ZottaOS

Announcing ZottaOS-Soft

A Soft Real-Time Kernel with Run-time Guarantees for Ultra-Low Power Microcontrollers

Hard real-time schedulers are well established among practitioners. On the assumption that all run-time uncertainties are removed and that upper bounds of the application workload are respected at runtime, hard real-time provides a cost effective and easy to implement basis for applications requiring predictable dynamic behavior. In hard real-time systems, all thread deadlines must be met. Soft real-time systems do not have time constraints that are as stringent.

When no schedulable guarantees can be given, hard real-time scheduling algorithms perform poorly. To guarantee that each thread meets its deadline, hard real-time systems are designed and tested under worst-case situations where threads have maximum executing time, minimum release intervals and minimum deadlines. Such a design usually results into an overly sized system having poor resource utilization. On the other hand, by not having to meet every deadline, the system becomes simpler and more cost-effective while taking greater advantage of the available resources.

ZottaOS-Soft improves on other commercial soft real-time kernels in that:

- It provides a powerful mechanism that allows a controlled and fair degradation of quality of service;
- It allows transient overload without jeopardizing the scheduling of the threads.

In ZottaOS-Soft, consecutive thread executions are partitioned into mandatory and optional invocations. Mandatory invocations have precedence over optional ones and are conditioned by a hard real-time scheduler. Optional invocations on the other hand are dispatched only when the scheduler can assert that they will not overrun their deadlines.

ZottaOS-Soft Specifications

- Supports hard and soft periodic and event-driven threads;
- Common execution stack for all threads to minimize RAM usage;
- Lock-free (non-blocking) management of scheduling queues minimizing time periods in which interrupts are disabled;
- Interruptible context switching;
- Support for queued concurrent FIFO between tasks;
- Support for concurrent LIFO update of arbitrary sized datum between tasks;
- Easy to use API;
- Compatible thread code for all ZottaOS kernels, only the thread creation call differs;
- Available for MSP430Fxxx and CC430Fxxx this coming May.

More information available on **www.zottaos.com**

The information presented in this document does not form part of any quotation or contract, is believed to be accurate and reliable and may be changed without notice. No liability will be accepted by the publisher for any consequence of its use. Publication thereof does not convey nor imply any license under patent or other industrial or intellectual property rights.