

Properties Coverification for HW/SW Systems

Mostafa Azizi¹, El Mostapha Aboulhamid¹ and Sofiène Tahar²

¹ *Département d'Informatique et de Recherche Opérationnelle, Université de Montréal*

² *Department of Computer and Electrical Engineering, Concordia University
{azizi, aboulham}@iro.umontreal.ca, tahar@ece.concordia.ca
Montreal, Canada*

Abstract. The coverification of a given HW/SW system consists of checking whether the implementation of the software and hardware parts and their integration fulfill or not some or all the specification requirements of this system. In the case of a distributed model, the SW and HW system blocks are described respectively by HLL (High Level Language) and HDL (Hardware Description Language) codes. When dealing with a unified model, both SW and HW components are implemented in the same language such as Java. In this paper, we propose a tool that allows designers to specify the properties of their systems in CPL (Coverification Properties description Language), and performs the coverification by simulation. The engine of this tool is implemented using the Java programming language and is mainly based on the management of threads.

I. Introduction

To avoid an eventual lengthy iterative codesign process of a given HW/SW system (that is often a result of an unsuccessful integration of hardware and software parts), performing an early coverification of the HW/SW integration before prototyping is strongly required. Coverification techniques currently used in the microelectronics industry are based on cosimulation environments, where SW and HW elements are respectively described in HLL (High Level Language) and HDL (Hardware Description Language), and simulated with specific simulators. The properties of a HW/SW system are included in its implementation codes, and are not separately expressed. The management of these properties (such as properties review, editing, description, checking, etc.) is hence time-consuming and inefficient.

We present in this paper a tool for the coverification of HW/SW systems. This tool is thread-oriented and currently assumes a unified description model of HW/SW systems. The software part of the HW/SW system and its interaction with the hardware part are described as a set of communicating threads. The system properties first are described in a CPL code (Coverification Properties description language), and then converted to Java threads in the objective of performing the coverification process.

The rest of this paper is organized as follows: Section II briefly reviews previous work related to coverification. Section III presents our proposed thread-oriented tool of coverification. Section IV illustrates the description of HW/SW system properties using CPL and their coverification. Finally, Section V concludes the paper.

II. Related work

The coverification process is tightly coupled with the cosimulation. Known coverification tools such as CVE-Seamless [1], Eaglei [2] and Ptolemy [3] are mainly based on cosimulation techniques.

The cosimulation approach simulates software and hardware parts and their interactions, known as virtual integration in the sense that this latter is made long before the prototyping process. In the case of a unified model [4, 5], only one simulator is used to perform cosimulation but when dealing with a distributed model [6, 7, 8, 9, 10], two or several simulators are required depending on how many languages are used to describe the software and hardware parts of the HW/SW systems. Other related work presents application cases such as the coverification of DPLL (Digital Phase Locked Loop) at NORTEL [11], and the HW/SW coverification performance estimation of a 24 embedded RISC core design at SIEMENS [12]. On the other hand, there are few papers in the literature that deal with the coverification using formal verification methods such as model checking [13, 14].

III. Multithreading-based coverification

When the design of a given HW/SW system is approximately achieved, designers integrate them to obtain the implementation of the whole system. The software part contains the original SW-code of the system and some data. The original SW-code encompasses memory addressing, hardware interfacing, and eventual functions decided after partitioning to be implemented by software. The hardware part is a set of circuit modules, registers, memories, IP blocks (blocks with Intellectual Properties), etc. The interface between hardware and software parts is built around a processor that runs software codes and manages the hardware signals. The problem of coverification is to verify if this global implementation satisfies the specification requirements of the system? This leads to check concurrently if software and hardware implementations satisfy their corresponding software and hardware specifications, and if the HW/SW integration respects the requirements of the global specification. Due to their complexity and heterogeneity, real HW/SW systems challenge current methods of verification and simulation.

The basic idea of our coverification technique consists of mixing the cosimulation process with a set of properties to be checked in concurrence [5]. This technique is summarized into four steps. The first one consists of organizing the software part, as a set of threads. The second step stores in observable registers the hardware signals, which are parameters of the HW/SW properties. The third step puts the system properties in threads and manages them. The fourth one finally starts cosimulating the system augmented by its properties. The execution flow of these steps is depicted on Figure 1. Details of the implementation and the description of the system properties are given in the following sections.

IV. Description and coverification of HW/SW properties

A. *Specification of properties in CPL*

The specification requirements of a HW/SW system are expressed as a set of properties described in CPL. CPL is a simple language that we have developed specially for the description of HW/SW properties. These latter will be validated by the coverification process. The parameters of coproperties (HW/SW properties) might belong to the HW part, to the SW part, or to both of them. In CPL, we specify which properties are concurrent and which can be executed together sequentially. For instance, in the example of Figure 2, $p1$ and $p2$ are concurrent between each other but $p2$ and $p3$ are not. $p2$ and $p3$ are independent in terms of concurrence and they are treated as a single property. The rest of the properties are considered by default as sequential and they are managed as a composed property. We note that CPL is strongly suitable for the description of properties at the behavioral level, and it could be used at the RTL-level as well.

Figure 1:
Execution flow of the steps
of our coverification
approach

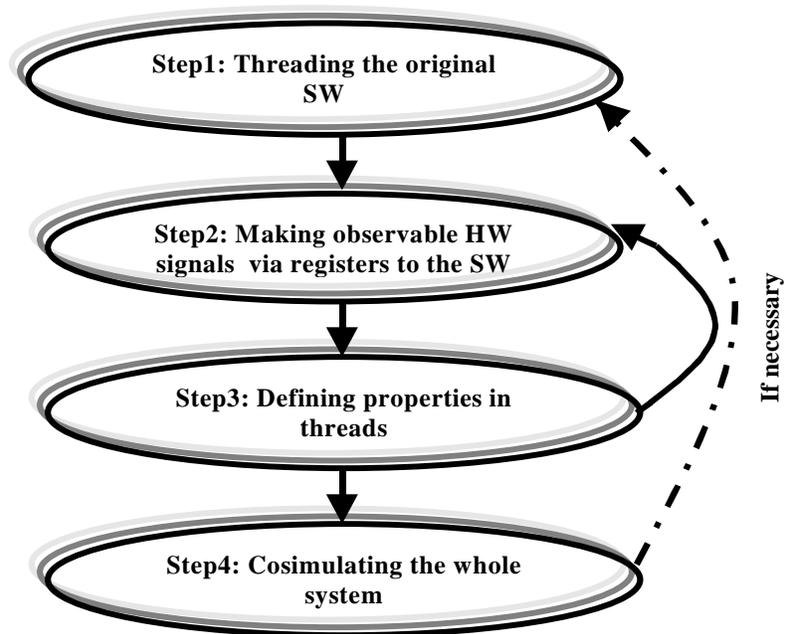


Figure 2:
Example of properties
described in CPL

```
p1 property ( flow_input - flow_output * sqr(2) <= 0.5 ) end

p2 property if ( temperature == 199 ) ( current <= 0.2 ) end

p3 property if ( signal_1 == 199 ) not( signal_2 == 134 ) end

p4 property switch ( state )
{
case S1 : not(signal_ack & signal_start);
case S2 : (red_light & ~(green_light & orange_light));
default : (reg_R = 1);
}
end

concurrent ( p1, p2);

sequential (p2, p3);
```

B. Coverification of properties

The CPL code describing the properties is translated to a subset of Java code strongly thread-oriented. The properties declared as concurrent in the CPL code are manipulated each one as a thread; those identified as sequential are translated and treated as one thread. The rest of the properties that are neither parameters of “concurrent (...)” instruction nor of “sequential (...)” instruction are put in one thread and validated as a composed property. The threads obtained after translation are added to the system implementation in order to perform functional simulation of the whole code. For the

implementation of the HW/SW system, we use a Java unified model in which the HW part is seen as a set of registers with read/write accesses. The draft tool of this coverification process is depicted on Figure 3.

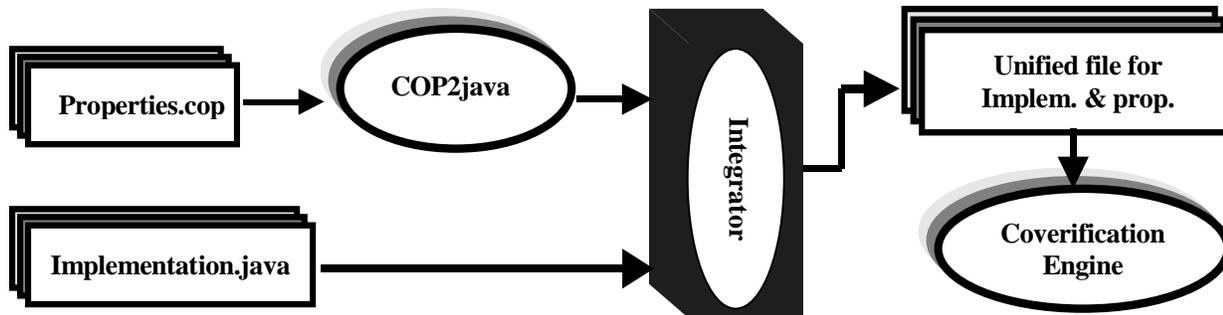


Figure 3: Global view of the semi-automatic coverification tool

V. Conclusions

In this paper, we have presented a tool for the coverification of HW/SW systems. This tool is based on the multithreading concept and cosimulation technology. The properties of a given HW/SW system are described in CPL, a simple language developed specially for the description of coverification properties. The whole implementation of the HW/SW system and its properties appears as a set of communicating threads. As perspective, we plan to complete this tool and make it fully automatic in the sense of managing properties and generating efficient test vectors for each property.

References

- [1] Mentor Graphics Corporation, "CVE-Seamless", <http://www.meng.com/>, 1998
- [2] View Logic, "Eagle", <http://www.Viewlogic.com/>, 1998
- [3] University of California at Berkeley, "Ptolemy project", <http://ptolemy.eecs.berkeley.edu/index.htm>, 1999
- [4] R. K. Gupta, C.N. Coelho Jr. and G. De Mecheli, "Synthesis and simulation of digital systems containing interacting hardware and software components", 29th Design Automation Conference (DAC'92), June 1992
- [5] M. Azizi, E.-M. Aboulhamid and S. Tahar, "Multithreading-based coverification of HW/SW systems", proceedings of the international conference of Parallel and Distributed Processing Techniques and Applications (PDPTA'99), Las Vegas, Nevada, USA, June 1999
- [6] D. Becker, R. K. Singh and S. G. Tell, "An engineering environment for HW/SW cosimulation", proceedings of the 29th Design Automation Conference (DAC'92), June 1992
- [7] D.E. Thomas, J.K. Adams and H. Schmit, "A model and methodology for HW/SW codesign", IEEE Design and Test of Computers, September 1993
- [8] J. Rowson, "HW/SW cosimulation", proceedings of the 31st Design Automation Conference (DAC'94), San Diego, June 1994
- [9] H. De Man, I. Bolsens, B. Lin, K. Van Rompaey, S. Vercauteren and D. Verkest, "Codesign of DSP systems", Kluwer Academic Publishers, 1997
- [10] C. A. Valderrama, F. Nacabal, P. Paulain and A. A. Jerraya, "Automatic generation of interface for distributed C-VHDL cosimulation of embedded systems: an industrial experience", proceedings of the 7th IEEE International Workshop on Rapid systems prototyping, June 1996
- [11] E. Hunnell and M. Lyons, "Coverification goes from cutting edge to mainstream: DPLL design demonstrates the viability of today's tools", Electronic Design, June 22, 1998
- [12] T. W. Albrecht, J. Notbauer and S. Rohringer, "HW/SW coverification performance estimation & benchmark for a 24 embedded RISC core design", Proceedings of 35th Design Automation Conference (DAC'98), pp. 808-811, San Francisco (California), June 1998
- [13] L. Lavagno, A. Sangiovanni-Vincentelli and H. Hsieh, "Emdedded systems co-design: synthesis and verification", Kluwer Academic Publishers, 1997
- [14] R. Kurshan, V. Levin, M. Minea, D. Peled and H. Yenigun, "Verifying HW in its SW context", proceedings of the IEEE/ACM International Conference on Computer Aided Design (ICCAD'97), San Jose, November 1997