Recent News

About LCTES

LCTES provides a link between the programming languages and embedded systems engineering communities. Researchers and developers in these areas are addressing many similar problems, but with different backgrounds and approaches. LCTES is intended to expose researchers and developers from either area to relevant work and interesting problems in the other area and provide a forum where they can interact.

Important Dates

Submission deadline	Feb. 1
Notifications by	Mar. 16
Camera-ready deadline	Mar. 25

Call for Papers

Embedded system design faces many challenges both with respect to functional requirements and nonfunctional requirements, many of which are conflicting. They are found in areas such as design and developer productivity, verification, validation, maintainability, and meeting performance goals and resource constraints. Novel design-time and run-time approaches are needed to meet the demand of emerging applications and to exploit new hardware paradigms, and in particular to scale up to multicores (including GPUs and FPGAs) and distributed systems built from multicores.

LCTES 2016 solicits papers presenting original work on programming languages, compilers, tools, theory, and architectures that help in overcoming these challenges. Research papers on innovative techniques are welcome, as well as experience papers on insights obtained by experimenting with real-world systems and applications.

Papers are solicited on, but not limited to, the following topics in embedded systems:

- Programming language challenges, including:
  - Domain-specific languages
  - Features to exploit multicore, reconfigurable, and other emerging architectures
  - Features for distributed, adaptive, and real-time control embedded systems
  - Language capabilities for specification, composition, and construction of embedded systems
  - Language features and techniques to enhance reliability, verifiability, and security
  - Virtual machines, concurrency, inter-processor synchronization, and memory management

- Compiler challenges, including:
  - Interaction between embedded architectures, operating systems, and compilers
  - Interpreters, binary translation, just-in-time compilation, and split compilation
  - Support for enhanced programmer productivity
  - Support for enhanced debugging, profiling, and exception/interrupt handling
  - Optimization for low power/energy, code and data size, and best-effort and real-time performance
  - Parameterized and structural compiler design space exploration and auto-tuning
- Tools for analysis, specification, design, and implementation, including:
  - Hardware, system software, application software, and their interfaces
  - Distributed real-time control, media players, and reconfigurable architectures
  - System integration and testing
  - Performance estimation, monitoring, and tuning
  - Run-time system support for embedded systems
  - Design space exploration tools
  - Support for system security and system-level reliability
  - Approaches for cross-layer system optimization
- Theory and foundations of embedded systems, including:
  - Predictability of resource behaviour: energy, space, time
  - Validation and verification, in particular of concurrent and distributed systems
  - Formal foundations of model-based design as basis for code generation, analysis, and verification
  - Mathematical foundations for embedded systems
  - Models of computations for embedded applications
- Novel embedded architectures, including:
  - Design and implementation of novel architectures
  - Workload analysis and performance evaluation
  - Architecture support for new language features, virtualization, compiler techniques, debugging tools
  - Achitectural features to improve power/energy, code/data size, and predictability
- Empirical studies and their reproduction, and confirmation