Chapter 2
Model-Driven Design and ASM-Based Validation of Embedded Systems

Angelo Gargantini
Università di Bergamo, Italy

Elvinia Riccobene
Università degli Studi di Milano, Italy

Patrizia Scandurra
Università di Bergamo, Italy

ABSTRACT

In the embedded system and System-on-Chip (SoC) design area, the increasing technological complexity coupled with requests for more performance and shorter time to market have caused a high interest for new methods, languages and tools capable of operating at higher levels of abstraction than the conventional system level. This chapter presents a model-driven and tool-assisted development process of SoCs, which is based on high-level UML design of system components, guarantees SystemC code generation from graphical models, and allows validation of system behaviors on formal models automatically derived from UML models. An environment for system design and analysis is also presented, which is based on a UML profile for SystemC and the Abstract State Machine formal method.

INTRODUCTION

In the Embedded System (ES) and System-on-Chip (SoC) design area, conventional system level design flows usually start by writing the system specification and developing a functional executable model. The system is refined through a set of abstraction levels, toward a final implementation in hardware and software. Nowadays it is an emerging practice to develop the functional model and refine it with the C/C++/SystemC languages (Gröetker et al., 2002). The hardware part of the system goes down to the Register Transfer Level (RTL) for the synthesis flow, while the software part can be either simulated at high (functional or transactional) level or compiled for an Instruction Set Simulator (ISS).

Recently, the increasing technological complexity coupled with requests for more performance...
and shorter time to market have caused a high interest for new methods, languages and tools capable of operating at higher levels of abstraction in the design process. Such an improvement has been achieved by exploiting visual software modeling languages like Unified Modeling Language (UML) (UML, 2005) to describe system specifications and generate from them executable models in C/C++/SystemC.

In the embedded system domain, the UML has been used mainly for documentation purposes. There are, indeed, some criticisms that are widely shared towards the use of the UML as modeling language. The UML includes a rich set of modeling elements that can be used for a wide range of applications, whereas modeling specific applications, as embedded systems and SoCs, would be easier using a more specialized (domain-specific) notation representing the basic elements and patterns of the target domain. Furthermore, the UML offers the possibility of designing the same system from different views (by different UML diagrams). Even if such a feature can be used to capture related aspects of a design from different prospective, it may result in inconsistencies between diagrams, when the use of UML is not coupled with a rigorous design discipline (Chen et al., 2003). Therefore, the use of the UML for the specific application domain of embedded system and SoC design requires:

- A domain specific language, called profile, built on the basic UML infrastructure and including domain-specific building blocks (defined using stereotypes) to model common design elements and patterns,
- A methodology defining how and when the profile notation should be used.

According to these requirements, in (Riccobene et al., 2005a, Riccobene et al., 2005b, Riccobene et al., 2007a) a UML2 profile for SystemC is presented, which lifts the SystemC language to the UML modeling level. It represents a set of structural and behavioral constructs for designing at system-level with automatic encoding in SystemC. With this approach, the standard UML language is applied as high-level system modeling language and operates in synergy with a lower-level system language. To foster the methodology in a systematic and seamless way, in (Riccobene et al., 2007b) a SoC design process, called UPSoC (Unified Process for SoC), is presented as extension of the conventional Unified Process (Arlow & Neustadt, 2002) for the SoC refinement flow. The UPSoC process has been developed following the principles of the Object Management Group (OMG) Model-driven Architecture (MDA) (Mellor et al., 2004) -- a framework for Model-driven Engineering (MDE) (Bezivin, 2005), according to which a system development process should rely on modeling, model transformations and automated mapping of models to implementations. Indeed, the UPSoC is based on model-to-model and model-to-code model transformations from abstract models towards refined (but still abstract) models and/or SystemC code models.

Even if modeling methods based on the UML and UML profiles have receiving increasing interest in the embedded system and SoC contexts as they allow modeling at a higher level of abstraction than that of system-level languages, UML-based design methods still suffer the lack of effective formal analysis (validation & verification) techniques. Formal methods and analysis tools have been most often applied to low level hardware design. But, these techniques are not applicable to system descriptions given in terms of programs of system-level languages (like SystemC, SpecC, etc.), since such languages are closer to concurrent software than to traditional hardware description (Vardi, 2007), and the focus in the literature so far has been mainly on traditional code-based simulation techniques. Moreover, classical analysis techniques are not directly applicable to UML-based design methods that lack of a strong mathematical foundation necessary for formal model analysis.