Editorial: Model-driven Embedded-system Design

In July 2008, the Artist Network of Excellence organized the second Artist Workshop on Models of Computation and Communication, MOCC 2008. The call introduced the workshop topic as follows:

> Embedded systems are omnipresent in modern society, and society crucially depends on their proper functioning. The complexity of embedded-system design however is increasing rapidly, through the use of multiprocessor cores, through the integration of embedded systems in ubiquitous networks, and through the increasing interaction between embedded systems and their users and environments. To obtain a reliable operation of embedded systems while maintaining resource efficiency, the embedded-system design process needs to be based on a solid basis of computational models.

The call then raised the following focus question:

> How can computational models be used to drive embedded-system design such that performance, quality and resource constraints are met?

After a successful workshop, with 14 contributions presenting different views on and answers to this question, we invited contributions for this special issue centered around the same theme, through an open call for papers.

We received 33 submissions. After a careful review process, nine of the submissions have been selected to appear in this special issue, after revisions based on reviewer feedback. Four of the papers that appear in this special issue resulted from work originally presented at MOCC 2008; five contributions were submitted in reaction to the open call for papers.

The first three papers build upon actor-based dataflow models of computation originating from the seminal work of Karp and Miller, Kahn, and Lee. The papers directly address the above question, providing performance analysis, resource dimensioning, and synthesis techniques for data processing applications. Such applications are becoming increasingly dynamic in nature. All three papers advance state-of-the-art results towards such more dynamic systems.

Marc Geilen presents in "Synchronous Data Flow Scenarios" techniques to compute tight, worst-case performance guarantees across multiple synchronous dataflow

graphs, where each graph characterizes a single mode of operation – a scenario – for the considered application. The approach builds upon a combined analysis of the steady-state behavior of scenarios and the transient behavior of scenario changes. The technique ultimately allows a tighter dimensioning of multiprocessor execution platforms than traditional worst-case approaches.

Maarten Wiggers et al. present in "Buffer Capacity Computation for Throughput-Constrained Modal Task Graphs" a buffer sizing technique for a novel dataflow model of computation, Variable-Rate Phased Dataflow, where dataflow actors may have different phases of computation, and where each phase may be executed a variable number of times. The model allows to capture data-dependent execution rates. The technique guarantees that the computed buffer capacities are sufficient to guarantee a required throughput, and as such can be used in multiprocessor design flows.

In "Analysis of SystemC Actor Networks for Efficient Synthesis", Joachim Falk et al. bridge the gap between computational models for analysis and programming models. The authors present techniques for analyzing, clustering, and scheduling actors extracted from SystemC models. SystemC is widely used for modeling and implementing embedded systems. The presented techniques use a classification of the extracted actors into three classes differing with respect to the dynamic nature of the performed computations. This allows to more aggressively optimize the static parts of an application.

The next two papers address the fundamental issue of functional correctness, which needs to be addressed before any type of system dimensioning or performance optimization is possible. The papers develop model-checking and related formal verification techniques in a context of embedded systems.

Niloofar Razavi et al. develop in "Sysfier: Actor-based Formal Verification of SystemC" a SystemC model checking technique by extracting actor models from the SystemC code that are amenable to model checking properties specified in temporal logic. The approach via the intermediate actor representation allows to leverage the theory and techniques developed for those actor models. The assumed actor model is the model originating from Hewitt and Agha, where actors are units of concurrency that communicate via asynchronous, non-blocking messages passing. The actor-based dataflow models of computation underlying the first three papers in this special issue can be seen as restricted versions of this general actor model of Hewitt and Agha, limiting their expressiveness, but increasing analysis potential. Razavi et al. show the feasibility of their model checking approach based on this more general actor model via several hardware verification and hardware/software co-verification case studies.

In "Component-Based Modeling and Verification of Dynamic Adaptation in Safety-Critical Embedded Systems", Rasmus Adler et al. develop component-based modeling techniques that enable various types of informal validation, such as simulation, and formal verification, such as theorem proving and model checking, of adaptive behavior in embedded systems. Thus, also this work addresses the already mentioned trend that embedded systems are rapidly becoming more dynamic in nature. The approach addresses the typical complexity problems in formal verification via component-based modeling and compositional analysis.

Cormac Driver et al. pursue in "Managing Embedded Systems Complexity with Aspect-Oriented Model-Driven Engineering" a different approach to manage design complexity, namely separation of concerns via aspect orientation. The presentation is based on the Theme/UML language and toolset. The approach supports the modular specification of both functional and non-functional concerns such as fault-tolerance and resource management, after which model composition and code generation steps guarantee correct-by-construction implementations. The method is illustrated with a pacemaker case study. Since the Theme/UML approach has not been specifically designed for embedded systems, the authors analyse the applicability to embedded systems and identify topics for further development.

The last three papers address different topics related to real-time systems. Real-time guarantees are typically very important in the proper functioning of embedded systems. At the same time, it is challenging to combine real-time guarantees with efficient use of resources, which is another crucial aspect for embedded systems.

In "Real-Time Performance Analysis of Multiprocessor Systems with Shared Memory", Simon Schliecker and Rolf Ernst present techniques for response-time analysis for applications that share resources in multiprocessor systems. The approach uses event-stream models, as opposed to often-used task-based analysis approaches. The analysis is compositional to avoid complexity problems, and it allows to obtain tighter results for multiprocessor systems with shared resources than earlier approaches. The event-stream abstraction is consistent with the dataflow abstraction underlying the first three papers. The various techniques are therefore to some extent applicable to the same types of systems.

Euiseong Seo et al. discuss in "Dynamic Alteration Schemes of Real-Time Schedules for I/O Device Energy Efficiency" heuristics to optimize the dynamic power management potential for I/O devices in non-preemptive real-time embedded systems. I/O devices consume considerable amounts of power. Switching between active and sleep modes may save energy, but the transitions take time, which affects real-time behavior of applications. The heuristics for on-line optimization of dynamic power management opportunities for I/O devices presented by Seo et al. lead to energy savings that are particularly relevant for embedded systems, in which energy is often a scarce resource.

Finally, Gianpiero Cabodi et al. present in "Boosting Software Fault Injection for Dependability Analysis of Real-Time Embedded Applications" non-intrusive fault-injection techniques based on the use of a loadable operating-system kernel. The approach allows to analyze dependability of real-time systems in an early stage of development, when the final execution hardware may not yet be available. To achieve this goal, the fault-injection kernel is combined with a virtual prototype of the hardware. The authors experimentally show the feasibility of decoupling dependability analysis from hardware availability.

In conclusion, the nine papers cover a wider range of topics related to model-driven embedded-system design. The presented material shows some trends. The two most prominent ones are the trend to address the increasingly dynamic nature of today's embedded systems, both in modeling and analysis, and the trend to avoid complexity by separating concerns through techniques such as component-based modeling, compositional reasoning, aspect orientation, and resource virtualization.

We thank the authors that submitted their work to this special issue for their efforts. We are grateful to the reviewers, whose careful reviews and valuable suggestions helped to improve the quality of the final result. We thank Jörg Henkel and Martin Büchti for their assistance in completing this special issue. We believe the end result is worth reading.

Twan Basten
Embedded Systems Institute, Netherlands
Eindhoven University of Technology, Netherlands

Rolf Ernst
Technische Universität Braunschweig, Germany

Guest Editors