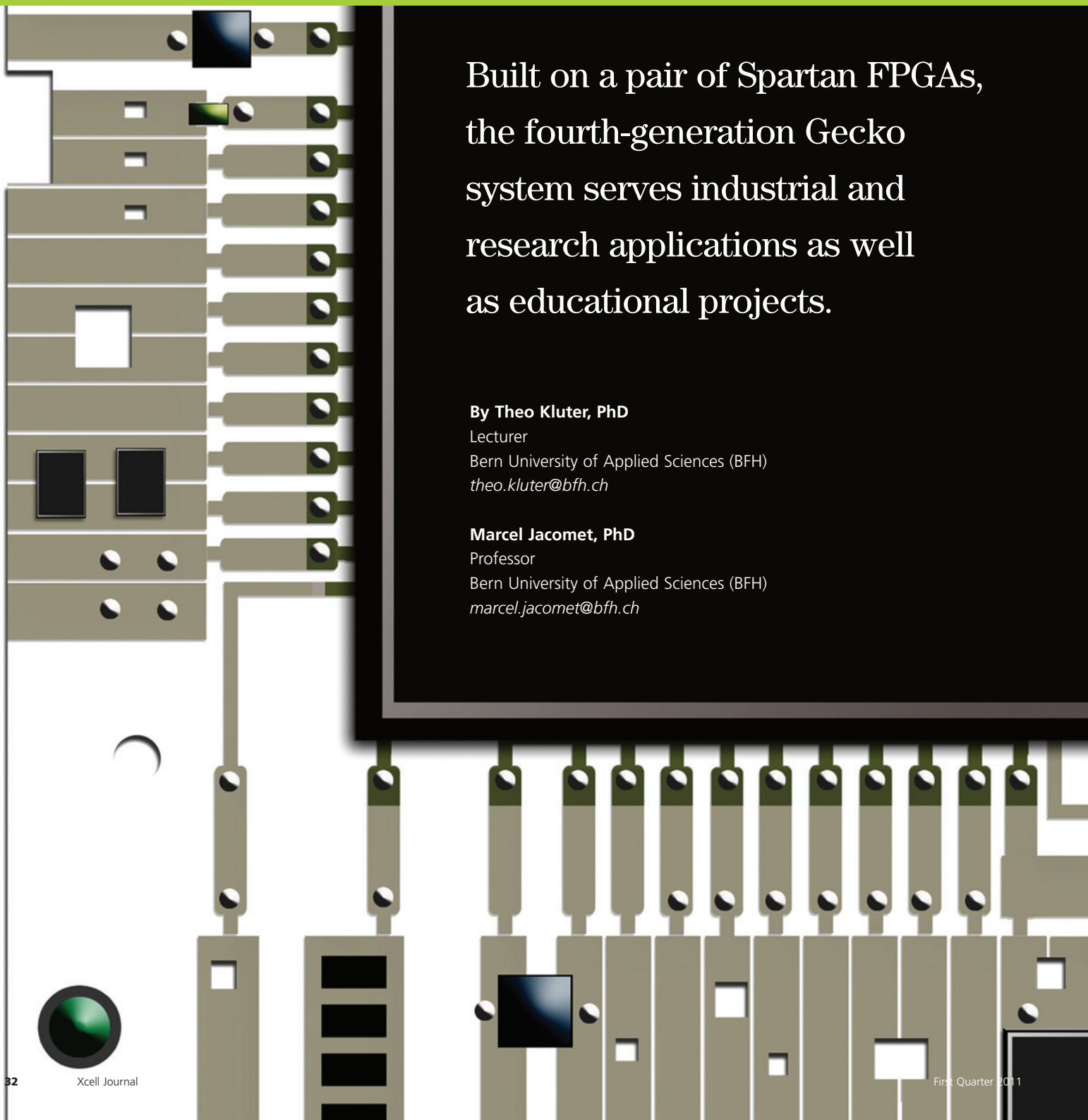
General-Purpose SoC Platform Handles Hardware/Software Co-Design



Built on a pair of Spartan FPGAs, the fourth-generation Gecko system serves industrial and research applications as well as educational projects.

By Theo Kluter, PhD
Lecturer
Bern University of Applied Sciences (BFH)
theo.kluter@bfh.ch

Marcel Jacomet, PhD
Professor
Bern University of Applied Sciences (BFH)
marcel.jacomet@bfh.ch

The Gecko system is a general-purpose hardware/software co-design environment for real-time information processing in system-on-chip (SoC) solutions. The system supports the co-design of software, fast hardware and dedicated real-time signal-processing hardware. The latest member of the Gecko family, the Gecko4main module, is an experimental platform that offers the necessary computing power to speed intensive real-time algorithms and the necessary flexibility for control-intensive software tasks. Adding extension boards equipped with sensors, motors and a mechanical housing suits the Gecko4main module for all manner of industrial, research and educational projects.

Developed at the Bern University of Applied Sciences (BFH) in Biel/Bienne, Switzerland, the Gecko program got its start more than 10 years ago in an effort to find a better way to handle complex SoC design. The goal of the latest, fourth-generation Gecko project was to develop a new state-of-the-art version of the general-purpose system-on-chip hardware/software co-design development boards, using the credit-card-size form factor (54 x 85 mm). With Gecko4, we have realized a cost-effective teaching platform while still preserving all the requirements for industrial-grade projects.

The Gecko4main module contains a large, central FPGA—a 5 million-gate Xilinx® Spartan®-3—and a smaller, secondary Spartan for board configuration and interfacing purposes. This powerful, general-purpose SoC platform for hardware/software co-designs allows you to integrate a MicroBlaze™ 32-bit RISC processor with closely coupled application-specific hardware blocks for high-speed hardware algorithms within the single multimillion-gate FPGA chip on the Gecko4main module. To increase design flexibility, the board also is equipped with several large static and flash memory banks, along with RS232 and USB 2.0 interfaces.

In addition, you can stack a set of extension boards on top of the Gecko4main module. The current extension boards—which feature sensors, power supply and a mechanical housing with motors—can help in developing system-on-chip research and industrial projects, or a complete robot platform ready to use for student exercises in different subjects and fields.

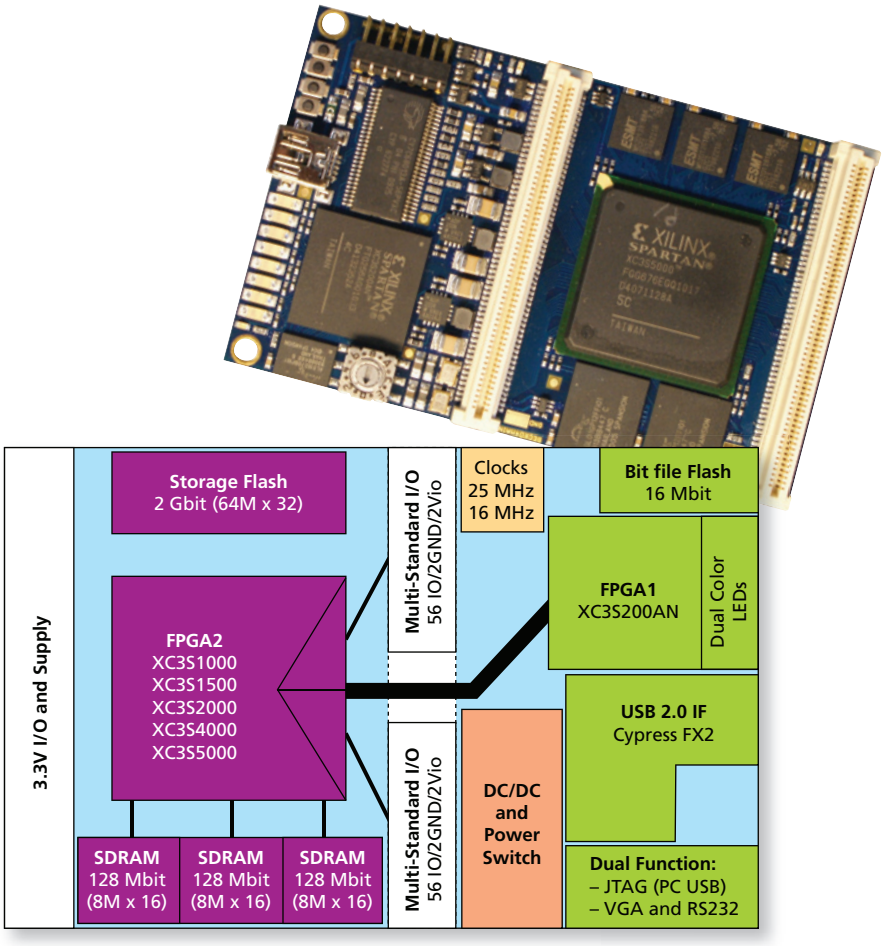


Figure 1 – The Gecko4main module is an eight-layer PCB with the components all on the top layer, reducing production cost.

FEATURES AND EXTENSION BOARDS

The heart of the Gecko4 system is the Gecko4main module, shown in Figure 1. The board can be divided into three distinct parts:

- Configuration and IP abstraction (shown on the right half of Figure 1)
- Board powering
- Processing and interfacing (shown on the left half of Figure 1)

One of the main goals of the Gecko4main module is the abstraction of the hardware from designers or student users, allowing them to concentrate on the topics of interest without

having to deal too much with the hardware details. The onboard Spartan-3AN contains an internal flash memory, allowing autonomous operation. This FPGA performs the abstraction by providing memory-mapped FIFOs and interfaces to a second FPGA.

Users can accomplish the dynamic or autonomous configuration of that second Spartan in three ways: via a bit file that is stored in the bit-file flash for autonomous operation, by the PC through the USB 2.0 and the USBTMC protocol, or by connecting a Xilinx JTAG programmer to the dual-function connecter.

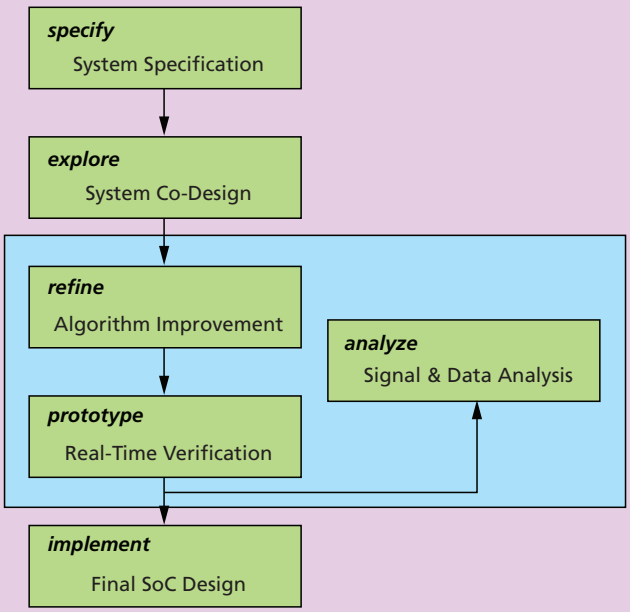
The main Spartan-3AN handles the first two options using a parallel-slave configuration protocol. The JTAG configuration method involves the direct intervention of the Xilinx design tools.

NEW TWISTS ON CHIP DESIGN

The inherently interdisciplinary character of modern systems has revealed the limits of the electronics industry's classical chip-design approach. The capture-and-simulate method of design, for example, has given way to the more efficient describe-and-synthesize approach.

More sophisticated hardware/software co-design methodologies have emerged in the past several years. Techniques such as the specify-explore-refine method give designers the opportunity of a high-level approach to system design, addressing lower-level system-implementation considerations separately in subsequent design steps. Speed-sensitive tasks are best implemented with dedicated hardware, whereas software programs running on microprocessor cores are the best choice for control-intensive tasks.

A certain amount of risk is always present when designing SoCs to solve new problems for which only limited knowledge and coarse models of the analog signals to be processed are available. The key problem in such situations is the lack of experience with the developed solution. This is especially true where real-time information processing in a closed control loop is needed. Pure-software approaches limit the risk of nonoptimal solutions, simply because it's possible to improve software algorithms in iterative design cycles. Monitoring the efficiency and quality of hardware algorithms is, however, no easy task. In addition, hardware algorithms cannot easily be improved on SoC solutions in the same iterative way as software approaches.



The specify-explore-refine design flow transforms into a specify-explore-refine-prototype-analyze design flow for SoC designs with real-time constraints.

In control-oriented reactive applications, the system reacts continuously to the environment. Therefore, the monitoring of different tasks is crucial, especially if there are real-time constraints. As long as precise models of the information to be processed are not available, as often occurs in analog signal processing, a straightforward design approach is impossible or at least risky. The flexibility of iteratively improving analog/digital hardware-implemented algorithms is thus necessary in order to explore the architecture and to refine its implementation. Again, this involves monitoring capabilities of the SoC in its real-time environment.

Thus, an important problem in the classical ASIC/FPGA hardware/software co-design approach is decoupling of the design steps with the prototyping hardware in the design flow, which inhibits the necessary monitoring task in an early design phase. Adding an extra prototyping step and an analysis step opens the possibility to iteratively improve the SoC design. The refine-prototype-analysis design loop best matches the needs of SoC designs with real-time constraints (see figure). If precise models of the plant to be controlled are missing, thorough data analysis and subsequent refinement of the data-processing algorithms are needed for designing a quality SoC. — *Theo Kluter and Marcel Jacomet*

While the latter option allows for on-chip debugging with, for example, ChipScope™, the former two provide dynamic reconfiguration of even a 5 million-gate-equivalent second FPGA in only a fraction of a second.

Many of our teaching and research targets require a fast communication link for features like hardware-in-the-loop, regression testing and semi-real-time sensing, among others. The Gecko4 system realizes this fast link by interfacing a USB 2.0 slave controller (the Cypress Fx2) with the USBTMC protocol. To both abstract the protocol and provide easy interfacing for users, the Spartan-3AN does protocol interpretation and places the payload into memory-mapped FIFOs.

Designers who do not need a large FPGA can use the Gecko4main module without mounting the left half of the PCB, reducing board costs significantly. In this setting, designers can use the Spartan-3AN directly to house simple systems—indeed, they can even implement a simple CPU-based system in this 200 million-gate-equivalent FPGA. For debugging and display purposes, the device provides eight dual-color LEDs, three buttons, an unbuffered RS232 connection and a text-display VGA connection at a resolution of 1,024 x 768 pixels (60 Hz). These interfaces are also available memory-mapped to the second FPGA.

BOARD POWERING

Due to the various usages of the Gecko4main module, powering must be very flexible. We provided this flexibility by means of the onboard generation of all required voltages. Furthermore, the module offers two possibilities of supplying power: the USB cable or an added power board. The attached power board is the preferred supply.

According to the USB standard, a device is allowed to use a maximum of 500 milliamps. The Gecko4main module requests this maximum current. If the design requires more than the available 2.5 watts, the board turns off automatically. The second way of supplying the unit is by attaching a power board. If these power boards are battery-powered, the Gecko4main module can function autonomously (as, for example, in a robot application).

If both a USB cable and a power board are connected (or disconnected), the Gecko4main module automatically switches seamlessly between the different supplies without a power glitch.

PROCESSING AND INTERFACING

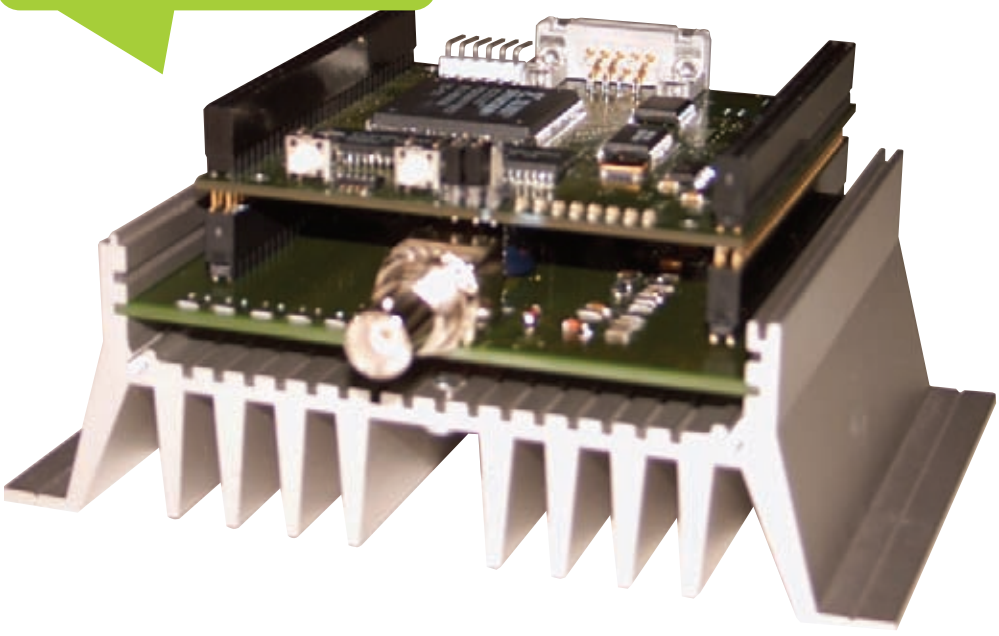
The processing heart of the Gecko4main module is the second FPGA, also a Spartan-3. The board can accommodate a Spartan-3 ranging from the "small" million-gate-equivalent XC3S1000 up to the high-end 5 million-gate-equivalent XC3S5000. The board shown in Figure 1 has an XC3S5000 mounted that was donated by the Xilinx University Program.

Except for the XC3S1000, all Spartan-3 devices have three independent banks of 16-Mbyte embedded SDRAM connected, along with 2 Gbits of NAND flash for dynamic and static storage. Furthermore, to interface to extension boards, the Gecko4main module provides two "stack-through" connectors. The one shown on the left in Figure 1 provides a standard 3.3-V-compliant bus including power supply pins and external clocking. This standard connector is backward-compatible with our Gecko3 system.

One of the advantages of the Xilinx FPGA family is its support for various I/O standards on each of the I/O banks of a given FPGA. To be able to provide flexibility in the I/O standard used on an extension board, the I/O connector shown in the middle of Figure 1 is divided into halves, each completely connected to one I/O bank of the FPGA. Furthermore, each half provides its own I/O supply voltage pin. If there is no attached extension board providing an I/O supply voltage, the I/O banks are automatically supplied by 3.3 V, placing them in the LVTTL standard.

For some applications, even the PCB trace lengths are important. To also provide the flexibility of implementing this kind of application, the Gecko4main module provides 38 out of the 54 available I/O lines that are length-compensated (+/- 0.1 mm).

Figure 2 – For an industrial project for a long-range RFID reader, a Gecko2 mainboard and an RF extension board inhabit a housing that also serves as a cooling element.



EXTENSIONS ARE AVAILABLE

The strength of the Gecko4main module lies in its reduction to the minimal required in terms of storage and peripherals for SoCs, and also in its stack-through principle. Both aspects allow for simple digital logic experiments (by simply attaching one board to the USB connector of a PC) or complex multiprocessor SoC emulation (by stacking several boards in combination with extension boards). The Bern University of Applied Sciences and several other research institutes have developed a variety of extension boards ranging from sensor boards up to complete mainboards. A full overview of our open-source Gecko system can be found at our Gecko Web page (*http://gecko.microlab.ch*) and on the OpenCores Web page (*http://www.opencores.org/project,gecko4*).

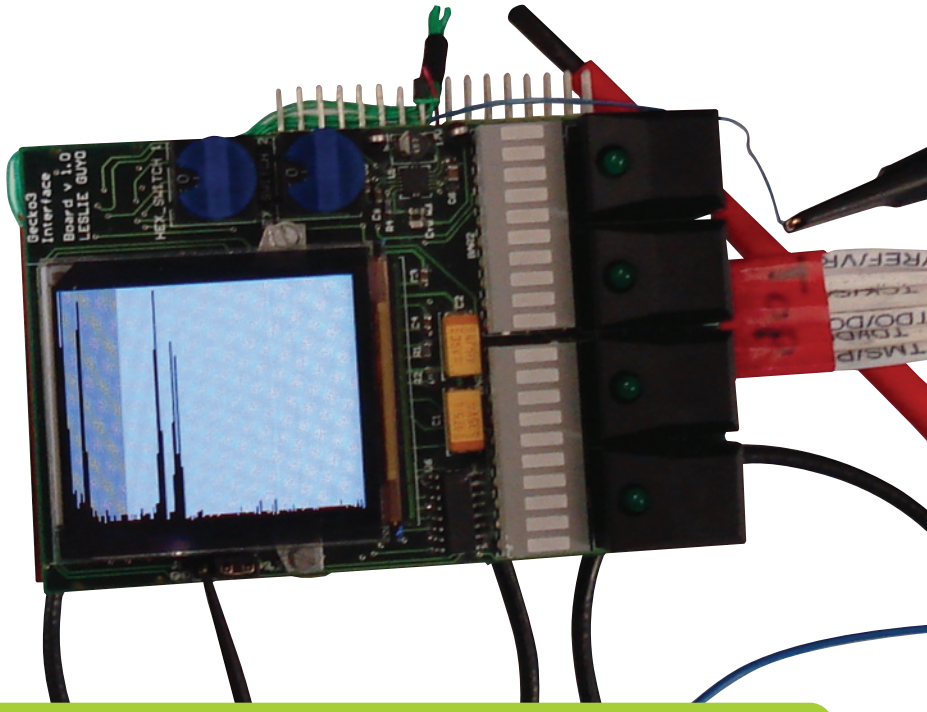


Figure 3 – An applied-research project is investigating high-speed hardware algorithms for optical-coherence tomography image processing. The image shows a depth scan (A-scan) illustrating various reflections of the sample as an intermediate result.

EDUCATION PLATFORM

The Gecko system is extensively used in the electronic and communication engineering curriculum at BFH. Different professors use the system in their laboratories, including the Institute of Human-Centered Engineering-microLab, in both the undergraduate and the master's curriculum. First-year exercises start with simple digital hardware designs using the Geckorobot platform, a small robot frame on which the Gecko3main can be attached. The frame provides two motors, some infrared sensors, two speed sensors and a distance sensor.

Robot control exercises in the signal-processing courses develop control algorithms for the two independent robot motors using the well-known MATLAB®/Simulink® tools for design entry, simulation and verification. A real-time environment allows the students to directly download their Simulink models into the Geckorobot platform and supports a real-time visualization of the control algorithm behavior of the running robot in the MATLAB/Simulink tools.

For this purpose, the students have access to the necessary Simulink S-functions and a MicroBlaze-based platform with the software and hardware IP interfacing the sensors and motors of the Geckorobot platform. Immediate feedback of the robot's behavior in real-world situations as well as in the MATLAB/Simulink design environment has a high learning value for the students. For more advanced student projects, we are planning drive-by-wire and cybernetic behavior of the robots in a robot cluster. Moreover, we are also developing a compressive-sampling Gecko extension board for audio signals through which students will learn—by listening—the impressive possibilities of this new theory as well as experience its limitations.

RESEARCH AND INDUSTRIAL PROJECTS

Outside the university, the Gecko platform is finding a home in applied research and industrial projects. In most cases researchers pair the Gecko4main module, for the computation-intensive tasks, together with application-specific extension boards. In an industrial project developing hardware algorithms for long-range radio frequency identification (RFID) readers, one group designed an RF power board as an extension board for the Gecko2main board (see Figure 2).

Currently, hardware algorithms for high-speed optical-coherence tomography (OCT) signal processing are under investigation. OCT is an optical signal-acquisition technology that makes it possible to capture high-resolution 3-D images in biological tissue using near-infrared light. In an ongoing applied-research project, a Gecko extension board with the optical interface is under development (see Figure 3). The Gecko4main module serves as a platform to develop high-speed OCT hardware algorithms.

TOOL CHAIN FOR GECKO

Our applied SoC design method for real-time information-processing problems is based on an optimal combination of approved design technologies

that are widely used in the industry (see sidebar). The presented design flow (Figure 4) is exclusively based on Xilinx and some third-party tools, and on adding dedicated interfaces and IP. Control algorithms developed in the MATLAB/Simulink environment can be compiled, downloaded and implemented in the Gecko4 platform for the real-time analyses by a simple and fast pushbutton tool environment. The tool chain represents a step toward the upcoming specify-explore-refine design methodology by adding a fast-prototyping feature. Fast prototyping is a crucial feature for an iterative improvement of control or signal-processing algorithms with feedback loops where precise models are not available.

Experience over the last 10 years has shown that the Gecko approach provides a good platform for educational, research and industrial projects alike. We believe that the capability of producing system prototypes very early in an SoC design flow is a major key to success in today's industry. It helps to secure the system architecture work, develop usable prototypes that can be distributed to internal and external customers and, last but not least, ensure that designers can start functional verification at a very early design phase.

Rapid prototyping meets the requirements for hardware/software co-design as an early function integration. It enables system validation and various forms of design verification. Very important is the real-time algorithm verification of analog/digital hardware and software realizations. We firmly believe that the Gecko4 platform enables all of these aspects of the new breed of complex SoC designs.

Acknowledgements

We want to thank the Xilinx University Program (XUP) for its generous donation of the FPGAs used for the production of both the third- and fourth-generation Gecko mainboards. Furthermore, we want to thank the OpenCores community for hosting and certifying our project. Finally, we thank all the students and research assistants who helped build the Gecko platform, in particular Christoph Zimmermann, who realized and was responsible for the third generation of the Gecko mainboard.

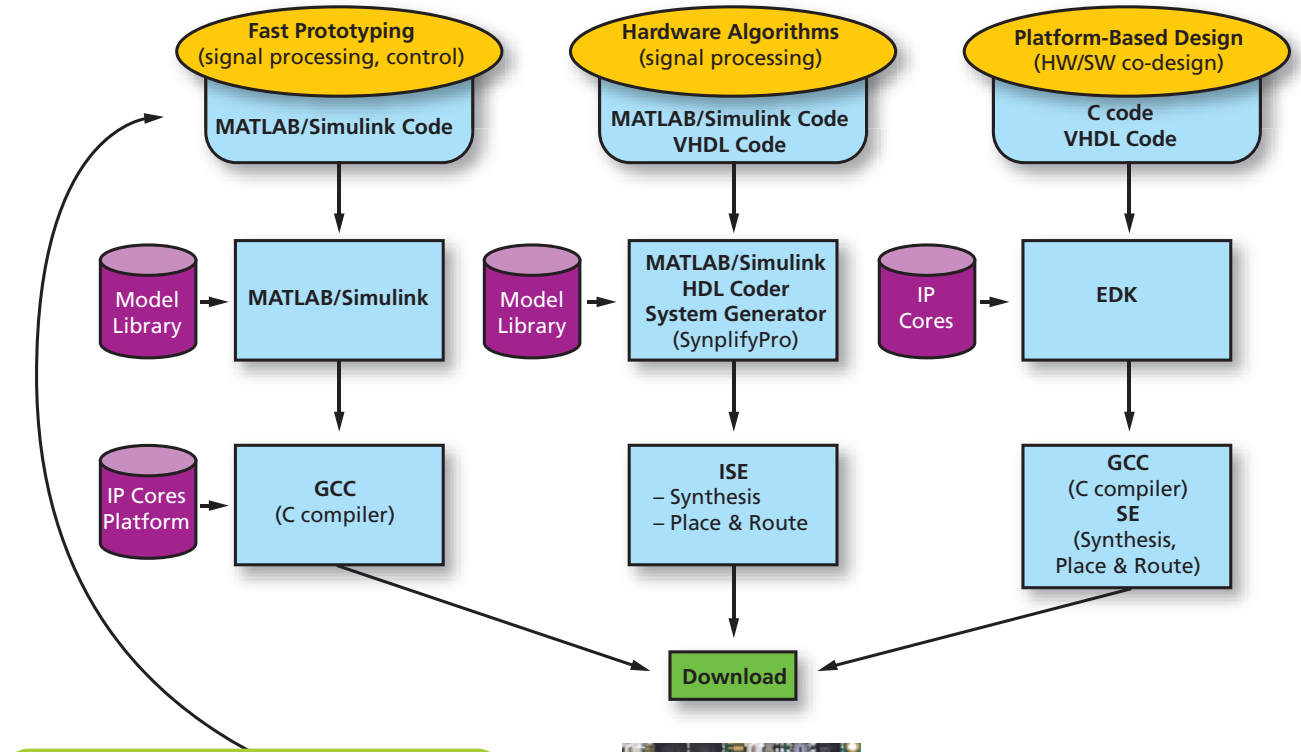


Figure 4 – General-purpose hardware/software co-design tool chain of the Gecko design platform includes a fast-prototyping environment for real-time analysis.