A symbolic frequency-domain simulation program for an analogue design environment

Rocío De Jesús Ventura, Lizbeth Ruz Armas, Jonathan Valencia Santa Rosa,
Luis Hernández-Martínez, Arturo Sarmiento-Reyes
National Institute for Astrophysics, Optics and Electronics
Electronics Department, CAD Group
P.O. Box 51, 72000, Puebla, Pue., Mexico

***Abstract*— A frequency-domain simulation program with the capability of obtaining fully symbolic transfer and network functions is presented. Such a simulation program is devised to work within an environment for analogue design. The program obeys a top-to-bottom design path while keeping a bottom-to-top verification path, which results in a hierarchical management of the device modelling libraries of the simulator. As far as possible, the program sticks with the well-known input grammar of the SPICE simulation program. The paper describes the general needs the simulator must fulfil as well as the characteristics regarding the input language, the output facilities and a snapshots of running example.**

I. Introduction

In a design methodology based on the alternative of structured design, the design procedure follows a path from top to down, while the verification of the on-going design is done in a down-to-top path. Therefore, the verification tool (the simulation program) needs to fulfil different issues regarding the hierarchy of the design. It clearly results that a general-purpouse simulation environment can not be useful under these scope because the such a solid and massive tool will slow down the design times and the overhead of the hardware will increase. Instead, a dedicated verification tool is used, with the capability of being easy to adapt to every phase of design and to the required hierarchy.

Because structured design [1], [2] focusses on the specifications in a hierarchical way, the needs established for simulation are also guided for the hierarchy of the design. For instance, when the verification of the behaviour of the circuit with respect to noise is accomplished, then the models used within the simulation program need to be linear and noisy exclusively. That is to say, the use of the models and its selection obey the hierarchy and in some cases a kind of orthogonalisation. Figure 1 depicts these concepts in a graphical way.

Not only do the simulation needs depend on the current level of hierarchy in the design, but also on the type a specification is being tackled. In this form, the domain of simulation is driven by both aspects. As a result, device modelling occurs also in a hierarchical way. It is worthy to mention that this is a consequence of the fact that models for analysis are quite

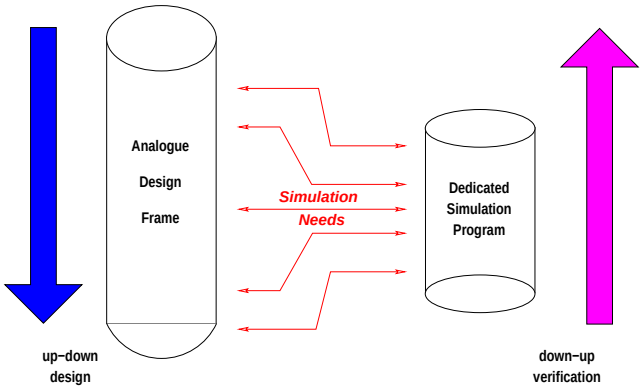

Fig. 1. A verification tool in combination with an analogue design environment.

different (both in nature and complexity) to models used for synthesis. In plain words, if the on-going design does not fulfil a given specification with a simple device model, neither will do with a more complex model. This is sketched in Figure 2.

The complexity of the models used in a verification tool for an automated analogue design environment is shown in Figure 3.

II. Frequency-domain verification

Several issues of analogue circuit design are better understood and verified if the verification of the on-going design is achieved in the frequency-domain. Examples of them are bandwidth, open-loop gain, stability, pole-loci, just for mentioning a few. An example of a more specific need is to determine the open-loop gain-pole product during the design of negative feedback amplifiers [3].

Frequency-domain verification results more useful to the designer when it is achieved in the form of symbolic simulation, specially for analogue design automation environments. Symbolic analysis of electric circuits gives the designer more insight on the circuit behaviour than ordinary numerical analysis [4], [5], leading to a better understanding of the functioning of the circuit.

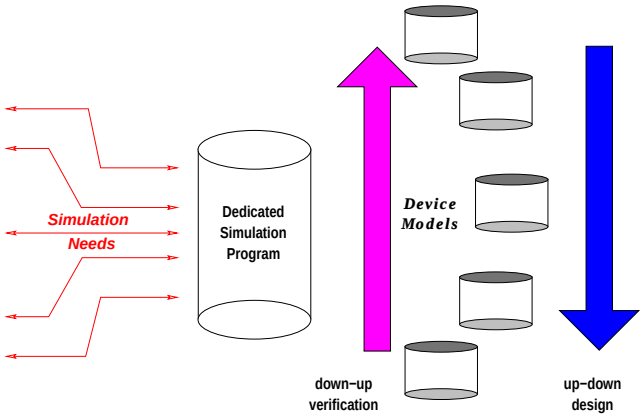

Fig. 2. Simulation needs within an analogue design environment.

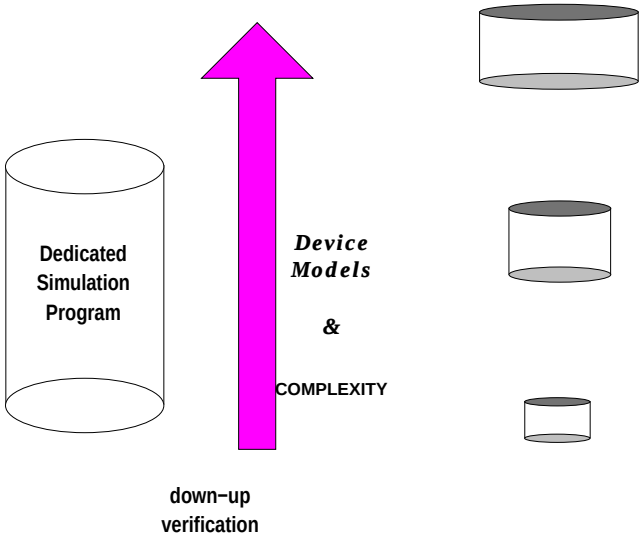

Fig. 3. Complexity of the device models for a simulator within an analogue design environment.

On one side, a numerical simulation gives as main result a set of numbers, tables or plots from which the designer can achieve further modifications to the circuit under design in order to fulfil the specifications. Notwithstanding the high accuracy of the results of numerical simulations, they are highly dependent on the parameters of the numerical methods inside the analysis engine used to solve the circuit equations. Examples of such parameters are those used in Newton-Raphson limiting schemas, the integration step size, the parameters of tolerance and maximum allowed error, etc.

On the other side, a symbolic simulation gives as result a set of symbolic (or semi-symbolic) expressions. By analysing these expressions, the designer can determine which parameters affect in larger degree a circuit function. Because the vast majority of the expressions emanating from symbolic analysis tend to be extremely complex, specially when analysing VLSI circuits, the scope of symbolic analysis has been reduced to small-sized circuits, or within design automation environments at top levels of design [6]. Besides, symbolic analyses allow the user to achieve further parameter substitution in order to carry out a numerical or semi-symbolic analysis or evaluations for display of variables in plots.

The difference between numerical and symbolic simulators is sketched in Figure 4.

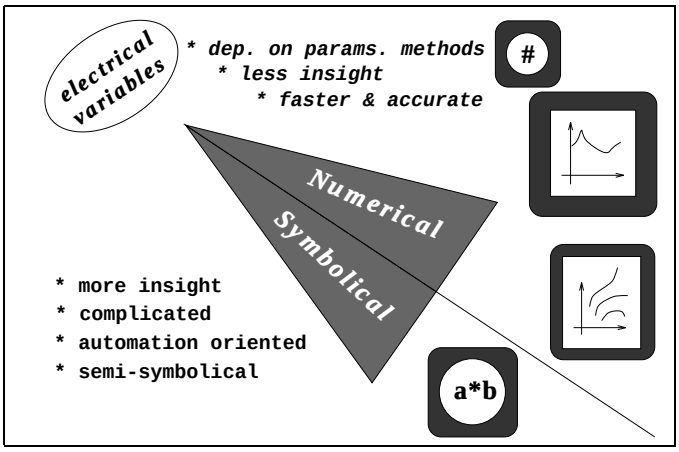

Fig. 4. Symbolical simulation vs numerical simulation.

MAPLE is a computer algebra system (CAS) that has the qualifications to be tool of selection for handling many computational issues within Electrical Engineering. Because the wide possibilities offered by MAPLE for handling symbolic expressions [7], [8], we have selected it as the programming environment for implementing the frequency-domain simulator.

III. Program Structure

The first step of circuit analysis consists in formulating the circuit equations in a suitable form by resorting, on one side, to the Kirchhoff laws and on the other side, to the relationships of the elements involving their own electrical variables. In the frequency-domain the equilibrium equation is established by using the well-known modified nodal analysis method.

The second step consists in solving the resulting equations by appropriate methods. In symbolic-oriented analysis, these methods are essentially analytical in nature. However, some of the aims remain being of numerical nature, therefore some numerical methods have to be implemented afterwards.

In summary, a symbolic-based package analyses a circuit by *(i)* translating the given circuit into a suitable representation, *(ii)* choosing and developing the analysis to be performed and *(iii)* giving the result in the form of an analytical or semi-numerical expression. This description is the basic structure of a simulation program as shown in figure 5.

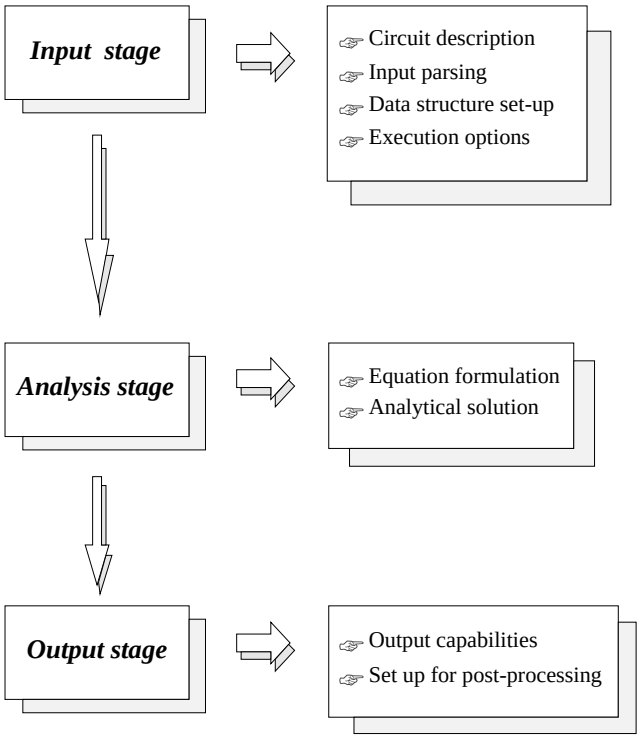

Fig. 5. Structure of a symbolic simulator.

The result of this paper is constituted by a dedicated AC simulation program that can be used in an automated analogue design environment. The input stage of the program includes a parser that is able to handle an input grammar that, as far as possible, resembles the grammar of of SPICE. In fact, the grammatical rules of our package constitute a super-set of the SPICE rules; which allows the user to enter any kind of branch relationship. This clearly results in more flexibility in the definition of the circuit components, specially those being nonlinear.

Besides, the set of elements accepted by the simulator includes singular elements used as basic cells in structured analogue design [1], such as nullors. Within the structured design methodology, the nullor constitutes the active (ideal) block of the amplifier. The nullor is a two-port composed by two elements: **the nullator** connected at the input port and **the norator** connected at the output port. The transmission matrix of the nullor is given as [9], [10]:

$$K = \begin{bmatrix} \frac{1}{\mu} & \frac{1}{\gamma} \\ \\ \frac{1}{\zeta} & \frac{1}{\beta} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad (1)$$

It clearly results that the nullor possesses infinite gains for all four transfer relationships, voltage (μ), current (α), trans-conductance (γ) and trans-impedance (ζ).

Lack of space does not allow us to expound all involved grammar definitions, but some of them are glossed in the following[1]:

```
comment::= '#' text
text::= <any string>
endfile::= 'end' | 'END' <end of file>

elemdef::= elemname'␣'nodelist'␣'value
         | elemname'␣'nodelist'␣'paramlist
         | elemname'␣'nodelist'␣'paramlist'␣'initval
         | elemname'␣'nodelist'␣'modelname
         | elemname'␣'nodelist'␣'subcircuit

elemname::= elemINIT <any SPICE identifier>
nodelist::=   node
            | nodelist'␣'node
value::= <any decimal valid in MAPLE>
paramlist::=   param
             | paramlist'␣'param

modelname::=   initmodel
             | onemodel
             | endmodel

node::= <any MAPLE integer or string>
param::= paramID '=' paramDEF

initval::= IndepEV '=' value
IndepEV::=   'i'
           | 'u'

paramID::= <any MAPLE variable>
paraDEF::=   value
           | EVfun
EVfun::=   ufunction   <any voltage function>
         | ifunction   <any current function>
ufunction::= <any MAPLE expression in u>
ifunction::= <any MAPLE expression in i>
```

IV. Output facilities

As stated above, the equilibrium equation of the circuit is established by using the modified nodal formulation. Hereafter, the simulator uses the MAPLE capabilities on its full extent in order to carry out the symbolic evaluations as well as to display the output of the small-signal analysis in the form of Bode and Nyquist plots.

Some of the features of the program are:

Symbolic transfer function
: It calculates the symbolic transfer function as the ratio of two electrical variables of the circuit. In this case, parameter values of the components are not used, instead their names are used as symbolic values in combination with the complex variable s.

Symbolic network functions
: It calculates the network function between any expressions involving electrical variables of the circuit. Here is where the symbolic evaluation engine of

[1]The symbol '␣' is used to denote a blank space.

MAPLE is used to substitute and reduce any complicated function. For instance, sensitivity calculation can be done straightforward.

Numeric transfer function
: It calculates the numeric transfer function as the ratio of two electrical variables of the circuit. In this case, all parameter values of the components are used, and the resulting transfer function is only a function of the complex variable s.

Bode plots
: The magnitude (both absolute and dB) and phase are plotted.

Nyquist plots
: The Nyquist plots are displayed.

Multi-plots
: Plots for more than one transfer function can be displayed, which allows the user to compare responses of the same circuit under different parameter values or even from different circuits.

$\boldsymbol{K}$ matrix calculation
: This feature has particular application when evaluating the amplifying characteristics of a circuit. When designing negative feedback amplifiers, the calculation of the $\boldsymbol{K}$ matrix is an evaluation that must be done at each step of design. All functions above can also be applied to the single elements of the matrix.

Figure 6 shows a portion of the snapshot of the CAD tool running under MAPLE. In the final presentation more details and snapshots will be given.

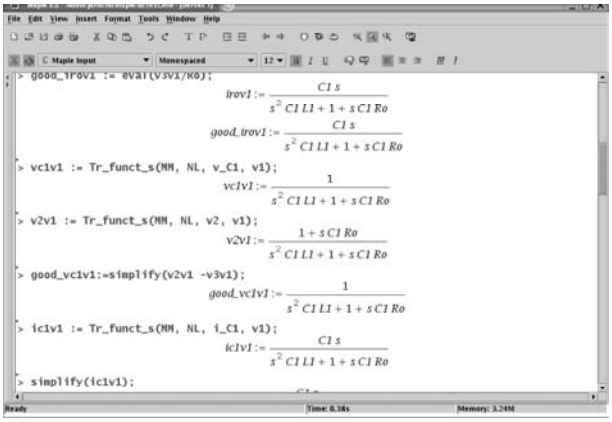

Fig. 6. Complexity of the device models for a simulator within an analogue design environment.

V. Conclusions

An application of symbolic simulation for automated analogue design environments has been fully developed under MAPLE. The simulation package is used as the verification tool within the structured design of amplifiers. The AC response, symbolic transfer function evaluation, numeric transfer function and plot facilities are among the main features of the simulator. The CAD tool has been developed under the CAS environment of MAPLE and takes advantages of the natural features for symbolic manipulation that MAPLE offers. The input file is specified in such a way, that remains a SPICE standard as much as possible, with the aim of allowing the users to use the same file for numerical and symbolic analysis. The modular programming allows the user to add new routines to the existing packages or to include new simulation variants.

Acknowledgment

This work has been supported by a CONACyT México research project under grant 42588-Y.

References

[1] G.L.E. Monna M.H.L. Kowenhoven C.J.M. Verhoeven, A. van Staveren and E. Yildiz, *Structured Electronic Design: Negative-feedback amplifiers*, Kluwer Academic Publishers, 2003.

[2] J. Stoffels, *Automation in High-performance Negative Feedback Amplifier Design*, Ph.D. thesis, Delft University of Technology, 1988.

[3] Arturo Sarmiento-Reyes y Luis Hernandez-Martinez, "Maple-based verification tool for structured analogue circuit design," *Proceedings of IASTED'2003*, July 2003.

[4] Pen-Min Lin, *Symbolic Network Analysis*, Elsevier, 1991.

[5] A. F. Schwarz, *Computer-Aided Design of Microelectronic circuits and Systems: Volume 1*, Academic Press, 1987, ISBN 0-12-632431-X.

[6] G. Gielen and W. Sansen, *Symbolic Analysis for Automated Design of Analog Integrated Circuits*, Kluwer, 1991.

[7] Bruce W. Char, Keith O. Geddes, and Gaston H. Gonnet, *Maple V language reference manual*, Springer-Verlag, New York, 1991, ISBN/ISSN: 0-387-97622-1.

[8] Andre Heck, *Introduction to Maple*, Springer-Verlag, New York, 1993, ISBN/ISSN: 0-387-97622-0.

[9] H.J. Carlin, "Singular network elements," *IEEE Trans. Circuit Theory*, vol. 11, pp. 67–72, March 1964.

[10] G.S. Moschytz, "Linear integrated networks: Fundamentals," *Bell Laboratory Series*, 1974.