Content

♦ Introduction SystemC-AMS

♦ Application Examples
  • Wired Communication Application
  • Automotive Sensor Application
  • Mixed Signal Embedded Core Application
  • Wireless Applications

♦ Conclusion
SystemC-AMS

♦ Extension Library for SystemC

♦ Developed by the OSCI Analog Mixed Signal Working group

♦ Prototype versions public available based on the Fraunhofer implementation

♦ Public version supports modeling of:
  • Non-conservative systems
  • Multi rate synchronous dataflow (SDF)
  • Linear electrical networks
  • Linear behavioral functions (linear transfer function numerator/denumerator and pole zero, state space),
  • Frequency domain simulation
  • Powerful trace functionality

♦ Experimental extensions available at FhG: Switched Capacitor solver, Nonlinear DAE solver with de-synchronization
SystemC-AMS is an extension of SystemC

• no changes to the SystemC implementation

→ use of the same SystemC implementation

→ no restrictions for the use of the SystemC language

• as far as possible, only LRM documented features used for the library implementation
# SystemC / SystemC-AMS language architecture

---

<table>
<thead>
<tr>
<th>Application</th>
<th>Methodology- and technology-specific libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Written by the end user</td>
<td>SystemC-AMS testbench utilities, electrical / mechanical macro models, …</td>
</tr>
</tbody>
</table>

## Core language
- **Analogue core language**
  - analogue modules, ports, signals, nodes, solver and synchronisation basics

## Predefined channels
- **Predefined analogue MoC**
  - (non-)conservative modules, linear DAE solver, constant time step synchronisation, …

## Utilities
- **Analogue utilities**
  - complex numbers, matrices, …

## Data types
- **Predefined domains**
  - electrical, mechanical, magnetic …

---

**Programming language C++**
Concept of SystemC-AMS

- **View layer**
- **Solver layer**
- **Synchronisation layer**
- **SystemC layer**

Solver 1
Solver 2

AMS Synchronisation

SystemC kernel

classical SystemC Layers
Why having different analogue Models of Computation?

♦ Modelling on different abstraction / accuracy levels yields the possibility to apply specialised algorithms, which are orders of magnitude faster than a general approach.

♦ It is possible to reduce the solvability problem significantly.

♦ Due to the encapsulation of analogue MoC / solvers SystemC-AMS models are very well scalable – very large models can be handled.

♦ Examples for specialised analogue Models of Computations (MoC):
  • Linear Networks / Differential-Algebraic Equation (DAE) systems
  • Non-linear Networks / DAE systems
  • Switched Capacitor Networks (leads to simple algebraic equation)
  • Dataflow solver for Signalflow Descriptions and Bond Graphs
  • …
Application of SystemC-AMS to a Voice Codec System

- Linear electrical networks (conservative)
- Discrete event (SystemC modules)
- C – Code for target processor

- Signalflow (non-conservative), frequency domain
- Embedded linear analogue equations
- Multi-rate static dataflow, frequency domain
SystemC/SystemC-AMS specific Advantages

♦ Can be **tailored** and optimized for specific applications

♦ Support of **customized methodologies** and their combination

♦ The **tradeoff** between accuracy, simulation performance and modeling effort can be **optimized** for each system part by using the interoperability of an arbitrary number of Models of Computations (MoC)

♦ **Encapsulation** of subsystems which leads to scalability and modularity

♦ Easy **software integration** and powerful debug possibilities

♦ Full **power of C++** available (e.g. language, libraries, encapsulation concepts)

♦ Easy **IP protection** by pre-compilation and integration into other tools and design flows via C interfaces
Application areas of SystemC-AMS

Description, Simulation and Verification for:

♦ Functional **complex** integrated systems

♦ **Analogue** **Mixed-Signal** systems / **Heterogeneous** systems

♦ **Specification / Concept and System Engineering**

♦ **System design**, development of a (“golden”) **reference model**

♦ Embedded **Software** development

♦ **Next Layer (Driver) Software development**

♦ **Customer model, IP protection**
Application Example
POTS - System

- Complete System functionality modeled
- All relevant analogue effects
- Digital parts “bittrue”, original code of embedded software
- Hundreds of simulation scenarios as regression tests available
- Simulation scenarios partially re-used for silicon verification
- Embedded software debugged before silicon
Simulation Time for Vinetic 2CPE System

♦ SystemC-AMS Simulation
  • 2 channel including: SLIC, externals, AFE, DFE, ASDSP and part of Carmel FW
  • 1 sec realtime → 1,5h simulation time

♦ VHDL RTL
  • 2 channel including: AFE, DFE, ASDSP, Carmel and Interfaces
  • 1 sec realtime → 300h simulation time

♦ Nano Sim (Fast CMOS simulator)
  • 2 channel including: AFE top level
  • 1 ms realtime → 15h simulation time

♦ Titan Simulation
  • 2 channel including: AFE top level
  • 1 ms realtime → 500h simulation time

♦ SystemC-AMS Simulation
  • only one channel
  • reduce sampling rate for analog blocks (used for FW simulation only)
  • 1 sec realtime → 90 sec simulation time

Source: Gerhard Nössing Infineon COM
ADSL / VDSL Systems

- Transient settling behavior
- Interaction Voice / Data transmission
- Training algorithm
- BER estimations
- Number of use scenarios
- Interaction of different lines
- Multi level simulation environment essential
Automotive Sensor Applications

TIER2

TIER1

OEM

pressure pulse
Simulation Performance Challenge for Automotive Applications

Temperature compensated TPMS sensor (MEMS)
- application (driving) 2 hrs. (1/t ~ 250µHz)
- battery voltage update rate ~ 20mHz
- temperature update rate ~ 200mHz
- pressure update rate (wakeup) ~ 2Hz
- analog processing rate ~ 1MHz
- digital processing rate ~ 4Mhz

Source: Wolfgang Scherr Infineon AIM
Automotive Sensor Projects

♦ Systemlevel model including the embedded processor on a cycle accurate level

♦ Switched capacitor converter

♦ Diagnose modes, offset calibration, temperature dependencies, noise, manchester interface, synchronization via supply voltage, ...

♦ Original code of embedded software

♦ TLM based modeling for processor communication

♦ IP protected customer model as Matlab/Simulink Module (mex – dll)

♦ Simulation performance ~ 10min /sec
Mixed-Signal Embedded Core Project

- Serial ATA physical layer chip set for 3 / 6 GBit serial data transmission e.g. to/from hard discs
- Concept engineering model of transceiver /receiver including the pll's
- Goal estimation of bit error rates, simulation of pll locking behavior
- Pin compatible VHDL-model for digital design via SystemC Modelsim integration as reference and stimuli generator
SATA Project Results

♦ Simulation performance ~ 2h/ms = 6e6 clock cycle

♦ PLL settling / locking

♦ Equalizer coefficient adaption

♦ Estimation of BER
Wireless Transmission System

- System level design and design space exploration
- Performance estimation (especially BER) for different Architectures
- Settling behavior
- Algorithm design
Wireless Transmission System

♦ Baseband Modeling

♦ Multi Rate Synchronous Dataflow Modeling due high oversampling rates

♦ Table based models for analogue imperfections

♦ Simulation time e.g. for BER of 16 billion bits take around 15 hours

Constellation diagram

phase noise only
balance error
quadrature error
bias error

Settling

BER

Quadrature Error * Pi [rad]
Conclusion

- SystemC-AMS extends SystemC for modeling analogue mixed signal behavior
- Together with SystemC, SystemC-AMS permits overall system modeling for different purposes
- Prototype available at: www.systemc-ams.org
- SystemC-AMS will be further developed and standardized by the OSCI AMS Working group – which is open for all OSCI members