Modeling Heterogeneous Embedded Systems with TTool

Daniela Genius, Marie-Minerve Louërat, François Pêcheux, Ludovic Apvrille, Haralampos-G. Stratigopoulos

Modeling Heterogeneous Embedded Systems with TTool

Daniela Genius*, Marie-Minerve Louërat*, François Pêcheux *
Ludovic Apvrille †, Haralampos Stratigopoulos *
* Sorbonne Universités, UPMC Paris 06, LIP6, CNRS UMR 7606,
Email: first_name.last_name@lip6.fr
† Télécom ParisTech, Université Paris-Saclay, Institut Mines-Telecom
Email: first_name.last_name@telecom-paristech.fr

***Abstract*—Embedded systems are increasingly heterogeneous, comprising digital and analog integrated circuits, sensors, and actuators. This paper presents a first step towards an integrated modeling and simulation tool for verification and virtual prototyping of heterogeneous embedded systems on different abstraction levels.**

I. Introduction

The complexity of recent embedded systems pushes current design techniques to their limits. In particular, the space to be explored is getting larger.

Model-oriented design of complex embedded systems is nowadays a current practice in software development for embedded systems, the hardware aspects of such systems are however less frequently designed using this kind of approach.

Many applications like e.g. robotics, automotive and autonomous systems require moreover heterogeneous modeling - including modeling of analog/mixed signal (AMS) and radio frequency (RF) features.

Nevertheless, the related work in the next section demonstrates the lack of tools offering at the same time heterogeneous system modeling, precise simulation, and formal verification.

II. Related Work

Well established tools like *Ptolemy II* [19][20], based upon a data-flow model, address heterogeneous systems by defining several sub domains and synchronization between these domains.

Metropolis [8] is based on a high level model and facilitates the separation of computation from communication concerns. Heterogeneous systems are taken into consideration, but no model for addressing the analog part is provided.

Other tools are based on the Amalthea environment [18] for exploring the design space of automotive applications on multi-core systems, but without relying on formal methods.

In the scope of [2], a mixed analog-digital systems proof-of-concept simulator has been developed [12], based on the SystemC AMS extension standard [17], [4]. Another simulator is proposed in [3]. Integration with software code for general-purpose CPUs and with an operating system is not yet addressed in these approaches.

UML/SysML based modeling techniques [24], [13] are popular with industry targeting embedded systems, but are still rarely used in the domain of heterogeneous system design. Furthermore, with some exceptions [22], [15], they do not lower the level of abstraction to cycle bit accurate level.

III. SystemC AMS Extensions

"SystemC AMS extensions" is a standard describing an extension of SystemC with AMS and RF features [23][17]. The usual approach for modeling the digital part of a heterogeneous system with SystemC [1] is to rely on the *Discrete Event* (DE) part of SystemC AMS extensions. The *Timed data Flow* (TDF) part adds support for signals where data values are sampled with a constant time step. The

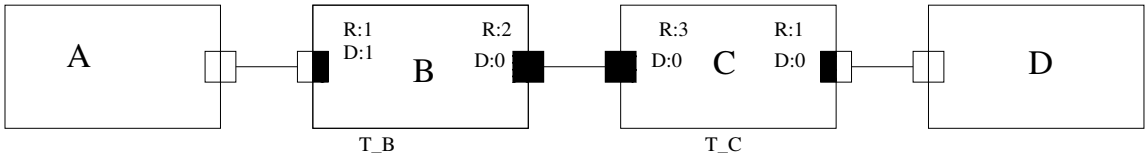

Fig. 1. Two TDF Modules interacting with two DE modules

Electrical Linear Networks (ELN) model of computation of System-C AMS relies on a continuous time domain.

A TDF module is described with an attribute representing the time step and a processing function. The time step is associated to a time period during which the processing function should be executed. The processing function corresponds to a mathematical function which depends on the module inputs and/or internal states. At each time step, a TDF module first reads a fixed number of samples from each of its input ports, then executes the processing function, and finally writes a fixed number of samples to each of its output ports.

A TDF port is described with three attributes:

- Tp represents the *time period*.
- R is the *rate* of data i.e. the number of read or written samples by a TDF port during each period.
- D models the *delay*, the number of samples of the TDF port when a simulation starts.

TDF modules can interact with the DE world using converter ports. In Figure 1, A and D are DE modules, B and C are TDF modules: thus, there are converter ports between A and B, and between C and D.

However, it is pointed out in [5] that it is hard to build a modeling environment supporting together at least DE and TDF, probably because of the difficulty to efficiently support the synchronization between the different models of computation. Indeed, the TDF model of computation is based on the Synchronous Data Flow (SDF) formalism that considers models as a network of synchronous data flow blocks, and does not easily match the one of DE systems. Recent work [6] [10] models the interaction between TDF and DE by colored timed Petri Nets.

IV. High Level Modeling with TTool

TTool [7] is a SysML based, free and open-source software for model-based engineering of embedded systems at different abstraction levels: **functional**, **partitioning**, **software design**, **deployment**. The method associated to these levels [15] details how to take hardware/software partitioning decisions at a high level of abstraction, and to regularly validate these decisions during software development (upper part of Figure 2).

Models of platforms are described in the hardware/software partitioning and deployment stages. Tasks destined to be implemented in hardware are represented as *hardware accelerators*.

Models are thus composed of hardware and software parts. Software tasks for the partitioning model are captured within the functional abstraction level, and software tasks used in deployments are captured in the software design abstraction level. In both partitioning and deployment, the computation part of tasks is then deployed to processors, and the communication and storage part is deployed to buses and memories.

An important advantage of TTool is that it offers an automated approach for formal verification and fast simulation on the three first levels of abstraction. Formal verification is based on internal model-checkers, or on external tools like UPPAAL [11]. On the lowest level (i.e. the deployment level), TTool now offers the generation of a virtual prototype that can be simulated with a cycle bit accurate simulation for Multi Processors System on Chip (MPSoC) [14]. Processor models stem from the *SoCLib* [21] public domain library written in SystemC. SoCLib targets shared-memory *multiprocessor-on-chip system* architectures based on the *Virtual Component Interconnect* standard which separates the components' functionality from

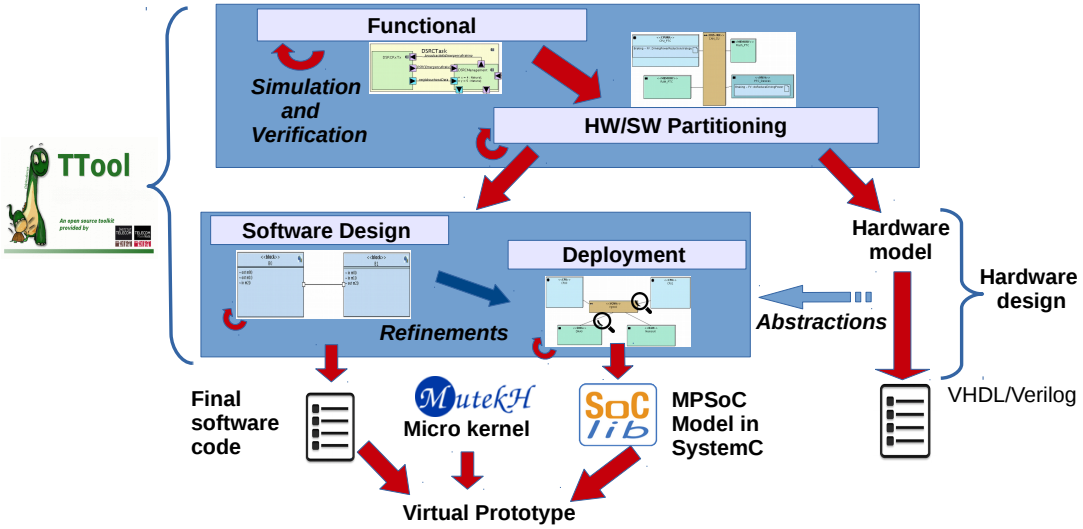

Fig. 2. Hardware/Software partitioning and Code generation for MPSoC platforms

their communication. The lower part of Figure 2 shows generated software code, cross-compiled for one of several general purpose processors and running under the MutekH [9] micro kernel on the SoCLib virtual prototype of the MPSoC.

Several case studies have been performed in order to explain how the abstraction levels of TTool relate, e.g. an automotive obstacle detection [14] and a rover [16]. While sensors, GPS, radar, etc. can be approximately modeled with highly abstracted digital blocks, the accuracy of our models and verifications would benefit from more realistic models taking into account the AMS part.

V. Integration of Analog Components

A. Contribution

Our initial idea was to add analog components to the partitioning level, treating them at the same, very abstract, level as the (digital) hardware accelerators. After the partitioning stage, just like for such digital components, analog components could be developed independently from the software tasks. Yet, the development of analog components must be compatible with the partitioning decisions that can be revised in the next abstraction levels (software design, deployment), and thus, abstractions of these analog components could be useful, typically during the deployment phase.

We have already augmented the graphical interface of TTool with the possibility to describe SystemC AMS blocks with their DE, TDF and converter ports. Figure 3 shows the panel we added to TTool to this purpose. Rates and delays of TDF ports as well as the time period and DE/TDF and TDF/DE converter ports can be configured and parameterized (left sub-window of Figure 3). However, the behavior of the SystemC AMS blocks must be provided directly in SystemC AMS since the abstraction of the behavior of these components is not likely to be easily modeled in a UML/SysML way: we thus currently rely on external descriptions. The SystemC AMS code can be entered in an editor (right sub-window of Figure 3).

We are currently implementing the generation of SystemC AMS code of the components as well as the top cells from these mixed graphical/textual descriptions. This generation will have three phases:

1) In a first phase, we aim at generating fully functional SystemC AMS top cells and analog components by limiting to platforms without any software part and without preexisting SoCLib components. To do this, we have selected

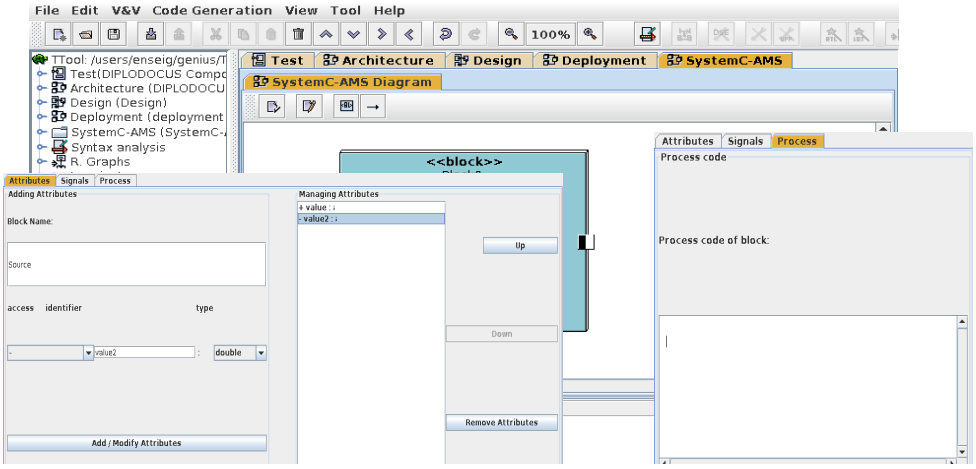

Fig. 3. Parameterizing of ports and SystemC AMS editor in TTool

simple existing SystemC AMS platforms from [3]: source-rectifier-sink, vibration sensor, etc. which we try to reproduce by our generator.

2) In a second phase, we intend to combine the SystemC AMS part with a digital MPSoC platform. The automated generation and configuration of the digital part of the hardware has already been addressed in previous contributions [14]: cache associativity, memory size, type of interconnect. On the software side, we limit to some assembler instructions directly loaded into memory. The (huge) remaining work is to combine this digital infrastructure with the analog part; we plan to start from the contribution described in [5].
3) In a third phase, we intend to run larger scope software, requiring an operating system and a linker script.

B. Case Study

A rover system meant to assist rescuers to find victims in debris is used as a case study. The rover features several sensors including ultrasonic and temperature sensors. Depending on the distance measured by the ultrasonic distance sensor, the sampling rate of the temperature senor is adapted and the motor receives commands to speed up or slow down.

Figure 4 presents the functional model that abstracts sampling rates of sensors with numerical (integer) values (type *Natural*). This model contains the two sensor blocks and the motor control, managed by a central control block communicating by channels (blue arrows) and through events (pink arrows). The current model is unable to express the analog nature of the temperature and distance signals. Moreover, the concrete values of the sampling rates cannot be given at this level. In Figure 5, *MotorContol* and *MainControl* are mapped to the general purpose CPU. The temperature and distance sensors are currently modeled as *hardware accelerator* blocks on the partitioning level. These will later be replaced by SystemC-AMS components generated from the mixed SysML/SytemC AMS description of TTool.

Figure 6 shows the deployment view, from which the SoCLib platform will be generated by extending the method described in [14].The software tasks, *MotorControl* and *MainControl*, are mapped to a general purpose processor.

Figure 7 finally shows the modules of the rover as SystemC AMS modules; the two analog modules are connected by converter ports, and the sampling rates are taken from the software design model. The corresponding code will be generated by our TTool extension.

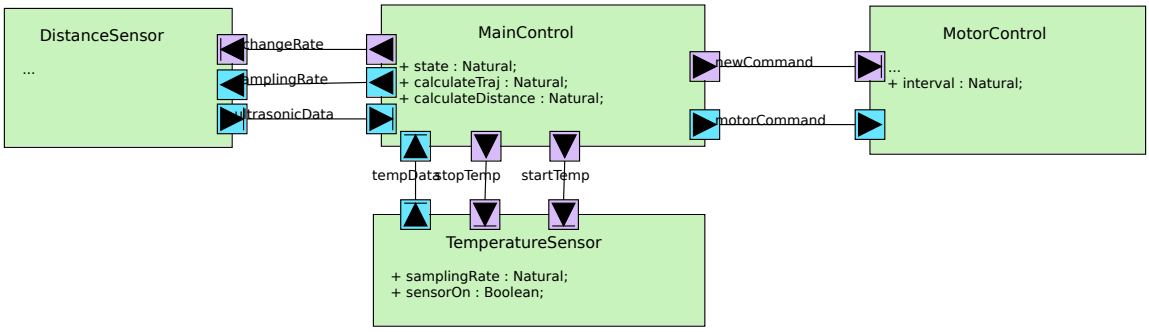

Fig. 4. Functional model of the rover

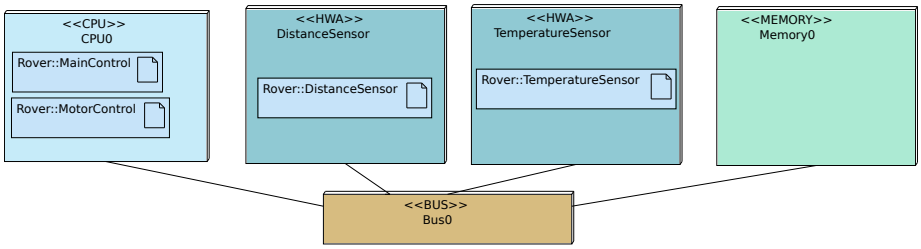

Fig. 5. Hardware/Software Partitioning of the rover

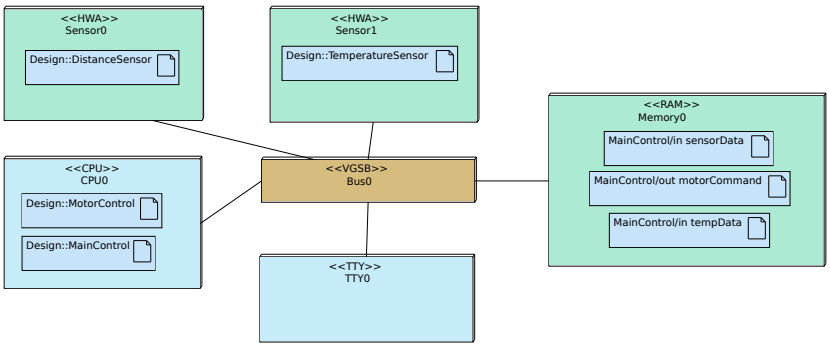

Fig. 6. Deployment view of the rover

VI. Conclusion and Perspectives

We outline a method to integrate analog components into a multi-level modeling tool for complex embedded systems. We are currently working on the automatic generation of heterogeneous platforms.

Once this step achieved, we will be able to simulate more complex heterogeneous systems with extended software parts running on (digital) general-purpose processors and (light) operating systems.

In the spirit of SoCLib, we have also started to define a library of basic analog components (filters, amplifiers, A/D converters), configurable to some degree, which should progressively be enriched. This will be another challenging task: each analog component has specific features like e.g. equations for which it is much harder to determine common aspects than for their digital counterparts.

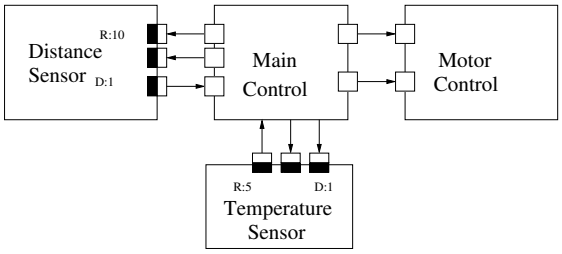

Fig. 7. SystemC AMS modules of the rover

REFERENCES

[1] SystemC. In *http://www.systemc.org*.

[2] Beyond Dreams (Design Refinement of Embedded Analogue and Mixed-Signal Systems). 2008-2011.

[3] Heterogeneous Inception. In *https://www-soc.lip6.fr/trac/hinception*, 2012-2015.

[4] Accellera systems initiative. *SystemC AMS extensions Users Guide, Version 1.0*.

[5] L. Andrade, T. Maehne, A. Vachoux, C. Ben Aoun, F. Pêcheux, and M.-M. Louërat. Pre-Simulation Formal Analysis of Synchronization Issues between Discrete Event and Timed Data Flow Models of Computation. In *Design, Automation and Test in Europe, DATE Conference*, Mar. 2015.

[6] L. Andrade Porras. *Principles and implementation of a generic synchronization interface between SystemC AMS models of computation for the virtual prototyping of multi-disciplinary systems*. PhD thesis, Université Pierre et Marie Curie, 2016.

[7] L. Apvrille. TTool, an open-source toolkit for the modeling and verification of embedded systems. In *http://ttool.telecom-paristech.fr/*.

[8] F. Balarin, Y. Watanabe, H. Hsieh, L. Lavagno, C. Passerone, and A. L. Sangiovanni-Vincentelli. Metropolis: An integrated electronic system design environment. *IEEE Computer*, 36(4):45–52, 2003.

[9] A. Becoulet. Mutekh. http://www.mutekh.org.

[10] C. Ben Aoun. *Principles and Realization of a Virtual Prototyping Environment for Composable Heterogeneous Systems*. PhD thesis, Université Pierre et Marie Curie, 2017.

[11] J. Bengtsson and W. Yi. Timed automata: Semantics, algorithms and tools. In *Lecture Notes on Concurrency and Petri Nets*, pages 87–124. W. Reisig and G. Rozenberg (eds.), LNCS 3098, Springer-Verlag, 2004.

[12] K. Einwich. *SystemC AMS PoC2.1 Library, COSEDA, Dresden*, 2016. http://www.coseda-tech.com/systemc-ams-proof-of-concept/.

[13] A. Gamatié, S. L. Beux, É. Piel, R. B. Atitallah, A. Etien, P. Marquet, and J.-L. Dekeyser. A model-driven design framework for massively parallel embedded systems. *ACM Trans. Embedded Comput. Syst*, 10(4):39, 2011.

[14] D. Genius and L. Apvrille. Virtual yet precise prototyping: An automotive case study. In *ERTSS'2016*, Toulouse, Jan. 2016.

[15] D. Genius, L. W. Li, and L. Apvrille. Model-Driven Performance Evaluation and Formal Verification for Multi-level Embedded System Design. In *Conferénce on Model-Driven Engineering and Software Development (Modelsward'2017)*, Porto, Portugal, Feb. 2017.

[16] D. Genius, L. W. Li, and L. Apvrille. Multi-level Latency Evaluation with an MDE Approach. In *Conferénce on Model-Driven Engineering and Software Development Modelsward'2018 (to appear)*, Funchal, Portugal, Jan. 2018.

[17] IEEE. *IEEE Std 1666.1 standard*. https://standards.ieee.org/findstds/standard/1666.1-2016.html, January 2016.

[18] K. Latif, M. Selva, C. Effiong, R. Ursu, A. Gamatie, G. Sassatelli, L. Zordan, L. Ost, P. Dziurzanski, and L. S. Indrusiak. Design space exploration for complex automotive applications: An engine control system case study. In *Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools*, RAPIDO '16, pages 2:1–2:7, NY, USA, 2016. ACM.

[19] E. A. Lee. Disciplined heterogeneous modeling. In D. Petriu, N. Rouquette, and O. Haugen, editors, *Proceedings of the ACM/IEEE 13th International Conference on Model Driven Engineering, Languages, and Systems (MODELS)*, pages 273–287. LNCS 6395, Springer-Verlag, Oct. 2010.

[20] C. Ptolemaeus, editor. *System Design, Modeling, and Simulation using Ptolemy II*. Ptolemy.org, 2014. http://ptolemy.org/books/Systems.

[21] SocLib consortium. The SoCLib project: An integrated system-on-chip modelling and simulation platform. Technical report, CNRS, 2003. www.soclib.fr.

[22] S. Taha, A. Radermacher, and S. Gérard. An entirely model-based framework for hardware design and simulation. In *DIPES/BICC*, volume 329 of *IFIP Advances in Information and Communication Technology*, pages 31–42. Springer, 2010.

[23] A. Vachoux, C. Grimm, and K. Einwich. Analog and mixed signal modelling with systemC-AMS. In *ISCAS (3)*, pages 914–917. IEEE, 2003.

[24] J. Vidal, F. de Lamotte, G. Gogniat, P. Soulard, and J.-P. Diguet. A co-design approach for embedded system modeling and code generation with UML and MARTE. In *DATE*, pages 226–231. IEEE, 2009.