TPI for improving PR fault coverage of Boolean and three-state circuits*

M.J. Geuzebroek A.J. van de Goor
Delft University of Technology, Faculty of Information Technology and Systems,
Department of Electrical Engineering, Testing Lab,
Mekelweg 4, 2628 CD Delft, The Netherlands
Email: M.J.Geuzebroek@ITS.tudelft.nl

Abstract

TPI can be used to improve the pseudo-random testability of circuit. However, many TPI algorithms are based on COP, which can only cope with Boolean circuits, while in the industry also three-state circuits are found. In this paper the testability analysis method COP and the COP based HCRF TPI algorithm are extended with three-state capabilities, and the HCRF TPI algorithm is adjusted in such a way that better PR fault coverage improvements can be achieved as well as for Boolean as for three-state circuits.

Keywords: Test point insertion, BIST, fault coverage, pseudo-random, COP

1 Introduction

Pseudo-Random (PR) pattern testing is an attractive technique for Built-In Self-Test (BIST) because very little hardware is required for test pattern generation. A Linear Feedback Shift Register (LFSR) can be used to generate the PR patterns.

Although the LFSR has very low hardware overhead, the patterns generated by the LFSR often fail to detect faults known to be *Random Pattern Resistant (RPR)*. RPR faults are faults that can only be detected by a very small set of test patterns. Most faults, the *random susceptible faults*, can be detected by a relative large set of test patterns. Because it is not feasible (with respect to test application time) to generate all test patterns, the number of test patterns generated by the LFSR is often limited to a very small subset. Still it is likely that with this subset the random susceptible faults can be detected, however it is unlikely that this set contains a pattern with which an RPR fault will be detected. Hence, most RPR faults remain undetected, reducing the fault coverage.

One technique that can be used to improve the PR fault coverage, is test point insertion (TPI). By inserting one or more TPs, a large cone of logic is bypassed such that an RPR fault can be covered by assigning only a few inputs, thereby significantly increasing the probability that a PR pattern covers the RPR fault. TPI requires identifying the RPR faults and trying to add as few TPs as possible to improve the testability. TPI algorithms for improving PR fault coverage with BIST can be found in [6, 7, 8, 11, 12, 13, 15].

Several of these TPI algorithms [7, 13, 15] use the *testability analysis (TA)* method COP [1] to find the best TP positions. COP can be used to estimate the probability that a fault will be covered by a PR pattern and hence determine which faults are RPR. This information is used by the TPI algorithms to determine where in the circuit TPs should be inserted, such that the number of RPR faults decreases and the overall PR testability of the circuit improves. COP assumes that the circuit is combinational and that all flip-flops (FFs) are scan-able. Results of the HCRF TPI method of Tsai[13], a COP based TPI algorithm, have shown that the PR fault coverage of a circuit can be increased significantly with a relative small number of TPs.

COP, and hence the TPI algorithms based on COP, can only cope with Boolean circuits. However, in the semiconductor industry also circuits containing three-state elements are found. These three-state elements introduce a high-impedance (Z) or floating state in addition to the Boolean states 0 and 1 [10]. Examples of three state elements are the three-state bus and the switch. Besides the Z value, also the unknown value U is introduced. Unknowns occur in the circuit when circuit inputs have fixed unknown values, e.g., from embedded memories, or when there are bus-conflicts (the bus is driven by a 0 and a 1 at the same time). Not every three-state circuit is suited to be implemented with BIST; it must meet at least the following requirements before BIST can successfully be applied:

1. There are no bus-conflicts in the circuit as bus-conflicts can cause circuit damage and can result in an unknown MISR signature.
2. Outputs that can float or can be unknown should not be

*This work was funded by Philips Semiconductors N.V, The Netherlands

connected directly to the MISR of the BIST in order to avoid an unknown MISR signature.

The MISR [3] is part of the BIST; after applying the PR patterns, the signature of the MISR of a circuit-under-test is compared with the signature of a fault-free circuit to check whether it is fault-free. With an unknown MISR, it cannot be checked whether the signature conforms to a fault-free circuit. Of course is this not allowed for proper BIST implementation. The BIST requirements can be satisfied when:

- all inputs with a possible floating/unknown value, e.g., inputs from embedded memories, are set to a known value by extra test-logic.
- bus-conflicts are avoided, by ensuring that bus-drivers of three-state buses can never be enabled at the same time, regardless of the circuit's input values.
- outputs that can float, e.g., due to floating buses, are pulled-up or pulled-down before their value is shifted into the MISR.
- outputs that cannot be pulled-up/pulled-down and still can float or can become unknown, are not connected to the MISR.

TPI for improving BIST fault coverage is only useful when the circuit is already suited for BIST. Still, the TPI algorithm has to make sure that it remains suited for BIST after TPI. In the remaining part of this paper, it is assumed that the three-state circuits meet these requirements and are suited for PR BIST.

In this paper COP and the HCRF TPI algorithm are extended with three-state capabilities, such that they can also be used on three-state designs. The HCRF TPI algorithm is adjusted such that even better PR fault coverage improvements can be achieved as with the original HCRF TPI algorithm as well as for Boolean, as for three-state circuits.

Section 2 describes COP and how it is extended with three-state capabilities. Section 3 describes how COP, including COP for three-state circuits, is used in the HCRF TPI algorithm. The original HCRF TPI algorithm of Tsai [13] uses a cost function based on the COP measures to find the best TP positions; the TP candidates that reduces this cost function the most, are the TPs that will be inserted. Section 3 introduces our proposed cost function with which even better fault coverage improvements can be achieved. Section 4 shows experimental results of the HCRF TPI algorithm for three-state circuits, and Section 5 concludes this paper.

2 COP for three-state circuits

For each signal line *l* in the circuit, COP [1] provides statistical values for the controllability and observability of

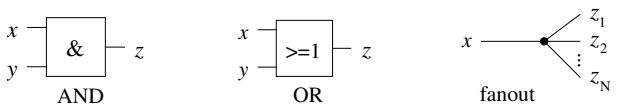


Figure 1. Elements in Boolean circuits

that line. In COP, the **controllability (C_l)** of a signal line *l* is defined as the *probability* that line *l* is 1, and the **observability (W_l)** of a signal line *l* is defined as the *probability* that a value change on *l* will lead to a value change on at least one output.

The output z of the AND-gate in Fig. 1, can *only* become 1 when both inputs x and y are 1. Thus the controllability of output z of an AND-gate can be calculated with:

$$C_z = C_x \cdot C_y \tag{1}$$

The output z of the OR-gate in Fig. 1, is always 1 except for the case that both inputs x and y are 0. Thus the controllability of output z of an OR-gate becomes:

$$C_z = 1 - (1 - C_x) \cdot (1 - C_y) \tag{2}$$

where $1 - C_x$ and $1 - C_y$ are the probabilities that line x, respectively y, is 0.

The input x of the AND-gate in Fig. 1 can only be observed on an output when input y has the non-controlling value 1, and the output z of the AND-gate is observable. Hence, the observability of input x of an AND-gate becomes: becomes:

$$W_x = C_y \cdot W_z \tag{3}$$

For the other standard Boolean gates, The COP controllability and observability equations can be derived in a similar way. The fanout, see Fig. 1 is a special case. It is obvious that all fanout branches have the same controllability as the fanout stem. The fanout stem is always observable, except for the case that none of the fanout branches is observable[1]. This leads to the following two equations for the controllability of fanout branches, respectively observability of fanout stems, where N represents the number of fanout branches:

$$C_{z_1} = C_{z_2} = \cdots = C_{z_N} = C_x \tag{4}$$

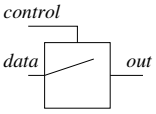
$$W_x = 1 - (1 - W_{z_1}) \cdot (1 - W_{z_2}) \cdots (1 - W_{Z_n}) \tag{5}$$

A stuck-at 1 (stuck-at 0) fault at line l can only be detected when l is 0(1) and l is observable at a circuit output. Therefore the **COP detection probabilities** for the stuck-at 1 and stuck-at 0 faults ($Pd_{l/sa1}$ and $Pd_{l/sa0}$) become:

$$Pd_{l/sa1} = (1 - C_l) \cdot W_l \tag{6}$$

$$Pd_{l/sa0} = C_l \cdot W_l \tag{7}$$

[1] not taking into account possible re-convergent fanout



out		control			
		0	1	Z	U
d	0	Z	0	U	U
a	1	Z	1	U	U
t	Z	Z	Z	Z	Z
a	U	Z	U	U	U

Figure 2. A switch and its truth table

In case of a Boolean circuit, if a line is not one, it is zero. Hence the 0-controllability ($C0_l$) of a line l, i.e., the probability that l is 0, can be calculated with:

$$C0_l = 1 - C1_l \tag{8}$$

as is used in Equations 2 and 6. However, in three-state circuits, this assumption is not longer true. Besides being 1, line l might also be Z or U. Therefore COP is extended with the Z-controllability (CZ_l) and U-controllability (CU_l) of a line l, i.e., the probability that l is floating, respectively the probability that l carries an unknown value. Still the probabilities on all possible values for a signal line should sum to 1. Given this, Equation 8 can been adjusted to Equation 9:

$$C0_l = 1 - C1_l - CZ_l - CU_l \tag{9}$$

In Boolean circuits only the values 0 and 1 are possible, therefore a value change on line l automatically means a 0↔1 change on l. But in three-state circuits, a value change does not automatically mean a 0↔1 change. A Z↔0 or a Z↔1 change on l can also result in a 0↔1 change on an output.[2] Because of the introduction of the Z value, the following three observabilities can be defined:

- $\mathbf{W_l}$: The original Boolean observability, the probability that a 0↔1 value change on line l results in a 0↔1 value change on an output.
- $\mathbf{WZ_l^0}$: The probability that a Z↔0 value change on line l results in a 0↔1 value change on an output.
- $\mathbf{WZ_l^1}$: The probability that a Z↔1 value change on line l results in a 0↔1 value change on an output.

A *U* value will never lead to a valid value change on an output, therefore there exist no observabilities like for example WU_l^0.

An example of the COP measures for a three-state element is given in the following text: In Fig. 2 the truth table for the three-state element switch, i.e., a bus-driver, is given. The COP controllabilities and observabilities can be extracted from the truth table. The table shows that the output is only 0(1) when the $control$ input is 1 and the $data$ input is 0(1). Also can be seen that the output floats (is Z) when at least one of the inputs is Z. The resulting controllabilities are given in the upper part of Table 1. A value change on a switch input can only be observed when this leads to a value change on the switch output. A 0↔1 value change on the $control$ input leads to a Z↔0 value change on out when $data$ is 0, and a Z↔1 value change on out when $data$ is 1. This results in the $W_{control}$ observability equation listed in Table 1. A Z↔0(1) value change on the control input does not lead to a detectable value change on the output. As a result, $WZ^0_{control}$ and $WZ^1_{control}$ are 0. The $data$ observabilities listed in Table 1 are found in a similar way.

[2] It is assumed that only 0↔1 value changes are detectably on an output. Therefore the observabilities of all other value changes on an output are 0.

Table 1. COP equations for a switch

$C0_{Out}=C1_{control}\cdot C0_{data}$
$C1_{Out}=C1_{control}\cdot C1_{data}$
$CZ_{Out}=C0_{control}+CZ_{data}-C0_{control}\cdot CZ_{data}$
$W_{data}=C1_{control}\cdot W_{out}$
$WZ^0_{data}=C1_{control}\cdot WZ^0_{out}$
$WZ^1_{data}=C1_{control}\cdot WZ^1_{out}$
$W_{control}=C0_{data}\cdot WZ^0_{out}+C1_{data}\cdot WZ^1_{out}$

Because of the introduction of WZ^0 and WZ^1, also the detectability probability definitions change. A stuck-at 1 fault at a line l is detectable, not only when l should be 0 and 0↔1 observable, but also when l should be Z and is Z↔1 observable. The detectability probability equations for three-state circuits become:

$$Pd_{l/sa1} = C0_l \cdot W_l + CZ_l \cdot WZ_l^1 \tag{10}$$

$$Pd_{l/sa0} = C1_l \cdot W_l + CZ_l \cdot WZ_l^0 \tag{11}$$

3 HCRF TPI for three-state circuits

3.1 The cost function and cost gradient equations

The goal of TPI is to obtain a maximum improvement in the PR testability of a circuit with as few TPs as possible. It is not advisable to exclusively rely on the COP controllabilities and observabilities, since due to their local nature, these are lacking the capability to analyze and describe the circuit testability problems from a more global point of view. In order to overcome this lack, [4] introduced the *cost function (CF)* and the *cost gradient values*. This CF is a measure for the testability of the entire circuit and is given in Eq. 12,

$$K = \sum_{f=1}^{F} K_f = \sum_{l=1}^{L} \left(K_{l/sa0} + K_{l/sa1}\right) \tag{12}$$

$$K_f = \frac{1}{Pd_f} \tag{13}$$

where K is the CF of the circuit, F is the number of faults, K_f is the cost contribution of fault f, L is the number of signal lines, and $K_{l/sa0}$ and $K_{l/sa1}$ are the cost contributions of the stuck-at 0 and stuck-at 1 faults at line l. Eq. 13

gives the cost contribution K_f for a fault f, where Pd_f is the detection probability of fault f.

The smaller the CF is, the larger all Pd_fs are. A larger Pd_f means a better PR testability; each fault has a better probability that a test pattern that detects the fault, is found in a PR test sequence.

Lisanke [4] defines two cost gradient values dK/dC_l and dK/dW_l as derivatives of the CF with respect to the controllability, respectively the observability, of line l. These cost gradient equations represent the change rate of the cost, due to an infinitely small change of the controllability, respectively observability. Still, these cost gradients are not very good indicators of the testability impact of a TP, as the insertion of a TP always causes the controllability, respectively observability, to change drastically. However, the cost gradients can still be useful, as will be described in Subsection 3.2.

Equation 14 gives the cost gradient equation for a 1-controllability change at the input x_j of a given gate, taken from [4].

$$\begin{aligned}\frac{dK}{dC_{x_j}} &= \frac{dK_{x_j/SA0}}{dC_{x_j}} + \frac{dK_{x_j/SA1}}{dC_{x_j}} \\ &+ \sum_{i,i\neq j}^{inputs} \frac{dK}{dW_{x_i}}\frac{dW_{x_i}}{dC_{x_j}} + \sum_{k}^{outputs} \frac{dK}{dC_{z_k}}\frac{dC_{z_k}}{dC_{x_j}}\end{aligned} \quad (14)$$

In Eq. 14, z_k are gate outputs, and x_i are gate inputs. The first two terms (first line) are due to the cost changes at line x_j itself due to its controllability change. The summations represent a chain-rule and are due to the changes in the observabilities of the other inputs of the gate, and the changes in controllability of the outputs of the gate due to the controllability change at x_j. A more detailed description of all cost gradient equations can be found in [4, 15].

In [4] it was assumed that there are only Boolean values. In case of three-state circuits, the CF not only takes into account the 1-controllability (C or C1) and observability (W), but also the 0-controllability (C0), Z-controllability (CZ), Z↔0 observability (WZ^0) and Z↔1 observability (WZ^1). As a result, also for these controllabilities and observabilities cost gradient values can be calculated, i.e., dK/dC0_l, dK/dCZ_l, dk/dWZ_l^0, and dK/dWZ_l^1. Each cost gradient equations also becomes more complicated due to the new controllabilities and observabilities. Eq. 15, taken from [15], shows for three-state circuits the cost gradient equation for a 1-controllability change at a gate-input x_j. The first two terms in Eq. 15 are due to the cost changes at line x_j itself due to the controllability change at x_j. The summation over all other inputs of the gate are due to the 0↔1, Z↔0 and Z↔1 observability changes at these inputs due to the 1-controllability change at x_j, and the summations over the gate outputs are due to the C0, C1 and CZ controllability changes at the gate outputs due to the controllability change at x_j. A more thorough description how these cost gradient equations have been derived, can be found in [15].

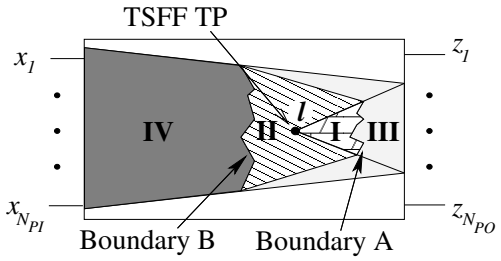


Figure 3. The HCRF TPI method [13]

3.2 COP in the HCRF TPI algorithm

Nowadays almost all circuits use scan-based test, and extra *scan flip-flops (SFFs)* are inserted into the circuit to drive the control points and to capture the output-data from the observation points. As these SFFs are already used to control or capture the data of the TPs, they can also be used as TPs themselves by replacing them by *transparent SFFs (TSFFs)*.[3]. In this paper it is assumed that TSFFs are inserted by the HCRF TPI algorithm. The HCRF TPI algorithm starts with calculating the global cost of the circuit using Eq. 12. For each TP candidate, an event-driven mechanism is used to calculate the impact of the TP candidate on the global cost. This mechanism is illustrated in Figure 3 and is briefly described below; a more thorough description can be found in [13].

Given a TP candidate at line *l* in Figure 3, in Regions **I** and **II** the event-driven mechanism will propagate the COP controllability/observability changes (and hence the cost changes) caused by the TP. When these changes drop below a threshold (Boundaries **A** and **B**), they are not propagated and explicitly recalculated any further. Because the controllability and observability changes have become small (below the threshold), it is assumed that the cost gradient equations are accurate enough to reflect the impact of the TP on the remaining part of the circuit, i.e., the impact on Regions **III** and **IV**. The cost reduction estimate for each TP candidate is calculated and finally the TP candidate with the highest cost reduction is the TP that will be inserted.

3.3 CF for improving PR fault coverage after TPI

The CF given in Eq. 12 has the disadvantage that it only focuses on improving the Pd_fs of the hardest-to-test faults, while other faults with a low Pd_f, but not as low as the

[3] A TSFF acts as a scan flipflop in test operation mode and as a buffer in normal operation mode

$$\frac{dK}{dC1_{x_j}} = \frac{dK_{x_j/SA0}}{dC1_{x_j}} + \frac{dK_{x_j/SA1}}{dC1_{x_j}} + \sum_{i,i\neq j}^{inputs}\left(\frac{dK}{dW_{x_i}}\frac{dW_{x_i}}{dC1_{x_j}} + \frac{dK}{dWZ^0_{x_i}}\frac{dWZ^0_{x_i}}{dC1_{x_j}} + \frac{dK}{dWZ^1_{x_i}}\frac{dWZ^1_{x_i}}{dC1_{x_j}}\right) \quad (15)$$
$$+ \sum_{k}^{outputs}\left(\frac{dK}{dC0_{z_k}}\frac{dC0_{z_k}}{dC1_{x_j}} + \frac{dK}{dC1_{z_k}}\frac{dC1_{z_k}}{dC1_{x_j}} + \frac{dK}{dCZ_{z_k}}\frac{dCZ_{z_k}}{dC1_{x_j}}\right)$$

hardest-to-test faults, are ignored completely. Therefore we propose a CF that also takes into account the other faults with a low Pd_f. This CF is based on the probability of detecting a fault after *NPAT* independent PR patterns. The probability that a fault f is not detected by a single PR pattern, and the probability that f is not detected after $NPAT$ PR patterns are:

$$P(f \text{ not detected}) = 1 - Pd_f \quad (16)$$
$$P(f \text{ not det. after } \textit{NPAT} \text{ pat.}) = (1 - Pd_f)^{NPAT} \quad (17)$$

This probability can also be used as CF for a fault. Faults that have a very low probability on being detected after $NPAT$ patterns, will contribute to the CF with a contribution near 1, while faults with a relative high detection probability will have a cost contribution of almost 0. With this CF, i.e., $NPAT$ CF, the cost contribution of a fault f becomes:

$$K_f = (1 - Pd_f)^{NPAT} \quad (18)$$

As the CF for a fault f has changed from Eq. 13 into Eq. 18, also the cost gradients will change. However, the cost gradients equations, e.g., the cost gradient with respect to a change in 1-controllability given in Eq. 15, will not change, because the CF has not been written in full; both Eq. 13 as well as Eq. 18 can be filled in. For the HCRF TPI algorithm for three-state circuits nothing changes but the CF and cost gradient values.

4 Experimental results

The HCRF TPI algorithm for three-state circuits has been implemented in the *Delft Advanced Test generation system (DAT)* [9]. It has been implemented in such a way that no TPs are inserted at signal lines that can float, as TSFF cannot deal with floating values. The algorithm has been tested upon several ISCAS [2, 5] benchmark circuits, and industrial three-state circuits from Philips, that suffer from RPR faults. Before TPI, all redundant faults have been excluded from the set of faults that define the global cost of the circuit, such that the TPI algorithm does not focus on improving the detectability of redundant faults.

The experimental results are listed in Table 2. The first six circuits are the ISCAS benchmark circuits, and the last six are the three-state industrial circuits. Column *No TPI (FE)* shows the fault efficiency (FE), i.e., the fault coverage of the detectable faults, after applying 32,000 PR patterns to the circuits, listed in Column *Circuit*, when no TPs have been inserted. The number in the circuit name represents the number of signal lines in the circuit. Column *HCRF TPI* shows results after HCRF TPI for three-state circuits has been performed; Column *#TP* gives the number of inserted TSFFs, Column *CPU* shows the CPU time spent on an AMD Athlon 1800+ machine given TPI with the $NPAT$ CF (Eq. 18), and Columns *FE(Org)* and *FE(NPAT)* show the FE with the original CF (Eq. 13), respectively $NPAT$ CF after the application of 32,000 PR patterns.

Table 2. HCRF TPI results

	No TPI	HCRF TPI			
Circuit	FE	#TP	CPU	FE(Org)	FE(NPAT)
c2670	89.08%	1	0.25s	100.00%	100.00%
c7552	97.03%	18	2.76s	100.00%	100.00%
s9234.1	93.49%	18	6.29s	99.76%	99.83%
s13207.1	98.58%	28	18.8s	99.99%	99.98%
s15850.1	96.14%	31	14.2s	99.89%	99.92%
s38417	94.95%	48	37.0s	99.98%	99.98%
Subtotal	95.56%	144	79.3s	99.94%	99.96%
p32118	90.61%	32	30.8s	98.95%	98.59%
p37021	91.78%	37	31.6s	98.99%	98.76%
p114605	90.47%	80	377s	97.66%	98.01%
p137498	95.49%	137	336s	99.55%	99.63%
p481470	83.73%	185	2079s	92.57%	94.10%
p596922	89.57%	317	3044s	96.44%	97.87%
Subtotal	88.30%	788	5898s	95.64%	96.80%
Total	88.72%	932	5977s	95.89%	96.98%

The results in Table 2 show that indeed the PR FEs are significantly improved after HCRF TPI for three-state circuits. The results for the ISCAS circuit show that the HCRF TPI for three-state circuits remains applicable to Boolean circuits. With the original CF already (almost) 100% FE after TPI has been achieved, and there is not much room for PR FE improvement for the $NPAT$ CF. Therefore there is hardly any difference between the FE results of these two CFs for the ISCAS circuits. After the insertion of 144 TPs in the ISCAS circuits, the FE improves from 95.56% to 99.94% with the original CF, and to 99.96% with the $NPAT$ CF. Although this means only 0.02% better FE, still 33% of the detectable faults that were not covered

with the original CF, become covered with the $NPAT$ CF ($(1 - \frac{100\% - 99.96\%}{100\% - 99.94\%}) \cdot 100\%$=33%).

The results of the industrial three-state circuits show that the HCRF TPI algorithm for three-state circuits indeed is applicable to three-state circuits and results in significant PR FE improvement. For the four largest three-state circuits, the $NPAT$ CF results in better PR FE improvement than the original CF. The larger four three-state circuits suffer from faults with very low Pd_fs, i.e., $\ll 10^{-12}$. HCRF TPI with the original CF only focuses on these hardest-to-test faults, while HCRF TPI with the $NPAT$ CF also takes into account faults with less poor, but still poor Pd_fs. By taking into account more faults with poor Pd_fs, HCRF TPI with the $NPAT$ CF is able to reach better PR FE improvement. The smaller two three-state circuits do not suffer from faults with very low Pd_fs and the original CF is capable enough to insert TPs. After the insertion of 788 TPs with the HCRF TPI algorithm circuits in the three-state circuits, the PR FE improves from 88.30% without TPs, to 95.64% after TPI with the original CF, and to 96.80% after TPI with the $NPAT$ CF. In other words, TPI with $NPAT$ CF covers 27% of the detectable faults that were not covered with the original CF ($(1 - \frac{100\% - 96.80\%}{100\% - 95.64\%}) \cdot 100\%$=27%).

5 Summary and conclusions

One of the main draw-backs of BIST is the often low pseudo-random (PR) fault coverage caused by Random Pattern Resistant (RPR) faults. Test Point Insertion (TPI) can be used to improve the PR testability of the circuit such that the PR fault coverage increases. There exist several TPI algorithms that improve the PR fault coverage for BIST. Many of these TPI algorithms are based on the testability analysis method COP. However, COP can only cope with Boolean circuits, hence also these TPI algorithms can only cope with Boolean circuits and cannot be applied to circuits containing three-state elements that are also found in the semiconductor industry. In this paper COP and a TPI algorithm based on COP, i.e., the HCRF TPI algorithm, have been extended with three-state capabilities. Experimental results have shown that with these extensions, HCRF TPI can also be applied to three-state circuits, resulting in significant fault coverage improvements from 88.30% without TPs to 95.64% after TPI.

The HCRF TPI algorithm uses a cost function (CF) to determine the position where to insert a TP. In this paper we have a proposed another CF with which better PR fault coverage improvement can be achieved than with the original CF. TPI with the original CF only focuses on improving the testability of the hardest-to-test faults, while the proposed $NPAT$ CF focuses on improving the testability of all faults which are hard-to-test, not only on the hardest-to-test faults. Experimental results of the HCRF TPI algorithm using this new CF have shown that the PR fault coverages can be further improved; for the three-state circuits they increase from 95.64% with the original CF to 96.80% with the proposed CF.

In this paper it has been shown that by extending the COP based HCRF TPI algorithm with three-state capabilities, and using a new CF in the TPI algorithm to find the best TP positions, significant PR fault coverage improvement can be achieved, both for Boolean as well as for three-state circuits.

Acknowledgments

The authors want to thank the people from Philips Semiconductors and Philips Research for their contribution to this research. Especially for providing us the research topics and the circuits to do the experiments, and for the financial support.

References

[1] F. Brglez, “On Testability Analysis of Combinational Networks”, *Proc. of Int. Symposium on Circuits and Systems*, pp 221-225, 1984

[2] F. Brglez, and H. Fujiwara, “A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in Fortran”, *Proc. IEEE Int. Symposium on Circuits and Systems; Special Session on ATPG and Fault Simulation*, 1985

[3] P.H Bardell, W.H. McAnney and J. Savir, “Built-In Pseudo-Random Testing of Digital Circuits”, John Wiley & Sons, New York, 1987

[4] R. Lisanke, F. Brglez, A.J. Degeus, and D. Gregory, “Testability-Driven Random Test-Pattern Generation”, *IEEE Transactions on Computer-Aided Design, Vol. CAD-6*, pp. 1082-1087, 1987

[5] F. Bgrlez, D. Bryan, and K. Kozminski, “Combinational Profiles of Sequential Benchmark Circuits”, *Proc. IEEE Int. Symposium on Circuits and Systems*. pp. 1929-1934, 1989

[6] Y. Savaria, M. Youssef, B. Kaminska, and M. Koudil, “Automatic Test Point insertion for Pseudo-Random Testing”, *Proc. of Int. Symposium on Circuits and Systems*, pp. 1960-1963, 1991

[7] B.H. Seiss, P.M. Trouborst, and M.H. Schulz, “Test Point Insertion for Scan-Based BIST”, *Proc. of 2nd Euro. Test Conf.*, pp. 253-262, 1991

[8] C. Schotten and H. Meyr, “Test Point Insertion For An Area Efficient BIST”, *Proc. International Test Conference*, pp. 515-523, 1995

[9] M.H. Konijnenburg and J.Th. van der Linden, “Data Structures and Algorthims of the Delft Automatic Test Pattern generation system (DAT)”, ”Delft University of Technology, Faculty of Information Technology and Systens, 1996

[10] J.Th. v.d. Linden, “Automatic Test Pattern Generation for Three-State Circuits”, PhD Thesis, Delft University of Technology, Faculty of Information Technology and Systems, Delft, 1996.

[11] N. Tamarapalli, and J. Rajski, “Constructive Multi-Phase Test Point Insertion for Scan-Based BIST”, *Proc. International Test Conference*, pp. 649-658, 1996

[12] N.A. Touba, and E.J. McCluskey, “Test Point Insertion Based on Path Tracing”, *14th IEEE VLSI Test Syposium*, pp. 2-8, 1996

[13] H.-C. Tsai, K.-T. Cheng, C.-J. Lin, and S. Bhawmik, “A Hybrid Algorithm for Test Point Selection for Scan-Based BIST,”, *Proc. of the 34th Design Automation Conf.*, pp. 478-483, 1997

[14] M.J. Geuzebroek, “‘The Deterministic BIST scheme”, *Thesis report*, Delft University of Technology, Faculty of Information Technology and Systems, 1997

[15] M.J. Geuzebroek, “Test Point Insertion techniques”, *Technical report*, Delft University of Technology, Faculty of Information Technology and Systems, 1997