

Tutorials

A1 Platform-Based Design for Systems-on-Chips

Wayne Wolf, Princeton University, USA

Jörg Henkel, NEC Research, USA

Embedded processors will be essential components of system-on-chips. However, there are so many architectural alternatives that designers may be overwhelmed by choice. Platform-based design is an emerging alternative that takes the best of hardware/software co-design and utilizes it within a reasonable methodological framework. Rather than design every SoC architecture from scratch, vendors develop architectural platforms for particular applications, such as wireless or video. Customers then customize with both hardware and software to provide competitive advantages for their systems. This approach is especially well-suited to standards-driven industries like wireless or video, since all vendors must comply with certain minimum requirements defined by the standard.

This tutorial will discuss the major issues in platform-based design. Vendors must be able to specify platforms that satisfy the needs of a broad base of customers yet allow customisation and product differentiation. Platform users must be able to evaluate the costs and benefits of hardware modifications to the platform, must be able to map generic software onto the platform, and must be able to verify the platform's functional correctness, performance, and power consumption. We will discuss techniques for choosing heterogeneous multiprocessor configurations to suit an application space, techniques for performance analysis of a platform, hardware customisation, the role of configurable hardware, software optimisations, and design verification challenges.

B1 SpecC Language and Design Methodology

Organizer: Daniel Gajski, University of California, USA

Tadatoshi Ishii, SpecC Consortium, J

Andreas Gerstlauer, University of California, USA

Jianwen Zhu, University of Toronto, CAN

SpecC is a specification and SLDL that is capable of representing specification, architecture, communication and implementation models of systems and digital products. It is based on CSP, FSM and DE models of computation. It has very well defined guidelines for modelling and refinement rules for transforming one model into the other.

The tutorial will cover: basic requirements for SLDL; general methodology with SpecC; SpecC modeling and tools; industrial-strength experiments; SpecC Consortium — goals and plans. This tutorial will offer a global picture on SL efforts and concentrate on design flow from specification to manufacturing, as a distinguishing characteristic from other similar tutorials.

Audience: This tutorial is intended for system and SOC designers, particularly for designers working on product specification and embedded software. It is also intended for design and system managers who are defining company design flows. It will also be beneficial to general DATE audience which will be exposed to a powerful SLDL alternative and system methodology.

C1 Testing Embedded-Core Based System-Chips

Erik Jan Marinissen, Philips, NL

Yervant Zorian, Logic Vision, USA

Advances in semiconductor technologies enable the design and manufacturing of complex system chips. Whereas in traditional IC design, reuse is limited to standard-cell libraries, modern system chips are increasingly designed by embedding large reusable models, the so-called cores. This tutorial provides an introduction into the motivation of core-based design and test development, highlights the new challenges related to core-based testing, and gives an overview of current industrial and academic practices in this domain. Special emphasis is placed on the current status of industry-wide efforts and standardization in IEEE P1500 Standard for Embedded Core Test and the VSI Alliance. The main modules of the tutorial are: Challenges in Embedded-Core Test; Architectural Elements for Core Test Access; Industry-Wide Efforts; SOC Testability Process, Tools, and Flows; Industrial Experiences.

Audience: IC designers, test engineers, and their managers, also researchers, test methodology developers, and test tool developers.

D1 CAD for Low Power: Exploring the Gaps between Theory and Practice

Enrico Macii, Politecnico di Torino, IT

Renu Mehra, Synopsys, USA

Roberto Zafalon, ST Microelectronics, IT

For battery-operated applications, power consumption has become a critical design issue. Furthermore, as density, size and complexity of the chips continue to increase, the difficulty in providing adequate cooling might either add significant cost or limit the functionality of the electronic systems. In the past 10 years, several techniques for designing low-power circuits have been presented in the scientific literature. However, only a limited number of them have found their way to a stable application in current design flows, and even fewer have been implemented as part of commercial CAD tools. Practical reasons, such as flow integration, compatibility with simulation, verification and testing frameworks, conflicts with architectural templates and platforms, are at the basis of this trend.

Purpose of this tutorial is providing to the audience an insight of what IC designers can do today to solve the power problem, what they would like to be able to do tomorrow, what they can expect from the EDA market in the near future, what academia forecasts as challenges in the not-so-near future of low power design. In order to deliver a more lively presentation, speakers will alternate on the podium to address the various issues. The topics that will be covered are: Why design low-power circuits; What to do by hand; what to do with tools; How some techniques have been automated; What tools cannot do and why; What next-generation tools should support; What next-generation tools will support; Where the effort should go in the long term.

A2 Scaling the Abstraction Cliff: High-level Languages for System Design

Stephen A Edwards, Synopsys, USA

Luciano Lavagno, University of Udine, IT

Traditionally, chip designers have been able to design systems using fairly primitive computational abstractions because of resource limitations: a chip could hold little more than a microprocessor or a single peripheral. With system-on-a-chip levels of integration, however, single chips will contain what once fit on many boards. Chip designers will be forced to start designing systems, and just as software developers rarely write millions of lines of assembly language, chip designers will be forced to move to higher-level languages.

This tutorial aims to shed light on the higher-level languages and models of computation that chip designers will be moving toward as integration levels increase. Part of our focus will be on dataflow-style languages, which are well-suited to the signal processing applications often found in embedded systems. Another focus will be on high-level control-dominated languages, including the graphical StateCharts and SDL or textual Esterel. In all cases, we will introduce the languages and the idea beneath them: their model of computation.

After a review of traditional hardware and software languages (Verilog, VHDL, C, and C++), we will discuss Java's model of concurrency; dataflow-oriented languages, both static and dynamic; control-dominated languages, including Esterel and StateCharts; and newer high-level languages, such as CoCentric System Studio, that combine some of these ideas. This tutorial is aimed at chip designers and prospective system designers who have some familiarity with traditional languages such as C or Verilog and are seeking to start thinking at higher levels of abstraction.

B2 Real Time Java for Embedded Systems

Wolfgang Rosenstiel, Universität Tübingen, D

Stephen Schmitt, Universität Tübingen, D

Due to its platform independency and its internet embedding the programming language Java has many advantages in programming distributed embedded systems. After a general motivation and the description of Java in the context of embedded systems the tutorial will mainly focus on two topics: (1) Java 2 Micro Edition (J2ME) including the virtual machine KVM-CLDC for real time applications with less strict real time requirements and (2) Real Time Java intended for applications with more strict real time constraints.

A reference implementation of the KVM-CLDC has been developed by Sun Microsystems for J2ME. This KVM is an interpreter based JVM with a platform independent thread system, a non compacting GC and a special pre-verified. The tutorial presents evaluations with respect to run time, memory footprint and portability support and effort. To avoid proprietary real time extensions for Java a working group has been founded to develop a real time Java standard. So far a specification is available. First implementations are expected by the end of year 2000. The

tutorial presents the real time Java concepts and especially concentrates on thread scheduling, memory management, thread synchronization and asynchronous events handling.

C2 Metrics, Techniques and New Developments in Mixed-Signal Testing

Gordon Roberts, McGill University, CAN

With the growing importance of analog circuits in commercial mixed-signal ICs and systems, this tutorial will outline the basics of mixed-signal testing. It begins with a brief introduction to test and its impact on product quality and cost. Then it reviews the basics of analog and digital circuits and outlines how manufacturing defects influence their behaviour. Subsequently, the use of standard sinusoidal-based techniques in testing analog and mixed-signal circuits is described. Issues of non-coherent and coherent sampling, and multi-tone testing are addressed. Noise issues are also considered, such as test accuracy (repeatability) and windowing operations. Finally, the tutorial looks at more recent developments underway in various research laboratories on DFT topics such as the IEEE 1149.4 mixed-signal test bus, fault-analysis and various DFT methods for testing A/Ds, CODECs, transceivers and PLLs.

Audience: Design engineers, test engineers, and their managers. No prior knowledge of mixed-signal IC testing is assumed. Basic understanding of electronic circuits and signal and system concepts is all that is required.

D2 Low-Power Mobile Wireless Communication System Design: Protocols, Architectures, and Design Methodologies

Sujit Dey, University of California, San Diego, USA

Anand Raghunathan, NEC USA C&C Research Labs, Princeton, USA

The demand for ubiquitous information access and manipulation (anytime, anywhere computing and communications) has created significant opportunities and challenges for semiconductor vendors, electronic system design houses, and EDA tools companies. The revenue from wireless voice/data handsets is expected to exceed that from PCs in the near future, and the use of wireless internet access is expected to overtake fixed internet access in the next few years. The above trends make it important to focus on addressing the challenges encountered in the design of mobile wireless communication systems.

One of the most important criteria in the design of such systems is to maximize the battery life, since it directly impacts the duration and extent of mobility. The increasing complexity of wireless communication protocols, increasing performance (bandwidth) requirements, and relatively slow growth in battery technology, make it critical to employ power-conscious protocols, power-efficient system architectures, and low-power design methodologies and tools, in the design of wireless communications systems.

The first part of this tutorial will present an introduction to the different functions performed in mobile wireless communications systems, including source coding, channel coding, modulation, power amplification, and various layers of the protocol stack, and describe power reduction techniques that can be employed at each step. Examples of low-power radio architectures will be presented, describing low-power features available in the various system components, as well as system-level integration and co-design tradeoffs.

The second part of the tutorial will present methodologies and tools for low-power system-level design, including system-level power estimation, HW/SW partitioning and mapping for low-power, low-power embedded software techniques, low-power memory and bus architectures, system-level power management, dynamic voltage and performance management strategies, and battery-friendly system design.