# Towards Increasing Test Compaction Abilities of SAT-based ATPG through Fault Detection Constraints

Stephan Eggersglüß*† Melanie Diepenbeck* Robert Wille* Rolf Drechsler*†

*Institute of Computer Science
University of Bremen
28359 Bremen, Germany
{diepenbeck,rwille}@informatik.uni-bremen.de

†Cyber-Physical Systems
DFKI GmbH
28359 Bremen, Germany
{Stephan.Eggersgluess, Rolf.Drechsler}@dfki.de

***Abstract*—Automatic Test Pattern Generation (ATPG) based on Boolean Satisfiability (SAT) is a robust alternative to classical structural ATPG. Due to the powerful reasoning engines of modern SAT solvers, SAT-based algorithms typically provide a high test coverage because of the ability to reliably classify hard-to-detect faults. However, a weak point of SAT-based ATPG is the test compaction ability. In this paper, we propose a new methodology which combines the classical SAT-based ATPG formulation with additional fault detection constraints resulting in a pseudo-Boolean optimization problem. This leads to an increasing test compaction ability of SAT-based ATPG. Experiments show that the resulting test set generated by pure SAT-based ATPG without any test compaction techniques can significantly be decreased by up to 49%.**

## I. Introduction

The test set size for the post-production test of digital circuits is an important factor. A large test set typically leads to high test costs and immense efforts are undertaken to reduce test pattern counts in order to reduce test costs. At the same time, a high fault coverage and high quality tests are necessary to guarantee the absence of manufacturing defects and, therefore, the production of high quality devices. With the increasing size of integrated circuits as well as with the recent developments of 3D-integrated circuits, the significance of test pattern reduction and robust ATPG algorithms are expected to grow.

*Automatic Test Pattern Generation* (ATPG) based on Boolean Satisfiability (SAT) [1], [2] has been shown to provide a high fault or test coverage for large industrial circuits [3], [4] since the powerful learning and implication techniques of modern SAT solvers are well suited to generate tests for hard-to-detect faults. Classical structural ATPG approaches typically have problems to cope with these kind of faults. Recently, SAT-based algorithms have also been shown to be well suited to generate high-quality tests targeting for example small delay defects [5]–[7] or As-Robust-As-Possible tests [8] where usually a significantly increased search space has to be considered. However, a drawback of SAT-based ATPG is that the test compaction abilities are typically not as good as the test compaction abilities of structural ATPG algorithms.

SAT-based dynamic compaction techniques were proposed in [9]. Here, a test cube for an initial fault is generated. Then, additional faults are targeted taking the pre-generated test as constraint. By this, the unspecified values of the test cube are assigned in a way that other faults can be detected as well. The SAT-based approach presented in [10] works in a different manner. A set of faults $F$ is chosen and the ATPG directly generates a test detecting all faults in $F$ if possible. However, this approach heavily relies on a heuristic which determines which faults could possibly be detected together.

In this paper, we propose a new basic test formulation which is able to improve the test compaction abilities of SAT-based ATPG. In contrast to previous approaches, the aim of this methodology is to generate an initial test for one fault which is able to detect a large number of other faults without explicitly targeting any other faults. In order to achieve this, the SAT-based ATPG formulation is combined with additional fault detection constraints influenced by fault simulation and path tracing techniques. These constraints determine if a fault can be detected and propagated locally. A pseudo-Boolean optimization procedure is then applied to the SAT-based ATPG formulation which leverages the powerful SAT solving engine and improves the local fault detection of the test.

By this, the generated test is typically able to detect a larger number of faults without targeting any of them explicitly. In general, applying pseudo-Boolean solving techniques to problems such as SAT-based ATPG where the majority of constraints are in CNF is similar to the application of direct SAT encodings as used in [5], [6]. Experiments show that the

test set size generated by SAT-based-ATPG can significantly be reduced. The proposed basic test formulation can easily be coupled with other dynamic or static test compaction techniques to achieve further reduction.

The remainder of this paper is structured as follows: The next section briefly reviews the basics on SAT-based ATPG. Afterwards, Section III outlines the idea of fault detection constraints before results obtained by an initial experimental evaluation are reported in Section IV. Finally, the paper is concluded and possible future work is discussed in Section V.

## II. Preliminaries

In contrast to classical structural ATPG which works directly on the gate-level netlist, SAT-based algorithms work on a Boolean formula in *Conjunctive Normal Form* (CNF). A CNF $\Phi$ is a conjunction of clauses. A clause $\omega$ is a disjunction of literals and a literal $\lambda$ is a Boolean variable in its positive ($\lambda$) or negative form ($\overline{\lambda}$). Due to the development of powerful implication and learning techniques, SAT solvers, e.g. MiniSat [11], are well suited to solve hard problem instances. In order to leverage the powerful solving techniques for ATPG, the ATPG problem has to be formulated in CNF [1] (see [4] for detailed information).

Each connection $x$ is assigned a Boolean variable $x$. The functionality of each gate $g$ is transformed into a set of clauses $\Phi_g$. The CNF $\Phi_C$ of the circuit $C$ is then constructed by a conjunction of the CNF of each single gate of $C$, i.e.

$$\Phi_C = \Phi_{g_1} \cdot \ldots \cdot \Phi_{g_n}.$$

In order to generate a test for a fault $f$, the circuit CNF $\Phi_C$ has to be augmented by additional constraints for fault detection and fault propagation $\Phi_F$ for the specific fault $f$, i.e.

$$\Phi^f_{\text{Test}} = \Phi_C \cdot \Phi^f_F.$$

The solution space of $\Phi^f_{\text{Test}}$ includes all possible tests which detects $f$ and the SAT solver provides one satisfying assignment which can be transformed into a test or proves that no such assignment exists.

Due to the powerful solving techniques, SAT solvers often work as core engines for other kind of solvers, e.g. *Pseudo-Boolean Optimization* (PBO) solvers like clasp [12]. PBO solvers can be applied to search for the best solution included in the solution space of a CNF.[1] In order to apply PBO solvers to the ATPG problem, the CNF $\Phi^f_{\text{Test}}$ has to be augmented by an optimization function $\mathcal{F}$. The function $\mathcal{F}$ is defined as follows:

$$\mathcal{F} = c_1 x_1 + \ldots + c_n x_n$$

Each Boolean variable $x_i$ is assigned a constant value $c_i$. The result of $\mathcal{F}$ is the accumulation of all constants for which the corresponding variable is set to 1. The task of the PBO solver is to search a solution which satisfies $\Phi^f_{\text{Test}}$ and at the same time minimizes (or maximizes) the result of $\mathcal{F}$.

## III. Integration of Fault Detection Constraints

In the classical SAT-based ATPG formulation, exactly one fault $f$ is targeted and the generated test is guaranteed to detect this specific fault. However, the tests are able to detect other faults as well. This is exploited during test compaction to obtain a small test set. The generated test is fault simulated and all faults which are detected by this test can be deleted from the fault list. The number of additional faults has a significant influence on the test compaction effectiveness. Dynamic test compaction techniques [13], [14] used in practice leverage the circumstance that tests typically contain many unspecified values ($X$). Once a test is generated, additional faults are explicitly targeted to check whether the unspecified values can be specified to detect these explicit faults. However, this is a cumbersome methodology, since additional faults have to be targeted very often. This is particularly serious for SAT-based ATPG since the time to generate the SAT instance is not negligible.

In order to address this, we propose a new SAT-based ATPG formulation which does not explicitly target additional faults but include additional fault detection constraints. These constraints are able to determine whether faults can be detected and propagated locally. The aim is to directly generate a test which satisfies as many local fault detection conditions as possible. This does not guarantee that these faults are actually detected since the fault detection conditions are only locally applied, e.g. they do not consider reconvergences and fault masking. However, the generated tests are typically able to detect a significant larger number of faults due to the integration of more information.

The fault detection conditions added to the SAT instance are influenced by fault simulation and path tracing techniques. Two additional variables $x_{f0}$ and $x_{f1}$ are assigned to each connection $x$ denoting whether the stuck-at-0 ($f0$) or stuck-at-1 ($f1$) fault[2] can be locally detected and propagated. In order to activate the local fault detection condition, i.e. to set $x_{f0}(x_{f1}) = 1$, the following properties have to be satisfied:

- The value of connection $x$ has to be inverted to the fault value, i.e. 0 for ($x_{f1}$) and 1 for ($x_{f0}$). Otherwise, the fault cannot be activated.

[1] This is only one application possibility of PBO solvers. These solvers can also be applied to a broader range of applications.

[2] Due to similarity of the stuck-at and transition fault model, this technique can be applied to both fault models. For reasons of simplicity the stuck-at fault model is considered only in this paper.

TABLE I
FAULT DETECTION CONDITIONS

| | |
|---|---|
| $\Phi_{Activate}$ | $x = 1 \rightarrow x_{f1} = 0$<br>$x = 0 \rightarrow x_{f0} = 0$ |
| $\Phi_{Gate}$ | $y = \text{cv} \rightarrow x_{f1} = 0$<br>$y = \text{cv} \rightarrow x_{f0} = 0$ |
| $\Phi_{Path}$ | $z_{f0} = 0 \wedge z_{f1} = 0 \rightarrow x_{f1} = 0$<br>$z_{f0} = 0 \wedge z_{f1} = 0 \rightarrow x_{f0} = 0$ |

- All other inputs of the successor gate $h$ have to assume the non-controlling value in order to propagate the fault through $h$.
- At least one local fault detection variable of the successor gate must be activated, i.e. equal to 1, to ensure that there is a propagation path to an observation point.

This results in the implications shown in Table I. The implications in this table are given for connection $x$ which is input of a gate $z$ with inputs $x, y$. The controlling value of gate $z$ is denoted by *cv*.

The implications for each connection are transformed into CNF (resulting in the CNF $\Phi_{\text{FDC}}$) and added to $\Phi^{f}_{\text{Test}}$ as local fault detection conditions. Note that these constraints do not alter the solution space with respect to the tests but are only used to activate/deactivate the fault detection conditions.

Then, an optimization function $\mathcal{F}$ is formulated. Since the goal is to maximize the activated fault detection conditions, $\mathcal{F}$ includes all fault detection variables whose corresponding faults have not been detected yet. Given a set of yet undetected faults $F = f_1, \ldots, f_n$, the optimization function $\mathcal{F}$ is formulated as follows:

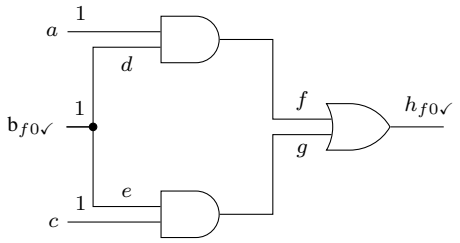
$$\mathcal{F} = (-1) \cdot f_1 + \ldots + (-1) \cdot f_n$$

By this, all constants which are associated with an activated fault detection variable are accumulated. Since PBO solvers typically minimize the result, each fault detection variable is associated with a negative variable. It would also be possible to prioritize certain regions by associating higher constant values to certain faults.

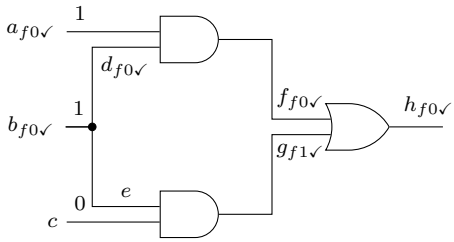
Applying a PBO solver to $\Phi^{f}_{\text{Test}} \cdot \Phi_{\text{FDC}}$ and the given optimization function $\mathcal{F}$, the solver provides the test with the maximum of activated fault detection conditions. This test typically detects a larger number of additional faults as a test generated with the classical SAT-based ATPG procedure. Note that this procedure does not prevent the use of additional test compaction techniques like dynamic compaction but rather complements these techniques.

These concepts are eventually illustrated by the following example.

*Example 1:* Consider the circuit shown in Fig. 1(a). Using the classical SAT-based ATPG formulation targeting e.g. a stuck-at-0 fault at signal $b$, a test pattern 111 may result. This only detects one further fault, namely a stuck-at-0 at signal $h$. If instead fault detection constraints as shown below the circuit of Fig. 1(b) are additionally applied, the pattern 111 does not satisfy the optimization criterion. Hence, another pattern, namely 110, is obtained. This pattern detects not only two, but six faults in total (namely stuck-at-0's at signals $a, b, d, f, h$ and a stuck-at-1 at signal $g$).




(a) Using classical ATPG formulation




$\Phi_{\text{Activate}}$: $a = 1 \rightarrow a_{f1} = 0$; $a = 0 \rightarrow a_{f0} = 0$; $b = 1 \rightarrow b_{f1} = 0$; $b = 0 \rightarrow b_{f0} = 0$; ...

$\Phi_{\text{Gate}}$: $a = \text{cv} \rightarrow b_{f1} = 0$; $a = \text{cv} \rightarrow b_{f0} = 0$; $b = \text{cv} \rightarrow a_{f1} = 0$; $b = \text{cv} \rightarrow a_{f0} = 0$; ...

$\Phi_{\text{Path}}$: $f_{f0} = 0 \wedge f_{f1} = 0 \rightarrow a_{f1} = 0$; $f_{f0} = 0 \wedge f_{f1} = 0 \rightarrow a_{f0} = 0$; $f_{f0} = 0 \wedge f_{f1} = 0 \rightarrow b_{f1} = 0$; $f_{f0} = 0 \wedge f_{f1} = 0 \rightarrow b_{f0} = 0$; ...

Opt. function: $\mathcal{F} = -a_{f1} - a_{f0} - b_{f1} - \ldots$

(b) Using proposed ATPG formulation

Fig. 1. Exemplary illustration

## IV. EXPERIMENTAL RESULTS

This section presents experimental results for the proposed approach. The proposed SAT-based ATPG procedure was implemented in C++. The solver clasp [12] was used as PBO/SAT solver. The ISCAS'85, ISCAS'89 and ITC'99 circuits were used as benchmarks. The following procedure was used for test generation: All stuck-at faults of the circuits were

TABLE II
EXPERIMENTAL RESULTS - TEST SET SIZE

| Circuit | #Faults | #Tests Classic | #Tests FDC | Red. |
|---|---|---|---|---|
| c1908 | 3766 | 203 | 136 | 33% |
| c2670 | 5060 | 230 | 172 | 25% |
| c3540 | 7036 | 227 | 116 | 49% |
| c5315 | 10384 | 295 | 243 | 18% |
| c6288 | 12512 | 19 | 16 | 16% |
| c7552 | 14888 | 298 | 214 | 28% |
| s713 | 1342 | 113 | 71 | 37% |
| s5378 | 10026 | 298 | 270 | 9% |
| s9234 | 17968 | 483 | 427 | 12% |
| s13207 | 24778 | 669 | 516 | 23% |
| s15850 | 30326 | 528 | 396 | 25% |
| s35932 | 67064 | 72 | 68 | 6% |
| s38417 | 73194 | 1357 | 884 | 35% |
| s38584 | 73404 | 584 | 465 | 20% |
| b01 | 192 | 20 | 15 | 25% |
| b04 | 2892 | 149 | 86 | 42% |
| b05 | 4372 | 59 | 48 | 19% |
| b14 | 42546 | 1766 | 1433 | 19% |
| b15 | 39054 | 1144 | 842 | 26% |

stored in a fault list. Then, a fault $f$ is taken from the fault list and a test is generated detecting fault $f$ using the proposed SAT-based ATPG formulation with maximized fault detection conditions. The test is fault simulated and all additional faults detected by this test are removed from the fault list. This is continued until all faults became classified.

Table II shows the impact of the procedure on the test set size. The number of target faults for each circuit is given in column *#Faults*. The number of generated tests of the proposed approach (column *#Tests FDC*) is compared against the number of tests of a classical SAT-based ATPG approach (column *#Tests Classic*). The reduction of the test set size is given in column *Red.* Both approaches do not use any further compaction techniques. The results show that the test set size of the proposed approach can be significantly reduced. The highest reduction can be achieved for circuit c3540 where only around half the tests are needed to detect all faults. The results clearly show the potential of this new SAT-based ATPG formulation with respect to the test set size.

## V. CONCLUSIONS AND FUTURE WORK

A new SAT-based ATPG approach has been presented which couples additional fault detection constraints with the classical SAT-based ATPG formulation. An optimization solver is applied to maximize local fault detection conditions in order to generate tests detecting a larger number of faults without explicitly targeting additional faults. Experiments shows the potential of the new SAT-based ATPG formulation and the impact on the test set size. Future work focuses on the combination of the proposed approach with classical compaction techniques for further test set reduction. Particularly, the application of this techniques for the reduction of delay tests, e.g. generated by timing-aware ATPG, should be investigated. Besides that, run-time remains an issue. The proposed approach still requires much more run-time than the classical approach. However, this is left to be addressed by the development of techniques or heuristics for run-time reduction, i.e. using incremental SAT.

## REFERENCES


[1] T. Larrabee, "Test pattern generation using Boolean satisfiability," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 11, no. 1, pp. 4–15, 1992.

[2] P. Stephan, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "Combinational test generation using satisfiability," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 9, pp. 1167–1176, 1996.

[3] S. Eggersglüß and R. Drechsler, "Efficient data structures and methodologies for SAT-based ATPG providing high fault coverage in industrial application," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 9, pp. 1411–1415, 2011.

[4] ——, *High Quality Test Pattern Generation and Boolean Satisfiability*. Springer, 2012.

[5] M. Sauer, A. Czutro, T. Schubert, S. Hillebrecht, I. Polian, and B. Becker, "SAT-based analysis of sensitisable paths," in *Proceedings of the IEEE Symposium on Design and Diagnosis of Electronic Circuits and Systems*, 2011, pp. 93–98.

[6] M. Sauer, J. Jiang, A. Czutro, I. Polian, and B. Becker, "Efficient SAT-based search for longest sensitisable paths," in *Proceedings of the IEEE Asian Test Symposium*, 2011, pp. 108–113.

[7] S. Eggersglüß, M. Yilmaz, and K. Chakrabarty, "Robust timing-aware test generation using pseudo-boolean optimization," in *Proceedings of the IEEE Asian Test Symposium*, 2012.

[8] S. Eggersglüß and R. Drechsler, "As-Robust-As-Possible test generation in the presence of small delay defects using pseudo-Boolean optimization," in *Proceedings of Design, Automation and Test in Europe*, 2011, pp. 1291–1297.

[9] A. Czutro, I. Polian, P. Engelke, S. M. Reddy, and B. Becker, "Dynamic compaction in SAT-based ATPG," in *Proceedings of the IEEE Asian Test Symposium*, 2009, pp. 187–190.

[10] S. Eggersglüß, R. Krenz-Bååth, A. Glowatz, F. Hapke, and R. Drechsler, "A new SAT-based ATPG for generating highly compacted test sets," in *Proceedings of the IEEE Symposium on Design and Diagnosis of Electronic Circuits and Systems*, 2012, pp. 230–235.

[11] N. Eén and N. Sörensson, "An extensible SAT solver," in *Proceedings of the International Conference on Theory and Applications of Satisfiability Testing*, ser. Lecture Notes in Computer Science, vol. 2919, 2004, pp. 502–518.

[12] M. Gebser, B. Kaufmann, A. Neumann, and T. Schaub, "Conflict-driven answer set solving," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2007, pp. 386–392.

[13] P. Goel and B. C. Rosales, "Test generation and dynamic compaction of tests," in *Proceedings of the International Test Conference*, 1979, pp. 189–192.

[14] S. Kajihara, I. Pomeranz, K. Kinoshita, and S. M. Reddy, "Cost-effective generation of minimal test sets for stuck-at faults in combinational logic circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 14, no. 12, pp. 1496–1504, 1995.