Viewpoint: Analog/Mixed-Signal (AMS) Extensions for SystemC

Martin Barnasconi, AMS Working Group Chairman, Open SystemC Initiative (OSCI)

More and more voices in the Semiconductor industry are calling for a renewed methodology for analog/mixed-signal (AMS) design, addressing the importance of having a solid system-level design approach that allows making early trade-offs between analog and digital functionality in hardware and software.

In December 2008, the Open SystemC Initiative (OSCI) revealed their first draft standard of the SystemC AMS extensions. The standardization of the AMS language extensions for SystemC is a first step towards bringing AMS into the digitally-oriented ESL world.

Who is driving this AMS initiative?

The OSCI AMS working group was formed in 2006 with the aim of standardizing AMS extensions for SystemC. The formation was strongly driven by the European Semiconductor industry including STMicroelectronics, NXP Semiconductors and Infineon. Full support has been given by research institutes and universities such as Fraunhofer Institute for Integrated Circuits (IIS), Ecole Polytechnique Fédérale de Lausanne (EPFL) and Vienna University of Technology.

Before the formation of the AMS working group, these research institutes and universities were already active in a study group defining the first concepts for extending SystemC with AMS features, as well as working on a prototype implementation known in the industry as "SystemC-AMS." The standardization of the SystemC AMS extensions is based on the knowledge gained from this work and has been enhanced to support an AMS design refinement methodology and to fulfill the latest requirements from the industry.

Yet another AMS standard?

The AMS draft 1 standard focuses on the system-level and architecture modeling aspects of designing and verifying complex AMS systems. By having AMS extensions for SystemC, users can build an executable description of the AMS system in a C++ based manner, enabling seamless integration with HW/SW architectures in SystemC and functional models or software developed in C and C++. As such, the AMS extensions should not be considered as a replacement of existing hardware description languages, but should be seen as a valuable addition to ESL design methodologies.

Design refinement methodology

The proposed AMS draft 1 standard facilitates a design refinement methodology for analog/mixed-signal systems supporting functional modeling for creating an executable specification, virtual prototyping, architecture exploration, integration validation, and other use cases. Similar to digitally-oriented ESL methodologies, different levels of design abstraction are defined, each with its own modeling formalism and associated AMS modeling behavior and accuracy. A unified modeling style allows easy "mix-and-match" of these different levels of abstraction, introducing an open and transparent modeling approach using the SystemC AMS extensions. This is essential to support a top-down design flow for analog/mixed-signal systems, where the design abstraction of the model is well defined and the interaction between models – even at different levels of abstraction – is well supported using the same simulation framework.

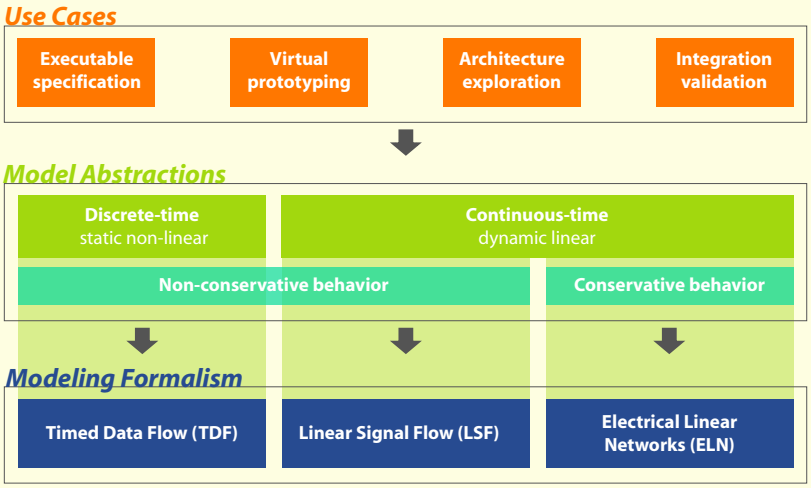
Model abstractions

The model abstractions supported by the AMS extensions are based on well-known methods for abstracting analog and mixed-signal behavior. The abstraction levels distinguish discrete-time from continuous-time behavior and non-conservative from conservative

SystemC AMS Extensions

Analog/mixed-signal system-level modeling and design refinement methodology

Use Cases, Model Abstractions and Modeling Formalism



Contents

- Time domain, small-signal frequency-domain, and small-signal frequency-domain noise analysis
- TDF modules for user-defined primitives
- Predefined LSF modules for signal flow primitives
- Predefined ELN modules for electrical primitives
- Ports/signals for intermodule communication

descriptions. Discrete-time modeling is particularly suited for signal-processing-dominated applications for which signals are naturally (over) sampled. If signals cannot be sampled, the analog behavior should be described as a continuous-time function, such as by describing the system as a set of differential and algebraic equations (DAEs).

Non-conservative system descriptions abstract physical quantities (e.g., voltages and currents) as independent real-valued signals. For a conservative description, the relation between these voltages and currents at the nodes is preserved and should satisfy Kirchhoff's laws.

Another method is to abstract dynamic non-linear behavior by static non-linear or dynamic linear behavior. For discrete-time models, the static non-linear behavior can be defined as an algorithm (e.g., using a polynomial function). For continuous-time models, the equations are computed using a linear DAE solver to keep the equations simple, resulting in efficient calculations.

By introducing these modeling methods, the AMS extensions will enrich SystemC to enable the creation of AMS behavioral models at different levels of abstraction.

Modeling formalisms

The SystemC AMS extensions define the essential modeling formalisms required to support AMS behavioral modeling at different levels of abstraction, consistent with the design refinement methodology. Established modeling formalisms are being standardized, introducing Timed Data Flow (TDF), Linear Signal Flow (LSF) and Electrical Linear Networks (ELN) modeling styles, which can be used in combination with SystemC descriptions.

Execution semantics based on TDF introduce discrete-time simulation without the overhead of the dynamic scheduling imposed by the discrete-event kernel of SystemC. Simulation is accelerated by defining a static schedule that is computed before simulation starts. To model continuous-time behavior, LSF or ELN descriptions can be used, for which simulations only require a simple linear DAE solver. Interactions with discrete-time TDF models then consider discrete-time data samples as continuous in time through interpolation techniques. With these methods, the continuous-time and discrete-event computations become "loosely coupled," reducing the simulation overhead.

The integration of this dataflow procedural processing and a linear DAE solver, combined with the existing event-based engine in SystemC, makes this approach a very flexible and efficient simulation solution that covers both mixed-signal and mixed-level aspects, facilitating analog/digital co-design for architecture studies and software development for embedded AMS systems.

Which applications can benefit from using the SystemC AMS extensions?

Not only has the trend towards having a System-on-a-Chip resulted in sophisticated digital computational engines on these chips, but the need to communicate and interface to the outside (analog) world has led to more and more mixed-signal content on these chips.

These embedded analog/mixed-signal systems are characterized by 1) having digital HW/SW interwoven with AMS functionality due to the need for calibration and control algorithms, and 2) mixed-signal processing including a dynamic model of the physical layer (PHY) in the modeling of the complete communication protocol (software) stack.

Martin Barnasconi, AMS Working Group Chairman, OSCI, and Product Manager AMS/RF System Design Methods, NXP Semiconductors, The Netherlands

Examples of such embedded analog/mixed-signal applications include telecommunication systems for wireless connectivity (e.g., WLAN, WiMAX) and cellular infrastructure (e.g., W-CDMA, HSDPA). Also, wired interconnect (e.g., HDMI, LVDS drivers) and wired telecommunication systems (e.g., ADSL, VDSL) have analog functionalities deeply intertwined with digital functions. In the automotive domain, examples include in-vehicle networking (e.g., CAN, FlexRay), wireless sensor networks (e.g., to monitor tire pressure) and true heterogeneous systems (e.g., to model electronics as part of a gearbox or kludge). An additional application domain consists of imaging sensors to model charge-coupled devices (CCD).

As the AMS extensions are developed as an additional layer on top of SystemC, the calibration and control algorithms, software protocol and analog parts of the physical layer can be ultimately modeled together.

Collaboration and standardization

The need for an AMS design refinement methodology starting at the system level has propelled the semiconductor industry, research institutes, universities and EDA/ESL vendors to collaborate in order to define a uniform and standardized modeling language based on SystemC.

Within the OSCI AMS working group, a consolidated view on the requirements for such a standard has been defined and the team successfully released the first draft standard of the SystemC AMS extensions at the end of 2008. The AMS working group will continue to advance and enhance the AMS standard based on input it receives during the public review period. The SystemC AMS extensions are a topic of discussion in upcoming user group meetings, tutorials and workshops at EDA conferences and events.

The SystemC community is encouraged to review the AMS draft 1 standard and provide feedback by joining the AMS discussion forum on www.systemc.org. In addition, members of OSCI can participate in the AMS working group, which will continue to steer the direction of the AMS standard.

Join now! Sign up at www.systemc.org.

SYSTEMC™

Open SystemC Initiative
Defining and Advancing SystemC Standards