Equivalence Between Test Generation and Pseudo-Boolean Minimization

Sidi M. Aourid

Publication: #1035

Department d'Informatique et de Recherches Operationnelle

Faculté des Arts et des Sciences

Université de Montréal

C.P. 6128 Succursale Centre-Ville, Montréal, PQ, H3C 3J7

Abstract

We show that generating test for combinational circuits can be reduced to an equivalent unconstrained Pseudo-Boolean (PB) minimization problem. The formulation of the objective function for the equivalent problem is based on the real transformation and the l2-norm. To simplify the formulation, we decompose the equivalent problem into two constrained sub-problems that are easy to solve. The advantage of this decomposition is that the solution to each sub problem is either unique or, if the PB function has more than one minimum, the two solutions are equivalent. The formulation for single-output circuits and multi -output circuits are given. In each case, excellent results are obtained, which show the efficiency and the effectiveness of the proposed equivalence.

1 Introduction

Automatic test pattern generation ATPPG consists in producing a set of vectors called a test pattern, capable of indication whether or not the circuit is correct. To evaluate the performance of logic circuits, the test vectors are applied to circuits under test (CUT) and these outputs are then compared with the expected (correct) outputs. If the response of the CUT differs from the expected response, then faults are present in the CUT. The subject of generating test vectors has received considerable attention, and many ATPG methods for combinational and sequential circuits has been proposed [9] - [7]. Existing approaches for combinational circuits can be classified into two categories [4]: **Structural methods**, like the D-algorithm [9], PODEM [3], FAN [2] and SOCTATES [10]; or **Algebraic methods** like the Boolean difference [5], the decomposition and factorization of Boolean expression [8], and so on. Another algebraic method proposed by SPIROS [6], consists in transforming the ATPG problem into a sequence of Pseudo-Boolean (BP) maximization functions. To guarantee a near minimal and complete test, a good decision must be made at each stage of the algorithm. This restriction is generally possible when the objective function has

more than one optimizing point. To overcome this difficulty another formulation based on real transformation and the L2-norm is proposed. The method consists in minimizing an unconstrained BP function. The minimum points of this problem correspond exactly to the maximum points of SPIROS's formulation. To avoid making a decision at each stage, we decompose the unconstrained problem into two constrained minimizations. The advantage if this decomposition is that the minimum points is either unique or, if the subproblem has more than one solution, the minimum points are equivalent. Consequently, the method guarantees a minimal and complete test set. Finally, we describe how the objective function for the equivalent problem can be implemented correctly in a neural network, Due to the high computational properties of the neural networks, this extensions can offer significant results in the synthesis for testability strategies.

2 Background and Basic Definitions

Let F_i be a set of faults at stage i (by stage i, we mean the iteration i for any algorithm). (X_i) is a test vector that detects a set of faults D_i. We say that (X_i) is optimal if it detects a maximum number if faults in F_i. At stage $i+1$, we put $F_{i+1} = F_i - D_i$: A set of faults not detected by the vector (X_i). Most algorithms proposed up to now for ATPG are based on the same principle. At each stage, they try to detect a maximum number of faults with a minimal test set, until for some index k, F_k is either empty or it contains undetectable faults only. The difference between existing methods lies in how a vector is generated for some specific faults during stage i. The following algorithm describes the paradigm of ATPG for combinational circuits.

The efficiency of the algorithm depends on the method used to produce test vectors. In general good algorithm must guarantee two constraints: ***First***, the test set must be complete and minimal; ***Second***, the CPU time required to generate this set must be small.

Recently, it has been shown that algebraic methods, in particular boolean difference

Algorithm 1 ATPG Paradigm

Require: Set of Faults: F
  while $F \neq \varnothing$ **do**
    Generate test vectors
    Select Detected Faults
    Update Set F
  end while

method, are good candidates for rapidly generating an optimal test set. The boolean difference of a function $F(x)$ with respect to variable x_i is given by [5]:

$$\frac{\partial F(x)}{\partial x_i} = F_{x_i} \oplus F_{x_i'}$$

Where, F_{x_i} and $F_{x_i'}$ are respectively, the positive and the negative Shannon's Cofactors. The boolean difference w.r.t. x_i compares the value of F, when $x_i = 0$ against when $x_i = 1$. $\frac{\partial F(x)}{\partial x_i} = 0 \iff F_{x_i} = F_{x_i'}$. In this case, we say that F is not sensitive to changes in input x_i, it is then *unobservable*.

The test sets that detect x_i stuck-at-0 and stuck-at-1 are defined by: $\{X : x_i \cdot \frac{\partial F(x)}{\partial x_i} = 1\}$ and $\{X : \bar{x_i} \cdot \frac{\partial F(x)}{\partial x_i} = 1\}$ respectively. Note that the above formulation can also be used to produce tests for stuck-at-faults on internal signal lines.

In order to develop our formulation, we transform, the boolean domain into a real domain. This transformation permits avoiding a complete enumeration in generating a test set [6]. It is defined by the function T and stated as:

$$\begin{aligned} T : B^n &\longrightarrow P_B \\ y &\longmapsto T(y) \quad where\ B = \{0, 1\} \end{aligned}$$

with the following properties:

$$\begin{aligned} T(y_1.y_2) &= T(y_1).T(y_2) \\ T(y_1 + y_2) &= T(y_1) + T(\bar{y_1}).T(y_2) = T(y_1) + T(y_2) - T(y_1).T(y_2) \\ T(y) &= 1 - T(\bar{y}) \\ [T(y)]^n &= T(y) \end{aligned}$$

T(y) is a pseudo-boolean function that maps the Boolean domain into a real domain. For simplicity, we can denote $T(y)$ by T_y.

$$\begin{aligned} T_y : B^n &\longrightarrow \mathbb{R} \\ y &\longmapsto T_y(x) \end{aligned}$$

Arithmetic addition, subtraction and scalar multiplication are defined as the corresponding in a real field. To illustrate the transformation we give the following example.

Example. *Consider the following boolean function:*

$$y = (\overline{x_1} \cdot x_2 + x_1 \cdot \overline{x_2}) \cdot x_3$$

Its real transformation is given by:

$$\begin{aligned} \Rightarrow T_y &= T\left(\overline{x_1} \cdot x_2 + x_1 \cdot \overline{x_2}\right) \cdot T(x_3) \\ &= \Big[T(\overline{x_1} \cdot x_2) + T(x_1 \cdot \overline{x_2}) - T(\overline{x_1} \cdot x_2) \cdot T(x_1 \cdot \overline{x_2})\Big] \cdot T(x_3) \\ &= [\overline{x_1} \cdot x_2 + \overline{x_1} \cdot x_2] \cdot x_3 \end{aligned}$$

with:

$$T(x) = x \quad and \quad T(\overline{x}) = 1 - x$$

3 Equivalence between ATPG and Pseudo-Boolean minimization

3.1 Test Patterns for Single-Output Circuits

Let C be a combinational circuits, $F = \{f_j : where\, f_j\, is\, a\, fault\, in\, C\}$

Lemma 3.1. *Suppose that α_j are boolean functions such that : $\alpha_j(x) = 1$ for $j = 1, \cdots, N$ then:*

$$\sum_{j=1}^{N} T(\alpha_j(x)) = N$$

Proof: By Using the definition of function α_j and the transformation's properties, the proof is self evident.

Assumption: Suppose now that Z is the output of circuit C, T_z is its transformation and T_{Z-f_j} is the transformation of the circuit C with the fault f_j. Then, the generation of test vectors is given by the following theorem.

Theorem 3.2. *The test vectors for the circuit with the above assumptions are the solutions to the 0-1 minimization problem (P):*

$$(P): \quad min_{x \in B^n}\, f(x) = \frac{1}{2}\left[N\Big(T_z(x) - 1\Big) + \Big(1 - 2T_z(x)\Big)\sum_{j=1}^{N} T_{z-f_j}(x)\right]^2 \tag{1}$$

Proof: The best pattern that detects faults f_j is a vector x that satisfies:

$$Z \oplus Z_{f_j} = 1 \quad for\, j = 1, \cdots, N$$

So by using the Lemma, we obtain:

$$\sum_{j=1}^{N} T(Z \oplus Z_{f_j}) = N$$

The solution of this equation is equivalent to:

$$minf(x) = \left| \sum_{j=1}^{N} T(Z \oplus Z_{f_j}) - N \right|_2$$

where $\|.\|_2$ is the L_2-norm.

which we can also write by using the transformation properties as:

$$(P) : min_{x \in B^n} f(x) = \frac{1}{2} \left[N(T_z - 1) + (1 - 2T_z).\sum_{j=1}^{N} T_{Z-f_j} \right]^2$$

which complete the proof.

In Spiro's formulation [6], the test pattern are generated by the solution to problem (S):

$$(S) : max \quad S(x)_{x \in B^n} = \sum_{j=1}^{N} T(Z \oplus Z_{f_j}) \tag{2}$$

Problem (P) and (S) are not equivalent, even though they achieve their optima at the same point. To show this, we give the following Lemma:

Lemma 3.3. *The optimal values for Problem (P) and Problem (S) are achieved at the same point.*

Proof: Let x^* and S^* denote the solution and the optimal value for Problem (S) respectively. Our objective is to show that every $x \in B^n$:

$$S(x) \leq S(x^*) \iff f(x^*) \leq f(x)$$

where f is the objective function for problem (P). Note that f can also be written as:

$$f(x) = \frac{1}{2}\Big[S(x) - N\Big]^2$$

From the definition of S, we have:

$$S(x) \leq S(x^*) \quad \forall\, x \in B^n \Rightarrow S(x) - 2N \leq S^* - 2N, \quad \forall\, x \in B^n \tag{3}$$

Now, consider that N is the number of faults in circuit C, then:

$$S^* \leq N \Rightarrow S^* - 2N \leq 0$$

so, (3) is replaced by:

$$S(x) - 2N \leq S^* - 2N \leq 0 \quad \forall\, x \in B^n \, and \, N > 0 \tag{4}$$

Combining Equations (3) and (4) yields to:

$$\begin{aligned}
& S(x).\Big(S(x) - 2N\Big) \geq S^*\Big(S^* - 2N\Big), \quad \forall\, x \in B^n \, and \, N > 0 \\
\Longrightarrow & S(x).\Big(S(x) - 2N\Big) + N^2 \geq S^*\Big(S^* - 2N\Big) + N^2 \\
\Longrightarrow & \Big(S(x) - N\Big)^2 \geq \Big(S^* - N\Big)^2 \\
\Longrightarrow & f(x) \geq f(x^*), \quad \forall x \in B^n \\
\Longrightarrow & f(x^*) = f^* = min\, f(x) \quad \forall x \in B^n
\end{aligned}$$

Which complete the proof.

3.2 Decomposition of Problem (P) into two Constrained subproblems:

The objective function for the Problem (P) is given by:

$$f(x) = \frac{1}{2}\Big[N.\Big(T_z - 1\Big) + \Big(1 - 2T_Z\Big).\sum_{j=1}^{N} T_{Z-f_j}\Big]^2$$

A test pattern detects a fault $f_j \iff T_Z \neq T_{Z-f_j}$. However, according to the transformation properties, we can easily show that $T_Z = 0$ or 1, which means that a pattern x detects a fault $f_j \iff T_Z = 1 - T_{Z-f_j}$. With this equation, we distinguish two cases independently: $T_z = 1$ and $T_Z = 0$. Transferring these two cases to the function $f(x)$, we

obtain two constrained problem $P1$ and $P2$ such that:

$$\begin{aligned} \underset{x \in B^n}{\text{minimize}} \quad & f_1(x) = \frac{1}{2}\Big[\sum_{j=1}^{N} T_{Z_{f_j}}\Big]^2 \\ \text{subject to} \quad & T_Z(x) = 1 \end{aligned} \tag{5}$$

and

$$\begin{aligned} \underset{x \in B^n}{\text{minimize}} \quad & f_2(x) = \frac{1}{2}\Big[-N + \sum_{j=1}^{N} T_{Z_{f_j}}\Big]^2 \\ \text{subject to} \quad & T_Z(x) = 0 \end{aligned} \tag{6}$$

In the 0-1 space, each subproblem guarantees a minimal set of vectors that detects a maximum number of faults in F. The advantages offered by this decomposition results from the convex nature of the function f_1 and f_2. Consequently, every local minimum is also global. This result is very important in the sens that global solution is the union of the global minima of problems (P1) and (P2), which yields an optimal minimal test set. The following algorithm describes our approach.

Algorithm 2 ATPG Paradigm

Require: Set of Faults: F
  while $F \neq \varnothing$ **do**
    Solve (P1)
    Select Detected Faults
    Update Set F
    Solve (P2)
    Update Set F
  end while

Example. *Taken from [11]*

Consider the circuit in Fig. 1, with the set of faults:

$$F = \{a-0, a-1, b-0, c-0, c-1, d-0, d-1, e-0, e-1\}$$

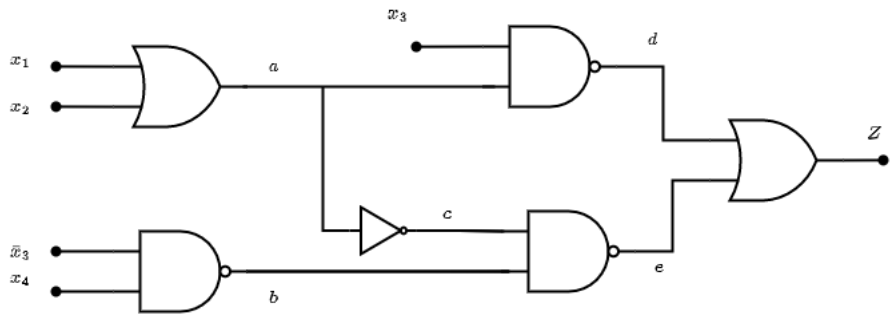

Figure 1: A Simple example circuit with Fault list - F

According to Fig.1, the output and its real transformation is given by:

$$Z = x_1x_3 + x_2x_3 + \bar{x_1}\bar{x_2}\bar{x_3}x_4$$

$$T_Z = x_1x_3 + x_1x_2x_3 + \bar{x_1}\bar{x_2}\bar{x_3}x_4$$

We also derive the real transformation of fault-outputs:

$$T_{Z_{a-0}} = \overline{x_3}.x_4$$

$$T_{Z_{a-1}} = x_3$$

$$T_{Z_{b-0}} = T_{Z_{c-0}} = T_{Z_{e-0}} = x_1.x_3 + \overline{x_1}x_2x_3$$

$$T_{Z_{b-1}} = x_1x_3 + \bar{x_1}x_2.x_3 + \bar{x_1}\bar{x_2}$$

$$T_{Z_{c-1}} = x_1.x_3 + \bar{x_1}x_2.x_3 + \bar{x_3}x_4$$

$$T_{Z_{d-0}} = \bar{x_1}\bar{x_2}\bar{x_3}x_4$$

$$T_{Z_{d-1}} = T_{Z_{e-1}} = 1$$

x_1	x_2	x_3	x_4	T_z	f_1
0	0	0	0	0	4.5
0	0	0	1	1	18
0	0	1	0	0	8
0	0	1	1	0	8
0	1	0	0	0	2
0	1	0	1	0	8
0	1	1	0	1	32
0	1	1	1	1	32
1	0	0	0	0	2
1	0	0	1	0	8
1	0	1	0	1	32
1	0	1	1	1	32
1	1	0	0	0	2
1	1	0	1	0	8
1	1	1	0	1	32
1	1	1	1	1	32

Table 1: All Possible Solutions

By formulating the objective function $f_1(x)$, we obtain Problem (P1):

$$
\begin{aligned}
(P1): \quad & minf(x) = \frac{1}{2}\Big[2 + 2.\bar{x_3}.x_4 + x_3 + 5.\bar{x_1}.x_2.x_3 + \bar{x_1}.\bar{x_2} + \bar{x_1}\bar{x_2}\bar{x_3}.x_4\Big]^2 \\
& S.T: \quad x_1.x_3 + x_1.x_2.x_3 + \bar{x_1}\bar{x_2}\bar{x_3}.x_4 = 1 \\
& \qquad\quad x \in B^n
\end{aligned}
$$

To solve this problem, we can for example use CPLEX [1], based on the Branch-and-Bound method. However, since the number of variables is not large and for illustration purpose, we can enumerate all the possible points exhaustively. The following table illustrates this:

According to Table 1, we see that with $T_Z = 1$, the minimum of function f_1 is 18, which occurs at $x^* = (0, 0, 0, 1)$. This vertex is the unique optimal test for the faults: $a-1, b-0, c-0$ and $e-0$. Then we update the set of faults F which is now $F = \{a-0, b-1, c-1, d-0, d-1, e-1\}$; and in the same way, we solve Problem (P2), such that:

$$(P2): \quad min\, f(x) = \frac{1}{2}\Big[-4 + 2.\bar{x_3}.x_4 + 2x_1.x_3 + 2\bar{x_1}x_2x_3 + \bar{x_1}.\bar{x_2} + \bar{x_1}\bar{x_2}\bar{x_3}.x_4\Big]^2$$

$$S.T: \quad x_1x_3 + x_1x_2x_3 + \bar{x_1}\bar{x_2}\bar{x_3}x_4 = 0$$

$$x \in B^n$$

In this case, we obtain three equivalent optimal vertices which are : $(0101), (1001)$ and (1101). The same procedure is repeated until all faults are detected. Table 2, summarizes our results.

Note that, the test patterns in the optimal solution column are equivalent. Then,one optimal test for circuit C is $T = \Big\{(0001), (0101), (0110), (0011)\Big\}$

Problem Solved	Optimal Solutions	Faults detected
P1	0 0 0 1	$a-1, b-0, e-0$
P2	0 0 0 1, 1 0 0 1, 1 1 0 1	$a-0, c-1, d-1, e-1$
P1	0 1 1 0, 0 1 1 1, 1 0 1 0, 1 0 1 1, 1 1 1 0, 1 1 1 1	$d-0$
P2	0 0 0 0, 0 0 1 0, 0 0 1 1	$b-1$

Table 2: Set Of Problems and their solutions

3.3 Test Patterns for Multi-Output Circuits

It is possible to extend our approach to multi-output circuits. To do this we make the following assumption.

Assumption: A fault f_j is detected by a test pattern if the effect of this fault is propagated to at least one output. Consider a circuit C with m outputs: $Z_1, Z_2, \cdots Z_m$ and a set of faults $F = \{F_j : j = 1, \cdots, N\}$. The assumption implies that:

$$\alpha_j = (Z_j \oplus Z_{1-f_j}) + \cdots + (Z_m \oplus Z_{m-f_j}) \quad \forall f_j$$
$$\Longrightarrow T(\alpha_j) = 1 \quad and \quad \sum_{j=1}^{N} T(\alpha_j) = N$$

Consequently, the test patterns for the set F are equivalent to minimize the following objective function:

$$\frac{1}{2}\Big[\sum_{j=1}^{N} T(\alpha_j) - N\Big]^2 \tag{7}$$

By applying the transformation's properties, this condition becomes:

$$(P_m :) \quad \frac{1}{2}\Big[\sum_{j=1}^{N}\prod_{k=1}^{m}(1 - T_{Z_k} - T_{Z_k - f_j} + 2T_{Z_k}.T_{Z_k - f_j})\Big]^2 \tag{8}$$

If we suppose that $T_{z_k} = 1$ or $0 \; \forall k$, then Problem (P_m) can also be decomposed into two subproblems (P_{m1}) and (P_{m2}):

$$\begin{aligned} (P_{m1}): \quad & min \quad f_1(x) = \frac{1}{2}\Big[\sum_{j=1}^{N}\prod_{k=1}^{m} T_{Z_{k-f_j}}\Big]^2 \\ S.T: \quad & T_{Z_k} = 1 \\ & x \in B^n \end{aligned} \tag{9}$$

and

$$\begin{aligned} (P_{m2}): \quad & min \quad f_2(x) = \frac{1}{2}\Big[\sum_{j=1}^{N}\prod_{k=1}^{m}(1 - T_{Z_{k-f_j}})\Big]^2 \\ S.T: \quad & T_{Z_k} = 0 \\ & x \in B^n \end{aligned} \tag{10}$$

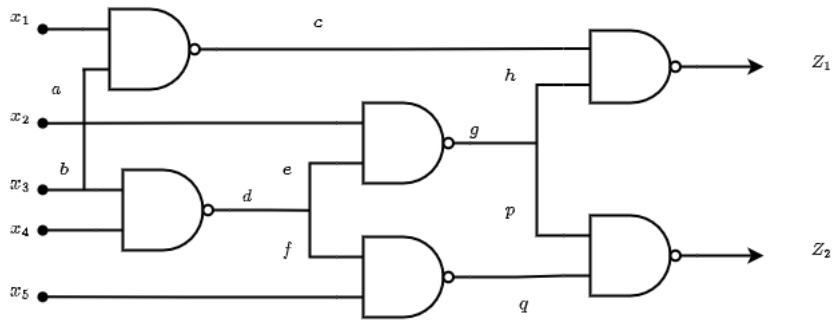

Figure 2: A Simple example circuit with Fault list - F

Note that if we put m = 1, we find the formulation given above for single-output. To illustrate that, we give an example:

Example. *Consider the benchmark Circuit C17 in Fig.2*

For this five inputs, two outputs circuit, test pattern for fault $c-0$ can be found as follows:

- Determine the function and its real transformation:

$$Z_1 = x_1.x_3 + x_2.\overline{x_3}.\overline{x_4}$$

$$T_{Z_1} = x_2 + x_1.\overline{x_2}.x3 - \overline{x_1}.x_2.x_3.x_4$$

$$Z_2 = \overline{x_3}.\overline{x_4}(x_2 + x_5)$$

$$T_{Z_2} = x_2 + \overline{x_2}.x_5 - x_2.x_3.x_4 - \overline{x_2}.x_3.x_4.x_5$$

- Apply the fault $c-0$:

$$T_{Z_{1_{c-0}}} = 1$$

$$T_{Z_{2_{c-0}}} = T_{Z_2}$$

Since $T_{Z_{1-c-0}} = 1$, the test pattern is generated by the problem (P_{m_2}). Note, that in this case the objective function f_2 is equal to 0, thus it suffices to look for any feasible point that

verifies the constraints). For example, we can choose $x^* = (0000X)$ where X is don't care.

4 Conclusion

In this paper, we have proposed a new method for generating test patterns for combinational circuits. Our approach is based on a pseudo-boolean minimization. To make the problem easy to solve we have shown that the general problem can be decomposed into two sub-problems easy to solve, such that the union of their solution represents the solution we are looking for. The examples show that by the this decomposition we guarantee a minimal and complete test set for all irredundant faults. In future work, we plan to implement our method in order to be able to test large combinational circuits and compare it with the known ATPGs.

References

[1] IBM ILOG Cplex. "V12. 1: User's Manual for CPLEX". In: *International Business Machines Corporation* 46.53 (2009), p. 157.

[2] Fujiwara and Shimono. "On the Acceleration of Test Generation Algorithms". In: *IEEE Transactions on Computers* C-32.12 (1983), pp. 1137–1144. DOI: 10.1109/TC.1983.1676174.

[3] Goel. "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits". In: *IEEE Transactions on Computers* C-30.3 (1981), pp. 215–222. DOI: 10.1109/TC.1981.1675757.

[4] T. Larrabee. "Test pattern generation using Boolean satisfiability". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 11.1 (1992), pp. 4–15. DOI: 10.1109/43.108614.

[5] P.N. Marinos. “Derivation of Minimal Complete Sets of Test-Input Sequences Using Boolean Differences”. In: *IEEE Transactions on Computers* C-20.1 (1971), pp. 25–32. DOI: 10.1109/T-C.1971.223077.

[6] Papaioannou. “Optimal Test Generation in Combinational Networks by Pseudo-Boolean Programming”. In: *IEEE Transactions on Computers* C-26.6 (1977), pp. 553–560. DOI: 10.1109/TC.1977.1674880.

[7] I. Pomeranz and S.M. Reddy. “The multiple observation time test strategy”. In: *IEEE Transactions on Computers* 41.5 (1992), pp. 627–637. DOI: 10.1109/12.142689.

[8] J. Rajski and J. Vasudevamurthy. “The testability-preserving concurrent decomposition and factorization of Boolean expressions”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 11.6 (1992), pp. 778–793. DOI: 10.1109/43.137523.

[9] J. Paul Roth, Willard G. Bouricius, and Peter R. Schneider. “Programmed Algorithms to Compute Tests to Detect and Distinguish Between Failures in Logic Circuits”. In: *IEEE Transactions on Electronic Computers* EC-16.5 (1967), pp. 567–580. DOI: 10.1109/PGEC.1967.264743.

[10] M.H. Schulz, E. Trischler, and T.M. Sarfert. “SOCRATES: a highly efficient automatic test pattern generation system”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 7.1 (1988), pp. 126–137. DOI: 10.1109/43.3140.