On Determining the Real Output Xs by SAT-Based Reasoning

Melanie Elm, Michael A. Kochte, Hans-Joachim Wunderlich
University of Stuttgart
Institute of Computer Architecture and Computer Engineering
Pfaffenwaldring 47, 70569 Stuttgart, Germany

***Abstract*—Embedded testing, built-in self-test and methods for test compression rely on efficient test response compaction. Often, a circuit under test contains sources of unknown values (X), uninitialized memories for instance. These X values propagate through the circuit and may spoil the response signatures. The standard way to overcome this problem is X-masking.**

Outputs which carry an X value are usually determined by logic simulation. In this paper, we show that the amount of Xs is significantly overestimated, and in consequence outputs are overmasked, too. An efficient way for the exact computation of output Xs is presented for the first time. The resulting X-masking promises significant gains with respect to test time, test volume and fault coverage.

***Index Terms*—X-Masking**

I. Introduction

During test application, unknown values (X values) can propagate from their sources to primary circuit outputs or scannable elements. Xs can stem from uninitialized memories within the circuit, from uninitialized or uncontrollable flip flops, from timing and synchronization problems or from tri-state circuitry. Unknown values propagate through gates if the gates have no controlling values on the off-path inputs. When Xs reach the compaction logic, they may corrupt the test signatures.

A generic structure for embedded test and built-in self-test for random logic circuits is depicted in figure 1. The circuit is configured with several scan chains. Not all flip flops or latches have to be made scannable. The scan chains are fed by a pattern generator, which can be reseeded internally or externally depending on the test application. The outputs are compacted by a compaction hardware. This can either be a space or a time compactor with or without feedback, or a combination of both. In order to protect the compactor signatures from corruption by X values, X-masking, X-tolerance or X-cancelling is employed. The fewer X values need to be handled, the higher is the compaction ratio and the better is the information revealed by the signature.

3-valued logic simulation is commonly used to compute the propagation of X values from the X sources to the inputs of the compaction logic. However, the 3-valued simulation is pessimistic w.r.t. the propagation of X values. This causes the introduction of X values at signals which actually have a defined logic value $\in \{0, 1\}$. The defined logic values result from signal reconvergences which 3-valued simulation is unable to evaluate correctly. Figure 2 depicts an example of a circuit structure under an input assignment with an X value. In 3-valued simulation, the output produces an X value. With the particular assignment, however, the output actually takes a logic value of 1 independent of the assignment of the X valued circuit input.

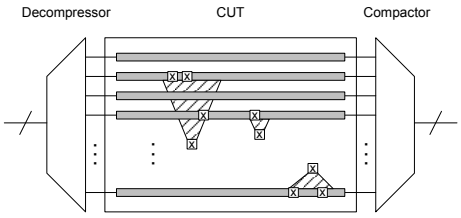

Fig. 1. Generic embedded test/BIST structure

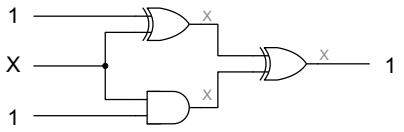

Fig. 2. Pessimism in 3-valued logic simulation

This work presents a novel method to accurately and efficiently determine the set of X valued output signals of a circuit for each pattern. The resulting reduction of X valued outputs directly impacts X-masking schemes and allows to optimize the design of compaction and X-masking logic, test pattern generation and test application, with favorable impact on design-for-test overhead, fault coverage, test time and data volume. Standard ATPG tools are not able to perform this analysis efficiently, as each random or deterministic pattern has to be dealt with separately.

The remainder of the paper is organized as follows: The next section discusses the state of the art in X-aware signature analysis and design-for-test. Section III presents the proposed method in detail. The impact of the method on X-masking is evaluated for a number of industrial circuits and with different scan configurations in section IV.

II. State of the Art

A. X analysis based on structural analysis

Multi-valued logic simulation with values from the domain of $0, 1, X$ and others is used to determine a pessimistic super set of propagation paths of X values and X valued circuit outputs. In n-valued simulation, propagation of X values stops at a gate only if the gate is controlled by one of the off-path signals. Reconvergences of propagation paths of X values are only pessimistically evaluated as the limited number of logic values cannot correctly track X propagation paths. Thus, the

set of pessimistic Xs can be divided into two disjoint subsets, the real Xs and the false Xs.

By increasing the accuracy of logic simulation, this pessimism can partly be overcome. By restricted symbolic simulation, as e.g. in [1] or applied to test generation as in [2], a subset of false Xs can be found. However, the methods cannot detect all false Xs at the outputs.

Structural circuit graph analysis techniques from the ATPG domain—termed static learning—can also be applied [3, 4] to increase simulation accuracy in presence of X valued signals. [5] showed that by incorporating indirect implications, some internal signals and circuit outputs with X values could be proven to actually have defined logic values. Static learning based methods are also incomplete, i.e. they cannot uncover all false X signals in the circuits. In addition, the search for the indirect implications requires many simulations of the circuit graph. The number of implications found is very high, and even with sophisticated learning criteria the size of the circuit graph increases significantly.

Another approximative technique, based on circuit partitioning, is proposed in [6]. Circuit partitions, comprising X valued fanout stems and their transitive fanout until the reconvergence, are simulated with both binary values 0 and 1 at the stem. The result at the reconvergence is then fed back into the 3-valued simulation of the whole circuit. This method is incomplete as well. In the subsequent sections, we will present a technique which is both complete and efficient.

B. Response compaction in presence of Xs

Different kinds of compaction schemes show different vulnerability to Xs and, in consequence, different techniques have been developed to prevent Xs from corrupting the compactor signature. They can be classified as X-tolerant space compaction, X-tolerant time compaction and X-masking for both space and time compaction. All of them benefit if the number of Xs to be masked can be reduced.

1) X-Tolerant Space Compaction: X-tolerant space compactors tolerate Xs by construction of the compactor. Most often these compactors employ error detecting or correcting codes to establish X-tolerance along with certain fault aliasing properties [7–9]. Others achieve X-tolerance by decoupling certain scan channels from each other in the response compactor [10]. In some cases a combination of X-tolerance with X-masking schemes can be found [11, 12].

However, X-tolerant compactors are only suitable if the amount of Xs is bound to a certain number p. If the amount of Xs fed into the compactor exceeds p, Xs will corrupt the signature. The limit p can be traded off against the compaction ratio. In order to optimize the compaction ratio, it is crucial to identify the real Xs instead of the pessimistic Xs.

2) X-Tolerant Time Compaction: Two types of time compactors and according X-tolerance schemes can be distinguished: finite impulse response (FIR) compactors as convolutional compactors [13] and infinite impulse response (IIR) compactors as MISR compactors (Multiple Input Shift Registers). Xs fed into FIR compactors corrupt only a subset of bits of the signature and can thereby inherently be tolerated and extracted on the tester [14]. The fewer Xs are fed into the compactor, the fewer signature bits are considered corrupted and thus, reveal defect information at the tester. By accurately identifying real Xs instead of PEXs, the amount of useable and informative signature bits can be maximized.

Xs fed into IIR compactors may affect all signature bits generated in all future compaction cycles. They can be tolerated by X-canceling schemes [15]. By symbolic simulation the Xs in the MISR signature are identified and can be canceled out by linearly combining MISR bits. The more Xs are fed into the MISR, the more dependencies with respect to Xs can be found in the MISR bits. Thus, canceling Xs becomes more difficult or impossible, which necessitates to increase the amount of MISR bits. Consequently, the identification of real Xs helps to optimize the cancellation logic and the compaction ratio.

3) X-masking: X-masking logic is synthesized in between the circuit outputs, i.e. scan outs, and the inputs of any arbitrary compaction logic [16–21]. During scan-out a predetermined X can be converted into a specified value by feeding it e.g. through a NAND gate and controlling the value of the output by an extra *mask input*.

Most masking approaches mask a super set of those scan cells exposed to X-propagation, where an optimization problem has to be solved. The X values determined by simulation form the so called ON-set, which needs to be masked. The scan cells carrying fault information should not be masked and are called the OFF-set. Masking some cells of the OFF-set does not necessarily result in loss of fault coverage if faults are propagated to several scan cells. All the other scan cells belong to the Don't-Care-set, which can also carry defect information not covered by the fault model.

Direct masking targets each X valued scan cell and synthesizes a logic function for generating the mask signals with the state of the pattern and bit counter as inputs. The fewer Xs have to be masked, the larger the Don't-Care-set is, and fewer area is needed for implementing the masking function [20, 21]. Indirect masking does not target a specific scan cell, but a complete scan chain or vector. Since a complete chain or vector is being masked, some cells from the OFF- and Don't-Care-set are masked as well. The higher the number of Xs, the more chains or vectors have to be masked and thus, overmasking increases with adverse impact on fault information. By identifying PEXs, the number of chains or vectors to be masked, and therefore overmasking, can be reduced.

For all the aforementioned X-handling schemes, masking, and tolerating schemes, it is beneficial to identify the circuit outputs carrying the real Xs instead of the pessimistic super set only. Thereby the efficiency of the schemes can be improved in terms of applicability, compaction ratio and information content of the signatures.

III. Exact Computation of real X Outputs

The exact identification of real X circuit outputs w.r.t. partial input assignments can be implemented by symbolic simulation of the circuit, as outlined below. This is computationally very expensive and sometimes even practically impossible. In section III-B, we present an efficient and accurate two-stage algorithm for the exact identification of real X outputs.

A. Identification of real X signals

Formal methods that accurately identify all real X signals would require the symbolic simulation of the circuit graph.

In a symbolic simulation, the circuit behavior is expressed in dependence of symbolic values. Symbolic simulation can be performed for example by constructing the ROBDD of the investigated signal in the circuit. By restricting the ROBDD [22] w.r.t. the specified input assignments, the accurate logic value of the signal can be determined. This however suffers from the known disadvantages of ROBDD synthesis as expensive representation of multipliers and strong dependence on variable ordering.

Another exact method is based on computing the forward implication at each gate by analyzing the intersection of the input cube with the implicants of the function and its inverse [23]. While the authors report results for circuits with up to 10 inputs, it is unclear whether this cube-based algorithm scales to circuits of thousands of inputs.

A directed analysis of each X valued circuit signal can also be implemented as a SAT-based problem instance or using ATPG tools. The underlying idea is to find at least two input assignments for which the considered signal takes complementary logic values, or to prove that no two such input assignments exist. In the first case, the considered signal is a real X, in the latter it is a false X signal. This check is equivalent to prove that both the stuck-at-0 and stuck-at-1 fault at the signal line can be activated. The additional difficulty lies in the fact that the proof has to be done for each pattern to be applied, in contrast to ATPG where only a single pattern is generated.

B. Efficient computation of real X outputs

In the following, we present an efficient method that allows to identify the exact set of real X output signals of a circuit for a given set of partial input assignments. The method is correct and complete, i.e. firstly, the value of any identified real X output depends on at least one X valued circuit input, and secondly, all real X outputs are found. Thus, the remaining X valued outputs form the set of false X outputs.

The evaluation of outputs causes rather high cost and can be restricted to the set of pessimistic X outputs as determined by 3-valued simulation of the set of partial input assignments. A simulation with more, but finite, values will be still pessimistic and does not solve the problem. For each input assignment, a SAT instance is created which allows to analyze the netlist outputs with X values (c.f. section III-B2 below). For each of these outputs the SAT instance is evaluated. The result of the algorithm is for each input assignment the set of real X outputs. Also, all false X outputs are implicitly identified and their actual fixed binary value is returned. The flow of the algorithm is depicted in figure 3.

1) 3-valued logic simulation: For each partial input assignment $\mathcal{A} : I \mapsto \{0, 1, X\}$ with I the set of netlist inputs, a 3-valued pessimistic logic simulation is performed to determine the values of the internal signals S and outputs O. Let O_x and S_x denote the subset of outputs O respectively signals S with a value of X. $I_x \subset I$ is the set of X valued inputs.

2) Generation of SAT instance for real X identification: For each output $o \in O_x$, we search for two different assignments $\mathcal{A}_1$ and $\mathcal{A}_2$ from $I_x \mapsto \{0, 1\}$, such that the resulting values of o under $\mathcal{A}_1$ and $\mathcal{A}_2$ differ. If such assignments do not exist, then the output o has a constant value $\in \{0, 1\}$ irrespective of the value assignments to the inputs I_x.

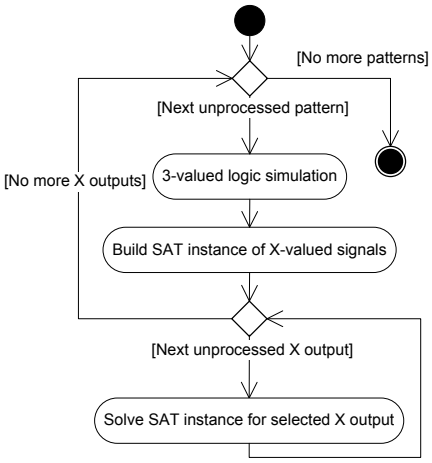

Fig. 3. Algorithm overview

To reduce the size of the generated SAT instance, we do not consider the complete input cone of the X valued output, but only the subset with signals from S_x (Fig. 4). S_x has already been determined by 3-valued simulation.

The SAT instance $P_{o,Ix}$ models two copies of this X valued input cone of output o. To represent different values in each copy of the cone, each signal s is encoded by separate variables s_1, s_2 in each copy. Thus, the two variables o_1 and o_2 represent the values of the output o in each copy. Each cone copy is described by the union C_{o1}, C_{o2} of the clauses representing the Boolean function of the gates that drive their output signal to an X value according to the 3-valued simulation. The outputs o_1 and o_2 are compared with each other by another two clauses D_o which are satisfied if and only if the values of o_1 and o_2 differ (Fig. 5):

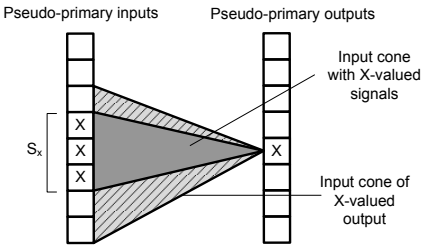

Fig. 4. X valued input cone of X valued output

$$D_o = \{\{o_1, o_2\}, \{\neg o_1, \neg o_2\}\}.$$

$$P_{o,Ix} = C_{o1} \cup C_{o2} \cup D_o.$$

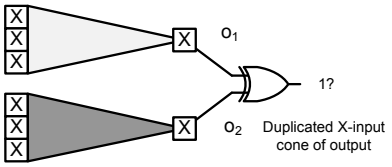

Fig. 5. Duplication of X valued input cone for considered X valued output

If this SAT instance is satisfiable, then there are (at least) two assignments to the inputs I_x for which the value at o differs, i.e. output o is a real X output. If the instance is not satisfiable, then no two assignments to I_x exist such that different values at o are generated. In this case, output o is a false X with a logic value independent of the assignment to I_x. The particular value can be easily

determined by 2-valued logic simulation of any (e.g. random) value assignment to I_x.

To avoid the overhead of generating a separate SAT instance for every output with potentially large overlapping of X valued input cones, a single SAT instance P_I is generated for each partial input assignment. This instance is formed by the union of the X valued input cones of the outputs, their copy and clauses D'_o for comparison:

$$P_{Ix} = \bigcup_{o \in Ox} (C_{o1} \cup C_{o2} \cup D'_o).$$

For each output o, the clauses for comparison D_o are extended to D'_o by a selector variable o_s which allows to chose the circuit output to be compared without altering the SAT instance:

$$D'_o = \{\{o_1, o_2, \neg o_s\}, \{\neg o_1, \neg o_2, \neg o_s\}\}.$$

If output o is to be evaluated, the selector variable o_s is set to 1 and all other selector variables are set to 0. This immediately satisfies all comparison clauses but the two related to o_s. Thereby the search space is effectively constrained to input assignments causing a difference at the considered output o.

IV. Experimental Results

This section evaluates the influence of the proposed algorithm for identification of real X outputs for a set of ISCAS'89 and larger industrial circuits kindly provided by NXP. The circuit properties are given in the next section. Then, we investigate the number of found real X outputs and compare the result with the indirect implication based approach from [5]. Section IV-C shows how this information impacts X-masking for chain respectively vector based masking schemes.

A. Circuit characteristics

Table I gives an overview of the characteristics of the industrial circuits. The second and third columns give the amount of primary and pseudo-primary inputs and outputs. The following columns show the maximum scan chain length and the amount of scan chains for two different scan configurations, the *original* and the *split* configuration. In order to reduce test time, we observe that many rather short scan chains are used today. For this reason, the *split* configuration is derived from the original one with many but very short scan chains. For circuit p378k, the two configurations are identical since the chains in the original configuration are already very short.

Circuit	#inp.	#outp.	Original scan conf.		Split scan conf.	
			#chains	length	#chains	length
p77k	3487	3400	13	303	143	28
p78k	3148	3484	65	64	195	22
p81k	4029	3952	8	514	144	29
p89k	4632	4557	18	963	306	57
p100k	5902	5829	18	792	270	53
p141k	11290	10502	24	486	264	45
p239k	18692	18495	40	541	360	61
p259k	18713	18495	40	541	360	61
p267k	17332	16621	45	494	360	62
p269k	17333	16621	45	494	360	62
p279k	18074	17827	55	409	385	59
p295k	18508	18521	11	1852	330	62
p330k	18010	17468	64	317	320	64
p378k	15732	17420	325	64	325	64
p388k	25005	24065	50	525	400	66
p418k	30430	29809	64	830	576	93

TABLE I
Circuit characteristics

B. Real X values on circuit outputs

Firstly, we investigate the degree of pessimism found in 3-valued logic simulation. For the set of industrial circuits, the number of real X outputs signals is determined for different X sources at the circuit inputs. Secondly, we compare the amount of found false X outputs with the results of the method based on indirect implications presented in [5].

In the first experiment, we assume a fixed number of inputs to be sources of X values. We investigate three different cases with 0.1%, 0.5% and 1% of the circuit inputs or flip flops as X sources. For each circuit, 16 different randomly generated configurations of X sources are investigated. In each configuration, the X sources are selected randomly and 32 random input patterns are assigned to the circuit. The propagation of the input X values to the outputs is computed by 3-valued simulation. Then, the real X outputs are determined for each input assignment. Table II lists for the three cases (0.1%, 0.5%, 1.0%) the X density at the outputs and the minimum, average and maximum ratio of real X outputs to pessimistic X valued outputs determined in the 3-valued logic simulation for the 16 X-source configurations. The X density denotes the average ratio of X valued outputs and the number of circuit outputs.

For the majority of circuits, a high percentage of X valued outputs can be proven to be false X. For the circuit p388k, in average only about 40% of the pessimistic X valued outputs are actually real Xs. In average over all circuits, just 75% of the X valued outputs are proven to be real X outputs.

In the second experiment, we compare the number of detected false X outputs with the indirect implication based method from [5]. We report results for the set of ISCAS'85/89 circuits evaluated in [5]. The experiment assumes that 50% of the circuit inputs are X sources. The circuit is evaluated with 32 random input patterns. Table III lists the average and maximum number of false X outputs per pattern for the method of [5] and the exact method proposed here.

For all circuits, the average and maximum number of identified false X outputs in the exact method exceeds the number of the indirect implication based method. For circuit s38417, the average and maximum number of identified false X outputs is more than 20x respectively 6x higher.

C. Impact on chain and vector masking

The exact analysis of real Xs is beneficial for all X-tolerating and X-masking architectures. In this section, we focus on the most simple ones, scan vector and scan chain masking, where either a complete vector or complete chain is masked per pattern. More complex schemes may benefit even more. Results are presented for the two different scan configurations mentioned above. Three different X-source distributions are chosen from the previous experiments: the distribution with minimum real X outputs, an average one and the one with the maximum amount of real X outputs. For these distributions we generate X-aware test patterns with a commercial ATPG tool and apply them to analyze the percentage of overmasked scan cells.

The results for the original scan configuration are reported in table IV. For the three X distributions, the table reports

Circuit	X-input ratio 0.1%				X-input ratio 0.5%				X-input ratio 1.0%			
	X-den.	Real X ratio [%]			X-den.	Real X ratio [%]			X-den.	Real X ratio [%]		
	[%]	min	avg	max	[%]	min	avg	max	[%]	min	avg	max
p77k	0.030	43.6	89.0	100.0	0.193	69.8	85.9	100.0	1.757	35.6	46.8	97.2
p78k	0.398	44.0	57.1	91.0	2.698	47.4	54.1	66.4	8.859	53.0	62.3	68.5
p81k	0.325	36.2	50.5	93.8	1.762	44.9	58.4	80.5	3.384	47.0	55.3	64.1
p89k	0.044	82.1	95.5	100.0	0.681	40.4	89.4	99.6	1.300	54.9	82.7	97.0
p100k	0.061	66.7	91.5	100.0	1.664	60.7	67.1	95.9	1.707	58.8	85.6	92.8
p141k	0.201	51.6	71.1	92.0	1.175	52.3	68.6	82.1	3.915	57.9	66.0	82.1
p239k	0.267	45.0	65.3	85.5	1.199	55.0	61.2	68.3	2.363	48.5	63.4	71.5
p259k	0.204	33.5	46.2	59.1	1.093	42.9	47.7	55.0	2.301	44.8	51.0	57.8
p267k	0.129	66.9	88.9	100.0	0.707	78.9	89.1	99.4	2.471	60.4	75.0	95.8
p269k	0.141	67.2	87.6	100.0	0.699	73.0	86.6	95.7	1.356	52.0	86.1	96.9
p279k	0.144	71.4	83.3	100.0	0.669	66.2	85.6	98.5	1.667	68.5	81.0	88.3
p295k	0.056	54.9	90.8	100.0	0.288	27.4	81.7	99.7	0.568	82.5	90.5	100.0
p330k	0.228	61.2	72.1	99.3	0.821	64.7	73.7	84.8	1.475	70.4	79.4	87.8
p378k	0.546	49.7	54.4	72.7	2.570	51.7	55.4	62.4	4.804	53.6	56.9	60.2
p388k	0.199	35.5	47.7	64.8	1.238	39.8	54.0	90.1	3.091	36.2	61.2	94.2
p418k	0.195	72.9	85.9	97.4	0.630	75.1	84.7	95.4	1.389	77.7	85.5	94.5

TABLE II
RATIO OF REAL X OUTPUTS FOR DIFFERENT X SOURCES CONFIGURATIONS. THE SMALLER THE PERCENTAGE, THE MORE FALSE XS WERE PESSIMISTICALLY IDENTIFIED BY LOGIC SIMULATION AND THE HIGHER IS THE GAIN BY EMPLOYING THE EXACT ANALYSIS.

Circuit	Indir. Impl. [5]		Proposed method	
	avg #F.X	max #F.X	avg #F.X	max #F.X
c2670	1.03	2	1.88	5
c5315	0.22	2	3.44	7
s5378	3.00	11	5.84	13
s9234	2.40	6	8.90	19
s13207	2.50	11	13.72	29
s15850	7.00	22	18.91	41
s35932	0.06	1	0.88	8
s38417	1.90	11	44.66	68
s38584	24.00	53	68.91	130

TABLE III
AVERAGE AND MAXIMUM NUMBER OF FALSE X OUTPUTS (F.X) FOUND BY [5] AND THE PROPOSED METHOD

the percentage of real X outputs w.r.t. PEX outputs, and the percentage of scan chains (resp. vectors) without false Xs in the exact analysis w.r.t. the chains (vectors) with at least one PEX value according to pessimistic 3-valued analysis:

$$result = \frac{\#XChains(\text{exact})}{\#XChains(\text{3-valued})} * 100$$

For circuit p77k and the avg. distribution, for example, only 67% of the PEX outputs are real X outputs. Consequently, only 72% of the scan vectors that capture a PEX value according to 3-valued simulation, actually capture real X values. 28% of the vectors are overmasked.

Clearly, the more Xs are present in the outputs, the more beneficial is an exact analysis of the real Xs. For the original scan configuration with few but long chains, vector masking is in general more precise than chain masking and thus the gain is higher for this scheme. Nonetheless, even for chain masking there is a gain for most circuits. For some circuits up to 10% of the masked chains are masked unnessecarily due to the pessimism of 3-valued logic simulation. For the vector masking, more than half of the masked scan chains are actually free of Xs in some cases.

Table V reports the corresponding results for the split chain configuration. In contrast to the results above, the gain is in most cases higher for the chain masking scheme, which is more precise for short but many scan chains. For some circuits, up to 30% of the masked chains were masked unnessecarily when the pessimistic evaluation is applied.

All the presented results reveal that a pessimistic X evaluation dramatically overestimates the amount of Xs in the circuit outputs. Hence, applying an exact analysis is beneficial for all X-tolerating and X-masking schemes and can help to reduce overmasking significantly.

D. Computing time

The cones with X valued signals that are mapped to a SAT instance are rather small. For the experiments of the last section, the number of gates ranges on average per circuit from 435 gates in circuit p89k to 22790 gates in circuit p378k, including the cone copy. The resulting SAT instances including the clauses for comparison range from 972 to 53259 clauses.

The algorithm has been implemented in a Java based EDA framework. All experiments have been conducted on an Intel Xeon CPU with 2.8 GHz frequency. The evaluation of a single output ranged from 4.37 ms (p77k) up to 25.66 ms (p378k) and includes the time for construction of the SAT instance.

V. CONCLUSIONS

Analysis of X propagation by n-valued logic simulation results in a pessimistic overestimation of X valued outputs. We presented a SAT-based approach for an exact identification of real Xs, which is beneficial for any X-handling method applied to test responses. The experimental results obtained for a large set of industrial designs reveal that up to 50% of the pessimistically identified X values are actually defined and in consequence, X-masking and X-tolerating schemes can be optimized to a large extent by employing an exact analysis. The analysis presented here is exact and feasible in terms of computing time even for recent industrial designs.

ACKNOWLEDGMENT

This work has partly been funded by the DFG under the contract WU 245-5.

REFERENCES

[1] J. Carter, B. Rosen *et al.*, "Restricted symbolic evaluation is fast and useful," in *Proc. International Conference on Computer-Aided Design*, 1989, pp. 38–41.

[2] S. Kundu, I. Nair *et al.*, "Symbolic implication in test generation," in *Proc. Conference on European Design Automation*, 1991, pp. 492–496.

[3] M. H. Schulz, E. Trischler, and T. M. Sarfert, "Socrates: a highly efficient automatic test pattern generation system," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 7, no. 1, pp. 126–137, 1988.

	Max. distribution, ratio [%]			Avg. distribution, ratio [%]			Min. distribution, ratio [%]		
Circuit	Real X	Chains	Vectors	Real X	Chains	Vectors	Real X	Chains	Vectors
p77k	90.9	98.7	91.1	67.2	99.5	72.1	43.4	96.1	66.1
p78k	68.4	100.0	100.0	62.4	91.4	89.8	51.6	85.2	95.5
p81k	72.3	100.0	72.8	69.6	100.0	70.3	66.5	100.0	68.6
p89k	95.8	94.8	96.0	86.8	97.5	86.6	58.0	91.8	57.8
p100k	89.4	99.7	89.9	85.6	99.6	86.3	63.9	97.7	66.1
p141k	77.2	99.0	79.2	57.1	97.9	62.3	57.8	99.9	97.4
p239k	73.7	99.1	87.5	63.9	99.5	73.3	61.3	99.1	72.5
p259k	47.6	91.0	55.3	47.9	94.7	57.6	44.6	94.6	55.2
p267k	93.1	100.0	94.8	72.1	99.7	88.8	58.8	99.9	85.2
p269k	97.6	100.0	98.2	85.0	100.0	89.6	53.7	99.6	62.7
p279k	88.5	99.4	91.4	83.1	99.4	88.9	69.9	100.0	81.3
p295k	83.7	100.0	85.6	89.8	100.0	90.2	83.8	100.0	84.3
p330k	87.5	99.3	92.3	81.3	99.4	88.1	71.1	99.5	83.8
p378k	57.5	88.8	99.0	55.0	87.2	100.0	52.1	86.3	100.0
p388k	90.6	92.3	86.0	53.5	92.5	65.3	43.3	94.7	61.2
p418k	93.3	99.8	96.8	84.9	99.7	90.7	78.4	99.7	87.2

TABLE IV

PERCENTAGE OF REAL X SCAN CELLS, CHAINS AND VECTORS W.R.T. THE PESSIMISTICALLY DETERMINED X CELLS, CHAINS AND VECTORS FOR THE ORIGINAL SCAN CHAIN CONFIGURATION AND THREE DIFFERENT X SOURCE DISTRIBUTIONS.

	Max. distribution, ratio [%]			Avg. distribution, ratio [%]			Min. distribution, ratio [%]		
Circuit	Real X	Chains	Vectors	Real X	Chains	Vectors	Real X	Chains	Vectors
p77k	90.9	97.6	97.1	67.2	96.0	94.4	43.4	70.1	91.2
p78k	68.4	98.2	100.0	62.4	88.0	98.9	51.6	82.9	99.2
p81k	72.3	87.9	92.7	69.6	92.7	92.6	66.5	88.7	95.8
p89k	95.8	96.9	98.0	86.8	97.6	92.6	58.0	87.7	69.5
p100k	89.4	97.8	94.5	85.6	96.5	93.4	63.9	91.8	88.4
p141k	77.2	87.5	96.1	57.1	81.7	95.0	57.8	94.4	100.0
p239k	73.7	86.5	99.1	63.9	84.7	97.5	61.3	84.1	99.1
p259k	47.6	65.0	90.8	47.9	67.4	93.7	44.6	67.5	93.1
p267k	93.1	99.7	99.7	72.1	99.4	97.1	58.8	96.6	90.2
p269k	97.6	99.7	99.5	85.0	98.7	99.9	53.7	96.3	96.3
p279k	88.5	98.3	95.9	83.1	97.6	98.1	69.9	95.9	99.2
p295k	83.7	99.1	99.4	89.8	97.7	99.0	83.8	95.8	98.3
p330k	87.5	98.2	99.5	81.3	98.4	99.1	71.1	98.1	99.1
p378k	57.5	88.8	99.0	55.0	87.2	100.0	52.1	86.3	100.0
p388k	90.6	78.3	98.0	53.5	75.6	97.2	43.3	76.0	98.2
p418k	93.3	98.6	99.9	84.9	98.2	97.1	78.4	98.3	97.3

TABLE V

PERCENTAGE OF REAL X SCAN CELLS, CHAINS AND VECTORS W.R.T. THE PESSIMISTICALLY DETERMINED X CELLS, CHAINS AND VECTORS FOR THE SPLIT SCAN CHAIN CONFIGURATION AND THREE DIFFERENT X SOURCE DISTRIBUTIONS.

[4] W. Kunz, D. Stoffel, and P. Menon, "Logic optimization and equivalence checking by implication analysis," *IEEE Trans. CAD*, vol. 16, no. 3, pp. 266 –281, mar 1997.

[5] S. Kajihara, K. K. Saluja, and S. M. Reddy, "Enhanced 3-valued logic/fault simulation for full scan circuits using implicit logic values," in *Proc. IEEE European Test Symposium (ETS)*, 2004, pp. 108–113.

[6] S. Kang and S. A. Szygenda, "Accurate logic simulation by overcoming the unknown value propagation problem," *Simulation*, vol. 79, no. 2, pp. 59–68, 2003.

[7] M. Sharma and W.-T. Cheng, "X-filter: Filtering unknowns from compacted test responses," in *Proc. IEEE International Test Conference*, 2005, pp. 1090–1098.

[8] Y. Han, Y. Hu *et al.*, "Theoretic analysis and enhanced x-tolerance of test response compact based on convolutional code," in *Proc. ASP-DAC.*, 2005, pp. 53–58.

[9] S. Mitra and K. S. Kim, "X-compact: an efficient response compaction technique," *IEEE Trans. CAD*, vol. 23, no. 3, pp. 421–432, 2004.

[10] W. Rajski and J. Rajski, "Modular compactor of test responses," in *Proc. IEEE VLSI Test Symposium.*, 2006, pp. 242–251.

[11] P. Wohl, J. Waicukauski, and S. Ramnath, "Fully x-tolerant combinational scan compression," in *IEEE International Test Conference.*, 2007, pp. 1–10.

[12] T. Rabenalt, M. Goessel, and A. Leininger, "Masking of x-values by use of a hierarchically configurable register," in *14th IEEE European Test Symposium.*, may 2009, pp. 149 –154.

[13] J. Rajski, J. Tyszer *et al.*, "Finite memory test response compactors for embedded test applications," *IEEE Trans. CAD*, vol. 24, no. 4, pp. 622–634, 2005.

[14] G. Mrugalski, A. Pogiel *et al.*, "Diagnosis with convolutional compactors in presence of unknown states," in *Proceedings of the IEEE International Test Conference.*, 2005, p. 10 pp.

[15] R. Garg, R. Putman, and N. Touba, "Increasing output compaction in presence of unknowns using an x-canceling misr with deterministic observation," in *26th IEEE VLSI Test Symposium.*, 2008, pp. 35–42.

[16] M.-T. Chao, S. Wang *et al.*, "Response shaper: a novel technique to enhance unknown tolerance for output response compaction," in *Proc. Int'l Conference on Computer-Aided Design*, 2005, pp. 80–87.

[17] V. Chickermane, B. Foutz, and B. Keller, "Channel masking synthesis for efficient on-chip test compression," in *International Test Conference.*, 2004, pp. 452–461.

[18] J. Rajski, J. Tyszer *et al.*, "X-press: Two-stage x-tolerant compactor with programmable selector," *IEEE Trans. CAD*, vol. 27, no. 1, pp. 147–159, 2008.

[19] H. Tang, C. Wang *et al.*, "On efficient x-handling using a selective compaction scheme to achieve high test response compaction ratios," in *Proc. International Conference on VLSI Design*, 2005, pp. 59–64.

[20] M. Naruse, I. Pomeranz *et al.*, "On-chip compression of output responses with unknown values using lfsr reseeding," in *Proceedings of the International Test Conference.*, vol. 1, 2003, pp. 1060–1068.

[21] Y. Tang, H.-J. Wunderlich *et al.*, "X-masking during logic BIST and its impact on defect coverage," *IEEE Trans. VLSI Systems*, vol. 14, no. 2, pp. 442–451, 2006.

[22] R. E. Bryant, "Graph-based algorithms for boolean function manipulation," *IEEE Trans. Computers*, vol. 35, no. 8, pp. 677–691, 1986.

[23] S. Chandra and J. Patel, "Accurate logic simulation in the presence of unknowns," in *Proc. International Conference on Computer-Aided Design*, 1989, pp. 34–37.