A New Approach for Testing MOS Circuits based on Large Defects

Reinhard Rauscher
University of Hamburg
Dept. of Computer Science
Hamburg, Germany
e-mail: rauscher@informatik.uni-hamburg.de

Bernd Schütz
University of Hamburg
Dept. of Computer Science
Hamburg, Germany
e-mail: schuetz@informatik.uni-hamburg.de

Abstract

This work presents a new approach for testing MOS circuits. The proposed model is based on the idea that for each kind of fault a dedicated set of test patterns can be generated. In particular for the case of large defects, we defined a dedicated fault model and suggested an approach for detecting such defects with a minimum amount of patterns.

1. Introduction

The steady increase of the device complexity of VLSI circuits and reliability requirements in electronic systems make it mandatory to increase the quality of IC-testing.

To this purpose, considerable effort has been made. Instead of the classical stuck-at-fault model more and more approaches based on bridging faults and stuck-open faults became common. This means that not only the logical structure of a circuit has to be considered but also the layout.

Therefore, an automatic test patterns generation procedure requires the following three inputs:

- the logical structure of the circuit,
- the layout, and
- the defects to be detected and the underlying fault model.

Furthermore, the costs for testing are rising: Torino ([10] pp. 98) shows that test costs increased from 20 % of product costs in 1986 to 33 % in 1991. This development results in the demand for testing an ASIC with a small set of test patterns. And the faster the error is found while processing the testsequence the more cheaply it becomes. As an example, a lot of our chips we tested, had large defects which could be detected with a very small set of special patterns.
On one hand increasing test costs imply, that a 100 % fault coverage is impractical in most cases and on the other hand a required coverage of a given percentage of the faults (see e.g. [11]) is not sufficient, as shown in Figure 1. Figure 1 shows a design with an overall fault coverage of for example 95 %. But assume, that nearly the whole chip area is easily testable, while there exists a small area, in which no faults could be detected. A high fault coverage in the traditional sense does not guarantee also a high fault coverage for parts of the design.

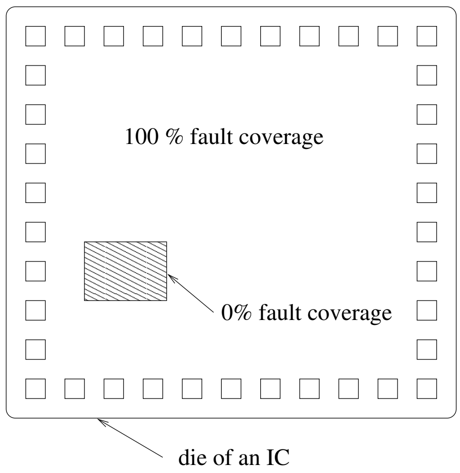

Figure 1. Problem of a high fault coverage

2. Approach for a new test procedure

To overcome these problems a dedicated test procedure has been developed. This is illustrated in Figure 2, which shows, that for each dedicated fault model a dedicated set of test patterns would be desirable, which would detect ”enough” of the defects assumed. Obviously, for each of these sets of pattern a different pattern generation procedure could be used.

This approach seems to be necessary because of the fact that there is a wide variation in defect appearances, particularily in the characteristics like size, shape, and location [1].

The advantage of this approach is, that the order of the different tests *and* the maximal amount of test patterns for each test procedure can be adjusted for each problemclass. This means that any known probability of different faults (depending on the fabrication process, technology, etc.) can be considered. Obviously, defects with a large occurance probability should be tested first.

Another advantage can be seen in collecting information about process dependent defects [4].

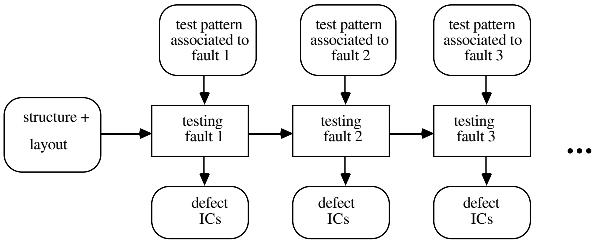

Figure 2. Fault oriented analysis

3. Approach for a dedicated failure

In the literature (e.g. [6, 9, 3, 5]) there exists a lot of work focussing on different defects and also on different fault models.

All of these approaches deal with small defects, which lead to a defect on one single wire or at most on two (in the layout) neighboured wires [8]. Instead of this, our approach aims to detect quite large defects. An example for such a large defect is given in [1] page 26. Additionally, optical controls of our ASIC designs showed, that the percentage of large defects should not be neglected. An optical inspection searching for large defects has also been shown to be quite expensive [1].
Considering the following demands:

- 'rough' testing using only a few test patterns (for early detection of some of the defect chips),
- fault detection in all parts of the chip area (see Figure 1), and
- detection of large defects,

we developed an approach which fulfills the requirements and also could be used in one of the first test procedures shown in Figure 2.

The underlying fault model of a so called 'large defect' is as follows:

A fault is caused by a physical defect in the surface for a destinct layer, in whose expansion for the simplification of the model a max. fitting in circular area with radius R is fit in (see Fig. 3). As an example, such a fault could be caused by a dust particle.

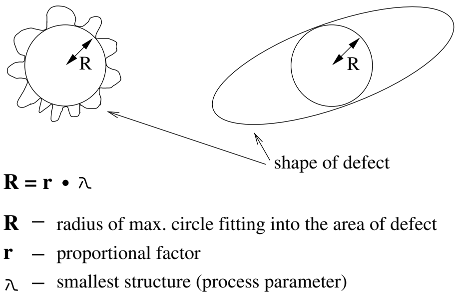

Figure 3. Illustration of 'large defects'

The basic idea of this new approach is, that the value of R is quite large in comparison to λ, so that there would be a couple of connection destroyed in case of a defect. So, instead of analyzing each net in the layout for stuck at faults, it would be sufficient to analyze only a few nets.

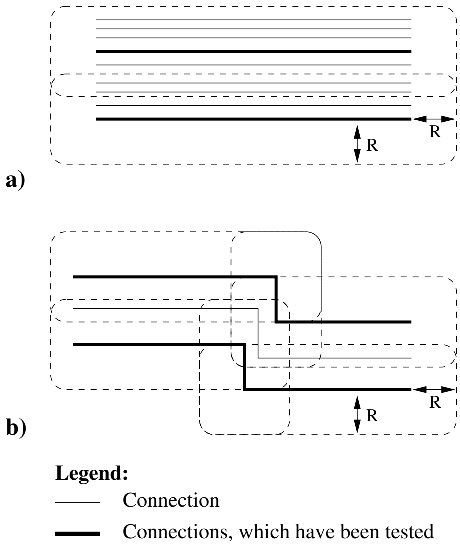

Figure 4. Illustration for the 'large defect model'

This is illustrated by Figure 4. Figure 4 a) shows two lines (bold lines), which could be tested by a given set of pattern. If only 'large defects' with a minimal size of R have to be analyzed, all connections between the two - bold lines - lines can be assumed to be faultless. This can be proven as follows: In the case where a defect with at least

radius R could affect any of these lines, at least one of the bold lines would be affected, too.
In the following, a wire is named *covered*, if along its whole length it is close (maximal distance R) to already tested wires.
A similiar situation to Figure 4 a) is shown by Figure 4 b). The only difference is, that both wires contain a dogleg.

This approach is based on the well known stuck-at fault model, but includes two extensions. The first one concerns the additional analysis of the layout, the second one the minimal size of the defects. The advantage is, that all usual test pattern generation procedures for stuck at faults could be used for the determination of the set of test patterns.

The amount of wires covered by the large defect of size R depends on r (see Fig. 3) and basically on the scale of integration. And this factor r is supplied to our system as an important parameter. Due to further increase of scale of integration also the value of r will increase in the future (see e.g. [2]).

4. Test pattern generation

The procedure of the test pattern generation is shown in Figure 5. In the initial state, no test pattern have been generated. Obviously no faults could be detected till now. This is illustrated at the top of Figure 5 (initial state). After generation of a first set of test patterns, the chip area covered by this set is shown in the middle of Figure 5 (state 1). An evaluation procedure determines the percentage of connections covered by the actual set of test patterns. In case this value is not sufficient, additional test patterns are generated.

The procedure stops if any of the following conditions becomes true:

- The percentage of connections covered exceeds a given minimal value $minper$.
- The percentage of wire segments covered exceeds a given minimal value $minseg$.
- There is no square of a minimal size $minsize$ remaining which contains only uncovered connections (see Figure 1).
- The amount of patterns exceeds a given maximal value $maxpat$.

4.1. Evaluation procedure

The evaluation procedure consists of an additional 'layer' (named *addlayer*) that determines the part of the chip area, which is covered by the already tested connections. A graphical illustration of this is given in Figure 5. In a second step, either the area not covered by the additional layer or the percentage of connections not covered is determined.

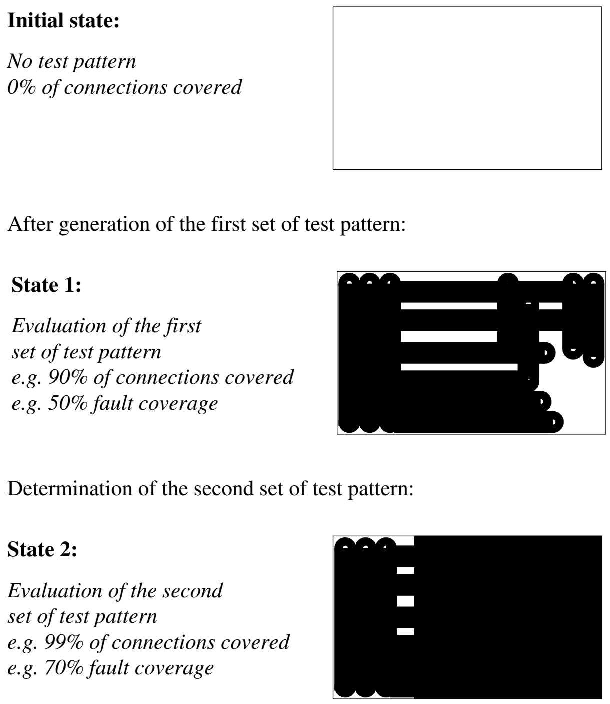

Figure 5. Illustration of the iterative generation of a set of test patterns

4.2. Pattern generation procedure

The pattern generation procedure utilized in our package is the usual procedure for the generation of test patterns for stuck at faults. In the first step, a randomly generated test pattern set [12] could be used. After each cycle of the evaluation procedure, a set of wires has to be determined in order to increase the area of the additional layer. This is done by successively analyzing each connection not yet covered. Obviously, wires can exist, which are not testable (caused by redundancy) and wires may exist, which are difficult to test. Using a simple testability measurement procedure we choose especially such wires, which are easy to control and to observe. For these wires the test patterns are generated. After this, the faults detectable by the current pattern are determined, and the additional layer will be increased. The program flow is illustrated in Figure 6.

5. Implementation

The implementation is given in the following pseudo code:

1. Initialization of $addlayer$, $minsize$, $minper$, $minseg$, $maxpat$.

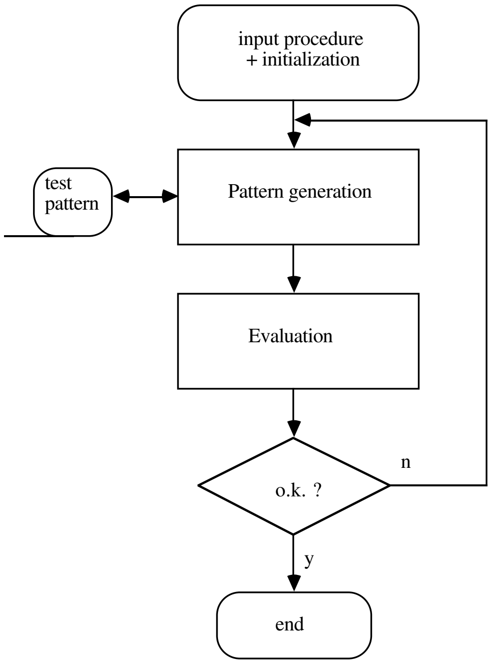

Figure 6. Program flow

2. Selection of a wire w_act to be analyzed. Next, determination of a well suited test pattern for the selected wire.

3. Evaluation procedure including modification of $addlayer$ (the additional layer)

4. Demands fulfilled ?
   If not goto 2.

5. End.

5.1. User interface

The user has to specifiy the following input parameters: $minsize$, $minper$, $minseg$, $maxpat$.

The output produced by our system is the layout including the additional layer and some statistical informationr, concerning the amount of patterns, the amount of wires covered by the additional layer, and the usual fault coverage.

6. The underlying model

The goal of our approach is to keep the problem environment for the programpackage as realistic as possible but eliminating unnecessary details. So, we used an academic model described in [7].

Consequently, we decided to eliminate all fabrication specific details. In this way the system itself becomes independent from a specific fabrication process and yields longer lifetime without a need for adaption to new technologies.

The information about the library cells is purely geometrical. For example, cells and pads are rectangular areas with destinct height and width and specified number of connections.

This guarantees independence from any specific cell libraries.

Wires are lines of equal width lying on a rectangular grid of nonspecified size. The wire area is discretized as a grid of constant unit distance. The components and the wiring segments are placed within two wiring layers.

Based on the underlying model, it can be assumed that the distance between two nets is normalized to the value 1, the width of a wire is also valued 1.

7. Results

In a first attempt, we analyzed the performance of the approach. It should be noted that the underlying router has not been a Greedy Router. In case that a Greedy Router would have been used the results would be remarkably better — see Figure 7 — than those achieved with our low performance router. For presentation purposes we chose another router to avoid the assumption, that the quality of our approach is coupled with properties of Greedy-Layouts.

Figure 7 contains the results of one series of experiments on small artificial and real chips; the chip code indicates the number of cells and whether it is an artificial chip (a) or part of a real world chip (r). The second column in Figure 7 indicates the number of different nets within the layout, and the third column describes the amount of patterns needed for the actual value of r. In case $r = 0$ (the classical approach) our approach would have the same assumption as usual approaches.

In practice, some pattern are able to detect more than one fault, may be on more than one wire, too. This is an invariant for both approaches (classical approach and our approach). And vice versa for a lot of wires more than only one pattern is necessary for testing the selected wire. For a comparison with usual approaches, it has been assumed that for each selected wire only exactly one pattern would be necessary.

Parameters of the results shown in Figure 7 have been $r = 5$, $minsize = 0$, $minper = 100$, $minseg = 100$, $maxpat = 5000$.

Our results show, that although choosing a small value for r a remarkable reduction in the amount of patterns can be achieved. Especially in case of larger designs a reduction up

name	#nets	#pat5	#pat7
a10	17	7	7
a20	32	16	15
a50	53	44	40
a100	75	55	51
a200	103	92	82
a500	233	208	184
a1000	241	227	200
r45	67	45	41
r246	268	171	140
r276	299	211	177
r690	646	441	369

Figure 7. Results for $r = 5$ and $r = 7$, respectively

to 50 percent could be achieved although using only small values for r.

Parameters of the results shown in Figure 8 have been: *minsize* = *0*, *minper* = *100*, *minseg* = *100*, *maxpat* = *500*. Different values of r concerning the example named "r690" have been evaluated.

r	#patnew
0	646
3	538
5	441
7	369
9	317
11	277
13	257
15	236

Figure 8. Results for example r690

As Figure 8 shows, the amount of patterns needed using our approach decreases remarkably while increasing the value of r. Keep in mind, that a 100% coverage has been demanded for Figure 8.

More interesting seems the case where only a 50 % coverage is necessary, as explained in Figure 9.
Parameters of the results shown in Figure 9 have been: *minsize* = *0*, *r* = *5*, *maxpat* = *500*. The figure shows the influence of the parameter $minseg$, specifying the minimal percentage of covered wire segments. Once more, the first column specifies the name of the design or design part. The second column describes the number of nets as usual. The third to the fifth columns specify the amounts of patterns reaching a coverage of at least 50%, 75% and 90% respectively.

The results show, that in general only a few nets have to be tested for getting a coverage of the wire segments of 50% or more.

name	#nets	#pat50	#pat75	#pat90
a10	17	2	3	3
a20	32	2	5	9
a50	53	5	10	19
a100	75	2	8	21
a200	103	9	21	37
a500	233	18	40	76
a1000	241	18	40	75
r45	67	5	13	23
r246	268	15	45	87
r276	299	20	50	102
r690	646	29	85	174

Figure 9. Results for minseg = 50%, 75%, and 90%

As an example Figure 10 shows the user interface of the display routine with a detailed view and a zoomed view of a chip-layout with 500 cells. This layout has been constructed by the 'Great Deluge' placement heuristic and the new academic routing routine.

8. Further activities

Some extensions of the underlying model and also the corresponding modification of the programs are planned. Firstly, some efforts will be attempted to determine the best suited initial nets (see e.g. Figure 4). Secondly, using the approach, that $minsize$ is of interest, we need an extension for determining whether, within a given rectangle, there is at least one wire segment.

References

[1] P. Chou, A. Rao, M. Sturzenbecker, F. Wu, and V. Brecher. Automatic Defect Classification for Semiconductor Manufacturing. *MVA*, 9(4):201–214, 1997.

[2] B. Dvari, R. Dennard, and G. Shahidi. CMOS Scaling for High Performance and Low Power - The Next ten Years. In *Proceedings of the IEEE*, volume (4)83, pages 595–606. IEEE Computer Society Press, April 1995.

[3] Y. El-Ziq. Automatic Test Generation for Stuck-Open Faults in CMOS VLSI. In *Proc. of Design Automation Conference*, pages 347–354. ACM Press, 1981.

[4] L. Huisman. Yield Fluctuations and Defect Models. *IBM Report*, 1994.

[5] N. K. Jha and S. Kundu, editors. *Testing and Reliable Design of CMOS Circuits*, volume 88. Kluwer Academic Publishers, Boston, MA, Dec. 1996.

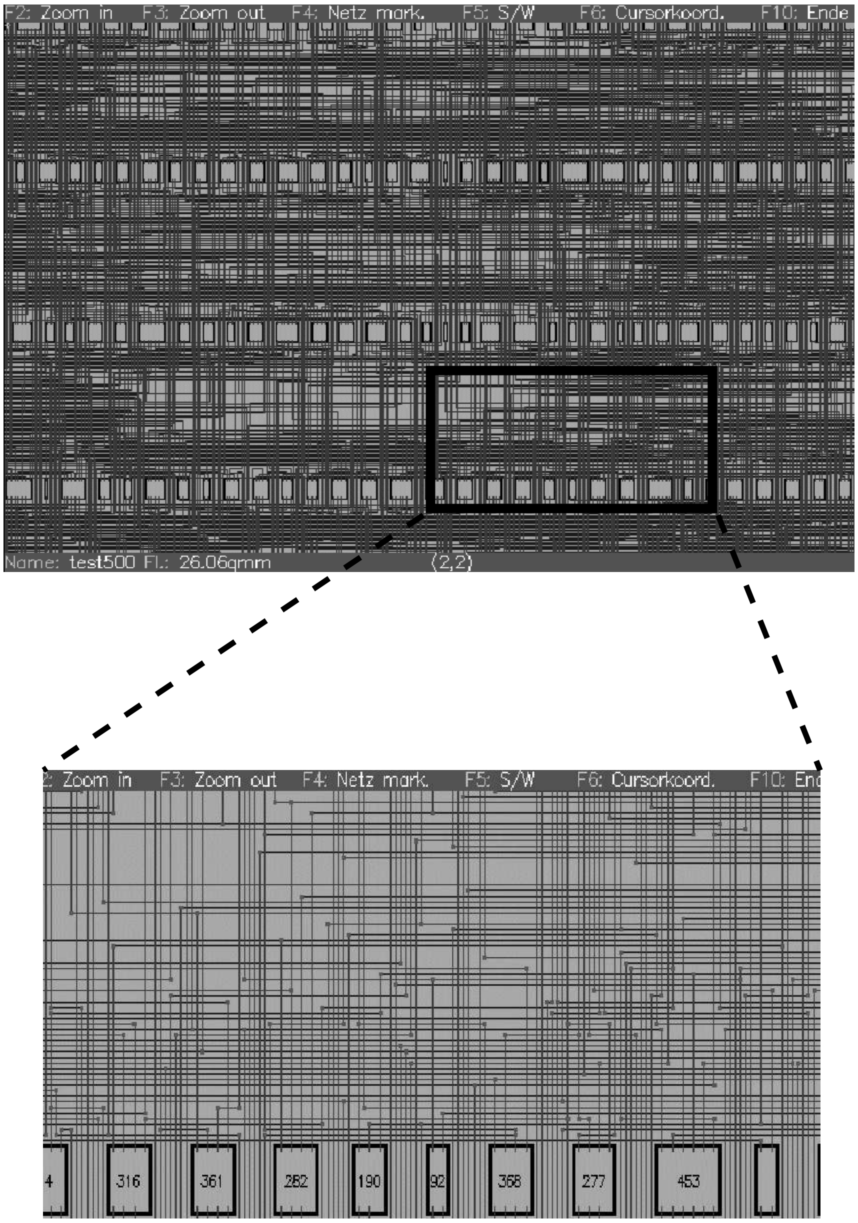

Figure 10. Minor part of a layout and a further zoomed section of a single channel

[6] W. Maly. Realistic Fault Modeling for VLSI Testing. In *Proc. of Design Automation Conference*, pages 173–180. ACM Press, 1987.

[7] R. Rauscher, D. Klawan, and H.-J. Bandelt. Results given by a New Evaluation System for Placement and Routing Heuristics. In M. Kavanaugh, editor, *Proceedings of EUROMICRO 96*, pages 259–266. IEEE Computer Society Press, 1996.

[8] G. Spiegel and A. Ströle. Eine einheitliche Methode zur Extraktion einfacher und mehrfacher Brücken- und Unterbrechungsfehler. In *Proc. of the 7th EIS-Workshop*, pages 119–128, Chemnitz, Nov. 1995.

[9] T. Storey and W. Maly. CMOS Bridging Fault detection. In *Proc. IEEE International Test Conference*, pages 842–951, Washington DC, 1990.

[10] J. Torino, editor. *Managing Concurrent Engineering - Buying Time to Market*. van Nostrand Reinhold, New York, 1992.

[11] R. Wadsack. VLSI: How much Fault Coverage is enough? In *Proc. International Test Conference*, pages 547–554. IEEE-Press, 1981.

[12] K. Wagner, C. Chin, and E. McCluskey. Pseudo-Random Testing. *IEEE Transactions on Computers*, C-36(3):332–342, Mar. 1987.