

# On Scan Chain Diagnosis for Intermittent Faults

Dan Adolfsson

NXP Semiconductors  
Corp. Innovation & Technology  
High Tech Campus 32  
5656AE Eindhoven  
The Netherlands  
dan.adolfsson@nxp.com

Joanna Siew

Philips Applied Technologies  
Digital Systems & Technologies  
High Tech Campus 5  
5656AE Eindhoven  
The Netherlands  
joanna.siew@philips.com

Erik Jan Marinissen

IMEC vzw  
Digital Components  
Kapeldreef 75  
B-3001 Leuven  
Belgium  
erik.jan.marinissen@imec.be

Erik Larsson

Linköpings Universitet  
Dept. of Computer Science  
Embedded Systems Lab  
SE-581 83 Linköping  
Sweden  
erila@ida.liu.se

## Abstract

Diagnosis is increasingly important, not only for individual analysis of failing ICs, but also for high-volume test response analysis which enables yield and test improvement. Scan chain defects constitute a significant fraction of the overall digital defect universe, and hence it is well justified that scan chain diagnosis has received increasing research attention in recent years. In this paper, we address the problem of scan chain diagnosis for intermittent faults. We show that the conventional scan chain test pattern is likely to miss an intermittent fault, or inaccurately diagnose it. We propose an improved scan chain test pattern which we show to be effective. Subsequently, we demonstrate that the conventional bound calculation algorithm is likely to produce wrong results in the case of an intermittent fault. We propose a new lowerbound calculation method which does generate correct and tight bounds, even for an intermittence probability as low as 10%.

## 1 Introduction

Diagnosis is the process of pinpointing the failure type and location on the basis of failing test responses. Diagnosis is increasingly important. Traditionally, diagnosis has the role of pre-processing step before physical failure analysis (PFA). PFA tools provide the ultimate defect confirmation in the form of a microscopic defect image. A diagnosis software tool gives the necessary direction guidance to the peephole view of these PFA tools. The detailed PFA investigations provide valuable information, but are, due to their time-consuming and labor-intensive nature, restricted to individual cases, such as customer returns. Recently, diagnosis also has started to play a role in yield and test improvement actions [1, 2]. Many of the ICs that fail the volume-production test are now subjected to diagnosis-only runs, that report on a statistically significant basis which types of failures and which failure locations are most prominent and need fixing. This high-volume diagnosis is only possible, because the diagnosis step is fully automated, and the time-consuming, labor-intensive PFA step is omitted.

Scan design is the most commonly practiced form of design-for-testability. The prime function of scan chains is to provide the on-chip test access infrastructure for the combinational logic. However, scan chains themselves can also be subject to defects, and hence also require testing and corresponding diagnosis methods. The test for the scan chain(s) covers both the scan chain connections, as well as the functional flip-flops that constitute the scan chain(s); in total, this might correspond to up to  $\sim 30\%$  of the standard-cell area [3] and hence account for a similarly-sized fraction of all standard-cell defects.

In this paper, we address scan chain diagnosis under the assumption of a single, but possibly intermittent fault. With ‘intermittent’, we mean that the fault can at random moments exhibit its fault effect, or not. The intermittent fault has an *intermittence probability*  $p$ , with  $0 < p < 1$ .

For  $p = 0$ , the scan chains are fault free, and for  $p = 1$ , the scan chain fault is *permanent*. Note that  $p$  is unknown at the start of the diagnosis process. Previous papers have published both theoretical insight [4] in and industrial evidence [5] for the actual occurrence of intermittent scan chain faults. In this paper, we restrict ourselves to six commonly-assumed functional fault types for scan chains, but now with intermittent behavior. The random intermittent behavior of the scan chain fault provides a difficult situation for diagnosis, as the fault effect might be absent at crucial moments. Consequently, the conventional scan chain diagnosis procedure will often result in wrong diagnosis conclusions. We propose a new scan chain test and a new bound calculation, which do provide accurate diagnosis results, even for low intermittence probabilities.

The remainder of this paper is organized as follows. Section 2 describes both the scan chain test and the logic test. Related prior work is reviewed in Section 3. Section 4 describes why the conventional scan chain test pattern is likely to miss intermittent faults, proposes an improved test pattern, analyzes its effectiveness, and shows that its associated costs are negligible. Section 5 demonstrates that, in the presence of intermittent faults, the conventional bound calculation is likely to give wrong results, and proposes a new lowerbound calculation method which does not suffer from this problem. Section 6 concludes this paper.

## 2 Scan Chain Test vs. Logic Test

We distinguish between (1) the scan chain test and (2) the logic test. The scan chain test (sometimes also referred to as ‘scan chain continuity test’ or ‘flush test’) focuses on the detection of faults in the scan chain (including the scan flip-flops themselves), while the logic test fo-

cuses on the detection of faults in the combinational logic in between the scan flip-flops. The differences between logic test and scan chain test are summarized in Table 1.

Property	Scan Chain Test	Logic Test
Tests	scan chains	combinational logic
Fault models	stuck-at, transition	stuck-at, delay, . . .
Test generation	fixed pattern	ATPG
# Patterns	one	hundreds
Execution protocol	scan-through	scan-in; apply; scan-out

**Table 1:** Comparison between scan chain test and logic test.

The scan chain test typically consists of one short pattern, which detects all fault types considered, and is independent from the circuit or scan chain design. Commonly used scan chain test patterns are 010, 00110, and 11001100<sup>1</sup> [6–8]. Optionally, this test pattern is preceded by an initialization sequence of the length of the scan chain; mandatorily, this test pattern is succeeded by a ‘push-through’ sequence of the length of the scan chain.

In this paper, we follow the majority of the publications in scan chain diagnosis and assume six functional scan fault models: *Stuck-at-Zero/One* (SA0, SA1) and *Fast/Slow-to-Fall/Rise* (FTF, FTR, STF, STR). Table 2 shows for the scan chain test pattern 11001100 what happens in the fault free case, and in the cases of these six fault types. In the second column the sensitive bits are indicated by means of a bold font. *Sensitive* bits are bits of the test pattern which are sensitive to the fault type at hand, i.e., bits of which the value will get inverted if the fault behaves according to that fault type. The third column of Table 2 shows the test responses in the fault-free case and for each of the six fault types. The failing bits in the test responses are indicated by means of underlining. As example, we take a closer look at what happens in the case of a Fast-to-Fall (FTF) fault. The pattern has only one falling (1-to-0) transition, viz. from Bit 4 to Bit 5<sup>2</sup>. The Fast-to-Fall fault models that the transition comes earlier than expected. Consequently, Bit 4 is the (bold) sensitive bit, and if the FTF fault indeed occurs, this Bit 4 is inverted from its correct value 1 to an (underlined) faulty value 0.

As can be seen in Table 2, the six fault types can be classified as either *0-Generator* or *1-Generator*. The 0-Generators (SA0, FTF, and STR) only generate faulty bits with logic value 0, while the 1-Generators (SA1, FTR, and STF) only generate faulty bits with logic value 1. Note that the faulty bits of 0-Generators FTF and STR are true subsets of the faulty bits of a Stuck-at-Zero fault; likewise, the faulty bits

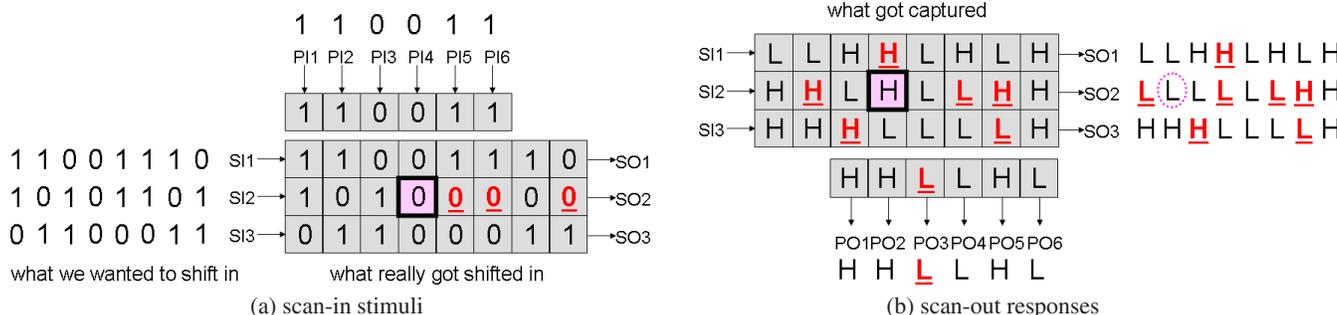
of 1-Generators FTR and STF are true subsets of the faulty bits of a Stuck-at-One fault.

Fault Type	Sensitive Bits	Test Responses
Fault free	11001100	11001100
SA0	<b>11001100</b>	<u>00000000</u>
FTF	1100 <b>1100</b>	1100 <u>0100</u>
STR	11001100	<u>10001000</u>
SA1	<b>11001100</b>	<u>11111111</u>
FTR	11001100	11 <u>101110</u>
STF	11001100	110 <u>1</u> 1100

**Table 2:** Sensitive bits and test responses for six common fault types.

The logic test does not explicitly go after faults in the scan chain, but as it utilizes the scan chain(s) to transport test stimuli and responses, a fault in the scan chain(s) will nevertheless in virtually all cases lead to faulty responses. Actually, a single faulty scan flip-flop typically leads to numerous faulty test response bits. This is due to the fact that a scan chain fault is active both during scan-in and scan-out, and during both operations can affect multiple bits. This is illustrated in Figure 1. The figure schematically shows the stimulus and response bits for a circuit with six primary inputs (PI1, . . . , PI6), three scan chains (between SI1 and SO1, between SI2 and SO2, and between SI3 and SO3) of length eight flip-flops each, and six primary outputs (PO1, . . . , PO6). Bit 5 of Chain 2 (indicated by a fat black square) suffers from a SA0 fault, i.e., all bit values scanned through this flip-flop become zero. (This case is used as a running example throughout this entire paper.)

Figure 1(a) shows which stimuli we intended to apply from the external test equipment, and, due to the single fault in Bit 5 of Chain 2, which stimuli really got scanned in. Note that the single faulty scan flip-flop already leads here to three corrupted stimulus bits (indicated by an underline), all in Chain 2. Figure 1(b) shows both which test responses got captured after the launch/capture cycle, and subsequently which test responses got scanned out to the external test equipment. The three corrupted stimulus bits in Chain 2 have spread to seven corrupted captured response bits, now in all chains and some primary outputs. Upon scan-out, the number of faulty responses in Chain 2 still grows larger, as the bad scan flip-flop affects even more bits during scan-out. In total, we observe nine faulty response bits! Note that the response in Chain 2, Bit 7 (indicated by means of a dashed circle) is correct, but only because it got corrupted twice; once during scan-in, and once during scan-out.



**Figure 1:** One faulty scan flip-flop typically leads to many faulty test responses.

<sup>1</sup>In this paper, scan chains shift from left to right, and we denote scan chain test patterns such that the right-most bit is shifted in first into the scan chain.

<sup>2</sup>In this paper, scan chain bits are numbered from 1 (at the downstream scan-out side) upwards (to the upstream scan-in side).

### 3 Related Prior Work

An effective scan chain test is able to determine the fault type, and, in case of multiple parallel scan chains, also pinpoint which scan chain is faulty. However, the scan chain test does *not* provide sufficient information to determine which scan flip-flop is faulty within a faulty scan chain; as the scan chain test is based on shift operation only, a faulty response is independent from the index number of the faulty flip-flop.

An overview of solution approaches to this problem is recently published by Huang et al. [9]. Solutions include adding extra on-chip hardware [10–12] or generating additional diagnostic test patterns [13, 14]. Although effective, these approaches are intrusive w.r.t. design and/or test flow.

A third solution approach is to exploit the fail data of the logic test (in addition to the fail data of the scan chain test) to perform scan chain diagnosis [15, 16]. Although the test patterns of the logic test were not generated with scan chain diagnosis in mind, they typically provide sufficient data to achieve accurate diagnosis results. As this approach does require only an additional diagnosis software run, and neither extra design-for-test hardware nor extra test patterns, the approach has gained quite some popularity in industrial settings. In this paper, we refer to this approach as the *conventional* scan chain diagnosis approach. It is described in more detail in the following four paragraphs.

Once the failing scan chain and fault type are determined by means of diagnosis of the scan chain test fails, the conventional approach consists of two steps: (1) *Bound Calculation* and (2) *Score Calculation*. Bound Calculation sets upper- and lowerbounds on the range of suspect flip-flops (formally excluding the flip-flops outside the bounds as fault candidates), while Score Calculation assigns suspect scores to the flip-flops within the bounds.

In Bound Calculation, all logic test patterns are iteratively re-simulated and compared with the fail data as observed on the ATE. Figure 2, which builds on the example from Figure 1, illustrates the logic re-simulation of a single logic test pattern during Bound Calculation. Diagnosis of the scan chain test fails has determined that Chain 2 suffers from a SA0 fault. In Bound Calculation, the logic test patterns are re-simulated with an X value in all flip-flops possibly affected by the fault during scan-in. This includes the sensitive flip-flops between the bounds, and all sensitive flip-flops downstream of them. Initially, the bounds are at the two extremes of this scan chain (see Figure 2(a)). The logic simulation preserves these X values; we can see them spread out to many responses, both in Chain 2 as well as elsewhere (see Figure 2(b)).

Subsequently the comparison between what came out of the logic re-simulation and the fail data actually observed on the ATE is as depicted in Figure 3. ATE fail bits are denoted as underlined red bold bits. The X-marked values might be due to faults occurring during scan-in and are ignored during the comparison (dark-gray). The remaining fails can only be caused during scan-out. Starting on the downstream (scan-out) side of Chain 2 and going upstream, we search for the first failing bit, and place our new upperbound for the faulty scan flip-flop location left of it. In the example, the upperbound (UB) is set at Bit 5. The rationale for this UB setting is that scan-out fails affect only bits in or upstream of the faulty flip-flop. Similarly, the new lowerbound for the faulty scan flip-flop location is set left of the last passing sensitive bit downstream of the upperbound; in the example, the last passing sensi-

tive bit is Bit 1, so the lowerbound (LB) is set at Bit 2. The rationale for this LB setting is that the sensitive Bit 1 would have failed in case it had passed the faulty scan flip-flop upon scan-out; the fact that it did not, indicates that Bit 1 must be downstream of the faulty scan flip-flop.

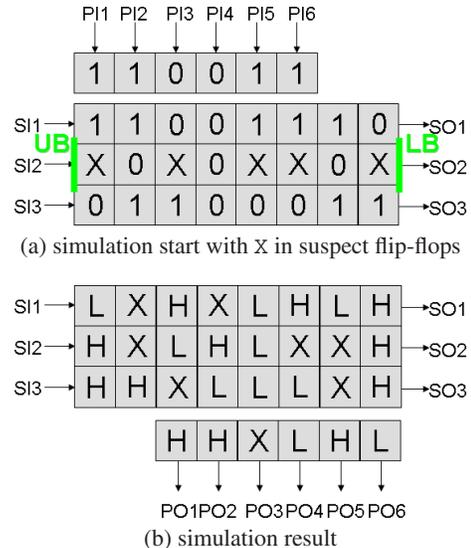


Figure 2: Simulation with Xs during Bound Calculation.

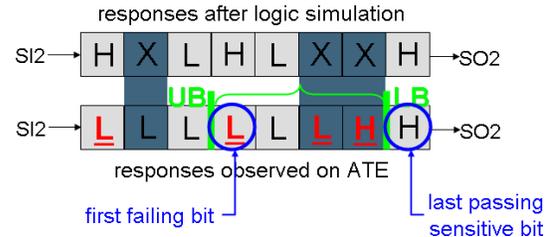


Figure 3: Comparison and bound setting during Bound Calculation.

This process is executed for all logic patterns and the upper- and lowerbound are set as narrow as indicated by any of the patterns. This process can be repeated iteratively, where every new run starts with tighter bounds and hence less Xs, until no further improvement of the bounds is obtained.

The conventional scan chain diagnosis approach assumes a single permanent scan chain fault. There are only few papers that address the problem of scan chain diagnosis in the presence of an intermittent fault. Ghosh et al. worked on this problem for core-based SOCs [17], but their diagnosis accuracy is only down to the core level, and does not identify individual failing scan flip-flops. Most work on scan chain diagnosis for intermittent faults is published by an author team led by Huang [4, 5, 18]. They provide both theoretical insight [4] in and industrial evidence [5] for the actual occurrence of intermittent scan chain faults. For possibly intermittent FTF and FTR fault types, assuming a single [5] or multiple [18] faults per scan chain, they present enhanced upper- and lowerbound calculation procedures and a statistical score calculation. Their approach has several shortcomings. Firstly, they have ignored the potential issues with intermittent faults during the scan chain test, despite the fact that an accurate diagnosis of the scan chain test results (1) is indispensable for determining whether we need to activate either logic or scan chain diagnosis, and (2) is a required pre-ample for determining faulty scan chain and fault type in

scan chain diagnosis. We address this in issue in Section 4. Secondly, [5] remarks that lowerbound calculation in the presence of intermittent faults is “kind of tricky”, without providing further details. In Section 5 of our paper, we analyze and explain the root cause of the problems with intermittent faults during bound calculation, showing that the bounds calculated in the conventional way might simply be incorrect. We also provide experimental data that illustrates the magnitude of the problem: how far off can the bounds be, and how often does that happen? Subsequently, we present an always-correct new way of bound calculation. Finally, the experimental results in [5, 18] are based on only a few individual cases of faulty scan flip-flops, and result in a relatively large number of fault candidates, especially for intermittence probability  $p = 60\%$ . Our experimental results in Section 5 provide a more thorough analysis, considering as fault source scan flip-flops at structured intervals all over the scan chain length and also considering more and lower levels of intermittence probability. Our improved bound calculation procedures are always correct, and produce very tight bounds (even for intermittence probabilities as low as  $p = 20\%$  or  $p = 10\%$ ), leaving little work for subsequent Score Calculation.

## 4 Scan Chain Test Pattern

We define a *fault activation sequence* for fault type  $f$  as a bit sequence within a scan chain test pattern that activates (or, for an intermittent fault, *potentially* activates) a fault of type  $f$ . Table 3 shows the fault activation sequences for each fault type. For example, a SA1 fault is activated by a 0, while a FTF fault requires a 01 transition.

Fault Type	Fault Activation Sequence	Faulty Response
SA0	<b>1</b>	<b>0</b>
FTF	<b>01</b>	<b>00</b>
STR	<b>10</b>	<b>00</b>
SA1	<b>0</b>	<b>1</b>
FTR	<b>10</b>	<b>11</b>
STF	<b>01</b>	<b>11</b>

**Table 3:** Fault activation sequences and corresponding faulty responses for the six fault types.

For fault type  $f$ , let  $a_f$  denote the fault activation count of  $f$  for a given scan chain test pattern, i.e., the number of times the fault activation sequence of  $f$  occurs in the test pattern. The *total fault activation (TFA)* is the sum of fault activation counts for all six fault types:

$$TFA = \sum_f a_f \quad (4.1)$$

For example: for scan chain test pattern 11001100,  $a_{SA0} = a_{SA1} = 4$ ,  $a_{FTR} = a_{STR} = 2$ , and  $a_{FTF} = a_{STF} = 1$ . In this example,  $TFA = 12$ .

We expect from a scan chain test to accurately detect if a scan chain is faulty, and if so, diagnose what fault type it suffers from. The conventional scan chain test pattern 11001100 is a sufficient test pattern for the diagnosis of a single *permanent* fault, as it meets the following two requirements: every fault type  $f$  has at least one fault activation sequence within the test pattern, i.e.,  $a_f \geq 1$  for all  $f$ , and different fault types lead to distinguishable test responses. However, in the case of *intermittent* faults, the conventional scan chain test pattern might not

be sufficient, due to the fact that the fault effect can be absent at random moments in time.

The first problem with the conventional scan chain test pattern 11001100 is that it is short; only eight bits. This means that all six faults considered have a low fault activation count. If we are unlucky, our scan chain suffers from an intermittent fault that just happens *not* to occur at these few crucial moments, and hence goes unnoticed. To increase the small activation counts, it is required to increase the length of the scan chain test pattern. The associated cost is that this will increase the test length, i.e., the test application time and the required vector memory on the ATE. Fortunately, since the conventional scan chain test pattern consumes actually only a very tiny fraction of the overall test length, we can easily increase its length substantially, without incurring a significant impact on the overall test length. Let  $n$  denote the length of the scan chain test pattern (in bits),  $t$  the number of logic test patterns, and  $s$  the maximum scan chain length (in bits), then test length  $T$  (in clock cycles) is expressed by

$$T = ((s + 1) \cdot t + s) + \quad (4.2)$$

$$(n + s) \quad (4.3)$$

where term (4.2) denotes the contribution of the logic test, and term (4.3) denotes the contribution of the scan chain test. For an example industrial SOC, typical parameters are  $n = 8$ ,  $t = 5000$  and  $s = 10000$  [19]. In this example, the parameters yield a total test length  $T = 50,025,008$ . Clearly, the contribution of the scan chain test pattern length  $n$  in the overall test length  $T$  is negligible. We can increase  $n$  with a factor 10,000 and still it contributes for only about 0.18% to  $T$ .

The second problem is to define the best content of the pattern for a given length  $n$  of the scan chain test pattern. The highest *TFA* is obtained by a sequence of alternating bit values, viz. 1010...1010. Unfortunately, this pattern suffers from poor diagnostic resolution. An expected 0 bit that fails as 1 could be caused by an (intermittent) version of any of the three 1-Generators: SA1, FTR, or STF. Similarly, an expected 1 bit that fails as 0 could be caused by an (intermittent) version of any of the three 0-Generators: SA0, FTF, or STR. As it does not provide a conclusive diagnostic verdict, we conclude that a sequence of alternating bit values is not a suitable scan chain test pattern.

The scan chain test pattern with the next-highest *TFA* value is an  $n$ -bit sequence with consecutive zeros and ones in runs of length two: 1100...1100. We first analyze this pattern for the 1-Generators SA1, FTR, and STF. For  $n = 8$ , a FTR can affect bits marked by  $x$ , while an STF can affect bits marked by  $y$ : 11xy11xy11x0. We make the following two observations. First, bits that are affected by FTR and STF faults are mutually exclusive. Second, any fail of both bits  $x$  and  $y$  must be due to a SA1, given our single-fault assumption.

Based on these two observations, we formulate three cases for the bits in the test responses.

1. If at least one  $x$  bit failed *and* at least one  $y$  bit failed, the fault is a SA1.
2. If *all* failing test response bits are of type  $x$ , the fault is either a FTR or a SA1.
3. If *all* failing test response bits are of type  $y$ , the fault is either a STF or a SA1.

The first case yields a conclusive diagnosis verdict. Some failing bits point towards a SA1 or FTR; other failing bits point towards a SA1 or STF. Under the assumption of a single fault, the fault present must be an intermittent SA1.

In the second case, it is not possible to immediately diagnose the fault effect to FTR or SA1. However, for a scan test sequence 1100...1100 of  $n$  bits, there are  $n/2$  0s, which are possibilities to detect a SA1, and  $n/4$  10 sequences, which are possibilities to detect a FTR. The probability that the fault is a SA1, and that all failing bits are of type  $x$ , is equal to the probability that no  $y$  bits are failing and given by:

$$(1 - p)^{(n/4)} \quad (4.4)$$

where  $p$  is the intermittence probability and  $n$  is the number of bits in the scan chain test pattern. This probability of the fault being a SA1 decays very fast with increasing  $n$ ; for a probability of intermittence as low as  $p = 0.1$  and  $n = 1000$ , the chance that this fault was a SA1 is less than  $1 \times 10^{-11}$ , and hence negligible.

A similar argument applies to Case 3. Therefore, under the requirement that  $n$  is sufficiently large, we can rewrite the three cases as follows.

1. If at least one  $x$  bit failed *and* at least one  $y$  bit failed, the fault is a SA1.
2. If *all* failing bits are of type  $x$ , the fault is a FTR.
3. If *all* failing bits are of type  $y$ , the fault is a STF.

The analysis above is for the 1-Generator faults (SA1, FTR, and STF); the analysis for the 0-Generators (SA0, FTF, STR) is similar.

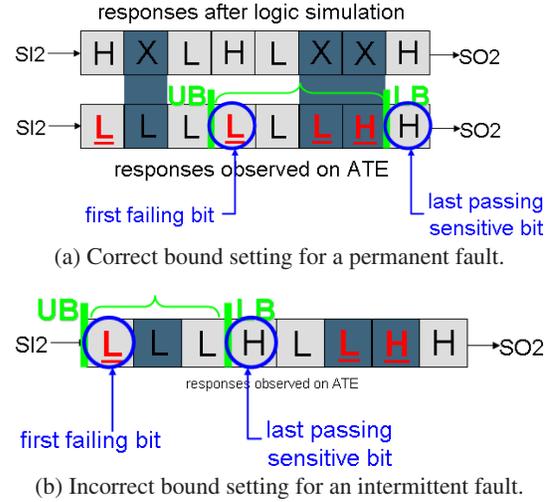
The scan chain test pattern that provides maximized *TFA* and sufficient diagnostic distinction between the six fault types considered is 1100...1100 for large-enough length  $n$ .

## 5 Bound Calculation

### 5.1 Conventional Bounds Go Wrong

We use the running example that has been used throughout this paper (in Figures 1, 2, and 3) to illustrate how the conventional Bound Calculation algorithm might produce wrong results for an intermittent fault. The example contains three scan chains. Bit 5 of Chain 2 contains a SA0 fault. In the first iteration of Bound Calculation (see Figure 3), the upperbound UB was set at Bit 5 and the lowerbound LB at Bit 2; this is again depicted in Figure 4(a). These bounds are correct, in the sense that the actual faulty scan flip-flop is contained within the bounds. Imagine now a situation in which the fault in Bit 5 is not a permanent, but an intermittent SA0 fault. Due to its intermittent behavior, the test response in Bit 5 of the Chain 2 was not faulty ( $\underline{L}$ ), but actually correct ( $H$ ); this situation is depicted in Figure 4(b). The subsequent Bound Calculation will now calculate the upper- and lowerbound to be at resp. Bits 8 and 6. The calculated bound interval is non-overlapping with the original case, does not include the actual faulty scan flip-flop, and hence is *wrong*.

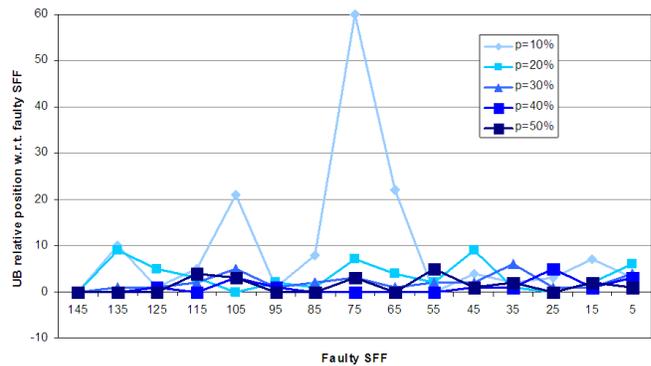
Experimental results confirm that intermittent faults indeed cause upperbounds to be loose and lowerbounds to be incorrect. We have performed experiments on ISCAS'89 benchmark circuit s38584 [20].



**Figure 4:** Bound setting for (a) a permanent and (b) an intermittent SA0 fault in Bit 5.

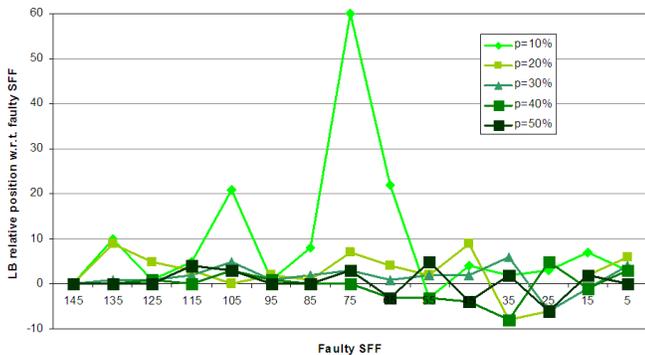
The full-scan version of this circuit contains 1452 scan flip-flops, which we partitioned over 10 balanced scan chains of lengths 145 and 146 flip-flops. In total 127 stuck-at logic test patterns were generated with NXP's in-house ATPG tool AMSAL. We simulated the fault effect of a single faulty scan flip-flop on these test patterns, for all scan flip-flops and six fault types. The intermittent fault effect was simulated by pseudo-random fault activation, which we performed at various intermittence probabilities  $p$ . Subsequently, upper- and lowerbounds were computed using the conventional procedures.

Figures 5 and 6 show the results of resp. upper- and lowerbound calculations for a single SA1 fault for scan flip-flops at intervals of ten in Chain 1, at various levels of intermittence probability  $p$ . The horizontal axes of the graphs show the index number of the faulty scan flip-flop, while the vertical axes show the relative position (in bits) of the calculated bound to the actual faulty scan flip-flop. Relative position 0 means that the bound is spot-on at the faulty scan flip-flop; a positive relative position indicates that the bound is upstream of the faulty scan flip-flop, while a negative relative position indicates that the bound is downstream of the faulty scan flip-flop. Note that a correct upperbound must be at a non-negative relative position (i.e., zero or positive), while a correct lowerbound must be at a non-positive relative position (i.e., zero or negative).



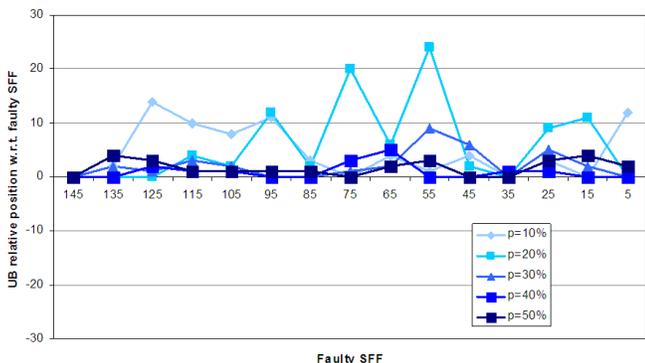
**Figure 5:** Relative position of the upperbound for a single SA1 fault in varying scan flip-flop locations in Chain 1 of circuit s38584 and for varying intermittence probabilities  $p$ .

Figure 5 shows that for all cases displayed, the upperbound is non-negative, and hence correct. However, often the upperbound is quite loose, and, as expected, this effect increases for decreasing intermittence probabilities. Figure 6 shows that for the majority of cases, the lowerbound is positive and hence *incorrect*. As expected, this effect increases for decreasing intermittence probabilities. Also, the effect coincides with those cases where the upperbound is loose. This can be explained, as the definition of the lowerbound is relative to the upperbound, and the looser the upperbound, the more chance for the lowerbound to be wrong.



**Figure 6:** Relative position of the lowerbound for a single SA1 fault in varying scan flip-flop locations in Chain 1 of circuit s38584 and for varying intermittence probabilities  $p$ .

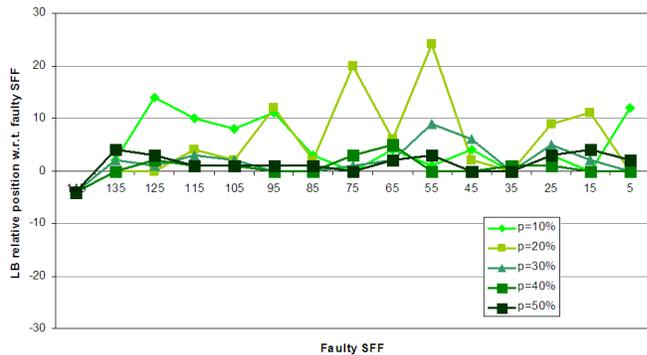
Figures 7 and 8 show similar upper- and lowerbound results on circuit s38584 for a single FTF fault in Chain 5 at varying scan flip-flop locations and for varying levels of intermittence probability  $p$ . Similar effects were observed for the other scan chains, and the other fault types, and in fact also for other circuits.



**Figure 7:** Relative position of the upperbound for a single FTF fault in varying scan flip-flop locations in Chain 5 of circuit s38584 and for varying intermittence probabilities  $p$ .

## 5.2 New, Correct Lowerbound Calculation

The fundamental reason for the conventional lowerbound calculation to go wrong on intermittent faults is that it is searching for a last passing sensitive bit. With an intermittent fault, a normally failing bit might incidentally turn into a passing one, and lead to an incorrect lowerbound, upstream of the actual faulty flip-flop. Hence, we have developed a new lowerbound calculation routine, which does not become incorrect due to intermittent faults. This is achieved by having

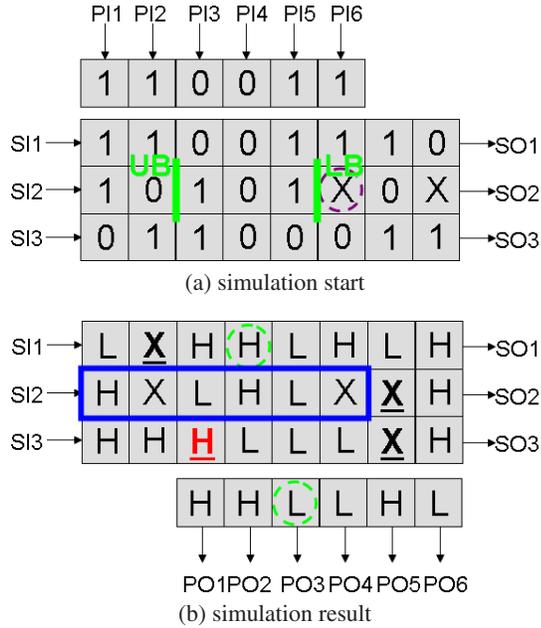


**Figure 8:** Relative position of the lowerbound for a single FTF fault in varying scan flip-flop locations in Chain 5 of circuit s38584 and for varying intermittence probabilities  $p$ .

the new routine not search for passing bits, but for *failing* bits instead. Intermittent faults can cause the new lowerbound to be looser than strictly necessary, but *not* wrong.

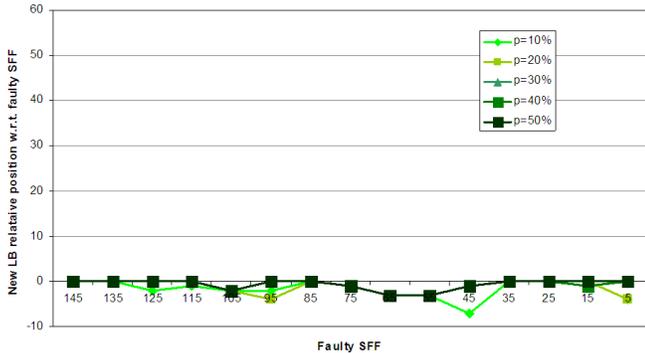
The new lowerbound calculation iterates over the scan flip-flops in the faulty scan chain, starting at the upperbound flip-flop and going downstream. For each scan flip-flop  $F$  under consideration, we temporarily assume it is *the* faulty flip-flop. We simulate all test patterns, while masking upon scan-in all sensitive bits downstream of and including  $F$  by  $X$ . Under the assumption that  $F$  is indeed the faulty scan flip-flop, all fault effects caused by scan-in are now masked by  $X$  values. After simulation, any non-masked fails observed should therefore be caused by the scan-out operation, and consequently be located upstream of or at  $F$ . If this is indeed the case, we move the lowerbound one bit downstream such that  $F$  is now included in the bound range, and go to the next iteration. If after the simulation we encounter any non-masked fails downstream of  $F$  or in the other, non-faulty scan chains, we know that our temporary assumption that  $F$  is *the* faulty scan flip-flop is not correct; upon scan-in, fails also occurred upstream of  $F$ . Hence, given the fact that there can only be one faulty scan flip-flop,  $F$  must be downstream of the lowerbound. In that case, the iteration is terminated with the new lowerbound set immediately upstream of  $F$ .

Figure 9 depicts a snapshot on the algorithm in action. Before this snapshot was taken, the upperbound has been calculated to be at Bit 6 of Chain 2. Subsequently, Bits 6 down to 4 have already been considered as *the* faulty flip-flop. At the moment of the snapshot, the algorithm has arrived at Bit 3 set as flip-flop  $F$  (indicated by the dashed purple circle in Figure 9(a)). We run a simulation with the sensitive bits downstream of and including  $F$  masked by  $X$ . For a SA0 fault, sensitive bits are bits with logic value '1'. Assuming  $F$  is indeed the faulty flip-flop, after simulation non-masked faulty responses should only occur upstream of or at  $F$  (indicated by the fat dark-blue rectangle in Figure 9(b)). Faulty responses are denoted by a bold underlined bit, whereas masked responses are denoted by bit value  $X$ . Two responses that would have been faulty if the fault at hand was permanent are indicated with a light-green dashed circle; due to the intermittent behavior of the fault, they are fault-free. Figure 9(b) shows one non-masked faulty response outside the dark-blue rectangular box, viz. Bit 6 in Chain 3 (in red). This proves that our temporary assumption that  $F$  is *the* faulty flip-flop is incorrect. Hence, the calculation is terminated and the final lowerbound is set immediately upstream of  $F$ .



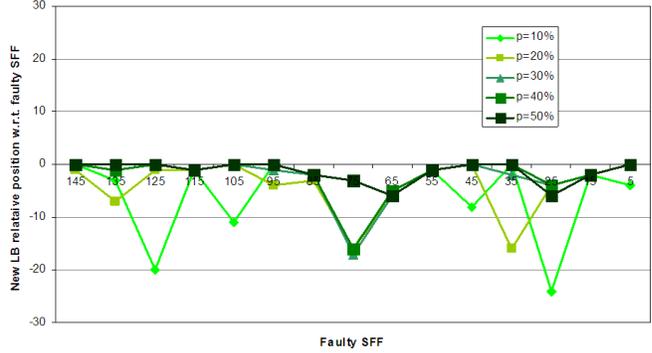
**Figure 9:** Snapshot of the new lowerbound calculation in action, with Bit 3 in Chain 2 figuring as potentially faulty flip-flop  $F$ .

Figures 10 and 11 show results of the new lowerbound calculation for circuit s38584. Figure 10 shows the relative position of the new lowerbound with respect to the actual faulty scan flip-flop for a single SA1 fault in varying scan flip-flop locations of Chain 1 for varying intermittence probabilities  $p$ . We see that the relative position of the new lowerbound is non-positive, and hence correct, for all cases. Also note that the new lowerbound is quite tight, viz. not more than seven positions away from the actual faulty scan flip-flop, even for  $p = 10\%$ . Figure 11 shows the same data for a single FTF fault in Chain 5. Also here, the new lowerbound is correct for all cases. The new lowerbound is a bit less tight, up to a case with position  $-24$  for  $p = 10\%$ ; given the low intermittence probability, we still consider this an acceptable result.



**Figure 10:** Relative position of the *new* lowerbound for a single SA1 fault in varying scan flip-flop locations in Chain 1 of circuit s38584 and for varying intermittence probabilities  $p$ .

Table 4 compares the results of conventional and new Bound Calculation for a single FTF fault in 14 scan flip-flop locations at regular intervals in Chain 5 of circuit s38584 and for varying intermittence probabilities  $p$ . Columns 2 and 4 of the table show the number of *misses* for resp. the conventional and new method. A miss is defined here as a case where the actual faulty scan flip-flops was not included



**Figure 11:** Relative position of the *new* lowerbound for a single FTF fault in varying scan flip-flop locations in Chain 5 of circuit s38584 and for varying intermittence probabilities  $p$ .

in the calculated bounds. We can see from the table that for the conventional Bound Calculation, the percentage of misses is quite significant and increases with decreasing intermittence probability  $p$ . The new Bound Calculation method, on the other hand, shows no misses at all, not even for the low intermittence probabilities. Columns 3 and 5 of Table 4 show the average size of the calculated bounds interval, i.e., upperbound minus lowerbound. Note that all bound intervals calculated are small. The bound interval of the conventional Bound Calculation method is the smallest, but, as we know from Column 2 in the table, often wrong. It is of no use to have a small bound interval if the actual faulty scan flip-flop is not included in it, as neither Score Calculation nor PFA will be able to pinpoint the actual fault cause. We observe that the bound intervals of the new method are slightly larger, and increase in size for decreasing intermittence probability  $p$ . However, their strong point is that the actual faulty scan flip-flop is always included in the bound interval.

Intermittence Probability	Conventional		New	
	Misses	Av. UB-LB	Misses	Av. UB-LB
$p = 100\%$	0 = 0%	1.36	0	2.43
$p = 80\%$	5 = 36%	1.36	0	3.00
$p = 60\%$	4 = 29%	1.36	0	2.71
$p = 40\%$	8 = 57%	1.36	0	4.71
$p = 20\%$	12 = 86%	1.36	0	8.29

**Table 4:** Comparison of conventional and new Bound Calculation results for a single FTF fault in varying scan flip-flop locations in Chain 5 and for varying intermittence probabilities  $p$

## 6 Conclusion

In this paper, we addressed the topic of scan chain diagnosis under the assumption of a single, but possibly intermittent fault. We showed that due to the short length of the conventional scan chain test pattern, it has a high probability to miss an intermittent fault. The solution is to extend the scan chain test pattern into a much longer sequence. We demonstrated that for a typical industrial IC, the length of the scan chain test pattern, and hence the detection probability of an intermittent fault, can easily be increased with a factor 10,000 without significantly impacting the overall test length. We also showed that a sufficiently extended version of the conventional scan test pattern, 1100...1100, provides sufficient diagnostic distinction between the six fault types

under consideration.

Subsequently we demonstrated that the conventional algorithm for Bound Calculation might go wrong for intermittent faults. The upper-bound might become arbitrarily loose, but at least is still correct. The conventional lowerbound, however, might simply be wrong, such that the bounded interval does not include the actual faulty scan flip-flop, and the scan chain diagnosis algorithm has no chance of producing a correct diagnosis result. We showed that this scenario is quite likely to happen, especially if the intermittence probability is low. We presented a new lowerbound calculation algorithm, which, at the expense of increased compute time, produces an always correct lowerbound, also in the case of an intermittent scan chain fault. Our experimental results show that the resulting new bound interval is both correct and narrow.

## Acknowledgments

During the work described in this paper, Joanna Siew was M.Sc. student at Linköpings Universitet, and Erik Jan Marinissen was employed by NXP Semiconductors. The work of Dan Adolffson and Erik Jan Marinissen was partly supported by the MEDEA+ 2A702 project 'NanoTest'. The authors thank Andreas Glowatz, Sandeep K. Goel, Friedrich Hapke, Camelia Hora, and Sergej Schwarz for their help and support during our work on scan chain diagnosis.

## References

- [1] Manish Sharma et al. Efficiently Performing Yield Enhancements by Identifying Dominant Physical Root Cause from Test Fail Data. In *Proceedings IEEE International Test Conference (ITC)*, October 2008.
- [2] Stefan Eichenberger et al. Towards a World Without Test Escapes: The Use of Volume Diagnosis to Improve Test Quality. In *Proceedings IEEE International Test Conference (ITC)*, October 2008.
- [3] Sandip Kundu. On Diagnosis of Faults in a Scan-Chain. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 303–308, April 1993.
- [4] Yu Huang et al. Intermittent Scan Chain Fault Diagnosis Based on Signal Probability Analysis. In *Proceedings Design, Automation, and Test in Europe (DATE)*, pages 1072–1077, February 2004.
- [5] Yu Huang et al. Statistical Diagnosis for Intermittent Scan Chain Hold-Time Fault. In *Proceedings IEEE International Test Conference (ITC)*, pages 319–328, September 2003.
- [6] Kuen-Jong Lee and Melvin A. Breuer. A Universal Test Sequence for CMOS Scan Registers. In *Proceedings IEEE Custom Integrated Circuits Conference (CICC)*, pages 28.5/1–4, May 1990.
- [7] Samy R. Makar and Edward J. McCluskey. ATPG for Scan Chain Latches and Flip-Flops. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 364–369, April 1997.
- [8] Fan Yang et al. On the Detectability of Scan Chain Internal Faults – An Industrial Case Study. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 79–84, April 2008.
- [9] Yu Huang, Ruifeng Guo, Wu-Tung Cheng, and James Chien-Mo Li. Survey of Scan Chain Diagnosis. *IEEE Design & Test of Computers*, 25(3):240–248, May/June 2008.
- [10] James L. Schafer, Fred A. Policastri, and Richard J. McNulty. Partner SRLs for Improved Shift Register Diagnostics. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 198–201, April 1992.
- [11] Samantha Edirisooriya and Geetani Edirisooriya. Diagnosis of Scan Path Failures. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 250–255, April 1995.
- [12] Wu Yuejian. Diagnosis of Scan Chain Failures. In *Proceedings IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT)*, pages 217–222, November 1998.
- [13] James Chien-Mo Li. Diagnosis of Single Stuck-At Faults and Multiple Timing Faults in Scan Chains. *IEEE Transactions on VLSI Systems*, 13(8):708–718, June 2005.
- [14] James Chien-Mo Li. Diagnosis of Multiple Hold-Time and Setup-Time Faults in Scan Chains. *IEEE Transactions on Computers*, 54(11):1467–1472, November 2005.
- [15] Kevin Stanley. High-Accuracy Flush-and-Scan Software Diagnostic. *IEEE Design & Test of Computers*, 18(6):56–62, November/December 2001.
- [16] Ruifeng Guo and Srikanth Venkataraman. A Technique for Fault Diagnosis of Defects in Scan Chains. In *Proceedings IEEE International Test Conference (ITC)*, pages 268–277, October 2001.
- [17] Swaroop Ghosh et al. Scan Chain Fault Identification Using Weight-Based Codes for SoC Circuits. In *Proceedings IEEE Asian Test Symposium (ATS)*, pages 210–215, November 2004.
- [18] Yu Huang et al. Efficient Diagnosis for Multiple Intermittent Scan Chain Hold-Time Faults. In *Proceedings IEEE Asian Test Symposium (ATS)*, pages 44–49, November 2003.
- [19] Ozgur Sinanoglu et al. Test Data Volume Comparison of Monolithic vs. Modular SOC Testing. *IEEE Design & Test of Computers*, 26(3):25–37, May/June 2009.
- [20] Franc Brglez, David Bryan, and Krzysztof Kozminski. Combinational Profiles of Sequential Benchmark Circuits. In *Proceedings International Symposium on Circuits and Systems (ISCAS)*, pages 1924–1934, May 1989.