Intermittent and Transient Fault Diagnosis on Sparse Code Signatures

Kochte, Michael; Dalirsani, Atefe; Bernabei, Andrea; Omana, Martin; Metra, Cecilia; Wunderlich, Hans-Joachim

Proceedings of the 24th IEEE Asian Test Symposium (ATS'15) Mumbai, India, 22-25 November 2015

doi: http://dx.doi.org/10.1109/ATS.2015.34

Abstract: Failure diagnosis of field returns typically requires high quality test stimuli and assumes that tests can be repeated. For intermittent faults with fault activation conditions depending on the physical environment, the repetition of tests cannot ensure that the behavior in the field is also observed during diagnosis, causing field returns diagnosed as no-trouble-found. In safety critical applications, self-checking circuits, which provide concurrent error detection, are frequently used. To diagnose intermittent and transient faulty behavior in such circuits, we use the stored encoded circuit outputs in case of a failure (called signatures) for later analysis in diagnosis. For the first time, a diagnosis algorithm is presented that is capable of performing the classification of intermittent or transient faults using only the very limited amount of functional stimuli and signatures observed during operation and stored on chip. The experimental results demonstrate that even with these harsh limitations it is possible to distinguish intermittent from transient faulty behavior. This is essential to determine whether a circuit in which failures have been observed should be subject to later physical failure analysis, since intermittent faulty behavior has been diagnosed. In case of transient faulty behavior, it may still be operated reliably.

Intermittent and Transient Fault Diagnosis on Sparse Code Signatures

M. A. Kochte[1], A. Dalirsani[1], A. Bernabei[2], M. Omana[2], C. Metra[2], H.-J. Wunderlich[1]
[1] Institut für Technische Informatik, Universität Stuttgart, 70569 Stuttgart, Germany
[2] University of Bologna, Bologna 40136, Italy

Abstract—**Failure diagnosis of field returns typically requires high quality test stimuli and assumes that tests can be repeated. For intermittent faults with fault activation conditions depending on the physical environment, the repetition of tests cannot ensure that the behavior in the field is also observed during diagnosis, causing field returns diagnosed as no-trouble-found.**

In safety critical applications, self-checking circuits, which provide concurrent error detection, are frequently used. To diagnose intermittent and transient faulty behavior in such circuits, we use the stored encoded circuit outputs in case of a failure (called signatures) for later analysis in diagnosis. For the first time, a diagnosis algorithm is presented that is capable of performing the classification of intermittent or transient faults using only the very limited amount of functional stimuli and signatures observed during operation and stored on chip.

The experimental results demonstrate that even with these harsh limitations it is possible to distinguish intermittent from transient faulty behavior. This is essential to determine whether a circuit in which failures have been observed should be subject to later physical failure analysis, since intermittent faulty behavior has been diagnosed. In case of transient faulty behavior, it may still be operated reliably.

Index Terms—**Diagnosis, intermittent, transient, concurrent error detection, code signature, self-checking, online testing.**

I. INTRODUCTION

In safety critical applications, such as medical, industrial, or automotive applications, failures in the field must be analyzed and diagnosed to identify potentially systematic root causes and avoid risks during future operation. Once a failure has been observed, the circuit can be analyzed in a workshop or during scheduled maintenance intervals.

In the automotive industry, suspected electronic control units (ECUs) are typically replaced for safety reasons and sent back to the OEM (original equipment manufacturer) even if the execution of functional tests does not reveal any failures in the workshop. However, returned units are often classified as *no-trouble-found (NTF)* or *no defect found*, i.e. the reported failure could not be reproduced or diagnosed at the OEM site [1], [2]. According to [3], the NTF rate of field returns in VLSI circuits can exceed 50%, which incurs high costs both for workshops and OEMs for replacements and failure analysis.

If the failure in the circuit has been caused by a single or multiple transient effects, the unit is not faulty, and there is no need for a replacement. If, on the other hand, the failure has been caused by an intermittent or permanent fault, the unit must be replaced. Intermittent faults are active depending on some (often unknown) activation condition. This activation condition may have temporal, Boolean, or other environmental dependencies. Intermittent faults may emerge due to latent defects or marginal hardware, circuit aging, adverse operation conditions (temperature, voltage, electromagnetic radiation, vibrations, etc.), or combinations thereof. Resulting errors can occur in bursts [4].

If an intermittent fault is the root cause of a failure in the field, even a thorough test in the workshop or at the OEM may not excite the faulty behavior because of complex fault activation and different operation conditions. For transient faults, the faulty behavior can also not be reproduced. However, without an evidence that the observed failures have been transient, the unit must be replaced to prevent security risks. By conventional failure analysis based on (re-)execution of tests followed by diagnosis, permanent faults can be effectively identified. However, it is in principle difficult to diagnose and distinguish intermittent from multiple transient faults.

To detect errors at runtime, self-checking circuits, which provide concurrent error detection, can be used. The employed concurrent error detection methods include structural redundancy (duplication with comparison), information redundancy by use of codes [5]–[7], watchdog circuitry [8], or synthesized assertions [9]. Here we consider self-checking circuits using separable codes, where data and check bits can be easily split apart. The generated check bits provide some information about failures at the time of their occurrence and can be captured for later analysis. In the following, these check bits are called *signatures* and stored in a signature log (SILO, Fig. 1). After a certain number of observed failures or in regular intervals, the data in the SILO is extracted and diagnosed offline to distinguish transient from intermittent root causes.

However, such a diagnosis is very difficult due to

1) extremely limited amount of failure data (low number of

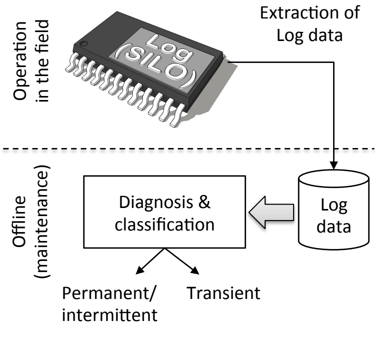

Fig. 1. Signature-based diagnosis flow

stimuli and signatures, signatures are highly compacted responses),
2) low quality of stimuli (*not* deterministically generated test patterns or patterns with high diagnostic resolution, but random or functional input stimuli), and
3) non-repetitiveness of the test.

Conventional diagnosis algorithms focus on the location of a fault and often assume a permanent nature of the fault. The model-independent diagnosis in [10] classifies the observed fault effects as conditional stuck-at faults that are active depending on an (unknown) arbitrary activation condition. Yet the algorithm is unable to classify transient and intermittent effects correctly. Algorithms tailored to highly compacted test responses [11] also exist, but they cannot distinguish transient or intermittent faults that happen *during operation*.

Diagnosis algorithms specifically targeting intermittent faults build a probabilistic model of the system and refine fault classification by repeated application of tests and measurement of responses [12]–[14]. The diagnosis algorithm in [15] splits up BIST into sessions, repeats failed sessions and analyzes intermediate test signatures. It uses Bayesian reasoning to classify ambiguous fault effects. Such approaches are not suitable for the problem at hand, since the repeated execution of tests with operation conditions encountered in the field is in general very difficult.

At system level, errors in memory can be statistically analyzed at runtime to identify similarities in the observed behavior and to conclude if a component suffers from transient or permanent impairments [16]. However, if very few failures are observed, the result of such reasoning has only little confidence.

This work presents how failure data available in a self-checking structure can be used to diagnose intermittent and transient faults. Code signatures along with the corresponding input stimuli are stored when failures are detected. A diagnosis method is presented for offline evaluation of stored data to distinguish intermittent and transient faults.

The next section gives an overview of the proposed method, followed by a brief discussion of the used terminology. Section IV presents the diagnosis algorithm, and Section V discusses the experimental results.

II. OVERVIEW

A self-checking circuit is capable of detecting errors concurrent to the functional operation. Typically, a checker circuit checks whether an encoded output is a codeword or not. If it is not a codeword, an error indication is given.

In the considered architecture (cf. Fig. 2), we consider self-checking circuits based on separable codes. The separable check bits are called signature in the following. The circuit is extended with a storage, called signature log (SILO). Upon error detection, both the input stimulus and the code signatures are stored. This requires that the primary and pseudo-primary inputs are equipped with shadow registers to allow extraction of the stimuli. The memory of the SILO can be (persistent) system memory or chip-internal or external FLASH, for instance. Available design-for-debug structures such as trace buffers can be reused. A small controller is responsible to store the stimuli and signature data. In principle, the input data can also be compressed using a low-overhead compression algorithm if a CPU is available.

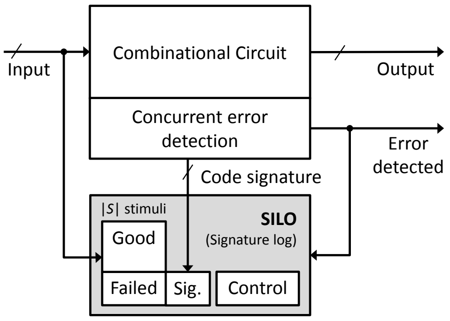

Fig. 2. Overview of the signature log (SILO) architecture

Apart from failing input stimuli and corresponding signatures, it is also important to store a set of input stimuli that did not trigger an error. This information helps during diagnosis to narrow down possible fault locations and fault activation patterns. The selection of non-failing input stimuli can be deterministic (e.g. at regular intervals, or a number of stimuli after a detected error) or random. However, storing consecutive input stimuli may impose additional overhead for at-speed high bandwidth data storage. Here, we assume that non-failing stimuli are selected at random intervals.

During a maintenance session, or if a certain number of errors have been detected, the data in the SILO is extracted for offline analysis. A diagnosis algorithm then analyzes the stored stimuli and signatures to distinguish intermittent from transient behavior. If the SILO contains only a single failing entry, a distinction between a transient and an intermittent faulty behavior is in principle not possible.

Storing the environmental and internal operation conditions of the system, like temperature, VDD, or the system state at the moment of failure, may further improve the diagnostic resolution. But this is beyond the scope of this paper.

III. TERMINOLOGY AND PROBLEM DESCRIPTION

The goal is to distinguish intermittent and transient faults in the circuit under diagnosis (CUD) using only the information stored in the SILO. Intermittent faults shall not be falsely classified as transient faults to avoid overlooking systematic reliability threats in the system.

Let S be the set of input stimuli stored in the SILO and $|S|$ be the number of stored stimuli. Furthermore, $F \subseteq S$ is the set of failed stimuli, i.e. stimuli that cause an error indication in the self-checking circuit. For the failed stimuli, the code signatures are also stored. For the remaining stimuli $S \setminus F$, the code signature is error-free and can be obtained by logic simulation.

The task is to determine whether the observed failures have been caused by an intermittent fault or multiple transient faults using only the $|S|$ stimuli and $|F|$ failed signatures stored in the SILO. Since different input stimuli lead to different signatures, a simple comparison of stored signatures is not meaningful to distinguish intermittent from transient faults.

In the following, we do not further investigate the classification of permanent faults since they can be easily detected and diagnosed by a structural test or repeated application of the stored failing simuli.

IV. DIAGNOSIS ALGORITHM FOR SIGNATURE CLASSIFICATION

The fault model independent diagnosis algorithm "Pointer" of [10] is adopted here. Pointer performs an effect-cause analysis of the input stimuli and responses. It uses the conditional stuck-at line calculus to characterize the faulty behavior of the circuit under diagnosis (CUD) by considering both the topology and the defect behavior. A conditional stuck-at line fault is a stuck-at fault at a location f with an arbitrary activation condition. This activation condition may for instance be of Boolean or temporal nature and allows to model intermittent or transient fault activation.

For each fault location f in the fault machine (circuit model) and for each stored stimulus $s \in S$, an evidence $e(f,s) := (\Delta\sigma_s, \Delta\iota_s, \Delta\tau_s, \Delta\gamma_s)$ is computed by fault simulation and response comparison between the fault machine under f and the CUD. $\Delta\sigma_s$ denotes the number of outputs that fail both in the CUD and the fault machine (cf. Fig. 3). $\Delta\iota_s$ and $\Delta\tau_s$ are the number of outputs mispredicted by the fault machine. $\Delta\iota_s$ is the number of outputs that fail in the fault machine, but are correct in the CUD. $\Delta\tau_s$ is the number of outputs that fail in the CUD, but are correct in the fault machine. $\Delta\gamma_s$ is the minimum of $\Delta\iota_s$ and $\Delta\sigma_s$. If $\Delta\gamma_s = 0$, the *conditional* stuck-at fault at f partially explains the faulty behavior of the CUD: f is active and the failing outputs in the fault machine are a subset of the CUD ($\Delta\iota_s = 0$), or f is not active ($\Delta\sigma_s = 0$). If $\Delta\gamma_s > 0$, the conditional fault at f is a less suitable suspect.

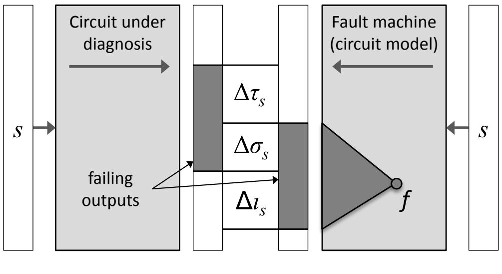

Fig. 3. Response analysis in Pointer diagnosis (adopted from [10])

The evidences are then summed up over the set of stimuli yielding $e(f,S) := (\sigma_S, \iota_S, \tau_S, \gamma_S)$, with $\sigma_S := \sum_{s \in S} \Delta\sigma_s$ (analogous for $\iota_S, \tau_S, \gamma_S$). In this way, the evidences capture the information provided by both failing and non-failing stimuli in S for diagnosis.

The fault locations are then ranked by their evidences, firstly in increasing order of γ_S, then in decreasing order of σ_S and finally in increasing order of ι_S [10]. The most likely candidate is called the *top suspect* $\hat{f}$ with evidence $e(\hat{f}, S)$. If the top suspect is indeed the root cause, σ_S will be maximum among all suspects and $\iota_S = \tau_S = \gamma_S = 0$.

Obviously, a single line suspect cannot explain defects affecting multiple lines. However, the evidence of such a suspect still allows a *classification of the faulty behavior* using the values of ι_S and τ_S [10]. If $\tau_S > 0$, a single (conditional) stuck-at fault in the fault machine cannot explain the faulty behavior. If $\iota_S > 0$, then the fault activation depends on a (unknown) condition, which may be temporal, Boolean, or related to the environment. Using the evidence, Pointer classifies the top suspect as single (permanent) stuck-at, single conditional stuck-at, multiple stuck-at, or multiple conditional stuck-at fault [10].

Yet this is insufficient for a distinction of transient and intermittent behavior caused by multiple transient effects or intermittent faults that affect multiple lines in the circuit. The evidence of a fault location f is thus extended by ϕ, the number of failing stimuli, i.e. stimuli for which $\sigma_s > 0$:

$$\phi := \big|\{s \in S | \sigma_s > 0\}\big|.$$

ϕ can be used to reason about the temporal nature of the observed faulty behavior since higher values of ϕ indicate frequent observation of failures at one fault location. For transient events, ϕ of the respective affected fault locations equals 1 with high probability. This is because it is unlikely that different transient faults affect the same location multiple times (typically only one failing stimulus per transient). In case of intermittent behavior that has been detected multiple times, ϕ will typically have values greater than 1. If there are multiple transient faults with overlapping output cones (some common failing signature bits), ϕ of the top suspect can be greater than one. This makes the distinction to intermittent faults impossible. In the worst case, a circuit that suffered from transient faults is replaced without need. As shown by the experiments below, this happens in 12.7% of the investigated scenarios. In most cases, however, ϕ of the top suspect is a good classifier to distinguish between transient and intermittent effects.

V. EVALUATION

The proposed diagnosis scheme is evaluated in experiments with fault injection of permanent, intermittent, and single and multiple transient faults (cf. Sec. V-A). In the evaluation, the achievable diagnostic resolution by use of the signature log SILO is assessed. The experimental flow is presented in Sec. V-B, the results are discussed in Sec. V-C.

A. Considered Fault Models

The following fault models are evaluated in the experiments:

Permanent faults: A Byzantine bridge fault between two signal lines (k, l) is injected into the circuit [17]. Depending on the input pattern, this fault may alter the values of line l, line k, or of both lines in the circuit.

The Byzantine bridge fault model has been included in the evaluation since it causes fault effects originating at two different lines in the circuit. Since it may be easily confused with transient effects at different lines, this complicates the diagnosis task.

Intermittent faults: A Byzantine bridge fault is injected into the circuit, but it is considered active only for a fraction act $(0 < act < 1)$ of detecting input stimuli. For the remaining stimuli that could detect the fault, the fault will not generate erroneous output values. These stimuli are called *detecting but non-failing stimuli*. The lower the values of act for intermittent faults, the more difficult is their distinction from transient faults.

As second type of intermittent fault, one intermittent stuck-at fault is injected into the circuit. The stuck-at fault is considered active only for the fraction act of the detecting input stimuli.

Transient faults: Different numbers of single event transients are injected into the circuit. Depending on the technology and transferred particle energy, a single or multiple signals in the circuit can be affected. The time period until the effect of a transient has been mitigated, may vary as well. Here we assume that a single line is affected for at most one clock period and inject a stuck-at fault for one clock cycle. The fault effect may still propagate to multiple circuit outputs.

B. Evaluation Flow

The fault injection experiments are conducted for the discussed permanent, intermittent ($act = 0.5$), and transient faults. The faults are selected randomly from the fault universe.

In the experiments, random pattern fault simulation is used to select the failing stimuli. $|F|$ failing patterns and signatures are stored in the Silo (cf. Sec. III). For an intermittent fault, the parameter act defines the fraction of detecting stimuli for which the fault generates erroneous output values.

If the Silo does not contain *detecting but non-failing stimuli*, the fault appears as permanently active. To reflect the intermittent nature of the fault (and increase the difficulty for the diagnosis), $|F| \cdot \frac{1-act}{act}$ *detecting but non-failing stimuli* are stored in the Silo.

Each injection, pattern selection, and diagnosis is repeated $N = 20$ times, and the results are averaged. In the experiments, $|S|$ is set to 20, and $|F|$ is set to 5. For transient faults, the Silo contains 1, 3, or 5 failing patterns depending on the number of injected transients. These numbers are chosen as an example. Experiments with a much larger number of non-failing stimuli ($|S| = 100$) have been conducted as well. The diagnosis quality did not significantly improve. If $|F| << 5$, the distinction between intermittent and transient faults is more difficult since the available data becomes insufficient. Experiments with $|F| = 3$ are briefly discussed at the end of the section.

In the experiments, a double error detecting (DED) Hamming code, a triple error detecting (TED) Hsiao code, and a cyclic code are investigated for self-checking circuits. The original circuit (plain) is also included to assess the diagnostic quality when responses are not encoded/compacted. This corresponds to the achievable diagnostic quality when duplication with comparison or triple modular redundancy is used and failing responses are stored.

C. Results

Table I shows the characteristics of the used ISCAS benchmark circuits and industrial circuits (*p*k*) kindly provided by NXP. The third and fourth columns give the number of primary and pseudo-primary inputs and outputs. The last three columns give the number of signature (check) bits depending on the used error detection code.[1]

TABLE I. Circuit characteristics

Circuit	Gate count	Inputs	Outputs (Plain)	Check bits for code		
				DED	TED	Cyclic
s5378	3223	214	228	8	9	12
s9234	5944	247	250	9	10	14
s13207	8668	700	790	10	11	20
s15850	10211	611	684	10	11	20
s35932	16353	1763	2048	12	13	22
s38417	23537	1664	1742	11	12	22
s38584	21462	1464	1730	11	12	22
p45k	38811	3739	2550	12	13	24
p77k	65015	3487	3400	12	13	24
p78k	68263	3148	3484	12	13	20
p81k	106450	4029	3952	12	13	24
p89k	80963	4632	4557	13	14	21
p100k	84356	5902	5829	13	14	23
p141k	152808	11290	10502	14	15	28

1) Size of the Silo*:* The required size of the Silo in bits can be computed from the number of inputs (I), the number of signature size or check bits (C), and $|S|$ and $|F|$ as follows:

$$\text{size}_{\text{SILO}} = |S| \cdot I + |F| \cdot C,$$

assuming that the bit-wise difference between predicted and generated check bits [6] is stored. For the largest circuit and the parameters discussed above, 272 Kibits are required if the uncompacted responses are stored, and 221 Kibits if the compacted signatures are employed.

2) Discussion of circuit p100k: Figure 4 shows the results of the diagnosis for circuit p100k and the different codes. The upper bar chart shows the diagnosis classification of the top suspect for 1, 3, and 5 transient faults as single stuck-at, single conditional stuck-at, multiple stuck-at, or multiple conditional stuck-at fault. A transient fault may be classified as *un*conditional single or multiple stuck-at if none of the non-failing stimuli in the Silo would detect a transient fault at that fault site. Thus, conditional activation cannot be concluded *in principle* from the data available in the Silo.

The black data points in the chart give the value of ϕ for the top suspect, averaged over the 20 iterations. If only one transient is injected, $\phi = 1$. For multiple transients, values greater than 1 are observed for the DED, TED, and cyclic code. This happens if there exists a transient at a fault location

---

[1] The check bits for a cyclic code are set to at most twice the number of DED bits and may be smaller if a generator is not found for that number.

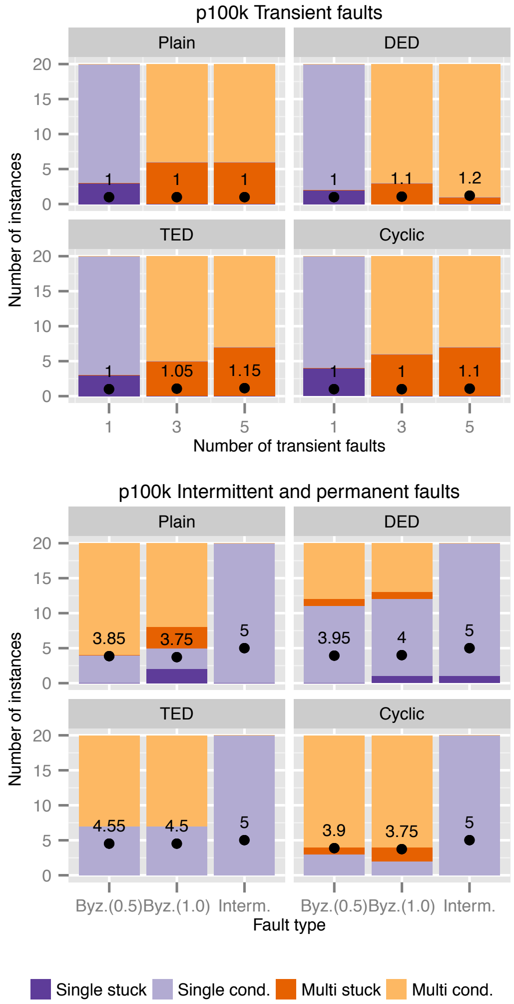

Fig. 4. Diagnosis result for circuit p100k (black points: ϕ averaged over 20 iterations)

f in the fault machine that can be detected by more than one of the failing stimuli F in the SILO and whose fault effect also propagates to the failing outputs in the stored erroneous signatures. The low values of ϕ, however, show that this case is infrequent. $\phi > 1$ is observed mainly for the DED, TED, and cyclic codes, which introduce global reconvergences at the output parity trees and thus, make it more likely that different fault locations affect the same failing outputs in the erroneous signatures. In the worst case with 5 injected transient faults and the DED code, in four out of 20 iterations $\phi > 1$.

The lower chart shows the diagnosis result for Byzantine bridge faults with $act = 0.5$ and $act = 1.0$ (permanently activated), as well as an intermittent stuck-at fault with $act = 0.5$. Since the bridge faults can alter the values of two lines in the circuit, the evidence of the top suspect may be classified as multiple stuck-at or multiple conditional stuck-at faults, depending on the stored stimuli in the SILO. Also, one of the two, or both of the affected lines may cause errors propagating to the outputs. Depending on which line is activated more often, ϕ may vary.

The values of ϕ are given as black data points again. For the intermittent stuck-at fault and 5 stored failing stimuli, the diagnosis identifies a top suspect that is detected exactly 5 times. For the bridge faults, ϕ may also equal 1. In that case, the fault effects could not be classified as intermittent but are considered transient because of insufficient failure data. This happens in the worst case four times in the 20 iterations (cyclic code, $act = 1.0$). On average over the different injected Byzantine and intermittent stuck-at faults, such a wrong diagnosis is observed for only 5.42% of the iterations. If the circuit goes into operation again and additional failure data is collected, the diagnosis can be repeated with the union of all failure data extracted so far, which improves the diagnostic result.

In summary, the identification of the top suspect using the diagnosis algorithm on the stored stimuli and signatures and using the value of ϕ to reason about the temporal fault activation is robust for most of the injected faults.

3) Results for all circuits: Table II shows the values of ϕ of the top suspects for all investigated circuits, also averaged over the 20 iterations. The results of the injection of a single transient fault is omitted since $\phi = 1$ for all circuits and codes. The last four rows of the table give the averaged values of ϕ over all the circuits.

For the plain circuits (no compacted signatures), ϕ is very close to 1 for multiple transient events, with a maximum of $\phi = 2$. For the DED, TED, and cyclic codes, the maximum value of ϕ is 4 (5 injected transient faults). Over all circuits, in 12.7% of the experiments, $\phi > 1$. In such a case, the faulty behavior is falsely classified as intermittent, and a defect-free unit would be replaced to avoid safety risks. On average over all circuits, the values of ϕ are still below 1.5.

For the intermittent stuck-at fault, ϕ equals 5. Thus, ϕ is a robust and safe means to classify the intermittent nature of this type of fault. For the Byzantine bridge faults, the values of ϕ can be as low as 1, which happens for 4.55% of the iterations. For these faults, the failure data was insuffcient to conclude the intermittency of the faulty behavior.

The runtime of the diagnosis algorithm for one circuit is at most 34.9s and has only low memory requirements. It can be easily conducted on a workstation in a workshop.

4) Results for $|F| = 3$: The diagnosis experiments are repeated assuming that only three failing signatures are stored in the SILO. This makes the distinction of intermittent and transient faults even more difficult. On average, for intermittent Byzantine bridges, ϕ ranges between 2.36 and 2.63 for the different codes, and for permanent Byzantine bridges between 2.45 and 2.56. Even with this limit on available failure data, it is possible to correctly identify intermittent faulty behavior for the majority of fault injections.

VI. CONCLUSION

The failure analysis of field returns incurs high costs. Often, circuits are classified as no-trouble-found. Here, we propose a diagnosis method for self-checking circuits. When failures are observed, the erroneous code signature and corresponding input stimulus are stored in a signature log on chip. The data in the signature log is later extracted for offline analysis. A diagnosis algorithm is presented that allows to classify the temporary nature of the observed failures. Experimental results using fault injection demonstrate that the vast majority of faults are correctly classified. This classification and the stored information can help to pinpoint the fault location and activation conditions for intermittent faults and to avoid needless replacements of units in case of transient faults.

ACKNOWLEDGMENT

This work was partially supported by the German Research Foundation (DFG) under grant WU 245/13-1 (RM-BIST).

REFERENCES

[1] D. A. Thomas, K. Ayers, and M. Pecht, "The trouble not identified phenomenon in automotive electronics," *Microelectronics Reliability*, vol. 42, pp. 641–651, 2002.

[2] H. Qi, S. Ganesan, and M. Pecht, "No-fault-found and intermittent failures in electronic products," *Microelectronics Reliability*, vol. 48, pp. 663–674, 2008.

[3] S. Davidson, "Understanding NTF components from the field," in *Proc. IEEE International Test Conference (ITC)*, 2005, pp. 1–10, paper 14.1.

[4] C. Constantinescu, "Trends and challenges in VLSI circuit reliability," *IEEE Micro*, vol. 23, no. 4, pp. 14–19, July 2003.

[5] D. A. Anderson and G. Metze, "Design of totally self-checking check circuits for m-out-of-n codes," *IEEE Trans. Computers*, vol. 22, no. 3, pp. 263–269, Mar. 1973.

[6] N. Touba and E. McCluskey, "Logic synthesis of multilevel circuits with concurrent error detection," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 16, no. 7, pp. 783–789, Jul 1997.

[7] A. Dalirsani, M. A. Kochte, and H.-J. Wunderlich, "SAT-based Code Synthesis for Fault-Secure Circuits," in *Proc. IEEE Symp. Defect and Fault Tolerance in VLSI and Nanotech. Systems (DFT)*, 2013, pp. 38–44.

[8] A. Mahmood and E. McCluskey, "Concurrent error detection using watchdog processors—a survey," *IEEE Trans. Computers*, vol. 37, no. 2, pp. 160–174, Feb 1988.

[9] R. Vemu, A. Jas *et al.*, "A low-cost concurrent error detection technique for processor control logic," in *Proc. Design, Automation and Test in Europe (DATE)*, March 2008, pp. 897–902.

[10] S. Holst and H.-J. Wunderlich, "Adaptive debug and diagnosis without fault dictionaries," *Journal of Electronic Testing: Theory and Application (JETTA)*, vol. 25, no. 4-5, pp. 259–268, Aug. 2009.

[11] S. Holst and H. Wunderlich, "A diagnosis algorithm for extreme space compaction," in *Proc. Design, Automation Test in Europe Conference (DATE)*, April 2009, pp. 1355–1360.

[12] S. Kamal, "An approach to the diagnosis of intermittent faults," *IEEE Trans. Computers*, vol. C-24, no. 5, pp. 461–467, May 1975.

[13] I. Koren and Z. Kohavi, "Diagnosis of intermittent faults in combinational networks," *IEEE Trans. Computers*, vol. 26, no. 11, pp. 1154–1158, 1977.

[14] J. De Kleer, "Diagnosing multiple persistent and intermittent faults," in *Proc. International Joint Conference on Artifical Intelligence*, 2009, pp. 733–738.

[15] L. Rodríguez Gómez, A. Cook *et al.*, "Adaptive Bayesian Diagnosis of Intermittent Faults," *Journal of Electronic Testing: Theory and Application (JETTA)*, vol. 30, no. 5, pp. 527–540, 2014.

[16] R. K. Iyer, L. T. Young, and P. V. K. Iyer, "Automatic recognition of intermittent failures: An experimental study of field data," *IEEE Trans. Computers*, vol. 39, no. 4, pp. 525–537, Apr. 1990.

[17] M. Renovell, P. Huc, and Y. Bertrand, "CMOS bridging fault modeling," in *Proc. IEEE VLSI Test Symposium (VTS)*, Apr 1994, pp. 392–397.

TABLE II. VALUES OF ϕ OF THE TOP SUSPECT FOR $|S| = 20$

Circuit	Code	Transient		Interm. act=0.5		Perm.
		3	5	SAF	Byz.Br.	Byz.Br.
s5378	Plain	1.20	1.30	5.00	3.85	3.85
	DED	1.15	1.70	5.00	3.70	3.55
	TED	1.20	1.50	5.00	4.35	4.45
	Cyclic	1.40	2.05	5.00	4.70	4.60
s9234	Plain	1.00	1.05	5.00	4.25	4.10
	DED	1.25	1.45	5.00	4.40	4.20
	TED	1.30	1.40	5.00	4.50	4.40
	Cyclic	1.15	1.30	5.00	4.10	4.10
s13207	Plain	1.00	1.05	5.00	4.10	4.15
	DED	1.25	1.50	5.00	3.95	4.05
	TED	1.15	1.25	5.00	4.50	4.35
	Cyclic	1.05	1.40	5.00	4.85	4.75
s15850	Plain	1.00	1.00	5.00	4.05	3.75
	DED	1.05	1.25	5.00	4.45	4.40
	TED	1.20	1.40	5.00	3.75	4.20
	Cyclic	1.20	1.40	5.00	4.45	4.25
s35932	Plain	1.00	1.00	5.00	3.80	3.85
	DED	1.50	1.50	5.00	4.15	3.75
	TED	1.10	1.25	5.00	3.50	3.80
	Cyclic	1.00	1.15	5.00	3.90	3.70
s38417	Plain	1.05	1.05	5.00	3.55	3.60
	DED	1.30	1.75	5.00	3.75	3.85
	TED	1.30	1.30	5.00	3.35	3.45
	Cyclic	1.05	1.00	5.00	3.80	4.00
s38584	Plain	1.00	1.00	5.00	3.75	3.55
	DED	1.10	1.60	5.00	3.45	3.70
	TED	1.00	1.40	5.00	4.15	4.00
	Cyclic	1.10	1.25	5.00	3.15	3.50
p45k	Plain	1.05	1.05	5.00	3.65	3.40
	DED	1.15	1.35	5.00	4.00	4.05
	TED	1.10	1.40	5.00	4.60	4.20
	Cyclic	1.10	1.25	5.00	4.40	4.40
p77k	Plain	1.00	1.00	5.00	4.50	4.30
	DED	1.25	1.20	5.00	4.45	4.40
	TED	1.10	1.20	5.00	4.85	4.55
	Cyclic	1.20	1.30	5.00	4.80	4.75
p78k	Plain	1.00	1.00	5.00	3.40	3.35
	DED	1.25	1.55	5.00	3.10	3.05
	TED	1.20	1.65	5.00	2.95	2.90
	Cyclic	1.15	1.40	5.00	3.20	2.70
p81k	Plain	1.00	1.00	5.00	4.10	4.20
	DED	1.45	1.65	5.00	3.55	3.70
	TED	1.20	1.55	5.00	3.85	3.55
	Cyclic	1.30	1.65	5.00	4.50	4.35
p89k	Plain	1.00	1.00	5.00	4.00	3.95
	DED	1.10	1.05	5.00	4.15	4.05
	TED	1.10	1.05	5.00	4.85	4.60
	Cyclic	1.50	1.75	5.00	4.55	4.55
p100k	Plain	1.00	1.00	5.00	3.85	3.75
	DED	1.10	1.20	5.00	3.95	4.00
	TED	1.05	1.15	5.00	4.55	4.50
	Cyclic	1.00	1.10	5.00	3.90	3.75
p141k	Plain	1.00	1.00	5.00	3.80	3.80
	DED	1.15	1.50	5.00	3.50	3.75
	TED	1.15	1.15	5.00	3.60	4.00
	Cyclic	1.15	1.35	5.00	4.15	4.10
Avg.	Plain	1.02	1.04	5.00	3.90	3.83
	DED	1.22	1.45	5.00	3.90	3.89
	TED	1.15	1.33	5.00	4.10	4.07
	Cyclic	1.17	1.38	5.00	4.17	4.11