A Systems Theoretic Approach to Behavioural Modeling and Simulation of Analog Functional Blocks

R. Rosenberger and S. A. Huss

Darmstadt University of Technology
Department of Computer Science
Alexanderstr. 10, D-64283 Darmstadt, Germany

Abstract

Analog simulation methodologies for the generation of macromodels of analog functional blocks, as reported in literature, are of limited use in practical circuit simulation due to frequent accuracy and efficiency problems. In this paper, a new approach to model the behaviour of nonlinear functional blocks is proposed. The approach is based upon the principles of systems theory. The outlined methodology supports the mapping of models from component into behavioural level. The nonlinearity of complex analog modules is reflected efficiently while the electrical signals are maintained.

1 Introduction

Executable system descriptions of analog circuits at component level, if achievable, result for practical applications in extreme simulation time and storage requirements. Therefore, more efficient methods than Spice-like simulation of structural system descriptions are required. Unlike in the digital domain [GDWL92] [Ram86], a widely accepted hierarchy of abstraction levels and transformations between different levels is still missing in analog design. A possible abstraction hierarchy is presented in fig. 1.

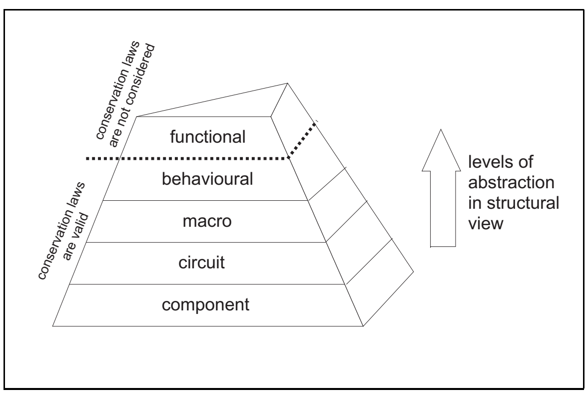

Fig. 1: Abstraction hierarchy

This hierarchy derived from [VM97] is related to the structural view of a design entity, i.e. an entity is described by means of "consists of" relations. At the component level there are transistors and passive devices available as basic elements. In turn, an entity at circuit level is described by means of opamps or comparators as components. The macro level is similar to the switch level found in abstraction hierarchies of digital systems, i.e., the functionality of a basic element is replaced by a more abstract representation including ideal components such as controlled nonlinear current or voltage sources. In contrast, models at the behavioural level are constructed by means of mathematical relationships between input and output signals. The connection pins of these models carry physical signals, which are subjected to conservation laws. In electrical systems, these are Kirchoff's current and voltage law. Finally, at the functional level, complex basic elements such as data acquisition or even modem blocks are available. At this level, the conservation laws are not valid at connection pins, i.e. signals are not physical any more. The envizaged transfer of information is indicated as in signal-flow models.

Several approaches are known from literature, which attempt an abstraction of functionality from the component and circuit level in order to improve simulation efficiency. Approaches to macromodel generation may be classified into four main groups: numerical, empirical, table look-up and computer algebra based methods. Numerical methods, such as [CL75] [CSV91] [JRS91], are frequently of limited accuracy especially for strongly nonlinear blocks. Empirical models were elaborated by many authors, e.g. [BCP74] [GH76] [MT94], but, in general, these models suffer from an unpredictable accuracy and from technology dependence. Table models and linearisation techniques achieve a high degree of accuracy and efficiency at the expense of considerable storage requirements [KY91]. An exploitation of computer algebra methods is an important alternative to numerical methods. Their applications shown so far unveil, in general, a lack of run time efficiency due to the complex task of setting up and solving large systems of algebraic equations [Bor96].

An approach to model analog blocks at functional level is presented in the following, which is aimed to cope with nonlinear behaviour of functional blocks and to consider large input and output signals in time domain operation of the block.

2 Systems theoretic background of behavioural models

Systems theory aims to provide generally applicable problem solving methods for different disciplines in science. It exploits the fact that real systems obey the same physical laws and show similar patterns of behaviour although they may be very different in their implementation.

System theoretic modeling traditionally considers two major types of general dynamical systems: differential equation and discrete time systems. Their specification formalisms are well-known, e.g. [Zei84]. New areas of application, such as simulation of mixed analog/digital circuits, require models that contain some continuous as well as some discrete parts. Systems theoretical foundations for these areas of application are given in, e.g. [Pra91]. The proposed approach is based on these foundations and uses the Differential Equation Specified System with Discrete Events (DESS&DEVS) [Pra91] as the generic concept for modeling the behaviour of analog functional blocks.

A drawback of existing approaches based upon systems theory is the underlying simulator architecture, which in fact is some kind of experimental environment without relevance to standards and industrial applications. The new modeling approach is not related to a particular simulator environment. It can easily be mapped onto any simulator offering mixed mode and behavioural hardware description language capabilities. The application examples show simulations runs with Mathematica [Wol88], which, due to the restriction of not offering real-time capabilities, shows qualitative correct results at low simulation speed rates. In addition, the algorithm was implemented in the EldoFAS simulation language [MM91] enabling run time and accuracy comparisons to simulation runs using Spice-like netlists.

The systems theoretic approach exploited in this paper promises an easy mapping onto different engineering disciplines.

2.1 Behavioural models for analog blocks

Analog functional blocks are defined as a Differential Equation Specified System with Discrete Events:

Definition 1 (System-level model)
A system-level model for an analog functional block is an 7-tupel:

$$BlockModel = \{X, Y, S, \delta_{int}, \lambda, ta, f\} \qquad (1)$$

consisting of:
Set of inputs $X = \{x_i | x_i \in I\!R, i = 1 \ldots n\}$
Set of outputs $Y = \{y_i | y_i \in I\!R, i = 1 \ldots m\}$
Internal states $S = \{s_i | s_i \in I\!R, i = 1 \ldots k\}$

The internal state variables are used to describe the functionality of the block. Currents and voltage differences are examples for internal state variables in electrical circuits. The set of outputs of the block can be calculated at any time step using the internal states and inputs.

State transition function for internal events $\delta_{int}(S) = \{m_i(S) | m_i(S) \in Methods_{Block}, i = 1 \ldots l\}$
The internal state transition function is evaluated when a scheduled event is processed: it updates the internal state variables. The internal state transition functions are *Methods* as defined in the next chapter.

Output function $\lambda(S, X) :\rightarrow Y$
The output function is a recursive function. Its parameters are input and internal states.

Time advance function $ta :\rightarrow ta_{ext} \cup ta_{int}$
The time function schedules the external and internal events.

Rate of change function $f :\rightarrow \frac{\partial s}{\partial t}$
The rate of change function calculates the continuos change of one or more internal states.

The next chapter shows, how the elements of the *BlockModel* fit into the modeling methodology. The abstract components of the BlockModel given in Eq. (1) need an instantiation for the envisaged application domain.

3 Modeling methodology

The behaviour of nonlinear functional blocks in terms of a relation between a set of input waveforms X and the resulting outputs Y is given by the implicit functions defined in the BlockModel. In the following, the basic ideas of this approach are summarized and the steps necessary to model a functional block are given. Three different cases must be considered for the generation of an analog circuit model:

1. The output function describes a discrete signal value when applying a discrete input signal to the block. This *direct dependency* is used to define the state transition function for internal states. Examples are found in DC analysis.

2. The output function describes a time dependent signal waveform when applying a discrete input

signal value to the block. This *indirect dependency* is used to define the time advance function ta_{ext} for external events and for the rate of change function f.

3. The output function describes a time dependent signal waveform when applying a continuos input waveform to the block. This is used for the recursive definition of the *output function* λ.

3.1 Direct dependencies

When considering a direct dependency of the output signal of a functional block on its input signals, one can state this relation as

$$y = f(x). \tag{2}$$

Independent variables are input currents or voltages and dependent variables are, e.g., output current or block delay values. One can generate the input-output relation by exercising circuit simulation runs for the structural description of the block at circuit level. In order to arrange for a behavioural description of the block according to Eq. (2), a description is required, which approximates the relationships for user defined accuracy values.

A so-called *method* is introduced for this purpose.

Definition 2 (Method)
The domain of the independent variable is devided into segments. Segments are defined as:

$$(x_1, x_2] = \{x | x \in I\!R, x_1 < x \leq x_2\}, \tag{3}$$

$$[x_1, x_2) = \{x | x \in I\!R, x_1 \leq x x_2\}. \tag{4}$$

Each segment is mapped onto a mathematical function such that:

$$\omega_1 : (x_1, x_2] \rightarrow f_1(x). \tag{5}$$

A method is defined as a set $\Omega = \{\omega_i : 1 \leq i \leq k\}$ of segments featuring the following properties:

Completeness:
D_f is the domain of the dependent variable.
If $x \in D_f \Rightarrow \exists i : x \in \omega_i$.

Uniqueness:
For every pair (i, j) of segments with $i \neq j$ there is the requirement of uniqueness:
$x \in \omega_i \Rightarrow x \notin \omega_j$.

C_0 -Smoothness:
For two segments $\omega_1 : (x_i, x_j] \rightarrow f_1(x)$ and $\omega_2 : (x_j, x_k] \rightarrow f_2(x) lim_{x \rightarrow x_j} f_2(x) = f_1(x_j)$ is valid. This property is important to ensure the convergence of the simulation process.

Accuracy:
For each input value x_i the corresponding output value y_i and the maximum error value ϵ yields $f(x_i) - y_i \leq \epsilon$.

Different error norms and distance metrics can be applied to calculate ϵ. Well-known error norms l_1, l_2 and l_∞ [Box78] may be used.

In previous work [Ros94a] some drawbacks of these error norms were demonstrated for particular data sets. To overcome these problems, an additional distance metric to calculate the error was implemented: the Euclidean Distance. It tolerates small phase-shifts and acts as a nonlinear low-pass filter in dependence of a scaling factor [YCJ91].

The most important mathematical task in parametrizing a method according to Def. 2 is curve-fitting [Rat83]. An example of such a direct dependency may be found in DC analysis. A method *DC* is used in this approach, it returns the value of the DC analysis at a discrete value of the input function. Figure 2 shows the method *DC* used in the application example in Chapter 4: it consists of 6 segments for a given accuracy value to be met.

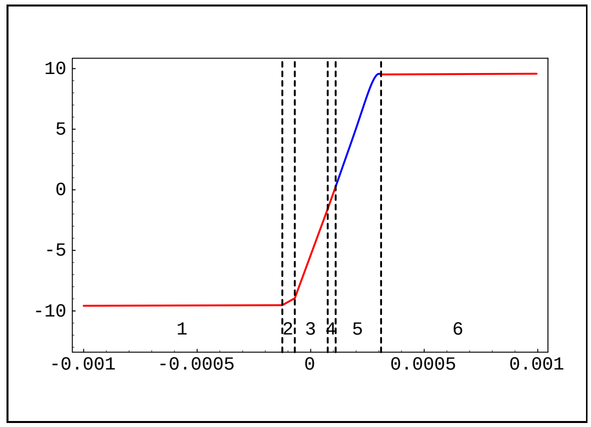

Fig. 2: Method *DC* as an approximation to the DC output signal.

3.2 Indirect dependencies

When applying a discrete input signal in time domain, the block returns a waveform, not a discrete value as for the direct dependency. In this case a set of output waveforms is generated.

These output waveforms form the data basis for the calculation of the values of functions as defined in the following. When generating the waveforms from circuit simulation, the following points must be considered:

At first, a basic circuit is needed as input for the simulator to generate the output waveforms for discrete input values. This basic circuit is usually a open-loop circuit for, e.g., opamps, with a discrete step value at the input. Although the open-loop circuit is not relevant in practical modeling, all properties sufficient to characterize the block may be obtained.

3.2.1 Time advance function for external events

When applying a step function waveform to the input of the block, a certain delay can be observed until this input produces relevant changes on the output. This time difference is the delay time of the functional block defined as:

Definition 3 (Delay time)
The input voltage switches from V_{in1} to V_{in2}. The output rises in sufficient time from V_{out1} to V_{out2}. With $t_{in50} = t|V_{in}(t) = \frac{V_{in2}-V_{in1}}{2}$ and $t_{out50} = t|V_{out}(t) = \frac{V_{out2}-V_{out1}}{2}$, the delay-time T_D is given as:

$$T_D = t_{out50} - t_{in50}. \quad (6)$$

All values necessary to calculate T_D can be obtained from the output waveform . This provides a direct relationship between V_{in}, the discrete value of the input step applied to obtain the waveform, and T_D, the particular delay time. The delay time varies with the value of V_{in} and with the rise or fall characteristic of the input signal. The methods library generates a method for this relationship as outlined before, whereas negative values of V_{in} denote a falling voltage at the input.

3.2.2 Rate of change function

Applying a (nearly ideal) step function waveform to the input of the block, there is no step waveform on the output pin. In contrast, the output signal raises with a certain slope value. This slope is described by a *slewrate*:

Definition 4 (Slewrate)
The voltage on the input pin rises from V_{in1} to V_{in2}. The output voltage rises in a sufficiently long time from V_{out1} to V_{out2}. Now the differences $V_{out10} = \frac{V_{out2}-V_{out1}}{10}$, $V_{out90} = \frac{9(V_{out2}-V_{out1})}{10}$ and time points $t_{out10} = t|V_{out}(t) = V_{out10}$, $t_{out90} = t|V_{out}(t) = V_{out90}$ are introduced. The resulting *slewrate* is defined as:

$$slewrate = \frac{V_{out90} - V_{out10}}{t_{out90} - t_{out10}}. \quad (7)$$

The relation of Eq. (7) corresponds to an approximation of the slope of the output signal. All necessary values are obtained from the results of a transient analysis of the functional block with step functions applied on the input. It is important to notice that *slewrate* is not constant for all values of the current step. Instead, there is a nonlinear dependency between the input difference and the resulting slewrate values. Therefore, a method as defined in Def. 2 are provided in order to obtain a mathematical description of this dependency. The method *slewrate* calculates the slope of the output signal depending on the size of the step value on the input of the block. This calculation is performed dynamically at every time step as detailed in Section 3.3.

Now the relation between the rate of change function f of Eq. (1) and the method *slewrate* is considered. It is claimed that

$$\frac{\Delta y}{\Delta t} := slewrate(\Delta x). \quad (8)$$

This assignment is derived from the following observations. At some discrete point in time $t^{(n-1)}$ the circuit is in a constant state, input and output pins show certain signal values $x^{(n-1)}$ and $y^{(n-1)}$, respectively. For the next time point $t^{(n)}, x^{(n)}$ and $y^{(n)}$ are to be obtained. Their differences are denoted as $\Delta x = x^{(n)} - x^{(n-1)}$ and $\Delta y = y^{(n)} - y^{(n-1)}$. The effect Δy caused by Δx is equivalent to the effect of a step waveform on the input with size Δx. This yields $\frac{\Delta y}{\Delta t^{(n)}} = slewrate(\Delta x)$, which defines the slope of a straight line that connects the points $(x^{(n-1)}, y^{(n-1)})$ and $(x^{(n)}, y^{(n)})$, respectively.

The method *slewrate* as given above models the slope of the output signal for the input increment Δx. According to Def. 3, the value returned by *slewrate* corresponds to the difference $\frac{\Delta y}{\Delta t}$ related to a discrete time step Δt, which, in turn, is related to Δx.

One key point of this approach is the fact that the transient simulation results are exclusively used to gain qualitative characteristic functions of the block, such as the slewrate, but not for quantities such as the absolute value of the output signal.

3.3 Output function

An input/output description of the block behaviour is needed to build the model. Therefore, the output waveform is denoted recursively at a discrete time point n as:

$$y^{(n)} = y^{(n-1)} + \Delta y^{(n)}. \quad (9)$$

The block model is used during a transient analysis of a arbitrary system description. So the difference quantity $\Delta y^{(n)}$ from Eq. (9) is denoted by means of the simulator integration time step $\Delta t^{(n)}$ resulting in

$$\Delta y^{(n)} = \frac{\Delta y}{\Delta t} \Delta t^{(n)}. \quad (10)$$

Taking into consideration that $\frac{\Delta y}{\Delta t}$ is the rate of change function defined in the previous section, it can be replaced by the method *slewrate*. The recursive output function is initialized using $y^{(0)}$ obtained from the DC simulation result. This yields the complete output function as

$$y^{(n)} = y^{(n-1)} + slewrate(\Delta x) * \Delta t^{(n)}, \quad (11)$$

where as

$$y^{(0)} = DC(x_0). \quad (12)$$

The simulation algorithm is based upon this recursive function. The discrete time points $t^{(i)}$ are determined by means of the two-level event queue summarized in section 3.4. At each discrete time step $\Delta t_{(i)}$ the resulting change of the output signal $\Delta y^{(i)}$ is obtained by multiplying $\Delta t^{(i)}$ and the slewrate of the step response function related to $V_{in}^{(i)}$. The output signal is constructed as depicted in Fig. 3.

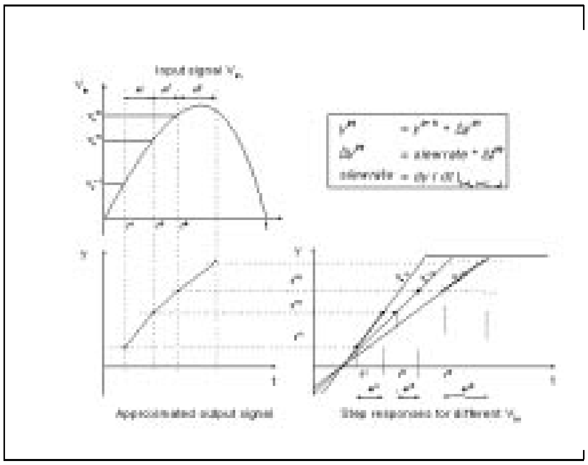

Fig. 3: Recursive construction of the output signal.

3.4 Simulation algorithm

Time-domain analysis is continuous in the sense that the simulator chooses the size of a time step to be as small as necessary for accuracy. The model schedules exact times for variables to change their values, therefor discrete time simulation is used. Since the simulator has to check effects of variable changes only at scheduled times, a significant speed-up of simulation time can be observed.

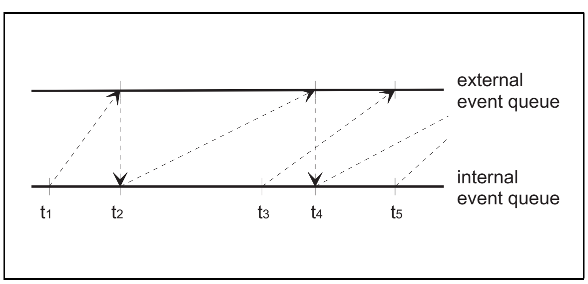

Fig. 4: Two-level event queue used in the simulation algorithm.

A two-level event-queue as depicted in Fig. 4 is proposed. The lower queue consists of the internal events ta_{int}. These are the events scheduled by the simulator, whereas the upper queue contains external discrete time events scheduled by the model. An external event is placed in the event queue each time any event is processed. The time values for external events are established by the state transition function of the delay time according to Eq. (1). Processing an event always means executing the state transition, the rate of change and the output function respectively, i.e. the actual set of inputs is used to update both the internal states and the set of outputs. The following pseudocode summarizes the handling of internal events. The functions called are methods as defined in section 3.1.

```
(* Processing internal events *)
when (event_on(time)) {
      (* State transition functions
      for internal states *)
      VD = difference_voltage(V_in);
      TD = delay_time[ VD ];
      (* Output function *)
      V_out= V_out^old +Δ * sr;
      (* New rate of change function *)
      sr^new = slewrate(V_D);
      (* Scheduling an external event *)
      schedule_event(time + TD, sr, sr^new);
      }
```

It is important to note that, although the methods *slewrate* and $delay - time$ are modeled using piecewise defined functions, the whole system is not a kind of piecewise linearisation because the functions f_i of Ω as given in Def. 2 model nonlinear dependencies. Furthermore, the algorithm outlined above is a direct implementation of the system-level model from Def. 1. The output signal is then constructed recursively as shown in Fig. 3.

The model requires continuous time simulation in conjunction with the ability to process events. These requirements are known from the simulation of mixed analog/digital systems, where the time-continuous analog part interacts with the discrete event based digital part. The upcoming IEEE standard denoted VHDL-AMS [Pag97] is an enhancement of VHDL 1076-1993. In VHDL-AMS, extensions to support the description and simulation of mixed continuous/discrete systems will be included. These mixed-mode modeling capabilities enable an easy mapping of the algorithm outlined above. The model can be easily adapted to a different simulation environment by changing some syntax elements. In fact, the proposed simulation algorithm has been successfully implemented in Eldo-FAS, Mathematica and Saber-MAST.

3.5 Instantiating the generic methods

An analog simulator, which works with the structural descriptions of functional blocks, is needed to parametrize the methods. Extensive mathematical functionality is needed to summarize the simulation results. The process of parametrizing methods from the simulation results and of collecting them in a table format is denoted as Characterization Plan. The Clang [SH91] characterization language is extended with a link to Mathematica [Ros94b] to process the characterization plan. This enables users to do arbitrary - even symbolic - calculations and to visualize the results.

Requirements to a visual simulation environment, such as interactive description of simulation and characterization tasks, transparent supply of the characterized data and high flexibility with respect to evaluation are realized in the visual simulation environment ViCE [MG96]. Through an automatic parallelization algorithm ViCE offers an efficient execution of large Characterization Plans, which otherwise would require run-times of several hours on a single workstations. Fig. 5 shows the Characterization Plan used to generate the method *slewrate* exploited in the application examples.

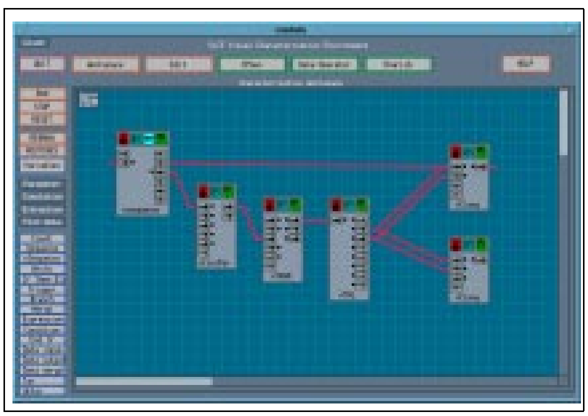

Fig. 5: Characterization plan yielding the method *slewrate*.

4 Application examples

The proposed modeling methodology is demonstrated in the following for an operational amplifier and a filter. Model instances for applications in time domain were produced as proposed in Chapter 3 using the methods library [Ros94a]. It takes just a few CPU seconds to instantiate a generic model. The method for the DC behaviour, as depicted in Fig. 2, is an example for a state transition function for internal states according to Def. 1. Additional methods include modeling of the currents on the input and output pins of the opamp.

The testbench for the functional blocks is a voltage follower as usually found in large signal analysis [JRS91]. Note that characteristic functions of the opamp established for the open-loop circuit apply directly for feedback operation. Simulation results were produced by means of Eldo-FAS Rel. 4.2.2.

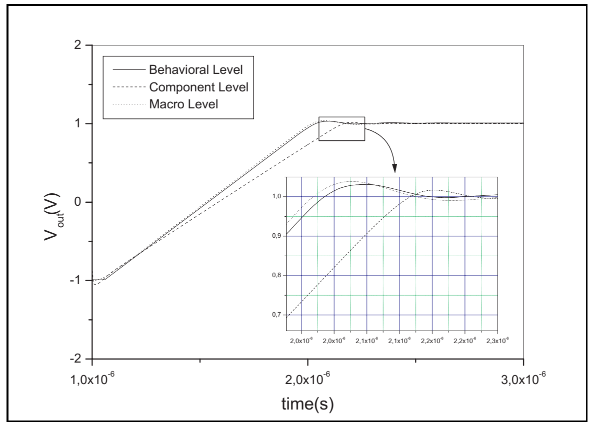

Fig. 6: Comparison of output waveforms.

An opamp available as a standard component is used as an example. The manufacturer of this bipolar circuit denoted as MOPA1 [Ray] provides, in addition to datasheets, two simulation models: a structural description at component-level and a macromodel derived from [BCP74]. Thus, it is possible to assess the modeling approach directly by comparing it to commercially available simulation models.

In the first evaluation process the complex time domain properties of the opamp, such as settling time and maximum overshoot [AH87], are examinated. The methods were generated at an accuracy level of 2%. Fig. 6 depicts the step response of the three simulation models of MOPA1.

Table 1: Comparison of simulation statistics for MOPA1 and biquadratic filter block.

Model	Block	Nodes	Mem. [kB]	Time [s]	Speed-up
Comp.-level	MOPA1	83	930	55	1
	Filter	327	1324	123	1
Macro-level	MOPA1	15	877	18	3
	Filter	55	1098	27	4
Behav.-level	MOPA1	3	596	6	9
	Filter	9	915	12	10

Output voltage waveform, settling time and maximum overshoot of the model derived as proposed in this paper are very close to the results of the component-level model at a speed-up value of about 10 and at a reduced storage consumption as summarized in Table 1. The envisaged accuracy level is met for the specified time domain properties, as depicted in the zoomed part of Fig. 6. In contrast, the macromodel produces results of questionable accuracy.

The biquadratic filter circuit depicted in Fig. 7 is used as a second example. A damped sine wave is used as input signal, as shown in the inset graph of Fig. 8. In this filter circuit, the opamps are modeled at component, macro and behavioural level. Fig. 8 shows the resulting output waveforms; the nonlinear effects caused by the input signal are maintained in all models. Table 1 summarizes the results.

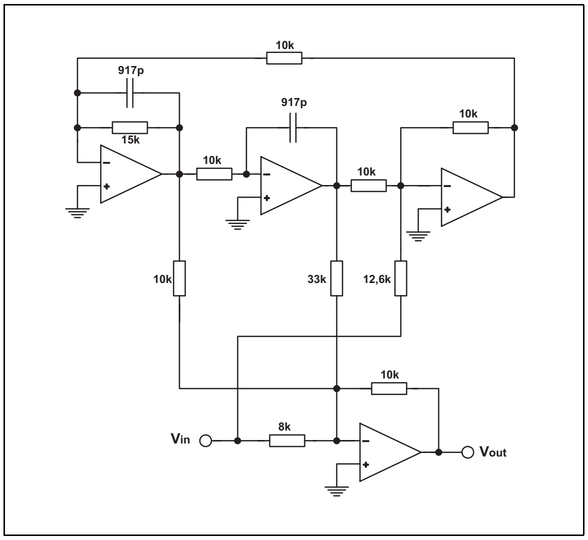

Fig. 7: Biquadratic filter block.

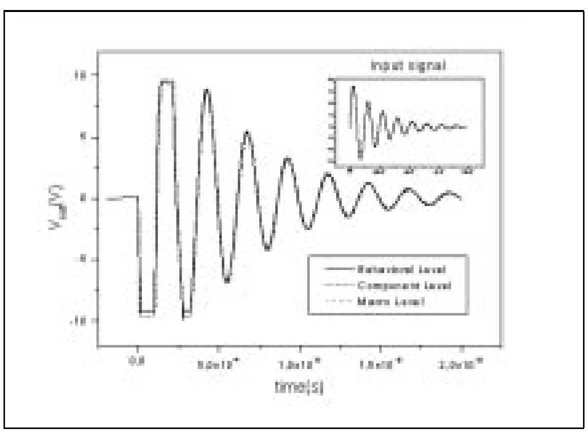

Fig. 8: Results from three different simulation models.

Speed-up values of more than one order of magnitude were observed depending on selections of accuracy levels, of the size of the property sets, and on different input waveforms.

Additional application examples can be found in [HH95].

5 Conclusions

The proposed approach to behavioural modeling of analog functional blocks at results in a considerable increase of simulation speed while preserving to a large extent the shape of signal waveforms. Its main advantage however is the reusability of the models in different circuits with arbitrary load conditions. The interdisciplinary systems theory background forms a basis for an easy mapping to both different disciplines and to mixed systems.

The actual implementation needs further improvements. An integration in a visual programming environment should free a human modeler from programming details when generating the models. The mixed-mode capabilities of VHDL-AMS will allow an easy adoption of the approach presented onto the forthcoming industrial standard. Applications from different engineering disciplines must be worked out in order to demonstrate that the approach is interdisciplinarily valid.

References

[AH87] P. E. Allen and D. R. Holberg. *CMOS Analog Design*. Saunders College Publishing, 1987.

[BCP74] R. Boyle, M. Cohn, and O. Pederson. Macromodeling of Integrated Circuit Operational Amplifiers. *IEEE Solid-State Circ.*, SC-9(6):353–64, 1974.

[Bor96] C. Borchers. Automatische Generierung symbolischer Verhaltensmodelle für nichtlineare Analogschaltungen. *4. GMM/ITG Diskussionstagung Entwicklung von Analogschaltungen mit CAE-Methoden*, pages 93–98, 1996.

[Box78] G. E. Box. *Statistics for Experimenters*. John Wiley and Sons Inc, 1978.

[CL75] L. O. Chua and P. M. Lin. *Computer-Aided Analysis of Electronic Circuits: Algorithms and Computational Techniques*. Prentice Hall, 1975.

[CSV91] G. Casinovi and A. Sangiovanni-Vincentelli. A Macromodeling Algorithm for Analog Circuits. *IEEE Trans. on CAD*, 10(2):150–60, 1991.

[GDWL92] D. Gajski, N. Dutt, A. Wu, and S. Lin. *High-level synthesis: Introduction to Chip and System Design*. Kluwer Academic Publishers, 1992.

[GH76] I. E. Getreu and A. D. Hadiwidjaja. An Integrated-Circuit Comparator Macromodel. *IEEE Solid-State Circ.*, SC-11(6):826–33, 1976.

[HAM95] M. Fiegenbaum and H. A. Mantooth. *Modeling with an Analog Hardware Description Language*. Analog Circuits and Signal Processing. Kluwer Academic Publishers, 1995.

[HH95] S. A. Huss and H. Hamad. Adjustable accuracy behavioral models for system-level analog simulation. *Proc. of the 4th int. conference on VLSI and CAD*, pages 435–438, 1995.

[JRS91] Y.-C. Ju, V. B. Rao, and R. A. Saleh. Consistency Checking and Optimization of Macromodels. *IEEE Trans. on CAD*, 10(8):957–67, 1991.

[KY91] P. Allen and K. Yoon. An Adjustable Accuracy Model for VLSI Analog Circuits Using Lookup Tables. *Analog Integrated Circuits and Signal Processing*, 1:45–63, 1991.

[MG96] S. A. Huss and M. Goedecke. A Visual Simulation Environment for efficient characterization of analog circuit behavior. *IEEE Proc. of the 39th Midwest Symp. on Circuits and Systems*, pages 339–342, 1996.

[MM91] A.Maynard and M. Maynard. *ELDO-FAS Dynamic System Modeling*. Anacad Computer Systems, 1991.

[MT94] H. T. Mammen and W. Thronicke. Object-oriented Macromodelling of Analog Devices. In *Proc. of the Internat. Conf. on Concurrent Engineering and Electronic Design Automation*, pages 331–36, 1994.

[Pag97] IEEE DASC 1076.1 Home Page. Analog and mixed-signal extensions to vhdl. http://vhdl.org/vi/analog/index.html, 1997.

[Pra91] H. Praehofer. System Theoretic Foundations of Combined Discrete-Continous System Simulation. Phd thesis, Johannes Kepler Universitaet Linz, Department of Systems Theory, 1991.

[Ram86] F. J. Rammig. Mixed-level Modeling and Simulation of VLSI Systems. In E. Hörbst, editor, *Logic Design and Simulation*. North-Holland, 1986.

[Rat83] D. A. Ratkowsky. *Nonlinear Regression Modeling*. Marcel Dekker Inc. New York, 1983.

[Ray] RLA Linear Macrocell Array Breadboarding Kit. Raytheon Company - Semiconductor Division.

[Ros94a] R. Rosenberger. Entwurf und Implementierung einer Methodenbibliothek zur Verhaltensmodellierung analoger Schaltungen. Diplomarbeit, Technische Hochschule Darmstadt, FB Informatik, Fachgebiet Integrierte Schaltungen und Systeme, 1994.

[Ros94b] R. Rosenberger. Integration von Mathematica in die Charakterisierungssprache CLANG mittels MathLink. Studienarbeit FB Informatik TH Darmstadt, 1994.

[SH91] G. Tränkle, S. Huss, and M. Gerbershagen. Automatic Performance Characterization of Analog Functional Blocks. *Analog Integrated Circuits and Signal Processing*, 1:277–286, 1991.

[VM97] F. Pellandini, V. Moser, and H. P. Amann. Behavioral Modeling of analogue Systems with absynth. In O. Leria J. Rouillard A. Vachoux, J.-M. Berge, editor, *Analog and Mixed-Signal Hardware Description Languages*. Kluwer Academic Press, 1997.

[Wol88] S. Wolfram. *Mathematica - A System for Doing Mathematics by Computer*. Addison-Wesley Publishing Company, 1988.

[YCJ91] R. Saleh, Y.-C. Ju, and V. Rao. Consistency Checking and Optimization of Macromodels. *IEEE Transactions on Computer-Aided Design*, 10(8):957–967, August 1991.

[Zei84] B. P. Zeigler. *Multifacetted Modelling and Discrete Event Simulation*. Academic Press, 1984.