Embedded and Cyber-physical System: Model Based Design for Analysis & Synthesis! - an ICES Workshop in collaboration with UC Berkeley

Monday 6 February 2012
9.00-15.30
Lecture room L52, Drottning Kristinas vägen, KTH Campus

PROGRAMME
9:15 - 10:00 "Time for High-Confidence Cyber-Physical Systems", Edward Lee (UC Berkeley)
10:00 -10:30 "Model Based Development at Ericsson", Diarmuid Corcoran (Ericsson)
10:30 -10:45 Break
10:45 - 11:30 "ForSyDe: Towards a Formal Software Synthesis Methodology for Predictable Embedded Multiprocessor Applications", Ingo Sander (KTH)
11:30 - 12:00 "Best Practices for Adopting Model-Based Design for Embedded Applications", Fredrik Häbring (Mathworks)
12:00 - 13:30 Lunch
13:30 - 14:15 "EAST-ADL architecture description language, overview, directions and challenges", DeJiu Chen and Lei Feng (KTH)
14:15 - 15:00 "Embedded systems design: few solutions, multiple challenges", Tiberiu Seceleanu (ABB)
15:00 - 15:30 Wrap up

[Presentations will be made available on line at: www.ices.kth.se]

SPEAKERS ABSTRACTS

KEYNOTE: "Time for High-Confidence Cyber-Physical Systems", Edward Lee (UC Berkeley)
All widely used software abstractions lack temporal semantics. The notion of correct execution of a program written in every widely-used programming language today does not depend on the temporal behavior of the program. But temporal behavior matters in almost all systems, particularly in networked systems. Even in systems with no particular real-time requirements, timing of programs is relevant to the value delivered by programs, and in the case of concurrent and distributed programs, also affects the functionality. In systems with real-time requirements, including most embedded systems, temporal behavior affects not just the value delivered by a system but also its correctness.

This talk will argue that time can and must become part of the semantics of programs for a large class of applications. To illustrate that this is both practical and useful, we will describe recent efforts at Berkeley in the design and analysis of timing-centric software systems. In particular, we will focus on two projects, PRET, which seeks to provide computing platforms with repeatable timing, and PTIDES, which provides a programming model for distributed real-time systems.
Edward A. Lee is the Robert S. Pepper Distinguished Professor in the Electrical Engineering and Computer Sciences (EECS) department at U.C. Berkeley. His research interests center on design, modeling, and analysis of embedded, real-time computational systems. He is a director of Chess, the Berkeley Center for Hybrid and Embedded Software Systems, and is the director of the Berkeley Ptolemy project. From 2005-2008, he served as chair of the EE Division and then chair of the EECS Department at UC Berkeley. He is co-author of nine books (counting second and third editions) and numerous papers. He has led the development of several influential open-source software packages, notably Ptolemy and its various spinoffs. He received the B.S. degree in Computer Science from Yale University, New Haven, CT, in 1979, the S.M. degree in EECS from the Massachusetts Institute of Technology (MIT), Cambridge, in 1981, and the Ph.D. degree in EECS from the University of California Berkeley, Berkeley, in 1986. From 1979 to 1982 he was a member of technical staff at Bell Telephone Laboratories in Holmdel, New Jersey, in the Advanced Data Communications Laboratory. He is a co-founder of BDTI, Inc., where he is currently a Senior Technical Advisor, and has consulted for a number of other companies. He is a Fellow of the IEEE, was an NSF Presidential Young Investigator, and won the 1997 Frederick Emmons Terman Award for Engineering Education.

"Model Based Development at Ericsson", Diarmuid Corcoran (Ericsson)
"Traditionally in large scale system and software engineering the process of moving design information and decisions from requirements phase to systems engineering phase to software design phase is very expensive and error prone. Many of the most serious and costly design errors in large scale systems are often related to misunderstanding, misinterpretation or just plain information corruption somewhere along this information dependency chain. At Ericsson we have used various flavors of model based development for system specification, implementation and verification. This talk tries to capture in brief our experiences, both good and bad, using these engineering approaches to mitigate and simplify the development of large scale communication systems.

"ForSyDe: Towards a Formal Software Synthesis Methodology for Predictable Embedded Multiprocessor Applications ", Ingo Sander (KTH)
The presentation addresses the increasing complexity of software design for multiprocessor embedded systems by proposing a design methodology that combines a formal foundation based on the theory of models of computation (MoCs) and the industrial system design language SystemC. Particular focus is given to the foundations for a software synthesis flow, which starts with an executable system model and yields an implementation on a multiprocessor system-on-chip.

The ForSyDe (Formal System Design) methodology provides the designer with SystemC modeling libraries that lead to executable system models from which abstract analyzable models can be extracted. Using these abstract models, the design space exploration, mapping and synthesis process can make use of the rich set of existing MoC theory by for instance incorporating scheduling and buffer optimization techniques to yield an efficient implementation on a multiprocessor system-on-chip. The presentation will also discuss to what extent performance guarantees can be given provided a predictable architecture is used as target architecture.

"Best Practices for Adopting Model-Based Design for Embedded Applications", Fredrik Häbring (Mathworks)
When transitioning to Model-Based Design for embedded systems development, it is essential to consider an overall plan spanning people, development processes, and tools. Choosing the right first steps are key for demonstrating short-term benefits and for establishing a culture that enables the full realization of the theoretical benefits of this approach. MathWorks will present a set of practical strategies for determining the first steps when deploying Model-Based Design and code generation in production development processes. These strategies have been gleaned from successful and not-so-successful transformations to Model-Based Design at companies from a variety of different industries.
“EAST-ADL - an architecture description language for communicating and managing automotive embedded systems (content overview, directions, and challenges)” DeJiu Chen and Lei Feng (KTH)

EAST-ADL is a domain-specific architecture description language (ADL) for model-based development of automotive embedded systems. Based on a standardized conceptualization of target embedded systems as well as the related engineering concerns, the language brings a potential for a wide range of benefits in regard to information management, system integration, model transformation and tool interoperability. The covered aspects include industrial methodologies, requirements, system design and implementation, variability handling, performance and dependability, system verification and validation. This talk provides an overview of EAST-ADL language and in particular highlights the future research directions and challenges.

"Embedded systems design: a few solutions, multiple challenges", Tiberiu Seceleanu (ABB)

We go during this presentation through some of the design aspects to be resolved by the embedded systems developer in the design process. Some challenges are triggered by advances in technologies (such as multicore/multiprocessing), while most of them can be seen today as “classical” (costs, performance, legacy, etc.).

The modern day designer is, in addition, extremely dependent on the (sometimes specific) tool environment. While at certain levels of design, automation is already a historic fact (compilation, synthesis), the complexity of modern systems places an increasing pressure on extending it to other levels, too. The quality of the final products relies also on a qualitative and performant process, heavily dependent on the tools supporting it, but these often change between projects, between products, etc...

We describe here a few solutions to help improving the life of the embedded systems designer, on the following directions.

- Tool interoperability
- Tool exchange
- Tool support for hardware/software codesign
- Tool support for multicore
- Long life cycle support via methodological and tooling features