MISSED: an Environment for Mixed-Signal Microsystem Testing and Diagnosis

H.G. Kerkhoff

MESA Research Institute
University of Twente
The Netherlands

G. Docherty

Advantest System Engineering GmbH
Munich
Germany

Abstract

A tight link between design and test data is proposed for speeding up test-pattern generation and diagnosis during mixed-signal prototype verification. Test requirements are already incorporated at the behavioural level and specified with increased detail at lower hierarchical levels. A strict distinction between generic routines and implementation data makes reuse of software possible. A testability-analysis tool and test and DFT libraries support the designer to guarantee testability. Hierarchical backtrace procedures in combination with an expert system and fault libraries assist the designer during mixed-signal chip debugging.

1: Introduction

Investigations within several major semiconductor companies have shown that the crucial time-to-market of mixed-signal microsystems, for example in telecommunication (e.g. GSM, modems), is often slowed-down by the time required for the development of appropriate tests and diagnosis of errors during the prototyping and production phase. Currently, the development work in generating the proper analog and mixed-signal tests and diagnostics in complex mixed-signal microsystems is considered to be the major bottleneck in reducing the crucial time-to-market.

The development of these tests and diagnosis is currently a highly manual, long and tedious process. It involves much communication between designers and the relatively few test engineer(s) and is hence prone to (expensive) errors.

There are a number of obstacles that have to be removed to solve the problems found in test program generation and diagnosis.

First, it should be verified at a very early stage of the design whether or not a system is testable and offers some level of diagnostics. It is obvious to start in this case at the *behavioural* level assuming a hierarchical design style [1, 2]. Although some tools have been developed for analog systems [3], no tools are yet available for the mixed-signal case. *Second*, DFT hardware should be available to guarantee/enhance the testability and some level of diagnostics. However, in the case of analog and mixed-signal circuits the developments are still in their infancy [4]. *Third*, general wave form generation and analysis functions are required on the target tester. This is a problem, as for the analog test-signal generation no general fault model is available. Hence, specification-driven test-signal generation seems the most promising at this moment. *Fourth*, not all testers are the same; the generation of test code based on the tester to be used is required. It is therefore not surprising that recently efforts have been started to formalise the interaction between the designers and test engineers to automate this process [5].

In the diagnostic phase, all previous results are used to locate errors within a certain level of detail. We assume this level to be a cluster of around 50 transistors. Diagnostics requires access to internal nodes by means of probes, either mechanical or e.g. electro-optical. Also in this case, mixed-signal systems are clearly causing a problem, because of such factors as the measurement of currents and the existence of soft errors. Finally, guided probing and diagnostic algorithms using techniques derived from the A.I. world, especially for the analog parts, are still in their infancy and require much more research.

The integrated software tool *MISSED* is aiming at solving many of the problems previously found in mixed-signal testing and diagnosis.

The primary goal in this set of tools is to formalise the interaction between designers and test engineers, by looking closely at the current test procedures in several industries and then strengthening the interaction between design and test, thus minimising the time-to-market.

The starting point of the test philosophy used is the 'macro testing' approach as proposed by Philips [6], enhanced by the reuse of generic procedures and routines which postpone distracting issues of implementation detail until late in the design process. *MISSED* assumes a

0-8186-3930-X/93 $3.00 © 1993 IEEE

hardware infrastructure of a workstation, where the design data resides, in combination with a verification or production mixed-signal test system and an automated prober system (e.g. electron-beam).

2: Mixed-signal design and testing

2.1: The mixed-signal design flow

The *MISSED* tool set assumes use of a suitable hierarchical design methodology. Based on the required specifications of the system, an initial architecture in terms of functional blocks (macros) is entered into the design database. This approach has shown to be also a useful partitioning with regard to testing. It is referred to as ' macro testing ' [6] and is extensively used by companies such as Philips.

The behaviours of all macros are described in terms of a (hardware) description language, such as VHDL, (AHDL ?) and MAST. We used a commercial schematic entry package and an associated mixed-level, mixed-signal simulator which uses the *C-language* for describing the behaviour of all macros. At this level, no details with regard to the actual implementation have to be given. However it is also possible to use macro models if they are already available, e.g. in the form of (parameterised) analog/digital module compilers.

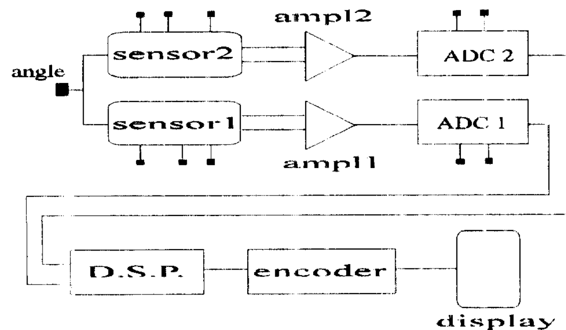

Fig.1. Schematic entry of a sensor/actuator mixed-signal micro-system as used for behavioural simulation.

As an example, figure 1 shows the schematic entry of a mixed-signal micro-system, while a part of the associated behavioural description in C is depicted in figure 2.

Based on this description, a behavioural mixed-signal simulation can be carried out, which can result in changes in the architecture or in the requirements of the individual macros. Actually, this simulation often reflects the functional behaviour of the whole system, and can hence be used at a later stage to support the semi-automatic generation of test signals for the micro-system.

The next step is that the design work is divided into sub tasks. This partitioning is often based on macros.

For the analog macros several design specialists are required, e.g. in designing filters, PLLs, amplifiers etc. They design the macros down to the layout level if required and carry out a large amount of SPICE-like simulations. Hence, these design specialists are well aware which of their (functional) parameters are essential for the correct behaviour of their macro.

```
Z06_SNSR IN(NR NP1 NP2 KLK) +
OUT (SIG1 VP1 VN1)
Z02_AMPL IN(VP1 VN1) OUT(VO1) PAR(200)
Z21_ADC IN(V01) OUT(OUTA) PAR(VREF)

/* ADC, model Z21 */
INPUTS: VIN
OUTPUTS: VALUE
PARAMS: VREF
BEHAVIOR:
{
long VALUE;
VALUE=(long)round(VIN/(2.0*VREF/1024.0));
SETBUS ( VOUT_BUS, 9, 0, VALUE);
}
/* END ADC */

Z03_DIV IN(OUTA OUTB)+
OUT(RGEM) PAR(0)
Z07_ENC IN(RGEM) OUT(DISP) PAR(0)
Z17_DISP IN(DISP) PAR(D)
```

Fig.2. Part of the behavioural description of the micro-system, and a detail of the description of a generic ADC macro (bold).

2.2: The mixed-signal test flow

In the test flow for mixed-signal chips, a clear distinction is still made between testing the digital parts and the analog / mixed-signal parts.

For the *digital* macros, like state machines and data paths, commercial tools are available to isolate/ access these blocks by automatic insertion of test hardware available in libraries. For generating the digital test vectors for these macros, either matured ATPG programs are used based on the topology of the macros, or specific patterns are provided (e.g. RAM). The more advanced design tools (e.g. Compass) even provide compilers for self test hardware for common macros like RAM, ROM and multipliers [7]. Also boundary-scan related elements can be automatically inserted, and even threshold structures to be used in parametric tests.

In the case of the *analog* and *mixed-signal* macros, testing tools are very rare, if available at all. It is hence not surprising that this part causes most of the problems and time-to-market delays of mixed-signal systems. In practice, the generation of the test programs for these parts is a highly manual and tedious task requiring much communication between the design team and the test engineer(s).

The first data a test engineer gets are the preliminary specifications of the mixed-signal chip. Later, the designers in the team provide a number of descriptive requests with regard to testing macros or chains of macros. For example, a description such as the analog input pin of the ADC is A1 and a sine of 489 Hz is provided, measure INL and DNL. Also some measurement limits are provided, e.g. DNL 1 lsb, which enables the test engineer to locate problems with regard to tester resource limitations. Next, a number of test modes

are developed which are used to indicate the input/output test paths, additionally required access and/or isolation, and suggested test signals available from the tester used. Based on this information the evaluation test program specification is written in a standardised (sometimes tester-neutral) format, showing the macros or chains of macros including tester hardware and associated settings and of course the detailed test conditions. The latter includes pinning, power-supply voltage ranges, macro parameters to be measured (e.g. DNL), and important settings (e.g. clock frequency). Based on this information, the actual test program for controlling the tester is written and load boards are designed. This process is very time consuming and sensitive to errors.

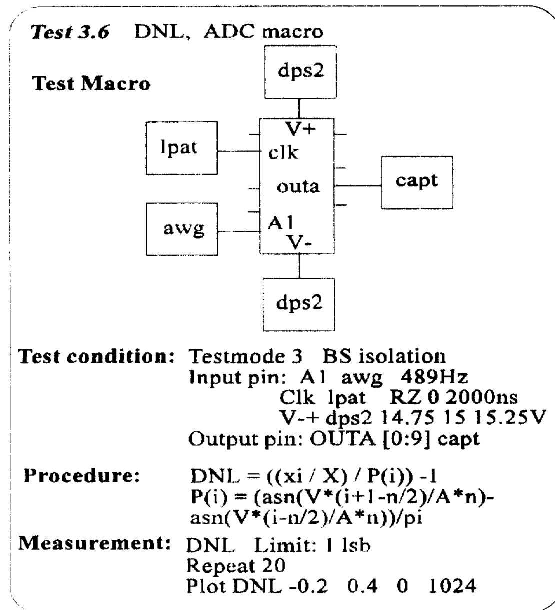

Fig.3. Example of a part of a test program specification of an ADC macro with regard to a DNL measurement.

An example of a part of the previous test program specification with regard to the ADC macro (fig. 1, 2) is shown in figure 3. It shows part of a specification of one of the many tests which are carried out with respect to the ADC. The pin names and associated tester resources are indicated in a drawing. The block labelled 'awg' indicates e.g. an arbitrary wave form generator. In the part test conditions, the specific ranges are given of the resources and also how access is obtained, in our case via boundary-scan. Furthermore the procedure of the measurement parameter is specified, which is DNL in the example.

Finally, the required measurement is specified including its limits and the required representation (DNL plot). Based on this test program specification, the test engineer constructs the final test program code to be executed on the mixed-signal test system.

3: Mixed-signal testability and DFT

Essential in the efficient generation of test patterns and getting access to internal nodes for diagnostic purposes is the observability and controllability of interesting parts of the system. In our approach, basically two levels of hierarchy are considered; the (functional) macros and the elements within macros which can be clusters of transistors. Two crucial parts must be available for reaching this goal; *first* a testability analyser is required which is able to analyse where testability problems might occur in the mixed-signal system. *Second*, reliable hardware must be readily available to subsequently improve the testability, which is the area of mixed-signal design-for-testability (DFT).

3.1: A mixed-signal testability analyser

In the past, an analog testability analyser (TASTE) has been developed for evaluating testability of analog macros based on sensitivity analysis [3]. It has now been investigated to what extent it can be used to evaluate testability problems in mixed-signal systems. As the tool is originally based on sensitivity analysis principles, a direct application to digital circuits fails.

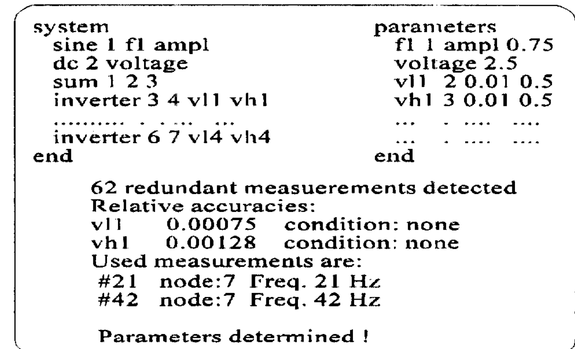

Fig.4. Simplified system and parameter description of a digital inverter-string circuit and associated results.

This problem is circumvented, however, if the behaviour of the circuits is described in terms of a piece-wise linear function incorporating a clear transition region [8]. Actually, the circuit is considered to be an analog component in this case, with the low (vl) and high (vh) voltages as essential (*analog*) parameters. A simple example is shown in figure 4, illustrating the system and parameter description, and the most relevant output of the tool.

As an alternative, it is also possible to incorporate stuck-at faults in TASTE. In this case a (macro) parameter is reserved for each fault. Depending on the

parameter value the fault is active (1) or not (0). As TASTE first calculates on the basis of the nominal value of parameters and next on the changes in the parameters, the influence of the fault on the observable points can be calculated.

3.2: Mixed-signal design-for-testability

Based on the results of the testability analyser, the designer is able to locate testability problems of the essential parameters of his system or macros. With the combined environment of a mixed-signal tester and an automated probing system (e-beam and/or nano probes) there are several possibilities to enhance the testability and also diagnostability.

In our approach, either test hardware (e.g. a multiplexer) is added or (e.g. e-beam [9]) probing points Note that the insertion of these elements can also be carried out at a behavioural level, without yet knowing the actual circuit implementation. Especially in the analog case, the electrical behaviour is crucial in terms of transfer and loading characteristics, as it can change and distort measurement results. A re-simulation of the enhanced system by the analyser will show whether or not the new situation is acceptable.

In terms of *digital* design-for-testability, a large number of solutions are available nowadays, ranging from scan techniques to self testing macros [10]. In the case of analog/mixed-signal design-for-test, the situation is still in its infancy. Most suggested techniques are directly derived from the digital world, such as e.g. analog (boundary-) scan equivalents or analog extensions [4]. However, also some original approaches emerge [4].

The different DFT techniques and their links to DFT hardware libraries of actual implementations are stored in our case in a knowledge base. They are maintained respectively by test engineers and library designers; the maintenance of the knowledge base represents a significant part of the operational costs of *MISSED*. This knowledge base also incorporates different analog DFT techniques, which can be suggested to the designer if problems of accessibility are detected while using the testability analyser. It includes e.g. boundary-scan methods, Iddq current testing, plain multiplexers or automatically inserted probing test paths. A rerun of the (behavioural) analyser will show whether the improvements are sufficient. With regard to very detailed *diagnostics* down to small clusters of transistors, no automation tools are available yet, except a software program which determines the best location for probing a net given the layout. Automatic insertion of observation points is possible if no direct access is obtained. It might however require a re-simulation because of slightly changed loading conditions.

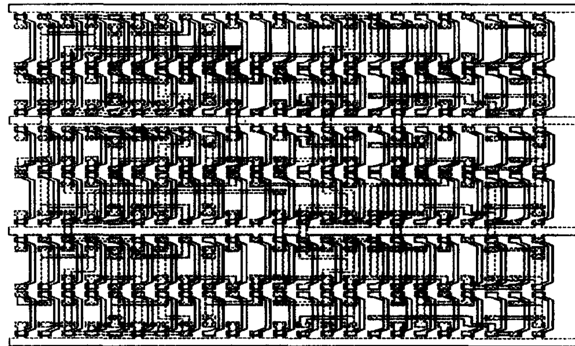

Fig.5. Library cell for getting access to an analog block implemented in a mixed-signal sea-of-gates environment.

For practical reasons, too many paths, a cluster of around 50 transistors has been chosen as diagnostic resolution. Figure 5 shows a DFT library cell in a mixed-signal sea-of-gates environment of a boundary-scan controlled analog high-performance multiplexer giving access to analog macros. This particular implementation uses a Philips CMOS 1.2 μm double-metal process. Amongst other things, the DFT implementation library contains a boundary-scan TAP controller [10], a number of analog/digital scanning cells, high-performance AD/DA converters and (e.g. e-beam) observation structures. The inclusion of Iddq monitors are currently under consideration.

4: The test verification environment of MISSED

The activation of the integrated software tool *MISSED* is started at the very beginning of the design process. It can be considered as a shell around other packages in a window (SUN) environment. Already during the behavioural simulation *test attributes* are linked to the macros, e.g. an AD converter, describing how to test these macros and chains of macros. This approach holds primarily for analog and mixed-signal (chains of) macros. Digital macros are treated in an advanced but conventional manner as already discussed in section 2.2..

Basically, the test problem is split into two parts. *First*, it is investigated by the designer which parameters in a macro and/or chain of macros are of crucial importance in his particular application. These can be local parameters (e.g. DNL of an ADC macro) or global parameters of a chain (e.g. frequency behaviour). The *second* problem of verifying and subsequently getting access to parameters of macros or a chain of macros has already been discussed in section 3.

With regard to the first problem, the designer pops up an interactive window showing a number of routines which can be applied to that type of macro (e.g. an ADC) or a blank editing space in the case of a new test routine

for a macro or chain of macros. This routine will be referred to as an *attribute* to that macro.

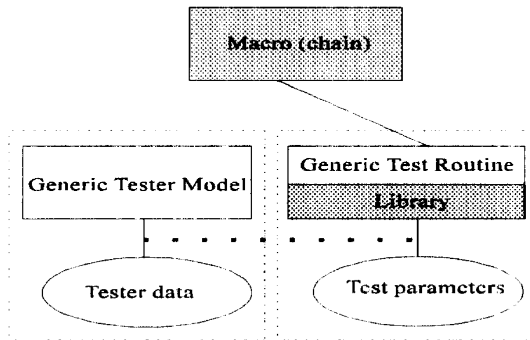

Fig.7. Global set-up of test data insertion and tester verification. Parameters are separated from generic routines.

Actually, the *generic* test routine of an *existing* macro is stored in a knowledge base (library, fig. 7) and is maintained by the test engineer. This part does not incorporate implementation dependent test data.

Based on his macro application, derived from the specification of the system, the designer is able to fill in initial test parameters (e.g. DNL requirements for an ADC), which are subsequently verified by the tester model. This is a behavioural generic model of a mixed-signal tester, linked to a data file which is tester dependent. The tester parameters are stored in a separate file, which is linked to the routine (tester data, fig. 7).

```
dnl_test (float vin, float ts, int n, float dnl []);
{
int codes [], nr_samples;
  histogram_test (vin,n,nr_samples, codes []);
  calculate_dnl(n, nr_samples, dnl [], vin);
}
calculate_dnl (int n, int nr_samples,
 int codes [], float dnl []);
{
int i, nr_codes;
float factor, p;
  nr_codes=pow(2, n);
  factor=2/n;
  for (i=1; i<=nr_codes; i++)
    {
    p=(asn(factor*(i+1-n/2))
    -asn(factor*(i-n/2)))/pi;
    dnl [i]=((codes [i]/nr_samples)/p)-1;
    }
}
```

Fig.8. Example of a part of an ADC test description (DNL) in the C language and associated test parameter settings.

The relationships between tester model, tester data, macro test routine and parameters are symbolically depicted in figure 7. The verification stage is carried out to be sure that the required tester resources are indeed available; it often takes the form of a simulation, as is the case in DANTES [5, 11] and look-alikes [13]. As the tester models are described as macros in the C-language, they can be considered as part of the total design hardware. The previously discussed mixed-signal simulator is used for simulation purposes.

It is however also possible in the *attribute* window to specify other than the standard tests in the C language. It is even possible to introduce a new super macro window consisting of a chain of macros, based on either existing or new macros. This generic routine can be filled in by the designer, the test engineer or both. A trivial example of such a test routine is shown in figure 8 for DNL in an ADC macro. Data with regard to the pinning and signal names have already been related via the behavioural description. During the more detailed design of the macro, more details about the parameters and test routines become available to the designer. Hence the test data becomes more detailed, while the *consistency* of data remains dynamically verified. In essence the test flow is not treated significantly different from the design flow.

An essential part of the tool is the collection of test data and routines of macros or chains of macros into a combined test program. Use is made of set theory and heuristics. No optimisation is carried out with regard to reducing test time or test data volume. Although a suggestion is made for a test program, this part is interactive and can be changed by the test engineer.

The last transformation of the generic test program into the actual code for the tester is done with an automatic test-program generator. The input is EDIF-Test and the output an implementation-depend test program, which uses the Advantest Test Language in our case. The generator is based on the " skeleton " approach of which a number of possible realisations and variations have been presented in the past [12].

5: The diagnostic environment of MISSED

The diagnostic part of *MISSED* during prototype verification consists of a very close interaction between the mixed-signal tester and the prober system, and of course the CA(D/T) data base. Although the prober system has been an e-beam prober in our case, also nano probes can be used, or a combination of both. Also in the diagnostic part, a clear division is made between the digital and analog / mixed-signal parts. Digital diagnosis can proceed fully automatically (stuck-at), whilst analog / mixed-signal diagnosis is in assistance form only.

Based on the results of the external tester, *digital* macros can be designated to be faulty because of the DFT incorporated in the design. These macros are subsequently diagnosed in more detail, based on hierarchical backtracing [14] up to a guaranteed detail level of a cluster of about fifty transistors. A fault dictionary approach is used to help to identify the problem clusters [15]. It is emphasised that during the (layout) design care

must be taken to provide upper metal at crucial cluster points, where testability analysis can again help. Only voltages are measured and the comparison with simulation results is automated. Based on the suspected area, the netlist and the fault dictionary, new test points can be suggested (guided probing). The diagnosis of *analog* and *mixed-signal* macros causes much more problems, because of the existence of catastrophic as well as soft errors. Again, from the previous macro design approach it is possible to point to a *possibly* faulty macro. If the prober used is an *e-beam*, at this moment, only voltages can be measured with limited resolution (~ 10 mV) where in addition averaging of data is carried out. These problems, and relatively long measurement times, obviously limit the diagnostic capabilities of the current e-beam systems. Possible solutions to increase its capabilities are the use of another spectrometer and column [16], and the addition of a (magnetic) nano probe for current measurements. However, nano probes still suffer from insufficient dynamic capabilities at this moment.

The diagnostic software part uses a combination of two approaches, the (hierarchical) model-based reasoning [17, 18] , and an expert system including knowledge of used macros in the past and the principle of accumulated fault evidence. Essential is the consistency verification of the used models in the design database with measurement results.

In the latter case the comparisons are based on ranges of parameters. For small analog and mixed-signal macros it is possible to create a fault dictionary if only catastrophic faults are assumed; otherwise a boundary case testing with induction reasoning must be used.

6: Conclusions

In order to reduce the time-to-market, an environment has been presented which provides a first step of automation in the process of generating test signals for mixed-signal systems and insertion of DFT to ensure testability and a degree of diagnostability during prototype debugging.

A very close link between design and test data is essential in the whole approach, as is a strict separation between behaviour and actual system implementation (function/form) dichotomy.

The current environment has some economic and technical limitations, and so does not claim to be a complete solution for all mixed-signal test and diagnostic problems. This is especially true for the limited analog test capabilities of electron-beam probes, which are still a subject of research.

Acknowledgements

The authors like to acknowledge the contributions of K. Lippe, R. Tangelder and A. Kamp of MESA and valuable discussions with D. Bradley (ASEG) and A. Hu (AAI).

References

[1] V. Chickermane, J. Lee and J.H. Patel, " Design for testability using architectural descriptions ", ITC 1992, Baltimore, USA, pp. 752-760.

[2] V. Pitchumani, P. Mayor and N. Radia, " Fault diagnosis using functional fault models for VHDL descriptions", ITC 1991, Nashville, USA, pp. 327-337.

[3] G.J. Hemink, B.M. Meyer and H.G. Kerkhoff, " TASTE: a tool for analog testability analysis ", IEEE Transactions on CAD, vol. 9, no. 6, June 1990, pp. 829-838.

[4] H.G. Kerkhoff, " Analog design for testability ", chapter 12 in "Analog VLSI", edited by M. Ismael and T. Fiez, Mc. Graw-Hill Publ. , New York, 1993.

[5] S.C. Bateman and W.H. Kao, " Simulation of an integrated design and test environment for mixed-signal integrated circuits ", ITC 1992, Baltimore , USA, pp. 405-414.

[6] E.J. Marinessen, K. Kuiper and C. Wouters, " Testability and protocol expansion in hierarchical macro testing ", ETC, Rotterdam, The Netherlands, 1993, pp. 28-36.

[7] Compass Design Automation, " Test tools V8R3 users manual ", 1993.

[8] J.G. van der Bijl and P.J.A. Hagendoorn, " Evaluation of the application of TASTE in mixed-signal circuits ", Report University of Twente, Enschede, The Netherlands, 1993.

[9] K.Hermann and Kubalek, " Design for e-beam testability ", Microelectronic Engineering, no. 7, 1987, pp. 405-415.

[10] R.P van Riessen, H.G. Kerkhoff and A. Kloppenburg, " Designing and Implementing an architecture with boundary-scan ", IEEE Design and Test of Computers, vol. 7, no. 1, February 1990, pp. 9-19

[11] W. Kao, J. Xia and T. Boydston, " Automatic test-program generation for mixed-signal ICs via design to test link", ITC 1992, pp. 860-865.

[12] R. Arnold et al., " Test/Agent: CAD-integrated automatic generation of test programs ", ITC 1992, pp. 854-859.

[13] A. Keady et al., " Mixed-signal automatic test program generation ", ETC 1993, Rotterdam, The Netherlands, April 1993, p. 528.

[14] A.C. Noble, " A diagnostic assistant for integrated circuit diagnosis ", 3rd Conf. on Electron and Optical beam testing of ICs, Como, Italy, pp. 78-85, September 1991.

[15] T. Yano, " Fast fault diagnostic method for e-beam tester ", ITC, 1987, pp. 561-565.

[16] A. Khursheed and A.R. Dinnis, " A time-of-flight voltage contrast detector for measurements on VLSI circuits ", Meas. Science Technology 1, 1990, pp. 581-591.

[17] R. Rastogi and K. Sierzega, " A new approach to mixed-signal diagnosis ", ITC 1990, pp. 591-597.

[18] A. Mc.Keon and A. Wakeling, " The automatic diagnosis of faults in analogue and mixed-signal circuits ", EDAC 1991, pp. 89-93.