Iranian Journal of Science & Technology, Transaction B, Engineering, Vol. 30, No. B2
Printed in The Islamic Republic of Iran, 2006
© Shiraz University

A NEW METHOD FOR IMPROVING PATH DELAY FAULT COVERAGE[*]

Z. BATENI[**] AND H. PEDRAM

Dept. of Computer Engineering, Amir Kabir University, Tehran, I. R. of Iran
Email: bateni@ce.aut.ac.ir

Abstract– In the test of a circuit for faults it is very desirable to have the minimum paths that are to be tested for a complete circuit test. At first, the best test vectors are chosen and then these pairs of vectors are applied to the circuit. Each of these pair of vectors is able to detect a certain number of the faults and the path delay fault coverage for each and all of them may be calculated.

Suitable choice of a path may result in the coverage of the whole circuit and increases detected faults. In this paper, a new method for choosing the minimum suitable paths is introduced. Decreasing the number of test paths causes an increase in the number of detectable faults by a certain number of test vector pairs, hence giving a better fault coverage.

In the presented method, a test path would be selected, if and only if, there exists at least one "two-piece" segment in that path which has not been tested before in the previously tested paths. If the chosen path has at least one "three-piece" (or "more-piece") segment the method will become more complicated. The method is applied to some ISCAS89 benchmarks circuits.

Keywords– Sequential circuit, path delay fault, fault coverage

1. INTRODUCTION

When testing a circuit for faults, a collection of suitable test vectors has to be chosen so that the highest number of faults is revealed .The ratio of revealed faults to the total number of faults is termed "fault coverage". Fault coverage is used as a means for measuring the efficiency of a test, that is, the closer the fault coverage is to 1, the better the chosen test vectors. In other words, testing all the paths in a circuit, path delay fault coverage will become 1 (%100 coverage) [1].

Delay faults or Timing Related Faults cause delay on a gate or a path [2]. In synchronous circuits, these faults cause the chip to be activated at a slower rate, but in asynchronous circuits which do not have clocks, these faults may cause incorrect operation of the circuit.

The delay faults are divided into two groups, path delay faults (PDF) and gate delay faults (GDF) [3-6]. In the gate delay faults, the effects of the faults in one or more gates are considered and the existence of delay in gates is the cause of an increased delay in the output. In the path delay faults the effect of the faults in one path is considered, and the existence of delay in a path increases the output delay [6, 7]. In gate delay faults, the small delay faults may be undistinguishable because the considered gate may be located on a path with low delay [6]. But in a path delay fault model, the delays are considered to be distributed on the path because the gate delay faults are a sub-set of path delay faults [8]. Therefore the path delay faults are more suitable for a delay faults test because they model a larger set of faults [5, 9, 1]. In this paper the path delay faults are considered.

[*]Received by the editors November 30, 2002; final revised form March 9, 2006.
[**]Corresponding author

Usually, a path in the circuit is begun from a flip – flop or primary input, and ends in a flip – flop or primary output [10]. A path may have two different types of delay faults, raising and falling path delay faults [4]. However, the goal here is the evaluation of circuit operation out of the distinct limits of their delay, no matter what the type of path delay fault is. Although a path delay fault model can show the actual fault more accurately, the number of paths in a circuit is increased exponentially for testing. Since in a test a collection of test vectors are used to find the faults in the circuit, and the efficiency of a test is measured by fault coverage, the importance of fault coverage optimization becomes more evident.

There exist different methods to optimize fault coverage in testing path delay faults. In some methods, at first, the longest paths are selected and other paths are omitted then the selected path is tested [1]. In other methods, at first, all paths are selected and then some paths are omitted in different ways that are mentioned in section 2. This omission occurs, provided that the path has been previously tested according to the former test patterns.

In the presented method, efforts have been made to optimize the path delay fault coverage. This method is both faster and more accurate, because it counts and controls two-piece segments instead of counting and controlling the segments one by one. The delay fault path may be chosen as a new path only if it contains at least a two-piece segment which is not included in any of the previously diagnosed delay fault paths. This will result in faster access to appropriate patterns.

On the other hand, this method is more accurate compared to the existing methods because it performs a general comparison among all patterns of the path delay fault. In other words, in case some two-piece segments are located in several common paths, many paths of delay faults can be omitted.

2. EXISTING METHODS OF PATH DELAY FAULT COVERAGE OPTIMIZATION

To optimize fault coverage in testing path delay faults, the existing methods try to omit some of the test paths in different ways. This omission occurs provided that the path has been previously tested according to the former test patterns. These methods are divided into three following types:

a) Pomerans & Reddy method:

The fault coverage optimization method based on the flag usage, in which different paths of the circuit are determined, was presented by Pomerans and Reddy for the first time. Each path may have the length of at least one, and at most, the depth of the circuit. For each segment in the path, 2 flags (ascribed to the rising and falling edges) are assigned [11, 12]. If a path is tested to detect delay fault, one of the attributed flags (the one pertaining to relevant edge) of every segment in the path will become '1'.

The next paths to be tested are chosen under the condition that at least one of their segments has not been previously detected in the other paths. In this method the bigger the path segments, the better the results [13, 14]. In practice this method is rather complicated and has some difficulties. One of which is that sometimes it loses some of the test patterns due to an inadequate amount of flags. For instance in Fig. 1,

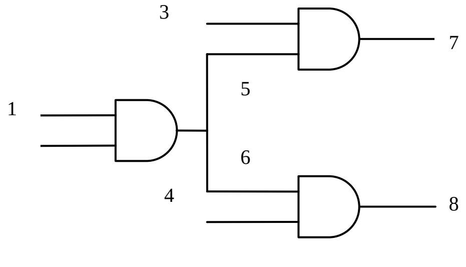

Fig. 1. Logic circuit for Pomerans & Reddy method

assuming that the paths (R-2-5-7) and (R-1-6-8) are tested for rising fault, the segments 1,2,5,6,7,8 are all tested and the rising fault flag of each has become '1'. However, if a test pattern exists for the paths (R-1,5,7) and (R-2,6,8) , it will be eliminated in this method because their flags have already become '1'.

b) Keerthi Heragu method

In another method presented by Keerthi Heragu *et al*., fault coverage optimization is performed based on using flags [14, 12], trying to improve the Pomerans & Reddy method. The procedure is that each segment in the circuit has a set of flags.
For example, for the AND gate of Fig. 2,

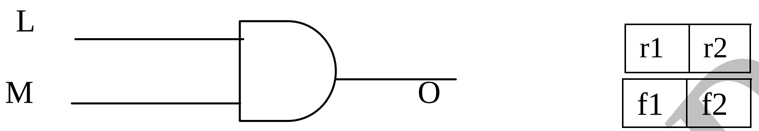

Fig. 2. SET S1 including r1, r2 and SET S2 including f1, f2

L and M are the inputs and O is the output of the gate. The line O has two sets of flags, S1 (for the rising edge) and S2 (for the falling edge). S1 has two different flags, r1 (to clarify that the rising of line O is due to the line L) and r2 (to clarify that the rising of line O is due to the line M). Similarly S2 has two different flags, f1 and f2.

Despite the fact that this method could be more appropriate comparing to the previous method, it sometimes puts a segment in several paths and it may reduce the number of detectable faults [14].

c) The modified Keerthi Heragu method:

In this method, instead of choosing a path that at least one segment has not been tested, the next paths are chosen for the test in the condition that at least one two-piece segment has not been detected in the other paths. The procedure may continue and in the subsequent phase, a path is chosen to be tested only if at least three-piece segments (four-piece segments or more) of it have not been detected in the other paths. The number of chosen segments can be increased to the full length of a path [15, 12].
Supposing that in Fig. 3

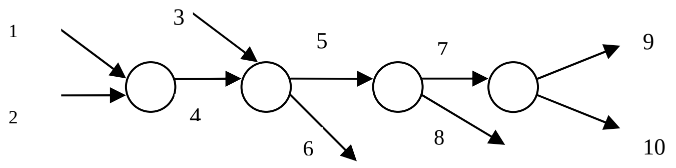

Fig. 3. In this circuit each circle is a logic gate

the path faults (1,4,5,7,10), (3,5,7,9) and (3,5,8) are evaluated and chosen, so that the flags for different lengths may be considered as:

(1) Length 1: 1, 3, 4, 5, 7, 8, 9, 10
(2) Length 2: 1-4 , 4-5 , 5-7 , 7-10 , 3-5 , 7-9 , 5-8
(3) Length 3: 1-4-5, 4-5-7, 5-7-10, 3-5-7, 5-7-9, 3-5-8
(4) Length 4: 1-4-5-7, 4-5-7-10, 3-5-7-9, 3

Based on the above subject, if two-piece segments are checked to choose each path, the flag of the segments 1-4 , 4-5 , 5-7 , 7-10 , 3-5 , 7-9 , 5-8 (expression 2) have become '1' by the previous faults. If three-piece segments are checked to choose each path, the flags of the segments expression 3 will become '1' by the previous faults. For four piece segments, the flags of expression 4 would become '1'.

So if one wants to evaluate the faults pertaining to the paths (1-4-5-8) and (1-4-5-7-9), for each chosen length may have

(5) Path fault (1-4-5-8)
(6) Length 1: 1,4,5,8
(7) Length 2: 1-4, 4-5, 5-8
(8) Length 3: 1-4-5, 4-5-8
(9) Length 4: 1-4-5-8
(10) Path fault (1-4-5-7-9)
(11) Length 1: 1, 4, 5, 7, 9
(12) Length 2: 1-4, 4-5, 5-7, 7-9
(13) Length 3: 1-4-5, 4-5-7, 5-7-9
(14) Length 4: 1-4-5-7, 4-5-7-9

For the path fault (1-4-5-8) (expression 5), if the chosen segments contain two-pieces, since the two-piece segments are 1-4, 4-5, and 5-8 (expression 7) and all of their flags have already become 1 by previous path faults(expression 2), and a path fault (1-4-5-8) will not be chosen for the test.

Similarly, in this path fault (expression 5), if the chosen segments contain three pieces, since the three-piece segments are 1-4-5, 4-5-8 (expression 8) and all of their flags have not already become '1' by the previous path faults (consider three-piece segment (4-5-8)), path fault (1-4-5-8) could be chosen for the test. Also for this path fault, if the chosen segments have four pieces, because the only four-piece segment is 1-4-5-8 (expression 9) and it’s flag has not already become '1' by the previous path faults, it could be chosen for the test.

But for the path fault (1-4-5-7-9) if the chosen segments contain two-pieces, since the two-piece segments are 1-4, 4-5, 5-7 and 7-9 (expression 12) and all of their flags have already become '1' by previous path faults(expression 2), path fault (1-4-5-7-9) will not be chosen for the test. In this path fault (1-4-5-7-9) if the chosen segments have a three-pieces segment, because the three-piece segments are 1-4-5, 4-5-7 and 5-7-9 (expression 13) and all of their flags have already become '1' by the previous path faults(expression 3), path fault (1-4-5-7-9) will not be chosen for the test.

As for the path fault (1-4-5-7-9) if the chosen segments have four-pieces, since the four-piece segments are 1-4-5-7 and 4-5-7-9(expression 14) and their flags have not already become '1' by the previous path faults, path fault (1-4-5-7-9) could be chosen for the test.

Therefore, if the chosen segments contain three-pieces, only the first path fault is valuable, but if there are four pieces in each of the chosen segments, both path faults are valuable and may be used to distinguish the faults. It should be noted that the higher the number of chosen segments, the greater the number of faults can be detected, but the CPU time increases as well.

3. METHOD OF USING THE TABLE

This is an estimating method for decreasing and optimizing the fault coverage. Figure 3 shows an arbitrary circuit graph. To simplify, only the rising transition in the beginning of each path is considered.

In this method, like the previous one, the next path shall be chosen for testing if at least one of its two-piece segments has not been tested before. This condition may be considered for three-piece (or more) segments that increases the accuracy, but make the method more complex. In this paper the condition is considered just for two-piece segments.

If the sequence of choosing path delay fault is as follows:

(15) Path fault (1-4-5-7-9)
(16) Length 2: 1-4, 4-5, 5-7, 7-9

because all the two-piece segments 1-4 , 4-5 , 5-7 , 7-9(expression 16) are being chosen for the first time, their flag has not become '1', and the path fault (1-4-5-7-9) (expression 15) will be chosen. Then the following path fault will be considered:

(17) Path fault (1-4-5-7-10)
(18) Length 2 : 1-4 , 4-5 , 5-7 , 7-10

Because the two-piece segment 7-10 (from expression 18) is being chosen for the first time, its flag has not become '1' and the path fault (1-4-5-7-10) will be chosen. Then the following path fault will be inspected:

(19) Path fault (1-4-5-8)
(20) Length 2: 1-4, 4-5, 5-8

Because the two-piece segment 5-8 (from expression 20) is being chosen for the first time, its flag has not become '1' and the path delay fault (1-4-5-8) will be chosen. Then the following path fault will be examined:

(21) Path fault (3-5-7-9)
(22) Length 2: 3-5, 5-7, 7-9

Because the two-piece segment 7-9 (from expression 22) is being chosen for the first time, its flag has not become '1' and the path fault (3-5-7-9) (expression 21) will be chosen. Now, if the method is performed again, but supposing that this path fault is not chosen for the first time because all of its two-piece segments, i.e. 1-4, 4-5, 5-7, 7-9 (expression 16), have previously been chosen in other three path faults and their flags have become '1', the path fault (1-4-5-7-9) (expression 15) would not be chosen.

To prevent this problem, the following procedure is presented in this paper: First, a table is made in which every column represents sequential two-piece segments of all paths. In this example, the Table columns are shown below

(1-4), (3-5), (4-5), (5-7), (5-8), (7-9), (7-10)

The rows of the table include all path faults, which based on the presented method contain path faults that should be tested.
In this example the rows of the table are as follows:

(1-4-5-7-9), (1-4-5-7-10), (1-4-5-8), (3-5-7-9)

At first, all the boxes of the table are zero. Then for each row (each path fault) the two-piece segments become distinguished and in the same row, the boxes of the two-piece segments existing in the path fault should be changed to '1'.

As an example, in Table 1, the first row (1-4-5-7-9) is path fault and has 1-4, 4-5, 5-7 and 7-9 two-piece segments (expression 16). Therefore in the first row the boxes belonging to the 1-4, 4-5, 5-7 and 7-9 columns should be changed to '1'. The procedure has to be done for all rows of Table 1. This operation is equal to making '1' the (rising or falling) flags of the two-piece segments of each path fault.

Table 1. Values for four paths

	(1-4)	(3-5)	(4-5)	(5-7)	(5-8)	(7-9)	(7-10)
(1-4-5-7-9)	1	0	1	1	0	1	0
(1-4-5-7-10)	1	0	1	1	0	0	1
(1-4-5-8)	1	0	1	0	1	0	0
(3-5-7-9)	0	1	0	1	0	1	0

Suitable choosing of the path faults (the table rows) should be done in a manner that for each chosen path fault , all of the columns which have '1' in the chosen row are covered. In other words, by choosing the path fault the flags of the existing two-piece segments in the path fault, become '1'. So, the columns which have '1' in them must be chosen and the rows that cause the column to be '1' should be considered. If any of these rows is not chosen, the flag of that two-piece segment may never become '1' (This means that it may never be tested). For example, in Table 1 , the column 3-5 has a '1' and the row which made the column 3-5 to have a '1', the row 3-5-7-9 , be considered.

For complete testing of the circuit, the other rows should also be considered to cover all the columns of the table (in order to make '1' the flag of each two-piece segment), in other words, to have a better fault coverage, the least number of rows should be chosen. Therefore those rows have to be selected that have the maximum boxes containing '1', unless they have been chosen before.

In Table 1 the rows (1-4-5-7-10), (1-4-5-8) and (3-5-7-9) are selected due to the columns 7-10, 5-8 and 3-5. Since all the columns (two-piece segments) are covered by these three rows, no other row is needed to be selected (i.e. path fault (1-4-5-7-9)).

For complete testing of the circuit in Fig. 3, all the paths (i.e. the rows) should be entered in the table and all of the two-piece segments as the columns. Then in accordance with each path fault, some of the table boxes (the existing two-piece segments in the same row) take the value '1' (Table 2).

Table 2. Values for all of paths

	(1-4)	(2-4)	(3-5)	(3-6)	(4-5)	(4-6)	(5-7)	(5-8)	(7-9)	(7-10)
(1-4-5-7-9)	1	0	0	0	1	0	1	0	1	0
(1-4-5-7-10)	1	0	0	0	1	0	1	0	0	1
(1-4-5-8)	1	0	0	0	1	0	0	1	0	0
(1-4-6)	1	0	0	0	0	1	0	0	0	0
(3-5-7-9)	0	0	1	0	0	0	1	0	1	0
(3-5-7-10)	0	0	1	0	0	0	1	0	0	1
(3-5-8)	0	0	1	0	0	0	0	1	0	0
(3-6)	0	0	0	1	0	0	0	0	0	0
(2-4-6)	0	1	0	0	0	1	0	0	0	0
(2-4-5-7-9)	0	1	0	0	1	0	1	0	1	0
(2-4-5-7-10)	0	1	0	0	1	0	1	0	0	1
(2-4-5-8)	0	1	0	0	1	0	0	1	0	0

4. REGULATIONS FOR USING THE TABLE

Finding the least number of test patterns is accomplished in several steps.

1. First, a table in which each column represents a sequential two piece segment of all paths is made.
2. The rows of the table include all path faults which are based on the presented method, containing a pair of test vectors that should be tested.
3. For each row, the two-piece segments of the same row are distinguished and the related boxes (containing the two-piece segments) in the same row becomes '1'. (This process is the same as making the flags of the two-piece segment '1').
4. After completing the table, only those rows are chosen in which for every column, there is only one '1' in that column.

5. All the columns should be chosen. In other words, all the two-piece segments paths should be tested, so the other rows must also be chosen to cover all columns of the table. In the meantime, the least possible number of rows have to be chosen for the sake of fault coverage optimization. Thus, the rows which contain the most '1's are chosen, providing that all of two-piece segment paths are tested.

How to perform the table method- In this paper, choosing and performing the appropriate pair patterns has been accomplished using the method presented in [5]. For each path with a known constant length, a flag collection proportionate to two-piece segments of each path is chosen, and their primary value is made '0'. The flags show whether the two-piece segments in this path have already been tested.

In this method, two types of flags are defined: one for the rising transitions and the other for falling transitions (in the beginning of the path). At first both categories of flags for each path fault (for each pair of test pattern) are given the value zero. After it is clarified that in a path a specific delay fault is going to be tested, its flag value becomes '1'.

Since each flag represents one box of the table, to identify it, the location of its column and row must be given. Meaning that the flag (i,j) represents a box from the table that is located in row i and column j. If in this row of the table each column is testable, the flag (i,j) pertaining to it will become '1'. After the completion of the table, these two steps must be performed:

First only those rows are chosen that for some column(s), there is only one '1' in that column and the chosen row. Then, all the columns that have one '1' on the row should be chosen. In other words, all the two-piece segment paths should be tested, so the other rows must also be selected to cover all columns of the table. Meanwhile, the least possible number of rows has to be chosen to optimize the fault coverage. Therefore those rows which are not previously selected should be chosen such that they contain the most '1's. The procedure must be continued until all two-piece segment paths are tested.

5. EXPERIMENTAL RESULTS

The presented method has been experimented on ISCAS89 benchmark circuits and the results are shown in Table 3. The length of chosen segments for each path is two-pieces. In this table, the results of the first two methods, the Pomerans & Reddy method and the Keerthi Heragu method are shown [14].

In Table 4, the results of the modified Keerthi Heragu method [15] and the method presented in this paper are compared. As it may be seen, the number of vectors taken in Table 3 is greater than the number of vectors taken in Table 4, but the fault coverage is still smaller and besides, the CPU time is greatly increased (because the number of the loops that are used for selection of the best path is increased).

The important points in Table 4 are that the first number of vector pairs taken is considerably reduced (if the number of vectors pairs taken is the same as Table 3, the fault coverage will be better), and second a better fault coverage is achieved in comparison to the previous methods.

Table 3. comparison of Pomerans & Reddy and Keerthi Heragu methods

name circuits	Number of test vectors applied	Fault coverage for method of Pomerans & Reddy	Fault coverage for method of Keerthi Heragu
S298	704	% 20.5	% 66.6
S526	1416	% 17.3	% 78.8
S820	1968	% 19.0	% 89.0

Table 4. comparison of modified Keerthi Heragu and presented methods

name circuits	Number of test vectors applied	Fault coverage for method of the modified Keerthi Heragu	Fault coverage for the presented method
S298	94	% 51	% 57
S526	206	% 72.3	% 79.4
S820	405	% 91	% 95

6. CONCLUSION

In this paper, a method to obtain the best "fault coverage" is introduced. In the presented method efforts have been made to optimize the path delay fault coverage. This method is both faster and more accurate, because it counts and controls two-piece segments instead of considering the segments one by one. The delay fault path may be chosen as a new path only if it contains at least one two-piece segment which is not included in any of the previously diagnosed delay fault paths. This will result in faster access to appropriate patterns. However, if the chosen path has at least one "three-piece" (or "more-piece") segment the method will become more complicated.

This method is also more accurate compared to the existing methods because it performs a general comparison among all patterns of path delay faults. In other words, in case some two-piece segments are located in several common paths, many paths of delay faults can be omitted.

Each path is considered to be selected from the table randomly, therefore paths would not have any priority with respect to each other. This method is applied to ISCAS89 benchmark circuits.

REFERENCES

1. Qiu, W., Lu, X., Wang, J., Li, Z., Walker, D. M. H. & Shi, W. (2004). A statistical fault coverage metric for realistic path delay faults. *IEEE VLSI Test Symposium*, 37-42, Napa Valley, CA, USA.
2. Takahashi, H., Boateng, O. & Takamatsu, Y. (1999).A method of generating tests with linearity property for gate delay fault in combinational circuits. *IEICE Transaction Info. and Sys., E82-D*(11).
3. Hulgaard, H., Burns, S. M. & Borriello, G. (1994). Testing asynchronous circuits: A survey. *Technical Report TR94-03-06,* Department of Computer Science and Engineering, University of Washington, Seattle.
4. Lam, W. K., Saldanha, A., Brayton, R. K. & Sangiovanni-Vincentelli, A. L. (1995). Delay fault coverage, test set size and performance trade-offs. *IEEE Transaction on Computer-Aided Design of Integrated Circuit and Systems, 14*(1).
5. Smith, G. L. (1985). Model for delay faults based upon path. *Proc. Int. Test Conf*, 324-349, Philadelphia, PA.
6. Varma, P. & Gheewala, T. (1993). Delay testing using a matrix of accessible storage elements. *IEEE International Test Conference*, 243-252, Baltimore, MD, USA.
7. Wen Wu, C. & Yuang Su, C. (2000). A probabilistic model for path delay faults. *Journal of Information Science and Engineering, 16*(5).
8. Norwood, R. B. (1997). Synthesis for scan reducing scan overhead with high level synthesis. A thesis submitted in UPC for the Ph.D. degree, Dept. of Elec. Eng. and Computer Science, Stanford University, Stanford California.
9. Westerman, G., Kumar, R., Stroud, C. & Heath, R. (1998). Discrete event system approach for delay fault analysis in digital circuit. *Proceeding of the American Control Conference,* 239-243, Philadelphia, Pennsylvania.
10. Scholz, H. N., Aadsen, D. R. & Zorian, Y. (1993). A method for delay fault self testing of macrocells. *International Test Conference*, 253, 261, Baltimore, MD, USA.
11. Pomeranz, I. & Reddy, S. M. (1992). An efficient non-enumerative method to estimate path delay fault coverage. *Proc. of the 1992 IEEE/ACM Int. Conf. on Computer-aided Design*, 560-567, Santa Clara, CA, USA.
12. Pardoi, C. G. (1999). Exact non-enumerative path delay fault simulation of sequantial circuits. *Thesis submitted to the State University of New Jersey for the degree of Master Science*.
13. Hergu, K., Agrawal, V. D., Bushnell, M. L. & Patel, J. H. (1997). Improving a non-enumerative method to estimate path delay fault coverage. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 16*(7), 759-762.
14. Heragu, K., Bushnell, M. L. & Agrawal, V. D. (1994). An efficient path delay fault coverage estimator. *Proceedings of the 31st Annual Conf. on Design Automation Conference,* 516-521, San Diego, California, USA.
15. Heragu, K., Patel, J. H. & Agrawal, V. D. (1996). Improving accuracy in path delay fault coverage estimation. *Proc. 9th International Conference on VLSI Design*, 422-425, Bangalore, India.