On Fault Modeling for Dynamic MOS Circuits

Hans-Joachim Wunderlich and Wolfgang Rosenstiel

University of Karlsruhe
Institut fuer Informatik IV (Prof. D. Schmid)
Postfach 6980, D-7500 Karlsruhe
F. R. Germany

Abstract:

Static nMOS and static CMOS circuits show some serious problems for fault modeling and testing. In this paper we point out, that most of these problems are avoided by using dynamic nMOS or dynamic CMOS circuits. Stuck-open faults in this case do not result in sequential behaviour. A logical fault model is presented, where a fault of a logic gate will cause either a faulty combinational function or a degradation of the performance.

Integrated test tools for technology dependent logical fault models based on random self test techniques are presented.

Keywords: Dynamic MOS, fault modeling, random testing, test pattern generation

1. Introduction

This work is related to the german E.I.S. project, a joint foundation of the government together with german universities and german industry concentrating on teaching and research in microelectronics and CAD.

All classical algorithms for test pattern generation or testability analysis are based on the assumption, that a faulty combinational circuit still remains combinational and digital. But this does not hold for circuits in static CMOS or nMOS technology, where the stuck-open faults may transform a combinational circuit into a sequential one [1] (see fig. 1).

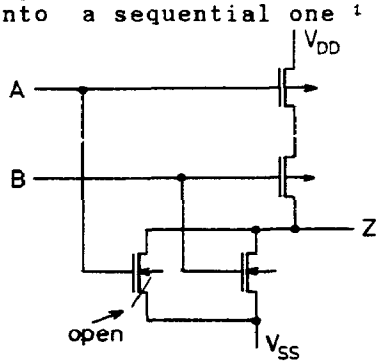


Figure 1: A faulty CMOS NOR

Supposed the marked connection in figure 1 is open, the function table would change as follows

A	B	$Z(t+\delta)$	$Z_{faulty}(t+\delta)$
0	0	1	1
0	1	0	0
1	0	0	Z(t)
1	1	0	0

This table shows the change into sequential behaviour, which is the most serious problem under the aspect of testing, since

- the fault injection algorithms of parallel, deductive or concurrent fault simulators doesn't work any more;
- because of races and spikes faulty nodes may be precharged and thus fault detection may be prevented;
- test pattern generation has to be performed both on switch level and for sequential circuits;
- scan path techniques fail since the state of the faulty circuit may change during shifting;
- the tools for testability analysis fail.

Furthermore CMOS faults may result in a correct logical behaviour with wrong timing, i.e. with longer switching delays (see Fig. 2).

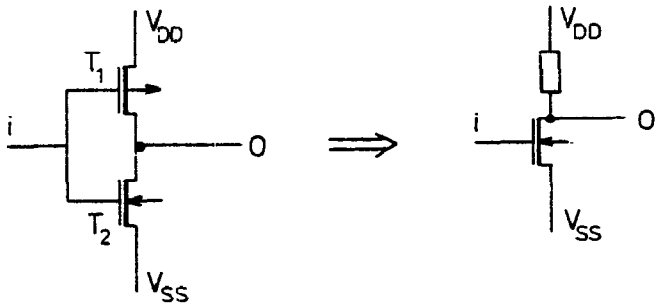


Figure 2: Performance degradation by a faulty transistor

0738-100X/86/0000/0540$01.00 ©1986 IEEE

With the usual assumption of a digital model [1] [2] the faulty behaviour can be explained as follows. If the resistance of T_1 is larger than the resistance of T_2, then a permanently closed T_1 changes the CMOS inverter into a pull down inverter. In order to test static CMOS logic, the physical fault model has to be mapped into a logical one. This is neither combinational nor purely digital, since the timing has to be taken into account. Contrary to the fault-free circuit the delay for the high to low transition of the output 0 of the faulty circuit would take more time corresponding to the resistance ratio.

Investigating pass transistor networks in nMOS logic the same results are shown [3]. But in a recent paper it was pointed out, that the LSSD technique is appropriate for circuits in domino CMOS [1]. In section 3 of this report we will show - more generally - , that **neither for dynamic nMOS nor for domino CMOS the commonly used physical fault model will cause any sequential logic.**

In section 4 we discuss, that all non redundant faults of the physical fault model can be detected by high speed testing with test sets generated for a combinational fault model.

In section 5 we present a test tool developed at our institute, which supports variable fault models according to the implemented technology. Here especially random tests and probabilistic testability analysis are emphasized.

2. Dynamic MOS circuits

In order to enhance the speed, to decrease the power dissipation or to reduce the chip area of MOS circuits several techniques to design dynamic nMOS or CMOS logic have been proposed. Our examinations concentrate on the most widely used design techniques of dynamic nMOS and dynamic CMOS. The common part of both is a switch network SN with two terminals S and D. The switches are interconnected at source and drain, the inputs (i_1,..,i_n) are connected to the gates of SN (see fig. 3).

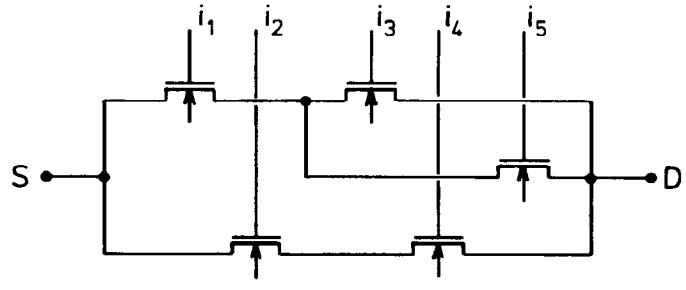


Figure 3: Switching network

The transmission function of SN, $T(i_1, .. , i_n)$, is a Boolean function being true, if a conducting path exists between S and D [5].

The principle of a domino CMOS gate is to control a p-channel and a n-channel transistor by a clock Φ, in order to precharge an internal node y by the p-transistor at Φ and to pull down the node y through the switching network SN and through the n-transistor at Φ. The inverted y is the valid output z (see fig. 4).

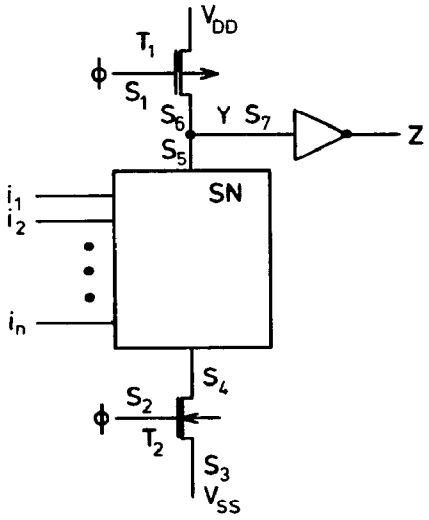


Figure 4: The construction of a domino CMOS gate

The logical function of a domino gate is exactly the transmission function of the involved switching network. A combinational network of domino gates is controlled by a single clock (Fig. 5).

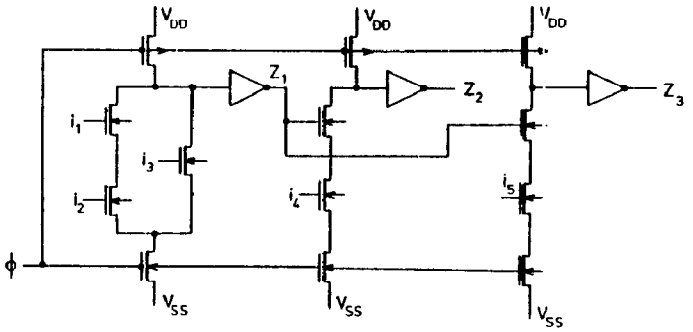


Figure 5: Example of CMOS-domino logic

At Φ the output nodes of all gates are low and thus at Φ each node either can be

pulled up and remain stable or doesn't change at all. This has a significant impact on testability, since races and spikes cannot occur.

A little bit more sophisticated is the construction of the dynamic nMOS gate, since we have to dispense with the p-transistor. Its principle is shown in fig. 6.

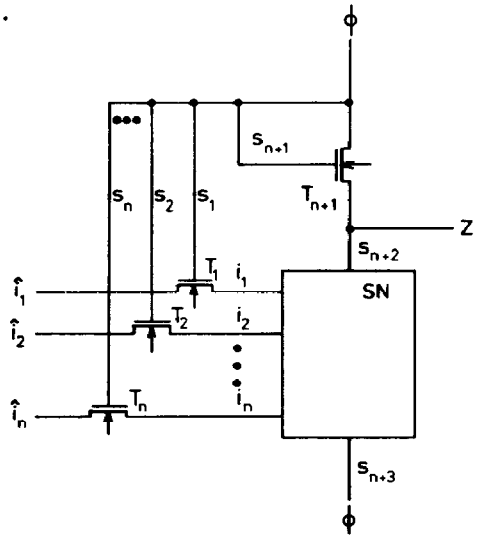


Figure 6: The principle of dynamic nMOS

A dynamic nMOS gate can be regarded as a conventional pull down network, where the terminals are not connected to source and drain but to the same clock Φ. The inputs are also controlled by that clock. If the clock Φ is is active, the output z is precharged and the input nodes may be charged according to their logical values. The transition of Φ from high to low turns T_{n+1} "off" and z will be pulled down, if the transmission function of SN is true. It should be noted, that the logical function of the gate is the inverse of the transmission function.

Obviously the inputs of the gate are blocked when the output z is valid. Therefore one has to use at least two non-overlapping clocks in order to build a combinational network by dynamic nMOS gates (fig. 7).

3. The fault model

The commonly used physical fault model for basic logical cells includes:

- a connection is open
- a transistor is permanently open
- a transistor is permanently closed

In the following we point out, that none of those faults will cause a "sequential behaviour" of a gate. More exactly, even at a faulty gate the valid output at the time Φ_{i+1} depends only on the inputs at time Φ_i.

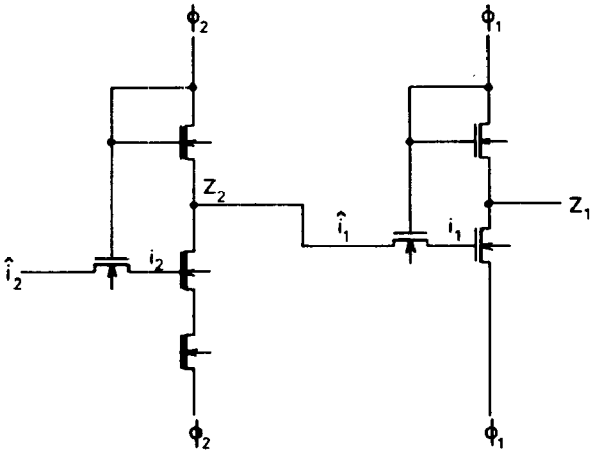


Figure 7: A combinational network in dynamic nMOS

Our considerations are based on the following assumptions:

A1: An open gate, which has no possible connection to power, has the logical value "low".

This assumption is based on measurements indicating that open gates loose their charge during operation [6].

A2: Test patterns have already been applied, which would charge and discharge each node within a fault free circuit.

This assumption can be justified by an appropriate test strategy (see section 4). In general both A1 and A2 are fulfilled by applying some random patterns during a few milliseconds.

By these assumptions it can be proved, that the transmission function of the switching network remains combinational for all faults involved [2]. Thus we can restrict our analysis on the specific dynamic parts of both type of circuits.

Dynamic nMOS:

Before defining the different fault classes we note, that the inputs of a gate clocked by Φ are precharged by the complementary clock (see. Fig. 7). Let us assume, we examine a gate clocked by Φ_1, and the complementary clock is Φ_2. According to figure 6 we can distinguish the following faults:

(Definition principle:

nMOS - 1, nMOS - 2, ... , nMOS - n:
⇒ Transistor 1,2,...,n open

nMOS - n+1,nMOS - n+2,...,nMOS - 2n:
⇒ Transistor 1,...,n closed

nMOS - 2n+1:⇒ Transistor n+1 open

nMOS - 2n+2:⇒ Transistor n+1 closed)

nMOS-i, i=1,..,n: The transistor T_i is permanently open.

According to the assumption A1 the node i_i is not charged. This fault will appear as a stuck at 0 at i_i ($s0\text{-}i_i$).

nMOS-(n+i), i=1,..,n: The transistor T_i is always closed.

During Φ_1 there is permanently a conducting path from i_i to i_i, and also during $\bar{\Phi}_1$ the clock Φ_2 will become active and will charge i_i. Thus the output z will be discharged if we have $t(i_1,..,1_i,..,i_n) = 1$. Consequently this fault is a $s1\text{-}i_i$.

nMOS-(2n+1): T_{n+1} is permanently open.

According to A2 z was already discharged at least one time. In spite of the fault this can be done successfully. But if z is low once, then the valid output at time Φ_1 can never be pulled up. The reason is, that z can only be charged by the network SN at the time Φ_1. But at the time Φ_1 the values of i_i are valid, SN is also conducting during Φ_1 and a is pulled down. Thus the fault is a s0-z.

nMOS-(2n+2): T_{n+1} is permanently closed.

Then there is a conducting path from drain to source. That means that the valid output at time $\bar{\Phi}_1$ is low. Thus the fault is a s0-z. So we have a very interesting fact where both cases - namely the precharge is open and the precharge transistor is closed - results in the same fault s0-z.

Open connections at S_{n+2} or S_{n+3} will cause a s1-z. Open lines at the input gates of S_i, i=1,..,n+1, have the same effect like an open transistor T_i. Open drain-source connections in SN also remain combinational. Since there are already results available [2] this case need not to be treated here.

Domino CMOS:

For CMOS-domino logic or SCVS-circuits some work has already be done [4] [7], where the difficulties caused by the first of the following faults are also mentioned:

CMOS-1: T_2 (fig. 4) is permanently closed.

This fault cannot be modeled at the usual level, since during $\bar{\Phi}$ all i_i (normally outputs of other Domino gates) are low and thus also $t(i_1,..,i_n)$ and no conducting path from S_7 to V_{ss} exists anyway. But because of the different propagation delays for the input signals of SN, the exact behaviour of the gate is not determinable. The fault may remain undetected since the transistor T_2 is not always necessary for logical but for timing reasons.

CMOS-2: T_2 is permanently open.

Then s_7 is never pulled down: s0-z.

CMOS-3: T_1 is permanently closed.

We have to distinguish two cases:

a) Resistance of T_1 << resistance of T_2 + resistance of SN:
Then S_7 will not be pulled down and therefore we have an s0-z.

b) Otherwise.
In this case S_7 needs more time (perhaps infinite) to be pulled down. Applying maximum speed testing may detect this fault as an s0-z. Some details are deferred to section 4.

CMOS-4: T_1 is permanently open:

Then S_7 has never been precharged and according to A1 this is a s1-z.

Open connections at the lines S_1, S_6 or S_7 are equivalent to fault CMOS-4, open connections at S_2, S_3, S_4 and S_5 belong to fault CMOS-3. An open at line i_i, i=1,..,n, results in one or more open transistors within SN and is already examined.

The output inverter has already been discussed in section 1 (see figure 2). If the p-transistor is permanently open, we have a s0-z. A permanently open n-transistor causes a s1-z by A2. If one of the transistors is permanently closed, the considerations of case CMOS-2 yield too.

4. The impact on test tools

Up to this point we have shown the following:

a) There is no fault, that changes a combinational behaviour into a sequential one for the investigated dynamic MOS circuits.

The classical test tools like fault simulation, testability analysis or test pattern generation, which work for ordinary pull down nMOS, are based on the injection of combinational faults. Thus those tools work for all digital faults within dynamic nMOS or domino CMOS as well, and for the remaining faults we will propose appropriate test strategies.

b) There exist some faults within domino gates, which will result in a decrease of the circuit speed. (cf. example in figure 2)

If one of those faults happens, a faulty bridging between power and ground is stated. It is proposed, that those shorts can be detected by leakage measurement during testing [8] . But our experiments have shown that it is hard to prove, whether one faulty conducting path within a large scaled integrated circuit leads to a significant and computable rise of the power dissipation.

Instead of leakage measurement we integrate self test features into our design like BILBOs [9] [10] and non-linear feed back shift registers [11] , which can create and evaluate test patterns by maximum speed of operation.

c) In CMOS-domino logic there is one fault, which may be not detectable.

This is the usual problem of redundant circuitry, which is constructed because of timing reasons. The only feasable approach is a most reliable design of the involved transistor, in order to prevent that short [12] .

We remember, that all results are achieved under the assumptions A1 and A2. If a deterministic test set is generated e. g. by PODEM [13] , then these assumptions can be fulfilled by applying the test set exactly two times.

Applying a randomly generated test set, these assumptions are also satisfied with a high confidence. If a fault has a very low fault detection probability, detection by the first pattern must be expected to need a long time. If the fault detection probability is high, there will be a large number of test patterns detecting it. That means that in both cases A1 and A2 hold.

Therefore random tests satisfy the assumptions A1 and A2 per se. Since furthermore random self tests also cover most of the timing faults in contrast to an external test, this test strategy is especially suited for dynamic logic. At our institute some supporting tools have been developed.

5. *Random testing with variable fault models*

The tool PROTEST (Probabilistic Testability Analysis) offers several features supporting random testing, its algorithms are presented to some extent in [14] . We will summarize them and concentrate on the new part dealing with variable fault models (fig. 8).

For combinational networks PROTEST determines:

signal probabilities:

The user has to specify for each primary input the probability, that the input is set logical "1" by a random pattern generator (it is usually 0.5). For those given input signal probabilities PROTEST estimates the signal probablitiy at each internal node.

fault detection probabilities

Again the user has to specify the input signal probability created by his random pattern generator. Then for each fault the probability is estimated, that it is detected by a random pattern.

test lengths

The user wants to know how many random patterns he has to apply in order to detect all faults. He specifies the input signal probabilities and the demanded confidence of the random test, and PROTEST computes the necessary test length.

optimized input signal probabilities

For each primary input a specific signal probability is computed, promising a increase of fault detection and a decrease of the necessary test length. Using those optimized input signal probabilities, the necessary test length can be reduced by orders of magnitudes [11] [14] .

optimized random patterns

Random patterns with distributions proposed by PROTEST are created.

static fault simulation

Since we are only dealing with combinational networks, a static fault simulation is sufficient, if the user wants to validate the predictions of PROTEST, before integrating some self test logic into the chip. Fault simulation using optimized random patterns can be as efficient as deterministic test pattern generation [15] .

The input of PROTEST is a circuit description and a functional description of the used logic cells .

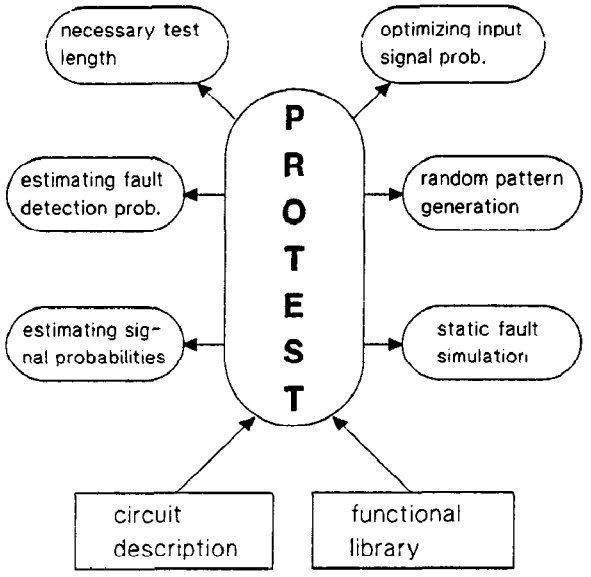


Figure 8: PROTEST

In the following we are concerned with the functional library, which must contain the fault free functions and all possible faulty functions of the used cells. All these functions are automatically generated using both a structural and a behavioural description of the cell. The cell description consists of:

1) Technology dependent parameters:
   -- nMOS pull-down network
   -- static CMOS
   -- bipolar
   -- dynamic nMOS
   -- domino CMOS

2) The list of cell inputs

3) The name of the cell output

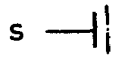
4) The description of the switching network

5) The assignment of the transmission function or its inverse to the cell output

This is transformed into a PASCAL function, performing the fault or the faulty function according to the fault model. For pull-down und dynamic nMOS and for domino CMOS the presented models are used. For bipolar and static CMOS we use the common stuck-at fault model. CMOS logic needs some modifications of the test pattern sets [16] , in order to ensure, that the possible transitions of each node are tested.

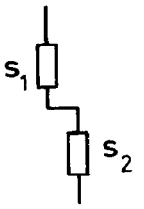
The switching network is described in an elementary way:

s primitive:



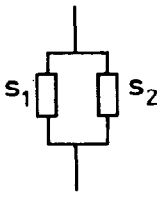
s

s_1 and s_2 are connected in series:



s := s_1*s_2

s_1 and s_2 are connected in parallel:



s := s_1+s_2

Example

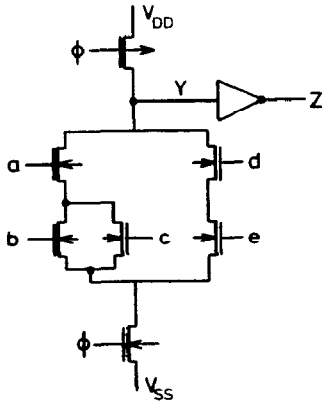


Figure 9: A domino MOS gate

The gate of fig. 9 is described as follows:

```
TECHNOLOGY domino-CMOS;
INPUT a,b,c,d,e;
OUTPUT u;
x1 := a*(b+c);
x2 := d*e;
u := x1+x2;
```

The logical cell has the following distinguishable fault classes:

Class	Fault	Faulty function
1	a closed	u = b+c+d*e
2	a open	u = d*e
3	b closed c closed	u = a+d*e
4	b open	u = a*c+d*e
5	c open	u = a*b+d*e
6	d closed	u = a*b+a*c+e
7	d open e open	u = a*b+a*c
8	e closed	u = a*b+a*c+d
9	CMOS-2 CMOS-3	u = 0
10	CMOS-4	u = 1

It should be noted, that fault equivalent classes are constructed (i.e. not every fault has to be described in the library, ex. classes 3,7). All created functions have the minimum disjunctive form. The internal representation of a library is a PASCAL program performing the fault free and the faulty functions.

Testability analysis and optimization by PROTEST, its performance and the resulting fault coverage are described in [14]. The creation of the fault library needs only a few seconds for a normal sized gate (less than 12 transistors of the switching net).

Summary

Analysing dynamic MOS logic we have shown, that a combinational circuit cannot be mapped into a sequential one by a fault of the common physical fault model. Some faults may cause lower speed of the gates and must be tested with high clock rates, preferably by self test techniques.

Self test by random patterns is supported by the tool PROTEST. PROTEST generates its logical library from a user given description, and creates automatically the set of all possible faulty functions according to the fault model of the used technology.

Acknowledgement

This research is partially supported by the BMFT (Bundesministerium fuer Forschung und Technologie) of the Federal Rebublik of Germany within the E.I.S. Project (Entwurf integrierter Schaltungen) under grant E.I.S. NT 28234.

References

[1] R.L. Wadsack: Fault Modeling and Logic Simulation of CMOS and MOS Integrated Circuits, The Bell System Technical Journal, Vol. 57, No. 5, May-June 1978

[2] M. Y. Tsai: Pass Transistor Networks in nMOS Technology: Synthesis, Performance, and Testing, IEEE, Int. Symp. of Circuits and Systems, 1983

[3] Chen H.H. et al.: Test Generation for MOS Circuits, Proc.Int. Test Conference, 1984

[4] V.G. Oklobdzija, P.G. Kovijanic: On Testability of CMOS-Domino Logic, Proc. 14th Int. Conf. on Fault-Tolerant Computing, 1984

[5] J. P. Hayes: A Unified Switching Theory with Applications to VLSI Design, Proceedings of the IEEE, Vol. 70, 10, 1982

[6] V. L. Rideout: One Device Cells for Dynamic Random Access Memories: A Tutorial, IEEE Trans. Electron Devices, Vol. 26, 6, 1979

[7] Z. Barzilai et al.: Fault Modeling and Simulation of SCVS Circuits, ICCD 84 Proceedings, 1984

[8] Malaiya, Y. K., Su S.Y.H.: A New Fault Model And Testing Technique for CMOS Devices, Proc. Int. Test Conference, 1982

[9] J. Mucha: Hardware Techniques for Testing VLSI Circuits Based on Built-In Test, Proc. COMPCON 81, Feb. 1981

[10] B. Koenemann, J. Mucha, G. Zwiehoff: Built-In Logik Block Observation Techniques, Proc. IEEE Test Conference Cherry Hill, 1979

[11] A. Kunzmann, H.-J. Wunderlich: Design automation of random testable circuits, Proc. ESSCIRC 1985, Toulouse, 1985

[12] Banerjee, P., Abraham, J.A.: Fault Characterization of VLSI MOS Circuits, Proc. of the International Conference on Circuits and Components, 1982

[13] P. Goel, B.C. Rosales: PODEM-X: An automatic test generation system for VLSI logic structures, Proc. 18th Design Automation Conference 1981

[14] H.-J. Wunderlich: PROTEST: A Tool for Probabilistic Testability Analysis, Proc. of the 22nd Design Automation Conference Las Vegas, 1985

[15] Wunderlich, H.-J.: Generating efficient random test sets (to appear)

[16] R. Chandramouli: On Testing Stuck-open Faults, Proc. 13th Int. Conf. on Fault-Tolerant Computing, 1983