracting for timing considerations. But the primary concern when designing a RTH system must be to consider both energy and time availabilities in order to avoid energy waste and energy lack and to guarantee timing requirements of real-time activities perpetually.

3. State of the Art

3.1. Real-time scheduling and energy management

Most of the research on energy-aware real-time scheduling focused on either minimizing the energy consumption or maximizing the system performance such as the lifetime achieved under the energy constraints [SCH 04]. In such works, rechargeability of the energy storage unit is always disregarded. These research activities use Dynamic Power Management techniques (DPM) and Dynamic Voltage and Frequency Selection (DVFS), [11]. Although both DPM and DVFS are able to reduce the power consumption of a device, they do not suffice for designing an efficient RTH system because there is a fundamental difference between power-aware and low-power technologies. In a RTH system, we have to make the best use of the available power and the goal of a scheduler is to assign real-time tasks to time slots such that all timing and power constraints are satisfied every time. We then say that the system operates in an energy neutral mode by consuming only as much energy as harvested.
Little work has explored the problem of scheduling real-time tasks in a uni-processor rechargeable system. The work by Allavena et al. was certainly the first one to concentrate on a rechargeable system with hard real-time constraints [1]. They addressed the problem of finding the scheduling of frame-based embedded systems which is able to execute all the tasks within the deadline. It relies on the unrealistic assumption that the power scavenged by the energy source is constant and all tasks consume energy at a constant rate. But in a real application, instantaneous power consumed by tasks varies along time depending on circuitry and devices required by the tasks. A solution is presented that schedules tasks in such a way that the wasted recharging energy is minimized and the energy storage level is at all times within two limits. Later in 2006, Moser at al. focused on scheduling tasks with deadlines, periodic or not. They proposed LSA (Lazy Scheduling Algorithm) and prove it to be optimal [10]. In that work, the consumption power of the computing system is characterized by some maximum value which implies that the total energy consumption of every task is directly connected to its execution time through the constant power of the processing device. In a very recent work presented in 2009, we relaxed this restrictive hypothesis [3].

3.2. RTOS and energy management

Several operating systems are available for wireless sensor networks : MANTIS from University of Colorado at Boulder, CONTIKI from the Swedish Institute of Computer Science, NANO-RK from Carnegie Mellon University and TinyOS from University of California at Berkeley. These open-source operating systems have been designed by the academic community for microcontrollers with small amounts of memory and have been used is a large variety of industrial projects. However, none of them provide software facilities for power management that jointly consider energy harvesting and timing constraints.
It has been now twenty years that GNU tools are used for embedded software. The unmistakable advantage of this model lies in the earning in flexibility and the possible option to modify the code in the

optics to answer specific needs. It lies as well in the training for several years, thanks to Internet notably, of a critical mass of developers. At the moment, companies already propose on the same CD-ROM, real-time kernels patches, network, development (editors, compilers, deboggers) and graphic tools allowing to realize applications and insure a technical support. Regrettably, neither the various versions of real-time Linux (RTAI [9] , Xenomai, RTLinux, Litmus^{RT}, Hard Hat Linux, Etlinux, Red Linux, KURT), nor the marketed owners executives do not answer suitably the very diversified requirements by the new applications powered through ambient energy source.
Linux is now everywhere, including in places where there are no power outlets. An appropriate design and power management implementation is required though, to allow the system to operate with an ambient energy source and to consume as little power as possible to preserve our environment.
By providing enhanced power management capabilities to RTAI, we will answer to the requirements of vendors of consumer electronics devices.

4. GreenPD : a new scheduling scheme

The intuition behind the GreenPD scheduler is to run tasks according to a Priority Driven (PD) rule but not necessarily as soon as possible. For example, GreenEDF will use Earliest Deadline First whereas GreenRM will use Rate Monotonic. However, before authorizing a task to execute, we must ensure that the energy level is sufficient to provide energy for all operations performed by occurring tasks (general processing, I/O operations, memory accesses,...) considering their worst case timing and energy requirements and the replenishment rate of the storage unit. When this condition is not verified, the processor has to be idle so that the storage unit recharges as much as possible and as long as the system will be able to meet all the deadlines i.e. the system will have available time to remain idle. We assume here that the system consumes no energy in the idle state. The way of scheduling tasks is based on the on-line computation of two quantities called slack time and slack energy, the latter one being a new concept dedicated to hard deadline tasks with energy constraints. The slack-based method for delaying tasks and making the processor inactive during recharging phases of the energy storage unit is the crucial point. On-line computing by how long the tasks should be delayed is possible thanks to a specific mechanism for computing the so-called slack time. GreenPD functional behavior is illustrated in figure 2.

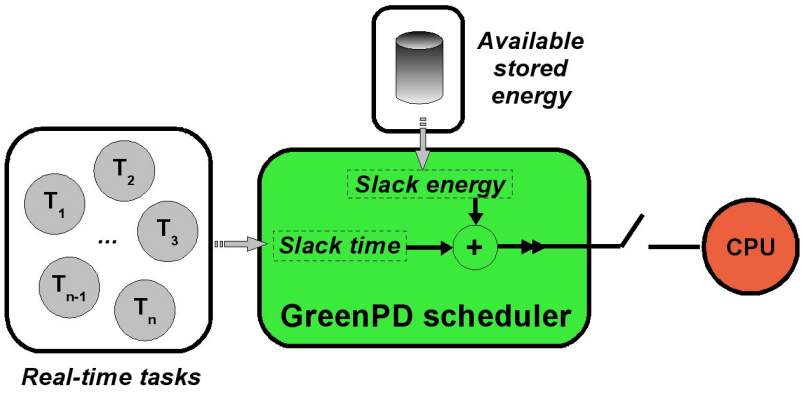

FIG. 2 – GreenPD scheduling scheme

We have proved through simulation how to schedule the tasks of a given application so as to guarantee their timing constraints perpetually by suitably exploiting both the processor utilization and the available ambient energy. This work is conducted both under static and dynamic priorities. GreenPD algorithm, relative concepts such as slack energy and simulation results can be found in [5]. Furthermore, we are now considering quality of service facilities. Our objective is to provide them as open-source software components on an additional shelf dedicated to dynamic power management.

5. CLEOPATRE Library

5.1. General framework

From 2002 until 2006, a library of free software components was developed within the French National project CLEOPATRE (Software Open Components on the Shelf for Embedded Real-Time Applications) in order to provide more efficient and better service to real-time applications, [12]. Our purpose was to enrich the real-time facilities of real-time Linux versions. RTAI was the solution adopted for this project because we wanted the CLEOPATRE components to be distributed under the LGPL license which is also the one used in the RTAI project. The LGPL allows proprietary code to be linked to the GNU C library, glibc. When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License (GPL) therefore permits such linking only if the entire combination fits its criteria of freedom. The LGPL license permits more lax criteria for linking other code with the library. Companies do not have to release the source to code which has been dynamically linked to an LGPLed library, which makes the use of such codes much more attractive.
The CLEOPATRE library offers selectable COTS (Commercial-Off-The-Shelf) components dedicated to dynamic scheduling, aperiodic task service, resource control access, fault-tolerance and now, QoS scheduling. An additional task named TCL (Task Control Layer) interfaces all the CLEOPATRE components and is considered as the task of higher priority in the whole system. It has been added as a dynamic module in $RTAIDIR/modules/TCL.o, and interfaces with the legacy RTAI scheduler defined in $RTAIDIR/modules/rtai_sched.o, as depicted in figure 3. As the CLEOPATRE interface is totally independent from the RTAI core layer, it can be directly used with Xenomai which supports the RTAI API and easily adapted to any other Linux real-time extension.

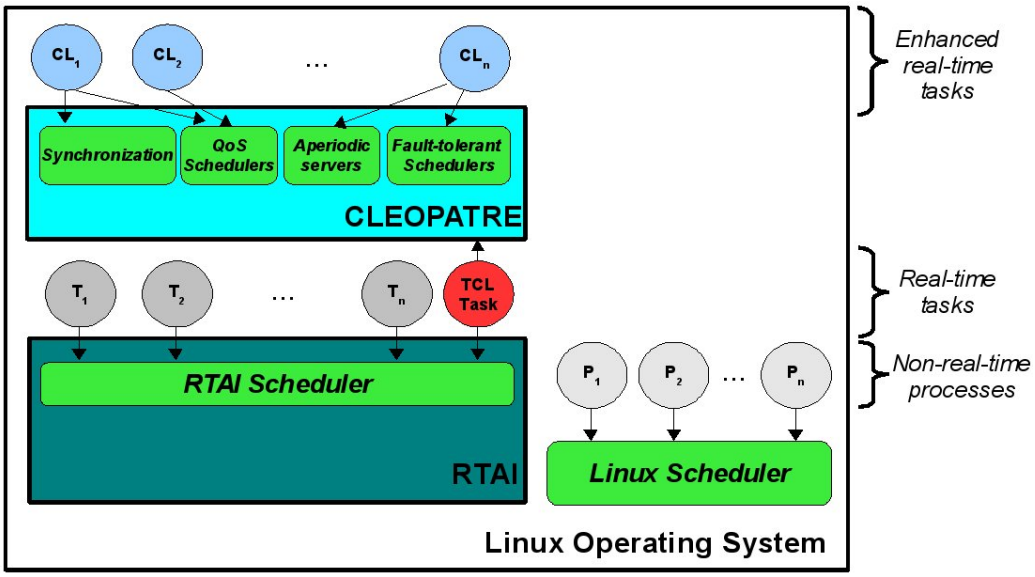

FIG. 3 – Cleopatre Library within Linux/RTAI

CLEOPATRE applications are highly portable to any new CPU architecture thanks to this OS abstraction layer which makes the library of services, generic [4]. The CLEOPATRE Off-the-Shelf components are optional except the OS abstraction layer (TCL) and the scheduler. At most one component per shelf can be selected. Since all components of a given shelf have the same programming interface, they are interchangeable. Everything needed to use and develop CLEOPATRE can be downloaded from the web site of the project : http : //cleopatre.rts-software.org.

5.2. Power management COTS

Of particular interest are techniques applicable to running systems, adjusting power parameters on-the-fly while ensuring real-time deadlines of running tasks are met. DPM software exploits recent advances in hardware to scale clocking information (such as CPU and core bus frequencies) and core voltages with low latency. This allows these parameters to be adjusted very frequently in order to realize power savings during brief idle periods or execution of tasks with lower performance and power demands. For GreenPD to be operational, the operating system has to automatically select desired power parameters based on system state, and more precisely two fundamental data : slack time and slack energy.

6. Conclusion

Ten years ago, the evolution of Linux passed by the transfer of the academic skills in real-time processing (scheduling, fault-tolerance, communication) and the contribution of answers to the needs of the new embedded applications (presence of overload conditions, quality of service guarantees, ...). Today, new generation applications are more energy consuming and rely on ambient energy sources in order to be "green". Our project is then to transfer new skills relative to open-source scheduling and power management facilities for such applications.

References

1. A. Allavena and D. Mosse. Scheduling of frame-based embedded systems with rechargeable batteries. *In Workshop on Power Management for Real-time and Embedded systems* (in conjunction with RTAS 2001), 2001.
2. H. Chetto and M. Chetto. Some results of the earliest deadline scheduling algorithm. *IEEE Transactions on Software Engineering*, 15(10) : 1261-1269, 1989.
3. M. Chetto and H. El Ghor. Real-time Scheduling of periodic tasks in a monoprocessor system with rechargeable energy storage. *In WIP Proceedings of the 30th IEEE Real-Time Systems Symposium*, December 2009.
4. M. Chetto, T. Garcia and A. Marchand-Queudet. Cléopatre : Real-Time facilities for Linux Based Control Applications. *Journal of Computing and Information Technology*, 15 (2) : 131-142, June 2007.
5. H. El Ghor, M. Chetto and R. Hage Chehade. A Real-Time Scheduling Framework for Embedded Systems with environmental energy harvesting. *Technical report of IRCCyN*, http ://hal.archives-ouvertes.fr, December 2010.
6. A. Kansal and J. Hsu. Harvesting aware power management for sensor networks. *In IEEE Proceedings of Design Automation Conference*, 2006.
7. J.-W.-S. Liu. Real-Time Systems. Prentice-Hall, 2000.
8. C.-L. Liu and J.-W. Layland. Scheduling algorithms for multiprogramming in a hard real-time environment. *Journal of the Association for Computing Machinery*, 20(1) : 46-61, 1973.
9. P. Mantegazza. DIAPM RTAI for Linux : Why's, what's and how's. *Real Time Linux Workshop*, University de Technology of Vienna, 1999.
10. C. Moser, D. Brunelli, L. Thiele and L. Benini. Real-time scheduling for energy harvesting sensor nodes. *Real-Time Systems*, Volume 37, Issue 3, Pages : 233 - 260, December 2007.
11. M.T. Scmitz, B.M. Al-Hashimi and P. Eles. System-Level Techniques for Energy Efficient Embedded Systems. *Kluwer Academic Publishers*, 194 pages, 2004.
12. M. Silly-Chetto, T. Garcia-Fernandez and A. Marchand-Queudet. CLEOPATRE : Open-source Operating System Facilities for Real-Time Embedded Applications. *The Journal of Computing and Information Technology*, 15(2) : 131-142, June 2007.