Design Automation and Test in Europe 2003 (DATE03) , pp. 310-315, March 2003.

A Method of Test Generation for Path Delay Faults Using Stuck-at Fault Test Generation Algorithms

Satoshi Ohtake[†], Kouhei Ohtani[‡] and Hideo Fujiwara[†]

[†] Graduate School of Information Science, Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara 630-0192, Japan
{*ohtake, fujiwara*}*@is.aist-nara.ac.jp*

[‡] Hyper Device Division, Semiconductor Company, Sanyo Electronics Co., Ltd.
180 Ohmori, Anpachi-Cho, Anpachi-Gun, Gifu 503-0195, Japan
OOTA021828@swan.sanyo.co.jp

Abstract

In this paper, we propose a test generation method for non-robust path delay faults using stuck-at fault test generation algorithms. In our method, we first transform an original combinational circuit into a circuit called a partial leaf-dag *using* path-leaf transformation. *Then we generate test patterns using a stuck-at fault test generation algorithm for stuck-at faults in the partial leaf-dag. Finally we transform the test patterns into two-pattern tests for path delay faults in the original circuit. We prove the correctness of the approach and experimental results on several benchmark circuits show the effectiveness of it.*

1. Introduction

For complex high speed VLSI circuits, delay testing is necessary to reach an acceptable quality level. Until now, many delay fault models have been investigated[1]. The path delay fault (PDF) model[2] is one of the most general models among them because distributed faults along paths can be tested and the delay size of detectable faults is scalable. However, the disadvantage with the PDF model is that the number of faults with respect to the number of gates in a circuit is exponential in the worst case and generating efficient (high fault coverage and compact) tests for faults in a short time is generally hard. In order to avoid the former disadvantage, a technique for selecting paths to be tested for combinational circuits[3] and techniques for re-synthesizing circuits such that the path count is reduced[4, 5] are proposed. The latter disadvantage can be avoided by synthesis-for-testability (SFT) techniques[1, 6], design-for-testability (DFT) techniques[1, 7] and *circuit pseudo-transformation* techniques[1, 8, 9, 10, 11].

In this paper, we target to ease test generation. The SFT and DFT techniques can make a given combinational circuit easily testable at the cost of additional hardware and increased delay. Therefore we aim to make a new test generation methodology for PDFs in combinational circuits within the framework of circuit pseudo-transformation. A circuit pseudo-transformation transforms a given circuit into different one tentatively during test generation. And then tests are generated for the transformed circuit. After that, the generated tests for the transformed circuit are transformed into that for the original circuit.

Several test generation algorithms for PDFs in combinational circuits have been proposed[1]. A two-pattern test must be generated for a PDF to launch a signal transition at the primary input of a path. To deal with signal transitions, multiple-valued calculi, such as 10-value calculus[12] and 13-value calculus[13], are often used in PDF test generation algorithms. Such a multi-valued calculus increases the complexity of the test generation compared to the stuck-at fault (SAF) test generation. However, it was shown by Saldanha et al. that a single SAF test generation tool can be used for the generation of robust tests for a combinational circuit by transforming it into a *rising(falling)-smooth-circuit*[8]. The transformation is a kind of circuit pseudo-transformation. Since SAF test generation tools are mature and highly efficient, it is conceivable that utilizing a SAF test generation tool for PDF test generation is very effective. An alternative method was also proposed by Gharaybeh et al. [10] where a given circuit is transformed into a two-level circuit and then non-robust tests are generated by a single SAF test generation tool. Although these methods can generate delay tests using single SAF test generation tools, they have the following disadvantages. For Saldanha's method, only robust test generation is dealt with. However, it is necessary to generate tests for non-robust testable PDFs to guarantee the quality of circuits. For Gharaybeh's methods, even if the number of target paths for test generation is significantly smaller than the total number of paths, the complexity of their circuit pseudo-transformation depends on the total number of paths in a given circuit.

Majumder et al.[11] considered a one-to-one correspondence between every untestable path in a circuit and a redundant SAF in its *unfolded* circuit. For well-well known path delay fault classifications, they characterized each class in terms of the testability of SAFs in a circuit. The notion of the characterization is also related to this work.

The contribution of this paper is a new test generation method for non-robust testable PDFs using an existing single SAF test generation tool based on circuit pseudo-transformation. Our method deals with non-robust test generation. We use a transformation called a *path-leaf transformation* as circuit pseudo-transformation and we transform a given combinational circuit into a circuit called a *partial leaf-dag*. The complexity of the path-leaf transformation depends on the number of target paths for test generation instead of the total number of paths. In this paper, we theoretically prove the correctness of the method and also show the effectiveness of it by performing several experiments on benchmark circuits. The experimental results show that our method can generate a test set with complete fault efficiency in short test generation time compared to a case using an ordinary commercial test generation tool. Furthermore, the test set generated by our method is significantly more compact than the one generated by the commercial tool.

2. Preliminary

A combinational circuit is composed of standard gates, such as AND, OR, NAND, NOR and NOT gates.

In a combinational circuit C, a path P is defined as an ordered set of gates $\{f_1, f_2, \dots, f_n\}$, where f_1 is a primary input and f_n is a primary output and the output of gate f_i is the input to gate $f_{i+1} (1 \leq i \leq n-1)$. Path P has a delay fault if propagation time of a rising or a falling signal transition through the path exceeds a specified limit. Such a delay fault on a path is called a *path delay fault (PDF)*[2, 1]. A path delay fault on P is referred to as $P \uparrow$ or $P \downarrow$, depending on whether the transition is rising or falling at f_n.

A *controlling value* for f_i is a value at its input that determine the value at the output independent of the other inputs and is denoted as $cv(f_i)$. Inversely, a *non-controlling value* for f_i is a value at its input which is not a controlling value of the gate and is denoted as $ncv(f_i)$. The input of f_i connected from f_{i-1} is called an *on-inputs* of f_i along P and denoted as $on(f_i, P)$. On the other hand, the inputs of f_i other than f_{i-1} are called *off-inputs* of f_i along P and denoted as $off(f_i, P)$.

PDFs can be classified into four categories by the conditions of their off-inputs: (1) *robust testable*, (2) *non-robust testable*, (3) *functional sensitizable* and (4) *functional unsensitizable* [1]. This paper targets (1) and (2) and does not distinguish them. Therefore, instead of using definition of *non-robust off-input* and *non-robust testability* in [1], we redefine as follows.

Definition 1 Let $\{f_1, f_2, \dots, f_n\}$ be a path in a combinational circuit C. Let g_{ij} be a gate such that $g_{ij} \in off(f_i)$. The off-input g_{ij} is called a *non-robust off-input* with respect to an input vector pair $\langle v_1, v_2 \rangle$ for C if $g_{ij}(v_2) = ncv(f_i)$, where $g_{ij}(v_2)$ is the value of g_{ij} when v_2 is applied to C. □

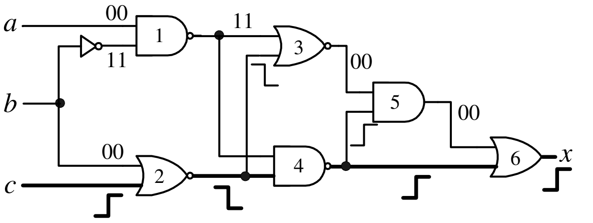

Figure 1. An NRPDF $c246x \uparrow$.

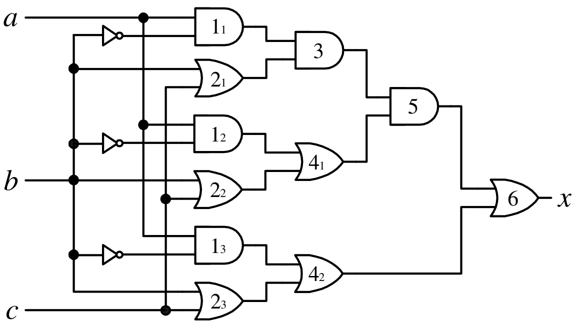

Figure 2. A full leaf-dag.

Definition 2 Let $P = \{f_1, f_2, \dots, f_n\}$ and $P \uparrow$ ($P \downarrow$) be a path in a combinational circuit C and a PDF on P, respectively. The PDF $P \uparrow$ ($P \downarrow$) is called a *non-robust testable PDF (NR-PDF)* if there exists an input vector pair $\langle v_1, v_2 \rangle$ of C such that at each $f_i \in P$, $f_i(v_1) \neq f_i(v_2)$ and $g_{ij}(v_2) = ncv(f_i)$ for each $g_{ij} \in off(f_i, P)$. Such an input vector pair is called a *non-robust two-pattern test*. □

Example 1 A PDF $c246x \uparrow$ shown in Figure 1 is an NRPDF because there is a two-pattern test $\langle 000, 001 \rangle$ and all the off-inputs meet the condition of non-robust off-input.

Definition 3 A *full leaf-dag* is a combinational circuit such that a fanout and an inverter are only permitted at the primary inputs and the output of an inverter is not allowed to have a fanout. □

Notice that full leaf-dag is originally referred to as *leaf-dag* in [9]. However, in this paper, to distinguish from *partial leaf-dag* defined as follows, we call leaf-dag full leaf-dag.

Example 2 A full leaf-dag is shown in Figure 2.

Definition 4 Let $\alpha = \{P_1, P_2, \dots, P_m\}$ be a subset of paths in a combinational circuit C. For each $P_i \in \alpha$, if there is no fanout and inverter on P_i except for the primary input and an inverter is not allowed fanout, C is called a *partial leaf-dag* with respect to α. □

Example 3 Consider a circuit shown in Figure 3. The circuit is a partial leaf-dag with respect to $c2_3 4_2 6x$.

Definition 5 A transformation from a combinational circuit C to a partial leaf-dag with respect to a path P composed of the following two steps is called a *path-leaf transformation* with respect to P.

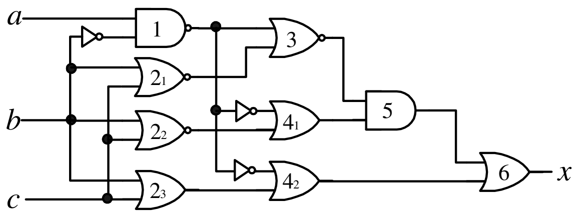

Figure 3. A partial leaf-dag with respect to a path $c2_34_26x$.

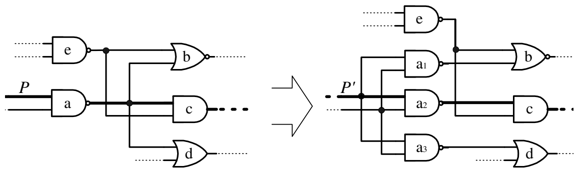

Figure 4. 1st step of path-leaf transformation: a fanout of a is moved to its input by duplicating a to a_1, a_2 and a_3.

Step 1: *Move all the fanouts on P to the primary input.* From the primary output, each fanout on P is moved from an output of a gate to its input by duplicating the gate (see Figure 4.)

Step 2: *Move all the inverters on P to the primary input.* From the primary output, each inverter on P is moved from an output of a gate to its input by replacing AND (resp. OR) gate by OR (resp. AND) gate using De Morgan Law (see Figure 5.)

The path in the transformed circuit corresponding to P is referred to as $L(P)$. □

We consider a combinational circuit C and path P in C. Let C^l_P be a partial leaf-dag with respect to $L(P)$ transformed from C with respect to P. Although the circuit structures of C and C^l_P are different, the functionalities of C and C^l_P are the same. There is one-to-one correspondence between paths P in C and $L(P)$ in C^l_P.

Example 4 We obtain a partial leaf-dag with respect to $c2_34_26x$ shown in Figure 3 by applying the path-leaf transformation with respect to $c246x$ to a combinational circuit shown in Figure 1.

Any combinational circuit can be transformed into partial leaf-dag with respect to its subset of paths by the path-leaf transformation. If we transform the circuit with respect to the set of its all the path, its full leaf-dag can be obtained.

Definition 6 Let C be a partial leaf-dag with respect to α where α is a subset of paths. The *I-edge* of $P \in \alpha$ in C refers to either the connection from primary input if no inverter is there, or else the connection immediately after the inverter. □

The I-edge is originally defined for full leaf-dag by Saldanha et al.[8]. In order to apply this notion to partial leaf-

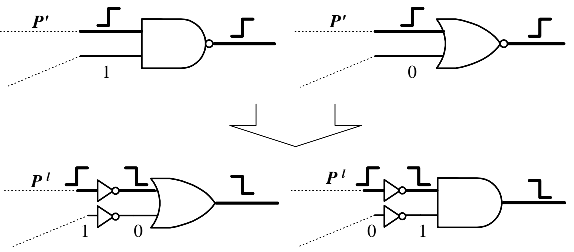

Figure 5. 2nd step of path-leaf transformation: an inverter on a gate is moved to its input by changing the type of the gate.

dag, our definition overrides the definition by Saldanha et al.[8].

In general, to determine a path in a combinational circuit, all the gates on a path must be specified. However, for a full leaf-dag, if an I-edge is specified, the path which contains the I-edge is uniquely determined [9]. For a partial leaf-dag with respect to paths α, $P_i \in \alpha$ is uniquely determined by specifying the I-edge on P_i.

3. Test Generation

In this section, we propose a test generation method for NRPDFs using a stuck-at fault (SAF) test generation algorithm. Given a combinational circuit C and a subset α of paths in C, the method proceeds as follows.

Step 1: We make a partial leaf-dag C^l_α with respect to α by applying the path-leaf transformation scheme to C with respect to every path in α.

Step 2: For C^l_α, we generate tests for SAFs on I-edges corresponding to paths in α using a single SAF test generation algorithm.

Step 3: We transform the generated tests for the SAFs in C^l_α into non-robust two-pattern tests for PDFs in C.

Notice that, even if a PDF is robust testable, our method may generate a non-robust test for the PDF. If robust tests are required for robust testable PDFs, robust tests can be obtained by applying the method proposed by Saldanha et al.[8] prior to our method.

We explain our method in detail in the following subsections. We first define correspondence between a PDF on P in C and an SAF on a I-edge of $L(P)$ in C^l_α. Next, we define a transformation which transforms a test pattern for the SAF in C^l_α to a two-pattern test for the corresponding PDF on P in C. Then, to prove the correctness of the proposed method, we show reducibility between the test generation problem for an NRPDF in C and the test generation problem for the corresponding SAF in C^l_α.

3.1. Correspondence between PDF and SAF

From the structural property of partial leaf-dag, for $P \in \alpha$ if the I-edge of $L(P)$ is specified, $L(P)$ is uniquely determined in C^l_α. In our test generation method, we target to generate a test for an SAF SA0 on the I-edge of $L(P)$ in C^l_α instead of a PDF $P \uparrow$ on P in C. An SAF SA1 on the I-edge is also targeted instead of $P \downarrow$ on P. The reason why we define such a correspondence between the PDF and the SAF is intuitively discussed in the following example. It is formally discussed in subsection 3.3.

Example 5 Consider a circuit shown in Figure 6(a). Figure 6(b) is its partial leaf-dag with respect to $c2_34_26x(= L(c246x))$ by applying the pah-leaf transformation to the circuit (a) with respect to $c246x$. In our test generation method, SA0 (resp. SA1) on the I-edge of $c2_34_26x$ is targeted for the generation instead of $c246x \uparrow$ (resp. $c246x \downarrow$). When a non-robust two-pattern test $\langle 000, 001\rangle$ (resp. $\langle 001, 000\rangle$) is applied to the circuit (a), the faulty behavior of a PDF $c246x \uparrow$ (resp. $c246x \downarrow$) can be observed at the primary output x shown in (c) (resp. (d)). If there exists a PDF $c2_34_26x \uparrow$ (resp. $c2_34_26x \downarrow$) in the circuit (b), when the same two-pattern test is applied, the faulty behavior can also be observed at the primary output x shown in (c) (resp. (d)). If there exists an SAF SA0 (resp. SA1) on the I-edge of $c2_34_26x$ in the circuit (b), when the second vector of the two-pattern test 001 (resp. 000) is applied, the faulty behavior can be observed at the primary output x as 0 (resp. 1). The observed responses of the second vector for both the PDF and the SAF are the same.

3.2. Test pattern transformation

Let v and i be a test pattern for an SAF on the I-edge of $L(P)$ in C^l_α and the primary input of $L(P)$. We transform v into a vector pair $\langle \tilde{v}, v\rangle$ as a non-robust two-pattern test for the corresponding PDF on P in C, where $\tilde{v}$ denotes that $\tilde{v}_i = \bar{v}_i$ for the coordinate i of v and $\tilde{v}$ and $\tilde{v}_j = v_j$ for each coordinate j other than i. Such a two-pattern test is reffered to as a *single input change (SIC)* two-pattern test. The reason why the transformed vector pair becomes a non-robust two-pattern test is described in subsection 3.3.

Example 6 Consider again the circuit shown in Figure 6(a) and (b). If a test pattern 001 (resp. 000) is generated to the SA0 (resp. SA1) on the I-edge of $c2_34_26x$ in the circuit of (b), the test pattern is transformed to a two-pattern test $\langle 000, 001\rangle$ (resp. $\langle 001, 000\rangle$) for the corresponding PDF on $c246x$ of the circuit (a).

3.3. Correctness of the proposed method

In this section, we show the reducibility between the test generation problem for SA0 (resp. SA1) on the I-edge of $L(P)$ in C^l_α and the test generation problem for $P \uparrow$ (resp. $P \downarrow$) on P in C.

Theorem 1 *A vector pair $\langle \tilde{v}, v\rangle$ of C is an SIC non-robust two-pattern test for $P \uparrow$ (resp. $P \downarrow$) on P if and only if there exists a test v of C^l_α for SA0 (resp. SA1) on the I-edge of $L(P)$.*

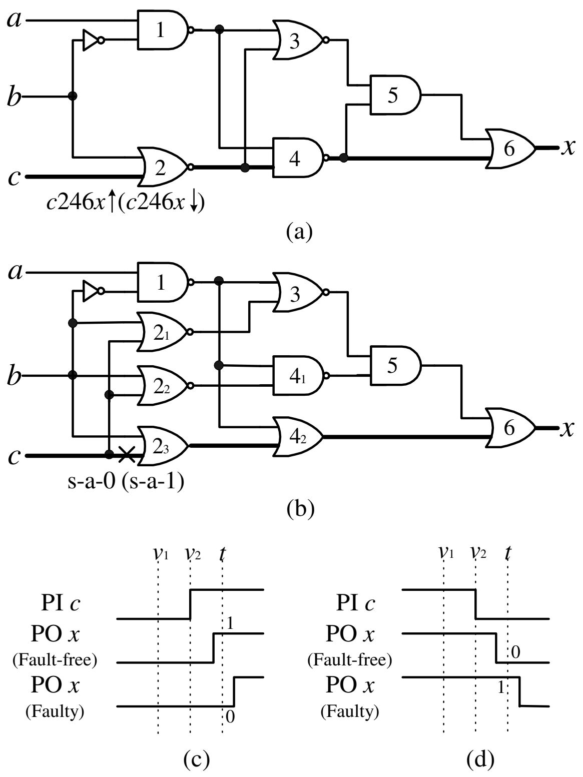

Figure 6. (a) a PDF $c246x \uparrow$ ($c246x \downarrow$) of a circuit, (b) its corresponding SA0 (resp. SA1) fault in the partial leaf-dag of the ciruit of (a) with respect to $c246x$, and (c) and (d) are faulty behaviors of $c246x \uparrow$ and $c246x \downarrow$, respectivery.

Proof: We show that a vector pair $\langle \tilde{v}, v\rangle$ of C is an SIC non-robust two-pattern test for $P \uparrow$ on P if and only if there exists a test v of C^l_α for SA0 on the I-edge e of $L(P)$.

If part: An input vector v of C^l_α is a test for SA0 on e. If v is applied to C^l_α, for each gate $f^l \in L(P)$, $g^l(v) = ncv(f^l)$ because v is a test for SA0 on e, where $g^l = \mathit{off}(f^l)$. If v is applied to C, for each gate $f \in P$, $g(v) = ncv(f)$, where $g = \mathit{off}(f)$, because, from Definition 5, $L(P)$ is obtained by moving all the inverters on P to the primary input of P. Therefore, if a vector pair whose second vector is v is applied to C, all the off-input of P meets the condition of Definition 1. Here, let i be the primary input of P. The primary input of $L(P)$ is also i because the primary input is unchanged by the path-leaf transformation of Definition 5. Since v is a test for SA0 on e, v makes e should be 1 if v is applied to C^l_α. If $\tilde{v}$ is applied as a first vector of the vector pair, the logic value on e becomes 0. Therefore, if the vector pair $\langle \tilde{v}, v\rangle$ is applied to C^l_α, a rising transition is induced at e. The rising transition is propagated to the primary output of $L(P)$ because there is no inverter on the partial path from e to the primary output. From

Definition 5, for each on-input $f^l \in L(P)$ in C^l_α, if a transition $ncv(f^l) \rightarrow cv(f^l)$ (resp. $cv(f^l) \rightarrow ncv(f^l)$) occurs, $ncv(f) \rightarrow cv(f)$ (resp. $cv(f) \rightarrow ncv(f)$) also occurs at the corresponding on-input $f \in P$ in C (see Figure 5). Therefore, if $\langle \tilde{v}, v \rangle$ is applied to C, some transition is propagated through P and the rising transition is induced at the primary output of P. Since the rising transition occurs at the output of P and all the off-input of P meet the condition of non-robust off-input, the vector pair $\langle \tilde{v}, v \rangle$ is an SIC non-robust two-pattern test for $P \uparrow$ on P in C.

Only if part: An input vector pair $\langle \tilde{v}, v \rangle$ of C is an SIC non-robust two-pattern test for $P \uparrow$ on P. If $\langle \tilde{v}, v \rangle$ is applied to C, some transition is propagated through P and the rising transition is induced at the primary output of P. From Definition 5, for each on-input $f \in P$ in C, if a transition $ncv(f) \rightarrow cv(f)$ (resp. $cv(f) \rightarrow ncv(f)$) occurs, $ncv(f^l) \rightarrow cv(f^l)$ (resp. $cv(f^l) \rightarrow ncv(f^l)$) also occurs at the corresponding on-input $f^l \in L(P)$ in C^l_α (see Figure 5). If the second vector v is applied to C^l_α, the logic value 1 is appeared at the primary output of $L(P)$. Since there is no inverter on the partial path from e to the primary output, SA0 in e is activated by applying v. Since all the off-inputs of P meet the condition of Definition 1 when $\langle \tilde{v}, v \rangle$ is applied to C, each off-input of $L(P)$ becomes a non-controlloing value when v is applied to C^l_α. Therefore, the vector v is a test for SA0 on e of $L(P)$ in C^l_α.

We show that a vector pair $\langle \tilde{v}, v \rangle$ of C is an SIC non-robust two-pattern test for $P \uparrow$ on P if and only if there exists a test v of C^l_α for SA0 on the I-edge e of $L(P)$. It is obvious that there exists a similar correspondence between $P \downarrow$ on P and SA1 on e. Thus, the theorem holds. □

Lemma 1 *A PDF $P \uparrow$ ($P \downarrow$) is non-robust testable if and only if there exists an SIC non-robust two-pattern test for $P \uparrow$ ($P \downarrow$) [10].*

The proof of Lemma 1 is available in [10].

Theorem 2 *The test generation problem for $P \uparrow$ (resp. $P \downarrow$) on $P \in \alpha$ in C can be reduced to the test generation problem for SA0 (resp. SA1) on the I-edge of $L(P)$ in C^l_α.*

Proof: From Lemma 1 and Theorem 2, it is obvious that the test generation problem for $P \uparrow$ (resp. $P \downarrow$) on $P \in \alpha$ of C can be reduced to the test generation problem for SA0 (resp. SA1) on the I-edge of $L(P)$ in C^l_α. □

4. Experimental Results

We evaluate effectiveness of the test generation method proposed in the previous section by experiments and discuss the advantages of the method. In the experiments, we used circuits from the ISCAS'85 benchmark suite and the combinational logic of circuits from the ISCAS'89 benchmark suite. Both the ordinary PDF test generation method and our proposed test generation method are applied to these circuits. The PDF test generation tool of TestGen (Synopsys) [14] is used as an ordinary PDF test generation algorithm and the SAF test generation tool of TestGen is used as the SAF test generation algorithm in our proposed method. Both tools are used on Ultra 30 (Sun Microsystems). In the experiments, we did not select paths to be targeted for test generation and thus we targeted all paths in test generation because the selection is not essential for the experiments.

Table 1 gives the results of both the ordinary test generation and our proposed method. In the table, columns in "Circuits" represent test generation results for each circuit. For each row, test generation results for both the ordinary test generation denoted as "Ordinary" and our proposed method denoted as "Proposed" are available.

Row "# Faults" denotes the number of faults targeted in test generation. For the ordinary test generation, TestGen did not list all the faults that should be tested for s1488, s1494, s838.1 and c880. Sub-row "Proposed" shows the total numbers of paths in the circuits. The number is also the total number of SAFs on I-edges in the circuits.

Rows "# Testable Faults" and "# Two-Pattern Tests" denote the number of non-robust testable faults for which two-pattern tests were generated and the number of two-pattern tests. For all the circuits, the ordinary test generation tool achieved complete fault efficiency with respect to the listed PDFs for all the circuits. The SAF test generation tool also achieved complete fault efficiency with respect to all the SAFs on I-edges of all the transformed circuits, that is, our proposed method achieved complete fault efficiency with respect to all the PDFs of all the circuits. The average number of faults detected per a two-pattern test for the ordinary test generation method is 1.03 and that for our proposed method is 6.15. Therefore, we can say test sets generated by our proposed method is more compact than that generated by the ordinary method. We believe that utilizing stuck-at fault simulation induced the good results instead of using more restricted path delay fault simulation.

Row "Test Generation Time (*sec.*)" denotes test generation time in second. The test generation time of our method is shorter than the ordinary one except for c880. For c880, although the number of faults listed by TestGen is 1/4 of the number of faults considered in our method, test generation time of our method is only double of that of TestGen.

Sub-rows "Ordinary" and "Proposed" in "Time Required for Listing Faults (*sec.*)" denote times required for making a file of a path delay fault list for a circuit by TestGen and for transforming a circuit into its full leaf-dag by our method, respectively. Time required for transforming a circuit into its full leaf-dag is shorter that that for listing paths by TestGen for most circuits.

Row "Circuit Size (*#gates*)" denotes the original circuit sizes of benchmark circuits for "Ordinary" and the circuit sizes of transformed circuits by the path-leaf transformation for "Proposed". In this experiments, we did not select the paths to be tested. Therefore, the size of transformed circuits became large compared to the original one. However, it is easily considerable that if the number of paths to be tested in a circuit is a tractable number, the complexity of

Table 1. Experimental results.

		Circuits							
		c17	s386	s382	s526	s1488	s1494	s838.1	c880
# Faults	Ordinary	22	414	800	820	1788	1802	2876	4520
	Proposed	22	414	800	820	1924	1952	3428	17284
# Testable Faults	Ordinary	22	414	734	720	1781	1781	2876	4477
	Proposed	22	414	734	720	1916	1927	3428	16652
# Two-Pattern Tests	Ordinary	22	372	699	698	1754	1754	2876	4432
	Proposed	9	78	92	99	214	205	1144	3451
Test Generation Time (*sec.*)	Ordinary	0.16	1.47	4.25	4.59	20.77	20.51	63.80	121.12
	Proposed	0.15	0.36	1.39	1.53	1.62	1.70	15.45	238.36
Time Required for Listing Faults (*sec.*)	Ordinary	0.04	0.23	0.52	0.53	2.21	2.27	5.07	13.91
	Proposed	0.39	0.40	0.45	0.46	0.50	0.50	0.51	0.79
Circuit Size (*#gates*)	Ordinary	6	159	158	193	653	647	446	352
	Proposed	15	206	494	431	1262	1282	2257	9772

applying the path-leaf transformation is also tractable and thus the size of the transformed circuit is also tractable.

From the above discussion, our test generation method is superior to the ordinary test generation for those benchmark circuits.

5. Conclusion

In this paper, we proposed a test generation method for path delay faults. In the method, a single stuck-at fault test generation tool is used to generate non-robust two-pattern tests instead of a path delay fault test generation tool. The advantage of the method is that non-robust test generation is dealt with and the complexity of the circuit pseudo-transformation in the method depends on the number of target paths for test generation instead of the total number of paths in a circuit. We theoretically proved the correctness of the method and showed the effectiveness of it through experiments. We confirmed that test generation time required for guaranteeing complete fault efficiency and the number of non-robust two-pattern tests to achieve complete fault coverage by using the method can be reduced compared to those by an path delay fault test generation tool.

Our future work is to ease the test generation problem for functional testable path delay faults by using a stuck-at fault test generation tool under the concept of circuit pseudo-transformation.

References

[1] A. Krstić and K.-T. T. Cheng: *Delay fault testing for VLSI circuits*, Kluwer Academic Publishers, 1998.

[2] G. L. Smith: "Model for delay faults based upon paths," in *Proc. of Int. Test Conf.*, pp. 342–349, 1985.

[3] K. Heragu, V. D. Agrawal and M. L. Bushnell: "Statistical methods for delay fault coverage analysis," in *Proc. of VLSI Design*, pp. 166–170, 1995.

[4] A. Krstić and K.-T. T. Cheng: "Resynthesis of combinational circuits for path count reduction and for path delay fault testability," in *Proc. of European Design and Test Conf.*, pp. 486–490, 1996.

[5] I. Pomeranz and S. M. Reddy: "On synthesis-for testability of combinational logic circuits," in *Proc. of 32nd Design Automation Conf.*, pp. 126–132, 1995.

[6] A. K. Pramanick and S. M. Reddy: "On the design of path delay fault testable combinational circuits," in *Proc. of 20th Fault Tolerant Computing Symp.*, pp. 374–381, 1990.

[7] P. Uppaluri, U. Sparmann and I. Pomeranz: "On minimizing the number of test points needed to achieve complete robust path delay fault testability," in *Proc. of VLSI Test Symp.*, pp. 288–295, 1996.

[8] A. Saldanha, R. K. Brayton and A. L. Sangiovanni-Vincentelli: "Equivalence of robust delay-fault and single stuck-fault test generation," in *Proc. of Int. Conf. on Computer-Aided Design*, pp. 418–421, 1992.

[9] W. K. C. Lam and R. K. Brayton: *Timed Boolean Functions: A Unified Formalism for Exact Timing Analysis*, Kluwer Academic Publishers, 1994.

[10] M. A. Gharaybeh, M. L. Bushnell and V. D. Agrawal: "Classification and test generation for path-delay faults using single stuck-at tests," *Journal of Electronic Testing: Theory and Applications*, Vol. 11, No. 1, pp. 55–67, Aug. 1997.

[11] S. Majumder, B. B. Bhattacharya, V. D. Agrawal and M. L. Bushnell: "A complete characterization of path delay faults through stuck-at faults," in *Proc. of Int. Conf. on VLSI Design*, pp. 492–497, 1999.

[12] K. Fuchs, F. Fink and M. H. Schulz: "DYNAMITE: An efficient automatic test pattern generation system for path delay faults," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 10, No. 9, pp. 1323–1335, Oct. 1991.

[13] T. J. Chakraborty, V. D. Agrawal and M. L. Bushnell: "Delay fault models and test generation for random sequential circuits," in *Proc. of Design Automation Conf.*, pp. 165–172, 1992.

[14] Synopsys, Inc.: *TestGen Tool Reference Manual Version 1999.10-TG4.1*, 1999.