Modeling Wireless Sensor Network Nodes Using SystemC-AMS

Michel Vasilevski, Hassan Aboushady, Francois Pêcheux and Laurent de Lamarre
University Paris VI, Pierre & Marie Curie
LIP6-SoC Laboratory, 75252 Paris, France

Abstract— **The paper presents a preliminary approach for the modeling and simulation of a complete Wireless Sensor Network with two nodes using SystemC-AMS, an open-source C++ library dedicated to the description of heterogeneous systems containing digital, analog, RF hardware parts as well as embedded software. The WSN node, or mote, detailed herein consists of a physical sensor, a continuous time sigma-delta converter with its associated decimation filter, an ATMEGA128 8-bit microcontroller running the embedded application and a QPSK-based 2.4 GHz RF transceiver. The paper starts with the structural description of the system as a hierarchical set of behavioural modules, then gives an insight on how multi-frequency simulation is handled in SystemC-AMS, and finally presents simulation results that are systematically compared with the Matlab reference in terms of accuracy and simulation time.**

I. Introduction

Needless to say that one of the great challenges of the next decade is pervasive/wireless computing. In this context, the ability to design optimal Wireless Sensor Networks is of paramount importance. To improve their competitiveness, major players in the microelectronics industry are faced with two antonymic issues : 1- the need to dramatically reduce the cost and design time of their products like SoCs or SIPs for economical reasons, 2- the lack of a unified design environment that can be used efficiently by system designers to model and simulate state-of-the-art systems (i.e. systems that encompass several research activity fields and combine on the same integrated circuit physics, analog and digital electronics, RF/micro-wave and software application). For the past 20 years, hardware description languages have been widely used to model and simulate systems belonging to various engineering fields, from digital and analog electronics to mechanics, RF and even battery cell chemistry. EDA industry proposed recently consistent modelling and simulation frameworks that allow for the description of systems from different disciplines and for the description of interactions between these systems. These frameworks use VHDL-AMS [1], [2] and Verilog-AMS [3] as effective backbones for the modeling. However, when dealing with WSN containing hundreds of nodes, and with a carrier frequency of 2.4 GHz, these frameworks show rapidly their limits in terms of interoperability and simulation performance. One possible solution to the modelling and simulation of ”More than Moore” multiprocessor heterogeneous systems is SystemC-AMS [4] [5], an extension to the existing library SystemC [6]. The first version has been released by Franhofer Gesellschaft EAS/IIS Dresden [7]. Figure 1, extracted from the SystemC-AMS documentation shows how the objective of multi-discipline modelling can be achieved with a set of interoperable userview layers corresponding to the aforementioned research fields. In practice, SystemC-AMS allows to describe mixed-signal modules and currently supports two user views and their associated semantic models: conservative and multi-rate Synchronous DataFlow (SDF). For the moment, the conservative view is restricted to linear networks and does not allow the design of real analog subsytems. For the level of modelling required by this system, the multi-rate synchronous dataflow approach is of much more interest. The key idea of this approach is to embed continuous-time modules into dataflow clusters. Modules communicate with others via directed data streams carrying time valued samples. A dataflow cluster may contain any number of dataflow modules whose execution is statically scheduled during simulation elaboration. A cluster is managed by a dedicated SystemC process that handles synchronization with the rest of the system. When scheduled by the SystemC simulation kernel, a dataflow cluster runs at a constant time step, defined by the sampling duration time assigned to one port of one of the modules and automatically propagated to others. Hence, SDF is specialy suited for communication systems like WSN with strong oversampling : a SystemC-AMS module can be seen as a simple dataflow class function (always named sig_proc()) which, at every time step, reads its SDF inputs, computes and accumulates results, and propagates them to the SDF outputs.

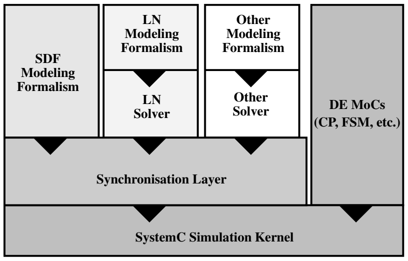


Fig. 1. SystemC-AMS architecture.

II. Wireless sensor network system

Structurally, a node consists of 4 parts, as shown in figure 2. One can notice that the node has four different clock domains (3.65 KHz, 8.55 MHz, 2.4 MHz, and 2.4 GHz), with frequency differences of several orders of magnitude. Section II.E shows how these domains are managed altogether by SystemC-AMS.

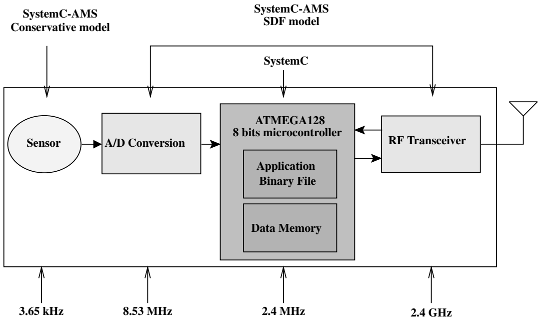


Fig. 2. Structure of a wireless sensor network mote.

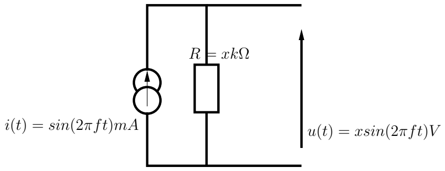


Fig. 3. Simple electrical model of a sensor.

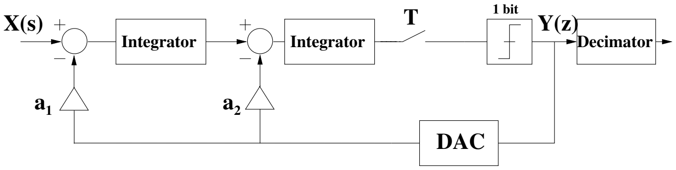


Fig. 4. Second order sigma-delta continuous-time modulator and decimator.

A. Sensor

The sensor part, described in figure 3, is quite unrealistic but is sufficient for our application. The current source and load resistor are modelled using the conservative view and means offered by SystemC-AMS. Listing 1 shows how the design of figure 3 can actually be coded in SystemC-AMS. The **sca_elec_port** (line 6) and **sca_elec_ref** (line 7) are conservative ports that obey GKL, and are very similar to VHDL-AMS terminals. The constructor **SC_CTOR** (line 17) instanciates the two SystemC-AMS linear devices (current source and resistor) and connects them according to figure 3. To be compatible with the ADC, the conservative value is converted into its synchronous dataflow equivalent through the use of the **sca_v2sdf** SystemC-AMS construct.

Listing 1. The sensor conservative subpart.

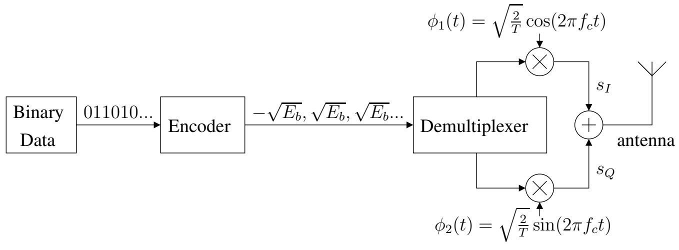
```
 1 #ifndef WAVE_H
 2 #define WAVE_H
 3
 4 SC_MODULE (wave)
 5 {
 6   sca_elec_port w1;
 7   sca_elec_ref gnd;
 8
 9   sca_isin  *i_sin_1;
10   sca_r     *i_r_1;
11
12   void init(double a, double f){
13     i_r_1->value = a*1000;
14     i_sin_1->freq = f;
15   }
16
17   SC_CTOR (wave) {
18     i_sin_1=new sca_isin("i_sin_1");
19     i_sin_1->p(w1); // pos
20     i_sin_1->n(gnd); // neg
21     i_sin_1->ampl=0.001; // magnitude in A
22
23     i_r_1=new sca_r("r1");
24     i_r_1->p(w1);
25     i_r_1->n(gnd);
26   }
27 };
28 #endif
```

B. A/D converter

To pragmatically experiment the capabilities of the SystemC-AMS library, the ADC that converts the analog measure coming from the sensor into its equivalent digital is nothing less that a second order sigma-delta 1-bit modulator with return-to-zero feedback [8], and a decimator using a third order FIR2 [9], that can be parameterized to generate a n-bit word. The oversampling rate can be set accordingly in order to follow particular specifications. In the SystemC-AMS methodology, the boxes presented in figure 4 are synchronous dataflow modules and correspond to one dataflow cluster, and therefore obey the semantic model of multi-rate synchronous dataflow graphs. Each SDF module is dedicated to the computation of a mathematical equation that consumes input data and produces output data. As an example, listing 2 gives the complete code for the integrator **integrator_sd** of the sigma-delta converter. The integrator, described as a SystemC AMS Synchronous Dataflow Module **SCA_SDF_MODULE** (line 4), takes two sdf ports as inputs, **in1** and **in2** (line 6), and generates the integrated result on **out**(line 7). The **init** member function (line 13) is called once during elaboration and allows to initialize the Laplace transform function matrix. The Laplace transform function is conveniently handled by the dedicated class **sca_ltf_nd** (line 11) and operates exactly like the VHDL-AMS construct. Each time the cluster process is triggered by the SystemC simulation kernel, the **sig_proc()** function (line 22) is called. **In1** and **in2** are read, the Laplace transform is performed, and the result written on the output **out**.

Listing 2. The complete SystemC-AMS source code for the SD integrator, using the Laplace transform SystemC-AMS native construct.

```
 1 #ifndef INTEGRATOR_SD_H
 2 #define INTEGRATOR_SD_H
 3
 4 SCA_SDF_MODULE (integrator_sd)
 5 {
 6   sca_sdf_in < double >in1, in2;
 7   sca_sdf_out < double >out;
 8
 9   double fs, ai, ki;
10   sca_vector < double >NUM, DEN, S;
11   sca_ltf_nd ltf1;
12
13   void init (double a, double f, double k) {
14     DEN (0) = 0.0;
15     DEN (1) = 1.0;
16     NUM (0) = 1.0/3.0;
17     ai=a;
18     fs=f;
19     ki=k;
20   }
21
22   void sig_proc () {
23     out.write(
24         ltf1(NUM,
25              DEN,
26              S,
27              fs*(ai*in1.read()-ki*in2.read())
28              )
29         );
30   }
31
32   SCA_CTOR (integrator_sd) {}
33 };
34 #endif
```



$\phi_1(t) = \sqrt{\frac{2}{T}}\cos(2\pi f_c t)$
Binary Data
011010...
Encoder
$-\sqrt{E_b}, \sqrt{E_b}, \sqrt{E_b}...$
Demultiplexer
s_I
antenna
s_Q
$\phi_2(t) = \sqrt{\frac{2}{T}}\sin(2\pi f_c t)$

Fig. 5. QPSK RF transmitter.

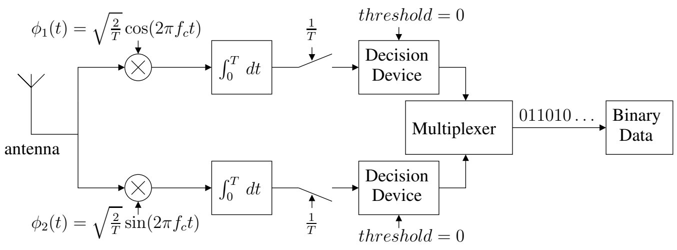


Fig. 6. QPSK RF receiver.

C. *ATMEL ATMEGA128 Microcontroller*

The ATMEGA128 microcontroller belongs to the ATMEL AVR devices [10]. The program executed by the microcontroller can be written in assembly language or directly in C. In the presented system, the 8-bit value coming from the ADC decimation filter is connected to digital port B of the microcontroller. The application software permanently reads port B, serializes the corresponding value and propagates the corresponding bitstream on pin 0 of port A. When receiving RF data, microcontroller reads pin 0 of port C.

D. *2.4 Ghz QPSK RF transceiver*

The RF transceiver, presented in figures 5 and 6 is responsible for converting the digital bitstream into RF information and vice versa. It uses a QPSK (Quadrature Phase Shift Keying) transmission scheme, with a **Fc** carrier frequency and a **Fb** data frequency. AWGN (Additive White Gaussian Noise) allows to take into account channel noise in the modelling of the RF communication channel and is necessary for calculating the fundamental RF characteristic BER (Bit Error Rate) vs. SNR (Signal-to-Noise Ratio). The QPSK transmitter consists of a 1-bit D/A converter, a demultiplexer, two mixers and a signal adder. The D/A converter, also called encoder shifts voltage levels of the input bitstream following energy user specifications. Multiplied by a cosine or sine oscillation into the mixer, signals **sI** and **sQ** are combined in the adder, this results in a QPSK-modulated signal. The received signal is mixed with a cosine and sine oscillation and each part is integrated. The integration of cosine mixed part permits to extract the I-part of information, Q-part is extracted from integration sine mixed part. A decision device digitalizes positive values to symbole '1' and negative values to symbole '0'. This block permits a hardware transmission error correction while the value received does not change its sign. The signal is finally multiplexed and get totally rebuilt. One can notice that neither the power amplifier (PA) nor the low noise amplifier (LNA) have been implemented in the design. Likely, the communication channel is considered as an ideal gain block with an additive white Gaussian noise (AWGN).

E. *Multi-frequency modelling and module interfacing*

In a SystemC-AMS design with a single clock domain, the **sig_proc()** function of each connected module of a dataflow cluster is called periodically, on the basis of the cluster sample duration time. The system designer can setup sample duration by assigning a value to a module port through the use of the **set_T()** function. The simulation sample duration is automatically propagated to each connected module of the cluster. To allow multi-frequency simulation, SystemC-AMS proposes the **set_rate()** function that can be called on a specific module port to define its relative sample rate.

III. Simulation results

For simulation purposes, some parameters have been set up, for each component. ADC oversampling rate is 64, decimator produces 8 bits. Gain values of $\Sigma\Delta$ feedback loop are specified from amplitude histogram analysis, they are set to 2 and 7/6. We used a 2.4 MHz clock frequency for microcontroller, so bit rate for RF transmission is 2.4 Mbps. Carrier frequency is 2.4 GHz.

Simulation sample frequency is set to 2.4 MHz for multiplexer output or demultiplexer input, these signals are converted from or to systemC digital signals so we need just one sample per bit.

After validation of behaviour, we analyse simulating durations (table I). We timed Matlab/Simulink vs. SystemC-AMS simulations, controlling the exact equivalent behaviour and the same number of samples used. Simulation of communication between 2 motes cannot be performed with Matlab, because of the complexity of microcontroller modelling. This problem reveals the advantage of SystemC-AMS simulation : we are able to simulate both digital and analog models simultaneously, Matlab/Simulink doesn't contain microcontroller models yet.

Now we verified accuracy of results with some more technical tests. We simulated a variable sine amplitude input, to compute SNR characteristics of ADC and we compared with similar matlab/simulink model result (fig. 7). Then, we set optimal sine amplitude for maximal SNR. In fact, the better amplitude is -5 dB, so we used $a = 0.56$. We can observe frequential responses to validate ADC behaviour and to compare with Matlab results. In the RF part, we wanted to run an analysis to visualise bit error rate according SNR variation, we needed 10 Kbits. Bit error rate is the number of bad received bits divided by the number of transmitted bits. A theorical BER is computed from AWGN characteristics and is compared with simulation results. Figure 9 shows similarities between simulation and theorical results.

RF mixer module has to compute 10000*10000 sample multiplications with cosine for each I and Q part. An other transmission modelling solution, like baseband equivalent, should be used soon. In this case we will be able to abstract carrier frequency transmission and reduce simulation sample time. Representations on figure 10 permit to image received symbol dispersion, according to different

TABLE I
Simulation results.

	Configuration	Simulation	Matlab	SystemC-AMS
ADC	OSR=64 8 bits output	1 ms 16*1024 pts for output	1.605s	0.934s
RF	2.4 GHz carrier freq.	416.67 μs 10^3 pts for μc input, 10^7 pts for RF i/o	2m30.746s	54.360s
2-mote WSN	Same settings	416.67 μs	–	4m19.572s

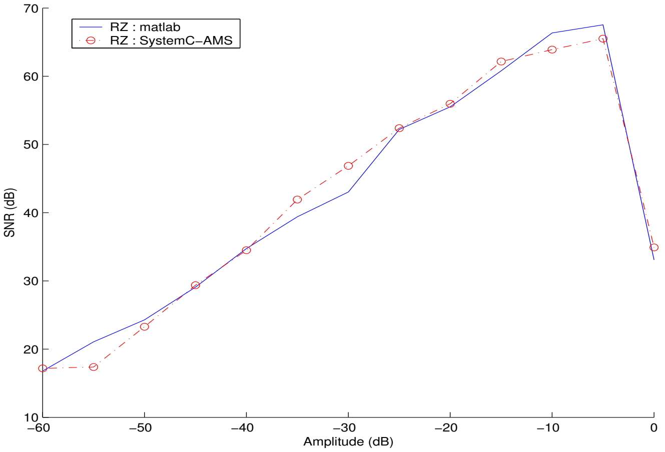


Fig. 7. SNR analysis for ADC in relation to input amplitude.

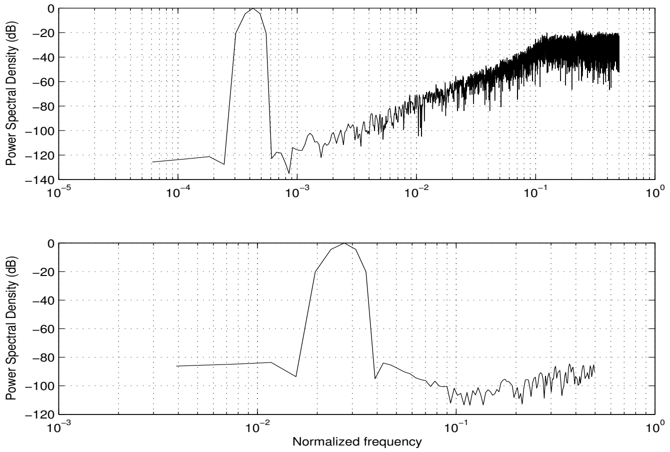


Fig. 8. Frequency response analysis of Sigma-Delta and decimator outputs.

non-idealities. Those figures represent 2-bit symbols received when a non-ideality is present. We can see the value dispersion and we can remember (section II.D) that this value is corrected by decision device when the dispersion is not too strong.

IV. Conclusion

In this paper, we have demonstrated that it is possible to model a heterogeneous system containing a sensor, an A/D , a micro-controller, as well as a RF transceiver analog, using the same description language, SystemC-AMS. All the tools needed to obtain simulation results are open source. Each analog/RF block has been verified and its performance has been compared to the performance of the same models under MATLAB. Simulation time for the SystemC-AMS models have been compared with MATLAB models.

We have also presented the prelminary results obtained from the simulation of two nodes WSN coded in C++, SystemC, and SystemC-AMS. This work can be seen as a starting point for the modelling of a more ambitious network with several nodes.

References

[1] J. Ravatin, J. Oudinot, S. Scotti, A. Le-clercq, and J. Lebrun, “Full transceiver circuit simulation using VHDL-AMS,” *Microwave Engineering*, May 2002.

[2] F. Pecheux, C. Lallement, and A. .Vachoux, “VHDL-AMS and Verilog-AMS as Alternative Hardware Description Languages for Efficient Modeling of Multi-Discipline Systems,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems(TCAD)*, Feb. 2005.

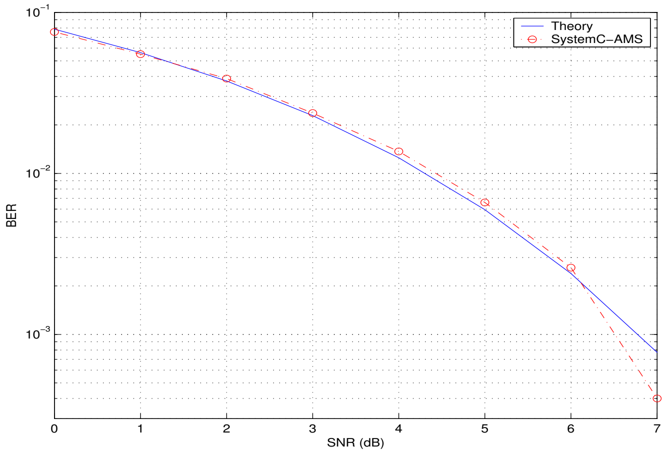


Fig. 9. Bit error rate for QPSK transmission through an AWGN channel.

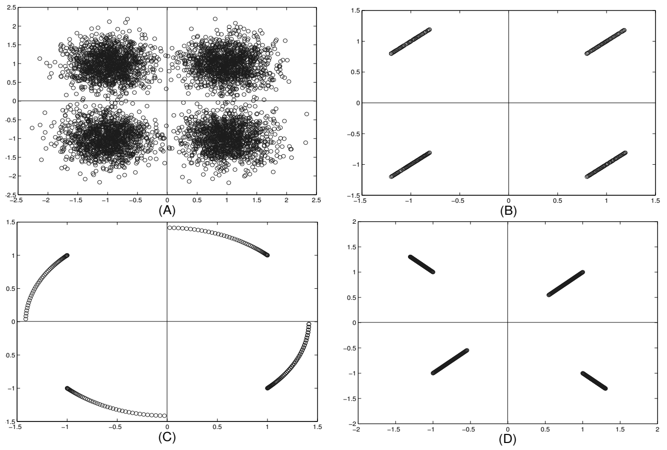


Fig. 10. Constellation of symbols received from QPSK transmission through AWGN channel (A), with a DC offset (B), frequency offset (C) and phase mismatch (D).

[3] P. Frey and D. O’Riordan, “Verilog-AMS: Mixed-signal simulation and cross domain connect modules,” *Proc. 2000 IEEE/ACM International Workshop on Behavioral Modeling and Simulation (BMAS), 2000, pp. 103 108.*

[4] A. Vachoux, C. Grimm, and K. Einwich, “Analog and Mixed Signal Modelling with SystemC-AMS,” *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2003.

[5] “SystemC-AMS 0.15,” Oct. 2006, http://www.systemc-ams.org/documents/systemc-ams-0-15.pdf.

[6] “SystemC,” http://www.systemc.org.

[7] “SystemC-AMS,” http://www.systemc-ams.org.

[8] H. Aboushady, F. Montaudon, F. Paillardet, and M. M. Louerat, “A 5mW, 100kHz Bandwidth, Current-Mode Continuous-Time Sigma-Delta Modulator with 84dB Dynamic Range,” *IEEE European Solid-State Circuits Conference (ESSCIRC) Florence,Italy*, 2002.

[9] H. Aboushady, Y. Dumonteix, M. Louerat, and H. Mehrez, “Efficient Polyphase Decomposition of Comb Decimation Filters in Sigma-Delta Analog-to-Digital Converters,” *IEEE Trandactions on Circuits and Systems-II (TCASII)*, Oct. 2001.

[10] “8-bit AVR Microcontroller with 128K bytes in-System programmable flash ATmega128 Datasheet,” Oct. 2006, http://www.atmel.com.