ISBN: 978-1-6654-4502-3

Part Number: CFP21DTW-USB

2021 IEEE East-West Design & Test Symposium (EWDTS) Proceedings

Batumi, Georgia, September 10 – 13, 2021

Proceedings of 2021 IEEE East-West Design & Test Symposium (EWDTS)

IEEE Catalog Numbers:
XPLORE COMPLIANT: CFP21DTW-ART
ISBN: 978-1-6654-4503-0

USB: CFP21DTW-USB
ISBN: 978-1-6654-4502-3

Additional copies of this publication are available from:
Curran Associates, Inc
57 Morehouse Lane
Red Hook, NY 12571 USA
Phone: (845) 758-0400
Fax: (845) 758-2633
E-mail: curran@proceedings.com

CONTENTS

SAT Solvers Application of Deriving All Test Pairs Detecting Robust Testable PDFs

A.Yu. Matrosova
Tomsk state university
Tomsk, Russia
mau11@yandex.ru

V.V. Andreeva
Tomsk state university
Tomsk, Russia
avv.21@mail.ru

V.Z. Tychinskiy
Tomsk state university
Tomsk, Russia
tvz.041@gmail.com

***Abstract*— It is known that if we have set of test pairs of neighbor Boolean vectors for robust testable PDF for each path considered in the given circuit, we may derive test sequence for these faults consisting of fragments that are characterized by minimal length and minimal power consumption. Conventionally they try to find at least one test pair. It is possible to find all test pairs using operations on ROBDDs. In the frame of SAT technology it is possible to find products presenting test pairs one by one. It seemed that having got several products presenting test pairs we may derive rather high quality test sequence for the given set of circuit paths. New approach to deriving test pairs in the frame of SAT technology is suggested. The approach is based on deriving the combinational circuit that represents all test pairs. This circuit is constructed from the given combinational circuit and the chosen circuit path. The complexity of the obtained circuit practically coincides with the complexity of the given combinational circuit. Using this circuit we may obtain products representing test pairs one by one.**

Keywords—combinational and sequential circuits, conjunctive normal form (CNF), Tseitin CNF, Boolean satisfiability problem, SAT solvers, robust testable path delay faults (PDFs)

I. INTRODUCTION

In this paper, we consider testing of robust testable Path Delay Faults (PDFs). It is known that such testing allows identifying the fault path and, if possible, removing its delay in order to increase circuit performance. Current approaches [1-6] are oriented to finding if only one test pair for rising (falling) transition of a circuit path. We suggest finding all test pairs consisting of neighbor Boolean vectors for each path from the given set. A test pairs deriving is based on the obtaining of the Tseitin CNF [7]. The complexity of this CNF deriving linearly depends on the number of gates in the circuit. The obtained CNF is fed to the input of the SAT solver, which allows getting all or several test pairs for the selected path.

In section II, the problem statement is discussed. In section III, algorithm of deriving test pairs is represented. Experimental results are presented in section IV. Section V is devoted to the concluding remarks.

II. PROBLEM STATEMENT

Consider the path α of single-output combinational circuit C, which need to be tested, as a sequence of variables: x_i, u_1, u_2,…, u_{r-1}, u_r. Circuit inputs are marked as x_1,…, x_n, outputs of it gates marked as u_1, u_2,…, u_{m-1}, u_m. Wherein the circuit is divided by levels [8], in which the gates of neighboring levels are connected by a precedence relation. In what follows, it is assumed that the levels are numbered from the inputs of the circuit instead outputs. Gates of a lower level always have lower numbers than gates from a higher level. The order of numbering of gates within one level can be arbitrary. The last variable u_m corresponds to the output of the circuit. The beginning and the endings of the path α marked with variables x_i and u_r, respectively.

III. ALGORITHM OF DERIVING TEST PAIRS

In the proposed algorithm test pairs are obtaining using a Boolean path difference [9]:

$$D_\alpha = (D_{u_r} / D_{u_{r-1}}) \wedge (D_{u_{r-1}} / D_{u_{r-2}}) \wedge \ldots \wedge (D_{u_2} / D_{u_1}) \wedge (D_{u_1} / D_{x_i}),$$

$$D_{u_i} / D_{u_{i-1}} = f_{u_i}^{u_{i-1}=0} \oplus f_{u_i}^{u_{i-1}=1},$$

where f_{u_i} is represents function of subcircuit C_{u_i}.

There are three cases as the result of Boolean difference $D_{u_i} / D_{u_{i-1}}$:

1. Trivial case: $D_u / D_{x_i} = 1$ when gate u is NOT / XOR / NXOR gate:

$$D_u / D_{x_i} = (x_k \oplus (x_i = 0)) \oplus (x_k \oplus (x_i = 1)) = x_k \oplus \bar{x}_k = 1;$$

$$D_{u_i} / D_{u_j} = (x_k \sim (x_i = 0)) \oplus (x_k \sim (x_i = 1)) = \bar{x}_k \oplus x_k = 1.$$

2. $D_u / D_{x_i} = (x_1 \wedge \ldots \wedge x_{i-1} \wedge x_{i+1} \wedge \ldots \wedge x_n)$, when gate u is the AND / NAND gate:

$$D_u / D_{x_i} = \left[(x_i = 0) \wedge \bigwedge_{k=1,\, k\neq i}^{n} x_k\right] \oplus \left[(x_i = 1) \wedge \bigwedge_{k=1,\, k\neq i}^{n} x_k\right] =$$

$$= 0 \oplus \left[\bigwedge_{k=1,\, k\neq i}^{n} x_k\right] = (x_1 \wedge \ldots \wedge x_{i-1} \wedge x_{i+1} \wedge \ldots \wedge x_n);$$

$$D_u / D_{x_i} = \overline{\left[(x_i = 0) \wedge \bigwedge_{k=1,\, k\neq i}^{n} x_k\right]} \oplus \overline{\left[(x_i = 1) \wedge \bigwedge_{k=1,\, k\neq i}^{n} x_k\right]} =$$

$$= 1 \oplus \overline{\left[\bigwedge_{k=1,\, k\neq i}^{n} x_k\right]} = (x_1 \wedge \ldots \wedge x_{i-1} \wedge x_{i+1} \wedge \ldots \wedge x_n).$$

3. $D_u / D_{x_i} = \overline{(x_1 \vee \ldots \vee x_{i-1} \vee x_{i+1} \vee \ldots \vee x_n)}$, when gate u is the OR / NOR gate:

$$D_u / D_{x_i} = \left[(x_i = 0) \vee \bigvee_{k=1,\, k\neq i}^{n} x_k\right] \oplus \left[(x_i = 1) \vee \bigvee_{k=1,\, k\neq i}^{n} x_k\right] =$$

$$= \left[\bigvee_{k=1,\, k\neq i}^{n} x_k\right] \oplus 1 = \overline{(x_1 \vee \ldots \vee x_{i-1} \vee x_{i+1} \vee \ldots \vee x_n)};$$

$$D_u / D_{x_i} = \overline{\left[(x_i = 0) \vee \bigvee_{k=1,\, k\neq i}^{n} x_k\right]} \oplus \overline{\left[(x_i = 1) \vee \bigvee_{k=1,\, k\neq i}^{n} x_k\right]} =$$

$$= \overline{\left[\bigvee_{k=1,\, k\neq i}^{n} x_k\right]} \oplus 0 = \overline{(x_1 \vee \ldots \vee x_{i-1} \vee x_{i+1} \vee \ldots \vee x_n)}.$$

In the last two cases if gate u has only two inputs (x_i and x_j), the Boolean difference D_u / D_{x_i} simplified to x_j and $\bar{x}_j$, respectively.

978-1-6654-4503-0/21/$31.00 ©2021 IEEE

Thus, the expressions for the Boolean differences of the path gates, and, as a consequence, the Boolean difference of the path D_α can be represented as an additional subcircuit of the circuit C (fig. 1-2):

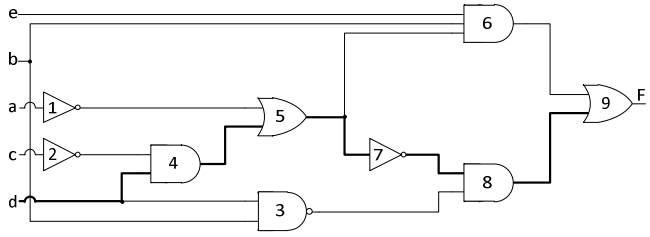

Fig. 1. Combinational circuit C.

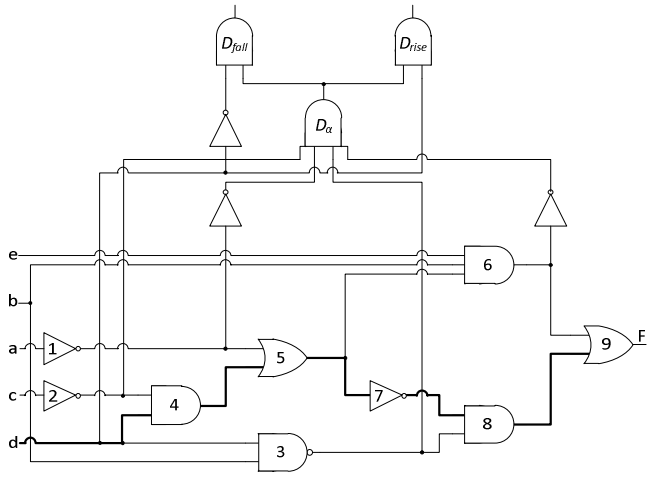

Fig. 2. Representation of the Boolean difference D_α and DNFs for rising (D_{rise}) and falling (D_{fall}) transitions as combinational circuits C_α, C_{rise}, C_{fall}.

Tseitin CNF of the gate y combinational circuit C derived according to the following rules:

$$y = \bar{x} : (x \vee y) \wedge (\bar{x} \vee \bar{y});$$

$$y = x_1 \oplus x_2 : (\bar{x}_1 \vee x_2 \vee y) \wedge (x_1 \vee \bar{x}_2 \vee y) \wedge$$
$$\wedge (x_1 \vee x_2 \vee \bar{y}) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{y});$$

$$y = \overline{x_1 \oplus x_2} : (x_1 \vee x_2 \vee y) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee y) \wedge$$
$$\wedge (\bar{x}_1 \vee x_2 \vee \bar{y}) \wedge (x_1 \vee \bar{x}_2 \vee \bar{y});$$

$$y = \bigwedge_{k=1}^{n} x_k : \left[\bigwedge_{k=1}^{n} (x_k \vee \bar{y})\right] \wedge \left[y \vee \bigvee_{k=1}^{n} \bar{x}_k\right];$$

$$y = \overline{\bigwedge_{k=1}^{n} x_k} : \left[\bigwedge_{k=1}^{n} (x_k \vee y)\right] \wedge \left[\bar{y} \vee \bigvee_{k=1}^{n} \bar{x}_k\right];$$

$$y = \bigvee_{k=1}^{n} x_k : \left[\bigwedge_{k=1}^{n} (\bar{x}_k \vee y)\right] \wedge \left[\bar{y} \vee \bigvee_{k=1}^{n} x_k\right];$$

$$y = \overline{\bigvee_{k=1}^{n} x_k} : \left[\bigwedge_{k=1}^{n} (\bar{x}_k \vee \bar{y})\right] \wedge \left[y \vee \bigvee_{k=1}^{n} x_k\right].$$

Note that Tseitin CNF of the circuit derive as the result of AND operation with the Tseitin CNFs of the gates of that circuit.

The resulting CNF K_α is a function of at most (n + m) variables, where n is the number of inputs of the circuit C, m is the number of elements of the subcircuit C_α (with the output D_α). It contains both the domain of unit and the domain of zero tuples of the function D_α. To extract the conjunctions of the DNF D_α, it is necessary to set the variable corresponding to the output of the circuit C_α equal to one. Then the resulting CNF $K_\alpha^{D_\alpha=1}$ is fed to the input of the SAT solver. The obtained solution in the form of an interval can be expanded. To do this, it is necessary to replace one of its components with an undefined value and substitute the values of the input variables in the CNF of the region of zero sets K_α ($K_\alpha^{D_\alpha=0}$). If the obtained CNF is not feasible, then the expansion of the interval is acceptable.

DNF D_α represents all v_2 test cases for rising and falling transitions that are not separate from each other. It is necessary to separate them. Denote with $D_{rise} = D_\alpha \wedge x_i$ and $D_{fall} = D_\alpha \wedge \bar{x}_i$ DNFs for a rising and falling transition, respectively.

Denote with D'_{rise} and D'_{fall} DNFs obtained from the DNF D_{rise} by deleting the variable x_i and $\bar{x}_i$, respectively. Then we denote with D_{rob} DNF which represents all test pairs of neighboring sets in the variable x_i for robust testable delay faults on the path α: $D_{rob} = D'_{rise} \wedge D'_{fall}$.

Note that the DNF D_{rob} does not contain the variables x_i and $\bar{x}_i$. Corresponding to the DNF D_{rob} conjunctions Boolean vectors in the space of variables $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$ are define test pairs in the space of n variables. Vectors of the pair are obtained by assigning the value 0 and 1 to the variable x_i. Further from this test pair triplets of vectors are subsequently formed that detect the delays of inverse switches of the signals on the tested path.

Since the D_α conjunctions are obtained one by one it is possible to obtain only part of them. Such approach may decrease count of the detectable faults (because there is a chance that one of the DNFs D_{rise}, D_{fall} will be empty) but will increase the speed of deriving test pairs.

On the other hand, it is possible to get guaranteed test pairs if we fed to the input of the SAT solver CNF K_{rob} obtained using similar transformations over K_α. However, this requires defining the rules for excluding a variable from the Tseitin CNF since the standard way of excluding a variable leads to a distortion of the given CNF and the loss of information about the structure of the circuit. In the Tseitin CNF, removing the variable x_i from the circuit is equivalent to the removal of the subcircuit with the output corresponding to the variable x_i. In this case, it is also required to remove all lines connecting the output of the excluded subcircuit with other elements of the circuit. As a consequence, the number of inputs of the associated circuit elements decreases, which requires their transformation (and changes in the corresponding Tseitin CNFs). In the general case, the Tseitin CNF of the gate is replaced by the Tseitin CNF of the transformed gate, which implements the function without excluded inputs. If the gate has no inputs left, the original subcircuit with output of this gate should also be excluded from the circuit.

It is required to clearly determine the correspondence of clauses to their gates to implement the correct removal of subcircuits from the Tseitin CNF. Further, while the gate

removing, it is required to determine its preceding gates which were encountered in the Tseitin CNF once and to exclude the corresponding clauses from the expression. To do this, it is proposed to save the duplicate clauses while the Tseitin CNF construction. At the same time, to shorten the notation, it is proposed to indicate a degree equal to the number of repetitions of the clause in the Tseitin CNF for each clause.

On the fig. 3 you can see the result circuit after excluding variable *d* from the circuit on the fig. 2. It should be noted that in this case the gates D_{rise} and D_{fall} degenerate (because after excluding one of the inputs they become buffer gates and give the same function after conjunction), and the element previously matched D_{α} will be corresponds to D_{rob} after the transformation.

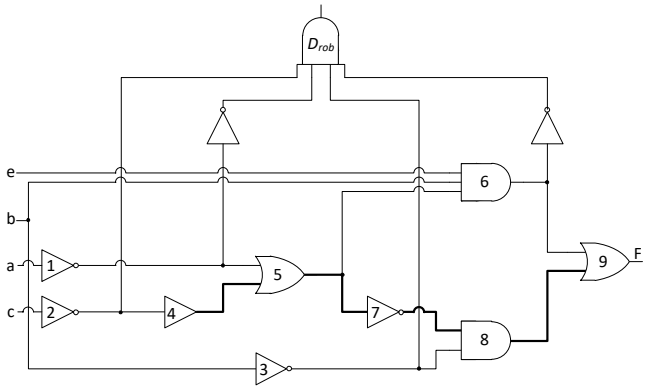

Fig. 3. Representation of the Boolean difference D_{α} as combinational circuit C_{rob}.

IV. Experimental Results

For circuits from ISCAS'89 we chose at least 10 of the longest path for each circuit output. In table I the information about the considered circuits is given. In the second column the circuit name are marked. In the third, fourth and fifth columns the numbers of inputs (Nin), outputs (Nout) and gates (Ngates) of circuit are given, correspondingly. In the fifth and sixth columns the numbers of total selected (Npt) and robust testable (Npr) paths of the circuit are represented. The seventh column shows the ratio (Pr) of robust testable paths to selected paths (in percentage)

TABLE I. Information about the considered benchmarks

No.	Benchmark	Nin	Nout	Ngates	Npt	Npr	Pr
1	s298	17	20	119	146	95	65%
2	s344	24	26	160	159	111	70%
3	s400	24	27	162	258	213	83%
4	s444	24	27	181	237	142	60%
5	s641	54	42	379	309	137	44%
6	s820	23	24	289	232	230	99%
7	s953	45	52	395	338	313	93%
8	s1196	32	32	529	334	162	49%
9	s1488	14	25	653	312	291	93%
10	s1494	14	25	647	336	306	91%

Table II presents the results of the algorithm that performs the obtaining of test sequences using Tseitin CNF and SAT solver:

TABLE II. Experimental results of an algorithm using a Tseitin CNF to build a test sequence

No.	Total count of switches	Peak switches
1	267	5
2	338	10
3	611	7
4	348	6
5	382	5
6	628	9
7	1101	9
8	500	12
9	1004	8
10	1067	8

Tables III-V present the comparative results of algorithms, in which the size of DNF D_{α} was limited by k conjunctions relative to the algorithm that detects all test pairs:

TABLE III. Experimental results for algorithm in which the size of DNF was limited by 10 conjunctions

No.	Reduction of execution time	Robust testable PDFs covering	Total count of switches	Peak switches
1	0%	100%	98%	100%
2	33%	98%	104%	125%
3	23%	96%	95%	100%
4	8%	100%	87%	100%
5	2%	100%	109%	83%
6	1%	100%	105%	100%
7	7%	100%	123%	82%
8	32%	98%	99%	100%
9	1%	100%	122%	80%
10	0%	100%	122%	89%

TABLE IV. Experimental results for algorithm in which the size of DNF was limited by 5 conjunctions

No.	Reduction of execution time	Robust testable PDFs covering	Total count of switches	Peak switches
1	13%	100%	99%	100%
2	55%	95%	88%	88%
3	46%	87%	81%	100%
4	23%	96%	83%	100%
5	5%	100%	109%	83%
6	6%	100%	104%	100%
7	15%	100%	122%	73%
8	51%	96%	97%	100%
9	1%	99%	121%	80%
10	3%	100%	121%	89%

TABLE V. EXPERIMENTAL RESULTS FOR ALGORITHM IN WHICH THE SIZE OF DNF WAS LIMITED BY 2 CONJUNCTIONS

No.	Reduction of execution time	Robust testable PDFs covering	Total count of switches	Peak switches
1	26%	99%	96%	100%
2	76%	92%	81%	88%
3	69%	62%	57%	100%
4	52%	77%	68%	100%
5	33%	100%	109%	83%
6	25%	97%	102%	100%
7	40%	97%	116%	73%
8	68%	95%	95%	100%
9	21%	96%	116%	80%
10	23%	95%	115%	89%

The experimental results show that even in cases where only two DNF D_α conjunctions are extracted using the SAT solver, the test sequence obtained on their basis detects over 90% of robust testable PDFs (according to Table I, such paths can be from 40 to 90 percent relative to the total number of paths in the circuit). At the same time, the time taken to build the test sequence decreases in proportion to the number of solutions obtained (on average, the acceleration is 11, 22 and 43 percent compared to the method that extracts all the solutions).

V. CONCLUSION

In this paper new approach to deriving test pairs that detect robust testable PDFs in logical circuits is proposed. This approach based on the usage of the Tseitin CNF and SAT techniques. Proposed method can be used on finding one, several, or all test pairs consisting of neighbor Boolean vectors for a circuit path. Each pair of neighbor Boolean vectors generates three neighbor Boolean vectors that detect delays of both rising and falling transitions of a path.

REFERENCES

[1] P. Lindgren, M. Kerttu, M. Thornton and R. Drechsler “Low power optimization technique for BDD mapped circuits” ASP-DAC 2001, pp. 615-621.

[2] R.S. Shelar, S.S. Sapatnekar “An efficient algorithm for low power pass transistor logic synthesis” ASP-DAC 2002, pp. 87-92.

[3] G. Gekas, D. Nikolos, E. Kalligeros, X. Kavousianos, "Power aware test-data compression for scan-based testing", ICECS 2005. 12th IEEE International Conference on, pp. 1-4.

[4] J.T. Tudu, E. Larsson, V. Singh, V.D. Agrawal, "On Minimization of Peak Power for Scan Circuit during Test", Test Symposium 2009 14th IEEE European, 2009, pp. 25-30.

[5] Z. Kotasek, J. Skarvada, J. Strnadel, "Reduction of Power Dissipation Through Parallel Optimization of Test Vector and Scan Register Sequences", IEEE International Symposium on Design and Diagnostics of Electronic Circuits and Systems,2010, pp. 364-369.

[6] V. Sinduja, S. Raghav, J. P Anita, "Efficient don't-care filling method to achieve reduction in test power", ICACCI, 2015, pp. 478-482.

[7] Tseitin G. S. O slozhnosti vyvoda v ischislenii vyskazyvanij (On the Complexity of Derivation in Propositional Calculus) // Zapiski nauchnyh seminarov LOMI AN SSSR. – 1968. – v. 8. – pp. 234-259.

[8] Armstrong, D. B. (1966). On Finding a Nearly Minimal Set of Fault Detection Tests for Combinational Logic Nets. // IEEE Transactions on Electronic Computers, EC-15(1), 1966. – P. 66–73.

[9] A.Yu.Matrosova, V.V. Andreeva, E.A. Nikolaeva Finding Test Pairs for PDFs in Logic Circuits Based on Using Operations on ROBDDs // Russian Physics Journal. 2018. Vol. 61, № 5. P. 994-999.

Camera-ready was prepared by Chumachenko S.
Approved for publication: 13.09.2021. Format 60x841/8.
Relative printer's sheets: . Circulation: 100 copies.
Published by SPD FL Stepanov V.V.
Ukraine, 61168, Kharkov, Ak. Pavlova st., 311