Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2008, Article ID 502768, 2 pages
doi:10.1155/2008/502768

Editorial

Operating System Support for Embedded Real-Time Applications

Alfons Crespo,[1] Ismael Ripoll,[1] Michael González-Harbour,[2] and Giuseppe Lipari[3]

[1] *Departament d'Informatica de Sistems i Computadors, Universidad Politécnica de València, 46022 Valencia, Spain*
[2] *Universidad de Cantabria, 39005 Santander, Spain*
[3] *Scuola Superiore Santa'Anna, 335612 Pisa, Italy*

Correspondence should be addressed to Alfons Crespo, acrespo@disca.upv.es

Received 18 February 2008; Accepted 18 February 2008

The rapid progress in processor and sensor technology combined with the expanding diversity of application fields is placing enormous demands on the facilities that an embedded operating system must provide.

Embedded systems can be defined as computing systems with tightly coupled hardware and software that are designed to perform a dedicated function. The word embedded reflects the fact that these systems are usually an integral part of a larger system.

We can find a large variety of applications where embedded systems play an important role, from small stand-alone systems, like a network router, to complex embedded systems supporting several operating execution environments as we can find in avionic applications.

This variety of applications also implies that the properties, platforms, and techniques on which embedded systems are based can be very different. The hardware needs can sometimes be achieved with the use of general purpose processors, but in many systems specific processors are required, for instance, specific DSP devices to perform fast signal processing. Memory management capabilities are necessary in some systems to provide memory protection and virtual memory. Special purpose interfaces are also needed to support a variety of external peripheral devices, energy consumption control, and so on.

Nowadays, the use of processor-based devices has increased dramatically for most of our activities, both professional and leisure. Mobile phones and PDAs are used extensively. Consumer electronics (set-top boxes, TVs, DVD players, etc.) have incorporated microprocessors as a core system component, instead of using specific hardware. This trend is expected to grow exponentially in the near future.

Embedded applications have some common features such as the following.

(i) Limited resources. They are often strong limitations regarding available resources. Mainly due to cost and size constraints related to mass production and strong industrial competition, the system resources as CPU, memory, devices have been designed to meet these requirements. As a result of these limitations, the system has to deal with an efficient use of the computational resources.

(ii) Real-time application requirements. Some of the applications to be run in these devices have temporal requirements. These applications are related with process control, multimedia processing, instrumentation, and so on, where the system has to act within a specified interval.

(iii) Embedded control systems. Most of the embedded systems perform control activities involving input data acquisition (sensing) and output delivery (actuation). Deterministic communications are also another important issue.

(iv) Quality of service. An efficient use of the system resources is a must in embedded systems. Feedback-based approaches are being used to adjust the performance or quality of service of the applications as a function of the available resources.

The challenge is how to implement applications that can execute efficiently on limited resource and that meet nonfunctional requirements such as timeliness, robustness, dependability, performance, and so on.

Moreover, applications on embedded systems include more and more functionalities in order to cope with the needs of the users in home environments, industry, leisure activities, vehicles, avionics, instrumentation, and so on. To offer services for these applications, a considerable effort has been made in research and development on innovative real-time operating systems architectures and services. Designers and developers of real-time operating systems have to consider many challenges arising from two opposite axes: efficient resource usage (processor, memory, energy, network bandwidth, etc.) and dynamic configuration and adaptation (component-based development and deployment, flexible scheduling, communications, etc.).

On the other hand, open-source operating system development has been recognized as a consolidated way to share experiences and developments in order to improve the quality and the reusability of the products. Currently, there are several distributions for embedded systems based on Linux and other open source developments.

This special issue focuses on new results of research work and development in the field of real-time operating systems for embedded applications with special emphasis on open source developments.

From the real-time operating system point of view, there are several topics that can be considered very relevant in the near future, illustrated as follows.

Partitioned systems

The development of embedded applications is entering into a new domain with the availability of new high-speed processors and low cost on-chip memory. As a result of these new developments in hardware, there is an interest in enabling multiple applications to share a single processor and memory. To facilitate such a model the execution time and memory space of each application must be protected from other applications in the system. Partitioning operating systems represents the future of secure systems. They have evolved to fulfill security and avionics requirements where predictability is extremely important. In avionics systems, for instance, running interrupts other than the system clock needed for cycling the partitions is discouraged. In a partitioning operating system, memory (and possibly CPU-time as well) is divided among statically allocated partitions in a fixed manner. The idea is to take a processor and make it appear as if there were several processors, thus completely isolating the subsystems. Within each partition, there may be multiple threads or processes, or both, if the operating system supports them. How these threads are scheduled depends on the implementation of the OS.

Innovative techniques

Innovative techniques in scheduling are needed to provide support for adjusting the load to the system needs, managing the dynamic allocation of memory under temporal and spatial constraints, managing energy to allow trading performance for reduced energy consumption in combination with the time constraints, providing fault-tolerance to deal with failure management and recovery, and so on.

Security

Embedded systems are getting more and more complex, dynamic, and open, while interacting with a progressively more demanding and heterogeneous environment. As a consequence, the reliability and security of these systems have become major concerns. An increasing number of external security attacks as well as design weaknesses in operating systems have resulted in large economic damages, which results in difficulties to attain user acceptance and getting accepted by the market. Consequently, there is a growing request from stakeholders in embedded systems to make available execution platforms which address both integrity and security concerns. For instance, it is important to avoid denial of service issues provoked by resource shortage (e.g., memory, CPU), while from an integrity viewpoint it is important to ensure availability of resources. It is also important to prevent malicious access to data created by another application.

Other aspects

Other aspects such as mutiprocessor system support, power-aware operating systems, real-time communications will have a relevant role in the next generation of embedded systems.

In this issue, several papers offer the particular vision of these issues. The first paper provides an approach of partitioned systems based on the L4 microkernel, whereas the second paper proposes a multiprocessor embedded system based on ASMP-Linux. The third and fourth papers deal with resource and reconfiguration management. The last two papers present application environments where the real-time operating systems present specific services to fulfill the requirements of these applications.

Alfons Crespo
Ismael Ripoll
Michael González-Harbour
Giuseppe Lipari