AN ADVANCED SYMBOLIC ANALYZER FOR THE AUTOMATIC GENERATION OF ANALOG CIRCUIT DESIGN EQUATIONS.

F. V. Fernández, A. Rodríguez-Vázquez and J. L. Huertas

Dept. of Design of Analog Circuits, Centro Nacional de Microelectrónica, Sevilla, SPAIN

Abstract

A tool for symbolic analysis of analog integrated circuits is presented featuring accurate simplification, pole/zero extraction, and tools for parametric AC circuit characterization. The program uses signal flowgraph methods and has been written in C for portability.

Introduction

Analog integrated circuits are typically designed by following a semi-empirical approach where design parameters are iteratively updated and evaluated via an *electrical numerical simulator* until a point of the design parameter space is found fulfilling the target specifications. For efficiency, however, the updating of the design parameters must strongly rely on *knowledge* about the circuit operation, which is not conveniently provided by electrical numerical simulators themselves. For that reason, resorting to small signal models and *hand analysis* are common tools for analog integrated circuit design.

Unfortunately, hand analysis becomes extremely laborious and error-prone even for elementary building blocks containing a few transistors, forcing the designer to use oversimplified models. This motivates interest in the development of symbolic analysis tools [1], [2]. Starting from a set of small signal model primitives, one such tool automatically analyzes arbitrary interconnections of the primitives and provides outputs in the form of symbolic system functions. Recently reported analog integrated circuit symbolic analyzers [3], [4] exhibit two drawbacks: 1) They just provide symbolic expressions for numerators and denominators of circuit system functions, not giving any additional information. Since for most practical circuits the associated system functions contain typically a huge number of terms, there is still a lot of work to be done by the user for interpretation of results and, hence, for exploitation of the tool capabilities. 2) They use a simplification technique which may result in large pole/zero displacements and, hence, in important modeling errors. In this paper a tool, called ASAP, for automatic AC modeling of analog integrated circuits is presented which overcomes above drawbacks.

ASAP inputs and outputs

Fig.1 is a conceptual block diagram showing the main operation flow of ASAP. The input file includes the following information:

a) *Input netlist* in SPICE format.
b) *Level of modeling* for the semiconductor devices and for the opamps. Model schematics are taken from a library. For illustration, Fig.2 shows the four levels considered for MOS transistors.
c) *Relative parameter size* for formula simplifications.
d) Requested analyses and error margin for formula simplification.
e) *Matched devices information.* Mismatches can explicitly appear as symbols.

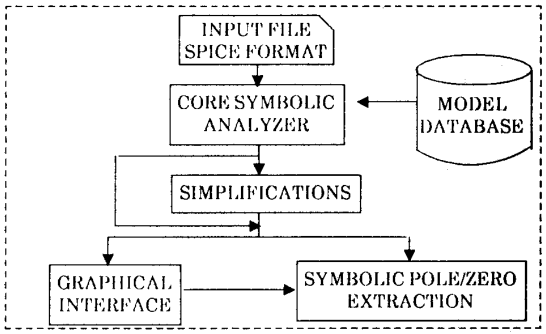

Figure 1: Conceptual diagram of ASAP.

Outputs provided by the program are in the form of *system functions* [5],

$$H(s,\mathbf{x}) = \frac{\sum_{i=0}^{N} s^i f_i(\mathbf{x})}{\sum_{j=0}^{M} s^j g_j(\mathbf{x})} \qquad (1)$$

where $\mathbf{x}^T = \{x_1, x_2, \ldots x_Q\}$ is the vector of circuit symbolic parameters (i.e., g_m, g_{ds}, g_{mb}, etc) and $f_i(\mathbf{x})$ and $g_j(\mathbf{x})$ are polynomial functions in the symbols.

Transfer characteristics (voltage and current gains, transimpedances and transadmittances) as well as *driving point immitances* can be asked for. Formulas for other AC specifications like *CMRR*,

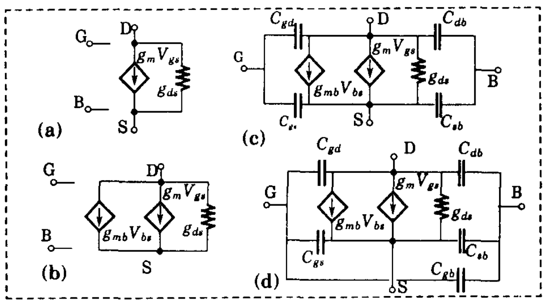

Figure 2: MOS transistor levels of modeling.

PSRR, etc, can also be calculated. ASAP is able to provide *complete as well as simplified expressions.* Besides, approximate symbolic expressions for poles and zeros can be calculated in an interactive way. To this purpose, and starting from numerical information about pole/zero positions the program uses heuristic rules to suggest possible approximations that are performed in case they are accepted by the user.

ASAP architecture

ASAP uses signal flow graph symbolic analysis methods. Fig.3 shows the internal flow of operations in ASAP. The circuit description is expanded by ASAP using models from a library. Then, this new circuit topology is reduced by substituting each group of shunted elements with admittance description by one corresponding element. These groupings allow an important reduction in CPU time. Besides, the program incorporates the feature of storing the information associated with each group of elements. Since this information is used for a final symbol expansion, where all the original symbols are recovered, the process of grouping is completely transparent to the user. Furthermore, this expansion avoids some imprecisions when the simplification of expressions is performed with the shunted elements. The resulting set of equations is solved by the use of Mason's determinant, in which the cancellations of terms have been greatly reduced.

To increase efficiency several innovations concerning the signal flow graph method have been incorporated into ASAP:

1) *Codification of symbols*: Two complementary systems (a bit-to-bit codification system and a prime number one) are used for that. The optimum codification system is then automatically chosen depending upon the characteristics of the symbols to be manipulated.

2) *Selection of the tree:* The efficiency of the symbolic analysis program is very sensitive to the chosen tree (e.g. variations of two orders of magnitude in CPU time and memory requirements have been measured for different trees associated to a Miller opamp). No algorithmic method exists for the selection of an optimum tree. In ASAP, a heuristic module has been developed which allows the selection of a "good" tree, not far from the optimum one.

3) *Loop enumeration*: The signal flow graph method requires the calculation of all the loops of any

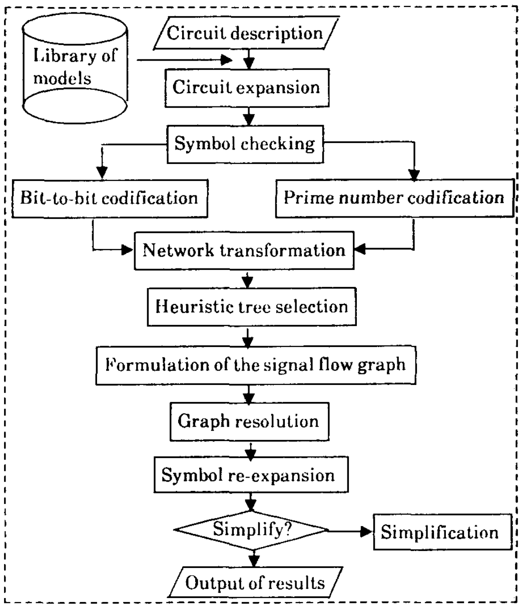

Figure 3: Architectural diagram of ASAP.

order. To reduce the need of memory resources, an algorithm has been developed which calculates the loops of *nth* order by using only those of *(n-1)th* order. The speed of the proposed algorithm is similar to others reported before but memory requirements have been significantly reduced as only one order loops must be stored at any time instance.

Formula simplification in ASAP

Symbolic analysis of common analog blocks, even those of low complexity, can produce complex symbolic expressions containing several thousand terms. Hence, simplification is mandatory.

In ASAP, simplifications are automatically performed by deleting the least significant terms in the coefficient polynomials $f_i(\mathrm{x})$ and $g_j(\mathrm{x})$ ($1 \leq i \leq N$, $1 \leq j \leq M$) in (1). As it was stated above, information about the relative size of the circuit parameters must be provided for simplifications to be carried out. Besides, the requested accuracy margin for dropping out non-significant terms must be also given as an input. Assume an arbitrary symbolic polynomial,

$$h_k(\mathrm{x}) = \sum_{l=1}^{L} h_{kl}(\mathrm{x}) \tag{2}$$

where $h_k(\mathrm{x})$ represents either $f_i(\mathrm{x})$ or $g_j(\mathrm{x})$ in (1) and where the different terms $h_{kl}(\mathrm{x})$ are products of symbols. The *basic technique* used in ASAP for deleting the *P* least significant terms of (2) is according to the following formula,

$$\frac{\sum_{l=1}^{P} \left| h_{kl}(\mathrm{x}_o) \right|}{\sum_{l=1}^{L} \left| h_{kl}(\mathrm{x}_o) \right|} < \varepsilon \tag{3}$$

where ε is the accuracy margin specified by the user.

Observe in (3) that a particular point $\mathbf{x}_0$ of the design parameter space has been considered for evaluation. Parameter values yielding this particular point can either be provided by the user or left as defaults. In the more general case a finite set of evaluation points is required as it is computationally impossible to go over the full range of variation of the different symbolic parameters. On the other hand, and as a difference to what is done in [4], summations in both the numerator and the denominator of (3) are made using the modules of the different terms of $h_k(\mathbf{x})$. For the numerator, it is so done to avoid problems in case there exist mutually canceling terms of significant magnitude, a common fact in calculation of second order characteristics. Concerning the denominator, using modules avoids excessively conservative simplifications and disparities among the margin errors applied to the coefficients of the different powers of s in (1).

However, although the basic simplification criterion performs correctly for many cases, under some circumstances its use may lead to important errors in pole and zero locations. To illustrate that, let us consider the Miller opamp of Fig.4. Assume transistors in the differential pair are matched, and consider the MOS transistor model of Fig.2(c) to evaluate the opamp differential voltage gain. After simplification with a specified error margin of 20% Table 1 results, giving the numerator and the denominator polynomials of the voltage gain.

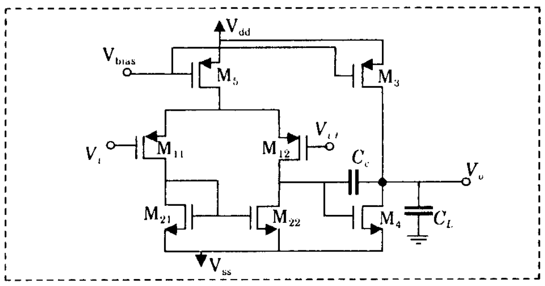

Figure 4: Miller operational amplifier.

Numerator =	Denominator =
$+2g_{m1}^2g_{m2}g_{m4}$ $+s(2C_{gs1}g_{m1}g_{m2}g_{m4}$ $-2g_{m1}^2g_{m2}C_c)$ $+s^2(C_{gs1}g_{m1}g_{m4}$ $(C_{db1}+2C_{gs2})$ $-2C_cC_{gs1}g_{m1}g_{m2}$ $-2C_cC_{gs2}g_{m1}^2)$ $-s^3(C_cC_{gs1}g_{m1}$ $(C_{db1}+C_{db2}+2C_{gs2})$ $+s^4C_cC_{gd1}C_{gs1}$ $(C_{db1}+2C_{gs2})$	$+2g_{m1}g_{m2}(g_{ds1}+g_{ds2})(g_{ds3}+g_{ds4})$ $+s2C_cg_{m1}g_{m2}g_{m4}$ $+s^2(2C_Lg_{m1}g_{m2}(C_c+C_{gs4})$ $+2C_cC_{gs1}g_{m2}g_{m4}+4C_cC_{gs2}g_{m1}g_{m4})$ $+s^3(2C_{gs1}C_Lg_{m2}(C_c+C_{gs4})$ $+4C_{gs2}C_Lg_{m1}(C_c+C_{gs4})+C_cC_{db1}C_Lg_{m1}$ $+2C_cC_{db1}C_Lg_{m2}+2C_cC_{gs1}C_{gs4}g_{m2}$ $+2C_cC_{gs1}g_{m4}(C_{db1}+2C_{gs2}))$ $+s^4(4C_cC_{gs2}C_L(C_{db1}+C_{gs1})$ $+2C_cC_{gs1}C_L(C_{gd1}+C_{db1})$ $+4C_{gs1}C_{gs2}C_{gs4}(C_c+C_L)$ $+C_cC_{db1}C_LC_{db5})$

Table 1: Small-signal voltage gain (1st criterion).

Fig.5 shows the pole locations as calculated by ASAP for both the complete and the simplified expressions of the gain. As it can be seen, 200% errors result in high frequency poles as a consequence of the simplification process.

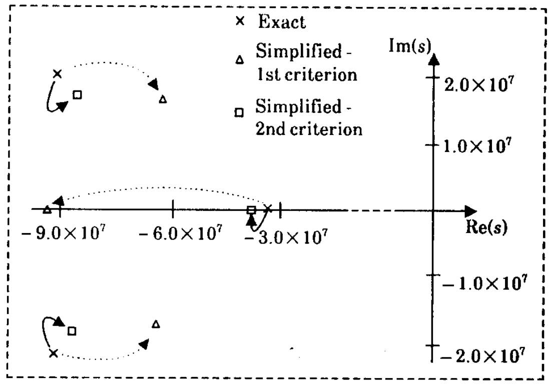

Figure 5: Pole displacement in simplified expressions.

This problem has been overcome via the development of a new simplification criterion. In this criterion the least significant terms are eliminated at small increments of the accuracy margin. At each step pole and zero displacements are controlled. Let us consider the general case of a pair of complex conjugate roots (i.e. two poles). Its contribution to the network function can be expressed as

$$H(s) = H'(s)\,\frac{1}{s^2 + 2\xi\omega_n s + \omega_n^2}$$

The real and imaginary part of the roots are allowed to vary while ω_n and ξ keep inside a limited range around their original value. Control on ω_n and ξ instead of the real and imaginary part of the roots prevents too conservative simplifications (i.e. the imaginary part of high frequency poles with grater real part than the imaginary one is very sensitive to small variations in the polinomial coefficients while its effect on circuit behavior is negligible). These constraints are simplified in the case of real roots to allow only small displacements over the real axis.

Besides, there also exists the possibility for the user to specify a frequency range where reducing the errors is specially important. In this case pole and zero displacements are more strictly limited in this band and constraints are gradually relaxed as root locations are far away from this region. Using this new criterion for the Miller opamp ASAP provides the expression of Table 2, the corresponding poles being the ones shown in Fig.5. As it can be seen, poles after simplification are kept much closer to the nominal positions than for the previous case.

ASAP graphical interface

Symbolic expressions generated by ASAP can be formatted and dynamically loaded for graphical representation. Two basic graphic interfaces have been developed so that symbolic outputs of ASAP can be easily interpreted: a) Representation of Bode diagrams, and b) Representation of pole and zero loci.

Bode diagrams for complete as well as for simplified expressions can be represented. It is also possible to get families of curves for different values of one generic parameter (any circuit component or transistor small signal parameter, any transistor dimension or the current across any element).

Numerator =	Denominator =
$+2g_{m1}{}^2g_{m2}g_{m4}$ $+\mathbf{s}(2C_{gs1}g_{m1}g_{m2}g_{m4}$ $+2g_{m1}{}^2g_{m4}C_{gs2}$ $-2g_{m1}{}^2g_{m2}C_c)$ $+\mathbf{s}^2(C_{gs1}g_{m1}g_{m4}$ $(C_{db1}+2C_{gs2})$ $-2C_cg_{m1}g_{m2}$ $(C_{db1}+C_{gs1})$ $-2C_cC_{gs2}g_{m1}{}^2)$ $-\mathbf{s}^3(2C_cC_{gs2}g_{m1}$ $(C_{db1}+C_{gs1})$ $+C_cC_{gs1}g_{m1}$ $(C_{db1}+C_{db2}))$ $+\mathbf{s}^4(C_cC_{gd1}C_{gs1}$ $(C_{db1}+C_{db2}+2C_{gs2}))$	$+2g_{m1}g_{m2}(g_{ds1}+g_{ds2})(g_{ds3}+g_{ds4})$ $+\mathbf{s}2C_cg_{m1}g_{m2}g_{m4}$ $+\mathbf{s}^2(2C_Lg_{m1}g_{m2}(C_c+C_{gs4})$ $+2C_cC_{gs1}g_{m2}g_{m4}+2C_cC_{gs4}g_{m1}g_{m2}$ $+C_cg_{m1}g_{m4}(C_{db1}+4C_{gs2}))$ $+\mathbf{s}^3(2C_LC_{gs1}g_{m2}(C_c+C_{gs4})$ $+C_cC_Lg_{m2}(2C_{db1}+C_{db5})$ $+C_Lg_{m1}(C_c+C_{gs4})(C_{db1}+4C_{gs2})$ $+2C_cC_{gs1}g_{m2}(C_{db3}+C_{gs4})$ $+2C_cC_{db2}C_Lg_{m1}+4C_cC_{gs2}C_{gs4}g_{m1}$ $+2C_cC_{gs1}g_{m4}(C_{db1}+C_{db2}+2C_{gs2}))$ $+\mathbf{s}^4(4C_cC_{gs2}C_L(C_{db1}+C_{gs1})$ $+2C_cC_{db3}C_{gs1}(C_{db1}+2C_{gs2})$ $+2C_cC_{gs1}C_L(C_{gd1}+C_{db1}+C_{db2})$ $+2C_{gs1}C_{gs4}(C_{db1}+2C_{gs2})(C_c+C_L)$ $+2C_{gs1}C_{gs4}C_{db2}C_L$ $+C_cC_{db5}C_L(C_{db1}+2C_{gs2}))$

Table 2: Small-signal voltage gain (2nd criterion).

The second interface allows to find out pole and zero coordinates as a function of one parameter (in the same previous sense of parameter). As the symbolic expressions are dynamically loaded, compilers and machine characteristics impose limits to the expression size. Hence, simplified expressions must be used. Accuracy in graphical representations founds the importance of obtaining simplified expressions with small errors in zero and pole locations.

Fig.6 has been obtained by ASAP for the voltage gain of Table 2. In Fig.6(a) the four poles and four zeros of the system function are plotted (as it is done by ASAP) versus the bias current of the differential input stage[†1]. For the output stage a bias current ten times greater than that of the differential pair was assumed. On the other hand, Figs.6(b) and (c) represent Bode diagrams of the voltage gain of this circuit. The X axis corresponds to the frequency (in Hz) in logarithmic scale. The Y axis in Fig.6(b) represents the magnitude in dB and in Fig.6(c) the phase in degrees. The families of curves are parametrized respect to the current across the bias transistor ($10^{-7} \leq I_{BIAS} \leq 10^{-4.5}$, in logarithmic scale).

Several pieces of useful information for the analogue design procedure can be extracted from a detailed examination of the graphic representations of Fig.6. For instance, for the different bias currents a tradeoff between gain and bandwidth is highlighted. Also, just to give another example, Fig.6(a) shows how to choose the bias current for a given dominant and non-dominant pole location. Moreover since for a selected bias current poles and zeros can be plotted again as a function of a new parameter much deeper insight can be gained into the circuit operation.

Conclusions

A symbolic analysis program has been presented which exploits the adequacy of the C language to deal with flowgraphs. Together with the inclusion of simplification techniques with enhanced accuracy, ASAP also incorporates the possibility of making symbolic pole/zero calculations.

In its current version ASAP is able to deal with the complexity levels arising in typical analog building blocks when described by device-level models. More complex circuits, for instance complete RC active filters, have to be described by means of macromodels. We are currently working in a new ASAP version which approaches the analysis of complex circuits in a hierarchical way.

1 Either conventional root locus diagram (where the root locations are drawn in the s-plane for different values of the parameter [5]) or diagrams like that in Fig.6 (where the real and imaginary part of the roots are separately drawn versus the parameter) can be provided.

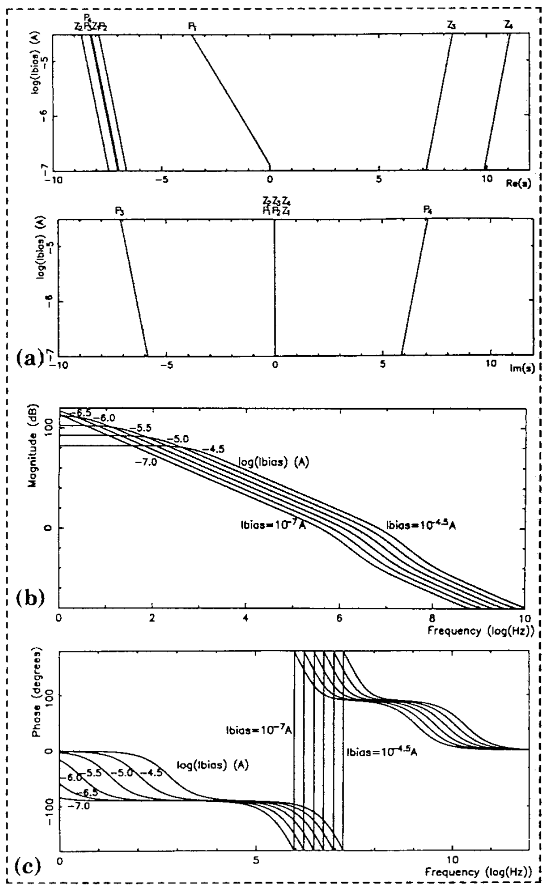

Figure 6:(a) Pole/zero location; (b) Magnitude; (c) Phase.

References

[1] P.M. Lin: "A Survey of Applications of Symbolic Network Functions", *IEEE Trans. on Circuit Theory*, Vol. 20, pp 732-737, Nov. 1973.

[2] T. Ozawa (editor): "*Analog Methods for Computer-Aided Circuit Analysis and Diagnosis*". Marcel Dekker, Inc., 1988.

[3] S. J. Seda et al: "A Symbolic Analysis Tool for Analog Circuit Design Automation", *Proc. IEEE 1988 ICCAD*, pp 488-491, Nov. 1988.

[4] G. Gielen et al: "ISAAC: A Symbolic Simulator for Analog Integrated Circuits", *IEEE Journal Solid-State Circuits*, Vol. 24, pp 1587-1597, Dec. 1989.

[5] B. C. Kuo: "*Automatic Control Systems*", Prentice-Hall, 1987.