

# Hardware/Software Interaction and Co-design Track

## Theme

The Hardware/Software Interaction and Co-design Track at RTSS 2007 will attempt to address critical issues related to architectures, languages, tools and methodologies for the design of embedded computing systems. The goal is to specifically highlight problems and solutions where timing constraints play a central role within the hardware/software co-design framework. Papers that propose novel techniques and models for handling real-time issues, or cast established results from the mainstream real-time systems domain into an embedded systems setting are especially welcome.

Embedded computing systems typically have limited computing resources, stringent timing, energy and thermal constraints, and high confidence requirements. This, coupled with the presence of multiple heterogeneous components such as microcontrollers, DSPs, smart sensors, and FPGAs, result in the modeling, analysis and design of such systems to be a challenging task. Hence, advanced tools that allow designers to simulate, build, debug and verify system design—both at hardware and software levels—play a crucial role in modern embedded system design. This special track aims to elicit original work that can shed new insight on how to manage this complex design process of hardware and software for embedded systems with a focus on timing related issues.

## Topics of Interest

This special track seeks papers in all areas of hardware/software co-design, as well as emerging areas that relate specifically to real-time embedded systems. Areas of interest include, but are not limited to:

- 1) Computer-Aided Co-Design Techniques: Specification and modeling, design representation, synthesis, partitioning, platform and power management, estimation, design space exploration, co-design for reliable systems.
- 2) Software for Co-Design: Software development environments, real-time operating systems, process scheduling, software synthesis, retargetable compilation, worst-case execution time analysis of embedded software, designing software and architectures for predictability.
- 3) Co-Design Architectures: Hardware/software interfaces, distributed, heterogeneous SoC architectures, re-configurable computing.
- 4) System Development Process: Design methodology, concurrent engineering, design reuse, process management, integration, novel simulation techniques, verification and testing, debugging, rapid prototyping.
- 5) Applications and Success stories: Major lessons that can be learned from success stories, applications such as multimedia processing, games, automotive electronics and healthcare/medical electronics.