Built-in Test for Hidden Delay Faults

Kampmann, Matthias; Kochte, Michael A.; Liu, Chang; Schneider, Eric; Hellebrand, Sybille; Wunderlich, Hans-Joachim

IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), Volume: 38 , Issue: 10 , Oct. 2019, Page(s): 1956 - 1968

doi: https://doi.org/10.1109/TCAD.2018.2864255

Abstract: Marginal hardware introduces severe reliability threats throughout the life cycle of a system. Although marginalities may not affect the functionality of a circuit immediately after manufacturing, they can degrade into hard failures and must be screened out during manufacturing test to prevent early life failures. Furthermore, their evolution in the field must be proactively monitored by periodic tests before actual failures occur. In recent years, small delay faults (SDFs) have gained increasing attention as possible indicators of marginal hardware. However, SDFs on short paths may be undetectable even with advanced timing aware ATPG. Faster-than-at-speed test (FAST) can detect such hidden delay faults (HDFs), but so far FAST has mainly been restricted to manufacturing test. This paper presents a fully autonomous built-in self-test approach for FAST, which supports in-field testing by appropriate strategies for test generation and response compaction. In particular, the required test frequencies for HDF detection are selected, such that hardware overhead and test time are minimized. Furthermore, test response compaction handles the large number of unknowns (X-values) on long paths by storing intermediate MISR-signatures in a small on-chip memory for later analysis using X-canceling transformations. A comprehensive experimental study demonstrates the effectiveness of the presented approach. In particular, the impact of the considered fault size is studied in detail.

Preprint

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2864255, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

Built-in Test for Hidden Delay Faults

Matthias Kampmann[1], Michael A. Kochte[2], Chang Liu[2], Eric Schneider[2]
Sybille Hellebrand[1], Hans-Joachim Wunderlich[2]
[1]University of Paderborn, Warburger Str. 100, 33098 Paderborn, Germany
[2]University of Stuttgart, Pfaffenwaldring 47, 70569 Stuttgart, Germany

Abstract— Marginal hardware introduces severe reliability threats throughout the life cycle of a system. Although marginalities may not affect the functionality of a circuit immediately after manufacturing, they can degrade into hard failures and must be screened out during manufacturing test to prevent early life failures. Furthermore, their evolution in the field must be proactively monitored by periodic tests before actual failures occur. In recent years small delay faults have gained increasing attention as possible indicators of marginal hardware. However, small delay faults on short paths may be undetectable even with advanced timing aware ATPG. Faster-than-at-speed test (FAST) can detect such hidden delay faults, but so far FAST has mainly been restricted to manufacturing test.

This paper presents a fully autonomous built-in self-test (BIST) approach for FAST, which supports in-field testing by appropriate strategies for test generation and response compaction. In particular, the required test frequencies for hidden delay fault detection are selected, such that hardware overhead and test time are minimized. Furthermore, test response compaction handles the large number of unknowns (X-values) on long paths by storing intermediate MISR-signatures in a small on-chip memory for later analysis using X-canceling transformations. A comprehensive experimental study demonstrates the effectiveness of the presented approach. In particular, the impact of the considered fault size is studied in detail.

Key words—Faster-than-at-Speed-Test, BIST, in-field test, reliability

I. INTRODUCTION

Advanced CMOS technologies have found their way into safety-critical applications. Self-driving cars are a prominent example, where nano-scale design enables the integration of complex control systems such as advanced driver assistance systems, but also introduces severe reliability threats. Besides new defect mechanisms and parameter variations, marginal circuit structures pose a particular problem. Immediately after manufacturing, marginal hardware may not affect the functionality of a circuit, however, it may gradually evolve into a hard failure and cause an **E**arly **L**ife **F**ailure (ELF). Similarly, in the field, originally fault free hardware may degrade into marginal hardware before it finally fails. In the first case, a major goal is to replace costly and time-consuming burn-in tests by ELF prediction during manufacturing test [1]. To handle the second case, the hardware must be monitored with periodic in-field tests [2], [3], [4], [5], [6]. Here, the challenge is twofold: On the one hand, in-field test must cope with limited resources, and on the other hand, the test must be proactive, i.e. marginalities must be identified before the circuit functionality is affected.

As shown in the Stanford ELF-Project, **S**mall **D**elay **F**aults (SDF) can point to marginal hardware [7]. For example, gate-oxide defects can result in small delays before actual hard failures occur. Testing for small delay faults thus helps to predict potential circuit failures and has therefore gained increasing attention in recent years [8]. However, depending on the slack of the propagation path, a small delay fault may not be detectable by a standard transition test. Timing aware ATPG tries to ensure that propagation paths are long enough [9], [10], supports hazard-free fault propagation or selects paths with switching times in certain clock intervals [11], [12], [13]. Nevertheless, some small delay faults can be propagated along short paths only, such that they are undetectable even by advanced timing aware ATPG. **F**aster-than-**a**t-**s**peed **T**est (FAST) targets these **H**idden **D**elay **F**aults (HDFs) by overclocking the circuit, typically using several frequencies up to three times higher than the nominal frequency [14], [15], [16], [17]. Silicon experiments have already demonstrated the effectiveness of this strategy [18], [19].

So far, research on FAST has mainly focused on manufacturing test, where an ATE is available for storing (encoded) test patterns and responses and taking over test control. To alleviate the need for costly high speed ATEs, programmable schemes for on-chip clock generation have been developed [20], [21], [22], [23], [24], [25]. These clock generators can also be used as basic building blocks for in-field testing. However, to achieve a fully autonomous built-in FAST, additional challenges must be addressed.

Firstly, FAST frequencies are tuned to make delays visible on short paths. However, this may lead to unpredictable values (X-values) at the end of long paths and prevent the computation of a unique expected "signature" after test response compaction. Despite comprehensive research on X-handling in built-in self-test (BIST) or embedded test [26], [27], [28], [29], [30], [31], [32], there is still a gap to fill. The distribution of X-values depends on the distribution of long paths and changes with the test frequencies. Even for a fixed frequency the distribution may vary due to parameter variations.

This paper combines and extends preliminary work published at ITC14 and ATS15.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2864255, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

Furthermore, the X-values will be clustered at outputs driven by many long paths, and only a few schemes take into account specific clustering of X-values [33]. With the targeted size of small delays, the test patterns may also change. The applied X-handling scheme must therefore be flexible and independent of the test set.

Secondly, hardware overhead and test time depend on the number of FAST frequencies. While during manufacturing test, simply repeating the test for a larger number of frequencies may still be affordable [14], [17], the limited resources during in-field test transform frequency selection into a complex optimization problem. Again, the targeted fault sizes will impact the detection profiles and the achievable solutions.

This paper presents a BIST scheme for FAST, which is suitable for autonomous in-field testing. The scheme is based on the preliminary results published in [34], [35]. It is compatible with the widely used STUMPS architecture [36] and can be deployed with state of the art approaches for built-in deterministic or mixed mode test pattern generation, e.g. [37]. Test response compaction relies on the X-canceling MISR [31], however, to keep the hardware simple, there is no X-canceling logic on-chip. Instead, in case of necessary resets, the intermediate signatures are stored in a small additional memory for later analysis. The scheme is supported by respective algorithms for optimal frequency selection and for minimizing the number of intermediate signatures. Furthermore, a comprehensive experimental study evaluates the presented approach. In particular, impact of the fault size is studied in detail.

The rest of the paper is organized as follows. In Section II the necessary background on small delay faults as well as on X-handling is related to the specific goals of this work, and the proposed BIST architecture is introduced. Subsequently, the problem of optimal frequency selection is formally defined in Section III. It is proven that this problem is NP-complete and the developed algorithms are explained. Section IV focuses on X-handling and describes an approach for minimizing the intermediate signatures. Finally, the experimental results presented in Section V demonstrate that FAST can be effectively employed also for in-field test.

II. PRELIMINARIES

A. Detection Ranges

Small delay defects have become the focus of intensive research in recent years. As a complete introduction into the topic is beyond the scope of this paper, the reader is referred to [8] for a more detailed overview. Within the framework of this paper, a small delay defect is modeled as gate delay fault and characterized by a fault location and a fault size. To keep the model simple, only primary inputs and gate outputs are considered as possible fault locations, and the fault effect is treated as a lumped delay added to the gate output. The model can easily be generalized, for example taking into account specific pin-to-pin delays, too.

Definition 1: A small delay fault is a pair $\varphi = (v, \delta)$, where v is a primary input or a gate output and δ is the delay added to the nominal gate delay.

The additional delay is often measured in terms of the standard deviation σ of the nominal gate delay. In this work, a gate is assumed faulty, if it has an additional delay exceeding 6σ.

As illustrated in Figure 1, fault detection requires a pair of test patterns propagating a delayed transition to the outputs as well as the selection of a proper observation time. To simplify the description, the term test pattern will always refer to a pair of test patterns in the following. For a standard delay test the observation time is set to $t_{nom} = 1/f_{nom}$, where f_{nom} denotes the nominal frequency of the circuit. However, the small delay fault φ in Figure 1 can only be propagated to outputs x and y, and the propagation paths have a slack larger than the fault size. Therefore, φ cannot be detected at time t_{nom}. Choosing a smaller observation time can make the fault visible at outputs x and y, but there is no contiguous interval of suitable observation times. For the given input transition from p_1 to p_2, the fault leads to a slow-to-fall fault in the output waveform of x and to a glitch in the output waveform of y. The fault can only be detected, if the observation time is selected within the interval I_1 or I_2. If the considered fault size is increased, the intervals I_1 and I_2 will also increase in this example.

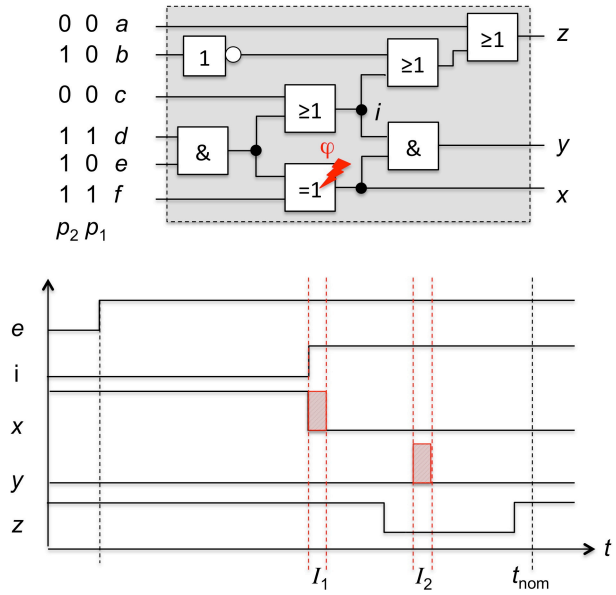

Fig. 1. Small delay fault – example.

However, in general, varying the fault size may completely change the detectability of a small delay fault. Consider the small example in Figure 2 for illustration. Assume that the first AND gate has a delay increasing from δ_1 to δ_2. In the first case, the fault $\varphi_1 = (d, \delta_1)$ leads to a glitch at the output e. It is detectable at any observation time in I_1, but it is hidden at the nominal observation time. In the second case, the fault $\varphi_2 = (d, \delta_2)$ has a larger fault size but cannot be detected at all, because it is masked by input c.

In practice, it also depends on the available test set whether a fault remains hidden or not.

Definition 2: Let φ be a small delay fault, P a set of test patterns, and let f be a frequency. If φ is not detected by P at

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2864255, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

frequency f, then φ is called a *hidden delay fault with respect to P and f*. The set of all hidden delay faults with respect to a test set P and a set of frequencies F is denoted by $\Phi_{HDF}(P, F)$. If only faults of a given size δ are considered, this is denoted by $\Phi_{HDF}(\delta, P, F)$.

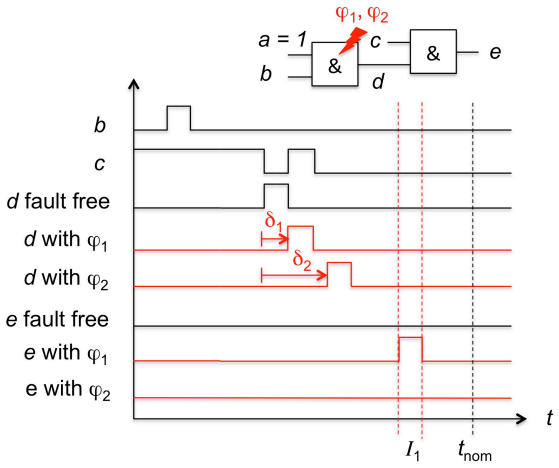

Fig. 2. Impact of the fault size.

The observation times for FAST can be selected within an interval $[t_{min}, t_{nom})$, where $t_{min} = 1/f_{max}$ is determined by the maximum possible frequency f_{max} during test.

Definition 3: Let φ be a small delay fault, P a set of test patterns, and let $t \in [t_{min}, t_{nom})$ be an observation time. Then t is called a *detecting observation time*, if φ is detected by capturing the test responses for P at time t. The set $I(\varphi, P)$ of all detecting observation times is called the *detection range of* φ *with respect to P*.

$I(\varphi, P)$ is the union of the intervals in which the fault free and faulty waveform differ. For each observation time in a detection range $I(\varphi, P)$, there is at least one circuit output and at least one test pattern in P, such that the fault free and faulty test responses are different. As demonstrated by the example of Figure 2, the detection range also depends on the fault size, and the detection ranges for two faults at the same fault site can be completely different.

To reflect pulse filtering in CMOS technology, a pessimistic approach as shown in Figure 3 is followed in this work. If a fault can be detected due to a small glitch below a given threshold, the respective interval is not added to the detection range. In the example this applies to the small glitch between interval I_1 and I_2. If a fault is masked by a small glitch, then the intervals next to the glitch are kept as disjoint intervals, as I_2 and I_3 in the example. The figure also illustrates that differences before the minimum observation time t_{min} are neglected.

In general, several different observation times will be required to hit the detection ranges of all targeted small delay faults. The problem of finding the best observation times will be discussed in detail in Section III.

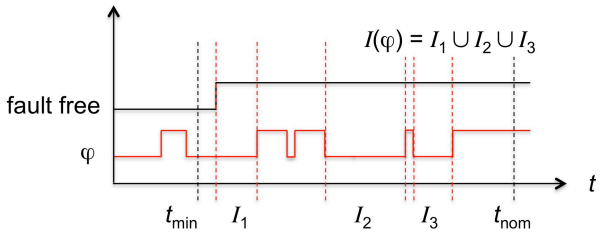

Fig. 3. Detection range of a fault φ.

B. Handling Unknowns

If the frequency is increased, computations on long paths may not have finished at the end of the clock period. In the example of Figure 1, the output z stabilizes shortly before t_{nom}. If an observation time in I_1 or I_2 is selected, the output still carries different intermediate values. In general, several smaller glitches can make the analysis very complex. To take this into account, in this work an X-value is assigned pessimistically, if the stable value is reached after the sampling time. As already pointed out, X-values will be clustered at the endpoints of long paths, and the distribution will vary with the test frequencies.

So far, only little research has been done on response compaction in the context of FAST. In [38] a special MUX-based compaction scheme has been proposed, which, however, requires a significant amount of control data during test application and discards a large fraction of response data by the multiplexers. Because of the specific requirements, standard X-handling schemes for BIST or embedded test cannot simply be re-used. Nevertheless, they can be the basis for handling unknowns in built-in FAST. Some typical examples for different X-handling strategies can be found in [26], [27], [28], [29], [30], [31], [32]. X-masking schemes mask out the X-values before they can enter the compactor [27], [30], [32] while X-filtering or X-canceling strategies can extract uncorrupted information after compaction [29], [31]. X-tolerant compaction schemes like X-compact or convolutional compactors can tolerate a certain amount of X-values without additional measures [26], [28], [39].

As test response compaction for FAST must be extremely flexible and adaptable to changing X-distributions, a programmable solution combined with post-processing has been chosen in this work. Among the known X-handling compaction schemes, the X-canceling MISR has been identified as the most suitable base scheme [31], [40]. The main idea of the X-canceling MISR is to analyze the MISR states by symbolic simulation and derive X-free information by linear combinations of MISR bits.

For a better understanding, this is briefly summarized for the small example of Figure 4. The X-bits represent unknown values, and the D-bits are deterministic bits necessary for detecting specific faults. The other bits in the test response do not contribute to fault detection. After shifting the first scan slice into the MISR, the MISR bits are $m_0 = X_0$, $m_1 = 0$, and $m_2 = 0$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2864255, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

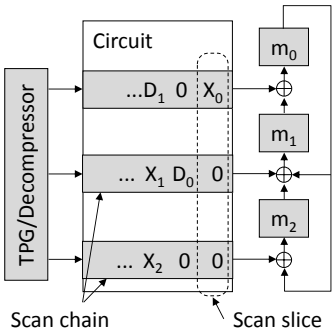

Fig. 4. Example for X-canceling MISR

With the next scan slice entering the MISR, the following equations are obtained for the state bits:

$$m_0 = 0,$$
$$m_1 = X_0 \oplus D_0,$$
$$m_2 = X_0.$$

Appropriate EXOR combinations of the MISR state bits provide:

$$m_0 = 0,$$
$$m_1 \oplus m_2 = X_0 \oplus D_0 \oplus X_0 = D_0,$$
$$m_2 = X_0.$$

This allows observing two X-free combinations of MISR bits, and in particular, the deterministic response bit D_0 can be observed as required. However, with three scan slices compacted in the MISR, the equations

$$m_0 = X_0 \oplus D_0 \oplus D_1,$$
$$m_1 = X_0 \oplus X_1,$$
$$m_2 = X_2$$

cannot be converted into a representation with X-free combinations of MISR bits. Thus, the intermediate signature obtained after the second scan slice must be analyzed, and the MISR must be reset. For the general case, this analysis can be efficiently implemented with the help of matrix representations and Gauss-Jordan elimination [31]. A deeper analysis of the *D*-bits can help to reduce the number of intermediate signatures to be stored [40]. If for example, the faults covered by D_0 are also covered by another *D*-bit in one of the following intermediate signatures, then there is no need to store the current signature. The problem of reducing the signature storage while maintaining the fault coverage is addressed in Section IV.

C. Architecture for Built-in FAST

This section gives an overview of the developed BIST scheme. The architecture shown in Figure 5 is compatible with a standard STUMPS architecture [36]. The test pattern generator and the MISR can be used both for a standard manufacturing or in-field test and for FAST. The hardware supports static and delay test. The test pattern generator can be configured as a pseudo-random pattern generator for LBIST and also as a decompressor for mixed-mode BIST or embedded test [41]. For delay test generation in launch-on-capture (LoC) or launch-on-shift (LoS) mode, any state of the art mixed-mode or deterministic pattern generator (TPG) can be used to provide the first pattern of a test pair (e.g. [37]). The second pattern is then obtained as the test response or as a single shift of the first pattern.

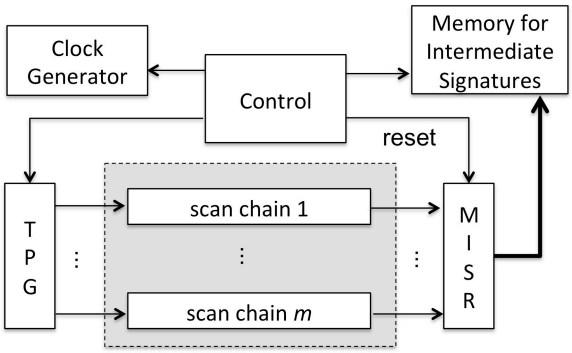

Fig. 5. BIST architecture for FAST.

Test response compaction is performed by a MISR, using X-canceling to extract fault information from possibly corrupted signatures [31]. As explained in Section II.B, the X-canceling MISR can only handle a limited number of X-values and must be reset once the limit is reached. To keep the scheme flexible, the X-canceling operations are not implemented in hardware as suggested in [31], but in case of a reset the intermediate signatures are stored in a small on-chip memory for later analysis. The technique does not impose any restrictions on the MISR, such that the same MISR can be used for standard test and FAST. As the experimental data in Section V.B show, shorter MISRs in general provide better trade-offs than longer MISRs.

The clock generator is the only specific add-on for FAST. It can be designed following the respective proposals in the literature [20], [22], [23], [24], [25]. For example, special clock chopping registers can be added to exploit an on-chip PLL for at-speed and even faster-than-at-speed test [20]. However, the test frequencies are then restricted to multiples of the PLL output frequency. In contrast to that, using phase interpolation to manipulate both rising and falling edges of the clock signal can achieve a desired frequency within a wide range of output frequencies, based on a single input signal [22]. Alternatively, delaying a reference clock signal by a programmable buffer line and combining the delayed signal with the reference clock supports a flexible generation of faster-than-at-speed clock pulses [23], [24]. The advantage of this scheme is that the frequency can be directly encoded into the test pattern. Furthermore, no time for switching the frequency is required, as both schemes do not rely on PLLs, which need to be locked to a frequency before they can be used. Further refinements even take into account the effects of process variations [25].

Besides standard tasks, the test control unit also interfaces the clock generator and takes care of resetting the MISR. It is programmable with the FAST parameters (test frequencies, reset times of the MISR), such that the hardware implementation of the BIST scheme remains independent of the optimization results achieved by the algorithms described in this paper.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2864255, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

For a specific circuit under test, the test time depends on the number of different frequencies and their associated patterns. The control data depend on the selected frequencies and the necessary resets of the MISR, which also determine the size of the memory for intermediate signatures. To ensure in-field testing with minimal resources, two problems have to be solved, namely optimal pattern and frequency selection and reduction of intermediate signatures. In practice, the size of the memory for intermediate signatures can also be decided upfront based on resource constraints. Again the number of intermediate signatures must be minimized to fit as much information as possible in the available memory and optimize the hidden delay fault coverage this way.

III. Pattern and Frequency Selection

This section describes the selection of frequencies and their associated test patterns in more detail. So far, most approaches for FAST rely on preselected equidistant frequencies between the nominal frequency f_{nom} and the maximum frequency f_{max}. Here, basic schemes sample each pattern at multiple frequencies [14], [17]. More sophisticated approaches select or generate patterns specifically for each frequency [16], [42], [43], [44]. However, as indicated by the examples in Section II.A, some faults can only be detected within a small range of observation times, which may not be covered by the preselected frequencies. To guarantee complete coverage, the frequencies must be specifically adapted to the detection ranges of the targeted faults. Of course, for an in-field test with limited resources, the number of different frequencies should be minimized.

A. Problem Statement and Complexity

Frequency selection for FAST can be formulated as the following optimization problem [34], [35].

Optimum Frequency Selection (OFS): Given a set Φ of hidden delay faults and their detection ranges $I(\varphi)$ for all $\varphi \in \Phi$. Find a minimum set of observation times $\mathcal{T} = \{t_1, ..., t_n\}$, such that for each $\varphi \in \Phi$ the intersection $I(\varphi) \cap \mathcal{T}$ is not empty.

The observation times t_i define the test frequencies $f_i = 1/t_i$. If two or more solutions of problem *OFS* exist, then the one with larger observation times should be selected to avoid noise and unnecessary X-values on long paths. Once a solution is selected, some faults may be detectable at several observation times depending on the applied test patterns. For such faults, the patterns detecting the faults at the largest possible observation times are selected for the same reason. More precisely, the set of hidden delay faults Φ is partitioned into groups $\Phi_i = \{\varphi \mid t_i$ is the largest observation time in $I(\varphi)\}$. During test, for each frequency f_i, only the patterns detecting the faults in Φ_i at time t_i have to be applied.

Before the developed algorithm for frequency selection is presented in Section III.B, the complexity of OFS is analyzed in the following. OFS is similar to the known NP-complete problem of finding a minimum hitting set[1] [45], [46].

[1] For simplicity we do not distinguish between optimization problems and their related decision problems.

Minimum Hitting Set (MHS): Let S be a finite set, and let $\mathcal{C}$ be a collection of subsets of S. Find a minimum subset H $\subset$ S, such that H $\cap$ C $\neq \emptyset$ for each C $\in \mathcal{C}$.

However, for MHS the subsets in $\mathcal{C}$ are discrete sets while for OFS the subsets to be hit consist of continuous intervals. Nevertheless, the problem MHS can be transformed into OFS in polynomial time, which proves that OFS is NP-complete.

Theorem: OFS is NP-complete.

Proof: (i) OFS is in NP, because for any given set of observation times, it can be checked in polynomial time whether it hits all detections ranges.

(ii) The problem MHS can be transformed into OFS as follows: Let $S = \{s_1, \ldots, s_m\}$ be a finite set and $\mathcal{C}$ a collection of subsets. For each $s_i \in$ S define an interval $I(s_i) := (i, i+1)$ and for each $C \in \mathcal{C}$ define a range $I(C)$ as

$$I(C) = \bigcup_{s_i \in C} I(s_i)$$

Solving OFS for $\Phi = \mathcal{C}$ and the detection ranges $I(C)$ provides a minimum set of observation times $\mathcal{T}$, such that for each $C \in \mathcal{C}$ the intersection $I(C) \cap \mathcal{T}$ is not empty. The set

$$H := \{s_i \in S : I(s_i) \cap \mathcal{T} \neq \emptyset\}$$

then provides a solution for MHS. ■

It should be noted that OFS is also similar to the one-dimensional geometric hitting set problem, which is solvable in polynomial time [47]. However, in this problem, S is a line and $\mathcal{C}$ is a collection of intervals while in OFS the elements of $\mathcal{C}$ are unions of intervals, which explains the gap in complexity.

B. An Algorithm for Optimum Frequency Selection

Having shown the NP-completeness of problem OFS, an exact solution in polynomial time cannot be expected. But OFS can be mapped to MHS in order to benefit from the intensive research on this problem. The mapping, however, should ensure that the solutions are robust against small timing variations and also against small clock variations. Consider Figure 6 for an illustration of this issue. The diagram in Figure 6 shows the detection ranges of seven small delay faults.

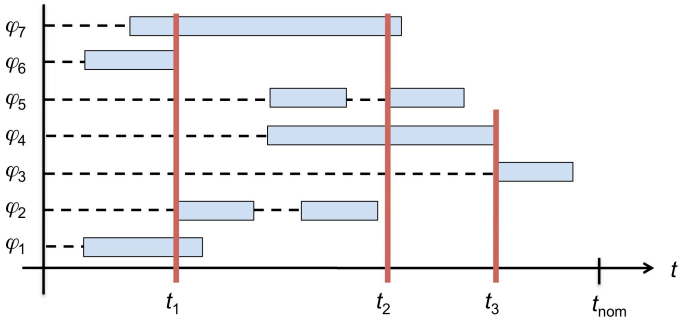

Fig. 6. Non-robust solution for OFS.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2864255, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

The selected observation times t_1, t_2, and t_3 hit all detection ranges, but for φ_2, φ_3, φ_5, and φ_6 only an endpoint of the detection range is hit. In case of small timing or clock variations, these faults may not be detected anymore. Although a fully variation-aware analysis is outside the scope of this work, the developed algorithms derive robust solutions in the sense of Figure 7, where the selected observation points hit inner points of all detection ranges.

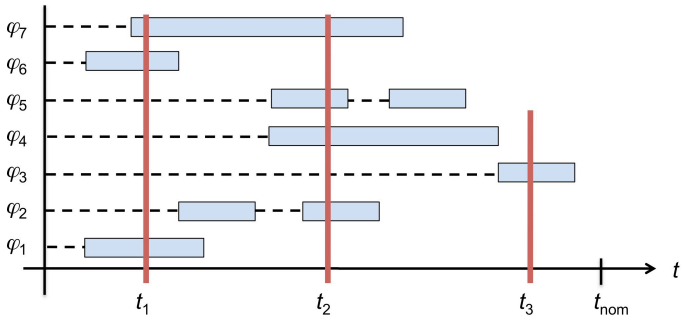

Fig. 7. Robust solution for OFS.

To guarantee robust solutions as in Figure 7, the intervals constituting the detection ranges are considered as open intervals. With this model, intersections of detection ranges cannot degenerate into a single point. This ensures a mapping of OFS to MHS, such that the observation times in the solution can be moved as far away from the interval borders as possible. The mapping is based on atomic intervals as defined in the following.

Definition 4: Let $\mathcal{I}$ be a set of detection ranges. An interval I is considered as an *atomic interval of* $\mathcal{I}$, if it can be obtained as an intersection of intervals in the detection ranges and it is minimal with this property, i.e. if there is an interval J in the detection ranges with $I \cap J \subset I$, then $I \cap J = I$.

As illustrated in Figure 8, the atomic intervals are obtained by intersecting the time axis with the start and end points of all intervals in all detection ranges.

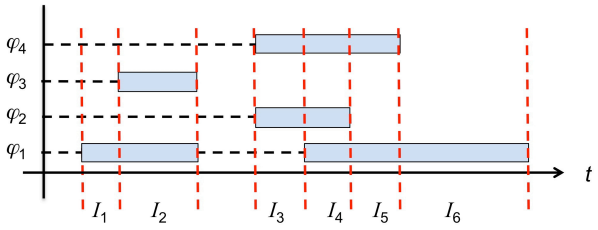

Fig. 8. Mapping detection ranges to atomic intervals.

Once the atomic intervals $I_1, \ldots, I_n$ have been computed for an instance of OFS, the corresponding instance of MHS is constructed starting with $S := \{I_1, \ldots, I_n\}$. For each fault φ a subset $C(\varphi)$ with all atomic intervals in the detection range $I(\varphi)$ is added to C. To extract the observation points for the original problem from a solution H of MHS, an arbitrary point can be selected from each atomic interval in H. Robustness is ensured by selecting the center points of the atomic intervals as observation times.

The mapping based on atomic intervals also allows for adjusting the frequencies, if the clock generator supports only a limited accuracy. In such a case, frequencies on the respective grid are selected in the atomic intervals. If an atomic interval is too small and does not contain a suitable frequency, it can be dropped from the detection range as illustrated in Figure 3 to ensure a solution supported by the clock generator.

In this work the hypergraph algorithm in [48] is applied to solve the minimum hitting set problem. This algorithm is based on a search tree and uses several intelligent reduction rules to achieve an overall runtime of $O(1.23801^{|S|})$ [48].

C. Hybrid Approach and Overall Workflow

Although the deployed hypergraph algorithm is very efficient for small and midsize circuits, the runtimes for larger circuits with many hidden delay faults may still grow beyond acceptable limits. Therefore, the problem size must be reduced as much as possible before optimum frequency selection is started. Similar as in mixed-mode BIST, where pseudo-random patterns are combined with deterministic patterns for the hard faults, a hybrid strategy for frequency selection is followed in this work. In the first phase, the test is performed with a set of predefined equidistant frequencies and standard transition test patterns. In the second phase, the set of remaining hard faults is further reduced by additional timing-aware test patterns before the problem OFS is solved. The detailed overall workflow is described in the following.

The flowchart of Figure 9 summarizes the procedure. Its inputs are a set of test patterns P_{init} for transition faults, the nominal test frequency f_{nom} with observation time t_{nom}, the maximum possible frequency f_{max} with observation time t_{min}, the targeted fault size δ, and a parameter k specifying the number of equidistant frequencies.

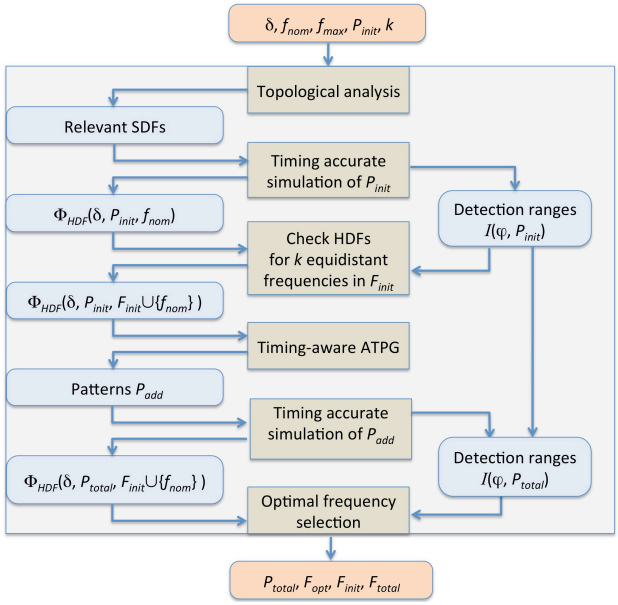

Fig. 9. Workflow for hybrid frequency selection.

The procedure starts with an initial fault set Φ containing all possible small delay faults of size δ. Through a quick topo-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2864255, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

logical analysis faults are removed from Φ based on the following two criteria: If the longest topological path is too short to detect a fault φ at f_{max}, then φ is undetectable for all available frequencies. If the shortest topological path is large enough to detect a fault φ at f_{nom}, then φ is either not detectable at all or f_{nom} is sufficient for fault detection. The remaining faults in Φ are the *relevant* SDFs to be processed further.

In the next step accurate timing simulation is performed for all relevant small delay faults to determine the set of hidden delay faults $\Phi_{HDF}(\delta, P_{init}, f_{nom})$ as well as their detection ranges. This exhaustive fault simulation is computationally very expensive and requires a high throughput simulator, which fully exploits data and structural parallelism inherent in patterns, gates, and faults. Furthermore, for accurate fault propagation, individual rising and falling pin-to-pin delays as well as glitch filtering at gates and fault activation and propagation by glitches along reconvergent signals should be supported. In this work the simulator described in [49] is used, which maps the simulation tasks to a graphics processor (GPU).

Then the set F_{init} of k equidistant frequencies for the first phase is determined by constructing the respective set of observation times $T_{init} := \{t_0, ..., t_{k-1}\}$ by $t_0 = t_{min}$, and $t_i = t_{min} + i \cdot (t_{nom} - t_{min})/k$ for all $i < k$. The faults in $\Phi_{HDF}(\delta, P_{\text{init}}, f_{\text{nom}})$, which are not detectable at any of the observation times in T_{init} constitute the set $\Phi_{HDF}(\delta, P_{init}, F_{init} \cup \{f_{nom}\})$ of hard-to-detect hidden delay faults. As these faults have only been addressed by transition test patterns so far, some of them may be detectable over longer propagation paths. To reduce the need for additional frequencies, additional test patterns are therefore generated in the next step. Since the set of hard-to-detect faults is typically much smaller than the initial fault set Φ, timing-aware ATPG is now computationally feasible [9], [10], [11], [12], [13]. The presented workflow relies on the timing-aware option of commercial ATPG tool. As an alternative, n-detect ATPG may be used, because it also tries to propagate faults over several different paths [50].

This yields an additional set of test patterns P_{add}, which can detect some of the hard faults at the nominal test frequency f_{nom} or at the initial equidistant frequencies F_{init}. The remaining faults are hidden delay faults with respect to $P_{total} := P_{init} \cup P_{add}$ and $F_{init} \cup \{f_{nom}\}$ constituting the set $\Phi_{HDF}(\delta, P_{init}, F_{init} \cup \{f_{nom}\})$. Timing accurate simulation is performed again to exactly determine $\Phi_{HDF}(\delta, P_{init}, F_{init} \cup \{f_{nom}\})$ and update the detection ranges. Finally, optimum frequency selection as described in Section III.B is started. The solution provides a set of optimal frequencies F_{opt}, as well as the set $F_{total} = F_{init} \cup F_{opt}$.

IV. Test Response Compaction

As explained in Section III.C the BIST architecture for FAST relies on X-canceling and stores intermediate signatures in a memory for later processing. The size of the memory depends on the number of intermediate signatures to be stored, and thus on the number and distribution of X-values, as well as on the number and distribution of D-bits. While the X-values can be limited to a certain extent by a proper selection of frequencies, the impact of D-bits is in the focus of this section.

For a given set Φ of hidden delay faults detectable at a given test frequency f, let $\mathcal{D}(\Phi, f)$ denote the set of deterministic response bits. Each bit $D \in \mathcal{D}(\Phi, f)$ detects a subset of hidden delay faults $\Phi(D)$. A first approach to minimize the required information for fault detection is solving the following set covering problem.

Problem D-Bit Cover: Given a set Φ of hidden delay faults detectable at a given test frequency f and the associated set of deterministic response bits $\mathcal{D}(\Phi, f)$. Find a subset $\mathcal{D} \subset \mathcal{D}(\Phi, f)$ of response bits, such that $\Phi = \cup_{D \in \mathcal{D}} \Phi(D)$.

Consider for example $\Phi = \{\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5, \varphi_6\}$, $\mathcal{D}(\Phi, f) = \{D_0, D_1, D_2, D_3, D_4, D_5\}$, $\Phi(D_0) = \{\varphi_1, \varphi_2, \varphi_5\}$, $\Phi(D_1) = \{\varphi_1, \varphi_4, \varphi_5, \varphi_6\}$, $\Phi(D_2) = \{\varphi_3, \varphi_4, \varphi_6\}$, $\Phi(D_3) = \{\varphi_2, \varphi_3\}$, $\Phi(D_4) = \{\varphi_3, \varphi_4\}$, $\Phi(D_5) = \{\varphi_6\}$. Then both $\mathcal{D} = \{D_1, D_3\}$ and $\mathcal{D}^* = \{D_0, D_2\}$ are valid solutions for the set covering problem.

However, the problem D-Bit Cover only reflects the situation before the test responses enter the MISR. To check whether the selection of D-bits actually reduces the required number of intermediate signatures, a symbolic analysis of the MISR state sequence is necessary. Figure 10 sketches the situation for the same example as before.

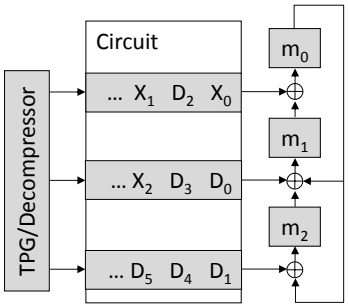

Fig. 10. Example for D-bit selection.

Combining the first scan slice with the MISR state provides $m_0 = X_0$, $m_1 = D_0$, and $m_2 = D_1$. With the next scan slice entering the MISR the state bits are:

$$m_0 = D_0 \oplus D_2,$$
$$m_1 = X_0 \oplus D_1 \oplus D_3,$$
$$m_2 = X_0 \oplus D_4.$$

After X-canceling row operations this results in

$$m_0 = D_0 \oplus D_2,$$
$$m_1 \oplus m_2 = D_1 \oplus D_3 \oplus D_4,$$
$$m_2 = X_0 \oplus D_4.$$

At this stage both $\mathcal{D} = \{D_1, D_3\}$ and $\mathcal{D}^* = \{D_0, D_2\}$ would ensure the observation of all faults in Φ with a minimum number of D-bits. Yet, executing one more compaction step yields

$$m_0 = X_0 \oplus X_1 \oplus D_1 \oplus D_3,$$
$$m_1 = X_0 \oplus X_2 \oplus D_0 \oplus D_2 \oplus D_4,$$
$$m_2 = D_0 \oplus D_2 \oplus D_5.$$

Now only $\mathcal{D}^* = \{D_0, D_2\}$ still ensures fault detection. Selecting $\mathcal{D} = \{D_1, D_3\}$ would require to store the second MISR-state as an intermediate signature. Thus only $\mathcal{D}^* = \{D_0, D_2\}$ minimizes the signature storage. The problem D-Bit Cover must be extended accordingly.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2864255, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

Problem Extended D-Bit Cover: Given a set of Φ of hidden delay faults detectable at a given test frequency f and the associated set of deterministic response bits $\mathcal{D}(\Phi, f)$. Find a subset $\mathcal{D} \subset \mathcal{D}(\Phi, f)$ of response bits, such that $\Phi = \cup_{D \in \mathcal{D}} \Phi(D)$ and the number of intermediate signatures is minimal.

To avoid complex symbolic simulation for all considered candidate solutions, in this work only the simpler problem D-Bit Cover is solved as a first approximation of Extended D-Bit Cover. Even the solution of D-Bit Cover is very complex, since the underlying set covering problem is known to be NP-complete [45, 46]. Standard solutions for the set covering problem work with covering tables and exploit row and column dominance to reduce the problem size before actually building the solution. As the problem instances for D-Bit Cover can grow very large, building and storing complete covering tables would require too much memory. Therefore, a greedy heuristic has been implemented, which is interleaved with fault simulation. The heuristic starts with an empty set $\mathcal{D}$ and simulates all test patterns. Whenever a new fault $\varphi \in \Phi$ is detected at an additional output bit D not yet contained in $\mathcal{D}$, then $\mathcal{D}$ is updated to $\mathcal{D} \cup \{D\}$. Then the necessary resets are determined by symbolic simulation of the MISR. The respective time steps provide the control data for the BIST scheme described in Section II.C.

It should be noted that aliasing is possible, if the fault sets $\Phi(D)$ for the deterministic bits are not disjoint. Then a fault effect visible at an even number of deterministic bits could be canceled out, if all bits appear in the observed combination of MISR bits.

V. Experimental Results

To evaluate the presented concepts and algorithms, an experimental study was conducted for the full scan versions of the ITC'99 [51] and some industrial benchmark circuits. The circuit characteristics are summarized in Table I.

TABLE I. Circuit Characteristics

Circuit	*# Gates*	*#PI + PPI*	*# PO + PPO*	t_{nom} *[ps]*	*# SDF*	*# P_{init}*	*FC_{TF}*
b14_1	12438	260	214	4171	66984	1344	66.05%
b15_1	6533	572	418	8826	37526	640	98.03%
b17_1	12858	1827	1348	3588	123880	1024	99.38%
b18_1	75618	4116	3085	4533	423216	2048	92.88%
b20_1	25547	533	450	4326	137774	1920	79.47%
b21_1	25561	534	450	4333	137646	1856	78.95%
b22_1	38568	786	664	4497	207448	2240	84.00%
p45k	22414	3739	2550	3191	127344	5568	99.96%
p78k	46504	3148	3484	1511	269024	128	99.99%
p81k	78665	4029	3952	1604	434998	640	99.92%
p89k	56662	4627	4557	2240	314776	1600	99.82%
p100k	53836	5902	5829	3040	301848	5312	99.81%
p141k	105347	11290	10502	2655	577266	1664	99.76%

Columns 2 to 4 show the number of gates, the number of primary and pseudo-primary inputs, as well as the number of primary and pseudo-primary outputs. Subsequently, the nominal clock period t_{nom} obtained with the SYNOPSYS SAED 90 nm library is reported in the 5th column. The number of all possible fault locations for small delay faults is listed in column 6, and column 7 reports the number of patterns in the initial pattern set P_{init}. Here a transition fault (TF) test set without any specific optimization was generated using a commercial ATPG tool. The transition fault coverage FC_{TF} reported by the tool is shown in the last column. The values for FC_{TF} range from 66.05 % to 99.99 % and can be viewed as an estimate of the maximum achievable coverage of small delay faults. Therefore all further experiments were restricted to fault locations with detected transition faults.

Both the hybrid workflow for pattern and frequency selection from Section III and the D-bit Selection from Section IV were analyzed using decreasing fault sizes $\delta = 18\sigma$, $\delta = 12\sigma$, and $\delta = 6\sigma$, where σ denotes the standard deviation of the nominal gate delay.

A. Pattern and Frequency Selection

Table II summarizes the major results obtained with the hybrid workflow from section III.C for the three different fault sizes $\delta = 18\sigma$, $\delta = 12\sigma$, and $\delta = 6\sigma$. In all experiments the minimum observation time was set to $t_{min} = 0.3 \cdot t_{nom}$, and $k = 6$ equidistant observation times were selected in the interval $[t_{min}, t_{nom})$ to obtain the initial set of frequencies, i.e. $F_{init} = \{1/t_0, \ldots, 1/t_5\}$ with $t_0 = 0.3 \cdot t_{nom}$, $t_1 = 0.42 \cdot t_{nom}$, $t_2 = 0.53 \cdot t_{nom}$, $t_3 = 0.65 \cdot t_{nom}$, $t_4 = 0.77 \cdot t_{nom}$, and $t_5 = 0.88 \cdot t_{nom}$. Depending on the circuit, the minimum observation time t_{min} thus ranges from around 450 ps to 2640 ps, which corresponds to frequencies in the range of several hundred MHz up to around 2 GHz.

TABLE II. Final Coverage of Hidden Delay Faults

Circuit	$\delta = 18\sigma$		$\delta = 12\sigma$		$\delta = 6\sigma$	
	# HDF	*FC_{HDF}*	*# HDF*	*FC_{HDF}*	*# HDF*	*FC_{HDF}*
b14_1	37122	83.42%	37297	81.91%	37503	79.94%
b15_1	18585	55.98%	18611	56.30%	18543	55.50%
b17_1	67582	82.07%	62769	81.56%	58998	81.12%
b18_1	231602	84.46%	223499	83.02%	214908	80.78%
b20_1	90883	86.69%	91437	85.01%	92008	83.52%
b21_1	90359	86.94%	91152	85.53%	92028	83.91%
b22_1	150184	87.02%	150536	85.77%	150365	84.01%
p45k	64656	84.74%	55517	76.82%	42495	84.08%
p78k	224573	95.37%	219058	95.41%	208870	95.94%
p81k	357911	93.06%	339236	92.40%	315086	91.85%
p89k	181099	81.16%	161448	79.83%	148393	78.77%
p100k	138855	83.75%	124718	84.50%	113198	85.07%
p141k	306029	88.32%	291403	87.80%	276565	86.61%

As explained above, only fault locations with detected transition faults were considered for further analysis. For each fault size, Table II shows the number of hidden delay faults identified by the workflow as well as the percentage FC_{HDF} of hidden delay faults that can finally be detected with the enlarged test set P_{total} and all frequencies F_{total}. Please note that hidden delay faults are only considered at locations with detectable transition faults, and that the coverage measured by

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2864255, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

FC_{HDF} is relative to the number of hidden delay faults but not to the total number of fault locations.

The results in Table II demonstrate that the hybrid workflow of section III.C ensures a very high coverage of hidden delay faults, which decreases only slightly with the fault size. The small portion of undetected hidden delay faults would require frequencies higher than f_{max} or additional test patterns. At this point it is also important to note that FAST is applied on top of a standard delay test, such that even the smaller hidden delay fault coverage for circuit b15_1 still contributes to a considerable increase in product quality.

The necessary effort, however, increases with shrinking fault sizes as indicated by Table III. For each fault size the number of additional test frequencies determined by optimal frequency selection as well as the overall test time are recorded. The test time is measured as the actual number of test patterns used during test. Please note that this number is not necessarily equal to the number of patterns in P_{total}, as during pattern and frequency selection some patterns may be used at several frequencies for different hidden delay faults while others may not detect any hidden delay faults and are discarded therefore.

TABLE III. NUMBER OF FREQUENCIES AND TEST TIME

Circuit	δ = 18σ		δ = 12σ		δ = 6σ	
	# Freq.	*Test time*	*# Freq.*	*Test time*	*# Freq.*	*Test time*
b14_1	42	1673	48	1944	53	2213
b15_1	14	395	16	457	24	579
b17_1	22	1761	26	1872	34	2343
b18_1	55	5105	62	5773	79	7228
b20_1	48	3450	53	4017	68	5142
b21_1	46	3515	54	4065	67	5246
b22_1	62	5520	65	6540	86	8393
p45k	11	1707	12	1468	16	1092
p78k	8	284	8	435	12	756
p81k	29	3912	29	4382	33	6131
p89k	34	5620	36	6051	39	7980
p100k	16	2106	22	2217	33	2767
p141k	27	5569	32	6775	40	9371

As expected, for smaller fault sizes more frequencies and longer test times are needed to maintain a comparable detection level. Finally, Figures 11 through 13 illustrate the ramp up of the hidden delay fault coverage in the three phases of the workflow. In each figure, the blue bars represent the hidden delay fault coverage $FC_{HDF,1}$ achievable with the initial test set P_{init} and the 6 equidistant frequencies in F_{init}. The red bars show the additional hidden delay fault coverage $FC_{HDF,2}$ obtained by adding timing-aware patterns P_{add} for the hard faults, and the green bars correspond to the additional hidden delay fault coverage $FC_{HDF,3}$ after optimal frequency selection. In all figures, it can be observed, that both the additional patterns and the additional frequencies actually ramp up the hidden delay fault coverage. Interestingly, the more difficult the detection of hidden delay faults gets with decreasing fault sizes, the higher is the gain achieved in the second and third step of the workflow.

The results in Figures 11 through 13 also show that a more naïve approach for FAST based on a fixed number of equidistant frequencies and a simple transition fault set (initial set up) can already detect a considerable amount of hidden delay faults. However, some faults remain undetectable and require specifically selected frequencies.

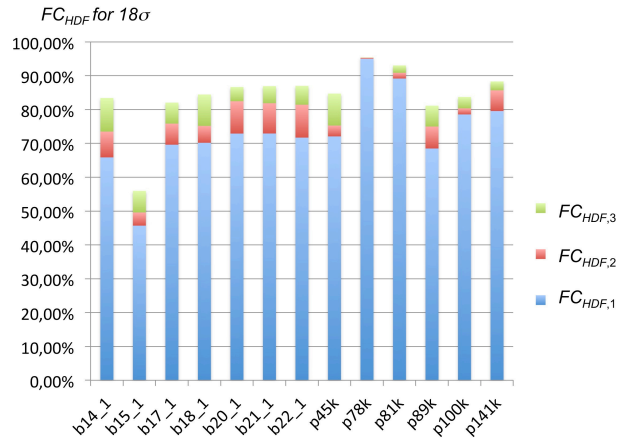

Fig. 11. Evolution of hidden delay fault coverage for 18σ.

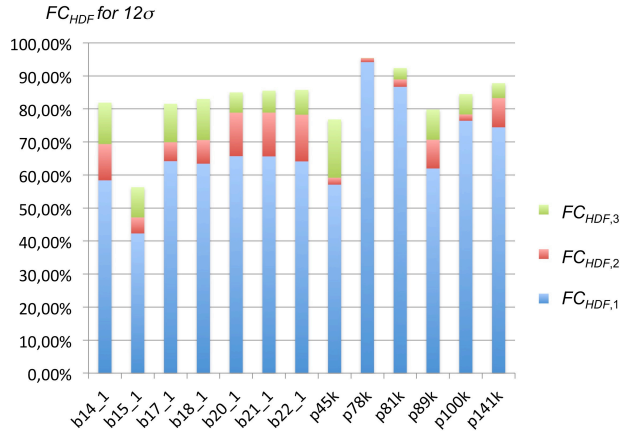

Fig. 12. Evolution of hidden delay fault coverage for 12σ.

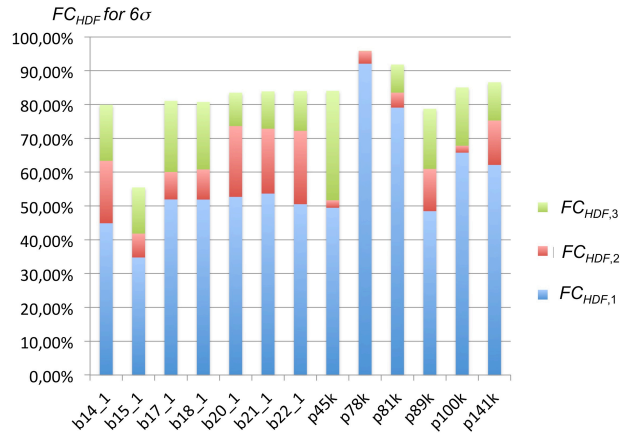

Fig. 13. Evolution of hidden delay fault coverage for 6σ.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2864255, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

In practice, the number of additional frequencies for maximum coverage may not always be acceptable. It might therefore be an option to trade-off hidden delay fault coverage against the overhead for additional frequencies. This is illustrated in Figure 14 for circuit b22_1 and a fault size of 6 σ. The curve shows the evolution of fault efficiency, which is measured as the hidden delay fault coverage normalized to the final coverage reported in Table II, for 5 of the 6 initial and 86 additional frequencies. The sixth initial frequency is omitted, because it covers only faults also detected by the additional frequencies. A fault efficiency of 95 % can be reached already with 25 frequencies. The high effort for the remaining 5 % shows that some hidden delay faults in the circuit can only be detected with individually adjusted frequencies.

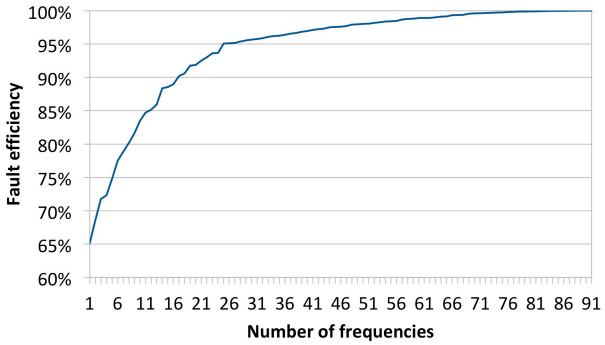

Fig. 14. Hidden delay fault coverage versus additional frequencies.

B. D-Bit Selection

To evaluate the approach for D-bit selection presented in Section IV, the selected test frequencies and patterns reported in the previous subsection were used. The observed X-rates confirm the importance of a flexible X-handling strategy. Figure 15 illustrates the evolution of X-rates during FAST for one example circuit. The curve shows the average number of X-values per scan slice as a function of the observation time. The observation times are normalized to the nominal observation time and range from 88 % down to 30 %, which corresponds to t_{min}. It can be clearly seen that the X-rates considerably increase with decreasing observation times. Similar trends can be observed for all other circuits.

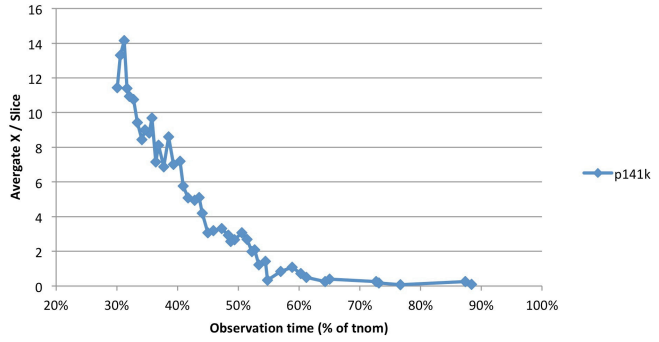

Fig. 15. X-rates per scan slice as a function of the observation time for circuit p141k.

The hidden delay fault coverage after compaction was determined by full symbolic simulation of all faults and all D-bits. As the results for the three different fault sizes δ = 18σ, δ = 12σ, and δ = 6σ are very similar, only the results for the most difficult case δ = 6σ are summarized in Table IV.

TABLE IV. D-BIT SELECTION

Circuit	*# D-Bits*	*MISR*	*# Selected D-Bits*	*# Sign.*	*Memory [kB]*	*FE*
b14_1	9645	16	51.57%	3317	6.48	98.82%
b15_1	5137	16	35.22%	1221	2.38	97.43%
b17_1	19878	32	40.14%	5470	21.37	99.40%
b18_1	89272	64	27.77%	12708	99.28	99.89%
b20_1	30177	64	42.29%	6241	48.76	99.85%
b21_1	31418	64	41.69%	6402	50.02	99.90%
b22_1	53401	64	39.77%	11659	91.09	100.00%
p45k	175253	64	3.30%	2440	19.06	99.92%
p78k	123714	64	18.38%	13116	102.47	99.96%
p81k	200081	64	19.05%	39093	305.41	99.81%
p89k	147235	64	13.36%	15069	117.73	99.34%
p100k	165992	128	6.69%	4091	63.92	99.99%
p141k	394313	256	10.10%	23797	743.66	100.00%

The second column in Table IV lists the number of D-bits carrying hidden delay fault information, and the third column indicates the MISR size, which corresponds to the number of scan chains in this experiment. The remaining columns report the results obtained using a greedy algorithm for the problem D-Bit Cover. The percentage of selected D-bits is listed in column 4, before the number of signatures to be stored, the resulting memory requirements in kB and the fault efficiency FE are presented in columns 5 to 7. Here, fault efficiency is defined as the following ratio

$$FE = \frac{\#\ HDFs\ observable\ after\ X\text{-}canceling}{\#\ HDFs\ observable\ in\ the\ uncompacted\ test\ response}.$$

Analyzing the results shows that in all cases a relatively small percentage of D-bits is sufficient to ensure a very high or even complete fault efficiency with feasible storage requirements. To complete the evaluation, the experiments were rerun for a fixed MISR size of 512. The obtained trade-offs are illustrated in Figures 16 and 17. As expected the fault efficiency slightly increases for the larger MISR. However, the overall storage requirements also increase considerably.

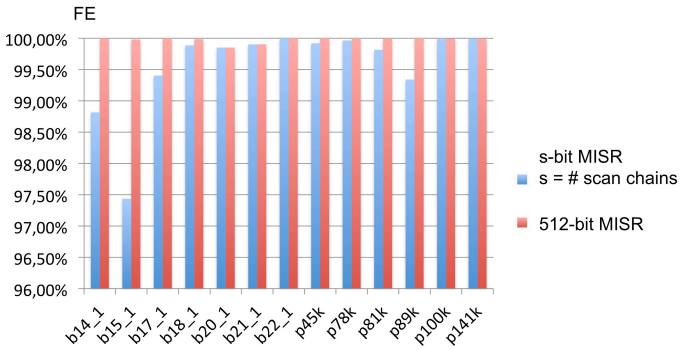

Fig. 16. Impact of the MISR size on fault efficiency.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCAD.2018.2864255, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems

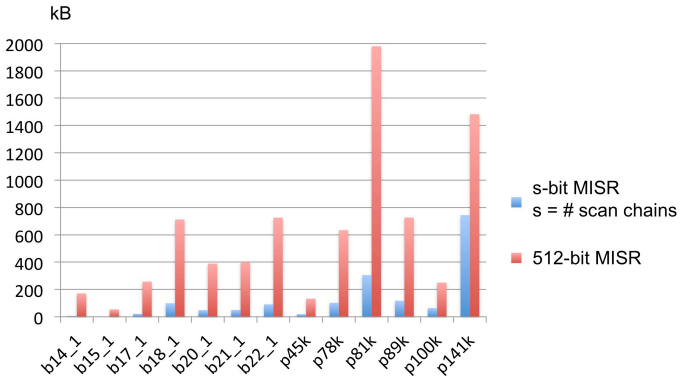

Fig. 17. Impact of the MISR size on signature storage.

It is interesting to note that for almost all circuits a smaller MISR size clearly provides a better trade-off between fault efficiency and signature storage, as the number of intermediate signatures is comparable for both small and large MISR sizes.

C. Runtimes

The algorithms described above have been executed on an Intel Xeon 5 processor with 12 cores and 128 GB RAM. For the GPU-based simulator, a GeForce 980 TI has been used. As the runtimes for the different fault sizes are very similar, only the breakdown for the smallest fault size $\delta = 6\sigma$ is shown in Table V.

TABLE V. RUNTIMES FOR 6σ IN SECONDS

CUT	*Simulation*		*Selection algorithms*			*Total*
	First	*Second*	*Freq.*	*Pattern*	*D-Bit*	
b14_1	511	446	1	44	10	1012
b15_1	524	239	1	13	2	780
b17_1	381	326	2	149	19	877
b18_1	9228	6636	11	738	251	16864
b20_1	1823	1790	3	189	45	3850
b21_1	1899	1933	3	210	44	4089
b22_1	3303	4422	5	362	106	8198
p45k	2413	283	1	241	17	2955
p78k	5812	191	4	112	102	6221
p81k	10647	8547	13	1498	387	21092
p89k	4687	20194	4	570	244	25701
p100k	11355	735	3	786	147	13026
p141k	22476	33688	12	2560	1385	60121

The second column shows the runtime for the exhaustive fault simulation using the initial pattern set P_{init}. The runtimes in the third column correspond to the second simulation run, which is performed for the hard faults that cannot be detected by P_{init} and F_{init}. For the second simulation run the timing-aware patterns in P_{add} are used. This can lead to high simulation times for some circuits despite the reduced sizes of the underlying fault sets. The fourth column shows the runtime for the hypergraph-based frequency selection, followed in column five by the runtime for selecting the respective patterns for each frequency.Finally, columns six and seven show the runtimes for the selection of D-bits and calculation of the intermediate signatures. The last column shows the overall runtime, which ranges from approximately 13 minutes for b15_1 to around 17 hours for the largest circuit p141k. For all circuits, the overall runtime is clearly dominated by the fault simulation times listed in columns 2 and 3.

VI. CONCLUSIONS AND FUTURE WORK

Small delay faults can be indicators of marginal hardware and should be monitored throughout the life cycle of a system. Hidden delay faults are particularly challenging, because they can only be propagated along short paths and require faster-than-at-speed test frequencies. The BIST approach in this paper enables periodic in-field testing for hidden delay faults, and thus a proactive detection of potential reliability problems before they cause actual failures. Nevertheless, there is a problem of false alarms, when small delay faults are detected which are actually caused by uncritical delay variations. To quantify and solve this problem a more comprehensive approach on variations is required, which is in the focus of ongoing research. Future research will also target more complex clock schemes and multi-cycle testing.

VII. ACKNOWLEDGEMENTS

Parts of this work have been supported by the German Research Foundation (DFG) under grants WU 245/19-1 and HE1686/4-1, FAST.

VIII. REFERENCES

[1] V. Malandruccolo, et al., "Design and Experimental Characterization of a New Built-In Defect-Based Testing Technique to Achieve Zero Defects in the Automotive Environment", IEEE Transactions on Device and Materials Reliability, Vol. 11, No. 2, June 2011, pp. 349-357.

[2] Y. Sato et al., "DART: Dependable VLSI test architecture and its implementation," Proceedings IEEE International Test Conference (ITC'12), Anaheim, CA, USA, 2012, pp. 1-10.

[3] U. Abelein et al., "Non-intrusive integration of advanced diagnosis features in automotive E/E-architectures," Proceedings Design, Automation & Test in Europe Conference & Exhibition (DATE'14), Dresden, Germany, 2014, pp. 1-6.

[4] J. A. Abraham et al., "Special session 8B — Panel: In-field testing of SoC devices: Which solutions by which players?," Proceedings IEEE VLSI Test Symposium (VTS'14), Napa, CA, USA, 2014, pp. 1-2.

[5] C. Eychenne and Y. Zorian, "From manufacturing to functional safety use, how infield built-in self-test architecture must evolve to support real time system constraints," 1st IEEE International Workshop on Automotive Reliability and Test (ART'16), Forth Worth, TX, USA, Nov. 2017.

[6] T. McLaurin, "Online LBIST for Automotive on a Multicore Processor," 1st IEEE International Workshop on Automotive Reliability and Test (ART'16), Forth Worth, TX, USA, Nov. 2017

[7] Y. M. Kim, et al., "Low-cost gate-oxide early-life failure detection in robust systems," Proceedings IEEE Symposium on VLSI Circuits (VLSIC'10), Honolulu, HI, USA, June 2010, pp.125-126.

[8] M. Tehranipoor, K. Peng, K. Chakrabarty, "Test and Diagnosis of Small-Delay Defects," Springer, 2012, ISBN 978-1-4419-8296-4

[9] X. Lin, et al., "Timing-Aware ATPG for High Quality At-speed Testing of Small Delay Defects," Proceedings 15th Asian Test Symposium (ATS'06), Fukuoka, Japan, Nov. 2006, pp.139-146.

[10] M. Yilmaz, K. Chakrabarty, and M. Tehranipoor, "Test-pattern selection for screening small-delay defects in very-deep submicron integrated circuits," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 29, No. 5, May 2010, pp. 760-773.

[11] V. Zolotov, et al., "Statistical Path Selection for At-Speed Test," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 29, No. 5, May 2010, pp. 749-759.

[12] S. Eggersglüss, R. Drechsler, "As-Robust-As-Possible Test Generation in the Presence of Small Delay Defects using Pseudo-Boolean Optimization," Proceedings Design, Automation and Test in Europe (DATE'11), Grenoble, France, April 2011, pp. 1291-1296.

[13] M. Sauer, et al., "Early-life-failure detection using SAT-based ATPG," Proceedings IEEE International Test Conference (ITC'13), Anaheim, CA, USA, Sep. 2013, pp. 1-10.

[14] H. Yan, A. D. Singh; "Experiments at Detecting Delay Faults using Multiple Higher Frequency Clocks and Results from Neighboring Die", Proceedings IEEE International Test Conference (ITC'03), Charlotte, NC, USA, Sep. – Oct. 2003, pp. 105-111.

[15] M. Amodeo and B. Cory, "Defining faster-than-at-speed delay tests," Cadence Nanometer Test Quarterly eNewsletter, Vol. 2, No. 2, May 2005.

[16] N. Ahmed, M. Tehranipoor, and V. Jayaram, "A Novel Framework for Faster-than-at-Speed Delay Test Considering IR-drop Effects," Proceedings IEEE/ACM International Conference on Computer Aided Design (ICCAD'06), San Jose, CA, USA, Nov. 2006, pp. 198–203.

[17] J. Lee and E. J. McCluskey, "Failing Frequency Signature Analysis," Proceedings IEEE International Test Conference (ITC'08), Santa Clara, CA, USA, Oct. 2008, pp. 1–8.

[18] R. Turakhia, W. Daasch, M. Ward, and J. Van Slyke, "Silicon evaluation of longest path avoidance testing for small delay defects," Proceedings IEEE International Test Conference (ITC'07), Santa Clara, CA, USA, Oct. 2007, pp. 1–10.

[19] S. Chakravarty, N. Devta-Prasanna, A. Gunda, J. Ma, F. Yang, H. Guo, R. Lai, and D. Li, "Silicon evaluation of faster than at-speed transition delay tests," Proceedings IEEE VLSI Test Symposium (VTS'12), Hyatt Maui, HI, USA, Apr. 2012, pp. 80-85.

[20] T. L. McLaurin and F. Frederick, "The Testability Features of the MCF5407 Containing the 4th Generation ColdFire® Microprocessor Core," Proceedings IEEE International Test Conference (ITC'00), Atlantic City, NJ, USA, Oct. 2000, pp. 151-159.

[21] R. Press and J. Boyer, "Easily Implement PLL Clock Switching for At-Speed Test," Chip Design Magazine, Feb./March 2006.

[22] S. Kaeriyama, M. Kajita and M. Mizuno, "A 1-to-2GHz 4-Phase On-Chip Clock Generator with Timing-Margin Test Capability," Digest of Technical Papers of the IEEE International 2007 Solid-State Circuits Conference (ISSCC), San Francisco, USA, 2007, pp. 174-175 ctd. on 594.

[23] R. Tayade and J. A. Abraham, "On-chip Programmable Capture for Accurate Path Delay Test and Characterization," Proceedings IEEE International Test Conference, Santa Clara, CA, USA, Oct. 2008, pp. 1-10.

[24] S. Pei, H. Li, and X. Li, "An On-Chip Clock Generation Scheme for Faster-than-at-Speed Delay Testing," Proceedings Design and Test in Europe (DATE'10), Dresden, Germany, March 2010, pp. 1353-1356.

[25] S. Pei, Y. Geng, H. Li, J. Liu und S. Jin, "Enhanced LCCG: A Novel Test Clock Generation Scheme for Faster-than-at-Speed Delay Testing," Proceedings 20th Asia and Sout Pacific Design Automation Conference (ASP-DAC), Chiba, Japan, 2015, pp. 514-519.

[26] S. Mitra and K. S. Kim, "X-Compact: An efficient response compaction technique for test cost reduction," Proceedings IEEE International Test Conference (ITC'02), Baltimore, MD, USA, Oct. 2002, pp. 311-320.

[27] M. Naruse, et al., "On-chip compression of output responses with unknown values using LFSR reseeding," Proceedings IEEE International Test Conference (ITC'03), Charlotte, NC, USA, Sep. – Oct. 2003, pp. 1060-1068.

[28] J. Rajski, et al., "Convolutional compaction of test responses," Proceedings IEEE International Test Conference (ITC'03), Charlotte, NC, USA, Sep. – Oct. 2003, pp. 745-754.

[29] M. Sharma and W.-T. Cheng, "X-Filter: Filtering Unknowns from Compacted Test Responses," Proceedings IEEE International Test Conference (ITC'05), Austin, TX, USA, Nov. 2005, pp. 1-9.

[30] Y. Tang et al., "X-Masking During Logic BIST and its Impact on Defect Coverage," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 14, No. 2, Feb. 2006, pp. 193-202

[31] N. A. Touba, "X-canceling MISR - An X-tolerant Methodology for Compacting Output Responses with Unknowns Using a MISR," Proceedings IEEE International Test Conference (ITC'07), Santa Clara, CA, USA, Oct. 2007, pp. 1-10.

[32] P. Wohl, J. A. Waicukauski, and S. Ramnath, "Fully X-Tolerant Combinational Scan Compression," Proceedings IEEE International Test Conference (ITC'07), Santa Clara, CA, USA, Oct. 2007, pp. 1-10.

[33] D. Czysz, et al., "On compaction utilizing inter and intra-correlation of unknown states," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 29, No. 1, Jan. 2010, pp. 117-126.

[34] S. Hellebrand, et al., "FAST-BIST: Faster-than-At-Speed BIST Targeting Hidden Delay Defects," Proceedings IEEE International Test Conference (ITC'14), Seattle, WA, USA, October 2014, pp. 1-8.

[35] M. Kampmann et al., "Optimized Selection of Frequencies for Faster-than-at-Speed Test," Proceedings Asian Test Symposium (ATS'15), Mumbai, India, November 2015, pp. 109-114.

[36] P. H. Bardell and W. H. McAnney, "Self-Testing of Multichip Logic Modules," Proceedings IEEE International Test Conference (ITC'82), Philadelphia, PA, USA, Nov. 1982, pp. 200-204.

[37] A. W. Hakmi, S. Holst, H. J. Wunderlich, J. Schlöffel, F. Hapke and A. Glowatz, "Restrict Encoding for Mixed-Mode BIST," Proceedings IEEE VLSI Test Symposium (VTS'09), Santa Cruz, CA, 2009, pp. 179-184.

[38] A. Singh, C. Han, X. Qian, "An Output Compression Scheme for Handling X-States from Over-Clocked Delay Test," Proceedings 28th IEEE VLSI Test Symposium, Santa Cruz, CA, USA, April 2010, pp. 57-62.

[39] S. Mitra, S. S. Lumetta, M. Mitzenmacher, "X-tolerant signature analysis," Proceedings IEEE International Test Conference (ITC'04), Charlotte, NC, USA, Oct. 2004, pp. 432-441.

[40] R. Garg, R. Putman, and N. A. Touba, "Increasing Output Compaction in Presence of Unknowns using an X-Canceling MISR with Deterministic Observation," Proceedings IEEE VLSI Test Symposium (VTS'08), San Diego, CA, USA, April – May 2008, pp. 35-42.

[41] J. Rajski, et al., "Embedded Deterministic Test," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 23, No. 5, 2004, pp. 776-792.

[42] X. Fu, H. Li, and X. Li, "Testable path selection and grouping for faster than at-speed testing," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 20, No. 2, 2012, pp. 236-247.

[43] B. Kruseman, et al., "On hazard-free patterns for fine-delay fault testing," Proceedings IEEE International Test Conference (ITC'04), Charlotte, NC, USA, Oct. 2004, pp. 213–222.

[44] T. Yoneda, et al., "Faster- Than-At-Speed Test for Increased Test Quality and In-Field Reliability," Proceedings IEEE International Test Conference (ITC'11), Anaheim, CA, USA, Sep. 2011, pp. 1-9.

[45] R. M. Karp, "Reducibility Among Combinatorial Problems," in R. E. Miller and J. W. Thatcher (editors), "Complexity of Computer Computations," New York: Plenum, 1972, pp. 85-103.

[46] M. R. Garey, D. S. Johnson, "Computers and Intractability – A Guide to the Theory of NP-completeness," W. H. Freeman & Co Ltd., New York, NY, USA, 1979

[47] P. K. Agarwal, E. Ezra, and M. Shair, "Near-linear approximation algorithms for geometric hitting sets," Proceedings 25th Annual Symposium on Computational Geometry (SCG '09), Aarhus, Denmark, June 2009, pp. 23-32.

[48] L. Shi and X. Cai, "An Exact Fast Algorithm for Minimum Hitting Set," Proceedings Third IEEE International Joint Conference on Computational Science and Optimization., May 2010, pp. 64–67.

[49] E. Schneider, et al., "GPU-Accelerated Small Delay Fault Simulation," Proceedings Design, Automation and Test in Europe (DATE'15), Grenoble, France, March 2015, pp. 1174–1179.

[50] I. Pomeranz and S. M. Reddy, "On n-Detection Test Sets and Variable n-Detection Test Sets for Transition Faults," IEEE Transactions on

Computer-Aided Design of Integrated Circuits and Systems, Vol. 19, No. 3, March 2000, pp. 372-383.

[51] F. Corno, M. Sonza Reorda and G. Squillero, "RT-Level ITC'99 Benchmarks and First ATPG Results," IEEE Design & Test of Computers, Vol. 17, No. 3, pp. 44-53, Jul. 2000. Benchmarks available at https://github.com/squillero/itc99-poli, last visited on 2018-06-12