Tutorials

A1 Built-In Self-Test for System-Chips and Beyond

Yervant Zorian, Logic Vision, USA

Intended Audience:
Designers, researchers and test engineers interested in learning about the state-of-the-art in BIST methods and practices for complex systems-chips.

With the emerging trend of designing core-based complex system-chips, Built-in Self-test (BIST) is becoming the solution of choice that provides testability reuse, IP protection, and facilitates silicon debug. In addition, it is a key enabler for mixing and matching self-testable cores from different sources onto a single chip.
This tutorial covers both the basic methods and current practices for integrating BIST into cores, chips and systems. First it introduces the basic building blocks to act as BIST sources and sinks for test pattern generation and output data compaction respectively, and then it discusses their incorporation into the design flow of random logic cores and embedded memories for supporting BIST, diagnosis and even repair functions. The overall test integration at the chip level is also addressed in this tutorial, including BIST planning, scheduling, control and standard access mechanisms such as IEEE P1500. The tutorial also includes methods and practices for obtaining self-testable systems by reusing BIST beyond chip level testing.

B1 Reuse of Virtual Components in System-On-Chip Environments

Dr. Natividad Martínez Madrid, FZI Karlsruhe, D
Prof. Dr. Wolfgang Rosenstiel, U Tuebingen, D
Dr. Ralf Seepold, FZI Karlsruhe, D

The design of microelectronic systems is heavily driven by the fact that transistor and feature size have constantly decreased over the years, while frequency and density have increased. This gain has been supported by the development of new technologies and manufacturing equipment, which provide mechanisms to improve design efficiency. As a consequence of this progress, very complex development environments have evolved. Since the major key to the success is to shorten time, the prosperity of a product is also closely related to the necessity of efficient methodologies to create new or enhanced products to an aggressive time-scale.
Intellectual Property (IP) and design reuse methodologies are expected as key enablers to face both short and long term development objectives. Due to the fact that the level of complexity constantly increases, reuse of approved designs and the design of efficiently reusable components become the most crucial enterprise. This tutorial will cover an IP checklist, overview on standardization initiatives (e.g., VSI, VCX, RAPID etc.), presentation of different business models, documentation of state-of-the-art approaches (industrial methodologies and current research) and examples for design reuse performed in running projects.

C1 System-Level Power Optimisation: Techniques and Tools

Luca Benini, DEIS, Universita' di Bologna, I
Giovanni De Micheli, CSL, Stanford University, USA

Energy-efficient design requires reducing power dissipation in all parts of the design. Since software does not have a physical realization, we use appropriate models for analysing the software impact on the hardware power consumption.
This tutorial will present recent advances in the field for energy-efficient design, as well as methods for exploring the energy consumption/performance trade-off in electronic systems. Topics include, but are not limited to, hardware synthesis techniques, memory organization, interface design, software compilation, hardware/software trade-offs, accuracy/power consumption trade-offs, operating system issues and dynamic power management.

D1 Design Technology for Building Digital Wireless Systems on a Chip

Rajesh K. Gupta, University of California, Irvine, USA
Mani B. Srivastava, University of California, Los Angeles, USA

Intended Audience:
This tutorial is targeted for engineers, CAD developers and researchers interested in system design techniques and design tool requirements for building networked wireless systems.

The progress in IC technology is making chips — that incorporate all the elements of a complete wireless radio system on a chip — a real possibility. Such an "antenna-to-network" chip would incorporate an RF front end, baseband digital signal processing, link layer coding functions for error, compression, and encryption, and medium access control and other network protocols. This requires integration of analog circuits, high performance custom signal processing datapaths and cores, customised logic, embedded processor, and complex software environments on the same chip. The design, simulation, implementation, and testing techniques required for such chips are complex, as are the metrics to evaluate the performance.
The current effort in standardization of pre-designed macromodules, often referred to as "core cells," is expected to play a major role in making it possible for designers to build complete and customised wireless systems on a chip. However, the diversity of macromodules required in such wireless systems on a chip represents a special challenge in almost all aspects of IC/System design. In this tutorial we present the state of the art in designing such systems, focusing on the digital aspects for wireless systems and the design tools for wireless system design. The presentation is roughly divided into three parts: basics of wireless systems; digital VLSI design issues for wireless systems; design tools and techniques for hardware and software for wireless systems.

A2 Finding Design Errors and Locating Defects: The Same Detective Story

M. Abramovici — Bell Labs — Lucent Technologies, USA
R. Aitken — Agilent Technology, USA

Intended Audience:
Designers, researchers, managers, and anyone determined to find out where those puzzling errors are coming from.

Suggested Prerequisites:
Logic and circuit design, basic testing concepts, and a strong desire to discover if indeed the butler did it.

Summary
Locating design errors is a key factor in the logic verification process. Errors can result from incorrect specifications, erroneous logic implementation, and/or timing problems. Just as accurate error location is needed to meet time-to-market goals, accurate physical defect location is essential in improving the quality of the manufacturing process, and rapid identification of a defective replaceable part is essential in achieving a cost-effective field maintenance and repair process. The efficiency of all forms of diagnosis has a great economic impact on the cost of a product during its entire life-cycle (cost-of-ownership). Unlike fault and defect diagnosis, the practice of logic debugging is more of an engineering art than a science, and tools for diagnosis in the logic domain are still in experimental/prototype stage.
The tutorial will present methods for finding design implementation errors and techniques for locating defects in circuits. We will point out the many similarities between logical and physical diagnosis, and we will emphasise the common principles that guide each diagnosis process. After a review of the basic concepts in diagnosis, we will present the established defect diagnosis procedures — fault dictionary, post-test fault simulation, and guided-probe/E-beam testing. Then we will focus on advanced diagnosis topics such as critical path tracing, deductive analysis, diagnosis for delay-faults, AI techniques, and methods for locating defects such as opens, shorts, and leakage in transistor-level circuits.
In the logic domain, we will present several techniques to automatically generate design verification tests, and several methods that locate the logic error(s) that cause mismatches between implementation and specification. We will also discuss techniques that automatically correct the located errors. The models where errors are located range from VHDL to transistor-level designs. In addition to logic errors, we will discuss techniques to locate timing problems in circuits, and discuss design-for-debug methods to simplify this process.

B2 Embedded Memories in System Design — From Architecture to Design Technology

Nikil Dutt, U.C.Irvine, USA
Francky Catthoor, IMEC and Univ. of Leuven, B
Doris Keitel-Schulz, Infineon, D

Intended Audience:
The tutorial is intended for system and architecture designers dealing with systems which include large amounts of memories, and also for (system-level) design tool developers and researchers who look at design methodologies and tools which can support this type of application design.

Today's technologies allow the integration of significant amounts of DRAM memory for applications such as data buffering, picture storage, and program/data storage. In quarter-micron technology, chips with up to 128 Mbit of DRAM and 500 kgates of logic are eminently feasible. This enlarges the system design space tremendously since system architects are no more restricted to standard commodity DRAMs. We will discuss the market for embedded DRAM applications as well as the associated challenges. In addition, the main architecture, circuit and test issues will be introduced. Next, we address issues in system design technology and compilation for embedded data-dominated multi-media applications which use such embedded memories. Techniques addressed include formal data-flow analysis, reuse and access analysis, the most important memory related data-flow transformations and cache related allocation transformations, techniques for memory estimation, coarse-grain and fine-grain compiler transformations to improve locality, data partitioning and layout schemes, instruction selection and code compression.

C2 Real Time and Digital Signal Processing Embedded Software

Eric Verhulst, Eonic Systems, B
Peter Marwedel, U Dortmund, D

The key to achieve more reliable system designs is to have the capability to specify and design the application at a higher level of abstraction. This allows the designer to focus on the application unhindered by the peculiarities of a particular hardware. At the functional level, this can be achieved by the use of a real-time operating system providing multi-tasking programming approach and pre-emptive scheduling. Fundamentals and examples of real-time operating systems will be presented during the first part of this tutorial. This includes the description of various communication primitives. At a lower level, this can be achieved by the use of higher level languages that are platform independent. However, current compilers for higher level languages exploit architectural features of embedded processors poorly and generate only inefficient code. Generating efficient embedded systems from such languages requires new, code optimisation techniques, which will be presented in the second part of the tutorial. Using real-time operating systems and higher level languages, one can not only obtain more efficient systems is less time, but also more reliable and maintainable applications.

D2 RF Front-Ends: Design and CAD Tools

Piet Wambacq, IMEC, B
Georges Gielen, Katholieke Universiteit Leuven, B
Peter Kinget, Broadcom, USA

The growth of wireless services and applications into a consumer commodity increases the need for low cost solutions for wireless transceivers. The front-ends of these transceivers, comprising RF, analog and digital circuits, need to be designed with strong specifications: high performance should be combined with a low cost, small size and a low power consumption. This requires the development of intelligent front-end architectures. A key issue in the design of such front-ends with a short time-to-market is the use of CAD tools. This tutorial covers front-end design aspects as well as an overview of CAD tools, both for the architectural level and the circuit level. In this way, the tutorial is structured in three parts:

1. Wireless transceiver system design issues.
2. High-level simulation of front-ends of digital telecom transceivers.
3. CAD tools for analog integrated circuits.