SystemC AMS Based Frameworks for Virtual Prototyping of Heterogeneous Systems

François Pêcheux
CNRS-Sorbonne University – UPMC-LIP6, France
francois.pecheux@lip6.fr

Christoph Grimm
Universität Kaiserslautern, Germany
grimm@cs.uni-kl.de

Torsten Maehne
Berner Fachhochschule (BFH), Switzerland
torsten.maehne@bfh.ch

Martin Barnasconi
NXP Semiconductors, The Netherlands
martin.barnasconi@nxp.com

Karsten Einwich
COSEDA Technologies GmbH, Germany
karsten.einwich@coseda-tech.com

***Abstract*—This paper presents the past, present, and perspectives of the SystemC AMS standard as well as several commercial and academic frameworks gravitating around it that have been used to develop use cases in various cyber physical domains. After an overview of the standard, two frameworks are described: the COSIDE environment that supports the virtual prototyping of embedded HW/SW systems and its interaction with AMS circuits. Second, the SICYPHOS framework that integrates SystemC AMS into the overall system development ecosystem. By an example we give details how SystemC AMS can be coupled with other simulation tools (OpenModelica) while keeping simulation speed and accuracy high. Another example shows how SystemC AMS can be used as foundation technology to create specialized user-defined models of computation (MoC).**

Keywords—SystemC; SystemC AMS; Heterogeneous Modeling; Verification

I. INTRODUCTION

The design of heterogeneous systems always involves different disciplines and physical domains, controlled at some point by embedded software. In traditional design flows, the various components of the system are usually designed by different domain experts with specific idiosyncrasies, and limited knowledge of their immediate environment. Problems arise from the fact that these components of various origins tightly interact in practice on the chip and any fault in the independently developed system parts can jeopardize the global system integrity at any time. Furthermore, heterogeneous systems tightly interact with their immediate, and often external, environment. As a result, the design process must take into account environmental conditions that are not always well known. Different components in these systems imply the management of tolerances, uncertainties, and deviations from the expected responses.

To address these issues, system-level simulation and model-based design methodologies are widely used. However, using those component models independently is not sufficient to evaluate the holistic behavior of the system and combining models from multiple domains is a challenging task. This is due to the different abstraction levels that have to be used to address the needs of the concerned engineering domain, the interaction of behaviors with very different time constants, and the need to execute the resulting models through different Models of Computation (MoCs). In this context, SystemC and its AMS extensions can be used with noticeable profit.

The literature on SystemC AMS is now abundant, and shows that various physical domains have been addressed, ranging from mechanics to biology, fluidics [27] and bond graphs [26]. With a substantial amount of contributions related to automotive and RF systems, this standard appeared over the years as an attractive virtual prototyping solution for many industry and academia designers.

The rest of this paper is organized as follows: Section 2 presents the origins of the SystemC AMS standard. Section 3 describes the virtual prototyping capabilities offered by the COSIDE environment, and Section 4 details the SICYPHOS environment with the help of a working automotive use case. Section 5 proposes another use case, which puts emphasis in the pragmatic encapsulation of ngspice into a holistic SystemC-AMS-based virtual prototype for RFID systems. Section 6 concludes the paper.

II. HISTORY OF SYSTEMC AMS AND MAIN ACHIEVEMENTS

The origins of today's SystemC AMS extensions can be traced back till 2001 [1,2]. This early SystemC-AMS prototype extended the then current SystemC 1.1 simulator with Synchronous Data Flow and Linear Electric Network MoCs. A SystemC AMS study group was formed soon-after [3] to further collect the requirements for SystemC AMS and implementation experience from other SystemC extensions to refine the semantics of the new MoCs and test them in the prototype [4]. The development of the SystemC AMS standard continued in the AMS Working Group (AMSWG) of the Open SystemC Initiative (OSCI) in 2006. The working groups mission is to define and develop the language, associated methodologies and class libraries for abstract modeling approaches for analog, mixed-signal, and RF systems in SystemC. 2010 marked the release of the SystemC AMS 1.0 standard [4], which first enabled to describe analog/mixed-signal behavior using three MoCs (Timed Data Flow (TDF), Linear Signal Flow (LSF), and Electrical Linear Networks (ELN)) as a natural extension to existing SystemC-based design methodologies. In March 2013, the SystemC AMS 2.0

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

standard update [10] was released including features for dynamic and reactive modeling at a high level of abstraction.

A further revision of the SystemC AMS standard was released on April 6, 2016 as IEEE Std 1666.1-2016. In combination with the well-established SystemC language (IEEE Std 1666-2011), a standards-based system-level modeling approach is now available to support functional modeling, model refinement, architecture exploration, and virtual prototyping of AMS systems.

One of the main achievements of the SystemC AMS language is probably the dynamic capabilities of the Timed Data Flow MoC. The initial TDF MoC of SystemC AMS was based on the traditional Synchronous Data Flow (SDF) MoC, which suffered from a severe restriction by imposing a fixed time step. In TDF, continuous signals are evaluated (sampled) in constant discrete time steps. Setting a small time step permits higher accuracy, but has a negative effect on the global simulation performance. Therefore, it is crucial to set the time step to the most suitable value, in order to achieve a good accuracy-performance trade-off. Unfortunately, in heterogeneous systems, the different parts of the system commonly expose extremely different time constants, which might differ by several orders of magnitude. Moreover, the same part of the system might change its behavior during operation, e.g., if it implements active and sleep modes or if it is dynamically reconfigurable like in the case of software-defined radios. It is therefore crucial to be able to modify the time step and rates, at which samples are consumed/produced at the ports of TDF modules, during simulation. It is helpful to be able to dynamically suspend the execution of a TDF model to wait on events from the SystemC Discrete Event (DE) MoC. The dynamic features introduced in the IEEE Std 1666.1-2016 language address these advanced use cases.

Members of the AMSWG participated in the development of the SystemC implementation of the Universal Verification Methodology (UVM) standard (IEEE Std 1800.2-2017). It consolidates verification best practices offering a unified approach for test and sequence creation, building verification components, test bench configuration, and simulation. SystemC AMS and UVM provide complementary solutions for efficient virtual prototyping of heterogeneous systems.

III. THE SYSTEMC AMS COSIDE FRAMEWORK

The COSEDA Technologies GmbH was founded in 2015 as a spin-off from the Fraunhofer Institute for Integrated Circuits (IIS), Design Automation Division (EAS), where the SystemC Proof of Concept simulator was initially developed. COSIDE [16] is an integrated design environment for heterogeneous systems. It closes the gap between the analog and digital domains as well as between the hardware and software worlds. The motivations for COSIDE were to speed up the modelling process, lowering the entry level for beginners, increasing the model quality, enable re-use and integrate system level design and simulation into the overall design flow. This will be achieved by graphical modelling capabilities, generation of code templates as well as large modelling libraries. So COSIDE offers a holistic system-level design entry by considering the different worlds of development jointly. On this way the overall system consisting of hardware, software, analogue as well as physical components can be co-designed and verified using virtual prototypes at different levels of abstraction. In practice, COSIDE offers an easy to use modeling framework to simulate and verify complete systems based on SystemC and SystemC AMS. It is the result of a long effort to propose to the ESL community an operational and complete virtual prototyping environment. Besides the advantages coming along with the underlying SystemC/SystemC AMS technologies like extremely fast simulation, a lot of import and export capabilities to numerous tools are provided. On this way COSIDE enables a seamless crossover from algorithm, concept, and architectural level design down to the implementation. Due COSIDE fully supports SystemC and the TLM standard any SystemC model can be integrated and thus third party digital platforms can be easily integrated. COSIDE integrates the Qbox, which provides based on the virtualization platform Qemu processor models. The environment comes along with debug capabilities like a scriptable mixed-signal waveform viewer, a SystemC/ SystemC AMS aware debugger and numerous library and utility functions. The integrated libraries provide modules at different abstraction levels and different modelling domains. Thus an RF library supports passband, baseband as well as envelope modelling. A mechanical library permits the modelling of the environment e.g. of complex sensors. Several communication libraries support the modeling of buses (e.g. LIN, CAN) as well as physical interfaces like SATA. Most of the COSIDE models can be used as templates for user defined models.

IV. THE SYSTEMC AMS SICYPHOS FRAMEWORK

SICYPHOS is an acronym for SImulation of CYber-PHysical Systems. The SICYPHOS framework shows how to embed SystemC AMS into a systems modeling ecosystem for modeling of heterogeneous systems. SysML provides the overall model of the system structure and component interfaces between different domains (e.g., software, electrical, AMS, mechanical, etc.). Its key aspects are shown in Figure 1.

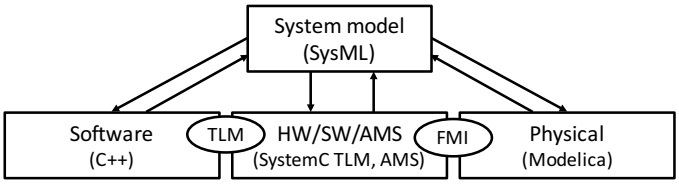

Figure 1: The SICYPHOS framework.

The SysML model is translated into domain-specific languages, in particular SystemC (AMS) or Modelica models. The generated models focus on two aspects:

- Cross-domain consistency of interfaces and the generation of simulator coupling interfaces.
- Cross-domain tracking of uncertainties like tolerances, noise, drift, model inaccuracies, etc. [5] supported by symbolic simulation of SystemC AMS models [6].

For this approach, the OpenModelica or JModelica environments are used for modeling of multiphysical systems,

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

actuators, sensors and their environment. The HW/SW co-design and analog-mixed signal modeling is done with SystemC AMS. Software can be modeled in C++; for interfacing with software models, e.g., TLM can be used or an instruction set simulator. The basis for the required simulator coupling is the TDF MoC of SystemC AMS. For integration with other Modelica simulators as well as MATLAB/Simulink, Dymola, ETAS ASCET, the code generation from SysML is capable to export the interface specification of the models following the Functional Mock-up Interface (FMI) standard. The proposed approach accelerates the systems design process through model-based systems development.

As a proposed co-simulation use case, a throttle valve actuator and associated control system have been modeled with OpenModelica and SystemC AMS. In an Electronic Throttle Controller (ETC), the response of the closed loop depends on the dynamics of the actuator. Therefore, the impact of varying properties, such as the gear ratio of the valve, the throttle blade inertia, supply voltage, and return spring constants, needs to be studied with the help of the developed model to understand the dynamics of the system and estimate the control performance. The FMI standard [9] enables the export of this model for use in other modeling environments.

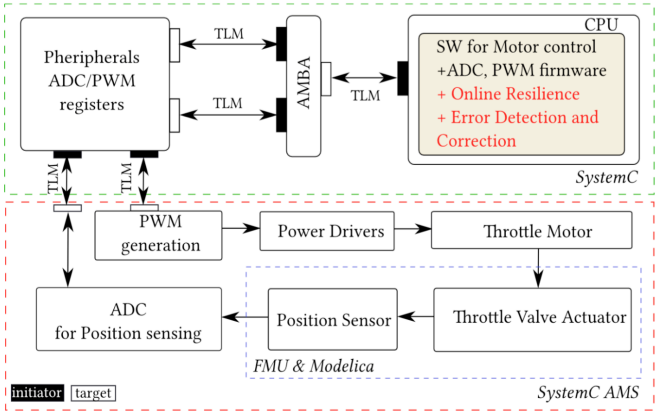

Figure 2: The SICYPHOS use case.

The Figure 2 illustrates the co-simulation set-up, which embeds the Modelica models of the sensor and actuator in the SystemC AMS model of the overall Hw/Sw system. To accelerate the development of the latter and increase simulation speed, Transaction level modelling (TLM) is used to abstract communication between processor and peripherals. The TDF MoC and its dynamic features are used in the co-simulation backplane. To increase the simulation performance and to reduce the computation requirements, we investigated the possibility of dynamic simulation based on triggers from the physical environment. To this end, the environment model in OpenModelica is able to generate dynamic events to the co-simulation framework. For instance, the events like opening or closing of the throttle valve can be randomized and can be used to create dynamic events. This in turn triggers the control algorithms executed on the SystemC processor models. These algorithms implemented in C/C++ can be reused for the application software development. Figure 3 shows an example of system response simulated by the SICYPHOS environment.

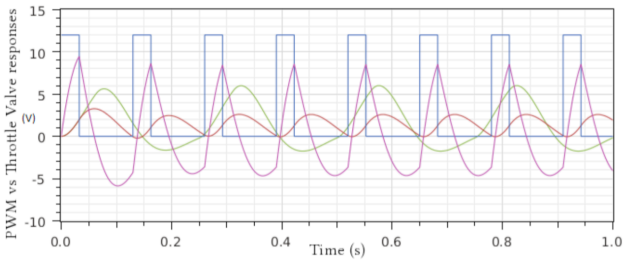

Figure 3: System response for throttle valve parameter variations.

V. SystemC AMS Use case : RFID

Several attempts have been made to introduce new MoCs in SystemC AMS to address new disciplines and modeling techniques, in particular MoCs that can handle non-linear behaviors. Unfortunately, these efforts needed to rely on non-standardized internal APIs of SystemC AMS complicating their implementation. By studying the pioneering works at Fraunhofer on SystemC AMS and at Berkeley on Ptolemy regarding multi-MoC synchronization, the following work developed a methodology to simplify the addition of new MoCs to the SystemC AMS proof-of-concept simulator.

The developed solution is based on the fundamental concept of a MoC hierarchy with master-slave semantics. Starting from the description of several Models of Computation, the relevant features, which constitute the essence of a MoC, have been extracted, i.e., the information required to represent any MoC in an abstract way. In a second step, the concept of master-slave relationship as an interaction mechanism between Models of Computation was formalized. Master-slave semantics are a powerful concept for facilitating the simulation of heterogeneous systems, which individual components may be described using the semantics of conceptually very different MoCs while still seamlessly interacting with each other. This guarantees the flexibility of the target virtual prototyping environment.

By requiring that a master MoC solver does not need to be aware of the existence of potential slave MoC solvers, any slave MoC solver must comply with its master's interface for model elaboration and simulation, i.e., fulfill all its requirements and meet all its expectations. This design decision makes it possible for these master-slave relationships to be represented as an encapsulation process. The approach specifies that all the available modes of interaction between MoCs to exchange information and synchronize their execution are statically defined and that they are provided by a slave MoC solver. This enables the automatic abstraction of a sub-model following the semantics of the slave MoC to a slave MoC solver object inside the master model. This hierarchical heterogeneity approach allows the user to focus only on the important and relevant design and modeling issues, not simulator related ones.

As an example, the passive RFID reading system in Figure 4 is modeled in SystemC AMS by means of three different MoCs, organized hierarchically: Discrete Event (DE) drives

978-1-5386-4881-0/18/$31.00 ©2018 IEEE

Timed Data Flow (TDF) that in turn drives the user-defined Electrical Networks (EN) MoC. The interesting point is that the master-slave semantics with a clearly defined API interface allowed to define a new EN MoC as an add-on to SystemC AMS that gives the end-user the possibility of the modeling of any linear or non-linear component as it in fact encapsulates the well-known ngspice simulator.

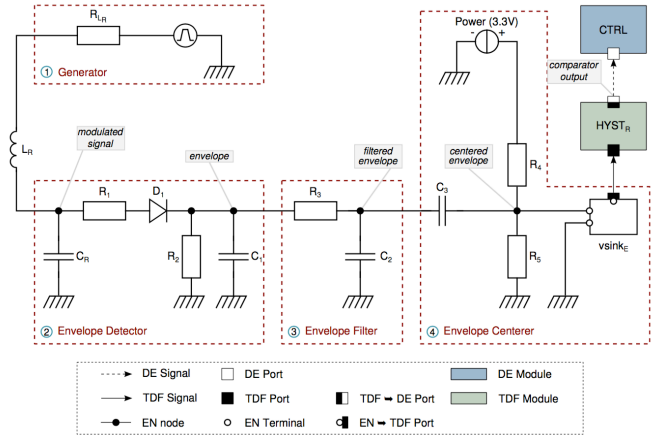

Figure 4: The RFID transceiver modeled with 3 MoCs.

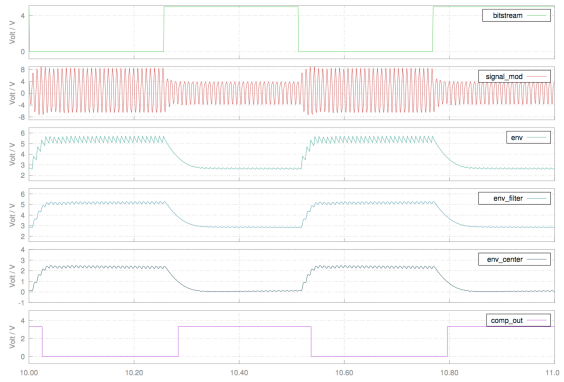

Figure 5: RFID virtual prototyping tag bitstream traces.

The RFID transceiver is composed of a transmission chain and a reception chain connected to the primary coil. The most interesting part, the transceiver reception circuit corresponds to an amplitude demodulator that aims at retrieving the information transmitted by the transponder. The demodulation chain is composed of an envelope detector, envelope filter, and an envelope centerer, all modeled with the EN MoC. The envelope detector aims at straightening the received signal, then removing its negative part as well as the carrier that supported the transmitted information. The virtual prototyping environment based on these principles is operational as stated by Figure 5.

VI. CONCLUSION

This paper presented the developments of the SystemC AMS extensions, resulting in an IEEE supported international language standard. The presented frameworks truly enable co-simulation of models developed with third-party tools in the SystemC AMS environment. Both commercial and academic tools have been described that fulfill the needs for efficient holistic simulation methodologies and formal model-based systems engineering. These methodologies seamlessly integrate both discrete and continuous domains with reasonable accuracy and enable highly promising and realistic system designs. They improve the modeling, co-simulation and verification at a high level of abstraction, and the embedded software developed in the virtual environments can be used directly in the final product.

REFERENCES

[1] C. Grimm, C. Meise, AnalogSL: A Library for Modeling Analog Power Drivers in C++. Proc. 2001 Forum on Design Languages (FDL'01) 2001.

[2] K. Einwich, C. Clauss, G. Noessing, P. Schwarz, H. Zojer: SystemC Extensions for Mixed-Signal System Design, FDL2001

[3] K. Einwich, C. Grimm, A. Vachoux. N.Martinez Madrid, F.R. Moreno, C. Meise, "Analog Mixed Signal Extensions for SystemC", OSCI 2002.

[4] Karsten Einwich, Christoph Grimm, Peter Schwarz, Klaus Waldschmidt: Mixed-Signal Extensions for SystemC. Forum on Design Languages 2002 (FDL 2002). ECSI, Sept.2002

[5] C.Grimm, M. Rathmair. "Dealing with Uncertainties in AMS systems", In: Proceedings of DAC 2017. IEEE 2017.

[6] C. Radojicic, C. Grimm, "Towards Verification of Uncertain Cyber-Physical Systems". In: Proceedings of SNR 2017. pp1-17, DOI: 10.4204/EPTCS.247.1

[7] Andersson,Christian,JohanÅkesson,ClausFührer,andMagnusGäfvert:*Import and Export of Functional Mock-up Units in JModelica.org*. Modelica Associa- tion, 2011, ISBN 978-91-7393-096-3.

[8] Åkesson, Johan, Karl Erik Årzén, Magnus Gäfvert, Tove Bergdahl, and Hubertus Tummescheit: Modeling and Optimization with Optimica and JModelica.org— Languages and Tools for Solving Large-Scale Dynamic Optimization Problems. Computers and Chemical Engineering, 34(11):1737–1749, November 2010.

[9] Bertsch, Christian, Elmar Ahle, and Ulrich Schulmeister: The Functional Mockup Interface - seen from an industrial perspective. In Proceedings of the 10th International Modelica Conference, http://dx.doi.org/10.3384/ecp1409627, March 2014.

[10] Barnasconi, Martin, Karsten Einwich, Christoph Grimm, and Alain Vachoux (editors): *Standard SystemC® AMS extensions 2.0 Language Reference Manua*. OSCI, 2013. http://accellera.org.

[11] Blochwitz,Torsten,MOtter,MArnold,CBausch,CClauß,HElmqvist,AJung - hanns, J Mauss, M Monteiro, T Neidhold, *et al.*: *The functional mockup interface for tool independent exchange of simulation models*. In *Modelica'2011 Conference, March*, pages 20–22, 2011.

[12] Damm, Markus, Christoph Grimm, Jan Haase, Andreas Herrholz, and Wolfgang Nebel: Connecting SystemC-AMS models with OSCI TLM 2.0 models using temporal decoupling. In 2008 Forum on Specification, Verification and Design Lan- guages, pages 25–30. IEEE, September 2008, ISBN 978-1-4244-2264-7. http: //dblp.uni-trier.de/db/conf/fdl/fdl2008.html\#DammGHHN08.

[13] Fritzson, P., P. Aronsson, A. Pop, Hakan Lundvall, Kaj Nystrom, Levon Saldamli, D. Broman, and Anders Sandholm: OpenModelica - A free open-source environ- ment for system modeling, simulation, and teaching. In Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE, pages 1588–1595, Oct 2006.

[14] Functional Mock-up Interface for Model Exchange and Co-Simulation, July 2014.

[15] *Model-Based Integration Platform for FMI Co-Simulation and Heterogeneous Simulations of Cyber-Physical Systems*, volume Proceedings of the 10th International Modelica Conference. Modelica Association and Linkoping University Electronic Press, 2014, ISBN 978-91-7519-380-9.

[16] Coseda Technologies GmbH. Coside. http://www.coseda-tech.com/coside-overview.

978-1-5386-4881-0/18/$31.00 ©2018 IEEE