Timed Functional Modeling for Mixed-Signal Boards in Maintenance Testing: A Case Study

Bertrand Gilles, Valérie-Anne Nicolas, Laurent Lemarchand, Lionel Marcé

Timed Functional Modeling for Mixed-Signal Boards in Maintenance Testing: A Case Study

Bertrand Gilles Valérie-Anne Nicolas Laurent Lemarchand Lionel Marcé
EA3883 LISYC - Département Informatique - Université de Bretagne Occidentale
20 avenue Le Gorgeu - CS 93837 - 29238 Brest Cedex 3 France
{gilles|vnicolas|lemarch|marce}@univ-brest.fr

Abstract

In the context of maintenance testing and diagnosis of faulty boards, a functional FSM *(Finite State Machine)-based model for mixed-signal boards has been introduced [1]. It has been extended for dealing with time sequences aspects. In this paper, the new modeling technique is presented.*

1. Introduction

Numerous test methods and techniques have been developed for circuit test [2, 3], associated to the different stages of product life-cycle, mainly at design and production levels. Surprisingly, not much interest has been thrown into testing during the maintenance stage. However, maintenance testing has its own specificity. Thus, our work is related to maintenance testing and focus more particularly on mixed-signal boards.

The maintenance stage is one of the step constituting the life-cycle of a board. This stage begins after the development/production cycle. Because of this location in the life-cycle, this stage is complex. First, the knowledge about the board is most often reduced for maintenance people: no designer direct knowledge, partial documentation, level of confidentiality (military, commercial aspects). Second, unitary in situ tests are not sufficient because of aged components and their interactions at tolerance limits. Moreover, large complexity of boards and safety aspects in embedded systems (avionics, automotive,...) have to be managed. All of this implies functional testing in order to check the board behavior, and to determine and replace faulty components in case of defective functionality. Since they only make use of the external behavior of the components, functional-based models may address a wide spectrum of situations concerning board maintenance testing: they may be adapted to the amount of information available (component specification levels), to the nature of the components (digital, mixed-signal, or analog) and to the goal of the test (go-nogo, fine-grain diagnosis oriented testing). Functional testing of component is not used during design or production stages because test software development is costly. It is mainly achieved at the system level in order to test the interactions between components and to check if the global system meets its specification requirements. Thus, there is no predefined functional tests available at the board level.

Moreover, because of lack of material, diagnosis and repair is often realized in an empirical way. Clearly, specialized tools are needed to guide or automate at least a part of the work involved in the maintenance stage. Our goal is to provide a help to board maintenance testing and diagnosis. We propose a method supported by a semi-automatic tool allowing the functional specification of the board, the definition of testing strategies and the automatic test data set generation. Because automation implies using formalism, the formalism has to be chosen to match background practitioners in order to be really useful. Talking with our industrial partner, we chose the FSM formalism which is well known by testing engineers.

We first present the FSM-based functional model for mixed-signal boards. New time modeling features are described next. Then, the model-based ATPG (Automatic Test Pattern Generation) is presented and we deal with a simple case study. Then, we show the implementation prototype. A discussion on future work ends the paper.

2. FSM-based board modeling

A board modeling for maintenance ATPG has been proposed in [1]. It relies on FSM-based functional models for the components of mixed-signal boards.

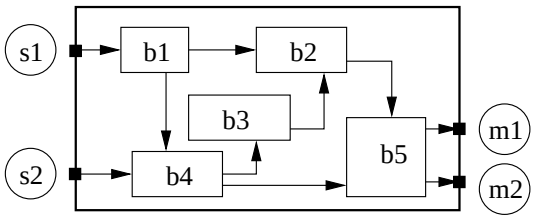

Figure 1. A board is an assembly of blocks.

2.1. Board level modeling

The board is first modeled at the *board level*, as a set of interconnected functional blocks, as depicted in picture 1. In addition to building blocks of the board, some external blocks are needed to model connections between the board primary inputs/outputs (PI/PO) and an automatic test equipment (ATE): external sources which supply input signals, or output measurement points.

Blocks are analog, digital or mixed-signal, and may have several inputs and outputs. Oriented links denote data exchanges between components. Signals exchanged on a link are characterized by their amplitude, form, frequency and type.

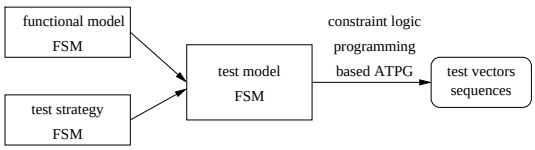

Figure 2. The test pattern generation process.

The board checking consists in testing each block individually using its associated FSM-*based test model*. This test model is created by merging a block *functional model* and a *testing strategy*, as depicted in picture 2. Test vectors for a component are generated by covering each transition of the component's test model. Since the block under test is often embedded within the board, without any test access mechanism (e.g. block b3 of picture 1), the functional models of adjacent components are used for justification and propagation of the block I/O up to PI/PO. Final vectors are computed using constraint logic programming (CLP).

2.2. Block level modeling

The proposed approach for the functional modeling of the components is based on communicating FSM, since these objects are flexible enough to handle various kinds of board specifications.

The functional model of each component is a set of communicating FSM. The FSM model may be specified with an appropriate graphical user interface (GUI), derivated from some VHDL/AMS subset specification, or instantiated from a parametrized functions library. The latter is mainly used for common analog or mixed-signal blocks. Test vectors lists are also usable. Sometimes, these are the simplest way for specifying blackbox-like blocks functionalities.

All of these specification techniques may be mixed, according to the nature of the system components, and to the kind and form of available descriptions for the different blocks.

The test model To generate appropriate test vectors for a given component, testing strategies are applied to the functional model [4]. This is realized mainly by extending the functional model FSM at I/O points, with new FSM pieces implementing the testing strategy. The test model for a component results from this merging. Since test patterns generation corresponds to FSM transitions covering, strategies are described as combinations of transitions. As a simplistic example, checking one digital output pin activity corresponds to some test vectors with 1 and some others with 0 for this pin. These vectors are generated from a FSM containing transitions for both the 0-value and the 1-value.

2.3. Time aspects

From experimented test engineers point of view, it appears that, for at least go-nogo testing, simple time management is often sufficient. In the context of maintenance testing, the modeling of accurate delay values is not necessary. These values are often either useless, or unavailable. The former arises when approximative clock frequency is set by engineer for test run, the latter when the board comes without timing information. However, at least sequences of values are necessary for meeting test requirements. Thus, ordering test data is mandatory. Since the model presented in [1] is based on communicating FSM, it is of interest to model time with such objects.

Our first approach to deal with time sequences is based on a simple clock model. Its FSM is similar to a board input model as time may be considered as an external data for the board. Multiple clocks may coexist in the same functional model. The different actions driven by a clock are specified by waiting for the `top` value on some transition of the receiving FSM. In our modeling, a behavior of the board is represented by a path from PI to PO in a set of communicating FSM. Thus, sensitizing a path leads to cross the clocks edges a number of times and thus to compute dates in terms of number of tops for the associated test data.

With this kind of modeling for clocks, time modeling is decoupled from the component modeling. Clock models

may be generated automatically and changed easily according to testing needs without modifying the remaining parts of the model.

However, this time management may not be sufficient in some cases. Suppose a modeling with two clocks Clk_1 and Clk_2 sending top_1 and top_2 respectively. These tops do not have an associated date (time stamp). The test pattern generation process explained in section 2.4 "asks" for some top events in order to obtain a test data for a component. This test data is a sequence dated in a relative way. Thus, the real dating of the test data comes from the ordering of the top events. If there is no constraint on the periodicity of the two clocks, a consistent timing may be associated to the test data. Otherwise, a generated test data may have a wrong timing because the sequence of top events may be conflicting with the period of each type of event.

A first approach to solve this problem is to increase the algorithmic complexity of the generation process in order to eliminate wrong sequences. An alternative approach consists in using time stamped events with a same time reference. Thus, the test data is dated in an absolute way. We choose the second approach in order to control the algorithmic complexity. Thus, we propose to manage time stamped events. This is achieved by using timed automata [5]. Indeed, timed automata allow to specify time-dependant behaviors with clocks (like periods) using the same time reference. This approach is illustrated on the case study (see section 3).

2.4. Model-based ATPG

As explained in section 2.1, ATPG is achieved by covering the test models. Covering a transition leads to meet the associated data constraints. The constraints are propagated up to the board's PI/PO, also modeled as FSM. The problem of test data generation is faced using CLP and classical algorithms for finite state machines (transition coverage, state coverage, path coverage). Thanks to CLP, test data are represented in a symbolic way, using ranges of values, dealing efficiently with analog and digital data representations in an uniform way.

Ranges of vectors are computed for reaching the test requirements. Actual values are defined at the end, making possible to take into account some ATE specificities.

3. Case study: The tachy board

The modeling technique, extended for dealing with time sequences aspects (see section 2.3) has been applied to a simple industrial case study. We first give a functional description of the *Tachy board*. Then, we present the board modeling, the testing strategies applied and finally give the expected board testing results.

3.1. Board Description

The Tachy board is a mixed-signal board. It has fourteen analog channels receiving DC signals coming from tachymetric generators. The main function of the board is to check in a cyclic way the values of input signals by comparing them to two voltage thresholds and write into RAM memory the time stamped number of each faulty channel. A channel is faulty if its analog signal is not between the two thresholds. The RAM memory is reseted every six minutes.

For sake of simplicity, we are presenting in the paper a three channels restricted version. This restriction has no incidence on the complexity of modeling and testing.

3.2. Board Modeling

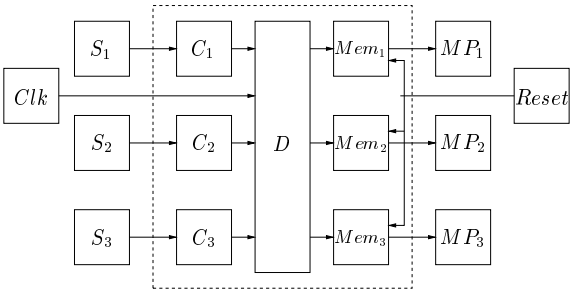

Figure 3. The Tachy board level modeling.

Figure 3 shows the Tachy board level modeling (see section 2.1). The board is delimited by the dashed rectangle and is made of three mixed-signal blocks $C_1 \cdots C_3$, one digital block D and three digital blocks $Mem_1 \cdots Mem_3$. C_i is a two-level comparator, D is the controller which checks cyclicly the comparators outputs and Mem_i is a memory (register). Blocks $S_1 \cdots S_3$ represent analog sources and block $MP_1 \cdots MP_3$ represent digital measurement points. Block Clk represents a clock signal and block $Reset$ represents a reset command.

Each component has to be described at the block level (see section 2.2). We now present the functional models of the different blocks of the Tachy board. Blocks interactions are embedded in blocks descriptions. We begin with the modeling of the board inputs/outputs, next the modeling of the mixed-signal part (comparators blocks), and finally the modeling of the digital part (controller and memory). Specification of accurate testing strategies and resulting test models are discussed last.

Since we make intensive use of communicating FSM, we explain this aspect first.

Communicating FSM The block set of the board corresponds to a set of communicating FSM. Since communica-

tions are involved, FSM transitions are decorated with labels of the form $S \rightarrow G[A]$ where S is an optional synchronization condition between FSM, G an optional boolean guard and A an optional action. The associated semantics is: "when the synchronization condition is verified, the boolean guard is then evaluated. If the guard is true, the transition is being crossed and the action is done".

The synchronization condition S may be a list of the expression $F\,?\,(d_1, d_2, \ldots)$ which means a blocking receiving of d_i data list from FSM F.

The communication between two FSM is realized with a queue. Sendings are allowed in actions as there are not blocking. $F\,!\,(d_1, d_2, \ldots)$ means a not blocking sending of d_i data list towards FSM F.

Board inputs/outputs The modeling of data I/O (board inputs/outputs) is particular. Data are not functional blocks of the board. However, to generate test data, we need to model them. This also allows to model characteristics of sources, generators and measurement tools of a specific ATE in order to produce a test program. Thus, each board input/output has a functional model (but no test model as it does not correspond to a board component).

Figure 4 presents the functional model of the analog S_i source. FSM S_i (by convention, FSM name is the same as block name) sends the x data (which characterises the signal) to FSM C_i. The Tachy board inputs ($S_1 \cdots S_3$) are modeled by three instances of this source.

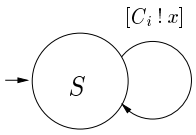

Figure 4. The functional model for a source $\mathbf{S_i}$.

Figure 5 (a) shows the functional model of the clock signal. FSM Clk sends a time stamped top event to FSM D. This behaviour is achieved using the y reference clock.

Figure 5 (b) shows the functional model of the reset signal. FSM Rst sends periodically (with a $T_1 = 6$mn period) a time stamped reset event to each FSM Mem_i, using the same y reference clock and a constraint where $\%$ represents the modulo operator.

Concerning outputs modeling, figure 6 shows the functional model of a measurement point MP_i. FSM MP_i just waits for data coming from FSM Mem_i or FSM D.

Mixed-signal part The mixed-signal part of the board is modeled by three instances ($C_1 \cdots C_3$) of one communicating FSM.

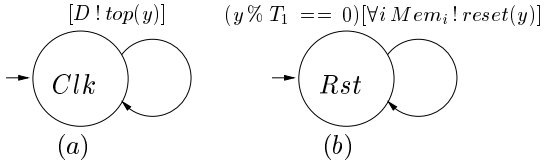

Figure 5. The functional model for the clock signal and the reset signal.

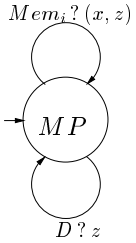

Figure 6. The functional model for the a simple measurement point $\mathbf{MP_i}$.

Figure 7 presents the functional model of the C_i comparator. FSM C_i waits for the x data coming from FSM S_i. Two different behaviors may occur when the x data is received: C_i sends the digital value 1 to FSM D if the received value does not fall within the expected voltage range, the digital value 0 is sent to FSM D otherwise.

Due to the physical characteristics of a comparator, a tolerance δ is introduced to ensure that the comparator will have a good response. This tolerance is expressed in threshold percent.

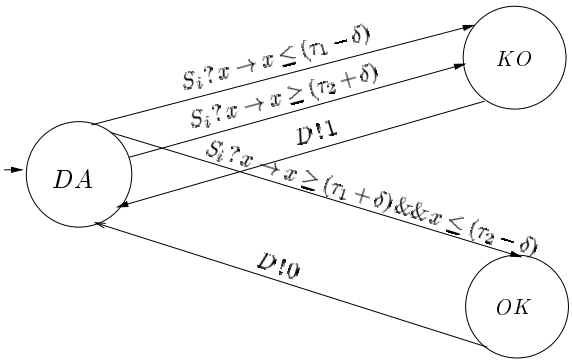

Figure 7. The functional model for a comparator $\mathbf{C_i}$.

Digital part The digital part of the board is modeled by four communicating FSM: one modeling the digital controller of the board and one for each three memories. We ex-

plain first the modeling of the controller and next the modeling of memory.

Figure 8 shows the functional model of the D controller. When a time stamped top event is received from FSM Clk, output data sent by a comparator C_i is read and checked. If the input signal of the comparator is faulty, then a time stamped error flag (which is equal to one) is sent to FSM Mem_i and D waits for the next top. Otherwise, for testing needs, the time stamp of the top is sent to FSM MP_i and D waits for the next top. As we may see, FSM D checks in a cyclic way the output of each comparator, starting with the comparator C_1 (start state). FSM D ensures that a comparator output is always read between two tops.

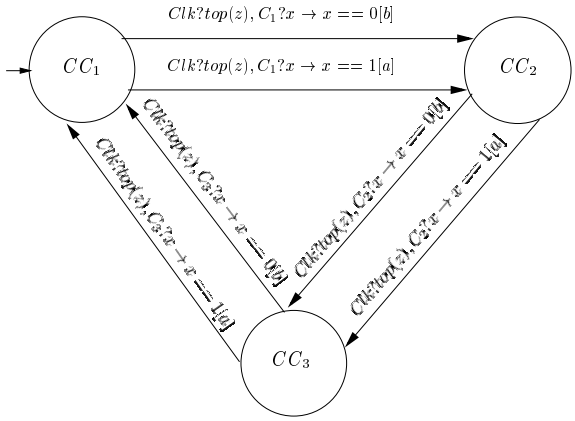

Figure 8. The functional model for the controller D with $\mathbf{a = Mem_i!(1, z)}$ and $\mathbf{b = MP_i!z}$.

The functional model of a memory Mem_i is depicted in figure 9. FSM Mem_i waits either for a time stamped data from FSM D or for a time stamped reset event coming from FSM Rst. The former sends the received data (the time stamped output value of the C_i comparator) to FSM MP_i. The latter sends a time stamped zero-value (because of the reset event) to FSM MP_i.

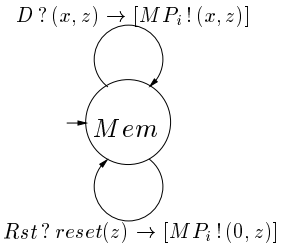

Figure 9. The functional model for a memory $\mathbf{Mem_i}$.

3.3. Testing strategies and test models

The default test model for the controller D is the same as its functional model because it is a digital component. This is also true for all the memories Mem_i. Indeed, a digital component is easily modeled with a FSM and in our method, the transition covering of this FSM is often sufficient for testing the component. When the digital default test model is not sufficient, we apply testing strategies to the outputs (measurement points).

The comparator component is implemented in our library of analog components. Thus, is has an associated default test model depicted in figure 10. It shows that four test data are required for the unitary test.

As previously mentioned, block testing strategies may be specified by the user to improve testing process. It represents the testing engineer skills. A component test model is obtained by merging the block functional model and its eventual associated testing strategy.

An example of testing strategy applied to measurement points is shown in figure 11: we have to check the dating of at least one error flag for each channel as well as the dating of at least one reset event and the dating of at least one good data on at least one channel.

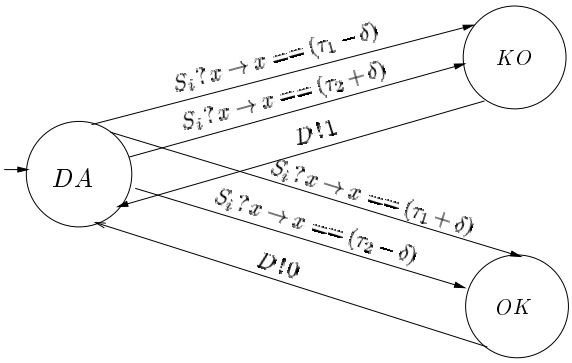

Figure 10. The test model for a comparator $\mathbf{C_i}$.

3.4. Board Testing

The test data set of the board generated with our method is: $TDS = \{TD_1, TD_2, TD_3, TD_4, TD_5\}$ with:

$$
\begin{array}{lll}
TD_1 = & (In = & ((\tau_{11} - \delta, ?, ?, top(z_1), \_), \\
 & & (?, \tau_{12} - \delta, ?, top(z_2), \_), \\
 & & (?, ?, \tau_{13} - \delta, top(z_3), \_)), \\
 & Out = & (([1, z_1], ?, ?), ([1, z_1], [1, z_2], ?), \\
 & & ([1, z_1], [1, z_2], [1, z_3]))) \\
\text{with} & \forall k. & 6k \notin [z_1, z_3] \text{ and } z_1 < z_2 < z_3
\end{array}
$$

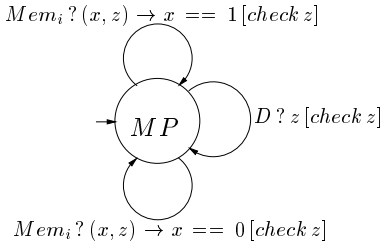

Figure 11. The test model for a measurement point $\mathbf{Mem_i}$ (functional model with an added testing strategy).

TD_2 is like TD_1 replacing $\tau_{1i} - \delta$ by $\tau_{2i} + \delta$.

$$\begin{array}{lll} TD_3 = & (In = & (?,?,?,\_,reset(z)),\\ & Out = & ([0,z],[0,z],[0,z])) \\ \text{with} & \exists k.\ z = 6k & \end{array}$$

$$\begin{array}{lll} TD_4 = & (In = & ((\tau_{11}+\delta,?,?,top(z_1),\_),\\ & & (?,\tau_{12}+\delta,?,top(z_2),\_),\\ & & (?,?,\tau_{13}+\delta,top(z_3),\_)),\\ & Out = & (([?,z_1],?,?),([?,z_1],[?,z_2],?),\\ & & ([?,z_1],[?,z_2],[?,z_3]))) \\ \text{with} & \forall k. & 6k \notin [z_1,z_3] \text{ and } z_1 < z_2 < z_3 \end{array}$$

TD_5 is like TD_4 replacing $\tau_{1i} + \delta$ by $\tau_{2i} - \delta$.

An input 5-tuple has the form $(S_1, S_2, S_3, Clk, Rst)$ and an output 3-tuple has the form (MP_1, MP_2, MP_3) where S_1, S_2, S_3 are the three analog sources, Clk the clock signal, Rst the reset command, and MP_1, MP_2, MP_3 the three digital measurement points (memory state).

? stands for an unspecified value and _ stands for no input value. τ_{1i} and τ_{2i} are the thresholds of a comparator C_i (associated with source S_i).

TD_1 means that 3 input test vectors are executed sequentially on the board, and that 3 corresponding output vectors are then observed.

First test vector puts the event *top* at time z_1 on the clock input, $\tau_{11} - \delta$ on the S_1 input, no event on the reset input, and whatever values on S_2 and S_3. Results observed are value 1 with time-stamp z_1 on MP_1 output, values on MP_2 and MP_3 outputs don't matter.

Then, second test vector puts the event *top* at time z_2 on the clock input, $\tau_{12} - \delta$ on the S_2 input, no event on the reset input, and whatever values on S_1 and S_3. Results observed are value 1 with time-stamp z_1 still on MP_1 output, value 1 with time-stamp z_2 on MP_2 output, value on MP_3 output doesn't matter.

Next, third test vector puts the event *top* at time z_3 on the clock input, $\tau_{13} - \delta$ on the S_3 input, no event on the reset input, and whatever values on S_1 and S_2. Results observed are value 1 with time-stamp z_1 still on MP_1 output, value 1 with time-stamp z_2 still on MP_2 output and value 1 with time-stamp z_3 on MP_3 output. Time-stamps z_1, z_2 and z_3 must match the constraint: $\forall k.6k \notin [z_1, z_3]$ and $z_1 < z_2 < z_3$.

TD_1 and TD_2 test faulty behaviors (thresholds exceeding for all channels). TD_4 and TD_5 test good behaviors (for all channels). TD_3 test the reset command.

We thus consider that this test data set is sufficient to test the board. The size of each test data of TDS is minimal, but we could have generated fewer test data with a bigger size. Note that the generalisation to the whole board is immediate: for fourteen channels, the size of the test data set is the same (5), but each test data except TD_3 corresponds then to sequences of 14 (instead of 3) input/output vectors of size 16 (instead of 5), and TD_3 is the same with input/output vectors of size 16 (instead of 5).

However, TD_4 and TD_5 may seem meaningless as they succeed whatever their output values. This is because there is no writing in memory for good input values, and thus memory keeps its initial state, which is not defined in TD_4 and TD_5. One improvement may be to sequence TD_3 before TD_4 and TD_5 to fix an initial memory state.

The resulting test data set of the board would then be: $TDS' = \{TD_1, TD_2, TD_{34}, TD_{35}\}$ with TD_1 and TD_2 as previously defined and

$$\begin{array}{lll} TD_{34} = & (In = & (?,?,?,\_,reset(z)),\\ & & (\tau_{11}+\delta,?,?,top(z_1),\_),\\ & & (?,\tau_{12}+\delta,?,top(z_2),\_),\\ & & (?,?,\tau_{13}+\delta,top(z_3),\_)),\\ & Out = & (([0,z],[0,z],[0,z]),\\ & & ([0,z_1],[0,z],[0,z]),\\ & & ([0,z_1],[0,z_2],[0,z]),\\ & & ([0,z_1],[0,z_2],[0,z_3]))) \\ \text{with} & \exists k_1. & z = 6k_1 \\ \text{with} & \forall k. & 6k \notin [z_1,z_3] \text{ and } z < z_1 < z_2 < z_3 \end{array}$$

TD_{35} is like TD_{34} replacing $\tau_{1i} + \delta$ by $\tau_{2i} - \delta$.

4. Prototype

We have partially implemented the FSM-based board modeling, the model-based ATPG and time management in a prototype tool. This prototype provides a GUI allowing high level description of mixed-signal boards. In addition, the GUI includes some facilities for the choice of a testing strategy, for the description of the board-ATE connection and for the description of the data (signals) flow. The GUI part of the prototype is written in C++ with the ILOG Views graphic library [6] and the ATPG part is implemented using CLP with the solver ECL^iPS^e [7]. The prototype, which

is still under development, has already been used in simple industrial case studies [1, 4].

5. Conclusion and future work

We have presented a method for the testing of mixed-signal boards in a maintenance context. An approach using timed automata has been proposed to deal with simple time aspects. In particular, it allows the modeling and testing in presence of multiple clocks (dependant or not) with different periods. The method has been validated on two simple industrial case studies. Nevertheless, we are also prospecting for improved testing strategies. Another objective is to extend the models to take into account more complex boards. Further work is required on industrial cases to validate the approach and exhibit its limits.

Acknowledgements

We wish to acknowledge the support provided by the Brittany Regional Council (France) under contract number 20041592, and Bruno Castel and Michel Le Goff from ISIS-MPP for their contribution to this project.

References

[1] B. Gilles, V.-A. Nicolas, L. Lemarchand, L. Marcé, and B. Castel. Towards a New Modelling of Mixed-Signal Boards For Maintenance Testing. In *Proceedings of the 11th IEEE International Mixed-Signals Testing Workshop, IMSTW'05*, pages 90–97, June 2005.

[2] Mark Burns and Gordon W. Roberts. *An Introduction to Mixed-Signal IC Test and Measurement*. The Oxford Series in Electrical and Computer Engineering. Oxford University Press, 2001.

[3] M. L. Bushnell and V. D. Agrawal. *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. Springer, 2000.

[4] V.-A. Nicolas, B. Gilles, L. Lemarchand, L. Marcé, and B. Castel. A Maintenance-Oriented Board Testing Approach. In *Proceedings of the 3rd IEEE International East-West Design and Test Workshop, EWDTW'05*, pages 143–147, September 2005.

[5] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[6] ILOG. *ILOG Views Fundation 5.0 User's Manual*, 2002.

[7] A. M. Cheadle, W. Harvey, A.J. Sadler, J. Schimpf, K. Shen, and M.G. Wallace. Eclipse: An Introduction. Technical Report IC-Parc-03-1, IC-Parc, Imperial College London, 2003.