Scan Chain Organization for Embedded Diagnosis

Melanie Elm, Hans-Joachim Wunderlich
Institut für Technische Informatik
Universität Stuttgart
Pfaffenwaldring 47, D-70569 Stuttgart, Germany
{elm, wu}@iti.uni-stuttgart.de

Abstract

Keeping diagnostic resolution as high as possible while maximizing the compaction ratio is subject to research since the advent of embedded test. In this paper, we present a novel scan design methodology to maximize diagnostic resolution when compaction is employed. The essential idea is to consider the diagnostic resolution during the clustering of scan elements to scan chains. Our methodology does not depend on a fault model and is helpful with any type of compactor.

A linear time heuristic is presented to solve the scan chain clustering problem. We evaluate our approach for industrial and academic benchmark circuits. It turns out to be superior to both random and to layout driven scan chain clustering. The methodology is applicable to any gate-level design and fits smoothly into an industrial design flow.

Keywords — Design for diagnosis, embedded test, scan design

1. Introduction

The central challenges in test and diagnosis of nano-scaled multi-million gate cores are twofold. On the one hand, an increasing amount of test data has to be applied to the core under test (CUT) and an increasing amount of test responses has to be evaluated. This task requires implementing embedded test techniques which may reach from test data compression schemes [22, 18, 8, 4] to complete built-in self-test (BIST) methods [29, 9].

On the other hand, design for manufacturing and yield ramp require detailed internal information for design validation, process monitoring and quality assurance. Thus, a strong request for any test data compression technique is to provide sufficiently detailed diagnostic information.

The basic structure used for embedded testing is shown in figure 1 [23]. A decompressor circuit transforms external data into wide test vectors, and a compactor transforms the test response vectors into narrow external data. If both decompressor and compactor work in an autonomous mode, the structure turns into the STUMPS scheme for BIST [1].

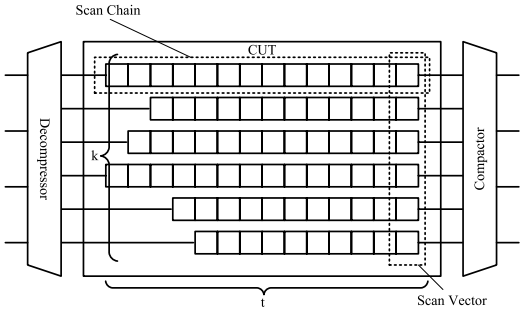

Figure 1. Embedded test scheme with k scan chains of maximum size t.

Up to now, the goal of compressing test data while keeping the diagnostic information has been achieved by using sophisticated test data compaction techniques, by modifying automatic test pattern generation (ATPG) algorithms, or by a combination of both.

Mainly three types of compactors can be found, which keep the diagnostic resolution: Those based on error correcting codes like X-compact [19] and its variations, convolutional compactors as in [24], or those using signature registers with scan chain masking and repetition as in [15]. These techniques have in common that they are efficient, if the number of flipflops which carry erroneous information is limited per scan-vector. For instance, the Saluja-Karpovsky code [26, 21] may identify up to two failing scan chains per vector. Thus the number of failing flipflops per

978-3-9810801-3-1/DATE08 © 2008 EDAA

scan vector has a great influence on the fault aliasing and fault cancellation probability.

For signature registers and convolutional compactors used in BIST, sophisticated techniques have been developed to reconstruct those flipflops from the signature which carry erroneous information. In [2], a dynamic partitioning scheme of the scan cells and test repetition are presented. The location of faulty flipflops in the scan vector determines the number of repetitions and thereby influences the test time.

For convolutional compactors, [20] proposed a backtracking algorithm to identify failing flipflops. In this case the amount of failing flipflops in a scan vector has impact on the efficiency of the search in the backtracking tree.

For stuck-at faults, ATPG can be controlled to make the failing outputs diagnosable [27] and in many cases, stuck-at faults can be identified without modifying ATPG [28]. Unfortunately, stuck-at faults are not the main subject of diagnosis for nano scaled circuits. In contrast, innovative diagnostic techniques analyze the circuit responses without the assumption of any fault model in order to identify the suspect regions of a circuit [12, 7, 11]. For each failing vector to be analyzed, they need to know exactly the set of failing flipflops, and ATPG based on a specific model cannot help.

In this paper, for the first time a method and an algorithm are presented to configure scan chains for improving diagnosability. The approach is not restricted to stuck-at faults like ATPG-based methods and is compatible with the state-of-the-art test compactors. In addition, the algorithm just generates configuration files for standard scan synthesis tools, does not require reordering of scan flipflops and fits into commercial tool chains without further modifications.

After the introduction we present the target scan chain structure for improving diagnosability. In the third section, scan synthesis is mapped to a graph partitioning problem. As the partitioning problem turns out to be NP-complete, an efficient heuristic is presented in section 4. The experimental results discussed in section 5 are obtained by evaluating large industrial designs. They show that the partitioning algorithm improves diagnosability significantly compared to both random and layout based scan insertion.

2. Diagnosable scan chain structures.

Each single line fault in a combinational network can only affect primary outputs which are part of its output cone, i.e. there is a topological path from the fault site to the primary outputs. The output cone of any single line fault is completely contained in any output cone of a primary input which is a predecessor of the fault line (figure 2).

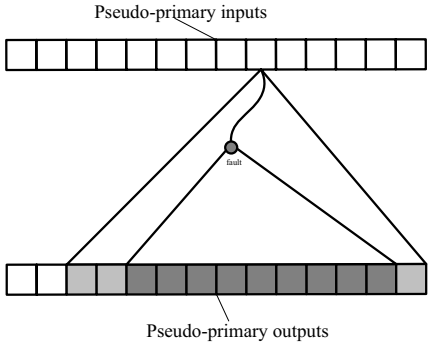

Figure 2. Output and super cone of a fault.

Scan design maps flipflops to pseudo-primary outputs and inputs, and if the flipflops of an output cone of a pseudo-primary input are included in a single scan chain, each response vector will contain at most one erroneous flipflop.

If a vector contains at most one error bit, any response compaction technique based on single error correcting codes like X-compact can identify the error location [19]. Unfortunately, the circuit structure may prevent that all output cones are mapped to single scan chains. Hence, we want to limit the number c of scan chains which cover a single output cone. If the output compressor is a simple parity checker, we have to select $c = 1$ even to avoid fault cancellation as shown in figure 3. The fault information is canceled out in the first shift cycle and it is preserved in the second.

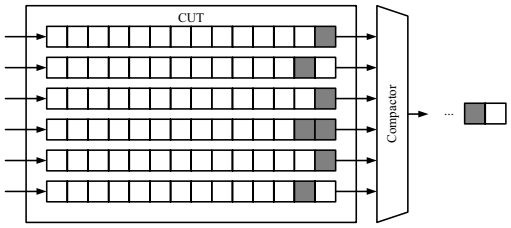

Figure 3. Fault cancellation in a parity tree: The gray scan elements contain the fault information.

If an output cone is distributed between $c = 2$ scan chains, all the errors are located by compactors based on the Saluja-Karpovsky code [21]. In general a BCH code will work for $c > 1$ [16, 25]. However, with increasing c the compactor size will increase exponentially, and it will be necessary to limit c.

Our optimization goal is to limit the number of scan chains covering an output cone by c for as many output cones as possible. Figure 4 shows a random scan organization vs. a diagnosable scan organization for $c = 1$.

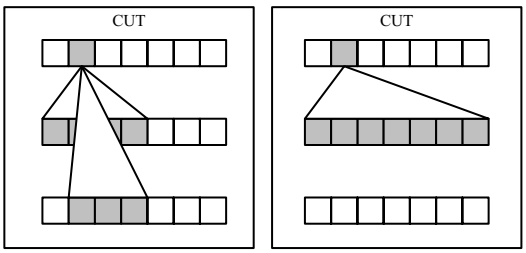

Figure 4. Ordinary scan chain organization (left) and scan chain organization for diagnosis (right).

The scan organization described above does not only support diagnosing single line faults. Even more complex cells and circuit areas are covered, as the outputs of these cells are completely contained in the output cones of the cell inputs. If lines of two different output cones are interacting, they still remain diagnosable in most cases. Standard bridging faults and crosstalks usually change the behavior of one line but not both per pattern, and are visible only in one cone at a time. However, byzantine faults which may be caused by some resistive bridges for instance [17] could affect two cones at the same time. This requires more diagnostic effort during analysis, which is beyond the scope of this paper.

3. Scan chain organization and graph partitioning

In order to obtain independence of any fault model, not the output cones of faulty lines, but the output cones of primary inputs and pseudo-primary inputs are considered. Our goal is to distribute most of the output cones over a number of scan chains not exceeding c. We achieve this by modeling output cones of primary inputs (PIs) and pseudo primary inputs (PPIs) in a hypergraph. Let $H = (V, E)$ with $V = \{v_1, v_2, v_3, \ldots, v_n\}$ be the set of scan elements in a circuit c. Let $E \subseteq \mathcal{P}(V)$ — a subset of the power set of V — be the set of hyperedges. Each hyperedge represents the scan elements of one output cone.

Additionally, let k be the maximum number of scan chains, and t be the maximum number of scan elements in one scan chain (figure 1). Let c be the maximum amount of faulty inputs the applied compactor can correct respectively detect. As already discussed, $c \in \{1, 2, 3\}$ are realistic choices.

By partitioning the hypergraph into k disjoint sets of vertices not bigger than t, where the amount of hyperedges spanning more than c partitions is minimized, we solve the aliasing problem for the chosen compactor optimally. In the following sections, we will use $\Pi(H) = \{p_1, \ldots, p_k\}$ to denote the partitioning.

Unfortunately, the problem of partitioning a graph into k balanced sets of vertices while minimizing the edge cut is NP-complete [3]. Let n be the number of vertices in a graph $G = (V, E)$. By setting $c = 1$ and $t = \lceil n/k \rceil$ we can map the balanced k-way graph partitioning problem to our k-way hypergraph partitioning problem in $\mathcal{O}(1)$. Thus the latter is NP-complete as well.

We do not try to solve an NP-complete problem exactly for multi-million gate circuits. Instead we present an efficient heuristic procedure in the next section.

There exists a variety of heuristics to solve the graph partitioning problem and related problems. Spectral methods find the eigenvalues and eigenvectors of the graph's Laplacian matrix and partition the vertices according to some heuristic derived from the eigenvectors [5]. Other heuristics calculate an initial partitioning and improve it by swapping vertices to other partitions [10, 14].

The multilevel hypergraph partitioning introduced by Kumar and Karypis first coarsens the hypergraph according to several topologically motivated heuristics [13]. After an initial partitioning they use a modified version of the Fiduccia Mattheyses algorithm [10] to return to the initial graph.

These popular heuristics to solve the hypergraph partitioning problem, generalize the problem to an extent, which does not fit to our instance. Instead of finding the global minimum edge cut, we are satisfied with minimizing the amount of edges cut more than c times. It makes no difference, if an edge cuts $c + 1$ or even $c + x$ partitions. In addition, these general heuristics do not target such large problem instances as introduced by nano scaled circuits.

A heuristic and implementation is necessary which works out fast for several hundred thousands of vertices and hyperedges, or even more as circuits will be growing. We will now introduce a linear time heuristic for partitioning a hypergraph into k partitions each of size not bigger than t, where the optimization goal is to minimize the number of hyperedges spanning more than c partitions.

4. Heuristic hypergraph partitioning

The linear time partitioning procedure implements a divide-&-conquer approach. It consists of three main steps.

4.1. Connected components

Each of the connected components of the hypergraph can be dealt with independently. The independent solutions for the components can then be combined to form the global solution.

Let $H^1 = (V^1, E^1), \ldots, H^m = (V^m, E^m)$ be the connected components of the hypergraph H, and let $\Pi(H^i) = (p_1^i, \ldots, p_k^i)$ be the corresponding partitions as determined below. Now the set $P := \{p_j^i | i = 1, \ldots, m; j = 1, \ldots, k\}$ has to be partitioned into k sets $P_1, \ldots, P_k \subset P$ in such a way that each $P_h, h = 1, \ldots, k$ represents at most t vertices, i.e. $\sum_{p \in P_h} |p| \leq t$.

If t is sufficiently large, this is nearly always possible. In rare cases, the partitioning for each component H^i has to be solved for some $t' < t$. For $t' = t/m$, a solution is guaranteed. For the rest of this section we assume now, that the hypergraph is connected.

4.2. Initial states

The main algorithm takes one vertex in each step and decides to which partition this vertex should be put. The initial state of the algorithm puts just one vertex into each of the partitions. In the course of the algorithm, a state is a partitioning of a subgraph, and the initial state is the partitioning of a subgraph with k vertices.

States are evaluated by a cost function, which on the one hand should display the quality of the recent partitioning and on the other hand the difficulty to calculate the next states of the algorithm. This is achieved by assigning a label $L(e)$ to each hyperedge $e \in E$ and a label $L(v)$ to each vertex $v \in V$. The label of an edge is the amount of partitions this edge is spanning at the current state. If an edge is spanning more than c partitions it is marked as inactive and is no longer considered when calculating the cost function. This reduces the computational complexity on the one hand and gives chance to distribute those "broken" edges over as many scan chains as possible on the other hand.

The label of a vertex is the maximum label of all the edges incident to this vertex. Additionally for each vertex we need $A(v)$, which is the amount of edges incident to v with this maximum label.

Let h be the number of inactive hyperedges in the current state of the algorithm. The following cost function puts penalty h^2 to broken edges and evaluates with $L(v)$ how many partitions already were used.

$$C(\Pi) \quad = \quad h^2 + \sum_{v \in V} L(v) \cdot A(v) \qquad (1)$$

Initially the cost function is used to evaluate the two states described below and to select the best of them:

- The k most distant vertices with maximum incidence, and
- the k hyperedges with highest incidence which don't share any incident hyperedges.

The run time of the prepartition step is in any of the two cases linear. Each edge respectively vertex is touched at most once, and only the incident vertices respectively edges have to be marked. Thus if n is the number of edges respectively vertices and q is the maximum incidence of an edge respectively vertex, then the prepartitioning can be done in $\mathcal{O}(n \cdot q)$.

4.3. Main Algorithm

The main algorithm adds one vertex after the other to the state obtained so far. For each vertex it is evaluated by the cost function, to which of the k partitions it should be added.

After the best partition has been found, the vertex is assigned to this partition and the labels and cost function are updated.

Not only the choice of the initial partitioning is important, but also the order in which the vertices are processed. We always choose the one with the highest label. As the label can not grow bigger than c, we can find the next critical vertex fast by sorting the vertices to buckets according to their label. The sorting can be done during the update of the label.

Up to this point we have linear run time. Again each vertex is touched once and, only those labels of edges and vertices have to updated which are incident to the current vertex. Again the complexity is $\mathcal{O}(n \cdot q)$.

5. Experimental results

The quality of a scan chain configuration is evaluated by using the concept of c-diagnosability. An output cone is called c-diagnosable ($c \in \mathbb{N}$), if it is distributed to not more than c scan chains. In order to be completely independent of any fault model we consider the output cones of flipflops.

The approach presented here is only required and designed for large circuits. For this reason, only the large ITC99 benchmark circuits are investigated [6]. In addition, the results for large industrial designs provided by NXP are reported.

As the ITC benchmark circuits do not come with scan chain information, we construct a random configuration and

compare our results. However, for the industrial designs the original scan chain information is available. The original configuration was generated by industrial tools taking into account both layout and RTL information.

Table 1 shows the percentage of cones which are c-diagnosable ($c = 2, 3$) for a random configuration, for the original configuration and for the optimal configuration of the industrial designs. The results for $c = 1$ are exemplarily depicted in figure 5. The results for the biggest ITC 99 benchmarks are reported in table 2.

Circuit	$c = 2$			$c = 3$		
	Random	Original	New	Random	Original	New
p35k	11,35	98,79	99,39	13,13	98,79	99,96
p45k	17,05	29,09	95,85	23,18	65,09	98,29
p77k	33,09	65	82,35	47,88	79,77	97,79
p78k	9,49	99,56	97,44	12,76	96,46	96,46
p81k	3,89	49,65	66,76	8,21	70,57	76,92
p89k	28,06	66,95	76,21	37,77	78,09	94,69
p100k	45,66	84,49	94,98	52,26	92,43	98,87
p141k	12,74	26,98	42,19	17,44	45,13	56,51
p239k	40,58	72,16	84,16	46,54	86,71	95,72
p259k	43,91	72,83	89,33	49,89	86,78	97,99
p267k	23,16	46,72	65,01	34,15	64,95	84,82
p269k	23,47	46,72	64,36	35,08	64,95	85,22
p279k	25,03	49,42	54,7	35,25	56,62	62,57
p286k	29,68	53,21	55,26	39,47	60,16	64,02
p295k	30,57	72,72	78,52	47,83	96	98,98
p330k	22,61	56,35	56,71	29,74	68,31	73,29
p378k	9,37	99,56	98,73	12,51	96,46	96,46
p388k	36,18	83,89	85,75	46,75	88,71	90,18
p418k	23,01	74,49	83,82	37,26	89,82	92,27
p483k	34,46	87,69	94,69	47,63	97,42	99,01
p500k	21,79	80,64	83,37	34,99	91,03	93,61
p533k	43,39	88,56	93,57	57,23	97,57	99,08
p951k	51,53	69,21	87,32	56,13	77,7	92,35
p1522k	15,26	50,41	75,05	75,05	56,25	90,61

Table 1. Results for NXP benchmarks.

All the percentages are calculated with respect to those cones not bigger than $c \cdot t$. Thus, in some cases the c-diagnosability for $c = 2$ is lower than that for $c = 1$.

Circuit	$c = 1$		$c = 2$		$c = 3$	
	Random	New	Random	New	Random	New
b12	15,56	72,22	60,18	93,81	61,86	81,36
b13	4,44	60	31,11	95,56	64,44	100
b14	0	61,11	0	65,38	0	20,65
b17	1,1	42,11	18,71	66,67	20,81	49,95
b18	0,75	40,09	9,76	39,69	11,46	82,21
b19	0,35	44,31	11,52	48,3	13,51	92,48
b20	0	51,82	0	14,63	0,26	52,34
b21	0	49,64	0	15,13	0	33,07
b22	0	47,64	0	27,1	0	78,01

Table 2. Results for ITC 99 benchmarks.

Random configurations lead to a rather poor diagnosability, which is lower than that of the original configurations for the industrial circuits. However, the clustering presented so far outperforms the original configurations significantly

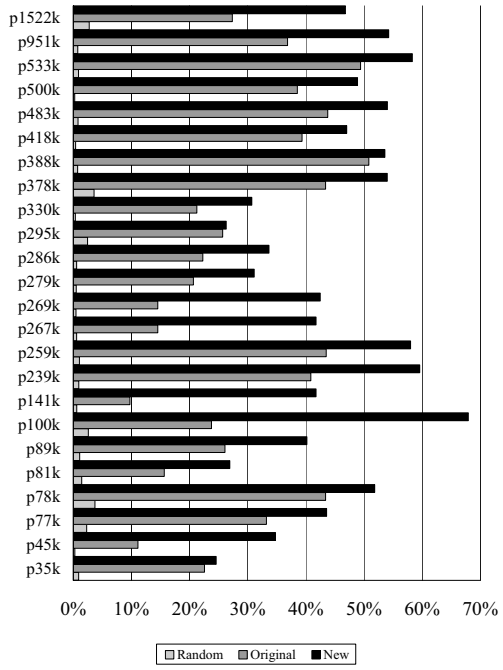

Figure 5. Results of NXP circuits for $c = 1$.

especially in those cases, where the original diagnosability is low.

In those cases where the original diagnosability is high, the new clustering method is able to find a similar solution in linear time and without considering layout information.

To summarize, the proposed algorithm improves the diagnosability of large areas of the circuits. This is obtained just by passing clustering information to the scan synthesis tool by using a constraint file, for instance.

6. Conclusions

For the first time, flipflops are clustered to scan chains in a way that the diagnostic resolution is improved during output compression. The approach fits into the standard tool chain of scan synthesis. For realistic industrial circuits, the partitioning procedure provides significantly better results than the layout and RTL based configurations.

Acknowledgment

The work has been funded by the DFG under contract WU 245/4-1. We would like to thank Alejandro Cook for his support on the implementation and Christian Zöllin for enlightening discussions.

References

[1] P. Bardell and W. McAnney. Self-testing of multichip logic modules. In *Proc. IEEE International Test Conference*, pages 200–204, 1982.

[2] I. Bayraktaroglu and A. Orailoglu. Deterministic partitioning techniques for fault diagnosis in scan-based BIST. In *Proceedings International Test Conference, 3-5 Oct. 2000, Atlantic City, NJ, USA*, pages 273–282, 2000.

[3] T. N. Bui and B. R. Moon. Genetic algorithm and graph partitioning. *IEEE Transactions on Computers*, 45(7):841–855, July 1996.

[4] K. Chakrabarty. Design of optimal linear space compactors for built-in self test. In *Conference Proceedings. IEEE Instrumentation and Measurement Technology Conference, 18-21 May 1998, St. Paul, MN*, volume 1, pages 413–418, 1998.

[5] P. K. Chan, M. Schlag, and J. Zien. Spectral k-way ratio-cut partitioning and clustering. In *30th Conference on Design Automation, 14-18 June 1993*, pages 749–754, 1993.

[6] F. Corno, M. Reorda, and G. Squillero. RT-Level ITC 99 benchmarks and first ATPG Results., 2000.

[7] R. Desineni, O. Poku, and R. D. Blanton. A logic diagnosis methodology for improved localization and extraction of accurate defect behavior. In *IEEE International Test Conference, Oct. 2006*, pages 1–10, 2006.

[8] R. Dorsch and H.-J. Wunderlich. Reusing scan chains for test pattern decompression. In *IEEE European Test Workshop, May 29 - Jun. 1, 2001*, pages 124–132, 2001.

[9] E. Eichelberger and E. Lindbloom. Random-pattern coverage enhancement and diagnosis for LSSD logic self-test. *IBM Journal of Research and Development*, 27(3), 1983.

[10] C. Fiduccia and R. Mattheyses. A linear-time heuristic for improving network partitions. In *19th Conference on Design Automation, 14-16 June 1982*, pages 175–181, 1982.

[11] S. Holst and H.-J. Wunderlich. Adaptive debug and diagnosis without fault dictionaries. In *12th European Test Symposium (ETS 2007), 20 May 2007, Freiburg, Germany*, pages 7–12. IEEE Computer Society, 2007.

[12] L. M. Huisman. Diagnosing arbitrary defects in logic designs using single location at a time (SLAT). *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 23(1):91–101, January 2004.

[13] G. Karypis and V. Kumar. Multilevel k-way hypergraph partitioning. In *Proceedings 36th Design Automation Conference, 21-25 June 1999, New Orleans, LA, USA*, pages 343–348, 1999.

[14] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal, February 1970*, 49(2):291–307, 1970.

[15] A. Leininger, M. Goessel, and P. Muhmenthaler. Diagnosis of scan-chains by use of a configurable signature register and error-correcting codes. In *Proceedings Design, Automation and Test in Europe Conference and Exhibition, 16-20 Feb. 2004*, volume 2, pages 1302–1307, 2004.

[16] S. Lin and J. Daniel J. Costello. *Error Control Coding*. Pearson Education, Inc., 2004.

[17] P. Maxwell and R. Aitken. Biased voting: A method for simulating cmos bridging faults in the presence of variable gate logic thresholds. In *Proceedings.International Test Conference, 17-21 Oct. 1993, Baltimore, MD, USA*, pages 63–72, 1993.

[18] E. McCluskey, D. Burek, B. Koenemann, S. Mitra, J. Patel, J. Rajski, and J. Waicukauski. Test data compression. *Design & Test of Computers, IEEE*, 20(2):76–87, 2003.

[19] S. Mitra and K. S. Kim. X-compact: an efficient response compaction technique for test cost reduction. In *Proceedings on International Test Conference, 7-10 Oct. 2002*, pages 311–320, 2002.

[20] G. Mrugalski, A. Pogiel, J. Rajski, J. Tyszer, and C. Wang. Fault diagnosis with convolutional compactors. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(8):1478–1494, 2007.

[21] J. Patel, S. Lumetta, and S. Reddy. Application of Saluja-Karpovsky compactors to test responses with many unknowns. In *Proceedings on 21st VLSI Test Symposium, 27 April-1 May 2003*, pages 107–112, 2003.

[22] J. Rajski and J. Tyszer. Test data compression and compaction for embedded test of nanometer technology designs. In *Proceedings on 21st International Conference on Computer Design, 13-15 Oct. 2003*, pages 331–336, 2003.

[23] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherjee. Embedded deterministic test. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 23(5):776–792, 2004.

[24] 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*, 24(4):62–634, 2005.

[25] T. Reungpeerakul, X. Qian, and S. Mourad. BCH-based compactors of test responses with controllable masks. In *15th Asian Test Symposium, 2006. ATS '06, Nov. 2006, Fukuoka*, pages 395–401, 2006.

[26] K. K. Saluja and M. Karpovsky. Testing computer hard-ware through data compression in space and time. In *Proceedings IEEE International Test Conference (ITC), 1983*, page 8389, 1983.

[27] H. Tang, C. Wang, J. Rajski, S. Reddy, J. Tyszer, and I. Pomeranz. On efficient X-handling using a selective compaction scheme to achieve high test response compaction ratios. In *18th International Conference on VLSI Design*, pages 59–64, 2005.

[28] H. Vranken, S. Kumar Goel, A. Glowatz, J. Schloeffel, and F. Hapke. Fault detection and diagnosis with parity trees for space compaction of test responses. In *43rd ACM/IEEE Design Automation Conference, 24-28 July 2006*, pages 1095–1098, 2006.

[29] H.-J. Wunderlich. BIST for systems-on-a-chip. *Integration, the VLSI Journal*, 26(1-2):55–78, 1998.