BEHAVIOURAL/MACRO MODELLING TO SPEED-UP ANALOGUE FAULT SIMULATION

Yavuz Kılıç (Member IEEE) and Mark Zwoliński[1] (Senior Member IEEE)
e-mail: Yavuz.Kilic@philips.com
Philips Semiconductors, Southampton, UK

Keywords: macromodelling, behavioural modelling, VHDL-AMS, fault simulation

Abstract

The main difficulty in generating test patterns for analogue and mixed-signal circuits is fault simulation. Analogue fault simulation is much slower than the digital equivalent. Two of the techniques to speed up the analogue fault simulation process are: fault dropping/collapsing, in which faults that have similar circuit responses compared with the fault-free circuit response and/or with another faulty circuit response are considered equivalent; and behavioural/macro modelling, whereby parts of the circuit are modelled at a more abstract level, therefore reducing the complexity and the simulation time. This paper discusses behavioural/macro modelling in order to speed-up fault simulation for analogue circuits.

I. Introduction

As transistor sizes keep shrinking, integrated circuits (ICs) have been growing in size and complexity. This growth in ICs causes testing to be much more difficult. For digital circuits the problem of testing can be simplified by using standard fault models and fast fault simulation. Faults in digital circuits can be modelled as stuck-at, bridging and open faults. These structural faults can then be used to generate functional test vectors. The objective in developing a test program for a digital circuit is to determine whether or not a fault exists using the smallest possible number of test vectors [2]. Therefore, test pattern generation is the process of selecting an optimal set of tests from all possible input patterns. This optimal test pattern selection can be done in an ad-hoc manner for small and simple circuits. For larger circuits the optimal set of tests can be chosen using algorithms such as the D-algorithm or PODEM [2].

A test pattern is evaluated by looking at its fault coverage. All faults detected with this pattern can be dropped from further consideration. Fault simulation is done for the assessment of the fault coverage. There are number of fault simulation techniques for digital circuits. Serial fault simulation is perhaps the simplest method. For each fault, a copy of the circuit with the fault inserted into it is created. Then, all the faulty copies of the circuits along with the fault-free original are simulated with the given test patterns. If the output of a faulty circuit differs from the fault-free output, that fault is considered to be detectable.

Another fault simulation technique for digital circuits is concurrent fault simulation [2]. The differences between the faulty and the fault-free circuit behaviours might be relatively small. Therefore, in concurrent fault simulation the aim is to avoid redundant element evaluation when the fault-free and faulty behaviours are the same, hence reducing the computational effort.

Analogue and mixed-signal fault simulation has been limited to the serial technique. Faster methods are not easily applied to analogue and/or mixed-signal circuits, because faults do not affect the circuit behaviour in a binary manner.

One way to speed-up fault simulation for analogue and mixed-signal circuits is to use behavioural or macro models, where parts of the circuit are modelled at a more abstract level, reducing the complexity and hence the simulation time. In this paper we summarise research in behavioural/macro modelling for speeding up analogue fault simulation. The structure of the paper is as follows. First, macromodelling for analogue circuits is presented. Then behavioural modelling is discussed with a case study. In section IV, behavioural modelling using Hardware Description Languages (HDLs) is summarised. In secion V, a behavioural fault model is developed in VHDL-AMS for an opamp circuit operating in inverting amplifier configuration. Finally, in section VI some conclusions are drawn.

[1] University of Southampton, Department of Electronics and Computer Science, Southampton, UK, SO17 1BJ

II. Macromodels for Analogue Circuits

Simulation at the transistor level for analogue circuits is computationally very expensive. Therefore, one way to reduce this high simulation cost is to partition a large analogue circuit into smaller functional blocks such as opamps and replace each functional block with its *macromodel* or to describe each block using mathematical equations (a *behavioural model*). This technique is sometimes called *hierarchical simulation* [3].

The word *macromodel* usually refers to a compact representation of a circuit that captures those features that are useful for a particular purpose while discarding redundant information [4]. Macromodels developed for SPICE-like simulators are basically electrical networks containing devices such as voltage-controlled voltage sources instead of the full transistor network and with fewer nodes than the original circuit.

Many circuits are designed in a modular style, in which functional units are connected to achieve the design specifications. The behaviour of the whole circuit is determined by how the individual units interact with each other, while what happens inside each is unimportant in terms of the behaviour of the entire circuit. The accuracy of a macromodel must, therefore, be defined in terms of how closely its input-output behaviour matches that of the original unit [4].

Since the early 1970s, a number of macromodels have been developed, mainly for integrated operational amplifier circuits (opamps) [3]-[14]. Boyle et al developed a macromodel for integrated bipolar opamp circuits [5]. This macromodel was six times less complex (in terms of the node count) than the original opamp circuit, and the simulation time was an order of magnitude faster than the device-level model.

The derivation of component values for the Boyle macromodel is not, however, straightforward. Some parameters are modelled using unbalanced input devices and other parameters interact. Therefore, a modular approach was suggested [8], in which a macromodel was derived simply from the published data sheets. Individual parameters were modelled separately and the results combined to provide the output response. Since the parameters were separated they did not interact and only those required were included.

Recent research has focused on how to capture the effect of a fault in an analogue circuit within its macromodel [1], [3], [15]. In, [3] the fault macromodelling problem was formulated in terms of deriving the macro parameter set, *B*, based on the performance parameter set, *P*, (gain, bandwidth, samples on the frequency or time response curves, etc.) of the transistor-level faulty circuit. The accuracy of the macromodel was evaluated by checking the consistency of the performance parameter set, *P*, between the transistor-level circuit and the macromodel.

Two steps are needed to obtain the macromodel for a functional block within an analogue circuit [3]:

1. Perform transistor level fault simulation for each faulty circuit to obtain the value of the performance parameter set *P*
2. Map each performance parameter set *P* to the corresponding macro parameter set, *B*. This is referred to as *parameter mapping*.

It was assumed that the transistor-level fault list is given and the macromodel structure and the performance parameter set, *P*, to be matched are predetermined by the circuit designer.

There are several ways to do parameter mapping. One simple approach is based on analytical design equations that express the macro parameter set, *B*, as analytical functions of the performance parameter set, *P*, and the value of *B* is derived by function evaluation. As analogue ICs get more complex, this approach is becoming more difficult. Another simple approach is to build an empirical mapping function, $B=F(P)$, based on a large number of data pairs (*P*, *B*), referred to as the *training set* [3]. Usually the training set is generated by randomly selecting *M* out of the *N* performance parameter sets for the faulty circuits obtained by transistor-level simulation and then the value of the macro parameter set *B* for each selected *P* is derived. The derivation of each data pair usually requires multiple runs of macromodel-level simulation [3].

Macromodelling in general and fault macromodelling in particular, using SPICE-like languages, nevertheless, have been shown to be very difficult [1], [3]-[18]. Therefore, another easier and perhaps more efficient way of modelling analogue circuits at a higher level is necessary.

III. Behavioural Modelling

A behavioural model describes a circuit block in terms of mathematical equations modelling the functionality of the block, for example, in terms of the input-output relationship. Behavioural modelling has been used for speeding up analogue simulation in general [19] and analogue fault simulation in particular [1], [15], [18], [20]. In [19], analogue circuits were modelled behaviourally in the C programming language. Broyden's method was used to formulate and solve the model equations in a custom simulator. Broyden's method was originally proposed in [21] as an algorithm for the solution of systems of nonlinear equations, i.e. of the derivatives of a set of functions. The main drawback with the work described in [19] is that since the technique does not require derivatives it cannot be used for small-signal analysis.

In [15], Chang et al presented a behavioural fault model derived from a macromodel of a CMOS operational

amplifier from the IEEE Mixed-Signal Benchmark Suite [22] (Figure 1).

The faulty macromodel was developed using DC-sweep analysis. The DC behaviour of the benchmark opamp operating in inverting, non-inverting and unity gain amplifier configurations was first investigated under different faulty conditions, as shown in Figure 2. Single transistor catastrophic faults, bridging/short and nearly open faults, and parametric faults with W (channel width), L (channel length) and V_t (threshold voltage) varied by ±10% were used for each transistor. Then an attempt was made to group the different faulty behaviours. By comparing the fault-free offset voltage measured at the inputs of the opamp operating in one of the three configurations with the equivalent faulty circuits, four different equivalent fault types were derived [15]: M4 drain-to-gate short (Type I), M5 drain-to-source short (Type II), M7 drain open (Type III), and M5 drain-to-source short (Type IV). The first three fault types existed for the opamp operating in the inverting configuration; the Type IV fault group was found for the non-inverting configuration.

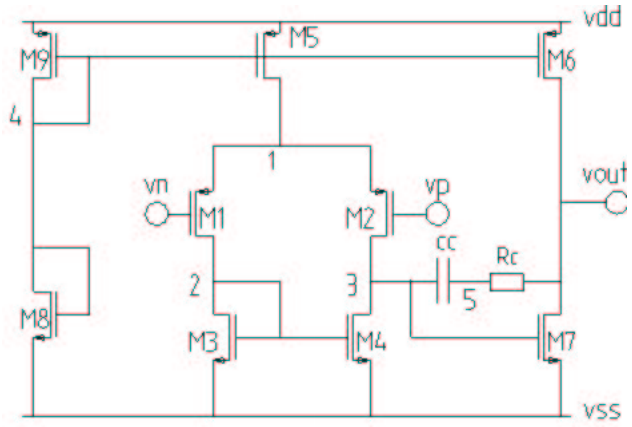

Figure 1. The 2-stage CMOS Miller opamp used in [15] for behavioural fault modelling.

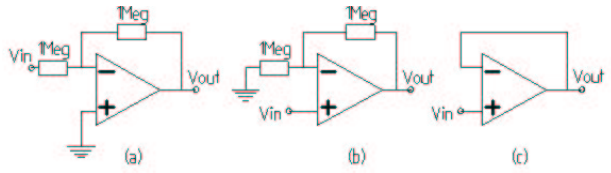

Figure 2. Three different configurations used in [15] for the benchmark circuit given in [22]: (a) Inverting amplifier, (b) non-inverting amplifier, and (c) unity gain buffer.

The input offset voltage (measured between the non-inverting and inverting inputs of the opamp in the closed-loop configurations) and the output voltage versus the input voltage for the fault-free opamp operating in the three configurations were determined by HSPICE simulations and are shown in Figure 3, Figure 4, and Figure 5, respectively.

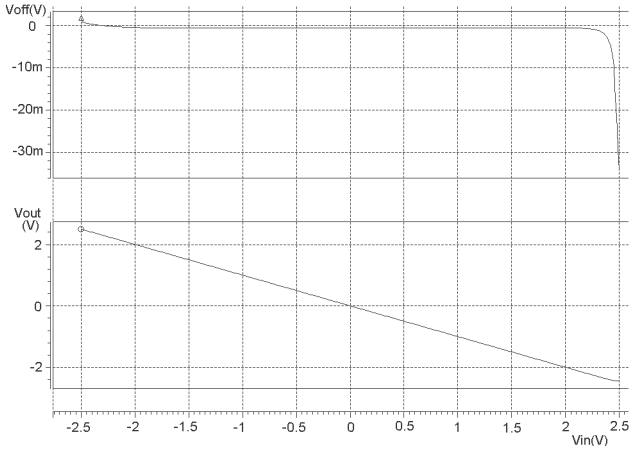

Figure 3. Input offset voltage and output voltage versus input voltage for the fault-free inverting amplifier.

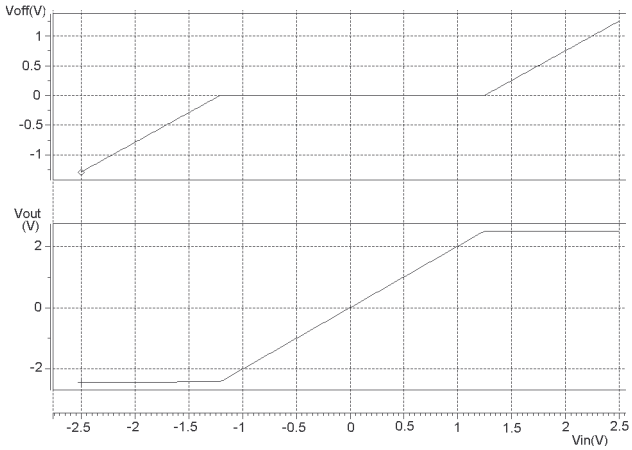

Figure 4. Input offset voltage and output voltage versus input voltage for the fault-free non-inverting amplifier.

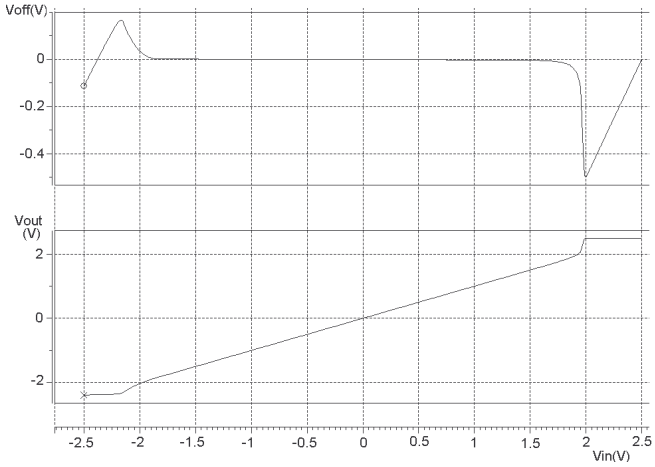

Figure 5. Input offset voltage and output voltage versus input voltage for the fault-free unity gain buffer.

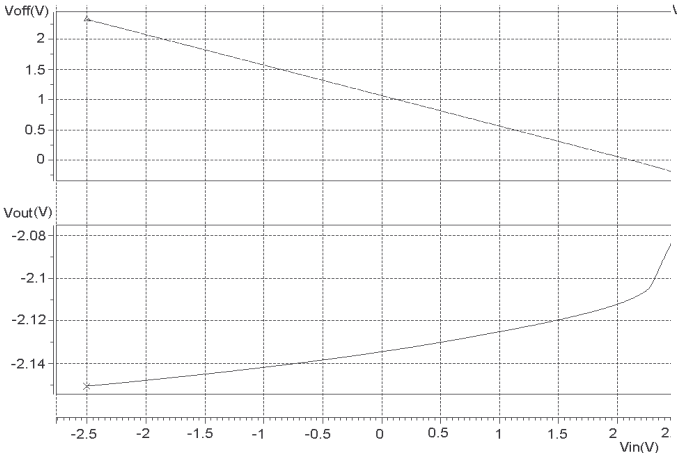

Figure 6. Input offset voltage and the output voltage for the Type I fault.

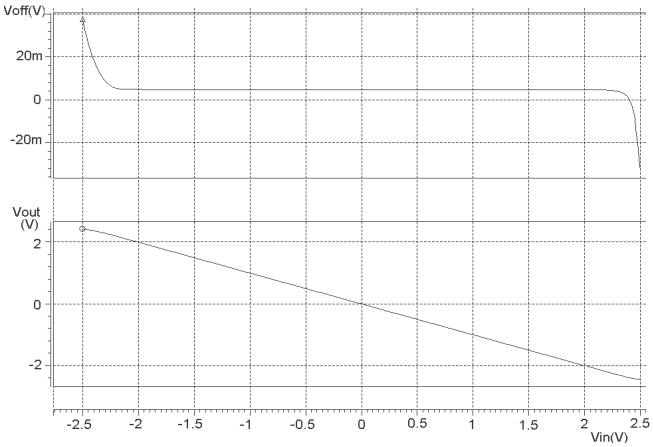

Figure 7. Input offset voltage and the output voltage for the Type II fault.

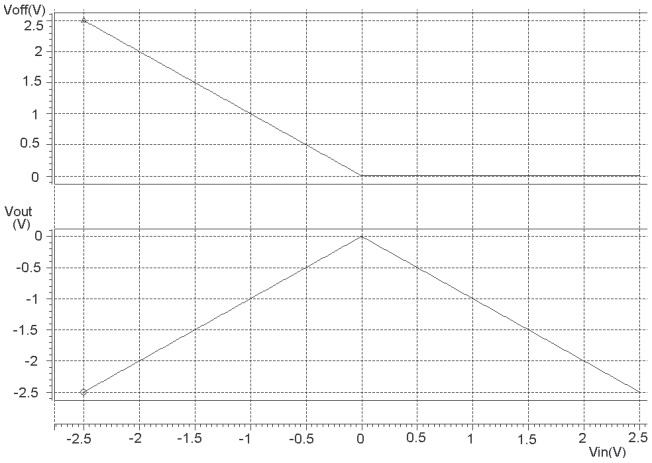

Figure 8. Input offset voltage and the output voltage for the Type III fault.

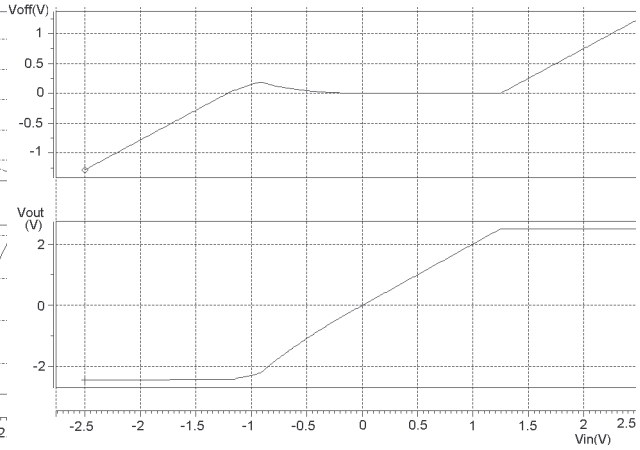

Figure 9. Input offset voltage and the output voltage for the Type IV fault.

The input offset voltage and the output voltage for each fault group with respect to the input voltage were also found by HSPICE simulations and are shown in Figure 6, Figure 7, Figure 8, and Figure 9, respectively.

As can be seen from Figure 6 and Figure 9 responses obtained for Type II and Type IV faults are quite similar to the fault-free responses in Figure 3 and Figure 4. Type II and Type IV input offset voltages are somewhat different from the fault-free responses. The input offset voltage has a small DC level for Type II faults, but has a non-linear characteristic for Type IV faults.

The remaining two faults have very different characteristics to the fault-free equivalents for both input offset voltages and output voltages. It can be concluded from the figures that a Type I fault causes the inverting amplifier output to be nearly stuck at a negative voltage near to the negative supply voltage level. A Type III fault causes the inverting amplifier output to have a non-inverting characteristic for the negative values of the DC input signal, and an inverting characteristic for the positive values of the DC input signal. As can be seen from the figures above, the input offset voltage at the inputs of the opamp has a linear characteristic for Type I faults, and a piecewise linear characteristic for Type III faults.

The macromodel given in Figure 10 for the inverting opamp was used to derive the input output relationship under fault conditions [15]:

$$V_{out} = A_{CL}\left[(1+m)V_{in} + k\right] \tag{1}$$

where A_{CL} is the closed-loop gain for the opamp. The parameters m and k are given in [15] as:

$$m = \frac{-R2}{D+R2} \tag{2}$$

and

$$k = aVos + bVdd + cVss \qquad (3)$$

where

$D = B\left(R2 \,//\, Ro \,//\, Rdd \,//\, Rss\right)$,

$B = \left(\frac{A}{Ro} - \frac{1}{R2}\right)\left(Rid \,//\, R1 \,//\, R2 \,//\, 2Ricm\right)$,

$a = \frac{R2 \,//\, D}{A_{CL}\left(R1 \,//\, R2 \,//\, 2Ricm \,//\, BR2\right)}$,

$b = \frac{SF}{A_{CL} Rdd}$,

$c = -\frac{SF}{A_{CL} Rss}$,

$A_{CL} = -\frac{R2}{R1}$,

$SF = Rdd \,//\, Rss \,//\, Ro \,//\left(R2 \,//\, R11\right)// \frac{Ro\left(R11 \,//\, R2\right)}{AR11}$

$R11 = R1 \,//\, 2Ricm \,//\, Rid$,

and A represents the open-loop gain.

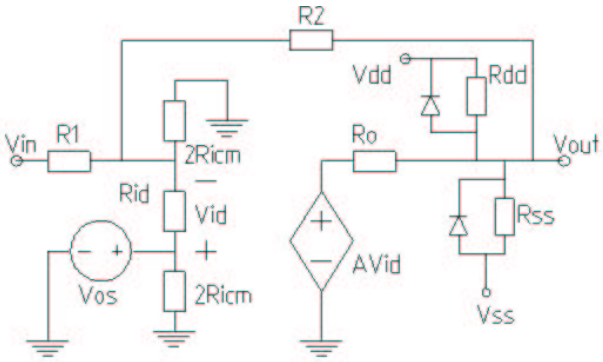

Figure 10. Macromodel used in [15] to derive the input-output relationship for the closed loop inverting opamp.

The non-ideal effects such as the input offset voltage, *Vos*, the finite open-loop gain, *A*, and the finite input and output resistances, *Rid* (differential mode input resistance), *Ricm* (common mode input resistance), *Ro* (output resistance), and the resistances from output node to the supply rails (*Rdd* and *Rss*) to model output stuck-at faults were taken into account in deriving equation (1). Note that for the fault-free case, *Rid*, *Ricm*, *Rdd*, *Rss*, and *A* would be infinite, *Vos*, and *Ro* would be zero, hence $m \to 0$, and $k \to 0$. When a fault causes the output to be stuck at some voltage level, $D \to 0$, therefore $m \to -1$, and k is the value of the stuck output voltage; the closed-loop gain, A_{CL}, is assumed to be unity. As they are dealt with elsewhere [15], the derivation of the above equations will not be given here.

In [15], the current limiting effect was also modelled. This is due to the finite supply voltage at the output of the opamp. It is claimed that the model covers all the parametric faults and 92.5% of the catastrophic faults that were considered. The model could not model M4 drain-to-gate short, M5 drain-to-source short, M1 open-gate faults for the non-inverting amplifier and M2 drain-to-gate short, M4 drain-to-gate short, M5 drain-to-source short, M1 open gate, M3 open source and M5 open gate faults for the unity gain buffer.

IV. Behavioural modelling using HDLs

HDLs have been in use for behavioural modelling and simulation of digital circuits as well as analogue electronic systems, fluid concentrations in chemical processes, and parachute jumps since 1960 [26]. Currently two of the most widely used standards for modelling digital designs are VHDL [23], and Verilog [24]. For analogue circuits, the choice has been between SPICE and proprietary analogue HDLs.

Analogue HDLs support the description of systems of differential and algebraic equations (DAEs). The solution of these systems varies continuously with time. Most analogue HDLs support both structural composition and conservation semantics, in addition to behavioural descriptions. Examples of such languages are FAS [27] SpectreHDL [28] and Verilog-A [29].

Mixed-signal design has depended on the use of separate HDLs for the analogue and digital parts or, again, on proprietary languages. Mixed-signal languages support both event-driven techniques and differential and algebraic equations in one simulator. Simulators in this category are MAST/Saber [30], VeriasHDL [30], AdvanceMS [27], Hamster [31].

Both VHDL and Verilog have been extended to analogue and mixed-signal design: VHDL-AMS [32], and Verilog-AMS [29]. The analogue extensions to VHDL and Verilog should alleviate the multiple-language problem [25].

Since VHDL-AMS was standardised in 1999 there has been some work done on fault modelling using VHDL-AMS. One reason for the limited progress is perhaps that there is not yet a robust VHDL-AMS simulator available that has all the VHDL-AMS constructs, such as procedural statements, implemented. Perkins et al attempted to use analogue VHDL for fault modelling and simulation with very limited success [1]. The authors used the HDL-A modelling language with the ELDO simulator from Anacad. Behavioural model simulation using HDL-A and ELDO was over 4.6 times slower than the macromodel simulation carried out using HSPICE, as reported in [1]. One of the reasons was that the semiconductor device models implemented in ELDO were not as efficient as those in HSPICE.

V. A VHDL-AMS behavioural fault model for the inverting opamp

A VHDL-AMS model for the behavioural model given in (1) is developed and given in Figure 11.

```
--behavioural opamp
LIBRARY DISCIPLINES;
LIBRARY IEEE;
USE
DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;

--entity
ENTITY op_behav IS
  GENERIC ( m : real := 0.0; --fault-free value
            k : real := 0.0; --fault-free value
            Acl : real := -1.0; --closed-loop gain
            rin : real := 4.0e5;);
 PORT (TERMINAL inn, outt : electrical);
END;

--architecture
LIBRARY DISCIPLINES;
LIBRARY IEEE;
USE
DISCIPLINES.ELECTROMAGNETIC_SYSTEM.ALL;
USE IEEE.MATH_REAL.ALL;

ARCHITECTURE behav OF op_behav IS
  quantity vout across iout through outt;
  quantity vin across iin through inn;

begin
iin  == (vin + (1.0 + m)*vin) / rin;
vout == Acl * (vin + m * vin + k);
end;
```

Figure 11. A VHDL-AMS behavioural fault model for the inverting operational amplifier for the fault-free case.

As can be seen from the Figure 11, it is much simpler to implement the behavioural model in VHDL-AMS compared with the macromodel development using SPICE-like languages.

V. Conclusion

In this paper, we have discussed behavioural and macromodelling techniques in order to speed-up analogue fault simulation process. We also have developed a behavioural fault model in VHDL-AMS for an opamp operating in inverting amplifier configuration. Capturing a circuit behaviour under faulty conditions at a higher level using mathematical equations (behavioural modelling) is somewhat simpler than trying to come up with the macromodels for that circuit. As VHDL-AMS and Verilog-AMS have now been standardised, it should be easier to develop behavioural fault models using these standard languages.

References

1. A. J. Perkins, M. Zwolinski, C. D. Chalk and B. R. Wilkins, “Fault Modeling and Simulation Using VHDL-AMS”, Analog Integrated Circuits and Signal Processing, vol. 16, pp. 141-155, 1998.
2. M. Abramovici, M.A. Breuer, A.D. Friedman, "Digital Systems Testing and Testable Design", IEEE Press, 1990.
3. C.-Y. Pan and K.-T. Cheng, “Fault Macromodeling for Analog/Mixed-Signal Circuits”, IEEE International Test Conference, ITC’97.
4. G. Casinovi and A. Sangiovanni-Vincentelli, “A Macromodeling Algorithm for Analog Circuits”, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 10, No. 2, pp. 150-160, February 1991.
5. G. A. Boyle, D. O. Pederson, B. M. Cohn and J. E. Solomon, “Macromodeling of Integrated Circuit Operational Amplifiers”, IEEE J. of Solid State Circuits SC-9, pp. 353-363, 1974.
6. C. Chalk and M. Zwolinski, “Macromodel of CMOS operational amplifier including supply current variation”, Electronics Letters 31, pp. 1398-1400, 1995.
7. P. Mandal, V. Visvanathan, “Macromodeling of the AC Characteristics of CMOS Op-Amps”, IEEE 1993 Conference on Computer Aided Design, Digest of Technical Papers, pp.334-339, 1993.
8. M. E. Brinson, D. J. Faulkner, “Modular SPICE macromodel for operational amplifiers”, IEE Proc.-Circuits Devices Syst., Vol. 141, No. 5, pp. 417-420, October 1994.
9. M. E. Brinson, D. J. Faulkner, “A SPICE Noise Macromodel for Operational Amplifiers”, IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications, Vol. 42, No. 3, pp. 166-168, 1995.
10. G. Krajewska and F. E. Holmes, “Macromodeling of FET/Bipolar Operational Amplifiers”, IEEE Journal of Solid-State Circuits, Vol. SC-14, No. 6, pp. 1083-1087, 1979.
11. C. Turchetti and G. Masetti, “A Macromodel for Integrated All-MOS Operational Amplifiers”, IEEE Journal of Solid-State Circuits, Vol. SC-18, pp. 389-394, 1983.
12. M. E. Brinson, D. J. Faulkner, “SPICE macromodel for operational amplifier power supply current sensing”, Electronics Letters, Vol. 30, No. 23, 10th November 1994.
13. R. V. Peic, “Simple and Accurate Nonlinear Macromodel for Operational Amplifiers”, IEEE Journal of Solid-State Circuits, Vol. 26, No. 6, pp. 896-899, 1991.
14. B. Perez et al, “A New Nonlinear Time-Domain Op-Amp Macromodel Using Threshold Functions and Digitally Controlled Network Elements”, IEEE

Journal of Solid-State Circuits, Vol. 23, No. 4, pp. 959-971, 1988.
15. Y.-J. Chang et al, “A Behavior-Level Fault Model for the Closed-Loop Operational Amplifier”, Journal of Information Science and Engineering, Vol. 16, No. 5, pp. 751-766, September 2000.
16. B. Al-Hashimi, “Behavioural Simulation of Filters”, IEE Colloquium on Analogue simulation: the dream & the nightmare, November 1995.
17. A. I. Kayssi, K. A. Sakallah, “Macromodel Simplification Using Dimensional Analysis”, 1994 Int. Symp. On Circuits and Systems, pp. 335-338, 1994.
18. M. Zwolinski, Z.R. Yang and T. J. Kazmierski, “Using robust adaptive mixing for statistical fault macromodelling”, IEE Proceedings: Circuits, Devices and Systems, vol 147, no 5, pp. 265-270, Oct 2000.
19. G. Casinovi, “Multi-Level Simulation of Large Analog Systems Containing Behavioral Models”, IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, pp. 1391-1399, Vol.13, No.11, November 1994.
20. E. Bruls, et. al. , “Analogue fault simulation in standard VHDL”, IEE Proc. Circuits Devices Syst., pp.380-385, Vol. 143, No. 6, December 1996.
21. C. G. Broyden, et. al., “A class of methods for solving nonlinear simultaneous equations”, Mathematics of Computation, vol. 19, no. 92, pp. 577-593, 1965.
22. http://www.ee.washington.edu/mad/benchmarks/benchmarks.html
23. VHDL Language Reference Manual, IEEE Standard 1076-1993.
24. Standard Description Language Based on the Verilog™ Hardware Description Language, IEEE Standard 1364-1995.
25. http://www.ednmag.com
26. E. Christen, and K. Bakalar, “VHDL-AMS, A Hardware Description Language for Analog Applications”, IEEE Transactions on Circuits and Systems-II: Analog and Digital Signal Processing, Vol.46. No.10, October 1999.
27. www.mentor.com
28. www.cadence.com
29. www.verilog.com
30. www.analogy.com
31. http://www.hamster-ams.com/
32. www.eda.org/analog