Chapter 8

EMBEDDED DESIGN PRACTICE

Both commercial and academic tools are available for the design of embedded systems. These tools come in three categories: system-level design, software design, and hardware design.

In this chapter, we will discuss the tools and frameworks available for these various examples of system design. We will also present examples of embedded system design and results for applications, such as JPEG encoder and an MP3 decoder. These results demonstrate the potential impact of the embedded system modeling, synthesis and verification technologies that have been discussed in this book.

8.1 SYSTEM LEVEL DESIGN TOOLS

The semiconductor revolution would not have been sustainable without the help of Electronic Design Automation (EDA) tools. Historically, the breakthrough of EDA came with the availability of the first Computer-Aided Design (CAD) tools for hardware synthesis (see Section 8.3). As we move to higher and higher levels of abstraction, new classes of tools gradually emerged with each new level. In recent years, we have seen a push towards development of so-called Electronic System-Level (ESL) tools. However, while there are many approaches that claim to provide ESL solutions, such as C-to-RTL tools implementing high-level synthesis of a single hardware unit (described in more detail in Section 8.3), true system-level solutions have to span the complete design space across hardware and software boundaries.

As described in detail throughout this book, a system-level design flow is typically separated into two parts: a frontend and a backend. The system design
frontend takes a description of the application and target architecture at its input. Applications are given in some MoC to describe the desired system behavior to be implemented. Target architectures can be given in the form of architectural constraints, associated parameters, architecture templates or complete pre-defined system-level netlists. In the frontend, application computation and communication is then mapped onto and implemented on the selected or synthesized target architecture. In the process, Design Space Exploration (DSE) is performed to optimize design metrics under a set of constraints. At the output of the frontend, models of the system at various levels of abstraction are generated for virtual prototyping of the system design. Predominantly, such system models will be TLMs described in some SLDL such as SystemC. Models can be simulated or analyzed to provide feedback about the feasibility and quality of the generated design. In addition, modeling guidelines such as the SystemC TLM standard [150] promise to enable easy exchange of component or design models between companies or design divisions and across tool chains.

In the backend, high-level system descriptions are then further synthesized down to a hardware or software implementation for each PE in the system. ESL design flows thereby rely on the availability of corresponding software or hardware synthesis tools (see Section 8.2 and Section 8.3, respectively). On the software side, final target binaries for each processor are produced. On the hardware side, high-level synthesis of behavioral, C-based component models down to RTL descriptions is performed. In both cases, synthesized PE models can be re-integrated into system TLMs for cycle-accurate co-simulation with the rest of the systems. On the software side, binaries are executed in an ISS that is integrated into the overall system simulation environment. On the hardware side, RTL or gate-level models in SLDL form are inserted for this purpose. As a result, a virtual prototype of the system platform is generated.

In the end, however, the desired result at the output of a system-level design flow is a physical system prototype or a system implementation that is ready for further manufacturing. Therefore, generated software binaries should be ready to be directly loaded into target processors and RTL models should be created in the form of standard HDL code (e.g., VHDL or Verilog) such that they can feed into traditional logic and physical synthesis processes.

Overall, being based on existing commercial or proprietary backend tools, the goal of system-level design tools is to develop and apply design automation techniques to the steps in the frontend. At any level, the first set of tools to always emerge are modeling and simulation solutions that allow designers to capture models and execute them in a validation environment. Consequently, most currently available commercial system-level approaches are focused on providing models and simulators either at the application, SLDL/TLM or HDL/RTL/ISS level. Looking ahead, academic research, in contrast, is aimed at the development of subsequent system-level synthesis and verification tools, which build