Test Time Reduction in Analogue/Mixed-Signal Devices by Defect Oriented Testing: An Industrial Example *

Hamidreza Hashempour[†] **Jos Dohmen**[†] **Bratislav Tasić**[†] **Bram Kruseman**[†]
Camelia Hora[†] **Maikel van Beurden**[†] **Yizi Xing**[‡]

Abstract

We present an application of Defect Oriented Testing (DOT[1]) to an industrial mixed signal device to reduce test time and maintain quality. The device is an automotive IC product with stringent quality requirements and a mature test program that is already in volume production. A complete flow is presented including defect extraction, defect simulation, test selection, and validation. A major challenge of DOT for mixed signal devices is the simulation time. We address this challenge with a new fault simulation algorithm that provides significant speedup in the DOT process. Based on the fault simulations, we determine a minimal set of tests which detects all defects. The proposed minimal test set is compared with the actual test results of more than a million ICs. We prove that the production tests of the device can be reduced by at least 35%.

1 Introduction

Test cost for analogue and mixed-signal (AMS) products is a growing concern within the industry [1]. One of the reasons is the long test times due to the growing complexity in combination with specification (or performance) based testing. Together with expensive instruments, these time consuming tests take a larger fraction of the production cost of an IC [2][3]. Techniques such as multisite testing have been applied with success to reduce test time by sharing instruments among multiple devices under test (DUT), however, they are running out of gas and novel directions need to be explored.

While structured test methods have become common practice in digital ICs, testing AMS ICs in a structured way is still in infancy [4]. Lack of fault models for AMS ICs renders fault oriented approaches difficult and sometimes fruitless. AMS DFT and BIST techniques are also lagging. According to [3] "no proven alternative to performance-based analogue testing exists and more research in this area is needed".

1.1 Previous works

DOT has been proposed as a structured test method for AMS ICs. While details vary slightly, it generally includes defect extraction, fault modeling, fault simulation, and test application. Defects are either extracted from layout by Inductive Fault Analysis (IFA) [5] or by processing netlist or schematic. Fault models range from simple resistive models for bridge and open defect mechanisms [6] to more sophisticated models such as *S*-parameters for frequency domain analysis [7]. Fault simulation involves injecting faults into the netlist of the AMS IC and performing analogue simulation (DC, AC, or Transient).

DOT has not been widely used though, a major drawback with DOT is the long time associated with analogue fault simulation. A technique for trading accuracy for speedup is presented in [2] in which a mixed-mode co-simulation of HDL code and SPICE models has been proposed for fault simulation. A similar technique that uses high level behavioral modeling is presented in [8]. Fast fault simulation of analogue circuits using specialized analogue simulators have been intensely studied in the past. Analogue fault simulation in linear circuits is studied in [9][10]. The work of [11] presents an example of fault simulation in nonlinear circuits when DC point of the faulty and fault-free circuits are equal. A relaxation technique for fast calculating DC point of nonlinear faulty circuits is discussed in [12]. Examples of fast fault simulators for transient analysis in linear circuits are DRAFTS and FLYER [13][14]. Fast fault simulation for transient analysis in nonlinear circuits has also been proposed [15][16].

Computer aided test environments for AMS testing have also been the subject of research and development. They encapsulate tools (e.g., fault extractor or analogue simulator) and facilitate execution of the DOT flow [17][18]. We use NXP's internal tool DOTSS (defect oriented test simulation system) in this work [19]. Reducing production test size has been studied in [20] where analogue fault mod-

*This work is carried out as part of the Catrene project "TOETS" [CT302] and is supported by the Dutch Ministry of Economic Affairs.
978-3-9810801-7-9/DATE11/©2011 EDAA.
[†]NXP Semiconductors, Central R&D, Foundation Technology, Design Methods, Physical Design Methods, Eindhoven 5656AE, The Netherlands.
[‡]NXP Semiconductors, BU Automotive, BL Automotive AMS, Nijmegen, 6534AB, The Netherlands.
[1]Also called Defect Based Testing, DBT.

eling is discussed followed by an analysis of specification tests and algorithms for fault-driven test selection. Reference [21] highlights redundancy within the performance tests of an RF device and proposes prediction models based on genetic algorithms to select a subset of tests.

1.2 Contribution

In this paper, we present a test time reduction technique and its application to an AMS IC that is already in volume production. The device is a CAN (controller area network) transceiver manufactured in a BiCMOS-DMOS (BCD) process. It is an automotive product and has a zero-defect quality requirement.

The proposed technique utilizes DOT; it extracts and simulates defects using production tests. The DUT netlist has 31,587 elements of which 16,859 are transistors. Although it is small compared to large System-on-Chip designs, it is quite challenging from a defect simulation point of view. For example, if simulating hundreds of tests takes one hour, it still takes several years to perform these simulations for 10,000 defects. We propose a novel analogue fault simulation algorithm that reduces time to weeks or days. Based on the simulation results, we determine a subset of tests that covers all detectable defects. This approach can be used on new DUTs to estimate and improve fault coverage as well as existing DUTs to determine a minimal test set and improve test time. We apply the latter approach. To validate that the minimal subset performs well, the production test data logs of 10 lots (over a million DUTs) are used to evaluate how many failing dies would have escaped if the selected tests were used instead of the original (full) set. We discover that the production tests can be reduced by 35% while maintaining quality. To our best knowledge, this is the first publication in which the DOT method is applied to a volume production AMS IC. The DOT flow is presented in Figure 1. It has 4 main steps, defect extraction, defect simulation, test selection, and validation.

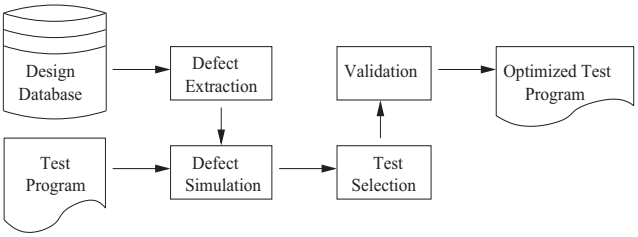


Figure 1: Basic steps in the DOT flow.

The rest of this paper is organized as follows: Section 2 presents details of the defect extraction in the DOT flow. Section 3 discusses the details of our fast simulation algorithm. Section 4 presents details of the test selection including combinatorial formulation and the utilized heuristics. Section 5 presents how the selected tests are validated by the production test data followed by Conclusions in section 6.

2 Defect Extraction

Different defect mechanisms can occur in the DUT. According to the production test data, bridge defects are the most dominant one followed by dislocation defects. Open defects are less likely in this technology so they are not used in the current flow.

To identify bridge defects, an IFA like approach is utilized; the DUT design database (layout) is passed through a parasitic capacitor extractor and the capacitor list is used in DOTSS to enlist bridge defects together with their relative likelihood. For dislocation defects, the cross section of library models that contain PN junctions are studied and the potential dislocation candidates are determined for each model. A script parses the DUT netlist and when it finds an instance whose model has potential dislocation defects, writes them into a dislocation defect list. We extracted about 38,000 bridge defects and about 1,500 dislocation defects. This paper deals with the 38,000 bridge defects, however, work is in progress to include dislocation and open defects in the near future.

3 Fault Simulation

The extracted defects are modeled as faults and injected one by one in the netlist of the DUT. The fault model is a simple resistor whose value is set according to the defect class (bridge) and the process information. The faulty netlists are then simulated. The full-chip analogue simulation of about 38,000 faulty netlists can be extremely time consuming. We use a novel fast simulation algorithm called *fault sensitivity analysis (FSA)* that is embedded within PSTAR[2]. We are also discussing this methodology with EDA vendors for its implementation within commercial tools such that a wider community can utilize it. We also use other internal tools of NXP (AnalogueShell and DOTSS) to facilitate handling the large volume of simulation data corresponding to 38,000 defects and 197 production tests as well as fault injection and simulator launches.

3.1 Fault Sensitivity Analysis

Fault sensitivity analysis (FSA) is a novel methodology for establishing the detection status of defects in the DOT flow. FSA allows very efficient analogue fault simulation with reasonable accuracy for transient analysis in nonlinear circuits. FSA is based on two features of fault simulation in DOT: 1) establishing the detection status of a fault is more important than the actual faulty circuit output value, and

[2]The in-house analogue simulator of NXP.

2) the output value is often measured at few time points compared to many time points of a transient simulation.

Figure 2 (a) shows time points for fault simulation of a given bridge with known resistance (therefore known conductance $G = \frac{1}{R}$). The transient simulation includes many time points but the test measurement points are only few as shown in Figure 2 (b). The idea is to calculate the faulty circuit output only at the measurement time points.

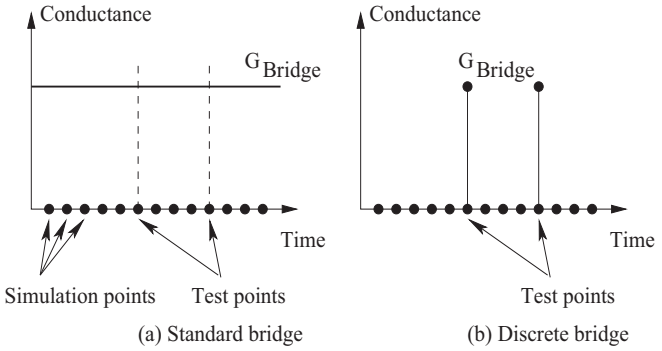


Figure 2: Time points reduction: (a) simulation calculates circuit at each and every time point (b) test only needs circuit results at the two (generally few) measurement points.

The question now is how to calculate the faulty circuit output at only the measurement points and avoid the rest of time points? This is achieved by using a discrete bridge model instead of the standard bridge model. The discrete bridge model has zero conductance for all time points, except for the test points; it is assumed that the discrete bridge influences the circuit only at the test points.

Hence, for all time points, a standard transient analysis of the golden circuit is performed and for the test points an additional analysis is performed for each fault. This analysis repeats the golden time step for the test point and reuses the golden data of the previous time point. The faulty circuit solution V^{fault} is computed from

$$\mathbf{F}(\mathbf{V}^{fault}) + \mathbf{G}_{fault}\mathbf{V}^{fault} = 0 \qquad (1)$$

where the golden solution satisfies the nonlinear equation $\mathbf{F}(\mathbf{V}^{golden}) = 0$ and the linear term $\mathbf{G}_{fault}\mathbf{V}^{fault}$ is the contribution due to the conductance of a fault ($\mathbf{G}_{fault}$). Equation (1) is solved using the Newton-Raphson method, which can be expressed as

$$\mathbf{V}_{i+1} = \mathbf{V}_i - (\mathbf{J}_i + \mathbf{G}_{fault})^{-1}(\mathbf{F}_i + \mathbf{G}_{fault}\mathbf{V}_i) \qquad (2)$$

where i is the iteration index and $\mathbf{J}_i$ is the Jacobian of $\mathbf{F}$ at $\mathbf{V}_i$. The golden solution is used as a start of the Newton-Raphson process and therefore the first iteration approximates the solution by linearizing the faulty circuit at the golden value $\mathbf{V}^{golden}$.

Figure 3 (a) illustrates eq. (1) in a graphical way. The faulty solution V_{out} is symbolically plotted versus the fault

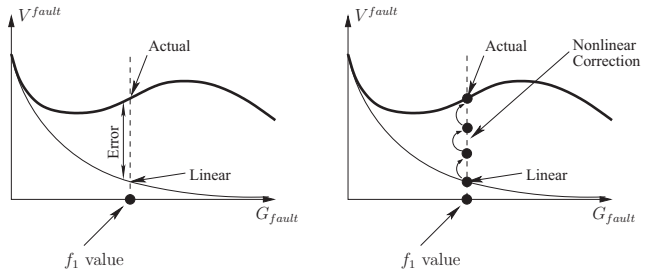


Figure 3: Fault Sensitivity Analysis: (a) Linear (b) Nonlinear

conductance. The bold curve points to the circuit behavior as a function of fault conductance given by eq. (1) and the thin curve is the behavior of the linearized faulty circuit. The value of the thin curve at the conductance of a given fault f_1 is referred to as the linear FSA result of f_1. The error of the linear FSA is also shown. The subsequent Newton-Raphson iterations are corrections of the nonlinear FSA which will eventually converge to the actual solution V^{fault} by applying eq. (2). An impression of the nonlinear FSA iterations is presented in Figure 3 (b).

Table 1 compares the standard (non-FSA) simulation with linear as well as nonlinear sensitivity analysis by applying different number of iterations. The FSA provides at least 173x speedup (corresponding to the nonlinear FSA with 10 iterations) over the standard simulation. These results are obtained for 38,000 defects and 75 tests with only one CPU used.

Table 1: Fault Sensitivity Analysis (FSA) performance

Simulation Methodology	Speedup	Wall clock time
Standard	1	24,228h (1010 days)
Linear FSA	1050	23h (1 day)
Nonlinear FSA (1 iteration)	515	47h (2 days)
Nonlinear FSA (5 iterations)	238	102h (4.2 days)
Nonlinear FSA (10 iterations)	173	140h (5.8 days)

FSA performs well for both catastrophic and non-catastrophic faults. In the former case, the impact is often large thus detection status is identified correctly with FSA.

3.2 FSA Accuracy

FSA aims to predict the outcome of fault detection correctly rather than the actual faulty value. Deviations from the actual faulty value are acceptable as long as FSA predicts the same detect status that the accurate non-FSA simulation (e.g. SPICE) would do. Nevertheless, there are cases for which FSA fails to predict the detect sta-

tus correctly. An experiment is devised to check this phenomenon: about 8,000 analogue-only defects (that is, both bridged nets are analogue nets/lines) are simulated for about 150 (of 197) tests using the standard non-FSA (accurate) simulation. The choice of defects and tests is to keep the simulation time manageable and the experiment practical. The detect statuses are stablished and checked versus those obtained by FSA. The percentage of defects whose status is correctly predicted by FSA (with reference to the standard non-FSA simulation) is called *matching* and used as a figure of merit for FSA. Figure 4 (a) presents the matching plots as a function of tests. Figure 4 (b) shows the distribution of tests versus matching. On average matching is 95.1% over all tests while about 87.4% of tests have matching more than 90%. The worst case matching is 82.92%; our investigation shows this worst case is because the nonlinear solver does not converge. Using a more robust algorithm for the nonlinear solver will improve the matching significantly.

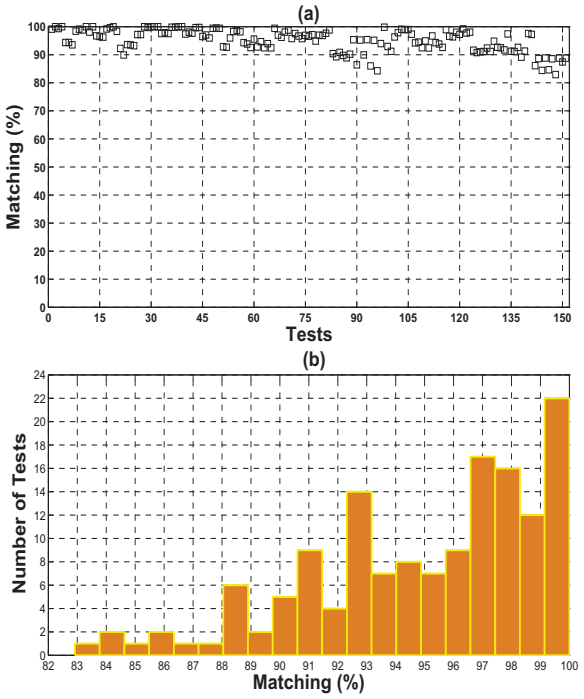


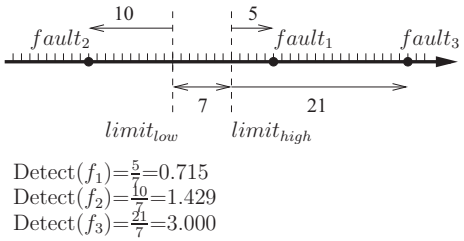
Figure 4: The matching results of FSA: (a) matching versus tests (b) histogram of the tests versus matching.

4 Test Selection

Test selection is the process by which a small (minimal) subset of the production tests is made that covers a required percentage of (detectable[3]) defects. Since the subset is smaller than the original set, test reduction and subsequently test time reduction can be achieved. The percentage of the covered defects is user adjustable, however by default, the flow covers all detectable defects (100% defect coverage).

[3]A defect that is detected by at least one test.

Input to the test selection is a detection matrix whose entries specify if a test (column) detects a defect (row) [22]. In [22] the entries are either 0 or 1 (undetect or detect) due to the digital nature of the DUT and its tests. In this work however, the entries are continues numbers ≥ 0. For each test there are two limits: if a test result falls within the limits, it is assumed passed, else failed. Similarly, if a test simulation result falls between limits, it is assumed undetected (0); otherwise detected (>0). The detection matrix entries show how far the simulation or measurement results are from the test limits, see Figure 5.



Detect(f_1)=$\frac{5}{7}$=0.715
Detect(f_2)=$\frac{10}{7}$=1.429
Detect(f_3)=$\frac{21}{7}$=3.000

Figure 5: Calculation of detection matrix entries for a test and three faults.

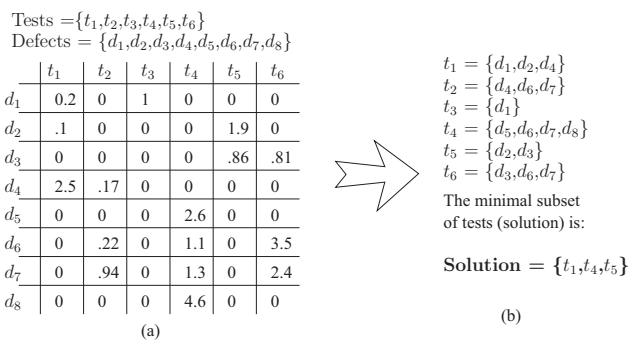
The larger the entry is, the better is the detection; for example, 0.715 indicates a weaker detection compared to 1.429. The user can also set a threshold to control how strong of detections he wants to include in the selection process. This is useful in accounting for process variations because out of limit measurements do not necessarily point to a faulty DUT, the DUT could be suffering from process variations. An example of the detection matrix is presented in Figure 6.

	$Test_{121}$	$Test_{122}$	$Test_{123}$
$Defect_{23}$	2.44	0.295	0.19
$Defect_{24}$	0.182	0.93	0.2
$Defect_{25}$	1.27	4.282	0.78
$Defect_{26}$	0.23	0.52	0.01

Figure 6: Example of detection matrix: detection of 'Defect 25' by 'Test 122' is a lot stronger than 'Defect 26' by the same test.

The detection matrix is mapped to an instance of *set-covering* combinatorial problem. In set-covering several sets are given as input that may have common elements. One must select a minimal number of these sets so that the selected sets contain all the elements that are in the union of the input sets. In the mapping, tests represent input sets and the defects they detect are the set elements (some defects may be detected by multiple tests). We must select a minimal number of tests (sets) such that they cover all defects detectable by the original tests. An example of test selection is shown in Figure 7.

Set-covering is an NP-Complete problem but heuristics



	t_1	t_2	t_3	t_4	t_5	t_6
d_1	0.2	0	1	0	0	0
d_2	.1	0	0	0	1.9	0
d_3	0	0	0	0	.86	.81
d_4	2.5	.17	0	0	0	0
d_5	0	0	0	2.6	0	0
d_6	0	.22	0	1.1	0	3.5
d_7	0	.94	0	1.3	0	2.4
d_8	0	0	0	4.6	0	0

Figure 7: Example of test selection: (a) 6 tests and 8 defects are given according to the detection matrix (b) a selection of t_1, t_4, and t_5 covers all 8 defects and reduces tests from 6 to 3, i.e., 50% test reduction.

exist to solve it efficiently. We use a greedy algorithm [23] that produces reasonably good results in our case. At each step, the heuristic selects the test that detects most of the defects and then removes those defects from consideration for the next step. It iterates until no defect is left to be detected. The heuristic is presented in Algorithm 1.

$T \leftarrow$ Tests; `/* original tests */`
$D \leftarrow$ Defects; `/* detectable defects */`
$S \leftarrow \emptyset$; `/* Minimal set, initially empty */`
while $D \neq \emptyset$ **do**
  $t \leftarrow$ the test that detects most defects;
  $S \leftarrow S \cup \{t\}$;
  $T \leftarrow T - \{t\}$;
  $D \leftarrow D-$ defects detected by t;
end

Algorithm 1: Selects the minimal test set S.

Test selection may include additional constraints; for example, some of the analogue tests are customer required specification tests and cannot be removed. The test selection must ensure such tests are always included (selected). Their presence means that certain defects are always covered; such defects can be removed prior to the selection process. A revision of Algorithm 1 that covers the aforementioned constraint is presented in Algorithm 2.

Often the analogue tests are not completely independent, some represent the same test applied at a different supply voltage. This highlights another important constraint during the test selection: to ensure at least one supply condition of any test is included in the selected test set. Effectively, it is not allowed to remove all supply conditions of a test. Algorithm 3 presents an extension of Algorithm 2 in which the above constraint is implemented. The extension at the end simply checks if there are tests without any supply condition and if so selects one of the conditions according to the most frequently used one. This feature minimizes the number of supply voltage switches during the production testing.

$T \leftarrow$ Tests; `/* original tests */`
$D \leftarrow$ Defects; `/* detectable defects */`
$T_s \leftarrow$ Spec Tests; `/* $T_s \subseteq T$ */`
$S \leftarrow \emptyset$; `/* Minimal set, initially empty */`
$T \leftarrow T - T_s$;
while $T_s \neq \emptyset$ **do** ; `/* Remove defects detectable by spec tests */`
  $t \leftarrow$ any test of T_s;
  $T_s \leftarrow T_s - \{t\}$;
  $D \leftarrow D-$ defects detected by t;
end
while $D \neq \emptyset$ **do**
  $t \leftarrow$ the test that detects most defects;
  $S \leftarrow S \cup \{t\}$;
  $T \leftarrow T - \{t\}$;
  $D \leftarrow D-$ defects detected by t;
end

Algorithm 2: Selects the minimal test set S and preserves spec tests.

Algorithm 2(T, D, T_s, S); `/* Run Algorithm 2 */`
$c \leftarrow$ the supply condition that appears most in S;
for $t \in T$ **do**
  if *no condition of* $t \in S$ **then**
    $S = S \cup \{t_c\}$; `/* select condition` c `of` t `*/`
  end
end

Algorithm 3: Selects the minimal test set S, preserves spec tests, and ensures at least one supply condition of any test is selected.

The production tests of our vehicle DUT include digital, analogue, and IDDQ tests. Non-digital tests (analogue and IDDQ) dominate the test time so we focus on the 197 such tests. Included in the 197 tests are up to 3 voltage supply conditions.

5 Validation

During validation, the production test data logs of 10 lots, containing 1.3 million dies, are analyzed. In this analysis continue-on-fail[4] is used, hence results for all tests are stored. All tests detect some defects although their efficiency varies widely. Some tests detect more than half of the rejected devices while others catch far less than 1% of the rejected ICs. Most of the tests are performed at all the 3 supply voltages. At each of them, we find unique detects. Hence, test time reduction is not as trivial as determining the best voltage setting to improve detectability.

[4]The test equipment does not stop after the first failing test and continues test application until the last test.

The analysis procedure to validate the algorithms (selections) has two steps: 1) apply the test program limits for 197 original tests and set the pass/fail boundary for all 1.3 million dies; 2) remove the dies that fail at least one test in the selected test set; any remaining die that fails any of the original tests escapes the selected tests (would-escape-die). The analysis results for the three selections associated with Algorithm 1, 2, and 3 are presented in Table 2. The number of dies that escape if the selected tests are used as well as test reduction and test time reduction figures are presented. The last column presents the actual test time reductions. This can be different from the test reduction since some tests can take longer to apply than others.

Table 2: Would-escape-dies versus selection algorithms

Selection	Escapes	Test Reduction	Test Time Reduction
Algorithm 1	14	46% (107/197)	37.3%
Algorithm 2	6	41% (116/197)	33.2%
Algorithm 3	**0**	**35% (128/197)**	**27.2%**

As seen, 14 and 6 dies would escape selection 1 and selection 2 respectively but selection 3 guarantees no escapes. Normalized with respect to the 1.3 million dies, the defect level (ppm) figures are at 11, 4.6, and 0 ppm for the three selections respectively. It is noteworthy that a would-escape-die does not necessarily mean a customer return. Such dies may suffer from process variation or only marginally fail the test.

6 Conclusions

A DOT approach is utilized to optimize production tests of an AMS automotive product with stringent quality requirements. The results are promising: using a fast analogue simulation algorithm enables us to simulate an extensive set of defects. Three selection algorithms are proposed and evaluated. These algorithms enable us to reduce the number of tests by 35% to 46%; corresponding to the actual test time reduction of 27% to 37%. Algorithm 3 is capable of detecting all outliers and fits the zero-defect automotive requirements. The other two algorithms result in more test time improvement but will miss a number of outliers, which are seen as a potential quality risk for automotive products.

References

[1] F. Poehl, F. Demmerle, J. Alt, and H. Obermeir, "Production Test Challenges for Highly Integrated Mobile Phone SoCs - A Case Study", *Proc. of European Test Symp.*, pp. 17-22, 2010.

[2] J. Park, S. Madhavapeddi, A. Paglieri, C. Barr, and J. A. Abraham, "Defect-Based Analog Fault Coverage Analysis using Mixed-Mode Fault Simulation," *Proc. 15th IEEE International Mixed-Signals, Sensors, and Systems, Test Workshop*, pp. 1-6, 2009.

[3] "The International Technology Roadmap for Semiconductors, Test and Test Equipment," Edition 2009.

[4] L. Fang, Y. Zhong, H. V. D. Donk, Y. Xing, "Implementation of Defect Oriented Testing and ICCQ testing for industrial mixed-signal IC," *Proc. IEEE Asian Test Symp.*, pp. 404-409, 2007.

[5] J. Chen, W. Maly, and F. Fergusen, "Inductive Fault Analysis of MOS integrated circuits," *IEEE Design and Test of Computers*, Vol. 2. No. 6, pp. 13-26, 1985.

[6] T. Olbrich, J. Perez, I. A. Grout, A. M. D. Richardson, and C. Ferrer, "Defect-Oriented vs. Schematic-Level based Fault Simulation for Mixed-Signal ICs," *Proc. IEEE Intl. Test Conf.*, pp. 511-520, 1996.

[7] E. Acar and S. Ozev, "Defect-Oriented Testing of RF Circuits," *IEEE Tran. on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 27. No. 5, pp. 920-931, 2008.

[8] L. Fanq, G. Grounthoud, and Hans G. Kerkhoff, "Reducing Analogue Fault-Simulation Time by Using High-Level Modeling in DOTSS for an Industrial Design," *Proc. IEEE European Test Symp.*, pp. 61-67, 2001.

[9] G. C. Temes, "Efficient Methods of Fault Simulation," *Proc. Midwest Symp. on Circuits and Systems*, pp. 191-194, 1977.

[10] A. Pahwa and R. A. Rohrer, "Band Faults: Efficient Approximation of Fault Bands for the Simulation before Diagnosis of Linear Circuits," *IEEE Tran. on Circuits and Systems*, Vol. CAS-29. No. 2, pp. 81-88, 1982.

[11] M. Tian and R. Shi, "Rapid Frequency-Domain Analog Fault Simulation under Parameter Tolerances," *Proc. IEEE/ACM Design Automation Conf.*, pp. 275-280, 1997.

[12] M. Tian and R. Shi, "Nonlinear Analog DC Fault Simulation by One-Step Relaxation," *Proc. IEEE VLSI Test Symp.*, pp. 126-131, 1998.

[13] N. Nagi, A. Chatterjee, and J. A. Abraham, "DRAFTS: Discretized Analog Circuits Fault Simulation," *Proc. IEEE/ACM Design Automation Conf.*, pp. 509-514, 1993.

[14] P. Variyam and A. Chatterjee, "FLYER: Fast Fault Simulation of Linear Analog Circuits Using Polynomial Waveform and Perturbed State Representation," *Proc. VLSI Design Conf.*, pp. 408-414, 1997.

[15] Z. R. Yang and M. Zwolinski, "Fast, Robust DC and Transient Fault Simulation for Nonlinear Analogue Circuits," *Proc. IEEE Automation, and Test in Europe Conf.*, pp. 244-248, 1999.

[16] J. Hou and A. Chatterjee, "Concurrent Transient Fault Simulation for Analog Circuits," *IEEE Tran. on CAD.*, pp. 1385-1395, 2003.

[17] C. Sebeke, J. P. Teixeira, and M. J. Ohletz, "Automatic Fault Extraction and Simulation of Layout Realistic Faults for Integrated Analogue Circuits," *Proc. European Design and Test Conf.*, pp. 464-468, 1995.

[18] A. Bounceur, S. Mir, L. Rolindez, and E. Simeu, "CAT Platform for Analogue and Mixed-Signal Test Evaluation and Optimization," *Proc. IFIP VLSI Conf.* , pp. 320-325, 2006.

[19] R. H. Beurze, Y. Xing, R. V. Kleef, R. J. W. T. Tangelder, and N. Engin, "Practical Implementation of Defect-Oriented Testing for a Mixed-Signal Class-D Amplifier," *Proc. European Test Workshop*, pp. 28-33, 1999.

[20] L. Milor and A. L. Sangiovanni-Vincentelli, "Minimizing Production Test Time to Detect Faults in Analog Circuits," *IEEE Tran.on CAD.*, pp. 796-813, 1994.

[21] H-G. D. Straitigopoulos, P. Drineas, M. Slamani, and Y. Markis, "Non-RF to RF Test Correlation Using Learning Machines: A Case Study," *Proc. IEEE VLSI Test Symp.*, pp. 9-14, 2007.

[22] F. Hapke, R. Krenz-Baath, A. Glowatz, J. Schloeffel, H. Hashempour, S. Eichenburger, C. Hora, D. Adolffson, "Defect-Oriented Cell-Aware ATPG and Fault Simulation for Industrial Cell Libraries and Designs," *Proc. Intl. Test Conf.*, pp. 1-10, 2009.

[23] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, "Introduction to Algorithms," *The MIT Press, 2nd Edition*, 2002.