A Rapid and Scalable Diagnosis Scheme for BIST Environments with a Large Number of Scan Chains

Jayabrata Ghosh-Dastidar and Nur A. Touba

Computer Engineering Research Center
Department of Electrical and Computer Engineering
University of Texas, Austin TX 78712-1084
E-mail: {dghosh, touba}@cat.ece.utexas.edu

Abstract

This paper presents a rapid and scalable built-in self-test (BIST) diagnosis scheme for handling BIST environments with a large number of scan chains. The problem of identifying which scan cells captured errors during the BIST session is formulated here as a search problem. A scheme for adding a small amount of additional hardware that provides the capability of performing very efficient search techniques to locate the error-capturing scan cells is proposed. The scheme can accurately diagnose any number of error-capturing scan cells. The error-capturing scan cells can be located in time complexity that is logarithmic in the total number of scan cells in the design using the proposed approach. The technique scales well for very large designs. The hardware overhead is logarithmic in the number of scan cells and linear in the number of scan chains.

1. Introduction

The dwindling cost of integrating transistors on a chip coupled with the rising complexity and cost of testing the chips, as well as ever growing time to market pressures, have helped speed the acceptance of design-for-test (DFT) in the design flow. As feature sizes continue to scale down, the time and cost for debug/diagnosis is becoming increasingly important as well and is leading towards the acceptance of design-for-debug/diagnosis (DFD) techniques. Such techniques are very important for reducing the time to market. Debug/diagnosis in a built-in self-test (BIST) environment is especially difficult and is the problem addressed in this paper. In order for BIST to be an effective test solution, techniques for reducing the time for debug/diagnosis are needed. A new DFD technique that permits rapid diagnosis in a BIST environment that contains a large number of scan chains is presented here.

The need for debug/diagnosis capability usually occurs during three phases in the lifetime of a chip. After the design is fabricated on silicon for the first time, popularly known as the "first silicon," a substantial amount of effort is put on debugging the first silicon. Such debug efforts usually weed out problems in the design, including design errors and design marginality. Due to the growing intricacy of the manufacturing process, device complexity, and random variations in the manufacturing process, all fabricated chips do not meet targeted specifications. The integrated circuit manufacturers need debug/diagnosis capability to identify any persistent anomaly in the manufacturing process to achieve better yield and reliability of their products. Field diagnosis is the last phase where diagnosis capability is required. Such field diagnosis can provide valuable information about the reliability of the device and might point out possible weaknesses in the design and/or production process for that device. The DFD hardware that we propose to put in a design is useful in all the three phases of diagnosis.

This paper studies the problem of diagnosis in a scan-based BIST environment. BIST has been gaining popularity as a mechanism for testing a wide variety of integrated circuits. But to ensure its overall success, a BIST environment must be able to provide similar diagnostic capability as a conventional scan-based external testing environment. A scan-based BIST environment for large designs generally consists of a test pattern generator (TPG) feeding a large number of scan chains. The output response of the circuit-under-test (CUT) is shifted out of the scan chains and into a multiple-input signature register (MISR). The final signature at the end of the BIST session is compared against a fault-free golden signature to ascertain pass/fail for the CUT. This final signature is so highly compacted that very little information can be extracted from it for diagnostic purposes unless the number of errors is very small. In general, there is no bound on the multiplicity of errors during BIST since a single defect can cause a large number of vectors to produce faulty responses and a large number of scan chains can capture those faulty responses. Diagnosis in a BIST environment adds an extra level of difficulty in comparison to diagnosis in a non-BIST environment. This is because it is first necessary to find out from the collected information which scan-elements

0-7695-0613-5/00 $10.00 © 2000 I

have captured faulty responses. Since present day high performance designs have shallow combinational logic depths between successive sequential stages, identifying which scan elements captured faulty responses is a powerful piece of information. Once this information is obtained, the diagnostic techniques for conventional non-BIST designs can be used to further narrow down the search space since such techniques would now need to only concentrate on cones of logic feeding the scan elements capturing faulty responses.

A scan-based BIST environment differs from a standard scan-based design in that the test vectors are generated on-chip using a LFSR and the output response from the scan chains is not directly shifted out to the tester, but is compacted using a MISR. After the BIST session is complete, only the final signature contained in the MISR is scanned out. Information about the individual output responses for each scan vector is lost. The straightforward solution to the loss of diagnostic capability in a scan-based BIST environment is to provide access from all the individual scan chains to the outside world and shift the output responses out and store them on a tester during BIST. This may not be possible for a number of reasons:

1. In a BIST environment, millions of vectors can be applied in a short amount of time. There generally will not be enough tester memory to store the output response for all the BIST vectors.
2. The scan clock frequency that is used during BIST may be too fast for the tester to accurately sample the output at.
3. There may be issues with availability of I/O ports and the additional routing that has to be performed to connect the scan output ports to the I/O pins.
4. There may be problems for systems-on-chip designs where a core with BIST in it might be embedded deep inside the design.

In this paper we propose a new DFD technique that does not require access to the individual scan chains and does not require scanning out the output responses during BIST. It uses very efficient search techniques to locate the scan cells that capture a faulty response across all of the scan chains in the design. It achieves accurate diagnostic resolution of any number of error-capturing scan cells. It is scalable for very large designs as the time complexity for diagnosis is logarithmic in the number of scan cells, and the hardware overhead is logarithmic in the number of scan cells and linear in the number of scan chains.

2. Previous Work

Early work in BIST diagnosis focused on trying to locate single errors from the signature of a single LFSR [McAnney 87], MISR [Chan 90], or multiple signature registers [Stroud 95], [Karpovsky 93]. Multiple errors were targeted in [Darmarla 95], [Karpovsky 93], [Aitken 89], but at the cost of large overhead.

More recently, two low overhead schemes [Wu 96], [Rajski 97], have been proposed for identifying the scan elements that capture errors during a BIST session. Both techniques rely on collecting multiple signatures for diagnosis and can accurately locate up to a pre-specified number of error-capturing flip-flops. These techniques do not require shifting out the contents of the scan chains. They only require scanning out the final signature. Wu and Adham [Wu 96] proposed a technique based on a Reed-Solomon code. The DFD hardware that is needed is a programmable MISR to collect multiple signatures. The MISR is programmed with different polynomials, and the BIST session is repeated for each polynomial to produce a signature. A set of non-linear equations is then solved to identify the set of scan cells that had faulty responses. This method requires scan chain selection hardware to select one scan chain at a time for diagnosis. Rajski and Tyszer [Rajski 97] proposed a technique that uses an LFSR to pseudo-randomly mask out different sets of scan cell responses when collecting multiple signatures. The BIST session is repeated and each time a different set of scan cell responses is pseudo-randomly masked out. By analyzing the signatures, the scan cells that had faulty responses are identified by the process of elimination. The DFD hardware that this technique uses is scan cell selection hardware and also a scan chain selection mechanism.

Research has also been done on trying to determine which BIST vectors produced errors [Savir 88], [Stroud 95], [Aitken 89], [Karpovsky 93], [Damarla 95], [Ghosh-Dastidar 99]. Identification of the failing test vectors is a much harder problem and requires either more hardware overhead or more signatures. This paper focuses only on the problem of locating the error-capturing scan cells, however, it can be used in conjunction with techniques that diagnose the failing vectors.

In comparison with existing schemes for locating the error-capturing scan cells, the BIST diagnosis scheme proposed here provides some nice advantages. While the schemes in [Wu 96] and [Rajski 97] are only able to accurately detect up to a pre-specified number of error-capturing scan cells, the proposed scheme can detect any number of error-capturing scan cells. For the proposed scheme, the number of signatures required for diagnosis is logarithmic in the total number of scan cells in the design, so it scales very well for large designs. The hardware overhead is comparable to [Rajski 97] and scales logarithmically in the number of scan cells per scan chain and linearly with the number of scan chains in the design.

3. Proposed Scalable Diagnosis Scheme

The general scan-based environment that we are targeting is shown in Fig. 1. It is the well-known STUMPS architecture [Bardell 87]. The test pattern generator (TPG) feeds multiple scan chains in parallel. The output responses captured in the scan chains are then shifted out and compressed in the MISR. In many cases, an exclusive-OR network may be present before the MISR to reduce the number of states in the MISR. Our proposed scheme is not affected by the presence of such a network, so we will not consider that in the following discussion.

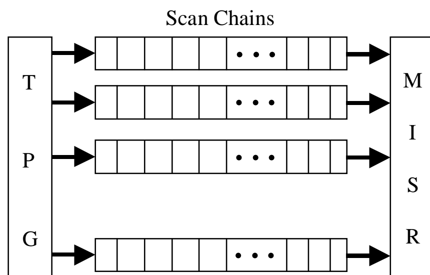

Figure 1. Scan-Based BIST Environment

3.1 Problem Abstraction

From an abstract point of view, diagnosing which scan cells captured a faulty response in a BIST session is basically a search problem. Search is a well-studied problem. Given a sorted sequence of elements, we can search any element from that sequence in *O[log n]* time complexity using a well-known optimal search strategy called binary search. Let us investigate how we can cast our problem of diagnosis into a search problem. We can view the multiple scan chains as a matrix (illustrated in Fig. 2) where each row represents a scan chain and each entry in the matrix represents a scan cell. Note that our diagnosis scheme does not depend on the assumption that all the scan chains are perfectly balanced, but we will assume that fact here to simplify the analysis.

Binary search uses the basic technique of partitioning to locate an element in a given sequence. In each step of a binary search, the algorithm partitions the remaining search space into equal halves and proceeds with the search in one of the halves. One way to partition the matrix is to select a string of columns and a set of rows. One such partition is marked in Fig. 2. Such a partition can be defined by the triplet *(**X**, Y, Z)* where:

X is the set of scan chains (rows) contained in the partition. For Fig. 2, ***X*** is the set {2,3}.

Y is the distance of the rightmost column of the string of scan cells from the MISR. For Fig. 2, Y = 4.

Z is the distance of the leftmost column of the string of scan cells from the MISR. For Fig. 2, Z = 6.

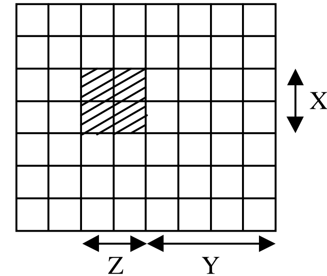

Figure 2. Matrix Representation of Scan Chains

The proposed diagnosis scheme relies on partitioning the scan cells in this way. By so doing, a binary search can be performed as well as other types of searches. This is accomplished using simple hardware and without loss of accuracy in identifying the error-capturing scan cells.

3.2 Hardware for Partitioning the Scan Cells

The general block diagram of the hardware for partitioning the scan cells is shown in Fig. 3. The subset of scan chains that are allowed to feed the MISR in each BIST session is controlled by register *X*. Scan chain *i* is allowed to take part in forming the signature in a BIST session if the bit position X_i in the *X* register is 1. So by setting appropriate bit positions in the *X* register to 0's and 1's, we can partition in the scan chains into any desired groups.

In addition to the scan chain selection hardware, there is also hardware to select the string of scan cells in the selected scan chains that are fed into the MISR. This is done using the registers *Y* and *Z*. Register *Y* is compared with the scan counter using comparator *C1* which performs a "greater than" comparison. The contents of register Z are compared against the same scan counter using comparitor *C2* which performs a "less than" comparison. The output of the scan chains is fed in the MISR only when both of the comparisons are true. In other words, only the scan cells that lie between distance *Y* and *Z* from the MISR are used in forming the signature.

Using this hardware, we can compute the signature for various partitions of the set of all scan cells. If the signature for a particular partition is error-free, then we know that none of the scan cells in that partition captured errors (assuming there was no aliasing in the MISR). If the signature is faulty, then we know that at least one of the scan cells in the partition did capture one or more errors.

As the number of scan cells per scan chain increase, the size of the *Y* and *Z* registers grow logarithmically. As the number of scan chains increase, the size of the *X* register grows linearly. Thus, the DFD hardware for this scheme scales well for very large designs

A workstation can be used to perform the diagnosis. For each BIST session, the workstation loads the values of

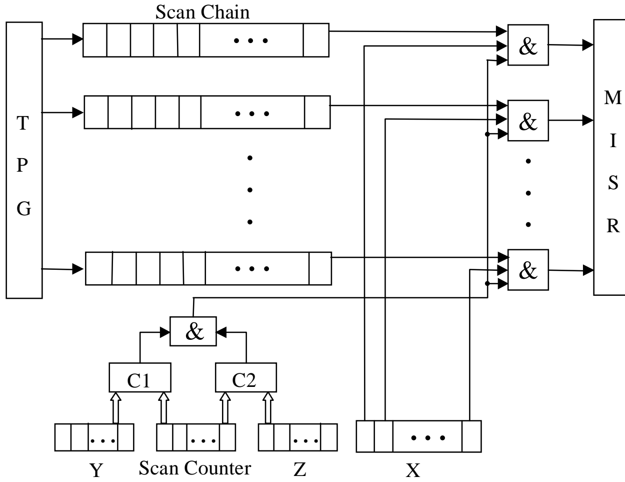

Figure 3. Block Diagram of the Diagnosis Scheme

the X, Y, and Z registers and starts the BIST session. When the BIST session is complete, the workstation reads the final signature from the MISR. The workstation compares that signature with the fault-free signature and loads the appropriate values of the X, Y, and Z register for the next BIST session based on the result. The fault-free signature for each individual scan cell can be pre-computed once, and then the fault-free signature for each BIST session can be rapidly computed by superposition. The bit-wise sum of each of the fault-free signatures of the individual scan cells that are shifted into the MISR during the BIST session forms the overall fault-free signature for the BIST session by the principle of superposition [Bardell 87].

3.3 Finding Scan Cells that Capture Errors

Given the DFD hardware described in the previous subsection, we can perform diagnosis very rapidly. Here we describe how to do diagnosis following the general strategy of binary search. Note, however, that this DFD hardware is general enough to perform other search strategies based on partitioning (other strategies will be described in Sec. 5). The diagnosis scheme based on binary search for detecting which scan chains contain error-capturing scan cells can be described as follows:

1. Initialize all bits in X_{error} to 1.
2. Partition X_{error} into two equal halves $X_{partition_1}$ and $X_{partition_2}$.
   2.1 Set $\mathbf{X} = X_{partition_1}$.
3. Run the BIST session.
4. Compare signature in MISR with fault-free signature.
   4.1 If signature is error-free, set $X_{error} = X_{partition_2}$. Go to step 2.
   4.2 If signature is erroneous, stop if the desired accuracy has been obtained or else set $X_{error} = X_{partition_1}$ and go to step 2.

Initially all scan cells can be a possible source of error. So all the bits in X_{error} are set to 1. The X_{error} set is partitioned into $X_{partition_1}$ and $X_{partition2}$. First $X_{partition_1}$ is tested to determine if it contains a scan cell in error. If $X_{partition_1}$ is in error, then it is partitioned further to zoom in on the scan cells in error. If not, then we know that $X_{partition_2}$ must be in error, so further partitions are performed on $X_{partition_2}$. This scheme will take *O(log n)* BIST sessions, where n is the number of scan chains, to identify a scan chain containing error-capturing scan cells. Not only can we use this scheme to identify which scan chains captured a faulty response, but we can also use the same strategy to partition the scan cells in a scan chain using the Y and Z registers to accurately identify the error-capturing scan cells within the scan chain. The procedure for that is as follows:

1. Set the bit in the X register corresponding to the scan chain that is to be diagnosed
2. Initialize $Y_{error} = 0$, $Z_{error} = m$ (where m is the number of scan cells in the scan chain).
3. $Z_{partition} = (Y_{error} + Z_{error})/2$.
4. $\mathbf{Y} = Y_{error}$, $\mathbf{Z} = Z_{partition}$.
5. Run the BIST session.
6. Compare signature in MISR with fault-free signature.
   6.1 If signature is error-free set $Y_{error} = Z_{partition}$, and go to step 3.
   6.2 If signature is faulty, stop if the desired accuracy has been obtained or else set $Z_{error} = Z_{partition}$, and go to step 3.

In the above procedure, the **Y** and **Z** registers select the partition of the scan chain that is shifted into to the MISR in a particular BIST session. If the partition is error-free then we know that the other scan chain partition marked by $Z_{partition}$ and Z_{error} contains scan cells in error. So further partitions are done for the interval marked by $Z_{partition}$ and Z_{error}. This scheme will take *O(log m)* BIST session. In each successive BIST session, we are continuously dividing the matrix shown in Fig. 2 into two equal halves.

Consider the case where we are trying to locate an error-capturing scan cell where the number of scan chains is 100 and the number of scan cells on an average in each scan chain is 120. Then our method will converge in $\lceil \log_2(100) \rceil + \lceil \log_2(120) \rceil = 14$ BIST sessions. Comparing this with schemes that diagnose one scan chain at a time, there is a significant advantage. Such schemes would first run BIST sessions with only one scan chain observable at a time. That alone will take 100 BIST sessions in the worst case and 50 BIST sessions on average. Then additional BIST sessions would be required to locate the error-capturing scan cell in the selected scan chain.

The method described in [Rajski 97] is probabilistic in nature and hence might not be able to identify the exact error-capturing scan cells in every case. The method described in [Wu 96] is accurate as long as the number of error-capturing scan cells is less than the error correction capability t of the *bEC* that is used. The proposed method can accurately identify any number of error-capturing scan cells. If there are multiple error-capturing scan cells, then in some steps both partitions might produce faulty signatures. In that case, the number of BIST sessions would be different from $O(\log n)$. The number of BIST sessions for diagnosing multiple error-capturing scan cells depends on the number of error-capturing scan cells and their distribution in the scan chain. In the next section, we will derive an expression for the number of BIST sessions in the worst case scenario.

4. Analysis of Diagnosis Scheme

If there is only one error-capturing scan cell, then we know that the binary search procedure will find the error-capturing scan cell with $\log_2(m*n)$ BIST sessions, where m is the average number of scan cells in a scan chain and n is the total number of scan chains.

In the general case where there are multiple error-capturing scan cells, then the number of BIST sessions depends on which partitions each of the error-capturing scan cells are in. The worst case scenario is when the error-capturing scan cells are equidistant from each other in the matrix. To illustrate this, consider the simple case where there is one scan chain containing 64 scan cells out of which 3 capture errors. The 3 error-capturing scan cells are nearly equidistant from each other, which constitutes the worst possible case for our scheme. It is the worst case because whenever we do a partition of the scan cells, the error-capturing scan cells will lay on different partitions. The diagnosis tree for such a worst case scenario is shown in Fig. 4. The leaf nodes in Fig. 4 marked with '*' are the error-capturing scan cells. The number of nodes in the tree minus the top node constitutes the number of BIST sessions required, which in this case is 30. Careful analysis of the diagnosis tree leads us to a closed form representation of the number of BIST sessions required to accurately diagnose t error-capturing scan cells in the worst case:

Let the total number of scan cells be N and the number of error-capturing scan cells t, then

Number of BIST sessions $B = (2t)\lfloor \log_2(N/t) \rfloor + 2t - 1$

The expression B is an upper bound on the number of BIST sessions, because if after testing one of the partitions we find it is error-free, then we know that the other partition must have a error-capturing scan cell. So we can straight away start partitioning the other partition without running a BIST session for it.

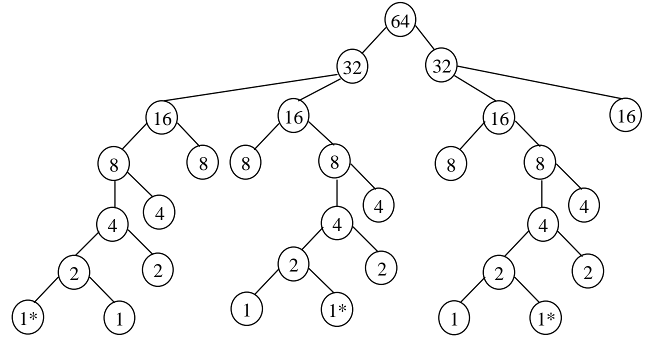

Figure 4. Tree Representation of the Search Strategy

Note that typically there will be a lot of spatial locality for the error-capturing scan cells because scan cells driven by overlapping cones of logic will tend to be near each other in the scan chain routing. Hence, a fault whose effects branch out to multiple scan cells will tend to cause errors in nearby scan cells. Thus, the worst case scenario of having equidistant error-capturing scan cells is very unlikely.

Consider the example depicted by Fig. 4 further. If there was only one error-capturing scan cell among the 64 scan cells, then our diagnosis scheme will terminate in 6 BIST sessions. But for 3 error-capturing scan cells, the number increases to 30. This is because in presence of multiple error-capturing scan cells. In the worst case, we are unable to remove part of our search space in each step. Consider the following alternative to a full binary search. To start with, we divide the 64 scan cells into groups of 8. Let us assume that we were lucky in this case, and 2 error-capturing scan cells ended up in one group, and the other scan cell remained in a separate group. So after the first set of 8 partitions, we have (2*8)=16 unresolved scan cells in 2 groups of 8 scan cells. Within these 2 groups let us assume worst case behavior as before. We partition these 2 groups into 8 groups of 2 scan cells. So in the worst case in this scenario the total number of partitions comes to (8+8+6)=22 a savings of 26.67% over the previous case. This illustrates the fact that for multiple error-capturing scan cells, it is possible that an alternative partitioning strategy to binary search could be better. In the next section, we describe techniques for partitioning the search space using structural information about the CUT.

5. Partitioning Using Structural Information

The proposed DFD hardware described in this paper is capable of implementing other partitioning strategies in addition to binary search. In binary search, the search space is partitioned into two equal parts in each step. It is possible to improve on binary search if we utilize structural information about the CUT. Using structural information, we can better partition the search space to locate the fault with fewer steps on average.

One piece of structural information that can be utilized is which scan cells are part of a single functional block. Groups of scan cells in the BIST architecture may all be part of the same functional block (i.e., module, core, etc.) such that a fault within that functional block may only affect that group of scan cells. If one of the scan cells in a particular functional block captures an error, then it is likely that any other error-capturing scan cells would also be a part of that functional block. By knowing which scan cells belong to which functional blocks, we can better partition the search space so that we can find the error-capturing scan cells more rapidly.

Another more thorough way to extract structural information from the CUT is to trace the cones of logic feeding each scan cell and construct a weighted graph G as follows:

1. Each scan cell in the CUT corresponds to a node in the graph G.
2. There exists a directed edge between two nodes i and j, iff they have an overlapping cone of logic.
3. The edge weight $W_{i,j} = C_{i,j} / C_j$, where $C_{i,j}$ is the size of the overlapping cone of logic between scan cells i and j, and C_j is the size of the cone of logic for scan cell j. $W_{i,j}$ is a heuristic that represents the probability of how likely scan cell j will be error-capturing given that scan cell i was error-capturing. The size of the cone of logic can be estimated by gate count.

To identify the cone of logic C_i for a scan cell i, we backtrace through scan cell i's input. We can stop backtracing once we encounter a primary input or another scan cell. The backtracing process does not need to continue beyond a scan cell because any error affecting a scan cell C_1 cannot propagate and affect any scan cell C_2 lying at the fan out of C_1 during a BIST session. Note that the construction of graph G is independent of any fault model, and so its use in our diagnosis scheme does not make our scheme fault model dependent.

Once we have extracted the structural information from the CUT and constructed the graph G, we can use this information to more rapidly locate the error-capturing scan cells. The proposed search strategy is as follows:

1. Perform binary search to identify one of the error-capturing scan cells, f. Note that scan cell f can be identified in $O(\log m*n)$ iterations, where m is the number of scan chains and n is the average number of scan cells in each scan chain.
2. From the graph G, identify all the scan cells that have an edge with f. Let us call that set G_f.
3. If size of the set G_f is small, then perform linear search, i.e., select one scan cell belonging to the set G_f in each BIST session and continue until all the elements in G_f have been considered.
4. If size of the set G_f is large, then select scan cells from the set in decreasing order of edge weights. If we identify another error-capturing scan cell j then reduce the set of possible error-capturing scan cells G_f by taking set intersection of G_f and G_j. Continue for all members of G_f.

In the above procedure, once an error-capturing scan cell f is identified, we construct a set G_f which depicts the other possible error-capturing scan cells. The members of the set G_f are validated either directly by selecting that scan cell i individually for a BIST session, or indirectly when we identify another error-capturing scan cell j and i does not lie in the set G_j. In the second case, i is dropped from the set of possible error-capturing scan cells G_f as any defect that affects scan cell i cannot affect scan cell j which has been observed to be in error, and so is inconsistent with the present validated information under a single defect assumption. The above procedure is guaranteed to identify all error-capturing scan cells provided there is only one actual defect location in the CUT. The multiple defect assumption can be easily incorporated in this scheme by running additional tests on partitions constructed out of scan cells not diagnosed to be in error. So we exclude all the scan cells that we have diagnosed as error-capturing in the additional tests. If any of the additional tests produce an erroneous signature, then the above procedure is applied for that partition. Note that our entire diagnosis process is independent of any fault model and is robust enough to handle cases where multiple defects are present in the CUT.

6. Results

We did a comparison of our diagnosis scheme with the one proposed in [Rajski 97]. The DFD hardware in [Rajski 97] randomly partitions the search space, whereas the scheme proposed here deterministically partitions the search space to perform a binary search or search strategies based on structural information. We did experiments to compare the techniques in terms of the number of BIST sessions required and the accuracy in diagnosing the error-capturing scan cells.

Table 1 shows results for a scan chain configuration with 500 scan chains where each scan chain has 1000 scan cells in it. Random errors were generated in each case. Each experiment was repeated a number of times with different random errors and the average result is shown. A primitive polynomial of order 16 was used for [Rajski 97] in all the experiments. The first column in Table 1 shows the number of scan chains that contain error-capturing scan cells, column 2 shows the number of error capturing scan cells in each scan chain. Columns 3, 4, 5 shows results for [Rajski 97] assuming pseudo-random partitioning is performed for diagnosing both the

Table 1. Experimental Results for 500 Scan Chains with 1000 Scan Cells per Scan Chain

Faulty Scan Chains	Faulty Scan Cells	[Rajski 97] Random Partition for Both Scan Chains and Scan Cells			[Rajski 97] Linear Search for Scan Chains Random Partition for Scan Cells			Proposed Scheme		
		BIST Sessions	Avg. Undiag. Chains	Avg. Undiag. Cells	BIST Sessions	Avg. Undiag. Chains	Avg. Undiag. Cells	BIST Sessions	Avg. Undiag. Chains	Avg. Undiag. Cells
1	2	80	0	0.5	540	0	0.5	44	0	0
1	3	104	0	0.1	564	0	0.1	59	0	0
1	5	136	0	0.2	596	0	0.2	82	0	0
2	5	256	0	0.2	692	0	0.2	160	0	0
2	10	384	0	0.3	820	0	0.3	262	0	0
2	20	904	0.2	0.1	1364	0	0.1	426	0	0
2	24	940	0.2	0.1	1400	0	0.1	500	0	0
2	32	1336	0.2	0.4	1796	0	0.4	592	0	0
2	36	1480	0.1	0.4	1940	0	0.4	654	0	0
3	5	344	0.1	0.2	788	0	0.2	240	0	0
3	10	536	0.1	0.3	980	0	0.3	393	0	0
3	20	1352	0.1	0.1	1796	0	0.1	639	0	0
3	24	1406	0.1	0.1	1850	0	0.1	650	0	0
3	32	2000	0.1	0.4	2444	0	0.4	888	0	0
3	36	2216	0.1	0.4	2660	0	0.4	981	0	0

scan chains and scan cells. Columns 6, 7, 8 shows results for [Rajski 97] where the scan chains are diagnosed one at a time. Columns 9, 10, 11 shows results for the proposed scheme with a binary search (no structural information was used). Experiments were performed for different numbers of scan chains in error and different numbers of scan cells in error. In all the cases, the proposed technique using binary search had fewer BIST sessions and correctly identified all the failing scan cells. For [Rajski 97], the number of BIST sessions were selected so as to keep the average number of undiagnosed scan chains or scan cells in every case to be small, less than 0.5.

As can be seen from the results, the proposed scheme requires fewer BIST sessions in all cases. Since it is capable of always finding all error-capturing scan cells, there were never any undiagnosed cells. Note that the hardware overhead for the proposed scheme is less than the hardware overhead for [Rajski 97] with random partitioning of both the scan chains and scan cells.

6. Conclusions

The proposed method adds DFD hardware to efficiently locate the error-capturing scan cells. Any number of error-capturing scan cells can be located accurately. Structural information about the CUT can be used to find partitioning strategies that are even more effective than binary search.

One of the nice properties of the STUMPS architecture for BIST is that it scales very well for large designs. The proposed DFD approach scales in the same manner and can provide a solution for rapid diagnosis for BIST.

Acknowledgements

This material is based on work supported in part by the National Science Foundation under Grant No. MIP-9702236, and in part by the Texas Advanced Research Program under Grant No. 003658-0644-1999.

References

[Aitken 89] Aitken, R.C., and V.K. Agarwal, "A Diagnosis Method Using Pseudorandom Vectors Without Intermediate Signatures," *Proc. of Int. Conf. on Computer-Aided Design (ICCAD)*, pp. 574-577, 1989.

[Bardell 87] Bardell, P.H., W.H. McAnney, and J. Savir, "Built-In Test for VLSI: Pseudorandom Techniques," *John Wiley and Sons*, New York, 1987.

[Chan 90] Chan, J.C., and J.A. Abraham, "A Study of Faulty Signatures Using a Matrix Formulation," *Proc. of International Test Conference*, pp. 553-561, 1990.

[Darmala 95] Darmala, T.R., C.E. Stroud, and A. Sathaye, "Multiple Error Detection and Identification via Signature Analysis," *Journal of Electronic Testing (JETTA)*, Vol. 7, No. 3, pp. 193-207, 1995.

[Ghosh-Dastidar 99] Ghosh-Dastidar, J., D. Das, and N.A. Touba, "Fault Diagnosis in Scan-based BIST using Both Time and Space Information", *Proc. of International Test Conf.,* pp. 95-102, 1999.

[Karpovsky 93] Karpovsky, M.G., and S.M. Chaudhry, "Design of Self-Diagnostic Boards by Multiple Signature Analysis," *IEEE Trans. on Computers*, Vol. 42, No. 9, pp. 1035-1043, Sept. 1993.

[McAnney 87] McAnney, W.H., and J. Savir, "There is Information in Faulty Signatures," *Proc. of Int. Test Conf.*, pp. 630-636, 1987.

[Rajski 97] Rajski, J., and J. Tyszer, "Fault Diagnosis in Scan-Based BIST," *Proc. of International Test Conference*, pp. 894-902, 1997.

[Stroud 95] Stroud, C.E., and T. Damarla, "Improving the Efficiency of Error Identification Via Signature Analysis," *Proc. of VLSI Test Symposium*, pp. 244-249, 1995.

[Savir 88] Savir, J., and W.H. McAnney, "Identification of Failing Tests with Cycling Registers," *Proc. of Int. Test Conf.*, pp. 322-328, 1988.

[Wu 96] Wu, Y., and S. Adham, "BIST Fault Diagnosis in Scan-Based VLSI Environment," *Proc. of Int. Test Conf.*, pp. 48-57, 1996