Fault Diagnosis in Designs with Extreme Low Pin Test Data Compressors

Subhadip Kundu*, Parthajit Bhattacharya*, and Rohit Kapur†
*Synopsys India Pvt. Ltd., Bangalore, India. Email: subhadip@synopsys.com, parthab@synopsys.com
† Synopsys Inc., Mountain View, CA, USA. Email:Rohit.Kapur@synopsys.com

***Abstract*—Diagnosis plays an important role to ramp up yield during IC manufacturing process. Limited observability due to test response compaction negatively affects diagnosis procedure. With modern compressors - targeting very high test data compression, diagnosis becomes even more complicated. In this paper, a complete diagnosis methodology focussing on a novel mapping algorithm has been described. The mapping algorithm maps failures from compressor pins to scan cells with great accuracy (even in presence of X-bits in the responses), so that, scan diagnosis can be used to find out the actual defects. Experimental results on industrial designs showed that the proposed method almost match scan based fault diagnosis results.**

I. Introduction

When a logic circuit fails a test, fault diagnosis is the process of narrowing down the possible locations of defect. Typically fault diagnosis of scan designs with compressors are performed in two steps. In the first step, failures are mapped back through the compressors to the scan cells, and in the second step, scan based fault diagnosis is performed. Since, diagnosis on scan designs is already an well studied subject, our research is focused on the mapping part of the diagnosis procedure.

Diagnosis algorithms can be classified according to compaction schemes that have been used. There are two fundamental classes of compaction devices that are used in VLSI circuit industry: (i) time compactors and (ii) combinational (space) compactors. Identification of failing scan cells from compacted test responses have been a major research area in a built-in self-test (BIST) environment specially for time compactors [1], [2], [3]. Some of these approaches are capable of identifying error bits up to a pre-specified value. External testing techniques mainly use space compactors which are relatively easy to map. In recent years, advancement of convolutional compactors [4], [5] which are based on a finite-input-response principle, have added a new dimension to scan response compaction. In [6], [7], [8] authors presents a novel accurate and time-efficient fault-diagnosis technique based on identification of failing scan cells for scan-based designs with convolutional compactors. The essence of the method is to use a branch-and-bound algorithm to narrow the set of scan cells down to certain sites that are most likely to capture faulty signals. This search is guided by a number of heuristics and self-learned information used to accelerate the diagnosis process for the subsequent test patterns. The approach uses information gained from diagnosis algorithm using an already mapped pattern. So, it requires an iteration between main diagnosis procedure and the mapping algorithm.

In this paper, we have proposed a new mapping algorithm which can identify the failing scan cells from the compressed responses generated using Scalable Scan Architecture (SAS) [9]. The compressor side of SAS architecture is similar to that of convolution compactors. The proposed method analyzes different cases that can occur in compacted responses and decide different ways to map them back to failing scan cells. For example, faulty compressed responses can consists of effects from a single flop, from multiple flops having independent effects, or from multiple flops with their effects mixed in the compressor. The proposed method can handle all these cases and identify the failing scan cell(s) effectively.

This paper have the following contributions -

- Identification of different cases from compressed responses and have different solutions for each of them.
- A novel branch and bound mapping algorithm based on incremental approach has been proposed which can identify the failing scan cells in a time efficient manner.
- The proposed approach can handle presence of don't cares (Xs) in the compacted response.
- The approach does not require an iteration between main diagnosis algorithm and mapping algorithm. From the failure log, it maps failures back to scan cells for every pattern, and scan diagnosis algorithm can work on it. So, the proposed technique can be used with any standard diagnosis algorithm without any modification.

The difference between the proposed approach to that mentioned in [8] is the selection procedure of a scan cell into the solution set. The proposed approach is also unique in its failure pattern selection strategy.

Rest of the paper is organized as follows: Preliminaries on test response compaction using SAS are given in Section II. In Section III, the proposed approach is explained. Section IV contains the mapping algorithm. Experimental results are given in Section V. Finally, we conclude in Section VI.

II. Scalable Adaptive Scan (SAS) compression

SAS is an hierarchical scan compressor solution which can work with flat Automated Test Pattern Generation (ATPG) techniques. SAS addresses the problem of reducing test data and test application time in a hierarchical and low pin count (single input and single output) test environment. The details about SAS compression schemes can be found in [9]. Compressor architecture of SAS has a similar structural design of

978-3-9815370-4-8/DATE15/©2015 EDAA

an convolutional compactor for efficient response compression. In its typical configuration, a convolutional compactor consists of two parts. The first part consists of single-output XOR compressor networks that are added to the end of the internal scan chain outputs. The compressor is based on Steiner triple systems and is designed to tolerate up to two Xs per shift with no loss of observability on any other chain. In SAS, AND-based X-masking network is also included in the first part. The second part consists of a shift register where the outputs of the compressor networks are connected. The shift registers performs another set of compression over the compressor outputs. Figure 1, shows a typical example of convolution compactors with one output with, six flip-flops driven by six chains.

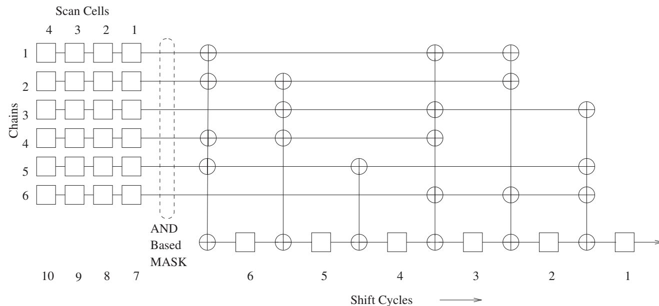

Fig. 1. SAS output side: convolution compactor [9]

A convolutional compactor can support arbitrary compaction rate for any number of outputs. In this paper, however, we will consider single-output compactors only. Convolution compactor and its various connections can be represented in polynomial format - known as injector polynomial. Details about it can be found in [8].

In convolution compactor, every scan cell's error can reach memory elements and visible in the test output in three possible shift cycles. We have called them impacted shift list of that cell. The combination of shift cycle where the effect of a scan cell can reach is unique for every scan cell. So, if a failure is captured in a single cell, it is very easy to identify the failing cell from compressed response. Diagnosis methodology relies on this property to accurately find the failing cell.

III. Diagnosis Methodology: failing pattern type

A fault when present in the circuit, may lead to many failing patterns. However, for effective diagnosis, not all failing patterns are required. We have experimentally found that limiting the failing pattern count, actually helps in accurately pinpointing the fault. This reason behind is aliasing of failure responses caused by the compressor network. Though convolution compactors can minimize aliasing, they cannot eliminate it. Specially in presence of Xs, aliasing is a major concern. Due to the aliasing, the mapping from compressor output to scan cell may not be unique and may lead wrong information to the diagnosis algorithm. Its better not to map these patterns because a wrong mapping will lead to many confusion in scan diagnosis. So, identifying patterns where we can be sure about the mapping procedure is extremely important. Based on this observation, we have analyzed each failing pattern separately and then processed it accordingly.

There can be three possibilities when mapping is performed on failing patterns based on their failure responses. A failure response may contain errors from (i) a single cell, (ii) multiple cells with their effects independent to each other, or (iii) multiple cells with their effect canceling each other at some shift cycles. These three cases need to be identified and treated separately. We know that a scan cell can affect three shift cycles in the failure response. So, if any failure response contains there shift failures, it is more likely that they are coming from a single cell. Since convolution compactor uses a Steiner triple redundant system while designing the compactor, each scan cell has their unique three shift cycles in the output which they can affect. This uniqueness help the mapping algorithm to find the cell. In the second case where two or multiple cells captured failures, the final failure response will consists of multiple of three shift failures. The mapping algorithm needs to identify each cell and map the failure back to those cells. In the last case, mapping becomes complicated because of possibility of aliasing due to fault cancelation. Information gathered from the previous steps is used prune the wrong candidates.

In this work, we have used the same algorithm for all three cases but in different passes. In the first pass, we only look for patterns with three shift failures. It may be possible that two or three scan cell failures of third type may result in a three bit format. But, if the resulting failures match that of a single scan cell, there is no way to distinguish between them. So, its safe to use patterns with three shift failures and map them. In the second pass, we consider patterns with multiple of three fails. Here, we are assuming that the shift cycles of each failing cells do not cancel each other. Since, the uniqueness of individual cell is still preserved, mapping of these patterns are also safe. In the third pass, if we still do not have enough failing pattern which are already mapped, we have tried the difficult third cases. We have used the information gathered from the previous passes to prune out the false candidates and try to map it correctly to the scan cells. Performing mapping over many passes helps us to target the patterns which are easy and safe to map first and use the information with the difficult ones. It minimizes the chances of error mapping and does not have an impact on time at all.

Presence of X's in the failure responses also complicates the mapping procedure. In SAS, though an AND gate based masking circuitry blocks most of the X's before reaching the compressor network, some X's can still be present in the responses. These X's may change the failure responses leading to aliasing. Thus the mapping algorithm should be able to handle X's also. The following section discusses it in detail.

IV. Mapping Algorithm

Our mapping algorithm is a greedy branch and bound based technique. We have used a similar data structure as described in [8]. The data structure is based on an matrix which stored the cell list that can affect a particular shift cycle at compressor

output. From the failing response, first the mismatched shift cycles are found out. Each scan cell has been assigned to a weight based on the number of error bit that can be explained by that cell. The maximum weight of a particular cell is three, since any cell can affect three shift positions. Once, the weight has been calculated for each cell, the cell with maximum weight is selected. Error response has been modified by setting the one bits which are explained by the cell, to zero. If there is any bit which is initially zero in the error response but in the list of the selected cell's impacted shift list, one has been introduced on that bit (if X is not present in that bit). The weight of the remaining cells are calculated base on the modified error response. The procedure repeats until all the bits in the error response are set to zero. The proposed algorithm can also backtrace when it is impossible to select a new scan cell. It returns to the parent node in the tree and tries another cell. This way, it can guarantee to find out a solution.

Each cell actually have three weights associated with them - (i) explain failure bit weight, (ii) X - weight, and (iii) explained pattern weight. Explain failure bit has already been explained. X weight of a cell is the number of bits in the X response that are in the cell's impacted shift list. Presence of X's can change the the failure response as well as the error signature. For example, let there be a single cell failure resulting a three bit failures at the compressor output. But, due to presence of a X, in one shift cycle the failure got cancelled. Now, many cells along with the correct one, will have explain failure bit weight equal to two. But, the actual cell will have X weight which can be useful to identify the cell. This way, X's can also be handled in our mapping algorithm. Explained pattern weight for a cell is the count of pattern in which the cell has been found as a candidate. This actually helps us to prune in case we don't have an unique cell to point out. Algorithm procedure is given in Algorithm 1.

Algorithm 1 Mapping procedure

Require: Failure responses of a test set at the compressor output
Ensure: Failures mapped at scan cells for failed patterns
1: pass = 0,
2: **while** $pass \leq 2$ *and mapped patt* $\leq$ *Pattern Limit* **do**
3:    **for** *pattern = first failing pattern* ***to*** *pattern* $\leq$ *last failing pattern* **do**
4:       total failures = number of failures for the pattern
5:       **if** *((pass == 0) and (total failures* $\neq$ *3))* **then**
6:          continue
7:       **end if**
8:       **if** *((pass == 1) and (total failures%3* $\neq$ *0) and (total failures == 3))* **then**
9:          continue
10:       **end if**
11:       Map the pattern according to branch and bound algorithm described in Section IV
12:    **end for**
13:    pass = pass + 1
14: **end while**
15: **for all** *Failed Patterns that are not mapped* **do**
16:    Pattern type is set to skipped
17: **end for**
18: **return**

The algorithm uses two variables - used and selected list for mapping. Used is a marker for every cell. It indicates whether the cell has already been used in building up the solution or not. If it has, then it is not selected again. Selected list is a stack data structure used to store cells at different depth of the solution. It is very useful for backtracing when it is impossible to select a new scan cell. Then, the scan cell at top of the selected list is removed and the algorithm tries a different path to find out the solution.

The algorithm limits number of mapping cells for a pattern depending on its number of failing bits. Since each cell can have three bit failures, initial value of cell limit is set by dividing the number of failing bit by 3. The mapping procedure tries to find out a solution within this cell limit. If they are unable to do so, cell limit is increased by one and the mapping procedure starts from beginning. We have limited our search upto two more cells from the initial value. This is done to reduce time for mapping. Also, for more cells, chances of aliasing is higher. Thus, the limit is justified. The failed patterns, which are not mapped, are marked as skipped. This way, during scan based fault diagnosis, these patterns are ignored. They are not considered in any further analysis.

The proposed algorithm is quite similar to that of [6], [7], [8], with the difference in selection procedure of the scan cells. In these works, authors have selected scan cell based on the right most one bit in the error response. The choice is actually based on the patterns that are targeted first. The authors have sorted the failing test set according to the number of failed bits of the pattern in increasing order. So, the first targeted patterns have only one failure. But, in practice, single bit failure patterns are actually most difficult and prone to wrong mapping. This is because, a single fail does not have any signature. Many cells can have the failed bit in their impacted shift list. The single bit fail may come from a single cell where the other two bits are masked out by X's, or it can be from multiple cells canceling their effect at some bits. There is no way to prune the wrong results leading high chances of wrong mapping. In fact, dealing with third type of failing patterns, is always difficult and prone to wrong results, specially with high X's. Thats why, we have chosen the three failed pattern first. The proposed greedy algorithm can efficiently find the failing scan cells in those cases. Also, the information gained by this phase, can be very useful while mapping the difficult patterns.

V. Experimental Results

Experiments have been carried out in different industrial design. Main focus of the experiments is to find out a direct way to compare with the scan based design and then analyze the differences. The description of the industrial designs which have is used in the experiment is given in the first four column of Table I. Second, third, and fourth columns indicate the size of the designs in terms of gate count, scan cell count, and chain count in thousands (K). Targeted compression of the SAS compressor over scan designs is set to 50X for all the circuits. Test patterns are generated using commercial ATPG. We have not reported the pattern count as it has been limited to 1000 for each circuit to reduce the experiment time. For each design, 100 different fault injections for stuck at and as well as transition faults have been carried out. Once the failure file is generated for SAS based designs, the mapping algorithm is invoked. The limit of the mapped pattern (described in Section

TABLE I
COMPARISON WITH SCAN DIAGNOSIS RESULTS

Designs	#Gate	#FFs	#Chain	Stuck at fault						Transition fault					
				Diagnosability		Resolution		CPU Time (Sec)		Diagnosability		Resolution		CPU Time (Sec)	
				scan	SAS	scan	SAS	scan	SAS	scan	SAS	scan	SAS	scan	SAS
D1	18.8 K	1.2 K	200	1.00	1.00	1.00	1.00	0.20	0.90	0.98	0.97	1.04	1.05	0.27	0.38
D2	737.2 K	56.3 K	500	0.98	0.97	1.64	1.64	4.48	15.62	0.95	0.95	2.49	2.51	5.74	19.99
D3	463.4 K	43.4 K	500	1.00	0.99	1.07	1.13	8.05	14.10	0.96	0.96	2.06	2.48	11.63	14.57
D4	446.1 K	22.1 K	600	0.98	0.95	1.20	1.20	2.60	3.93	0.80	0.78	1.90	1.93	11.63	14.57
D5	578.5 K	54.5 K	400	1.00	0.99	2.46	2.49	3.84	9.77	0.91	0.91	1.50	1.70	8.92	19.76
D6	1005.9 K	106.1 K	1000	1.00	1.00	2.46	2.49	14.56	54.11	0.92	0.88	1.44	1.47	12.17	28.73
D7	187.7 K	25.4 K	480	1.00	1.00	1.28	1.46	2.06	6.50	0.94	0.94	1.34	1.49	3.28	7.95
D8	293.3 K	25.4 K	500	1.00	0.99	1.69	2.05	4.76	7.08	0.93	0.91	1.43	1.46	6.43	8.60
D9	697.3 K	77.1 K	400	1.00	1.00	1.07	1.13	14.13	18.58	0.90	0.90	1.42	1.43	25.72	37.50
D10	460.5 K	50.2 K	500	1.00	1.00	1.88	2.00	6.88	7.92	0.92	0.81	1.52	1.36	8.40	13.66
Avg.				**1.00**	**0.99**	**1.58**	**1.66**	**6.16**	**13.85**	**0.92**	**0.90**	**1.61**	**1.69**	**9.42**	**16.57**

IV), is set to the half of the total failed pattern for every circuit. For example, let design D3 has 200 failed patterns for an injected fault. When the mapping procedure can map 100 of them, it terminates. Limiting the failing patterns does not have much effect of the diagnosis result.

Once mapping is completed, normal diagnosis of scan design is used on for the mapped patterns. The diagnosis algorithm used is based on inject and evaluate strategy. It starts with a backtracing from the failing cells and find out its input cone. If mapping procedure maps a wrong cell, backtracing will be wrong, and diagnosis result will suffer immensely. Thus, it is extremely important to not map a wrong cell.

Diagnosis algorithm is measured against two metric -

- *Diagnosability*: It is a measure reflecting the percentage of defects which can be correctly identified.
- *Resolution*: The total number of defect candidates reported by a tool is defined as the diagnostic resolution.

Ideal value of both diagnosability and resolution should be 1.

Experimental results found on different circuit are reported also in Table I. The values indicated on the Table I are found out taking an average over 100 experiments. We have found out diagnosability and resolution for scan as well as SAS designs. Average diagnosability of scan for stuck at fault is very close to 1 (rounded to two decimal places) which indicates that almost every time the injected fault has been found. For SAS, value of diagnosability for stuck at fault is 0.99 which is very close to ideal value. Similar observation can be found in case for transition faults also. Resolution or number of reported faults for both scan and SAS are very close to the ideal value. This indicates that there is almost no loss in diagnosis result even with high test data compression. The cpu time taken by both the cases is also reported. Increase in cpu time is justified because of extra effort required for mapping.

For design D10, diagnosability for transition fault in SAS losses more with respect to scan. This design is X heavy and expected values contains many X's. Due to this reason, mapping procedure is not as efficient as it is for other designs for transition fault.

VI. CONCLUSION

In this paper, we have presented a novel mapping algorithm which can map failures from compressor output to the scan cells. The proposed method can identify different kind of failing patterns and decide the required mapping procedure. The algorithm mapped patterns over different passes targeting the patterns which are easier to map. Then, it uses the information gathered from initial passes to map the difficult patterns. Experimental results on different industrial designs have shown that even with presence of high test data compressor, diagnosability and resolution remains almost unaffected. This proves the efficiency and effectiveness of the proposed mapping techniques. Currently, we are working on designs with high number of X's to make the mapping better for them.

REFERENCES

[1] J. Chan and J. Abraham, "A study of faulty signatures using a matrix formulation," in *Proceedings of InternationalTest Conference*, Sep 1990, pp. 553–561.

[2] C. Stroud and T. Damarla, "Improving the efficiency of error identification via signature analysis," in *Proceedings of IEEE VLSI Test Symposium*, Apr 1995, pp. 244–249.

[3] S. Mitra and K. Kee, "X-compact: an efficient response compaction technique," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 3, pp. 421–432, March 2004.

[4] J. Rajski, J. Tyszer, C. Wang, and S. Reddy, "Convolutional compaction of test responses," in *Proceedings of International Test Conference*, Sept 2003, pp. 745–754.

[5] J. Rajski, J. Tyszer, C. Wang, and S.Reddy, "Finite memory test response compactors for embedded test applications," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 4, pp. 622–634, April 2005.

[6] G. Mrugalski, A. Pogiel, J. Rajski, J. Tyszer, and C. Wang, "Fault diagnosis in designs with convolutional compactors," in *Proceedings of International Test Conference*, Oct 2004, pp. 498–507.

[7] G. Mrugalski, A. Pogiel, J. Rajski, and J. Tyszer, "Diagnosis with convolutional compactors in presence of unknown states," in *Proceedings of International Test Conference*, Nov 2005, pp. 394–404.

[8] G. Mrugalski, A. Pogiel, J. Rajski, and J.Tyszer, "Fault Diagnosis With Convolutional Compactors," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 8, pp. 1478–1494, Aug 2007.

[9] A. Chandra, R. Kapur, and Y. Kanzawa, "Scalable Adaptive Scan (SAS)," in *Proceedings of Design, Automation Test in Europe Conference*, April 2009, pp. 1476–1481.

[10] A. Dutta and N. Touba, "Using Limited Dependence Sequential Expansion for Decompressing Test Vectors," in *Proceedings of International Test Conference*, Oct 2006, pp. 1–9.