Comprehensive Fault Modeling and Simulation in CMOS ICs

B. Riccò M. Favalli P. Olivo
DEIS, University of Bologna
Viale Risorgimento 2, 40136 Bologna, Italy

Abstract

This paper presents a brief overview of fault modeling and simulation in CMOS circuits with particular regard to cases not adequately represented in terms of the classical fault model. The most important models used to describe transistor faults, bridgings and delay faults are presented and the main problems to be solved for their implementation in circuit simulators are discussed in some detail. Typical coverage results of these type of faults obtained with stuck-at oriented test sequences are also presented in order to show the need to explicitly consider non stuck-at faults when evaluating the quality of test sequences.

1 Introduction

The exhaustive functional verification of the correct logic behavior of modern digital ICs is rapidly unfeasible, because of the complexity of the implemented functions. To cope with this problem, the testing philosophy has been changed by assuming that errors in the IC behavior are provoked by internal faults due to physical failures and process defects [1].

As a consequence of this assumption, IC testing is transformed in the more manageable problem of verifying that circuits are free of all possible faults.

To reduce the large multiplicity of possible cases to be considered, all possible physical failures and defects are grouped in a much smaller number of categories (depending on the kind of degradation induced in the circuit behavior) each described by a specific fault model [1,2].

Such models, can be conveniently used, together with the knowledge of the circuit structure, in order to generate and evaluate test sequences aiming at distinguishing faulty from fault-free circuits. These sequences must detect all possible faults inside the circuit by exposing them as errors at observable outputs.

To this regard, the test generation complex of VLSI circuits requires the use of computer aids, generally based on the same simulation techniques largely used in the design phase. In particular, CAD tools must be used in order to:

- evaluate the quality of test sequences, in general measured as the percent of detected faults (fault coverage);
- improve fault coverage, by generating specific test vectors.

These operations are performed by means of fault simulation and test generation algorithms, respectively. For these reasons, fault models must be well suited for simulation. In addition to the simplifications inherent to fault modeling, further hypothesis are made in order to reduce the number of possible faults to be considered. Among these, the most important ones are that faults can occur only one at a time (single fault assumption) and are permanent, i.e. if present cannot be removed.

Naturally, fault models must be consistent with circuit description and simulation levels, among which the most important ones are schematically illustrated in Table 1.

Furthermore, at each simulation level, fault models must lead to adequate trade-offs between the conflicting needs of accuracy in describing the behavior of faulty circuits and efficiency. In the case of large circuits featuring a relevant number of possible faults this latter requirement in practice prevents the use of simulation levels below the logic one.

In bipolar and nMOS circuits, the stuck-at fault model, in which a line is locked in the logic state 1 or 0 independently of driving networks, is overwhelmingly used since it can represent a large number of different physical failures and is ideally suitable for logic simulation.

In CMOS circuits, however, several physical mechanisms and process irregularities give rise to failures that, depending on the circuit structure, result in shorts, breaks and faulty devices [3,4,5,6,7,8], most of which cannot be represented simply in terms of stuck-at lines. Unfortunately, these faults represent a relevant fraction of those encountered in real CMOS ICs [3,5,6].

In particular, the classical stuck-at 0/1 fault model does not describe failures that: 1) by interrupting a conductive path in the pull-up or pull-down network of a CMOS gate, lead to a high impedance signal node; 2) by creating a conductive path between power supply (VDD) and ground, give rise to intermediate voltages at signal nodes.

From the electrical point of view, the first type of failure can induce sequential effects, since high impedance nodes can retain their previous states because of the non negligible capacitances typical of CMOS gates. The other case, instead, leads to conductance conflicts that must be solved in order to determine how intermediate signal voltages are propagated (i.e., whether or not they produce logic errors).

At logic level, a large number of failures of the first type can be modeled as transistors that are locked in their non conductive state independently of the driving signal (transistor stuck-open faults) [1,9,10], while those belonging to the second category can be often modeled as transistors that are always conductive (transistor stuck-on faults) [1,9,10] or with the presence of undesired connections between circuit nodes (bridging faults) [11,12].

A further category of faults results in floating transistors gates: in this case the transistor behavior depends both on technological features and capacitive coupling, thus it is not well predictable from the point of view of logic simulation.

In addition, the so called delay fault model has been introduced [13,14] to represent failures affecting the circuit dynamic behavior by increasing propagation delays. With the increasing speed of modern microcircuits, this type of fault is becoming progressively more important.

Because of the significant occurrence probability of non stuck-at faults, neglecting the models mentioned above during fault simulation leads to unrealistic fault coverages, with the consequence of a low yield of the whole testing process [15].

Since faults may affect the circuit operation in different ways (for instance, modifying the logic function or timing), different testing techniques can be used to detect their presence [16]. Among
these, the most important one is the observation of the static response of the circuit (since this technique is overwhelmingly used, in this paper we will refer implicitly to it). In addition, however, the measurements of quiescent currents (current testing) and the observation of the circuit timing (delay fault testing) are also used.

All these techniques may be applied by means of automatic testing equipments (ATES) or by means of extra circuitry integrated on chip. The relative advantages of different techniques depend not only on the achievable levels of fault coverage, but also on other issues, such as test generation and evaluation case, time required to test a circuit and ease of implementation.

For a given circuit description level, a set of fault models and a detection technique, fault simulation is used to study the effectiveness of test sequences. This is usually accomplished by comparing the response of faulty and fault-free circuits. To this regard, several general algorithms have been developed to reduce the complexity of fault simulation (deductive, concurrent, parallel and critical path).

In this paper, however, we will not discuss extensively fault simulation, but will restrict our attention to the specific problem of non stuck-at fault modeling and simulation in CMOS circuits.

2 Fault Models

This section presents the most important fault models relevant for CMOS ICs. Although such models have a more general validity, for simplicity our discussion will be restricted to the case of static circuits.

2.1 Stuck-at lines

The stuck-at fault model represents a line locked in the logic state 1 or 0 independently of the driving networks (for instance as a result of shorts with the power supplies).

Even if unable to represent all possible failures occurring in CMOS circuits, the stuck-at model plays a central role in the theory and practice of digital circuit testing, because:

1. all test generation and fault simulation algorithms have been originally developed for this kind of fault, essentially because it has the advantages of being technology independent and ideally suitable for logic simulation;

2. the detection of a stuck-at fault implies the observability of the involved signal line, and this represents a necessary condition for the detection of other type of faults possibly affecting such a line. Because of this property, in several cases the detectability of different faults can be deducted from that of stuck-ats.

2.2 Transistor Stuck-Open

A transistor stuck-open fault (namely a FET locked in a non conductive state independently of the control signal) represents physical such failures as missing source and/or drain contacts, broken connection lines. Such a fault, if activated, may interrupt a conductive path thus leading to a high impedance signal node that temporarily retains its previous state.

Because of this sequential behavior, the detection of this fault requires at least two test vectors. The first is used to initialize the gate output to a logic value opposite to that driven by the network (pull-up or pull-down) including the faulty device. The second is required to activate the fault: to this purpose it must create between one of the power supplies and the output of the fault-free gate, a path that is conductive if and only if the faulty device is on; furthermore such vector must propagate the gate output at value primary outputs (POs). As an example, Fig. 1 shows the detection of a stuck-open fault in a CMOS gate. At this regard it is interesting to notice that the second test vector may be substituted by a sequence of vectors, each of them activating the fault while only the last vector propagates the gate output value. This case, however, has the drawback that the initialization state stored in the high impedance node can be more easily removed by leakage currents.

2.3 Transistor Stuck-On

A transistor stuck-on fault (namely, a FET locked in the conductive state independently of its controlling voltage) represents physical failures such as drain-to-source shorts and errors in threshold voltage adjustment. In the presence of such a fault, a conductive path can be incorrectly activated, thus connecting a signal node to both power supply and ground. As a consequence, the signal node is driven to an intermediate voltage that depends on the conductance ratio between the faulty and the correct subnetworks. An example of a stuck-on fault in a CMOS gate is shown in Fig. 2.

A faulty signal can produce logic errors at the POs only if it lies on the wrong side of the gate logic threshold (VT), namely the one opposite to the fault-free case. Therefore, in fault simulation, the conductances of correct and faulty networks must be realistically evaluated.

Naturally, the conductance of the stuck-on device depends on the kind of failure for instance in the case of a short between source and drain terminals the conductance, in general, will be greater than in the case of an error in VT adjustment.
Since a stuck-on fault (depending on the values of conflicting conductances) may not be detectable as a steady state error at POS, current testing may be used to detect the increase in static current due to the faulty conductive path between power supply and ground.

2.4 Bridging Faults

This model describes the presence of shorts between two or more connection lines (possibly due to the presence of extra metal or diffusions, as well as breakdown of insulating layers). In general, in order to reduce all possibilities to a manageable number, only bridgings between two lines (or nodes) at a time are considered, even if a more general approach should be based on a ranking of bridgings on the basis of their occurrence probability (strongly dependent on circuit layout [9]).

Bridging faults may or may not create feedback in the circuit. The two cases are referred to as Feedback Bridging Faults (FBF) and Non Feedback Bridging Faults (NFBF), respectively. Since the detection of the former is normally easy [21], hereafter we will consider only the case of NFBF.

In general, to detect a NFBF the two shorted nodes must be driven to opposite values (in the fault-free circuit), furthermore, the node that in the faulty circuit changes its logic value must be observable.

Bridgings may be also classified, depending on the location of the shortened nodes. The most important cases, in this sense, are bridgings between:

1. gate output lines [11, 21, 22];
2. nodes internal to a gate [23];
3. gate and drain (or source) of a transistor [7].

Examples of bridgings belonging to these categories are illustrated in Fig. 3.

With regard to case 1, extensively treated in the literature, the fault can be activated by controlling the two shorted nodes at opposite logic values. As a consequence, a conductive path is created between the power supply and ground through the bridging path and the voltage of the shorted nodes assumes an intermediate value that depends on the conductance ratio between the conflicting driving networks. Therefore, in the case of CMOS circuits, bridgings cannot be simply modeled as wired functions as done in Ref. [21, 22]. In fact, with CMOS macro-gates, the conductances to be considered are, in general, strongly dependent on the applied test-pattern and the faulty behavior can change from wired-or to wired-and and viceversa (see Fig. 4), furthermore, the fault in some cases may be not detectable.

Finally, the conductance of the bridging path itself may be finite and technology dependent with important consequences on fault detectability.

In all cases, current testing can be conveniently used as an alternative to the static response observation.

2.5 Delay faults

This fault model, whose importance is growing with the increasing speed of modern ICs, represents physical failures giving rise to degradation in timing performance of digital circuits without
altering the logical function [14,24,25,26]. Delay faults can be detected as transitions (at the input of flip-flops or at POs) taking place after a sampling instant determined by a clock phase.

In the case of CMOS circuits, delay faults may be typically due to failures lowering the conductance of transistor paths [14,26]. In addition, transistors stuck-on and bridgings (even if not detectable from a static point of view) may slow signal transitions [16], thus giving rise to delay faults.

At logic level, two different delay fault models have been considered [27,28]. The first, called gate delay fault model, consists of an increase in the rise or fall time of transitions occurring at gate inputs and outputs. The second, denoted path delay fault model, describes a increased propagation delay in a circuit path. This latter, however, is unpractical because, in general, the number of possible propagation paths is exceedingly high.

The detection of a delay fault requires two test vectors, the first initializing the circuit and the second originating a transition (as shown, for a simple case, in Fig. 5).

![Figure 5: Example of a gate delay fault consisting of a slow-to-rise fault on input line A of a CMOS NAND gate (for instance, due to a poorly conducting transistor in the pull-down network). Simplified waveforms of a test set detecting the delay fault are illustrated.](image)

The detection of a delay fault depends on the time at which delayed transitions occur at the inputs of the sampling circuitry. This, in turn, depends on the delays of the sensitized paths and on the size of the delay fault.

In particular, the possibility that an increase in the gate propagation delay produces a detectable delay fault depends on the sum of all delays along sensitized paths from primary inputs to the sampling logic through the faulty gate.

A delay fault test may be invalidated by hazards. If the detection of a delay fault is independent of circuit delays and insensitive to hazards phenomena, the corresponding test set is called robust [13,29,30].

With regard to CMOS circuits, existing fault models are able to represent delay faults only in primitive CMOS gates (NAND, NOR, NOT), but are unable to take into account the case of circuits realizing more complex logic functions. In addition, the dynamic effects of faults such as transistors stuck-on and bridgings are not yet conveniently modeled.

## 3 Fault Simulation

In this section we will consider the problems related to the simulation of the non stuck-at faults introduced in the last section.

In general, the target of advanced fault simulation is to achieve a realistic and reliable evaluation of fault coverage and this requires the use of a sufficiently complete set of fault models. Furthermore, the requirements of calculation efficiency implies severe constraints on the complexity of the considered models since the CPU time increases with the degree of detail introduced in the circuit description [1].

From a practical point of view, it must be stressed that, while several satisfactory techniques exist for fault simulation, up to 10,000 of gates, for larger circuits these become inadequate and satisfactory approaches are still to be found.

In perspective, hierarchical techniques should be used in order to simulate with a good level of accuracy only the parts of large circuits needing detailed descriptions. So far, however, little progress has been done in this direction.

As known, fault simulation algorithms, were originally developed at gate-level to study the detection of stuck-at faults as steady state errors at POs. Since the elementary approach of repeating a simulation for each fault leads to high simulation costs, special attention was particularly devoted to reduce the number of calculations to be performed. In particular, the parallel algorithms exploit the machine parallelism to simulate several faulty circuits at the same time [31,32], while deductive methods [33] and concurrent methods [34] take advantage of simulating only the differences between faulty and fault free circuits; finally the critical path method performs an analysis of the fault free circuit without explicitly simulating individual faults [35].

In general, the cost of fault simulation is quadratic in the number of gates [36]. All the algorithms mentioned above, in principle, may be extended to the case of CMOS circuits although, of course, this implies the implementation of the fault models specific of this type of circuits.

### 3.1 Fault simulation of transistor faults

Let us first consider the study of the detection of transistor faults (stuck-open and stuck-on) that is usually performed by simulating the steady state response of the circuit. At this regard, two different approaches can be taken, according to whether transistor faults are:

1. implicitly considered using gate-level simulators;
2. treated explicitly by using switch-level simulation.

Furthermore, in general, in order to correctly model transistor stuck-open faults, memory effects related to persistence of voltages at high impedance nodes must be taken into account. In the case of transistors stuck-on, instead, conductance conflicts must be evaluated.

Early works [9,37] model transistor faults as stuck-ats in extra circuitry that must added to each CMOS gate (including flip-flops to model the sequential behavior due to transistors stuck-open). This approach, although attractive since it allows the use of conventional stuck-at fault simulation, is unsatisfactory in large VLSI circuits, because of added logic blocks becomes excessively large.

As an alternative, a straightforward technique has been presented [38] to derive gate-level equivalent networks of Fully Complementary CMOS (FCMOS) gates implementing logic functions more complex than elementary NAND, NOR functions presenting the property that stuck-at faults at their inputs are equivalent to transistor faults in the original circuit.

To avoid inefficient transformations required by the two approaches mentioned above, a tree structure can be used [39] to represent FCMOS gates during fault simulation. Such a structure allows to implicitly consider transistor faults and to use techniques (such as fault-collapsing) to reduce the computation time.

All these techniques, although retaining the efficiency of gate-level simulation, present some drawbacks.
3.3.3 Detectability of stuck-at and non-stuck-at faults
In order to detect any fault at POs the circuit line at which the faulty signal originates must be observable. Since this condition coincides with that required for stuck-at detection, a linkage can be established between the detection of stuck-at and more general types of faults.

For this reason, it is interesting to study the relationship between coverages on stuck-at and non-stuck-at faults with regard to different stuck-at oriented test generation methods (deterministic and random) [40]. This, in particular, because in practice only the stuck-at fault model is normally used in test sequence generation and evaluation.

More specifically, it is interesting to evaluate the coverage of non-stuck-at faults obtainable with stuck-at oriented test pattern in order to decide whether or not specific treatment of CMOS faults is advisable.

In particular, it has been shown [40] that deterministic stuck-at oriented test sequences provide unsatisfactory coverages (Fig. 6) on transistors stuck-open. In this respect, random test sequences achieve significantly better results (Fig. 7). This is due to the fact that, as already mentioned, the detection of stuck-open faults requires at least a suitable sequence of at least two test vectors and such a sequence is more easily found with randomly generated test patterns that, for the same value of stuck-at coverage, are normally significantly longer. In any case, the coverage of stuck-open faults obtained with stuck-at oriented test sequences is normally rather low. Therefore, this kind of fault should be explicitly taken into account in test generation and fault simulation.

The coverage of transistors stuck-on, instead, is intrinsically low because of the presence of several undetectable faults.

Finally, in the case of bridgings without feedback, both deterministic and random test vectors may provide unsatisfactory fault coverages (see Fig. 8 and Fig. 9) essentially because of the presence of undetectable faults.

3.3.4 Performance
In order to evaluate the CPU time required for fault simulations including the non stuck-at faults discussed in this paper, we have considered a typical combinational circuit implemented with FC-MOS macro-gates (NANDs, NORs, and various kinds of AND-OR-INVERTER or OR-AND-INVERTER blocks). The average CPU time ($t_{FS}$) for each test vector needed to perform a fault simulation without fault dropping, using the macro-gate level fault simulator described in [41], is 2.2 than that required when only stuck-at faults are considered. In our case with a circuit featuring 967 macro-gates $t_{FS}$ is 54ms on a SUN-Sparcstation.

Since it is more general, switch-level fault simulation presents a substantial increase of computational complexity (mainly due to the explicit consideration of transistors). In particular, it gives rise to simulation times about 7-8 times longer that those needed for the macro-gate level fault simulation. On the other hand, switch-level fault simulation can consider circuits that cannot be treated at gate-level. As a consequence, of a mixed level approach, combining switch and macro-gate descriptions should be ideally suited for fault simulation of complex CMOS circuits.
A first significant limitation is that the correct evaluation of 
conductions (required to deal with transistor stuck-on faults) is not performed. This latter problem has been approached at 
gate and macro-gate level [40,41], where by implicitly considering the transistor level implementation of FCMOS gates, the conductions of conflicting networks originated by transistors stuck-on are correctly compared.

In addition, an important drawback is that these techniques are not suitable to model typical CMOS circuits such as, for instance, those based on dynamic effects. To deal with these cases switch-level simulators, either based on a boolean algebra [42] or on lattices [43], have been developed. With this approach, digital circuits are described as a set of interconnected transistors, each considered as a digital switch with a “strength” belonging to a finite set of values providing a qualitative description of the transistor conductance. Furthermore, capacitances related to circuit nodes are considered in order to deal with dynamic effects.

In this way, the circuit steady state response can be evaluated for a very large class of CMOS circuits; furthermore, switch-level simulation is ideally suitable to treat transistors faults. For this reason, fault simulators for CMOS circuits have been developed extending the concurrent technique to switch-level [44,45].

Switch-level simulation, however, presents some problems when considering faults such as transistors stuck-on and bridgings in which conductions have to be carefully evaluated and compared.

This difficulty is due to the fact that switch-level algorithms usually perform only a qualitative analysis of conductions (for instance by assuming the conductance of the parallel of conductive devices to be equal to the largest individual contribution), that is normally sufficient in the case of fault-free circuits, but may lead to wrong conclusions in the presence of faults. To overcome this problem, switch-level techniques specifically oriented to fault simulation have been developed [46,47].

As for computation time, switch-level simulation is, of course, more complex than that carried out at gate-level, consequently, it is also necessarily less efficient.

### 3.2 Fault simulation of bridging faults

Differently from transistor faults, some type of bridging faults, in particular bridgings between gate output lines, have been considered even in technologies (for instance nMOS) in which the behavior of faulty circuits may be described in terms of wired-and/or functions. In CMOS circuits, however, bridgings can give rise to a behavior of the shorted network that changes from wired-and to wired-or (or viceversa) depending on the used test vector, thus invalidating results based on the fixed wired-and/or assumptions [21,22]. In this case too, the problem can be solved only by evaluating and comparing the output conductions of shorted gates [40,41].

The particular (but likely) case of shorts between transistor gate and drain has been treated [40] with an approach based on considerations derived from electrical simulation that has been shown a good degree of detectability for this kind of faults.

As for the more general type of bridgings, those between gate outputs and transistors terminals may be treated both at gate- or switch-level [46], while the more general case of shorts between gate internal nodes or between nodes belonging to different gates must necessarily be studied at switch-level [44,46].

### 3.3 State of the art in the simulation of transistors and bridging faults in CMOS ICs

This section presents some of the more relevant points about transistors stuck-open, transistors stuck-on and bridgings, whose detection can be studied by means of the simulation of the steady state response of the circuit.

To this regard, before considering specific points, Table 2 provides a general overview of the classes of circuits that can be treated and the faults that can be simulated for each simulation level.

<table>
<thead>
<tr>
<th>Fault simulation level</th>
<th>Circuits</th>
<th>Faults</th>
</tr>
</thead>
<tbody>
<tr>
<td>gate</td>
<td>NAND, NOR, NOT</td>
<td>stuck-open bridgings</td>
</tr>
<tr>
<td></td>
<td></td>
<td>stuck-on bridgings</td>
</tr>
<tr>
<td>macro-gate</td>
<td>FCMOS</td>
<td>stuck-open bridgings</td>
</tr>
<tr>
<td></td>
<td></td>
<td>stuck-on bridgings</td>
</tr>
<tr>
<td>switch</td>
<td>general CMOS</td>
<td>stuck-open bridgings</td>
</tr>
<tr>
<td></td>
<td></td>
<td>stuck-on bridgings</td>
</tr>
</tbody>
</table>

Table 2: Simulation levels required to treat transistors and bridging faults in CMOS circuits (the number in the last column used for bridging faults refer to the points discussed in section 2.4).

#### 3.3.1 Advances in the accuracy of fault modeling

In order to increase the reliability of evaluations of fault coverage with regard to transistors stuck-on and bridgings, it is necessary to model faulty devices whose conductance is different from that of normal transistors as well as and extra conductive paths with finite conductance [41].

In addition, a probabilistic fault model has been developed [48] to solve problems due to the simple threshold comparison of conflicting conductances that may lead to unrealistic fault coverages. Such an approach, in fact, takes into account realistic statistical variations of transistor parameters (both fault-free and faulty) and bridgings.

#### 3.3.2 Fault simulation for current testing

As already mentioned, a possible method to detect faults giving rise to conductive paths between power supply and ground (i.e. transistors stuck-on and bridgings) consists of measuring the increase of static currents with respect to the fault-free circuit.

In order to evaluate the coverage obtainable with current testing, it is necessary to simulate the steady state behavior of the circuit.

To this purpose, the fault simulation techniques illustrated at points 3.1 and 3.2 can be used, although, of course, the propagation of fault effects toward POs is not required and this significantly simplifies the computation. In addition, the intermediate voltage of faulty signal nodes is of no interest; nevertheless, conductances should be carefully taken into account in order to identify the conditions (test vectors) able to maximize the amount of current drawn.
gate-level equivalent circuit of the macro-gate represent a drastic although, recently, an attractive approach to the problem of delay only informations about the steady state response of the circuit, modeling at switch-level (in the case of fault-free circuits) has been form delay fault simulation, in the case of ChlOS circuits some including pass-transistors is not yet available.

The simulation of failures degrading the timing performance necessarily requires accurate modeling of the circuit dynamical behavior. In order to contain computation time, such type of simulation is normally carried out at logic level by means of algorithms that starting from a gate model describing propagation delays, evaluate the timing of signals propagating inside the circuit (in a way similar to time-based logic simulation [49]).

In essence, this simulation allows to determine whether or not the transition activating the considered delay fault is propagated toward an observable circuits line through a sensitized path. This can be achieved using transition fault simulation techniques [50] that perform only a qualitative analysis of the circuit dynamic behavior in order to determine if the activated transition is correctly propagated toward latches and/or POs. As an alternative, more sophisticated delay fault simulation techniques can be used in order to obtain also a quantitative evaluation of the size of detected faults [13,14,29,51].

Although, in general, several efficient techniques exist to perform delay fault simulation, in the case of CMOS circuits some specific problems have to be solved.

In particular, the modeling of delays is sufficient in the case of standard gates (NAND, NOR, NOT), but a realistic and accurate approach to the case of CMOS complex (macro-) gates and circuits including pass-transistors is not yet available.

The main problem at this regard is that in the case of macro FCMOS gates, the delay of an output signal (in the fault-free circuit) in general depends on the state of all transistors in the network involved in the signal transition. For this reason, assigning a predefined value to such a delay at each terminal of a gate-level equivalent circuit of the macro-gate represent a drastic simplification that and may lead to serious errors. Unfortunately, existing switch-level simulations [42,43] provide only informations about the steady state response of the circuit, although, recently, an attractive approach to the problem of delay modeling at switch-level (in the case of fault-free circuits) has been presented [52].

4 Conclusions

As shown in this paper, fault simulation of CMOS circuits cannot be limited to the use of the simple stuck-at model because this does not adequately represent a significant fraction of the failures encountered in reality. In particular, CMOS circuits exhibit faults that produce a sequential behavior even in combinational blocks and others that produce intermediate signal voltages (due to the presence of conductive paths between power supply and ground). In addition, partly because of the type of faults just mentioned, CMOS circuits may present a degradation of their dynamic performance possibly resulting in the so called delay faults.

In practical applications, however, only stuck-at oriented test vectors and simulators are used and this leads to significant over-estimations of the reliability of testing procedures.

At research level, a considerable effort has been done in order to develop adequate models for non stuck-at faults and to implement these within simulators solving some specific problems such as, for instance, the accurate modeling of complex conflicting networks. The results of such efforts are a set of tools that, although, undoubtly interesting, still presents serious limitations in terms of accuracy and/or computation efficiency.

Therefore, further work is needed, particularly in the direction of hierarchical simulators able to treat parts of the circuits at different abstraction level so that when some blocks are treated with great detail (for accuracy), the largest part of the circuit can be simulated very efficiently.

References
