

Preface

Electronic System Level Design, or ESL Design, is generally understood as the set of tools, methodologies and design techniques applied to modern electronic systems design, from high-end chips and systems, to embedded devices, to integrated hardware and software systems. Given the complexity of current systems, advanced tools and methodologies have become absolutely essential to achieve the necessary productivity, quality, cost and performance expected in a design process. One of the important tenets in ESL Design is the need for early design analysis. This is done mainly through high-level modeling and simulation, performance and power analysis and functional verification, before committing the design to lower-levels of abstraction aimed at synthesis and optimization. This is especially true for complex systems involving different types of components such as processors, custom blocks and software. In fact, it may be totally impractical to simulate such systems at a low-level of representation such as register-transfer level, due to extremely long simulation times. High-level models are simpler to write, understand, optimize and debug than lower-level models, and they can simulate significantly faster. The more the design can be refined, optimized and verified at a high-level of abstraction, the higher the overall design productivity, the better the quality and consequently the lower the cost of the final result. However, high-level models and development environments are not without their own difficulties. It is not simple to write a high-level model at the appropriate abstraction level which will result in the best trade-off between architectural details and simulation speed. For this end, researchers have formalized different abstraction levels at different levels of architectural and timing accuracy. Depending on what types of design analysis need to be done, one abstraction level may be more suitable than another. This adds to the modeling complexity, and in the worst case, if multiple models need to be developed, it may start reducing the productivity advantage of a high-level modeling methodology. A high-level modeling and simulation framework is a complex software engineering challenge. Different types of models, such as a processor model, custom blocks and application software, need to be compiled, linked, executed and debugged together. This goes far beyond the correct individual modeling of a block using a high-level language. A successful high-level design methodology depends heavily on how well automated this framework is. SystemC is a system-level specification and design language (based on C++

classes) that has been widely adopted for high-levels of abstraction modeling. SystemC also emergence of transaction-level models (TLM) as an abstraction layer and modeling style capable of fully separating the computation part from the communication part of a model. Despite the widespread acceptance and use of SystemC and TLM, it is still very complex to create models which achieve a perfect balance between the required architectural detail for the desired types of analysis and simulation speed. This book addresses the very issues raised above. It presents a high-level design methodology, support tools and framework capable of full system modeling and design exploration, including components such as processors, custom blocks and application software. One of the main contributions of the book is the detailed description of ArchC. ArchC is an architectural description language (ADL) which allows designers to model instruction-set architectures and automatically map them into SystemC simulatable descriptions. ArchC is an ADL capable of detailed modeling of instructions and supporting processor architectures. Since ArchC generates an executable SystemC model, it can be fully integrated with other SystemC models using transaction-level interfaces, which allows the specification, modeling and simulation of complete systems. The book also explains in detail how the overall software environment works, including how the models are compiled, linked, executed together and debugged as a platform. Detailed examples using single and dual core platforms, coupled with custom accelerators and running real life applications are presented. Power modeling is also addressed in this book. Authors describe an interesting extension to SystemC called PowerSC, which allows switching and power information to be gathered and computed during and integrated with the functional simulation. Important to readers should the fact that the platform presented is Open-Source and available for download, whereas comparable systems available today are proprietary. This will certainly help researchers and developers alike to jumpstart their modeling efforts by using a readily available platform and development tools. Readers interested in a good overview of ESL methodologies as well as those interested in practical implementation details of architectural-description languages, platform modeling and support tools will be well served by this book.

Visiting Professor Reinaldo A. Bergamaschi
Odysci, Brazil/USA
University of Campinas, Campinas, Brazil



<http://www.springer.com/978-1-4020-9939-7>

Electronic System Level Design

An Open-Source Approach

Rigo, S.; Azevedo, R.; Santos, L. (Eds.)

2011, IX, 146 p., Hardcover

ISBN: 978-1-4020-9939-7