Guest editorial

Due to the pervasive nature of information technology (IT), information processing is finding its way from desktop applications to applications in *embedded systems*. In embedded systems, IT is integrated (embedded) into a larger product. IT is typically integrated such that the user hardly realizes that information processing takes place. Hence, the user will not buy a certain product because of the IT components that are used. The user does not care about clock frequencies and processor types employed in embedded systems, but he does care very much about the functionality. Examples of embedded systems include information processing subsystems in control systems, telecommunication equipment, consumer electronics, information appliances and smart homes. Embedded systems usually exhibit the following characteristics:

- they have to be dependable,
- the have to be efficient in terms of energy, weight, size and cost,
- they have to guarantee privacy,
- they come with customized user interfaces, not with generic screen-, keyboard- and mouse-based interfaces,
- many embedded systems are real-time systems (they have to meet real-time constraints).

Many of the early embedded systems were based on specialized hardware. Currently, designers try to implement most of the functionality in software and use processors. Key reasons for this include the flexibility provided by software as well as the difficulties of implementing complex functionality in hardware. The software generation techniques for embedded systems have to take the special characteristics of embedded systems into account. For example, they have to consider that most processors used in embedded systems are optimized for an application domain or even a certain application in order to provide the required efficiency. As a result, processors used for digital signal processing typically comprise the following special features:

- specialized instructions (for example, multiply/accumulate instructions),
- heterogeneous register sets (different registers have a different functionality),
- specialized addressing modes (for example, modulo addressing to be employed in implementing ring buffers)
- restricted parallelism:
  - parallel arithmetic, move and pointer update operations,
  - subword parallelism (for example, multimedia instructions supporting packed data types),
  - very long instruction word (VLIW) machines.

Compilers have to exploit these characteristics; otherwise they would generate inefficient code.

Specialized approaches are also required for other problems to be solved in embedded system design:

- specification languages should have appropriate language features for describing embedded systems,
- operating systems have to guarantee real-time constraints, must have a small memory footprint and hence should be customized for the application at hand,
- processors have to be optimized for certain application domains or even for certain applications,
- memory systems should be customized to make them power-efficient,
- verification techniques have to be able to verify real-time properties.

Work on compiler-related issues was stimulated in Europe by the CHIPS project funded by the European Commission in the mid-nineties. The first project workshop was held at Schloss Dagstuhl, Germany, and it was a real success. Therefore, a series of follow-up workshops was organized in the following years. This is a list of previous and planned future workshops:

Workshop	Location	Dates
1	Schloss Dagstuhl, Wadern, Germany	Aug. 31st to Sept. 2nd, 1994
2	Leuven, Belgium	March 18th to March 20th, 1996
3	Witten, Germany	March 4th to March 6th, 1998
4	St. Goar, Germany	Sept. 1st to Sept. 3rd, 1999
5	St. Goar, Germany	March 20th to March 22nd, 2001
6	Berlin, Germany	June 19 to June 21, 2002

The first workshops were called *Workshop on Code Generation for Embedded Processors*. Later, the scope was extended beyond code generation. The new scope is reflected in the new title *Software and Code Generation for Embedded Processors* (SCOPES). The sixth workshop will be organized together with LCTES (languages, compilers and tools for embedded systems). LCTES/SCOPES'02 will be a conference-level event.

SCOPES does not try to compete with well-established publication channels, such as conferences or journals. Rather, the best contributions at the workshop are considered to be candidates for being published in a book or as a journal in a paper. The book *Code Generation for Embedded Processors*, edited by G. Goossens and myself contains contributions from the first workshop. Other papers were published in a special issue of *Design Automation for Embedded Systems* in 1999 and in a special issue of *ACM TODAES* in 2000.

The current special issue is based on the best papers from the 5th workshop. The authors of thirteen papers were invited to contribute to this special issue. All submitted papers were reviewed by a team of international reviewers. More than 60 reviews were performed to select the papers to be included in this special issue. Conflicts of interest were handled by the editor-in-chief. Finally, a total of eight papers were accepted for publication in this issue of IEEE TCAD. I would like to thank all reviewers for performing the reviews and for enabling us to keep a rather tight schedule (the deadline for the submission of the final version of the contributions was just five months after the workshop!).

The first paper called "**Software Controlled Processor Speed Setting for Low-Power**" is written by A. Acquaviva, L. Benini and B. Riccó. The paper presents a software-controlled approach for minimizing the energy-consumption in real-time multimedia processing. The key idea is to adjust the clock rate of the processor such that the real-time constraints are met while the energy consumption is minimized.

The second paper is authored by L. Gauthier, S. Yoo and A. Jerraya. Its title is "**Automatic Generation and Targeting of Application Specific Operating Systems and Embedded Systems Software**". The paper describes a technique for reducing the size of a real-time operating system (RTOS) by configuring the RTOS such that only the functionality required for a certain application is implemented.

The third paper is a paper by J. Wagner and R. Leupers, entitled "**C Compiler Design for a Network Processor**". Network processors are processors optimized for network applications. They possess instructions that speed up processing bit-streams found in serial network protocols. The authors describe compiler-optimizations exploiting these instructions.

The energy and the time required to access memories is known to increase with the size of the memory. Therefore, it makes sense to use several smaller memories instead of a single large memory. An application of this idea to the design of caches is described by P. Petrov and A. Orailoglu in their paper "**Towards Performance and Power Effectiveness in Embedded Processors: Customized Partitioned Caches**", which is the fourth paper.

The next paper is written by S. Rajagopalan, S. Malik, S. P. Rajan, K. Takayama, S. Rigo and G. Araujo and is entitled "**A retargetable VLIW Compiler Framework for DSPs with Instruction Level Parallelism**". In this paper, the authors present an application of a standard VLIW compilation environment for exploiting parallelism provided by DSP architectures. One of the results of the paper is that a compiler framework for VLIW machines can indeed be used to exploit the irregular parallelism found in DSP

architectures.

A special case of design space exploration using a parameterized compiler is presented in the following paper called "**Analysis of the Influence of Register File Size on the Energy Consumption, Code Size and Execution Time**". The authors are L. Wehmeyer, M.K. Jain, S. Steinke, P. Marwedel and M. Balakrishnan. The paper is in line with the recent trend towards low-power design and the analysis of power and energy consumption is one of the strong points of this paper.

The seventh paper is authored by A. Hoffmann, A. Nohl, G. Braun, O. Schliebusch, T. Kogel and H. Meyr. Its title is "**A Novel Methodology for the Design of Application Specific Instruction Set Processors (ASIP) Using a Machine Description Language**". The authors describe the generation of a toolset for software generation from a description of the target processor. The level of automation that has been obtained with this approach is remarkable.

The last contribution is the paper "**Bitwidth Cognizant Architecture Synthesis of Custom Hardware Accelerators**" by S. Mahlke, R. Ravindran, M. Schlansker, R. Schreiber and T. Sherwood. The authors use bitwidth information for grouping operations into clusters and then allocate functional units to these clusters. Experimental results indicate that the gate-count of architectures synthesized with the PICO-synthesis tools can be substantially decreased with the proposed method.

I hope that this special issue will be one of your key references for work on processor- and software-based embedded systems. Please enjoy reading the special issue!

Peter Marwedel

Guest Editor

Dortmund, August 2001