An Incremental Specification Flow for Real Time Embedded Systems

Alex Niemegeers, Alcatel SRD, F. Wellesplein 1, B-2018 Antwerp, Belgium (alex.niemegeers@alcatel.be)
Gjalt de Jong,, Alcatel, SRD, F. Wellesplein 1, B-2018 Antwerp, Belgium (gjalt.de_jong@alcatel.be)

Abstract

The fast growing complexity of today's real time embedded systems necessitates new design methods and tools to face the problems of integration and validation of complex systems. We have combined a number of different hardware and software methods into one system level design method. The proposed flow is based on UML concepts, executable specifications and platform based design.

1. INTRODUCTION

Integration and validation of complex systems is a bottleneck in the traditional design flow. We have used a new system level design approach in the design of a high speed VDSL modem. To tackle the integration and validation problem, we have combined a number of different hardware and software methods into one system level design method [1,2]. Our flow uses UML concepts [1] for specifying both the hardware and embedded software components (documentation flow), executable specifications and platform based design.

2. DOCUMENTATION FLOW

A number of well-defined specification documents describe the system at a well-defined abstraction level, and provide the necessary blueprints that document the system throughout the flow. Our flow adds a number of system level descriptions above the traditional detailed module level descriptions and hence provides a better system overview. The documentation consists of a number of graphical representations at different hierarchical levels. At the system level, we specify the interaction of the system with its environment. At an intermediate level, the realization of the use cases is specified by a static view (block diagram) and a dynamic view (message sequence chart). The graphs have a well-defined syntax and are accompanied by textual descriptions. Our documentation flow is inspired by the UML software methodology, and is here applied to the specification of real time embedded systems, consisting of software *and* hardware components. The system level descriptions allow the validation of a number of design aspects before any code is available. By specifying a selected number of interface transactions in the graphical representations and a selected number of communication attributes, we prepare the executable specification of the functional architecture at the token based performance level.

3. EXECUTABLE SPECIFICATION

In a second step, the blueprints are used to write the executable specification of the system. The written specifications are replaced with a set of documents and a piece of executable code. The choice of the abstraction level [3,4] of the executable specification is a key issue in our system level design flow. This choice determines what effort is spent in building the model, but also what design aspects can be validated and/or verified before the RTL design phase. Our executable specification describes the functional architecture at the token based performance level. In this model a strict separation is made between the description of the communication of an entity and its functionality. This allows us to abstract from the data-flow functions and validate the control aspects (i.e. the trickiest aspects in most of our applications) only, with a considerable gain in effort and simulation time. The split between data-flow and control-flow allows a further division of the specification and integration problem in smaller sub-problems.

4. PLATFORM BASED DESIGN

To validate the hardware-software partitioning of our design before the RTL phase, our system level design flow includes the mapping of the functional architecture, defined in the executable specification, onto a physical architecture consisting of a general purpose processor and a digital signal processor. The architecture is common to a wide range of access telecommunication systems. Our platform supports the abstraction level we have defined for the previous steps.

5. CONCLUSION

Our flow divides the specification, integration and validation problem into a number of smaller sub-problems. Key in our flow is the entry abstraction level, the use case driven approach, the orthogonality of control flow and data flow aspects and platform design.

REFERENCES

[1] UML, http://www.rational.com/uml/
[2] I. Bolsens, et al., 'Hardware/Software Co-design of Digital Telecommunications Systems.' Proc. of the IEEE, 85(3):391-418, March 1997.
[3] RASSP Taxonomy Working Group, 'RASSP VHDL Modeling Terminology and Taxonomy', revision 2.3, June 23, 1998, http://www.atl.external.lmco.com/rassp/taxon/rassp_taxon.html
[4] VSI initiative, http://www.vsi.org