UML 2 and SysML: an Approach to Deal with Complexity in SoC/NoC Design

Yves Vanderperren, Wim Dehaene
Dept. of EE, ESAT-MICAS, Katholieke Universiteit Leuven, B-3001 Leuven, Belgium

Abstract

UML is gaining increased attention as a system design language, as indicated by current standardization activities such as the SysML initiative and the UML for SoC Forum. Moreover the adoption of UML 2 is a significant step towards a broader range of modeling capabilities. This paper provides an overview of the impact of these recent advances on the application of UML for SoC and NoC development, proposes a model-driven development method taking benefit of the best techniques recently introduced, and investigates the design of power efficient systems with UML.

1. Introduction

Larger scale designs, increased mask and design costs, first-time-right requirements and shorter product development cycles motivate the application of innovative system-on-chip (SoC) and network-on-chip (NoC) methodologies which tackle complex system design issues. There is a noticeable need for design flows towards implementation starting from higher level modeling, system level analysis capabilities, design space exploration means and links between HW and embedded SW design. Still the concepts of system level modeling, analysis and refinement are not well understood and subject to various interpretations, due to the wider scope of application at higher abstraction levels. Several entry points into the flow can be defined from high level modeling languages. UML for SoC is attracting significant interest last few years [1, 2, 3]. The fusion of some of the best ideas from the digital hardware and software engineering domain provides indeed major benefits [1]. A common and structured environment for capturing the requirements, a unified view of the system, an environment complementary to SystemC are some of the key advantages identified while using UML. Significant issues remain however. The most important one is the need for customizing UML towards the specificity of SoC design. Despite the success of UML in its main areas, development tools are slow to realize UML full potential in a SoC context. The UML for SoC Forum in Japan is working on a specific UML profile for SoC to solve these shortcomings and improve tool interoperability [4].

In the remainder of this paper the important features offered by UML 2 and SysML are summarized. Next it is illustrated how these features can be efficiently applied to SoC/NoC design.

2. UML 2.0

The UML 2 specification is in its finalization phase. It represents a significant milestone in the evolution of UML to support complex systems. Some concepts from UML-RT/ROOM, such as the notion of ports, are now standard. The representation of the structural aspects of a system becomes clearer with the composite structure diagrams. The capability to model dynamic and parallel behavior has drastically improved with UML 2, e.g.

- Sequence diagrams can have fragments. This allows the representation of loops, alternative sequences and parallel message exchanges.
- Activity diagrams are based on Petri nets and improve the representation of concurrent flows of operation.
- Timing diagrams are new and stress the importance of time when showing the interaction between objects and their change in state.

Beyond these notation enhancements, the Model Driven Architecture initiative of OMG promotes the development of platform independent (PIM) UML models which are mapped onto platform specific (PSM) UML models. This is a particular approach to model-driven development, which emphasizes the role of (executable) models.

3. SysML

The Systems Modeling Language (SysML) is a joint initiative of OMG and the International Council on Systems Engineering (INCOSE) [5]. The purpose is to refine UML and define a general purpose modeling language for systems engineering. This covers complex systems which include a broad range of heterogeneous domains, in particular hardware and software. Several similarities exist between the methods used in the area of systems engineering and complex SoC/NoC design, such as the need for accurate requirements capture, system specification and simulation, system validation and verification. The SysML initiative will result in a profile extending UML 2 to systems which are not purely software based. It introduces a requirement diagram to structure the requirements and link these to test procedures. It proposes a unified and domain-neutral solution to represent the system architecture and flows of information by means of SysML assembly, which refines the composite structure diagram.

4. An example of model-driven development

We consider the case of an 802.11g wireless LAN SoC which must be compatible with two different modulation schemes in the 2.4 GHz band. It is assumed that the MAC is an available platform to which the physical (PHY) layer capability is appended (hardware and software).

Capturing requirements efficiently is the first step. It is crucial to start on a sound basis the design of multi-disciplinary systems. SysML requirements diagrams are a standard means to structure the IEEE specifications and improve the traceability to the test suite. Next, use case (UC) analysis helps to build a firm understanding of what the system should do. The focus is on capturing the goals of the system (the PHY) from an external and black-box point of view (the MAC and the radio). Primary (expected) and secondary (failure handling) system responses are documented. Use cases are basically textual but can be organized and visualized using UML diagrams (UC, state, activity, sequence diagrams). If written with rigor and care, UC analysis provides a means for regular and effective test during system and architecture design as well as at product delivery.

The design process gradually moves from UC analysis to architecture modeling. The system is opened up, broken down into large-scale subsystems to which clear responsibilities are assigned. In this example, we identify

- the Radio-PHY interface: automatic gain control, clear channel assessment, data transfer with rate adaptation
- the PHY-MAC interface: processing of MAC requests
- the two modems (OFDM and CCK/DSSS)

SysML assembly describes the system in terms of its structure and the data flow. Sequence diagrams annotated with real-time constraints (QoS requirements) realize the use cases (functional requirements), validate the proposed high-level architecture and stress the interactions between the subsystems. Timing diagrams help to spot real-time constraints. We refine the top-level use cases and consider the subsystems similarly, while algorithmic work follows the conventional line. Gradually implementation details (e.g. memories, time-critical blocks to be implemented in hardware) are introduced. An executable model of the architecture is built. It is well-known that SystemC enables the integration of an early version of the embedded software and a model of the hardware. UML is applied to document the SystemC model as classically done with software artifacts. This documentation helps furthermore to describe the behavior of the blocks implemented in hardware, which improves the communication with the HW engineers. The architecture model is validated by means of the use cases (functionality) and the algorithmic model (performance) [1]. It provides a verified, executable specification as input to the detailed implementation.

This approach can be extended to NoC. UML is indeed able to cope with the increased system complexity. Unreliable communication between the tiles of the NoC is supported by lost messages in sequence diagrams. The question of mapping applications to the fabric can be associated to the problem of mapping PIMs to PSMs.

As power consumption is becoming a technological concern with major importance, UML should also facilitate the development of power-efficient systems. The presented approach allows the specification of power needs at system level to identify power management solutions early in the design. Applying the UML-RT profile [6] beyond its original focus of time aspects, a battery can be modeled as a passive resource with specific attributes (e.g. energy density), while dynamic power management solutions are equivalent to a resource manager applying a resource control policy to maintain the battery lifetime. However this is not sufficient. Message exchanges, tasks etc. need proper annotation to capture the energy cost associated with the communication or the computation, in order to be able to apply power optimization techniques such as [7, 8]. This eventually raises again the issue of standardization of the annotations, which should be tackled by future versions of the UML for SoC profile.

5. Conclusions

The recent advances in UML and SysML present a valuable milestone for the application of UML to modern chip design, as illustrated by our example. Further formalization and standardization are needed. Also power related aspects need increasing attention to ensure the applicability of UML to SoC/NoC design in the future.

6. References

[1] Y. Vanderperren et al., "A SystemC Based System On Chip Modelling and Design Methodology", in "SystemC: Methodologies and Applications", W. Mueller, W. Rosenstiel, J. Ruf, Kluwer Academic Publishers, 2003

[2] "New SoC Design Methodology Based on UML and C Programming Languages", *Fujitsu press release*, April 2002

[3] G. Martin, "UML for Embedded Systems Specification and Design: Motivation and Overview", DATE 2002

[4] T. Hasegawa, "An Introduction to the UML for SoC Forum in Japan", 1st UML for SoC workshop, DAC 2004

[5] "Systems Modeling Language: SysML", Draft Specification v0.85, SysML Partners, www.sysml.org

[6] "UML Profile for Schedulability, Performance, and Time Specification", v1.0, OMG, www.omg.org

[7] K. Lahiri, A. Raghunathan, S. Dey, "Communication Architecture Based Power Management for Battery Efficient System Design", DAC 2002

[8] R. Bergamashi, Y. Jiang, "State-Based Power Analysis for Systems-on-Chip", DAC 2003