TTool for DIPLODOCUS: An Environment for Design Space Exploration

Ludovic Apvrille

Ludovic.Apvrille@enst.fr

GET/ENST – System-on-Chip laboratory – France

http://labsoc.comelec.enst.fr/

Abstract

"TTool for DIPLODOCUS" is an open-source UML-based toolkit for the design space exploration of system-on-chip. Its strength relies in its abstraction and formal verification capabilities.

1. Introduction

Our contribution relies on an environment for designing System-on-Chip (SoC) applications at a high level of abstraction. Our approach follows four principles: use of a high-level well-known language (UML), separation of application and architecture, data abstraction and at last, use of simulation and static formal analysis techniques on abstract models.
Based on those four principles, we have developed a three-step design space exploration methodology for SoC:

1. The system is modelled in terms of communicating tasks. Modelling is focused on control part of the application.
2. The underlying hardware architecture is modelled using generic hardware components, such as microprocessors, buses, and hardware accelerators.
3. At last, the system is mapped onto the architecture, and the efficiency of the so-built system is evaluated.

At step (1) and (3), the system may be verified using simulation or formal analysis techniques. All this methodology is under implementation within a UML profile called DIPLODOCUS[1]. This contribution focuses only on step (1). We sketch out the main capabilities of DIPLODOCUS and its related toolkit named TTool.

2. DIPLODOCUS: Application Modelling

A UML profile customizes the UML language for a given domain of systems. DIPLODOCUS relies on UML class and activity diagrams for the application modeling step of our methodology

- A DIPLODOCUS class diagram is used to model the system in terms of classes (named *dclasses*, or *tasks*) using the following communication features: channels, events, and requests. Data abstraction is a key point: channels do not convey values, but only a number of samples (data abstraction). Events are used to signal a change, and requests are used to spawn tasks. Channels, events and requests are modeled using associative UML classes.
- All tasks shall have a behavior modeled within a UML activity diagram. DIPLODOCUS activity diagrams offer operators such as write / read in channel, wait or send an event, send a request. Other operations such as variable manipulation, loops and tests rely on usual UML activity diagram operators.

1 DIPLODOCUS stands for DesIgn sPace exLoration based on fOrmal Description teChniques, Uml and SystemC

One of the strength of the DIPLODOCUS profile relies in its formal semantics. Indeed, all DIPLODOCUS models may be translated into a LOTOS specification [LOT 89] or into a UPPAAL Specification [BEN 04]. For simulation purpose, a translation of DIPLODOCUS models into SystemC code has also been defined.

3. TTool for DIPLODOCUS

To implement DIPLODOCUS, we have reused a toolkit - named TTool - developed for another UML profile called TURTLE [APV 04]. Now, TTool makes it possible to model DIPLODOCUS diagrams, and from those DIPLODOCUS diagrams to perform SystemC based simulations [SYS 07], and LOTOS [CAD] or UPPAAL [BEN 06] based formal analysis (see Fig. 1). In TTool, no knowledge of SystemC, LOTOS, or UPPAAL is necessary to use analysis techniques: a press-button approach has been developed. It totally hides underlying simulation or validation languages to DIPLODOCUS users. Note that *TTool for Diplodocus* is an open-source toolkit that can be downloaded at http://labsoc.comelec.enst.fr/turtle.
TTool works as follows (see Fig. 1). DIPLODOCUS class and activity diagrams are first performed, using a built-in graphical user interface. Then, these diagrams may be derived into:

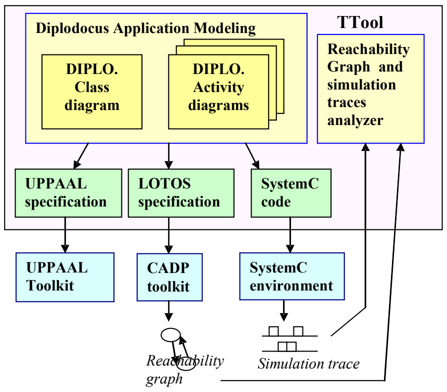

Figure 1: Main principles of TTool

- A UPPAAL specification. UPPAAL offers simulation and model-checking capabilities.
- A LOTOS specification. CADP [CAD] is used, transparently, to generate a reachability graph than can be analyzed directly from TTool (search for deadlock situations, etc.), minimized (with CADP) or visualized.
- A System C code. This code contains parts corresponding to diagrams and other parts for tracing some particular events (read and write in channels, etc.). Simulation traces can be analyzed from TTool and displayed using external tools.

4. Case study: A Telecommunication Subsystem

WIDENS (Wireless Deployable Network System) is a European project for the implementation of ad-hoc mobile communication networks for public safety organizations. As a case study, we started from the specification of different services offered by the mobile terminals. These services are written in C and Matlab. Among these services, we have chosen OFDM (de)modulation at physical layer called the Broadcast Channel Encoder (BCH).

The modeling of BCH with DIPLODOCUS is composed of a class diagram with 12 dclasses connected throughout channels, events and requests. Five dclasses model the functional blocks, and some others are used for initialization of lookup tables and buffers at system startup. *BCH Encoder* is the main dclass which requests other dclasses using *REQUEST* in a sequence. Some dclasses further request others. An excerpt of this diagram is represented at Fig. 2. Also, Fig 3. depicts the activity diagram of *CRC_Table* dclass. This dclass loops 256 times. In the main body of its loop, it first requests a new task (*REQ_CRCbit*) with two parameters (1, and poly_val). Then, its sends one sample in channel named *c*, performs one operation on an integer (rectangle with I), waits for one sample on channel *crc*, and finishes with sending one sample in channel *CRCTable_in*. Once this loop is over, its sends an event named *CRC_TABLE_DONE*.

From this specification, we were able to generate SystemC code, and simulate it to obtain traces. We have also generated a LOTOS specification, and obtained a reachability graph with around 350,000 states and more than 1.5 millions of transitions. A few properties have been proved, for example, the "right" sequence of actions between two functional modules. Also, FIFOs related to some communication channels were proved not to contain more that a fixed number of samples.

4. Conclusion

The implementation of the DIPLODOCUS Profile under TTool offers a user-friendly environment for the design space exploration of complex systems. DIPLODOCUS differs from other environments in its operators' abstraction capabilities and in its formal semantics.

Future work will obviously focus on next methodological steps that are the modelling of hardware components and the mapping of tasks over those hardware components.

5. References

[APV 04] L. Apvrille, J.-P. Courtiat, C. Lohr, and P. de Saqui-Sannes, "TURTLE: A Real-Time UML Profile Supported by a Formal Validation Toolkit", IEEE Transactions on Software Engineering, Vol. 30, No. 7, pp. 473-487, July 2004.

[APV 06] L. Apvrille, W. Muhammad, R. Ameur-Boulifa, S. Coudert and R. Pacalet, "A UML-based Environment for System Design Space Exploration", 13th IEEE International Conference on Electronics, Circuits and Systems (ICECS'2006), Nice, France, Dec. 2006

[BEN 04] J. Bengtsson and W. Yi. "Timed Automata: Semantics, Algorithms and Tools". In Lecture Notes on Concurrency and Petri Nets. W. Reisig and G. Rozenberg (eds.), LNCS 3098, Springer-Verlag, 2004.

[CADP] CADP: http://www.inrialpes.fr/vasy/cadp/

[LOT 89] ISO/IEC 8807, Information processing systems - Open Systems Interconnection - LOTOS – "A formal description technique based on the temporal ordering of observational behaviour", 1989

[RTL] RTL toolkit: http://www.laas.fr/ RT-LOTOS

[SYS 07] SystemC, http://www.systemc.org/

[TTool] TTool: http://labsoc.comelec.enst.fr/turtle/

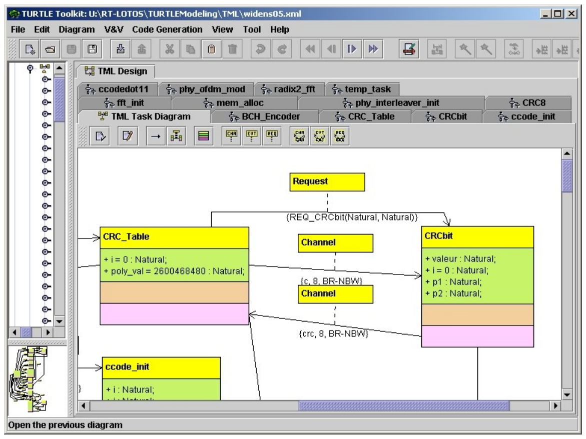

Figure 2: DIPLODOCUS Class Diagram

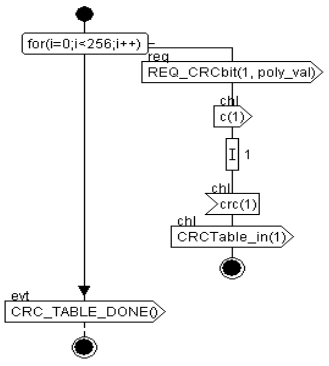

Figure 3: DIPLODOCUS Activity Diagram