Accelerated Analog Fault Simulation by Concurrent Fault Detection

Mustapha Dhifi *

Abstract

This paper presents an analog fault simulation approach based on concurrent fault detection. By performing fault simulation with fault dropping, high computational effort is saved. The concurrent fault detection can be applied to most analyses and is compatible with other speedup procedures. Experimental results demonstrate the effectiveness of the presented fault simulation procedure.

1 INTRODUCTION

Testing of analog circuits still do not attain the maturity of that of digital circuits. This is in part because of the very time consuming fault simulation needed for test stimuli generation and for measuring the test quality by means of the fault coverage. Attempts have been made in several ways to cut down the fault simulation time. While some researchers reduced the fault set by means of techniques such as fault collapsing, inductive fault analysis [1] and L^2RFM [2], others were concerned with the simulation task itself. The focus in this paper will be on reducing the simulation time, which can be dealt with in different ways. Instead of analyzing the circuit on the transistor level, simulations could be run faster, analoguous to digital circuits, on higher levels using macromodels or behavioral description languages. However, macromodels have to be generated first. Available macromodels are mainly restricted to some circuits such as opamps. Behavioral models are generally valid for a certain range of input. When used in fault simulation, the inputs of the modeled partitions of the circuit are not easily manageable, since an injected fault almost drives the circuit outside its normal operation scope. Consequently, the behavioral models could lead to erroneous results. Because of the handicaps related to the fault simulation on higher levels, procedures to speed up the simulation should focus on the circuit level. However, the simulation on the transistor level is very time consuming. Therefore techniques have to be developed to make such simulation useful, in spite of the underlying expensive approach. In this paper a fault simulation approach based on the analysis of circuits on the transistor level and the use of a concurrent fault simulation will be presented. The fault simulation time consumption is reduced by avoiding computational effort related to a simulation beyond an achieved fault detection.

* University of Hannover - Germany
Institute of Electromagnetic Theory. E-mail: *dhifi@tet.uni-hannover.de*

This paper is organized as follows: after the state-of-the-art, the analog fault simulation is described in section 3. In section 4, an approach for a concurrent fault detection is presented. The paper is illustrated by experimental results in section 5 and concluded in section 6.

2 STATE-OF-THE-ART

Fault simulation speedup has been the subject of many investigations. Attempts concerning special circuits have been reported e.g. for RF circuits in [3]. In [4] fault simulation based upon opamp macromodels has been presented. Fault simulation using behavioral models has been reported in [5]. Approaches to speed up the simulation on the transistor level share the objective of avoiding unnecessary computations. Tian and Shi presented in [6], especially for DC fault simulation, an approach based on one-step Newton-Raphson iteration implemented by Householder's formula. In [7] the circuit inactivity has been reported to be considered in reducing the system matrix order. In an advanced approach [8][9], the computational effort is reduced by only computing the differences between circuits simulated concurrently. In [8] a concurrent fault simulation approach like that applied to digital circuits has been presented. Here a number of faulty circuits are simulated concurrently with the fault-free version. The objective of this algorithm was not to re-evaluate parts of the faulty circuit behaving in the same way as in the fault-free case. Chatterjee and Hou [9] implemented the concurrent fault simulator CONCERT based on saving computational effort using residual similarities between the faulty and fault-free circuits. This approach was accomplished by techniques for fault ordering and state prediction.

In this paper an accelerated fault simulation approach using a concurrent fault detection will be presented.

3 THE FAULT SIMULATION SCHEME

To reduce the costs of production testing, fault simulation should be performed towards optimized test sets. Before simulation, fault set, simulation fault models and detection tolerances should be available. These can be provided by a preprocessor to the fault simulation. The fault set can be gained using e.g. inductive fault analysis (IFA). The fault models are a mapping on the electrical level of the defects the considered faults are based on. The detection tolerances are usually obtained by tolerance analysis.

With the common fault simulation approach, circuits are simulated in a series or in parallel. The responses of the faulty and fault-free circuits are subsequently compared based on the detection tolerances. Here the evaluation process is run irrespective of the simulation. In contrast, in our fault simulator the fault detection algorithm is integrated in the circuit analysis algorithm. Here the circuit analysis proceeds as long as the fault is still not detected. For an emerging analysis point meeting the detection criteria, the circuit analysis is broken off and the fault simulation continues with the next faulty circuit. The simulation speedup is achieved by leaving out a superfluous circuit analysis in terms of testing (from break point to the end of analysis interval).

In the following the complete fault set will be denoted with F, the detected faults with F_d and the undetected faults with F_u ($F = F_d \ U \ F_u$). Assuming the stimuli $S_1 .. S_m$ are applicative stimuli for the detection of the faults considered, the fault simulation scheme is as follows:

1. Based on the fault list and the fault models the netlists of the possible faulty circuits are generated.
2. Starting with $i=1$ ($F_{u,0}=F$), the good and the faulty circuits are simulated using the stimulus S_i as follows:

2.1 The fault-free circuit is simulated, beside the results needed -as a reference- for the fault detection, the results of the first analysis point are also saved.

2.2 The faulty circuits are simulated using values gained from the analysis results saved in (2.). The circuit analysis is run with a concurrent fault detection and the simulation is aborted as soon as the considered fault is detected. This is when the evaluation variable is meeting the detection criteria at the actual analysis point (section 4). Thereby the simulation need not be carried out in the whole analysis interval. The new set of undetected faults is:

$$F_{u,i} = F_{u,i-1} - F_{d,i}$$

3. If ($F_{u,i} \neq \phi$), fault simulation proceed with the next simulation run using the stimulus S_{i+1} for the reduced fault set $F_{u,i}$.

After every simulation run of the faulty circuits (2.2), the fault simulator updates the list of undetected faults. For the evaluation variable, the detection's analysis step x_i and respective response y_i as well as deviation at the detection analysis step are reported for every detected fault. These data are then used in the production test optimization.

To cut down the fault simulation time, it is likely to run a series of fault simulations using different stimuli, each simulation run with a subsequent fault detection and fault dropping. This is usually done by beginning e.g. with a simple operation point analysis, than a DC analysis and if necessary a time domain analysis. Depending on the circuit under test, the presented approach might greatly save an overhead related to such repetitive fault simulation runs in two respects. First, since simulation, fault detection and fault dropping are done concurrently and any simulation beyond detection is controlled, fault simulation may start with an expensive analysis without worrying about a related time consumption. A time-domain analysis which takes into account all nonlinearities of a circuit would detect more faults than a DC analysis. Second, in contrast to the common approach, a long stimulus (large analysis interval) may lead to a higher fault coverage without automatically prolonging the fault simulation time. Thus a possibly undesirable variety of stimuli and consequently of test sets could be limited.

Furthermore, the presented approach saves memory and time usually spent to store simulation results necessary for results evaluation. Parallel simulation easily accomplishes the presented procedure, since here the simulations of the n different faulty circuits run independently and are consequently easy to process.

4 CONCURRENT FAULT DETECTION

The concurrent fault detection is performed based on gathered results during simulation and on detection tolerances. These are usually gained by Monte Carlo (MC) analysis considering the effects of process parameters on the circuit behavior. The most accurate way to determine the fault detection tolerances should be to perform MC simulations on the fault-free and all faulty circuits. However, this approach is very time-consuming. Techniques to reduce the number of MC simulations attempt to deduce necessary information mainly from the simulation results done for the fault-free circuit [10].

For the sake of better understanding, the concurrent fault detection will be explained in the following based on a simple detection criteria. The tolerances provided by a preprocessor are here represented by an array of bounds ε_i relative to each analysis point result y_i of the fault-free circuit. In the following frequency sweep, DC sweep and time-domain analyses will be considered as they are mostly used. Depending on the analysis applied, the simulation results could have different lengths. Consider first the frequency sweep and DC sweep analyses. The circuit response for a variable y is expressed in a fixed number of pairs (x_i, y_i), where x_i is a sequential source or frequency value. The values of the sweep variable (x-axis) will be denoted by an array $\boldsymbol{x}$, the responses of the fault-free and a faulty circuit version will be represented by arrays $\boldsymbol{y}$ and $\boldsymbol{y_f}$, respectively. As the analysis proceeds, the signature of the faulty circuit is compared in pairs to that of the fault-free circuit. i.e. for every x_i, the circuit responses y_i and $y_{f,i}$ of the fault-free and faulty versions, respectively, are evaluated based on the fault detection tolerances. The analysis is broken off as soon as a difference beyond a respective (to the current analysis point) threshold is registered.

In contrast to the AC or DC analysis, the simulation results of a transient analysis do not depend primarily on the analysis, but more on the activity of the circuit under consideration. Since circuits to be simulated in the fault simulation process are obtained after fault injection, these circuits generally have different activities. Consequently their simulation results have different lengths. In contrast to DC or AC analysis, the output sequence (x_i, y_i) for different circuit versions may not only be different in y_i but also in x_i. This makes the concurrent evaluation of the simulation results of a transient analysis more difficult compared to the other analyses. Enforcing the simulator to perform a transient analysis for all circuit versions at user-defined times is absolutely not recommended. How could such results be automatically evaluated in a concurrent manner ?

Analog to the considerations above, the analysis time points will be designated with the time arrays $\boldsymbol{t}$ and $\boldsymbol{t_f}$, the circuit responses with $\boldsymbol{y}$ and $\boldsymbol{y_f}$ for the fault-free and faulty circuit, respectively. The concurrent fault detection is carried out using a linear interpolation at specified times t_i*. Assuming an analysis time interval $[t_0,t_n]$ and should the fault be not detected at t_0*=t_0 , the succeeding evaluations should take place at t_i* as follows:

Case-1: ($t_{f,1} < t_1$) , t_1* is assigned to $t_{f,1}$. Here the faulty circuit is more active than the good circuit and t_i* is assigned to $t_{f,i}$ for all $t_{f,i} \le t_1$ as long as the fault is not detected . i.e. when using the simplified detection criteria if:

$$(y_i^* - y_{f,i}) < \varepsilon_i$$

with ε_i the fault detection tolerance and y_i* the interpolation of the fault-free circuit response $y(t)$ between the values y_0 and y_1 at t_i*:

$$y_i^* = \frac{y_1 - y_0}{t_1 - t_0} t_i^* + y_0$$

For an emerging j with ($t_{f,j} > t_1$) , t_i* is assigned to t_1 and a new time interval should be considered $[t_1,t_2]$ or $[t_{f,j-1},t_{f,j}]$, depending on the circuit activity of both good and faulty circuits.

Case-2: In the opposite case ($t_{f,1} > t_1$), t_1* would be assigned to t_1 and the interpolation is done for the response of the faulty circuit $y_f(t)$ between $y_{f,0}$ and $y_{f,1}$ at every t_i*. The computational effort for evaluation is negligible compared to that used for analysis. Thus the fault detection should, as long as the fault is not detected, be carried out many times for t_i*=t_i between t_0 and $t_{f,1}$ in spite of nonexistent analysis results of the faulty circuit at these times. Analogous to case-1, a new interval should then be considered.

As long as the fault is not detected, the circuit analysis proceeds with a concurrent fault detection in the new interval based on the same procedure mentioned above for the first time interval.

The concurrent fault detection can also be carried out with other detection approaches like other tolerance determination or based on extended detection criteria considering e.g. high slopes. Here a fault is likely to be considered as undetected even if the responses are considerably different. To make the fault detection more severe a user may specify that a fault should only be dropped if it is detected p times or if the faulty circuit response meet the considered detection criteria at q successive simulation points.

5 EXPERIMENTAL RESULTS

The fault simulation approach presented so far is implemented with extended simulation and detection features in an experimental tool. The tool provides analog fault simulation with different options such as fault dropping and assignment of starting values for the faulty circuits to decrease the computational effort for circuit analysis. In the following fault dropping will be considered. Fault simulation was run for the µ741 circuit, a MOS operation amplifier (OpAmp) and for biquadratic filter as well as for a fifth order filter in MOS technology (5-OFM), the circuits are summarized in Table 1.

For the sake of a clear insight, fault simulation was performed with a DC sweep analysis. The following hard faults related to transistors have been considered: Shorts between the three transistor terminals (MOS: gate, drain and source, bipolar: collector, base and emitter) as well as opens at each terminal. The resistances $10^{-2}\Omega$ and $10^{8}\Omega$ have been used to model shorts and opens, respectively.

Circuit	OpAmp	µ741	5OMF	Biquad
Trans.	7	21	49	63
Res.	3	16	19	56
Cap.	2	1	12	5
Faults	42	126	294	378

Table 1: Investigated Circuits

To demonstrate the time savings when using the presented approach, fault simulation has been run for the mentioned circuits using both approaches: the common one and the presented one with a concurrent fault detection (CFD). Each fault simulation was run with one stimulus. The fault

detection was carried out based on detection tolerances relative to the respective analysis point $\varepsilon_i = \varepsilon_{ri} * y_i$, where $\varepsilon_{r,i}$ is a coefficient (deduced from tolerance analysis) at analysis point *i*. $\varepsilon_{r,i}$ between 0.03 and 0.05 has been considered in results evaluations. For the circuits mentioned above, the fault simulation results concerning mainly time consumption and computational effort are summarized in tables 2-5 . Beside the total simulation time, the total count of iterations consumed in the fault simulation is also recorded. Furthermore, as no time domain analysis is performed, the total count of operation point analyses can also be used as a confident measure for the computational effort consumed during the DC sweep analysis.

Consider first the MOS OpAmp (Table 2). Using the presented approach, the computational effort and the time spent for circuit analysis are reduced to 40%. The total simulation time is cut down to 31%.

MOS opamp	common	CFD	saving
Sim. time (s)	84	5	60%
Iterations	2821	874	69%
Analysis points	4242	789	

Table 2: FS results of MOS OpAmp

The speed up achieved for the bipolar OpAmp (Table 3) was 2,5. For the 5-OMF (Table 4) the computational effort could be cut in half and the fault simulation could be run nearly two times as fast. The fault simulation of the Biquad (Table 5) was carried out nearly 6 times faster than that performed by the common approach.

µ741	common	CFD	saving
Sim. time (s)	176	54	70%
Iterations	28230	8010	72%
Analysis points	12726	679	

Table 3: FS results of the µ741 circuit

5-OFM	common	CFD	saving
Sim. time (s)	1018	489	52%
Iterations	77630	30359	61%
Analysis points	29694	2539	

Table 4: FS results of the fifth order Filter

Biquad	common	CFD	saving
Sim. time (s)	2712	472	83%
Iterations	160570	79554	51%
Analysis points	113778	7740	

Table 5: FS results of the Biquad

6 CONCLUSION

An analog fault simulation approach based on concurrent fault detection has been presented. This approach is not restricted to a special analysis and is compatible with other techniques to speed up the fault simulation process. For 4 sample circuits, experimental results have been presented with a series of fault simulations. The presented approach was demonstrated to be 2 to 6 times faster than the standard approach without any trade-off in test efficiency. Future work will focus on the incorporation of other fault simulation acceleration techniques.

7 REFERENCES

[1] J. P. Shen, W. Maly, F. J. Ferguson, *"Inductive Fault Analysis of CMOS Integrated Circuits"*, IEEE Design&Test, 12-1985.

[2] M. J. Ohletz, *"Realistic Fault Mapping Scheme for the Fault Simulation of Integrated Analogue CMOS Circuits"*, ITC '96, pp. X-x.

[3] R. Telichevsky, K. Kundert, I. Elfadel, J. White, *"Fast Simulation Algorithms for RF Circuits"*, Custom Integrated Circuits Conference '96, pp.437-444.

[4] M. Zwolinski, C. Chalk, B. R. Wlikins, *" Analogue Fault Modeling and Simulation for Supply Current Monitoring*, EDTC '96, pp. 547-552.

[5] B. Straube, W. Vermeiren, U. Namyslo, *"On Problems with Hierarchical Analogue Fault Simulation"*, IEEE European Testing Workshop '97, pp. 1-5.

[6] M. W. Tian, C. -J. R. Shi, *"Nonlinear Analog DC Fault Simulation by One-Step Relaxation"*, VLSI Test Symposium '98, pp. 126-131.

[7] J. S. Augusto, C. F. B. Almeida, *"Fast Fault Simulation of Linear and Nonlinear Circuits with Fault Rubber Stamps"*, Internationa Mixed-Signal Testing Workshop '97, pp. 28-37.

[8] M. Zwolinski, A. D. Brown, C. D. Chalk, *"Concurrent Analogue Fault Simulation"*, International Mixed-Signal Testing Workshop '97, pp. 42-47.

[9] J. Hou, A.Chatterjee, *"CONCERT: A Concurrent Fault Simulator for Analog Circuits"*, International Mixed-Signal Testing Workshop '98, pp. 3-8

[10] S. J. Spinks, I. M. Bell, *"A Comparison of the Relative Accuracy of Fault Coverage Analysis Techniques based on Analogue Fault Simulation"*, International Mixed-Signal Testing Workshop '96, pp. 17-22.