

# Fault Dictionary Based Scan Chain Failure Diagnosis

Ruifeng Guo                      Yu Huang                      Wu-Tung Cheng  
Mentor Graphics Corp.      Wilsonville, OR 97070      USA  
{ruifeng\_guo, yu\_huang, wu-tung\_cheng}@mentor.com

## Abstract

*In this paper, we present a fault dictionary based scan chain failure diagnosis technique. We first describe a technique to create small dictionaries for scan chain faults by storing differential signatures. Based on the differential signatures stored in a fault dictionary, we can quickly identify single stuck-at fault or timing fault in a faulty chain. We further develop a novel technique to diagnose some multiple stuck-at faults in a single scan chain. Comparing with fault simulation based diagnosis technique, the proposed fault dictionary based diagnosis technique is up to 130 times faster with same level of diagnosis accuracy and resolution.*

## 1. Introduction

In high volume manufacturing environment, fast and accurate fault diagnosis is critical to help yield analysis team to quickly root cause failures. Generally, fault diagnosis techniques can be classified into two categories: cause-effect and effect-cause [1]. Given a set of tester failures, effect-cause diagnosis first identifies a list of candidate faults based on failures observed on a tester. Then fault simulation is performed for the candidate faults. The simulated responses are compared with tester failures to identify the defect location. This approach is powerful because it could handle advanced fault models. However, when the number of candidate faults is large, diagnostic fault simulation becomes a time consuming process.

Cause-effect diagnosis is based on pre-calculated fault signatures, which are stored in a fault dictionary. Cause-effect diagnosis uses matching information between tester failures and the stored fault signatures to quickly identify diagnostic candidates. One concern about fault dictionary based diagnosis is that the fault dictionary size is huge for a large design, which is computationally expensive to create and utilize. Dictionary-based diagnosis generally lacks the flexibility to diagnose defects that do not behave as stuck-at faults.

With wide adoption of scan-based design-for-test techniques, scan chain failures become an important factor that impacts yield analysis and product ramp up. In this paper, we propose a fault dictionary based diagnosis technique for scan chain fault diagnosis. We first describe a technique to create small fault dictionary for scan cell faults. Then we describe a diagnosis

technique that provides fast fault diagnosis for single stuck-at fault, timing fault, and some multiple stuck-at faults in a scan chain. The proposed fault dictionary based diagnosis is up to 130 times faster than state of the art fault simulation based diagnosis techniques while providing comparable diagnosis resolution and accuracy to simulation-based diagnosis technique.

The paper is organized as follows. Section 2 reviews previous research of scan chain fault diagnosis. Section 3 provides definitions and fault models used in this paper. In Section 4, we analyze fault signatures of scan cell faults and propose a technique to construct small fault dictionary for scan chain faults. Section 5 describes fault diagnosis techniques for single fault and multiple stuck-at faults. Experimental results are provided in Section 6. Section 7 concludes the paper.

## 2. Background

Scan chain failure diagnosis is the process to identify a defective scan cell in a scan chain. A defect in a scan chain can cause wrong values to be captured into or shifted out a scan cell. Scan chain fault diagnosis techniques have been investigated in the past [5-20]. These techniques can be classified into three categories: tester-based, hardware-based, and software-based diagnosis techniques. Tester-based diagnosis techniques [8, 11, 12] use tester to control and observe defect responses at different test conditions to identify a failing scan cell. These techniques are usually very time-consuming. Hardware-based diagnosis techniques [5, 7, 9, 10] are effective in isolating scan chain defects. However, these techniques require special design of scan chains with extra hardware overhead which may not be acceptable. Software-based techniques use algorithmic diagnosis procedures to identify failing scan cells [6, 13, 14, 15, 16, 17, 18, 19, 20]. Most recent software-based diagnosis techniques belong to fault simulation-based diagnosis.

In this paper, we explore fault dictionary based diagnosis techniques for scan chain fault diagnosis. One major concern about fault dictionary based diagnosis is the size of the fault dictionary. Previous researches on fault dictionary based diagnosis are mainly for faults in system logic and focus on reducing the size of a fault dictionary [2, 3, 4]. A dictionary based diagnosis technique for scan chain fault diagnosis was discussed in [7]. However, the technique proposed in [7] only

works for designs with special scan chain architecture described in [7]. From this aspect, it belongs to hardware-based diagnosis techniques and cannot be applied to general scan designs.

In this paper, we propose a fault dictionary based diagnosis technique for scan chain fault diagnosis. It handles single stuck-at fault, timing fault and some multiple stuck-at faults. This technique doesn't require any modification to the existing scan architecture and can be applied to general scan designs. We address dictionary size issue by minimizing redundancy of fault signatures such that a small fault dictionary can be constructed. Differential signatures for stuck-at and timing faults are stored in the fault dictionary. Based on the differential signatures, we propose a diagnosis technique for single stuck-at and timing fault in a scan chain. A novel technique to identify some multiple stuck-at faults in a faulty scan chain is also developed to diagnose scan chains with multiple stuck-at faults. To the best of our knowledge, this is the first paper that fault dictionary is used in software-based scan chain failure diagnosis.

### 3. Definitions and Fault Models

We use the following definitions in this paper. The **length** of a scan chain is the total number of scan cells in the scan chain. Each scan cell in a scan chain is given an index. The cell connected to scan chain output is numbered 0 and the cells in the scan chain are numbered incrementally from scan output to scan input sequentially. The scan cell with index  $N$  is referred to as scan cell  $N$ . The scan cells between the scan chain input and the scan input pin of a scan cell are called the **upstream cells** of this scan cell, while the scan cells between the scan chain output and the scan output pin of a scan cell are called the **downstream cells** of this scan cell.

We consider faults on scan cell input pin, output pin or inside a scan cell that impact scan shift process. The fault types considered in this paper include stuck-at-0, stuck-at-1, transition faults (including slow-to-rise, slow-to-fall, and slow faults), and hold-time faults (including fast-to-rise, fast-to-fall and fast faults). During scan chain failure diagnosis, chain test patterns are used to identify failing scan chains and fault types [10, 13, 14, 15, 18]. A stuck-at-0 or stuck-at-1 fault causes unload values on chain test patterns to be constant 0s or 1s. A transition fault causes 0 to 1 and/or 1 to 0 transitions to be delayed by one scan shift cycle while a hold time fault causes 0 to 1 and/or 1 to 0 transitions to be faster by one scan shift cycle. Detailed descriptions of these fault models and their impacts to scan chain operations can be found in earlier publications [10, 13, 14, 15, 18]. Given a set of test patterns, the **fault signature** of a scan cell fault is the set of mismatches (or failures) between the faulty circuit responses and the expected good circuit responses.

### 4. Fault Dictionary for Scan Chain Faults

In this section, we analyze the redundancy of fault signatures among scan cell faults and describe a technique to create small fault dictionary for scan chain fault diagnosis. Note that the calculation of fault signature for chain test patterns is straightforward and usually doesn't require fault simulation. In the proposed fault dictionary, we don't store fault signatures for chain test patterns and only fault signatures for scan test patterns are stored.

During scan load shifting, all downstream scan cells of a faulty scan cell are impacted by the defect. During scan unload shifting, all upstream scan cells are impacted. Usually a scan chain fault causes a lot of failures for a scan test pattern. It would be too expensive to build a fault dictionary with full fault signatures of all scan cell faults. In order to reduce the size of scan chain fault dictionary, we have the following observation:

**Observation:** *Fault signatures of two adjacent scan cells are usually slightly different.*

We analyzed four industrial circuits and counted the number of different failures caused by adjacent scan cell faults for 50 scan test patterns. The results are shown in Table 1. From Table 1, we can see that each scan cell fault has thousands of failures in its full fault signature, while the number of different failures of adjacent scan cell faults is less than 10 for the first three circuits and is 28.8 for the fourth one. Comparing with the large number of failures for each scan cell fault, the number of different failures of adjacent scan cell faults is relatively small (less than 0.1% of full failures). Based on this observation, we can see that in order to reduce fault dictionary size, we can ignore the common failures between two adjacent scan cell faults while only keeping the small number of different failures in a fault dictionary. The difference between the fault signatures of two adjacent scan cell faults are called **differential signature**. For each scan chain, we store full fault signature for one scan cell, while for all the other scan cells, we only store their differential signatures.

**Table 1: Number of Different Fails of Adjacent Scan Cells**

	Average # of failures per fault	# of different failures of adjacent cell faults
Circuit 1	16000	5.9
Circuit 2	8600	7.6
Circuit 3	210000	6.8
Circuit 4	92000	28.8

Figure 1 shows an example of how a fault dictionary is constructed. The upper part of Figure 1 shows the full fault signatures for scan cells 0, 1, 2, and 3. Each line in a box denotes a failure for the corresponding scan cell fault. For example, the first failure for scan cell 0 SA0 fault is "1 chain0

5”, which means that SA0 fault at scan cell 0 causes a failure for scan pattern 1, at scan cell 5 of scan chain “chain0”. The lower part of Figure 1 shows how we construct a fault dictionary by storing differential signatures. For scan cell 0, we keep its full signature, i.e. all four failures are kept in the fault dictionary. For scan cell 1, we calculate the difference between its fault signature and the fault signature of scan cell 0. It has one more failure “1 chain1 6” while missing one failure “1 chain1 1”. These two failures are the difference between the fault signatures of SA0 faults at scan cells 1 and 0. They are stored in the dictionary for scan cell 1 SA0 fault. Similarly, for scan cell 2, we compare the full fault signatures of SA0 faults at scan cells 2 and 1. SA0 fault at scan cell 2 misses failure “1 chain1 3”. This failure is stored for scan cell 2 SA0 fault in the fault dictionary. By storing the differential signatures between adjacent scan cell faults, we minimize the redundancy of fault signatures in the fault dictionary.

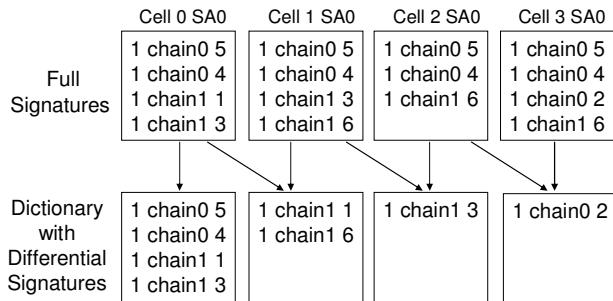


Figure 1: A Fault Dictionary with Differential Signatures

During fault diagnosis, when we need the fault signature of a scan cell fault, for example SA0 fault at scan cell 1 in the above example, we can calculate it by taking the “exclusive-or” operation of the differential signatures of SA0 faults at scan cells 0 and 1. One drawback of this method is that if we need the full fault signature of scan cell N, we need to calculate the full fault signatures of scan cells 1, 2 ... (N-1). In our diagnosis flow, we try to avoid using full fault signature of a scan cell fault, which will alleviate the computation of “exclusive-or” of full fault signatures. Detailed usage of the differential signatures is described in Section 5.

## 5. Fault Dictionary Based Diagnosis

### 5.1 Diagnosis of Single Fault in a Faulty Chain

The fault type of a single stuck-at fault or timing fault in a faulty chain can be identified by applying a chain test pattern and observing the unload values [14, 15, 16, 18, 20]. Once we know the fault type, the exact location of the failing scan cell can be identified by matching the observed failures on a tester with the scan cell fault signatures. For permanent stuck-at or timing fault, we expect complete match between tester failures and full fault signatures of the defective scan cells. For a single fault, diagnosis results of the proposed technique should be the same as those of fault simulation based diagnosis [14, 15, 20].

To compare with tester failures, one way is to restore full fault signatures for all scan cells before comparing with tester failures. In a fault dictionary, for each scan chain we store full signatures for only one scan cell. We call this full fault signature the *base signature* of the scan chain under diagnosis. We use the scan cell in the middle of a scan chain as the base cell. Full fault signatures of other scan cells can be calculated based on the differential signatures in the fault dictionary. However, given thousands of failures in the fault signature of a scan cell fault, it is computationally expensive to compute full fault signatures and compare with tester failures. In order to alleviate this issue, for each scan cell we only compute the difference between its fault signature and the base signature of the faulty scan chain, which is called *relative signature* to the base signature. We also compute the difference between tester failures and the base signature of the faulty chain under diagnosis, which is called *relative tester failures* to the base signature. We then compare the relative signatures with relative tester failures. If a relative signature is the same as the relative tester failures, the corresponding full fault signature of the scan cell fault also matches the full tester failures. For most scan cells, the relative signatures contain much less failures than full fault signatures. This makes it more efficient to calculate the relative signature for each scan cell fault and compare it with the relative tester failures.

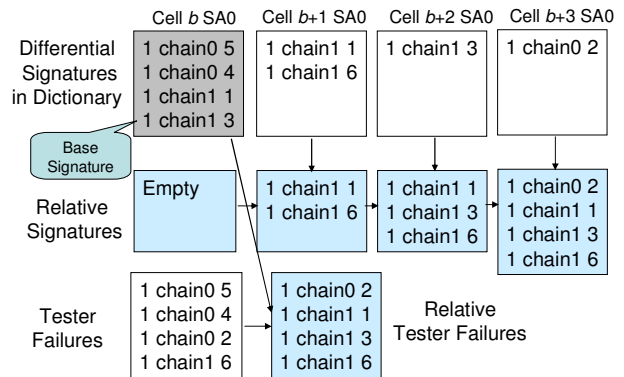


Figure 2: An Example of Relative Signature Calculation

Figure 2 shows an example of calculating relative signatures and relative tester failures. In this example, the full fault signature of scan cell *b* is used as the base signature of the faulty scan chain. The relative tester failures are calculated by taking the difference between the base signature and the tester failures. Because the fault signature of scan cell *b* SA0 fault is the base signature, the relative signature for this fault is empty. For scan cell *b+1*, its relative signature is the same as its differential signature which reflects the difference between full fault signature of scan cell *b+1* SA0 fault and the base signature. For scan cell *b+2* SA0 fault, its relative signature is calculated based on the relative signature of scan cell *b+1* SA0 fault and the differential signature for scan cell *b+2* SA0 fault. The difference between these two sets of failures consists of the relative signature for SA0 fault at scan cell *b+2*. Similarly, we can

calculate relative signature of SA0 fault at scan cell  $b+3$  by taking the difference between its differential signature and the relative signature of SA0 at scan cell  $b+2$ . In this example, since the relative signature of scan cell  $b+3$  SA0 fault perfectly matches the relative tester failures, this fault will be reported as a suspect.

If we find a scan cell whose failing signature completely matches tester failures, we continue check the scan cells next to it until we reach one whose failing signature doesn't match tester failures. This will make sure that we identify all the consecutive cells that are potential diagnosis candidates. If we couldn't find a scan cell whose failing signature completely matches tester failures, we know that it is not a single permanent stuck-at or timing fault. Multiple stuck-at faults or intermittent fault are then considered for diagnosis.

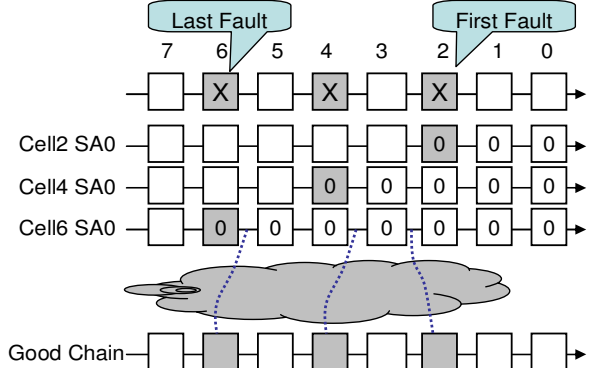
**5.2 Diagnosis of Multiple Stuck-At Faults in a Faulty Chain**

We propose a method to diagnosis multiple stuck-at-0 faults or multiple stuck-at-1 faults in a single scan chain. In this paper, we only target multiple stuck-at faults that cause the same stuck-at behavior at the scan chain output, i.e. either they all show as SA0 fault at the chain output or they all show as SA1 fault at the chain output for chain test patterns. Diagnosis for arbitrary stuck-at faults in a scan chain will be studied in our future work.

During scan load process, the last fault (the fault closest to scan chain input) will mask the fault effect of all other faults in the scan chain. During scan unload process, the first fault (the fault closest to scan chain output) will mask the fault effect of all other scan cell faults. For combinational test patterns where only one capture cycle is applied between scan load and scan unload, the impact of the faults that are between the first and the last faults are masked by the first and the last faults. In order to identify the faults between the first and the last faults, sequential test patterns with more than one capture cycles or scan patterns with special shift and unload procedures [16, 17] are required to observe their fault effects. In this paper, we focus on the identification of the first and the last faults in a scan chain.

For multiple stuck-at faults in a scan chain, we first try to identify the last fault. During scan load process, the last fault masks fault impact of all other scan cell faults in the same scan chain. We use the example shown in Figure 3 to illustrate. In this example, there are 3 SA0 faults in a scan chain on scan cells 2, 4 and 6 respectively. The last fault is at scan cell 6. During scan load process, the SA0 fault at scan cell 6 causes scan cells 6 through 0 to get value 0. Scan cell 4 SA0 fault causes cells 4 through 0 to get value 0 and the SA0 fault at scan cell 2 causes cells 2, 1 and 0 to get value 0. Note that the impact of the SA0 fault at scan cell 6 masks the impact of the SA0 faults at scan cells 4 and 2 during scan load process. In other words, the impact to scan load values of the last fault is the same as that of the multiple faults. Furthermore, the failures

observed on good scan chains or primary outputs (POs) are only impacted by scan load values of the faulty scan chain. Based on this analysis, we can say that failures observed on good chains and POs caused by the last fault should be the same as those caused by the multiple scan cell faults. These failures can be used to identify the location of the last fault.



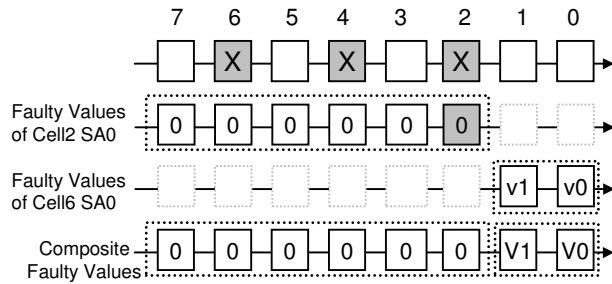
**Figure 3: Fault Impact on Scan Load Values and Diagnosis of Last Fault Using Good Chain Failures**

While diagnosing the last fault, we partition tester failures to two groups. One group contains failures on the faulty scan chain and the other group contains failures on good scan chains and POs. Similarly, for the differential signatures stored in a fault dictionary, we partition them to fault signatures with only faulty chain failures and fault signatures with only good chain and PO failures. To identify the last fault location, we only need to compare the fault signatures with tester failures for failures on good scan chains and POs. A match between these two sets of failures indicates a candidate scan cell of the last fault.

Once the last fault is identified, we move on to identify the first fault. This time, we focus on using failures on the faulty scan chain. Note that during scan chain unload process, for scan cells between the first fault location and the scan chain input, the first fault masks the fault effects of other faults in the faulty scan chain. For the scan cells between the first fault location and the scan chain output, scan unload values are not impacted by any scan cell faults during scan unload process. Unload values of these scan cells should reflect the scan capture values which are determined by the scan load values. As we already know that the last fault in a scan chain dominates the impact to scan load values. Its fault signature reflects the captured values of the cells that are in the downstream of the first fault.

In order to identify the first faulty cell  $N$ , we create a *composite signature* with two pieces of information for faulty chain failures: (1) failures on scan cell  $N$  and its upstream scan cells and (2) failures on its downstream scan cells. The first piece of information is extracted from the fault signature of the fault at scan cell  $N$  while the second piece of information is extracted from the fault signature of the last fault identified in the previous step. Given a candidate of the last fault, we search for

a scan cell  $N$  such that the composite signature from the last fault and the fault at scan cell  $N$  matches faulty chain failures observed on a tester. Note that the above analysis works for combinational test patterns and sequential test patterns where the faulty scan cells don't impact the capture values of fault-free scan cells in the faulty scan chain. Figure 4 shows an example on how to create a composite signature of faulty scan chain failures. In this example, assume from previous step we know that scan cell 6 is a candidate of the last fault and we want to know whether scan cell 2 is a candidate for the first fault. The composite signature consists of two pieces of information: (1) failures on cells 7 through 2 and (2) failures on cells 1 and 0. For failures on scan cells 7 through 2, we use the fault signature for SA0 fault at scan cell 2. For failures on scan cells 1 and 0, we use the fault signature for SA0 fault at scan cell 6. If the composite signature matches faulty chain failures observed on silicon, scan cell 2 is a candidate of the first fault. In case that there are multiple candidates for the last fault, we use the candidate cell with the lowest index during the first fault identification process. This heuristic helps avoid excessive searching effort while still achieving good diagnosis accuracy.



**Figure 4: First Fault Diagnosis Using Faulty Chain Failure**

In case that the above technique failed to identify a single scan cell fault signature or a composite signature that exactly matches tester failures, we assume that there is an intermittent fault in the scan chain. For intermittent fault diagnosis, we use similar matching and scoring techniques as described in [14]. Scan cells in faulty chain are ranked based on matching scores between the fault signatures and tester failures.

### 6. Experimental Results

Experiments were performed on several industrial circuits to evaluate the effectiveness of the proposed technique. Design features of each circuit are given in Table 2. For each circuit, we calculated the differential signatures of all scan cell faults for 50 scan test patterns and stored them in fault dictionaries based on the technique described in Section 4. Depending on the design sizes and chain lengths, CPU times to create fault dictionaries are quite different. For the three circuits used in this work, the CPU times to create fault dictionaries for all the stuck-at and timing faults took from several hours to several days.

Once the fault dictionaries are created, our first experiment is to evaluate the size of the fault dictionary and its impact to the

diagnosis memory usage. Table 3 shows the number of failures for full fault signatures and the number of differential failures in fault dictionaries. We also show the reduction ratio between these two numbers. For all the three circuits, comparing with full fault signatures, the proposed technique reduces the number of failures to be stored in the fault dictionaries by 535, 300 and 2408 times respectively. It is interesting to point out that the reduction ratios are roughly proportional to the scan chain lengths of the circuits. Table 4 shows the impact of fault dictionaries on memory usage of the diagnosis tool. From Table 4, we can see that with fault dictionaries, the memory overhead varies from 2.8 MB to 60MB (or from 1.8% to 13.3%).

**Table 2: Designs Used in the Experiments**

Circuit	#Gate	#Chain	#ChainLength
A	70K	8	725
B	701K	27	430
C	506K	16	2570

**Table 3: Comparing Fault Dictionary and Full Signatures**

	#Failures for Full Signatures	#Failures in Fault Dictionaries	Reduction Ratio
A	528M	0.98M	535
B	522M	1.74M	300
C	16.2G	16.78M	2408

**Table 4: Diagnosis Tool Memory Usage**

Circuit	Without fault dictionary	With fault dictionary	Memory Overhead	Overhead Percentage
A	102.3MB	105.1MB	2.8MB	2.7%
B	550.3MB	560.3MB	10.0MB	1.8%
C	451.1MB	511.3MB	60.2MB	13.3%

Our next experiment is to evaluate the effectiveness of the diagnosis flow. We perform single fault diagnosis and multiple fault diagnosis separately. For each category, 100 faults (or fault pairs for multiple fault diagnosis) are randomly injected for each circuit. We perform fault simulation to create simulation mismatches between expected responses and fault simulation responses. The simulation mismatches are used as tester failures in our experiments. We also compared the proposed diagnosis technique with previously published diagnosis techniques [14, 15, 20]. In our experiment, if a diagnosis result contains the target scan cell and if it has less than five scan cells as candidates, we call it a good diagnosis which can provide useful information for physical failure analysis. The diagnosis results are shown in Table 5. From this table, we can see that fault simulation based diagnosis [14, 15] (under columns "Sim") and the proposed dictionary based fault diagnosis (under columns "Dic") achieve the same diagnosis results for all single fault diagnosis cases. This is reflected in the percentage of good diagnosis in the second and third columns in Table 5. The last three columns show the CPU times used by fault simulation based diagnosis [14, 15], the proposed fault dictionary based diagnosis and their ratios. It can be seen that fault dictionary

based diagnosis takes much less CPU time than fault simulation based diagnosis. For example, for circuit C, it takes fault simulation based diagnosis more than 4 hours to diagnose 100 test cases, while it takes only 4 minutes for fault dictionary based diagnosis. The average speedup over fault simulation based techniques varies from 8 to 97 for the three circuits. Note that the fault simulation based diagnosis used in our experiment uses similar speedup techniques as presented in [20]. The speedup for circuit A is small due to it is a small circuit and non-simulation activities (e.g. reading tester failures, loading fault dictionaries) take a larger portion of total diagnosis run time.

**Table 5: Diagnosis Results for Single Faults (100 cases)**

Circuit	%Good Diag		Total CPU Time (seconds)		
	Sim.	Dic.	Sim.	Dic.	Ratio
A	80%	80%	250	31	<b>8</b>
B	52%	52%	3556	36	<b>97</b>
C	52%	52%	15138	246	<b>57</b>

**Sim:** Fault simulation based diagnosis **Dic:** Dictionary based diagnosis

In Table 6, we compare fault dictionary based diagnosis and fault simulation based diagnosis for multiple stuck-at fault diagnosis. The percentage of good diagnosis is calculated based on good diagnosis of the first fault and the last fault, each fault is given 50% credit for a test case. From Table 6, we can see that fault dictionary based diagnosis achieves similar or a little better diagnosis results (for Circuit B) than fault simulation based diagnosis [14, 15, 20], but the CPU times used by fault dictionary based diagnosis are much shorter. For example, for circuit B, it takes fault simulation based diagnosis 1.8 hours to diagnose 100 test cases, while it takes less than one minute for fault dictionary based diagnosis. The average speedup over fault simulation based diagnosis varies from 14 to 130. Note that the percentage of good diagnosis cases is much lower than single fault diagnosis. One reason is due to the nature of the complicated behavior of multiple faults. In our experiments, we have seen that multiple stuck-at faults behave just as a single stuck-at fault while this single fault candidate doesn't match any injected faults.

**Table 6: Diagnosis Results for Multiple Stuck-At Faults**

Circuit	%Good Diagnosis		Total CPU Time (seconds)		
	Sim.	Dic.	Sim.	Dic.	Ratio
A	35%	35%	1008	68	<b>14</b>
B	34%	41%	6565	50	<b>130</b>
C	25%	24%	11909	546	<b>22</b>

**Sim:** Fault simulation based diagnosis **Dic:** Dictionary based diagnosis

## 7. Conclusions

We described a dictionary based fault diagnosis technique for scan chain failure diagnosis which can be used for yield analysis in high volume manufacturing diagnosis flow. Differential signatures are stored in fault dictionaries to minimize the

redundancy of fault signatures of adjacent scan cell faults. Based on the differential signatures stored in a fault dictionary, we proposed a diagnosis technique that diagnoses single stuck-at fault, timing fault and some multiple stuck-at faults in a single scan chain. Experimental results on industrial designs show that comparing with fault simulation based diagnosis techniques the proposed diagnosis technique significantly improves the diagnosis speed up to 130 times while still achieving similar or slightly better diagnostic resolution and accuracy. Our future work will focus on diagnosis of more sophisticated defect behaviors caused by multiple faults in a single scan chain or multiple scan chains.

## References

- [1] M. Abramovici, M. Breuer, A. Friedman, "Digital Systems Testing and Testable Design", IEEE Press, 1995
- [2] I. Pomeranz, S. M. Reddy, "On Dictionary-Based Fault Location in digital Logic Circuits", IEEE TCAD, Jan. 1997, pp. 48-59
- [3] B. Chess and T. Larrabee, "Creating small fault dictionaries", IEEE TCAD, March 1999, pp. 346-356
- [4] P. Bernardi, M. Grosso, et al., "A pattern ordering algorithm for reducing the size of fault dictionaries", VTS 2006, pp.386-391
- [5] J. Schafer, et al. "Partner SRLs for Improved Shift Register Diagnostics", Proc. VLSI Test Symposium, 1992, pp. 198-201
- [6] S. Kundu, "Diagnosing Scan Chain Faults," IEEE Trans. On VLSI Systems, Vol. 2, No. 4, December 1994, pp. 512-516
- [7] G. Edirisooriya, S. Edirisooriya, "Scan Chain Fault Diagnosis with Fault Dictionaries", ISCAS 1995, pp.1912-1915
- [8] K. De and A. Gunda, "Failure Analysis for Full-Scan Circuits", ITC, 1995, pp. 636-645
- [9] S. Narayanan, A. Das, "An Efficient Scheme to Diagnose Scan Chains," Proc. Int'l Test Conference, 1997, pp. 704-713
- [10] Y. Wu, "Diagnosis of Scan Chain Failures," Proc. Int'l Symp. on Defect and Fault Tolerance, 1998, pp. 217-222
- [11] P. Song, et. al., "Diagnostic techniques for the IBM S/390 600MHz G5 Microprocessor", Proc. ITC, 1999, pp. 1073-1082
- [12] J. Hirase, N. Shindou and K. Akahori, "Scan Chain Diagnosis using IDDQ Current Measurement", Proc. Asian Test Symposium, 1999, pp. 153-157
- [13] K. Stanley, "High Accuracy Flush and Scan Software Diagnostic", Proc. IEEE Workshop on Yield Optimization & Test, 2000
- [14] R. Guo, S. Venkataraman, "A Technique For Fault Diagnosis of Defects in Scan Chains", ITC, 2001, pp. 268-277
- [15] Y. Huang, W.-T. Cheng, S.M. Reddy, C.-J. Hsieh, Y.-T. Hung, "Statistical Diagnosis for Intermittent Scan Chain Hold-Time Fault", ITC, 2003, pp.319-328.
- [16] J. S. Yang, S. Huang, "Quick Scan Chain Diagnosis Using Signal Profiling", ICCD, 2004
- [17] A. Crouch, "Debugging and Diagnosing Scan Chains," EDFAS, Vol. 7, Feb., 2005, pp 16-24
- [18] J. Li, "Diagnosis of Single Stuck-at Faults and Multiple Timing Faults in Scan Chains", IEEE Trans. On VLSI systems, Vol. 13, No. 6, June 2005, pp. 708-718
- [19] J. Li, "Diagnosis of Multiple Hold-Time and Setup-Time Faults in Scan Chains" IEEE Trans. Computer, Vol. 54, No. 11, Nov. 2005 1467-1472
- [20] Y. Kao, W. Chuang and J. Li, "Jump Simulation: A Technique for Fast and Precise Scan Chain Fault Diagnosis", ITC 2006, no. 22.1