Book Reviews

A career in system-level design research

Grant Martin
Tensilica

Reviewed in this issue

Embedded System Design: Modeling, Synthesis and Verification, Daniel D. Gajski, Samar Abdi, Andreas Gerstlauer, and Gunar Schirner (Springer, 2009, ISBN: 978-1441905031, 358 pp., $99).

My good friend Steve Leibson has coined Leibson's law: "It takes 10 years for any disruptive technology to become pervasive in the design community." Leibson's law has been validated by the time it took for design technologies such as automatic placement and routing, RTL synthesis, and some formal verification techniques to become widely accepted. All these technologies are commonplace today and have been for many years.

If it has taken one Leibson period for many techniques in the RTL-to-gates part of chip design to become accepted, what can we say about system-level design? Some areas of system-level design methodology began in the research community in the 1970s; many more started in the 1980s and are arguably only being accepted today or in the past few years. This double or triple Leibson period seems typical as designers move up from the smaller domain of devices and blocks to consider design as a whole system. Yet at least system-level techniques, such as virtual prototyping and high-level synthesis, appear to have taken off, or at least reached a good starting point of acceptance in the design community.

One of the most active researchers, teachers, and leaders in system-level design for many years is Daniel Gajski of the University of California, Irvine. Together with students (many who have become academic leaders and influential parts of the industry in their own right), colleagues, and partners, he has explored all parts of system-level design, both widely and deeply. Some of the concepts and tools developed during these efforts have formed an essential part of spinout companies, and others have been adopted by major users in the design community. Some of Dan Gajski's concepts, such as the Gajski Y-chart from 1983, have become standard tools with which to describe design methodologies.

So when it came time for Gajski and his colleagues to write about system-level design methodologies, in their new book *Embedded System Design: Modeling, Synthesis and Verification*, they had a wealth of material to work with. Indeed, figuring out what to say in a book on system design must have been a challenge, but one to which the authors have risen admirably.

This book deals with the need for both breadth and depth in describing system-level design by its structure. Two general chapters—an introductory one, and one on system design methodologies—and a concluding chapter on embedded-design practice bracket five more detailed chapters that cover the major topics of embedded-system design: modeling, system synthesis, software synthesis, hardware synthesis, and verification. These detailed chapters cover their topics in two ways: descriptions of the methods and technologies that address each topic, illustrated with examples drawn from the research of the authors and the authors' colleagues.

Industrial state of the art in system design focuses on three of the book's topics: modeling, hardware synthesis, and verification. In the modeling domain, the authors use layered decomposition to illustrate the modeling of both processors (whether programmable or hard-wired), and communications. The focus on layered communications modeling, using the ISO/OSI seven-layer model as a mechanism for decomposition and explanation, is welcome, because

0740-7475/10/$26.00 © 2010 IEEE Copublished by the IEEE CS and the IEEE CASS

many designers focus most of their attention on modeling function and spend too little time on modeling communications. As the authors show, the way communications is modeled often has a greater impact on the speed of system models than the way in which the individual function blocks are modeled. From the rich set of choices available to them, the authors focus on natural modeling levels such as transaction-level, bus-cycle-accurate level, and cycle-accurate level.

The hardware synthesis chapter is an excellent exposition of all the major techniques used in modern high-level synthesis tools. In this case, the authors draw on their own research and knowledge, illustrating each area with simple well-explained examples. This gives the reader a better understanding of high-level synthesis approaches that cannot be gained by using commercial tools alone, since the tools' inner workings are generally hidden from the user. The verification chapter is more of a survey of the standard methods of simulation and formal verification, illustrating the techniques most commonly used with some explanatory details, especially concerning types of formal verification.

One of the chapters those in industry will find most useful is the comprehensive survey of embedded-design practice in chapter 8. Many academic and commercial tools are covered, and the authors also provide a short yet useful overview for each one. Of course, changes in the commercial world outpace published data, and the recent acquisitions of VaST and CoWare by Synopsys, and Virtutech by Intel, will likely change some of the details that readers might see in the commercial tools space. Nonetheless, the book's general concepts will stand the reader in good stead in assessing commercial tool offerings.

If we turn, momentarily, back to the chapters on system and software synthesis, here we see expositions of what might yet come, even if not reflected in commercial tools and industrial practice. The authors draw on their considerable research in these domains to suggest what is possible regarding automation and tool assistance in creating complex system architectures and developing embedded software. Here the lag between advanced research and commercial practice is particularly large. In fact, the existence of this gap through the long history of system design research has been a reason why some people have dismissed the whole idea of system-level tools. But the recent adoption of system modeling and high-level synthesis tools has spotlighted what can and does work rather than what may yet be adopted.

I do wish there were more detail in the book on realistic industrial examples that use the techniques and tools the authors describe. Chapter 8 has a short case study on an MP3 decoder designed with some of the research tools explained earlier. Here I think a chapter on its own, with more details on the one example (for instance, more details on the modeling approach) and a couple additional examples, would have rounded out the book and given the reader better understanding of how these techniques can be used. Perhaps that would make a good companion volume.

THIS BOOK IS A great introduction to embedded-systems design methodologies and tools, both those in use and those yet to come. It is a good summary of the pioneering work of Gajski and his colleagues in system-level design, highlighting progress made and directions for future developments. It is also a good learning vehicle for those who wish to understand this area and apply it to their own work. Students in embedded design should find it a good introduction and survey. Even more important, it will be useful to those in industry wanting a better overview of what is possible. I heartily recommend it. ■

■ Direct questions and comments about this department to Grant Martin, 2424 Raven Road, Pleasanton, CA 94566; gmartin@ieee.org.