Linking Codesign and verification by mean of E-LOTOS FDT

Pierre Wodey, Fabrice Baray
LIMOS
BP 10125 F-63173 Aubière Cedex France
{Pierre.Wodey, Fabrice.Baray}@isima.fr

Abstract

This paper presents an approach for linking Design and Verification environments in the context of hardware/software codesign of complex systems, based on refinement steps of the system implementation. We describe the advantage in the integration of verification in the refinement process for detecting easily and early design errors. Generally, design tools are based on a specific internal representation of the system, and classical approach for linking design and verification consists in translating this representation into a verification dedicated representation. The originality of the proposed approach in this paper consists in applying in parallel the refinement transformations on two specific representations of the system : one dedicated to the implementation and synthesis and the other dedicated to verification. A simple example shows the advantages of using such an approach when considering model checking verification techniques : the size of the model is significantly decreased. In this study, we consider the COSMOS Codesign environment, the OPEN/CAESAR verification toolbox and the E-LOTOS language as verification dedicated representation of the system.

1 Introduction

Designing complex systems requires ever more hardware and software parts, where some components are specifically designed for the application and some components are reused (microprocessors, real time kernels, ...). In order to help the designers of such systems in their tasks, Codesign methods and tools were introduced [GM93, Wol94, GV95, ELLSV97]. The basic architecture and design flow of such methods and tools are the following :

- the specification of the system, which is independent of the implementation technology (hardware/software) of the different parts of the system;
- the use of tools based on the specification which allow designers to simulate and verify the system at a high level of abstraction ;
- the hardware/software partitioning consisting in defining what should be implemented in hardware or in software ;
- the synthesis of the hardware and software parts leading to the definition of a virtual prototype [VCJ96] described generally in C and VHDL languages ;
- the validation and evaluation of the virtual prototype by means of cosimulation [VNPJ96], performance analysis ...

When considering open target architectures, the transformation from the specification to the virtual prototype is performed interactively by a sequence of refinement steps. At each step, a decision is given by the designer, the codesign tool integrating automatically this decision by transforming the description of the system.

In such methods, verification tasks are performed either by formal verification on the entry specification level or by cosimulation at the virtual prototype level. But, as the refinement is decided by the designer, errors can be introduced in the system at each refinement step. The detection of such errors is performed at the virtual prototype level. This tasks is difficult and uncertain, as :

- deadlocks induce generally active loops in the generated model,
- the link between the generated code and the initial description in not easy to perfom,
- there can be several errors in the design at the virtual prototype level, so they are several decisions to modify, but they are difficult to identify,
- the virtual prototype describes the system at a low level of abstraction, the description is thus complex.

Thus, there is a need to perform validation/verification of the system after each refinement step. So, if an error is introduced, the responsible decision is directly identified. With such an approach, the designer obtains a virtual prototype satisfying some properties which are no more to be verified by cosimulation.

Some Codesign tools integrate built-in verification in order to evaluate properties of the system during the refinement process. The link between the design representation of the system and the verification representation of the system is done by a translation of languages : from the internal representation to a verification dedicated representation.

Formal verification is based either on theorem proving techniques or on model checking techniques. In both cases the size of the system representation, either as a logic formula or as a state based model, is a problem which limits the use of the corresponding verification technique. A state based model allows also validation by simulation, bisimulation or properties verification expressed in temporal logic.

Our work consists in linking a codesign tool with a verification tool at each refinement step. The considered codesign tool is COSMOS from TIMA laboratory, and the verification tool is OPEN/CAESAR from INRIA. The originality of this work consists in the use of two formalisms in parallel during the refinement process instead of a translation oriented solution. The first formalism is dedicated to the implementation of the system (COSMOS built-in SOLAR format) and the second is dedicated to the validation/verification (E-LOTOS language : one entry language for OPEN/CAESAR). In this approach we do not need to translate one formalism into another, but we apply in parallel the transformations on both descriptions of the system.

Actually, the implementation oriented formalism has a semantic that is not dedicated to verification, it includes low level constructs. So, the verification description, obtained by translation includes the model of these low level operations. We show that the proposed approach leads to a simpler model which is more accurate and efficient for validation/verification.

In section 2, we present COSMOS Codesign environment, in section 3 the verification techniques and the introduction to OPEN/CAESAR tools set.

Then, in section 4, we present the proposed solution for linking design and verification tools in comparison with a classical translation oriented interface.

Finally we show the compared results on a simple example where a designer decision introduces a deadlock in the system.

2 COSMOS Codesign environment

In our study, we consider the COSMOS tool developed at TIMA laboratory, which is characterized by :

- heterogeneous entry descriptions of the system (SDL, VHDL),
- the use of an intermediate format SOLAR describing the system and the communication channels among processes,
- the implementation of processes in hardware or software and the implementation choices of communications are performed by an iteration of refinement steps decided manually by the designer,
- automatic generation of the C-VHDL virtual prototype from the completely refined SOLAR description of the system,
- cosimulation environment of the virtual prototype.

This tool is thus a designer aid for the generation of an implementation of the systems from its specification. This aid consists in automatic generating the communication mechanisms implemented among the various processing elements and the automatic generation of synthetizable VHDL code for hardware processes and C code for software processes.

The choices introduced by the designer concern :

- implementing a given processing element in hardware ;
- implementing a given processing element in software ;
- implementing a communication channel among processing elements with a given protocol ;
- organizing the communication architecture by merging channels and/or flattening levels of hierarchy.

Each introduced choice induces a transformation on the SOLAR representation. The intermediate format SOLAR is a hierarchical state machine oriented model completed with communication channels representation. The communications are based on hardware signals assignment semantics (i.e. waveforms).

3 Verification

The verification techniques can be classified in two categories :

- theorem proving methods : the system is described as a proposition using a given logic. These verification tools prove complex theorems from axioms and by application of inference rules,

- formal model based verification methods : the verification is performed on a formal model of the system characterizing its behaviour (states and evolution). The models are for instance Petri nets, automaton or graph model of the system. The first kind of verification, the model checking, consists in finding deadlocks or in comparing two models. A second kind of verification consists in verifying that the systems satisfies a property, described in temporal logic, by computing it on the model.

The characteristics of the tools based on formal model are the following :

- the considered model ;

- the properties which can be verified : deadlocks, livelocks, ... ;

- the eventual complementary analysis tools : simulation, ...

Model checking techniques are generally full automatic processes. But, it is common to come up against the state explosion difficulty. One cause of this problem is the parallel composition of interacting processes.These techniques are accurate for control oriented systems rather than data dominated systems.

In the context of our study, the verification tool is needed to detect design errors introducing deadlocks in the system for communication implementation choices. What we need is thus a tool that :

- detects efficiently deadlocks in a system ;

- is as automatic as possible (push button function).

According to that, a model checking technique is evidently more accurate.

We choose OPEN/CAESAR tool [Gar98], developed at INRIA (France) because it has good performance and can reach a large number of states.

It is a complete tools set for verification (deadlock detection, model checking), and for execution analysis. It allows exploration like traces, reachability, simulation, random execution and test generation. In addition, it is open to different description languages, soon for the new E-LOTOS language.

4 Linking Codesign and Verification

Our work consists in linking COSMOS with OPEN/CAESAR. For the OPEN/CAESAR input formalism we choose E-LOTOS [Que97] language because :

- it allows to describe a system at different levels of abstraction ;

- with regard to LOTOS, E-LOTOS is much more pragmatic at the data type and behaviour levels ;

- the formal semantic of the language allows to develop efficient compilers and verification tools (Traian compiler in development at INRIA),

- the notion of modules allows to describe generic components which can be reused in a system ;

- its expressiveness allows the verification of very complex systems.

4.1 Translation oriented link

A classical link technique would consist in translating the COSMOS intermediate format (SOLAR) into the future admissible input formalism to OPEN/CAESAR (E-LOTOS).

The drawback of using a translation technique stands in the fact that the implementation oriented formalism includes low level operations that lead to a complex verification model. Some abstractions are to be defined in order to obtain a E-LOTOS model that really contains the eventual deadlocks.

When considering graph oriented verification techniques whose limit stands in the explosion of the number of states it is necessary to generate a description as simple as possible. But, with such a translation technique, the limits of CADP tools box is quickly reached.

4.2 Multiple languages oriented link

In order to obtain a more accurrate model for verification, we propose a parallel oriented technique (figure 1), where the transformations are equivalently applied on both implementation and verification oriented models. The verification description in E-LOTOS is based on a high abstraction level of the system at each refinement step. The description includes only the essential features and characteristics of the system which are useful for verification.

The constraint of this approach is that the properties of the verification model must be the same as those of the implementation model according to the validation and verification which are performed. This is ensured by the

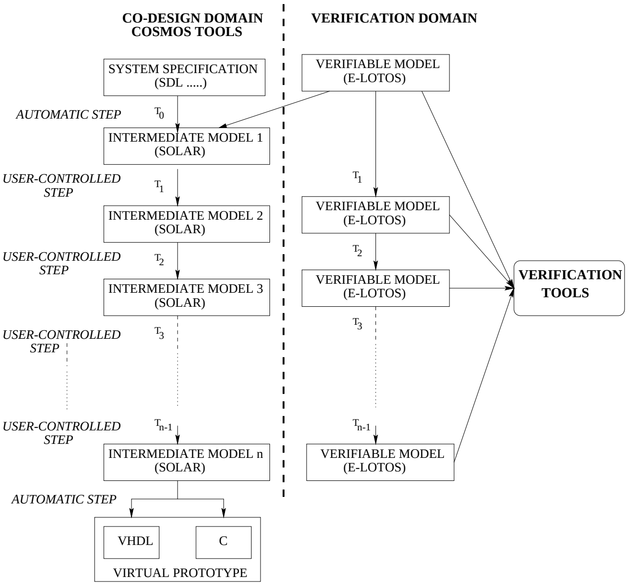
CO-DESIGN DOMAIN
COSMOS TOOLS
VERIFICATION DOMAIN
SYSTEM SPECIFICATION (SDL .....)
VERIFIABLE MODEL (E-LOTOS)
AUTOMATIC STEP
T0
INTERMEDIATE MODEL 1 (SOLAR)
USER-CONTROLLED STEP
T1
INTERMEDIATE MODEL 2 (SOLAR)
VERIFIABLE MODEL (E-LOTOS)
USER-CONTROLLED STEP
T2
INTERMEDIATE MODEL 3 (SOLAR)
VERIFIABLE MODEL (E-LOTOS)
VERIFICATION TOOLS
USER-CONTROLLED STEP
T3
USER-CONTROLLED STEP
Tn-1
INTERMEDIATE MODEL n (SOLAR)
VERIFIABLE MODEL (E-LOTOS)
AUTOMATIC STEP
VHDL
C
VIRTUAL PROTOTYPE

Figure 1. Parallel link technique between design and verification

precise definition of equivalent transformations rules for the two representations of the system.

The equivalent transformations are summarized as follow :

- for selection of hardware or software implementation of a processing element, the verification transformation is the identity function (no transformation has to be performed) ;

- for selection of a communication implementation, the transformation consists in the introduction of a minimal high level E-LOTOS model of the implementation protocol and in the correction of communication protocols for the processing elements ;

- for merge of communication channel, the corresponding E-LOTOS channels (gates) are merged by applying merging rules.

A minimal high level E-LOTOS description of a channel implementation is an E-LOTOS description that leads to a minimal number of states in the generated model.

Such a strategy conduces in a verification description as simple as possible, characterized by a high level of abstraction and minimal size of data structures.

5 Case study

We consider a simple case study that illustrates the proposed approach for linking codesign and verification. The results are also compared with those coming from a translation oriented approach. At first, we briefly present the example and one step of refinement in the COSMOS codesign environment.

5.1 System structure and principles

The system is composed of three processes, two producers and one consumer. The consumer manages the communication ratio of each producer, and according to this ratio, either :

- accepts data from any producer ;

- accepts only data from the producer whose communication ratio is low.

The communications are unidirectional, from the producers to consumer. The system is described in SDL. With this description, a first intermediate model is automatically generated with COSMOS tools (see method in figure 1). Then, we consider a refinement step consisting in the implementation by a FIFO communication model among the three components. The two producers write their data in the same FIFO. By this action, the designer introduces a deadlock in the system. The E-LOTOS description obtain by translation from SOLAR is composed of about 300 lines and is not described in this paper.

After one step, which consists in choosing a FIFO file for the communication channel, this next description is generated. It contains a high level generic description of the FIFO, and the new specification of the system.

5.2 Results and discussion

In order to verify the description dedicated to verification, the OPEN/CAESAR tool-box is used. But unlike LOTOS, the E-LOTOS language is not yet fully supported. So equivalent LOTOS descriptions are generated. Table 1 illustrates, for a FIFO of size one and two and for the translation oriented (*Translated O.* in the table) and parallel generated descriptions (*Parallel G.* in the table) :

- the size of the labeled transition system (LTS) before and after reduction. The reduction is performed by Aldebaran tool by application of a strong equivalence relation conserving deadlock properties of the system ;

- the depth (number of steps) of the deadlock sequence.

The results N.A. (Not Available) mean that they could not be obtained within a computing time of 6 hours on an Ultra Sparc 30.

	before red.		after red.		depth
	states	trans.	states	trans.	
Translated O. FIFO size 1	1693	3787	1469	3448	295
Parallel G. FIFO size 1	78	97	14	18	5
Translated O. FIFO size 2	N.A.	N.A.	N.A.	N.A.	N.A.
Parallel G. FIFO size 2	232	434	31	52	6

Table 1. CADP verification results

The large difference between the two results shows that a translation oriented approach for linking codesign and verification reaches his limit earlier than the proposed technique with two parallel formalisms. This approach allows then to manage more complex systems for verification because it reduces significantly the effect of states explosion.

With the proposed approach, the graph generation and deadlock search are performed quiet instantaneously. So the verification is performed efficiently by a push button like function. This approach has to be applied on larger systems in order to evaluate its limits in complexity. This will be studied in the future.

6 Conclusion

In this paper, we propose an approach to link the Codesign tool COSMOS and the OPEN/CAESAR validation/verification toolbox. COSMOS is based on refinements of the system and verification is needed when the designer chooses the implementation of communications. We intend to implement the verification as a push button function of the system.

Classical link approach would consist in translating the design format of COSMOS (SOLAR) into an admissible input language of OPEN/CAESAR (E-LOTOS in our case). But we propose to apply the refinements in parallel on the design description and on an E-LOTOS description dedicated to the verification. The aim of this approach is to manage a verification description at a high level of abstraction in order to lessen the size of the generated model for verification. Such a high level description can not be generated by a translator from SOLAR to E-LOTOS, because the implementation oriented semantic is composed of low level communication mechanisms.

Finally, we show on a simple example the compared results obtained by a translation approach and the multiple languages approach. These results indicate a significant gain in the size of the model and thus also in computing time for the verification.

Our future work will consist in optimizing the description of high level communication descriptions. Case studies on large systems are also under work.

References

[ELLSV97] S. Edwards, L. Lavagno, E.A. Lee, and A. Sangiovanni-Vincentelli. Design of Embedded Systems: Formal Models, Validation, and Synthesis. In Giovanni De Micheli, editor, *Proceedings of the IEEE, Special issue on Hardware/Software Co-design*, volume 85, pages 366–390. The institute of electrical and electronics engineers, inc., March 1997.

[Gar98] Hubert Garavel. OPEN/CAESAR : An Open Software Architecture for Verification, Simulation and Testing. In *TACAS'98, Tools and Algorithms for the Construction and Analysis of Systems, Lecture Notes in Computer Science*, 1998.

[GM93] R.K. Gupta and G. de Micheli. Hardware-Software Cosynthesis for Digital Systems. *IEEE Design & Test of Computers*, 10(3):29–41, September 1993.

[GV95] D.D. Gajski and F. Vahid. Specification and Design of Embedded Hardware-Software Systems. *IEEE Design & Test of Computers*, 1995.

[Que97] Juan Quemada. *Final Commitee Draft on Enhancements to LOTOS*. ISO/IEC JTC1/SC21/WG7 Project 1.21.20.2.3, 1997.

[VCJ96] C.A. Valderrama, A. Changuel, and A.A. Jerraya. Virtual Prototyping For Modular And Flexible Hardware-Software Systems. *Journal of Design Automation for Embedded Systems*, 1996.

[VNPJ96] C.A. Valderrama, F. Nacabal, P. Paulin, and A.A. Jerraya. Automatic Generation of Interfaces for Distributed C-VHDL Cosimulation of Embedded Systems: and Industrial Experience. In *7th International Workshop on Rapid Systems Prototyping, (Greece)*, June 1996.

[Wol94] W.H. Wolf. Hardware-Software Co-Design of Embedded Systems. *Proceedings of the IEEE*, 82(7), July 1994.