From System Specification To Layout: Seamless Top-Down Design Methods for Analog and Mixed-Signal Applications

R. Sommer[1], I. Rugen-Herzig[1], E. Hennig[1], U. Gatti[2], P. Malcovati[3], F. Maloberti[3], K. Einwich[4], C. Clauss[4], P. Schwarz[4] , G. Noessing[5]
[1]Infineon Technologies AG – Germany
[2]Siemens ICN – Italy[3], University of Pavia – Italy
[4]FhG IIS – Germany, [5]Infineon Technologies AG – Austria

Abstract

Design automation for analog/mixed-signal (A/MS) circuits and systems is still lagging behind compared to what has been reached in the digital area. As System-on-Chip (SoC) designs include analog components in most cases, these analog parts become even more a bottleneck in the overall design process. The paper is dedicated to latest R&D activities within the MEDEA+ project ANASTASIA+. Main focus will be the development of seamless top-down design methods for integrated analog and mixed-signal systems and to achieve a high level of automation and reuse in the A/MS design process. These efforts are motivated by the urgent need to close the current gap in the industrial design flow between system specification and design on the one hand and block-level circuit design on the other hand. The paper will focus on three subtopics starting with the top-down design flow with applications from circuit sizing, design centering, and automated behavioral modeling. The next part focuses on modeling and simulation of specific functionalities in sigma-delta design while the last section is dedicated to a mixed-signal System-on-Chip design environment.

1. Introduction

Electronic Design Automation (EDA) is a key factor for fast and efficient development of complex mixed-signal/mixed-domain systems. The ultimate goal of EDA is to provide a comprehensive top-down design method with secured constraint propagation between all design steps from system definition down to integrated-circuit mask generation. In this respect, the techniques and tools that are available today for the design of analog and mixed-signal components are lagging considerably behind the ones from the digital domain. Therefore, as soon as a system includes some analog functions, EDA efficiency and comprehensiveness are dramatically reduced. In the following sections three topics which are in the focus of the ANASTASIA+ project are presented. At first, a top-down design flow with applications from circuit sizing, design centering, automated behavioral modeling, and automated layout generation will be described. Following, modeling and simulation of specific functionalities for $\Sigma\Delta$ design will be demonstrated. The third topic will cover a mixed-signal System-on-Chip design environment.

2. Top-Down Design

2.1. Overview of the analog top-down design flow

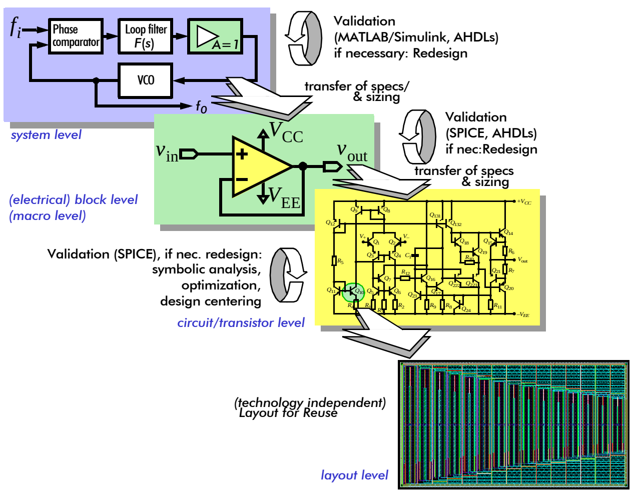

f_i
Phase comparator
Loop filter F(s)
A = 1
VCO
f_o
system level
Validation (MATLAB/Simulink, AHDLs) if necessary: Redesign
transfer of specs/ & sizing
v_{in}
V_{CC}
V_{EE}
v_{out}
(electrical) block level (macro level)
Validation (SPICE, AHDLs) if nec:Redesign
transfer of specs & sizing
Validation (SPICE), if nec. redesign: symbolic analysis, optimization, design centering
circuit/transistor level
(technology independent) Layout for Reuse
layout level

Figure 1: Top-down mixed-signal design-flow incorporating bottom-up knowledge and backannotation of low-level effects

Fig. 1 gives an overview of the analog part of the top-down design and refinement process which is addressed by the ANASTASIA+ project. Starting with the system-level design, the specifications and interfaces of the next refinement step will be derived. Here, some bottom-up knowledge has to be brought in since the different blocks will have to be redefined and split into new subblocks until the whole circuit structure only contains subblocks which have a realization on transistor level (top-down). Macro- and behavioral modeling as well as sizing and optimization techniques will help to guide the whole design process.

2.2. Mixed-Signal/Mixed-Level System Description and Simulation Models

While specialized EDA tools are already available for many individual design stages, no satisfactory solutions exist yet for performing the step from one level to the next and for system simulations with mixed abstraction levels. Actually, tools such as Matlab, Cossap, Saber, or Mathematica as well as modeling languages such as VHDL, VHDL-AMS or C/C++ can be used for either system or block design but, currently, not for both at the same time.

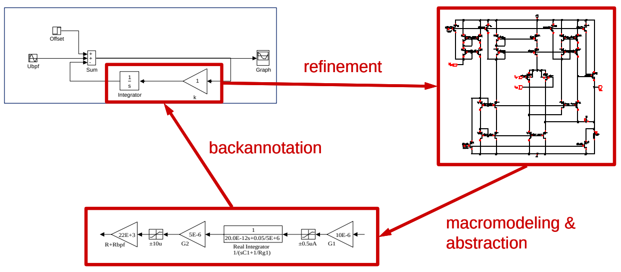


Figure 2: Example of a mixed signal/mixed level system description as part of the refinement and macromodeling process

Fig. 2 shows different abstraction levels of a block in the development process. Starting on system level with a MatLab/Simulink model, a subblock –here an integrator block– has been selected for the subsequent refinement step. The refined block on transistor level is shown on the right-hand side in Fig. 2. Once the topology has been sized according to the specification on block level (e.g. for gain and the integration constant) it is necessary to derive a macromodel from this transistor-level block in order to annotate it back to system level. The refined system-level subblock (bottom of Fig. 2) will then be used for a *verification on system level*. In this way it can be analyzed on system level whether e.g. additional (parasitic) poles and zeros of the transistor level circuit block have an influence on the overall system behavior, and whether a modification of the system-level model (topology or block specifications) will be necessary.

From the above example it becomes obvious that the ultimate goal is to have one description for all levels and a seamless methodology to handle the different abstraction levels as well as the refinement and backannotation processes. For practical use an integration into flow and framework together with the incorporation of automatic behavioral model generation is mandatory.

Several parts in reaching this goal are missing or incomplete. The most obvious break is a missing link between the signal flow or MatLab/SimuLink level and the transistor level. On the one hand, this is due to the lack of interfaces between tools used on different levels. On the other hand, the above-listed languages and tools do not have the capabilities needed for the description and simulation of all required levels. Moreover, standards like VHDL-AMS have only recently emerged. Thus, they did not have a noticeable effect on today's design practice yet, and these languages still lack important capabilities (like frequency domain or synthesis and sizing support in case of VDHL-AMS). Moreover, the evaluation of different AHDL simulators showed large deficiencies in the subsets of their language support.

2.3. Circuit Sizing, Design Centering, and Automated Behavioral Modeling Techniques

Analog circuit design on the block level comprises the following main steps: topology creation/selection, sizing, and design centering as well as characterization (Fig. 3). Today, topologies are mostly designed manually with a schematic editor or selected from a (fixed) topology database; there is still little CAD support for generating new topologies. On the other hand, major progress in EDA for interactive circuit sizing has been made in the Medea project SADE, a predecessor project of ANASTASIA+. Yet, a number of important problems, such as sizing with respect to noise or RF characteristics, are still on the agenda.

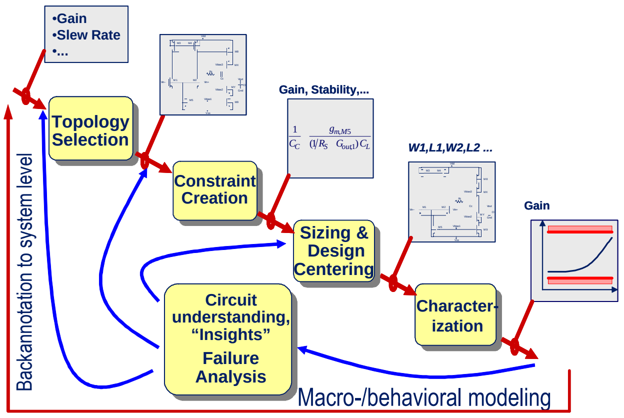

•Gain
•Slew Rate
•...
Gain, Stability,...
W1,L1,W2,L2 ...
Gain
Topology Selection
Constraint Creation
Sizing & Design Centering
Character-ization
Circuit understanding, "Insights" Failure Analysis
Backannotation to system level
Macro-/behavioral modeling

Figure 3: Analog block design environment: From block specification to sized circuit topology

Design centering

After nominal design, design centering is usually applied to increase the yield. Current design centering tools incur high initialization costs due to the time-consuming data preparations needed for a centering run. Moreover, the time and cost factor of worst-case calculations using Monte Carlo simulations is unreasonably high with today's design complexities. Another challenge is posed by local parameter variations (e.g. mismatch); their influences increase strongly in the deep sub-micron range. At project start there have been no satisfactory solutions for analyzing these influences or for their optimization using statistical or deterministic methods. First successful project results include the development of an algorithm for calculating performance specific worst-case parameter sets. Moreover, a new approach in mismatch analysis has been realized which allows for selection of transistors with large influence (by sensitivity analysis) as well as for an identification of mismatch-relevant transistor pairs (by simulation) [18].

Behavioral modeling

Behavioral models play an important role in modern circuit design. They serve as executable specifications in top-down system design and as abstract descriptions of functional blocks needed for a reuse-oriented design style. Traditionally, behavioral models are coded manually. As this is a time-consuming task, the level of automation needs to be increased. Methods for automated model generation have been developed in the SADE project; however, as of project start, only small-signal and DC behavior could be modeled automatically. Meanwhile first promising results have been achieved in processing transient behavior. New ranking methods which find non dominant parts in behavioral equations have been developed and implemented in the symbolic analysis toolbox Analog Insydes [16, 17]. Additionally, optimal generation of HDL templates from model equations for optimal simulation performance is under investigation. This issue is strongly related to the activities presented in section 2.2 since both model and the simulator have to fit together: the most compact model or the best description language is useless if it cannot be simulated efficiently.

Layout generation

For analog blocks most of the layout design is still done manually. A high number of constraints, especially for RF circuits, have to be considered in order to avoid too many parasitic effects which will require a loop back after having resimulated the layouted circuit. While layout generation for devices is already state of the art, the automation on block level lacks behind: the first commercial tool is currently entering the market, although in academia and company internal research institutes very promising results have been achieved during the last 5 years.

The work in the ANASTASIA+ project is focused on methodologies for automated (reusable) layout generation for analog/mixed-signal macros. Two aspects are being investigated: how to generate reusable layout and how to automatically adapt existing layout to a resized circuit or a migrated (between similar processes) one.

To achieve this goal, it is essential to combine module generation on device and macro level with efficient place and route methodology as well as compaction technology. Constraints, e.g. for mismatch (symmetry, device shape and orientation), crosstalk and substrate coupling or rules for device merging have to be considered in all steps.

3. Modeling and Simulation of Specific Functionalities – ΣΔ

Sigma-delta (ΣΔ) modulators are the most suitable A/D converter topologies for digitizing with high-resolution analog signals characterized by a bandwidth much smaller than the sampling frequency. With these architectures, resolutions up to 19-21 bits [1] and/or sampling frequency higher than 50 MHz [2] can be reached using standard IC technologies. These features make the ΣΔ solutions very attractive for a number of applications, such as audio systems, receivers for communication apparates, sensor interface circuits and measurement systems. Furthermore, their inherent linearity, robust analog implementation and low sensitivity to analog component imperfections are additional key advantages.

ΣΔ modulators can be implemented either with continuous-time or with sampled-data techniques, but the most popular approach is based on a sampled-data solution with switched-capacitor (SC) implementation. Indeed, SC ΣΔ modulators can be efficiently realized in standard CMOS technology and included in complete mixed-signal systems without any performance degradation. For this reason in this paper we will deal only with simulation methods for SC ΣΔ modulators.

In ΣΔ modulator design the need for dedicated computer tools is particularly severe. Indeed, they are mixed-signal non-linear circuits and hence time-domain simulations are mandatory for the optimization of their performance, especially if high-resolution or high-speed systems are considered. In principle various approaches for the transient simulation are already available, like device models (e. g. SPICE), finite-difference equations (e. g. SWITCAP), custom numerical models (typically in C++ language), etc. However, in different measures all of them exhibit some disadvantages. Fig. 4 [3], classifies the different tools in terms of three main characteristics: accuracy, speed and flexibility (intended as modeling capability plus reusability). Moreover, the post-processing algorithms for the evaluation of modulator performances (SNR, IP3, ENOB, histograms, etc.) are other qualifying features for the various tools.

Approach	Accuracy	Speed	Flexibility
Device models [4]-[5]	☺	☹	😐
Custom models (C++) [3]	😐	☺	☹
Finite-difference equations [6], [7]	😐	☺	😐
Circuit-based macromodel [8]	☺	☹	☺
Time-domain macromodel [9]	☺	😐	☹
Table-lookup model [10]	😐	😐	☹
Behavioral models (TOSCA) [11]	☺	😐	☹
Proposed solution	☺	☺	☺

Figure 4: Comparison among the different tools for the simulation of ΣΔ modulators

SPICE is a conventional electrical simulator and, despite its precision, it is not suitable for the analysis of ΣΔ modulators because of the extremely long simulation time. Depending on desired accuracy and oversampling ratio, tens of thousands sampling periods are needed just to achieve a single data point for the SNR-vs-input amplitude curve with an unacceptable waste of time.

Custom models are just suited for a specific structure and cannot be easily adapted to a different modulator topology (especially regarding exotic architectures). This situation is quite difficult to handle when accurate simulations of a number of non-idealities and, eventually, the performance comparison between different architectures are needed.

The circuit-based macro-model essentially represents an equivalent circuit built up with a minimum set of passive and active devices already available in electrical simulators like SPICE. Modulator building blocks are described by means of simplified circuits, and specifications are used as model parameters. Non-idealities can be introduced in the models. This approach guarantees a good degree of accuracy and reusability, but the speed improvement with respect to device-level simulation is poor.

Time-domain macro-models are based on a set of equations describing the transient behavior of a specific circuit. The specifications of the circuit represent the model parameters. Again this approach is not flexible at all, but allows us to introduce dynamic non-linearities.

The simulators based on finite-difference equations are programs usually written in C language that exploit the z-domain description of the transfer function of sample-data networks. They can be general-purpose like SWITCAP (or its evolution AWEswit [9]) or especially devoted to oversampled modulators like MIDAS. They achieve an excellent speed of simulation, but the non-idealities modeling capabilities are poor. Moreover, both simulators operate on netlists and do not offer a user-friendly human interface.

Table-lookup models use a two steps procedure. First tables of input and output points are extracted for the ΣΔ sub-blocks by using conventional electrical simulators. Then, the obtained tables are utilized instead of the original circuit for global transients simulations. However, this approach does not seem to guarantee high accuracy (< 80 dB) in SNR estimation (static errors only), and tables are not reusable. The speed of this approach depends on the size of the tables.

TOSCA is a behavioral general-purpose program for the simulation of oversampling converters. It is possible to describe a ΣΔ modulator at building-block level, such as integrators and quantizer. Thus, with some limitations, it allows the description of a number of different topologies.

However, it is not possible to incorporate in TOSCA user-defined models or to introduce new limitations unless the main code is modified. Moreover, it is based on a netlist description.

Further behavioral simulator with a more custom approach, like SSDSIM [10] or DSM [11], represents a compromise between finite-difference equations and TOSCA achieving intermediate performances.

The proposed behavioral modeling technique, based on a complete set of blocks described using standards high-level languages like Simulink [12], [13] and VHDL-AMS [14], guarantees a good compromise for the simulation of ΣΔ modulators in terms of speed, modeling capabilities and reusability with parameters adjustments.

Moreover, the libraries of models can be easily integrated in popular design flows [15], thus allowing the analysis of the modulator structures at different level of abstraction and with a complete control of the project at various stages of the design process.

The proposed modeling toolbox allows us to simulate any sigma-delta modulator architecture (either low-pass or band-pass), with single or multi-bit quantizer, including most of the building blocks' non-idealities, such as thermal noise and limited operational amplifier performances. The models use behavioral equations to calculate the effect of the non-idealities on the output of each block at the end of each clock cycle, thus strongly reducing the simulation time (only one point per clock period has to be calculated). Moreover, the library contains specific functions for the post-processing of the output data stream allowing the calculation of most popular figures, such as SNR-vs-Amplitude, Signal-to-Noise and Distortion Ratio (SNDR), etc.

As an example, let us consider the second-order low-pass SC ΣΔ modulator shown in Fig. 5, modeled both with SIMULINK® and VHDL-AMS. The design parameters used for the simulations are summarized in Fig. 6. These values correspond to the typical performance required for sensor applications. In particular, in this case a minimum SNDR of 96 dB (i.e. a resolution of 16 bits) is required.

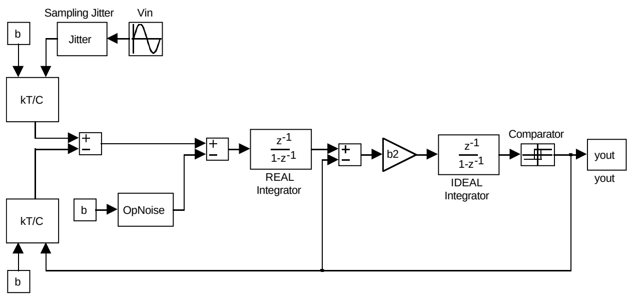


Figure 5: Low-pass second-order ΣΔ modulator model

Parameter	Value
Signal bandwidth	BW = 100 Hz
Sampling frequency	f_s = 50 kHz
Oversampling ratio	M = 250
Number of samples considered	N = 65536
Integrator coefficients	$b = b_2 = 0.5$

Figure 6: Parameters of the second order low-pass ΣΔ modulator model

Fig. 7 shows the power spectral density (PSD) of the modulator output obtained in simulation with the operational amplifier finite bandwidth and slew-rate model. The slew-rate (SR) and bandwidth (GBW) values used in the simulation are 0.1 V/µs and 100 kHz, respectively. The finite bandwidth and slew-rate lead to harmonic distortion, thus degrading the SNDR performance of the ΣΔ modulator.

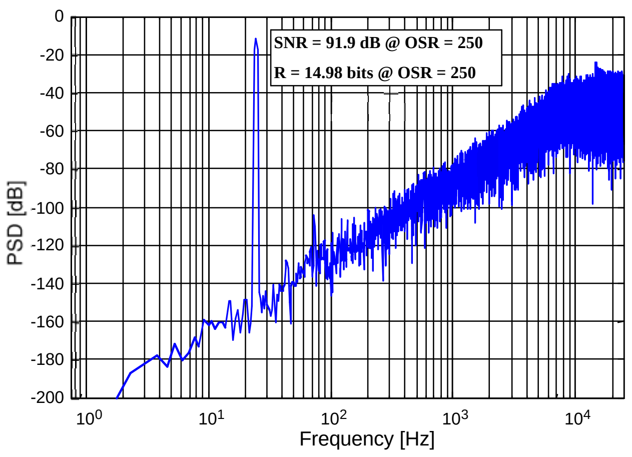


Figure 7: PSD of the ΣΔ modulator output with the operational amplifier finite bandwidth and slew-rate model

SD Modulator Parameter	SNDR VHDL-AMS	SNDR Simulink
Ideal modulator	98.5 dB	99.1 dB
Sampling jitter (Δτ = 16 ns)	98.6 dB	98.8 dB
kT/C noise (C_s = 1.25 pF)	93.4 dB	94.0 dB
Input-referred operational amplifier noise (V_n = 73 µVrms)	94.5 dB	94.8 dB
Finite dc gain ($H_0 = 1\cdot10^3$)	97.6 dB	98.9 dB
Finite bandwidth (GBW = 11.25 MHz)	98.5 dB	99.1 dB
Slew-rate (SR = 4 V/µs)	98.5 dB	99.1 dB
Modulator simulated including all of the non-idealities	91.4 dB	91.6 dB
Measurement results on the integrated prototype	90.2 dB	

Figure 8: SNDR and resolution of the second order low-pass ΣΔ modulator

Finally, Fig. 8 compares the SNDR of the ideal modulator, which is the maximum obtainable with the architecture and parameters used, with the SNDR achieved with the same architecture when one single non-ideality at a time is introduced. The data are obtained either with Simulink and VHDL-AMS. In both environments we implemented the same equations. The small deviations between the results obtained are due to the difference in the solvers used in the two environments and to the inherent fluctuations of the random number used to model the noise. Moreover, the overall SNDR dB achieved in simulation considering all of the non-idealities is compared with the measured data obtained on the integrated prototype, fabricated using a double-poly, double-metal 2 µm CMOS technology. The values of the parameters used in the simulations correspond to the design parameters of the chip.

4. Mixed-Signal System on Chip design environment

4.1. Motivation

As already pointed out in the introductory section, the complexity of integrated solutions for mixed-signal applications is increasing rapidly. Thereby, the interconnection between software, analog and digital hardware is becoming tighter while the number of connections is increasing and the level of interaction is becoming much more complicated. As a consequence, for the design of digital hardware, software and algorithms of such systems, it is essential to include the analog components and the system environment into the overall simulation. Therefore, simulation performance is crucial. While hardware description languages like VHDL-AMS or Verilog-AMS principally allow a description of a mixed-signal system the description style especially of higher abstraction levels is often not appropriate (and thus not efficiently simulatable): The performance of currently available mixed-signal simulators is orders of magnitude too low for an overall system verification of a complex SoC.

To overcome this gap, there have been a lot of activities for a wide usage of C++ as a hardware description language (SystemC [21], SpecC [20], Cynlib [19], OCAPI [23]), whereby SystemC seems to become a standard. Unfortunately, these activities covered only digital domains (hardware, software and communication). Hence, the importance of an analog and mixed-signal extension of the C++ methodology is also pointed out in the MEDEA road map [24]. Consequently, the SystemC road map has scheduled a mixed-signal extension for version 4.x which is planed for 2003. First concepts were discussed in [26], [25] and [27].

4.2. Extensions

Fig. 11 shows a simplified view of a mixed signal system of a Subscriber Line Interface and Codec Filter (SLICOFI) [22]. The different domains, their interconnections and the preferable Model of Computation (MoC) are shown.

To simulate such a system efficiently, the current scope of SystemC has to be extended by two Models of Computation (Fig. 9), a linear DAE solver and a frequency-domain solver. Both have been developed as prototype within the project.

A linear equation solver is sufficient to handle many applications on system level. Thus, blocks consisting of transfer functions, state-space equations or linear electrical networks can be modeled. In conjunction with the static dataflow MoC, it could be demonstrated that this solver works very efficiently [25].

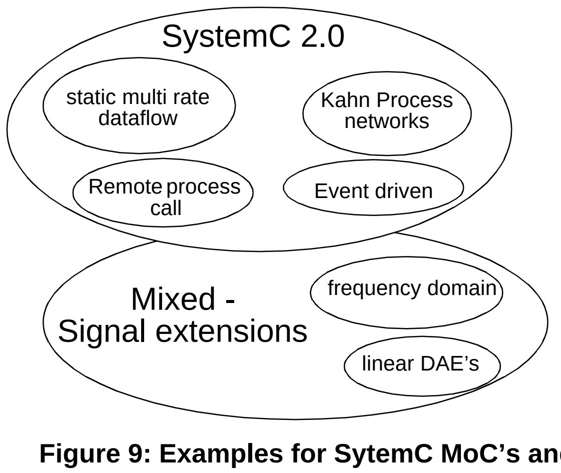


Figure 9: Examples for SytemC MoC's and proposed extensions

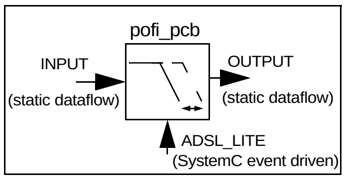
Especially telecommunication systems are specified in the frequency domain. Consequently, the first design steps are done in the frequency domain, which also means that a time-domain implementation has to be verified using a frequency-domain specification. Therefore, a requirement was to implement a „block-oriented" frequency-domain solver. To tackle this problem, frequency-domain behavior of blocks with an embedded linear equation system can be calculated with the same equations used for the time domain. In the current implementation, the user must provide a frequency-domain description for event driven and dataflow primitive blocks (like digital filters).

4.3. Implementation principles

The linear equation solver was implemented to handle and to solve equations embedded in a dataflow domain (transfer functions, state space equations, ...) as well as linear networks.

```
SDF_MODULE(pofi_pcb) {
 sdf_inport<double> INPUT;                  //dataflow inport
 sc2sdf_inport<bool> ADSL_LITE;             //SystemC inport
 sdf_outport<double> OUTPUT;                //dataflow outport
                                      :
     void init() {
     double wpre0;                          double wpre1;
     wpre0=2.0*M_PI*FG0;                    wpre1=2.0*M_PI*FG1;
     A0(0)=1.0;                             A1(0)=1.0;
     A0(1)=1.41/wpre0;                      A1(1)=1.41/wpre1;
     B0(0)=K;                               B1(0)=K;            }

 void sig_proc() {
     if(ADSL_LITE) OUTPUT=LTF(A1,B1,S,ltf_id1,INPUT);
     else          OUTPUT=LTF(A0,B0,S,ltf_id0,INPUT);
 }
                                      :
```



$$H(s) = \frac{K}{1 + \frac{1}{2\pi FG} s}$$

Figure 10: Transfer function embedded in dataflow block with event driven control input port

As an example of the former, Fig. 10 shows the system-level model of an analog post-filter using a linear transfer function representation (LTF) embedded in a static dataflow block.

For the latter, node voltages and (if necessary) some branch currents of a linear circuit have to be calculated. Hence, a system of equations has to be set up from a netlist description of the network. Using the modified nodal analysis method (MNA), the matrices can be simply constructed by utilizing the typical contributions („stamps") of each linear circuit element. Models of linear elements like R, L, C, controlled sources, linear transformers and transmission lines (by scattering parameters) have been implemented in this way.

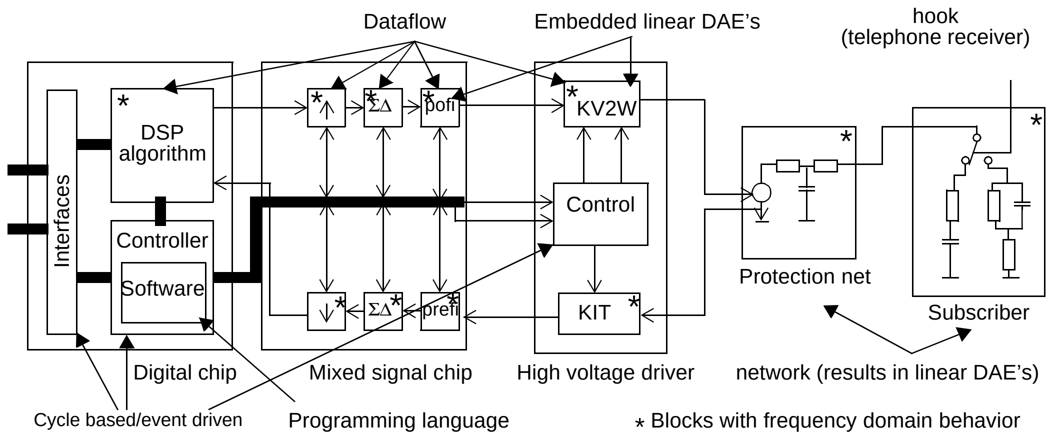


Figure 11: Use of different MoC's in a mixed-signal design

```
                                    :
elec_wire w1, w2, w3, w4, w5, tr;          //nodes
elec_gnd gnd;                              //reference node

   double Rp1=60.0, Rp2=40.0;              //parameter
   double Cp=1e-12, Lp=1e-3;

     V vbslic(w1,gnd,2.0);
     R rp1 (w4,w2,Rp1);
     R rp2 (w2,w3,Rp2);
     C cp  (w2,gnd,Cp);
     L lp   (w3,tr,Lp);
     CCVS i2v (w5,gnd,w1,w4,1.0); //Current Controlled Voltage
```

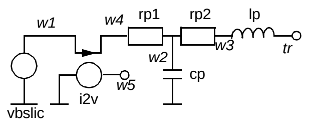


Figure 12: C++ network example

Fig. 12 shows an example of a simple C++ netlist description.

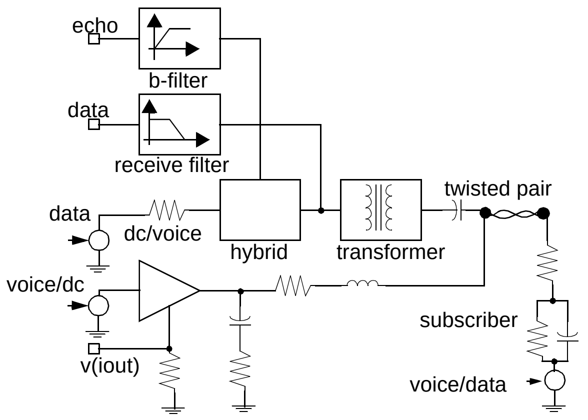


Figure 13: ADSL line driver front-end network

In the current version, each independent network will be included during simulation set-up as a dataflow primitive block. The input and output ports of this block are derived from the connections between dataflow blocks and the analog network (e.g. a dataflow input to a voltage source or a node voltage which is used as input for a dataflow block). If this dataflow block is called from the scheduler, a time interval equal to the time distance of two samples is calculated by the DAE solver.

4.4. Results for the analog extension

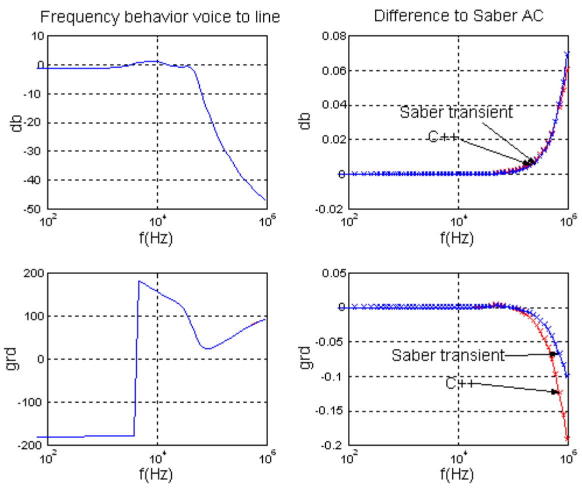


Figure 14: Transient simulations compared in frequency domain

Fig. 13 shows the top-level circuit topology of an ADSL line driver front-end. This front-end is modeled by approx. 50 linear network elements (R, L, C, controlled sources). Fig. 14 shows the simulation results compared to a Saber AC and transient simulation. The AC simulation is used as reference. For the Saber and C++ transient simulations, a multi tone signal with logarithmically spaced frequencies has been used. By post-processing of the transient simulation results with Matlab (FFT, ...), the transfer function in frequency domain was calculated and compared with the AC simulation. The sample rate was fixed to 17 MHz by the ADSL chip set. For 40 ms of real time Saber took with default parameters 2670 sec. whereas the C++ description took 122 sec. It should be mentioned that the time for applying the input signal and tracing the output signals turned out to be negligible.

5. Summary and Conclusion

The ANASTASIA+ project is focussed on a seamless top-down design flow for analog and mixed-signal applications. Progress has been achieved in many different areas of the project: Description languages and simulation models are being investigated. Thus, gaps in block design automation are being detected and closed by novel techniques in circuit sizing, design centering, and behavioral modeling. A new modeling toolbox for simulating any ΣΔ modulator architecture has been developed. Finally, a mixed-signal SoC design environment is being developed which extends the current SystemC standard by analog functionality to allow for handling of linear systems (transfer functions, linear electrical networks) in time and frequency domain.

6. References

[1] O. Nys and R. K. Anderson, ”A 19-Bit Low-Power Multibit Sigma-Delta ADC Based on Data Weighted Averaging”, IEEE Journal of Solid-State Circuits, vol. 32, pp. 933 -942, Jul. 1997.

[2] Y. Geerts, M. S. Steyaert, W. Sansen, ”A High-Performance Multibit CMOS ADC”, IEEE Journal of Solid-State Circuits, vol. 35, pp. 1829 -1840, Dec. 2000.

[3] S. R. Norsworthy, R. Schreier, G. C. Temes, ”Delta-Sigma Data Converters. Theory, Design and Simulation”, IEEE Press, Piscataway, NJ, 1997.

[4] X. Li, M. Ismail, ”Fast Simulation of Sigma-Delta Modulators Using SPICE”, Circuits& Devices Magazine, pp. 7-9, Jan. 1999.

[5] K. Suyama, S. Fang, Y. Tsividis, ”Simulation of Mixed Switched-Capacitor/Digital Networks with Signal-Driven Switches”, IEEE Journal of Solid-State Circuits, vol. 25, pp. 1403 -1413, Dec. 1990.

[6] L. A. Williams, B. Wooley, ”MIDAS – A Functional Simulator for Mixed Digital and Analog Sampled Data Systems”, Proc. IEEE Int. Symp. Circuits and Systems, 1992, pp. 2148-2151.

[7] J. C. Lin, J. H. Nevin, ”A Modified Time-Domain Model for Nonlinear Analysis of an Operational Amplifier”, IEEE Journal of Solid-State Circuits, vol. 21, pp. 478 -483, June 1986.

[8] V. Liberali, V. F. Dias, M. Ciapponi, F. Maloberti, ”TOSCA: A Simulator for Switched-Capacitor Noise-Shaping A/D Converters”, IEEE Trans. on CAD, vol. 12, pp. 1376 -1386, Sept. 1993.

[9] R. Trihy, R. Rohrer, ”A Switched Capacitor Circuit Simulator: AWEswit”, IEEE Journal of Solid-State Circuits, vol. 29, pp. 217 -225, March 1994.

[10] M. Lansirinne, K. Halonen, ”SSDSIM – A Very Fast and Versatile Simulator for ΣΔ-Modulators”, Proc. of ECCTD’99, pp. 1071-1074, Sept. 1999.

[11] C. M. Wolff, L. R. Carley, ”Simulation of Delta-Sigma Modulators Using Behavioral Models”, Proc. IEEE Int. Symp. Circuits and Systems, 1990, pp. 376-379.

[12] S. Brigati, F. Francesconi, M. Malcovati, D. Tonietto, A. Baschirotto, F. Maloberti, ”Modeling Sigma-Delta Modulator Non-Idealities in SIMULINK®”, Proc. IEEE Int. Symp. Circuits and Systems, 1999, pp. 384-387.

[13] SIMULINK® and MATLAB® User’s Guides, The MathWorks, Inc., 1997

[14] IEEE Standard VHDL Analog and Mixed-Signal Extensions”, IEEE Computer Society, Approved 18 March 1999, IEEE-SA Standards Board.

[15] ADVance-MS User’s Manual”, Document Number 315000(i) for Software Version v1.3_2.1, Mentor Graphics.

[16] *Analog Insydes*: www.analog-insydes.de

[17] R. Sommer, E. Hennig, M. Thole, T. Halfmann, T. Wichmann, “Analog Insydes 2 – New Features and Applications in Circuit Design”, in *Proc. SMACD2000*, Lisbon, Oct. 2000

[18] F. Schenkel, M. Pronath, H. Graeb and K. Antreich,“A Fast Method for Identifying Matching-Relevant Transistor Pairs”, Proc. CICC ’2001, May 2001

[19] CynApps, „Systems Modeling with Cynlib“, www.cynapps.com, 1999

[20] D. D. Gajski, J. Zhu, „SpecC Specification language and Methodology“, Kluwer Academic Publishers, Boston 2000

[21] Synopsys Inc., CoWare Inc., Frontier Design Inc. „SystemC User’s Guide Version 1.1“, www.systemc.org, 2000

[22] B. Zojer, R. Koban, J. Pichler, G. Paoli, „A Broadband High-Voltage SLIC for a Splitter- and Transformerless Combined ADSL-Lite/POTS Linecard“, IEEE J. Solid-State Circuits, vol. 35, pp 1976-1987, Dec. 2000

[23] S. Vernalde, P.Schaumont, I.Bolsens, „An Object Oriented Programming Approach for Hardware Design“, IEEE Computer Society Workshop on VLSI’99, Orlando, April 1999

[24] MEDEA „The MEDEA Design Automation Roadmap - Design Automation solution for Europe“: www.medea.org, V.5 pp 52-55

[25] K. Einwich, C. Clauss, G. Noessing, P. Schwarz, H. Zojer, „SystemC Extensions for Mixed-Signal System Design“, FDL01 Sept. 3-7 2001 Lyon, France

[26] C.Grimm, C.Meise, P.Oehler, K.Waldschmidt and W.Fey; „AnalogSL: A Library for modeling analog power drivers with C++“, FDL01 Sept. 3-7 2001 Lyon, France

[27] T.E. Bonnerud, B. Hernes, and T. Ytterdal, "A Mixed-Signal, Functional Level Simulation Framework Based on SystemC for System-on-a-Chip Applications," in Proc. Custom Integrated Circuits Conference (CICC), San Diego, California, pp. 541-545, May 2001.