MISMATCH: A Basis for Semi-Automatic Functional Mixed-Signal Test-Pattern Generation*

Hans Kerkhoff, Ronald Tangelder, Han Speek and Nur Engin
MESA Research Institute / University of Twente
P.O. Box 217
7500 AE Enschede, The Netherlands

Abstract

This paper describes a tool which assists the designer in the rapid generation of functional tests for mixed-signal circuits down to the actual test-signals for the tester. The tool is based on manipulating design data, making use of macro-based test libraries and tester resources provided by the test engineer, and computer-based interaction with the designer.

Keywords: *functional ATPG, mixed-signal testing, CAD & CAT link*

1. Introduction

1.1 Motivation

In the past years, the complexity as well as the level of performance of mixed-signal devices has dramatically increased. Together with the reduced time-to-market, this has put a heavy pressure in terms of time and personnel on developing efficient test programs for such devices. This especially holds for the *analogue* parts in the design.

There are a number of factors which account for this: first, the development of tests often starts late in the design phase, second the gap between design and test environment is still not effectively bridged and the tests for the analogue parts are always generated *manually* by experienced and hence scarce mixed-signal test engineers up to now. With regard to the first two issues several approaches [1,2] have been suggested in the production test environment in the past four years of which the "DANTES" initiative [1] is the best known one. However, an effort towards (semi-)automatic test-signal generation in this kind of environment, thereby shifting the role of the test engineer, has not yet been undertaken.

1.2 Goal

This paper focuses on investigating possibilities and limitations in macro-based mixed-signal design and prototype testing environments to provide a basis for semi-automatic functional test-signal generation for mixed-signal devices. For simplicity, we limit the mixed-signal testing problem to register-controlled analogue circuits. As for digital parts structural testing generation is also assumed to be available, we will emphasise on the analogue parts.

2. The CAD & CAT environment

Investigations on typical (mixed-signal) integrated-circuit design practices and existing CAD and CAT environments show a number of opportunities as well as obstacles in using design and test data as a basis for functional test-pattern generation.

2.1 Opportunities

Among the opportunities is the fact that human beings tend to simulate and verify *functional* behaviour in a CAD environment. Also design-partitioning systems are usually based on *functional* decomposition of complex systems into the interconnection of smaller ones. This matches well with the often practised *functional* testing of especially the analogue parts of a circuit. Furthermore, tests of often used functional blocks or *macros* can be available in a test library in a CAT environment irrespective of their actual implementation. To summarise, a vast amount of important data with regard to functional testing is already available but is not used !

* This paper has been prepared within the ESPRIT "AMATIST" project

2.2 Obstacles

However, there are also a number of obstacles. In the present CAD systems, simulation results are overwritten after each simulation, thereby destroying very vital information. Hence a selective copy operation is required to save this information. In addition, the design environment is a virtual reality, where for instance timing resolutions and clocking frequencies do not have to be realistic in a practical sense and simulations are event-driven. This is a problem in the CAT environment, where one has to deal with real-world cycle-based testers, having obvious physical limitations in timing resolutions etc. These obstacles can be circumvented by having a "tester-resource check" on the simulation data and convert event-driven simulation results to cycle-based ones [1,2].

The development of CAT tools is dominated by ATE vendors. Unfortunately these environments often employ proprietary (low-level) languages with many manual interactions to describe their tests to their tester. The same is true for describing tests for analogue macros in the test libraries. However, recent activities in the ATE world include the use of (V)HDL-(A) in directly describing tests to a test system [3] enabling a common exchange platform for the design and test environment. As a consequence, test libraries can also be described by (V)HDL-(A).

Another potential drawback is that tests in these libraries not always have a direct link to functional simulations and behavioural descriptions in the design environment. For example, a system designer is not likely to verify a macro ADC by using a histogram approach as is done by a test engineer. There is no common solution for this problem yet and requires research how to link functional simulations to certain test practices.

Based on the previous remarks, a tool called "MISMATCH" has been developed, essentially employing the opportunities and solving or circumventing the obstacles. The user is the *designer*. The link with the tester world has been established in our case by employing hardware and software from an advanced VXI-based mixed-signal verification tester [4].

3. The tool MISMATCH

3.1 Goal and boundary conditions

Starting point of our research of developing the (*designers'*) tool MISMATCH (**MI**xed-**S**ignal **MA**nipulation **T**ool for **Ch**ips) for semi-automatic functional test-generation for signals is that the assumed design style is *macro* based. In other words the total system is built up from functional blocks like operational amplifiers and filters etc. This is especially important for the analogue test-generation part, as the digital part is already mostly covered by automatically generated structural tests. All design data, like description of the design at all hierarchical levels and (input, output) simulation data are assumed to be available in the CAD framework. Furthermore, the availability in the CAT environment of a test library containing generic test procedures for macros is assumed [5] or has to be created by a *test engineer*. Test costs have not been considered in MISMATCH at this stage of development. Comparisons of simulated output results and measurements are also not considered as this assumes *feedback* of measurements in MISMATCH. In this stage, these issues are still handled at the CAT level.

Early in the research it became clear that a completely automatic generation of high quality test signals based on available design data does not seem feasible. One of the trivial limitations being of course the quality and quantity of design data which is provided by the individual designer. Our reference point with regard to the term "quality" is how well the generated functional tests cover defects provided by IFA calculations and simulations [6].

3.2 Design framework

In the first section it has become clear that the framework where the exchange of design and test data has to take place is of extreme importance. In an attempt to find out to which extent existing (CAD) frameworks offer this possibility instead of developing one of our own from scratch, two alternative frameworks have been investigated. Among the criteria of suitability were: accessibility of the data, available data manipulation tools and their flexibility, as well as their link to common front- and back-end design tools and the test environment.

An academic framework developed within the Dutch NELSIS project was investigated first. Although being a very open framework (available source codes), the link to front-end and test tools is respectively poor and absent. Furthermore, the number and capabilities of the available data-management routines were extremely limited. In addition, the chosen binary-data model was not very suitable for our purpose.

Next, a well-known commercial framework was investigated. Although having an excellent connection to its proprietary front- and back-end tools and possible links to the tester environment, the openness of the framework was completely determined by a propriety procedural language which can only provide limited access and data manipulation. Initial investigations indicated that a number of our required operations are not directly supported by this language interface. This is

supported by experiences which others had with other well-known commercial frameworks.

Hence it was decided to construct a framework of our own. The basic idea of our framework was to construct an environment were all relevant data, coming from either design or test environment, can be imported and exported in ASCII. This has the advantage that many design and test tools can handle data in ASCII. The disadvantage is that parsers based on for instance "lex" and "yacc" have to written. Furthermore, import and export of large amounts of data can be time consuming.

Figure 1 illustrates the location of MISMATCH within our design and test environment.

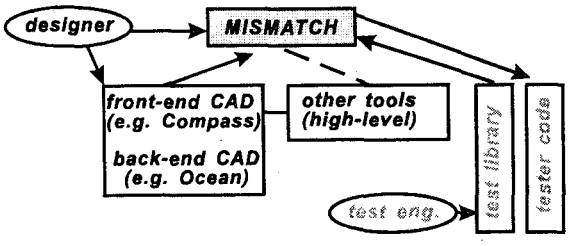

Figure 1: *Links between MISMATCH and our CAD tools, test library, tester code and users.*

The user interface is located within MISMATCH and will be discussed later. The block labelled " test library " represents test routines of macros which are provided by the test engineer. For describing the routines, VHDL is used as in the end the tests for the test system can be described in VHDL(-A) anyway. In the end, the test-engineer is still free to add or alter data in the source code for the tester if required.

3.3 Data manipulation

In practice, the tool works as follows. During functional simulation (starting from behavioural, down to structural), CAD data is copied in ASCII to MISMATCH. This will start with information about pinning (IN,OUT, digital, analogue, power, clock). MISMATCH will monitor whether this information is provided and alert the designer in its window if not. The choice which design files are considered to be correct and are exported to MISMATCH is *up to the designer.*

Furthermore, the designer has to label these files in terms of a number of subclasses, basically coding the intention of a test, e.g. for amplitude-frequency simulation. In addition, the type of macro, for instance an operational amplifier, has to be indicated in terms of a subclass also. C++ is a suitable language to accommodate such constructions.

MISMATCH extracts essential parameters from the inputs as well as the outputs of these files by means of a method. For instance in the case of an amplitude-frequency simulation result, it abstracts the amplitude and frequency range of the input, and e.g. the -3dB points. These methods can be implemented by means of C routines. Furthermore the link is made to the related macro, thereby substituting the extracted parameters into the test procedure file. Before this, the parameters can be additionally manipulated, e.g. by applying heuristic rules often used by test engineers. An example of this is adding 10% of a range to ensure a better view of the system slightly outside the designed region. Second, it is verified that the resulting values of the parameters are within range of the tester used. Both issues can be implemented by means of (sub)classes and methods. Finally, the results are presented to the user in terms of a window of that macro.

3.4 User interface

Figure 2 shows how a macro test window could look like; The MISMATCH window is active in parallel with the actual design program.

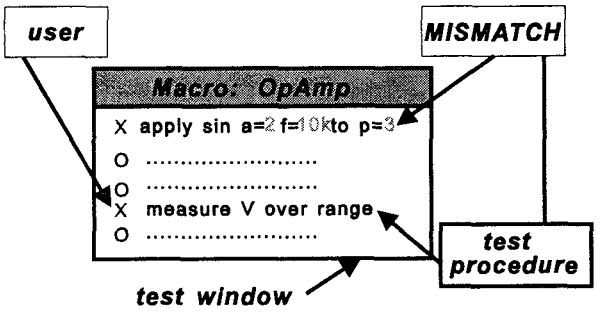

Figure 2: *Example of a window with regard to a chosen macro test and its sources*

The designer can select in the window which tests are relevant in his application. The default tests originate from the test library of that macro. The actual values in these tests are obtained from MISMATCH. Blinking parameters indicate which parameters could not be obtained by the design data available. By clicking these parameters suggestions (HELP function) can be given which type of data is required in terms of simulation data, for instance phase-frequency data. The designer can also change the parameter values directly, even values already provided by MISMATCH.

The sequence of design is such that first all macros in the design have to be completed in this way. In addition, the digital codes have to be provided for each macro how access can be obtained. The latter can be provided from functional simulations at the top level. Finally, the top level window is constructed which looks similar in appearance as

the macro window. There are two sources of data. First, data from top level functional simulations is used in a straight-forward manner, as test procedures for the total application will generally be not available. The other source originates from the macro windows, where the link is made between *all individual* macro inputs and outputs (including digital control) and the inputs and outputs of the top level via the architectural data. At this stage of the research, the assumption has been made that all analogue and mixed macros are fully accessible providing the proper digital control settings. Although this is generally not the case, it simplifies the construction of the overall top-level tests now being a simple addition of test sequences. The construction of top-level functional tests based on the interconnection of macros has *not* been addressed yet. Unfortunately, this is not as straight-forward as in the pure digital case and will require much more research.

3.5 Tester interface

With regard to the last link, to convert the generated tests into actual tests, already a lot of work has been carried out in the past by ATE vendors. Our approach follows translation of VHDL(-A) based tests into tester source code as is already demonstrated in [3]. Data export from MISMATCH to the tester is in ASCII.

4. Conclusions

The tool MISMATCH has been presented to support the generation of functional tests for mixed-signal (register-controlled analogue) devices. It makes use of design data available in the CAD database and test procedures and tester resource models in the CAT environment. Future extensions include DfT support during the design in the case of incomplete analogue accessibility and transparency analysis of macros based on behavioural descriptions. Furthermore, much more research is required to link often used test procedures with common functional simulations.

References

[1] J.Q Xia et al., " Dynamic Test Emulation for EDA-Based Mixed-Signal Test Development Automation ", ITC 1995, Washington, USA, pp. 761-770.

[2] S. Bullock, " Report on a pilot project successfully implementing a Design-to-Test Methodology", ITC 1995, Washington, USA, pp. 771-780.

[3] K. Helmreich, " How virtual test becomes reality", Proc. of ED&T , session 10D.1, Paris, France, 1996.

[4] B. Schneider and S. Soegaard," IntegraTEST: The New Wave in Mixed-Signal Test ", ITC 1995, Washington, USA, pp. 750-760.

[5] H. Kerkhoff and G. Docherty, " MISSED: An Environment for Mixed_Signal Microsystem Testing and Diagnosis", ATS 1993, Beijing , China, pp. 88-93.

[6] R.J.A. Harvey et al.," Defect Oriented Test Development Based on Inductive Fault Analysis ", Proc. International Mixed-Signal Testing Workshop, Villard de Lans, France, June 1995, pp. 2-9.