PROTEST: A TOOL FOR PROBABILISTIC TESTABILITY ANALYSIS

Hans-Joachim Wunderlich

Universität Karlsruhe
Institut für Informatik IV
(Prof. Dr. Detlef Schmid)
D-7500 Karlsruhe
Federal Republic of Germany

Abstract

The CAD-tool PROTEST (Probabilistic Testability Analysis) is presented. PROTEST estimates for each fault of a combinational circuit its detection probability which can be used as a testability measure. Moreover it calculates the number of random test patterns which must be generated in order to achieve the required fault coverage.

It is also demonstrated that the fault coverage will increase and the necessary number of random patterns will drastically decrease, if each primary input is stimulated by test patterns having specific probabilities of being logical "1". PROTEST uses this fact and determines for each input the optimal signal probability for a randomly generated pattern.

1.0 Introduction

The test of digital systems by random patterns makes it possible to dispense with the generation of test patterns from a description of the circuit structure. Automatic test pattern generation (ATPG) is one of the most expensive tasks in VLSI-design. The costs are high even if scan-paths, scan-sets, LSSD or similar "design for testability" methods are used (see [EiWi77], [MuSa81]). These methods reduce ATPG for arbitrary digital systems to ATPG for combinational circuits. But the size of the resulting combinational circuits in VLSI exceed the capacity of the available on general purpose computers. Test generation for integrated circuits with 10^4 - 10^6 transistors requires a computing time which can be counted by magnitudes of days [Goel81].

An alternative to ATPG is the test with randomly generated patterns. This can be done either by using additional test hardware on the chip for self testing or by an external test.

In order to do self test all storing components of the chip are usually configured as one or more feedback shift registers during testing [Much81]. In this special testmode these registers generate pseudo-random patterns for the combinational part of the circuit and evaluate and compress the responses by signature analysis [HeLe83].

An external random test is carried out by shifting randomly generated patterns in a scan path.

Both external and self test reduce the test problem to the test of combinational circuits. The application of random patterns requires the determination of the necessary test length in order to get the desired fault coverage of the commonly used stuck-at fault model on the gate level. This problem is solved by the procedure called PROTEST (Probabilistic Testability Analysis). It estimates signal probabilities and fault detection probabilities in combinational circuits. They are defined as follows:

Signal and fault detection probabilities:

Let I be the set of the primary inputs of the circuit S. During testing each input $i \in I$ is stimulated by a randomly generated signal, which is logical "1" with probability p_i. These probabilities determine a tupel of boolean random variables $T := \langle \underline{p}_i \mid i \in I \rangle$ with $P(\underline{p}_i = 1) = p_i$.

The signal probability p_k of a node k of S is the probability, that k is logical "1" if the circuit is stimulated by a sequence implementing T.

The fault detection probability p_f of a fault f is the probability that this fault is detected, if the circuit is stimulated by a sequence implementing T.

The computation of the fault detection probability can be reduced to the computing of signal probabilities (see

0738-100X/85/0204$01.00 © 1985 IEEE

sect. 3). For combinational circuits without reconvergent fan-out V. and P. Agrawal proposed an algorithm calculating the signal probabilities [AgAg75].

All known algorithms for the general case show exponential complexity. Recently the author has shown that computing the signal probabilities is NP-hard [Wu84]. Therefore the exact determination of signal probabilities is not an efficient test tool. Consequently the tool PROTEST tries to estimate such probabilities only and works with nearly linear effort. A similar tool -STAFAN - was presented by Jain and Agrawal [AgJa84], which extrapolates such probabilities from runs of logic simulation. Savir et al. proposed a method to determine upper and lower bounds of the signal probabilitiy of a node [BDS84], - PROTEST however computes a real number as estimation of both signal and fault detection probability.

The input for PROTEST is a description of a combinational circuit. The output is:

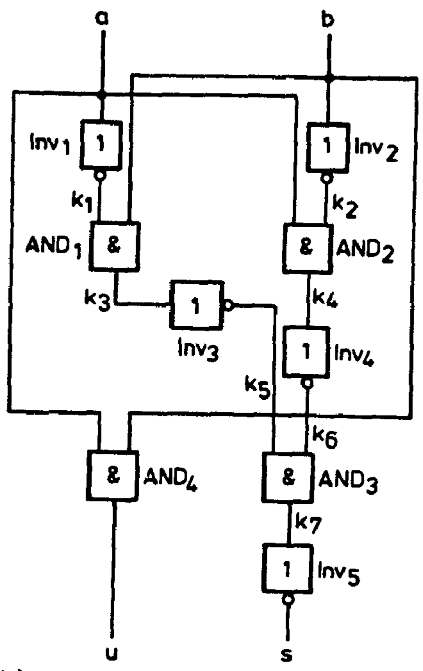
- an estimation of the signal probabiliy at each node for a given tupel of signal probabilities at each primary input;
- an estimation of the fault detection probability of each fault for a given tupel of signal probabilities at each primary input;
- the number of patterns which must be generated to reach a required fault coverage with a desired confidence.
- a tupel of signal probabilities for the primary inputs which maximizes the fault detection probabilities of the faults in the circuit;
- random pattern sets, which are logical "1" at each primary input with the optimized signal probability calculated above;
- results of a static fault simulation with these patterns.

In this paper we present the algorithms and results of PROTEST.

2.0 The estimation of the signal probabilities

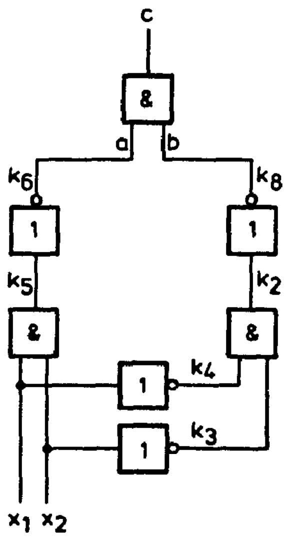
PROTEST accepts combinational circuits with arbitrary boolean functions as basic components. To simplify the notation in this paper only inverters and 2-input ANDs are used.

Let $S := \langle I,O,K,B \rangle$ be a combinational circuit with the set of primary inputs I



$I = \{a,b\}$
$O = \{u,s\}$
$K = \{a,b,k_1,k_2,k_3,k_4,k_5,k_6,k_7,u,s\}$
$B = \{Inv_1,Inv_2,Inv_3,Inv_4,Inv_5,AND_1,$
$AND_2,AND_3,AND_4\}$

Fig. 1: The notation for a combinational circuit.



$V(a,b) = \{x_1,x_2\}$
x_1 has the immediate successors k_4,k_5
x_2 has the immediate successors k_3,k_5

Fig. 2: Joining points.

and the set of primary outputs O. They are subsets of the set K of all nodes. B is the set of the logic components (see fig. 1).

For two nodes $x,y \in K$ the set of joining points $V(x,y)$ consists of those nodes $k \in K$, which have at least two immediate successors, one of them is on a path to x and the other is on a path to y (see fig. 2).

If the node c is an output of an AND with the inputs a and b, then there is a reconvergent fan-out at c if and only if $V(a,b) \neq \{\}$.

Let $\langle \underline{p}_i \mid i \in I \rangle$ be a tupel of boolean random variables, one for each primary input. This tupel induces boolean random variables $\underline{p}_k$ for each node $k \in K$. In order to determine the signal probability $p_k = P(\underline{p}_k = 1)$ we must distinguish 4 cases:

1) $k \in I$:

The signal probability for each primary input is given.

2) k is output of an inverter with the input a.

Then we have $p_k = 1 - p_a$

3) k is output of an AND with the input a and b and with $V(a,b) = \{\}$.

Then $p_k = p_a * p_b$.

4) k is output of an AND with the inputs a and b and with $V(a,b) \neq \{\}$.

For each subset $v \subseteq V := V(a,b)$ we define the boolean formula

$$A^V_v := \prod_{x \in v} x \prod_{y \in V-v} \overline{y}.$$

Each of those formulas represents one instance of the logical values at the nodes in V. Let $P(\underline{p}_a | A^V_v)$ be the conditional probability of a being logical "1" if A^V_v is true. Then we have

(1)
$$p_k = p^V_k :=$$
$$\sum_{v \subseteq V} p_{A^V_v} * P(\underline{p}_a | A^V_v) * P(\underline{p}_b | A^V_v).$$

The main effort in determining p_k arises in case 4, because here the number of summands increases exponentially with the cardinality of V(a,b).

But an estimated value s_k of p_k is achieved by restricting the evaluation of formula (1) to a bounded subset $W \subseteq V$ with cardinality $|W| \leq n$. Then we have

(2)
$$s_k = p_k^W :=$$
$$\sum_{v \subseteq W} p_{A^W_v} * P(\underline{p}_a | A^W_v) * P(\underline{p}_b | A^W_v)$$

In order to get a small error $|p_k - p_k^W|$ we have to select appropriate elements for $W \subseteq V$. Let us introduce the notations for the standard deviation

$$S(\underline{p}_a) := \sqrt{p_a(1-p_a)}$$

and for the covariance

$$\mathrm{Cov}(\underline{p}_a, \underline{p}_b) := P(\underline{p}_a \& \underline{p}_b) - p_a p_b.$$

For every logical formula A a formula indexed with A, e.g.

$$S_{|A}(\underline{p}_a) := \sqrt{P(\underline{p}_a | A)(1 - P(\underline{p}_a | A))}$$

is defined as the restriction to the case of A being true.

With an arbitrary enumeration $\langle k_1, \ldots, k_m \rangle$ of V, with the sets

$$V_0 := \{\},\ V_{i+1} := V_i \quad \{k_{i+1}\}$$

and with

$$d_{i+1} := p_k^{V_{i+1}} - p_k^{V_i},\ i = 0, \ldots m-1,$$

we have for r>s:

$$p_k^{V_r} - p_k^{V_s} = \sum_{j=s+1,\ldots,r} d_j.$$

For $j := 0, \ldots, m-1$ and with $A^{\{\}}_t$ the following equation holds:

$$d_{j+1} = \sum_{t \subseteq V_j} p_{A_t^{V_j} * k_{j+1}} *$$
$$P(\underline{p}_a | A_t^{V_j} * k_{j+1}) * P(\underline{p}_b | A_t^{V_j} * k_{j+1}) +$$
$$p_{A_t^{V_j} * \overline{k}_{j+1}} * P(\underline{p}_a | A_t^{V_j} * \overline{k}_{j+1}) *$$
$$P(\underline{p}_b | A_t^{V_j} * \overline{k}_{j+1}) -$$
$$p_{A_t^{V_j}} * P(\underline{p}_a | A_t^{V_j}) *$$
$$P(\underline{p}_b | A_t^{V_j}).$$

By means of Bayes' formula and some straightforward transformations one achieves

$$d_{j+1} = \sum_{t \underline{c} V_j} p_{A_t}{}^{V_j} * \mathrm{Cov}(\underline{p}_a, \underline{p}_{k_{j+1}}) * \mathrm{Cov}(\underline{p}_b, \underline{p}_{k_{j+1}}) / S(\underline{p}_{k_{j+1}}) | A_t{}^{V_j 2}.$$

The difference between the total estimation error $p_k{}^V - p_k{}^{\{\}}$ and the error which arises if we take the joining point x into consideration $p_k{}^V - p_k{}^{\{x\}}$ is therefore

$$\mathrm{Cov}(\underline{p}_a, \underline{p}_x)\mathrm{Cov}(\underline{p}_b, \underline{p}_x)/S(\underline{p}_x)^2.$$

An appropriate strategy to select a subset $W \underline{c} V$ is to minimize the value of

$$\left|\sum_{x \in V-W} \mathrm{Cov}(\underline{p}_a, \underline{p}_x)\mathrm{Cov}(\underline{p}_b, \underline{p}_x) \,/\, S(\underline{p}_x)^2\right|.$$

Indeed there are some special cases, where the optimality of this strategy is provable [Wu84]. But in general this procedure is a heuristic to estimate signal probablities and to estimate the possible error in estimation.

This procedure is part of PROTEST and requires the parameters MAXVERB, determining the maximal cardinality of W, and MAXLIST, determining the maximal length of the paths in which joining points are searched.

3.0 The fault detection probability

The computation of fault detection probabilities can be transformed into the computation of signal probabilities in a rather trivial way. But this yields quadratic complexity and therefore it is not appropriate for all applications.

What is indeed possible with linear complexity is the estimation of the probability of <u>single path sensitizing</u> to an output. A test pattern sensitizes a single path from a pin x of some logical component of the circuit to a primary output o, if there is exactly one path from x to o, in which the logical value at each node depends from the value at x. The detection probability of a stuck-at-i, i=0,1, fault at x can be estimated by the probability that x has the value "NOT(i)" and that at the same time a single path is sensitized. This can be reduced to the calculation of signal probabilities too. This method still needs a considerable computing time.

But for many applications the modeling of the signal flow is sufficent. In this case the transformation of the additional boolean formulas into new parts of the circuit is avoided.

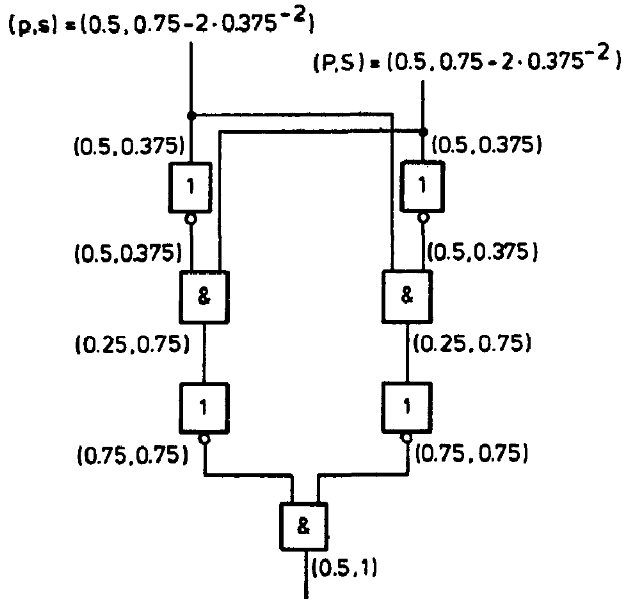
Each component of the circuit represents a boolean function $f: \{0,1\}^n \text{ --> } \{0,1\}$. All those functions can be mapped in a unique way into arithmetic functions $f: [0,1]^n \text{ --> } [0,1]$ by the transformations $x \text{ |--> } 1-x$ and $x\&y \text{ |--> } x*y$. Now for each pin x of a component we define a function s(x), which represents the probability that there exists a sensitized path from x to a primary output. Let x be the output pin of f, $x_1,..,x_m$ the input pins of other components which are directly connected to x, and $e_1,..,e_n$ the input pins of f.

We define the associative operation $z ¢ y := z+y-2zy$. Then we set

$s(x) := s(x_1) ¢ \ldots ¢ s(x_m)$ and

$s(e_i) := s(x) * (f(p_{e_1},..p_{e_{i-1}},0,..,p_{e_n}) ¢ f(p_{e_1},..p_{e_{i-1}},1,..,p_{e_n}))$.

An alternative model for circuits with a large number of primary outputs is $s(x) := 1 - (1-p_{x_1})*..*(1-p_{x_m})$.



<u>Fig. 4:</u> Signal modeling

Both alternatives are implemented in PROTEST and they determine fault detection probabilities by $x0 := p_x s(x)$ and $x1 := (1-p_x)s(x)$.

The accuracy of these procedures is sufficient for all applications described. Moreover PROTEST offers also the option to estimate the probability of single path sensitization.

4.0 The validity of PROTEST

Agrawal and Mercer describe in [AgMe82] how they transformed the results of the testability measure SCOAP into values called P_{SCOAP} corresponding to the fault detection probability. For each fault they determined also the variable P_{SIM} which is the quotient of the number of patterns detecting the fault and the number of applied test patterns. The investigations in [AgMe82] show that there is only a correlation 0.4 between P_{SCOAP} and P_{SIM} even for pure combinational circuits.

For each fault f the value of $P_{PROT}(f)$ is the estimated detection probability by PROTEST. The variables P_{PROT} and P_{SIM} however correlate with P_{SIM} with more than 0.9. These correlations were obtained with more than 10 circuits. Two examples are the TTL ALU SN74181 - called ALU - and the circuit MULT, which computes A + B + C * D for 8 bit wide data. MULT is built with 1568 gate equivalents according to the proposal of [Hart80].

Table 1 shows for MAXVERB = 4 and MAXLIST =100 the maximal value M_{err}, which has been reached by the formula $|P_{PROT}-P_{SIM}|$ for some fault f.

$$A_{err} := \sum_{f} |P_{PROT}-P_{SIM}| / (\text{Number of faults})$$

is the average difference between the real values obtained by simulation and those estimated by PROTEST. C_o is the correlation coefficient of P_{PROT} and P_{SIM}.

	M_{err}	A_{err}	C_o
ALU	0.15	0.04	0.97
MULT	0.48	0.11	0.90

Table 1: Maximal and average errors and correlations

Figure 5 shows the correlation diagram for the ALU, where each fault is positioned according to its values of P_{PROT} and P_{SIM}.

Figure 6 is the analogue for MULT. One easily notices that in general P_{SIM} is higher than P_{PROT}.

This bias is caused by the very simple modeling of the signal flow, which does not take into account that sensitization of several paths at the same time may also result in fault detection. The under-estimation of the fault detection probability has consequences for computing test lengths.

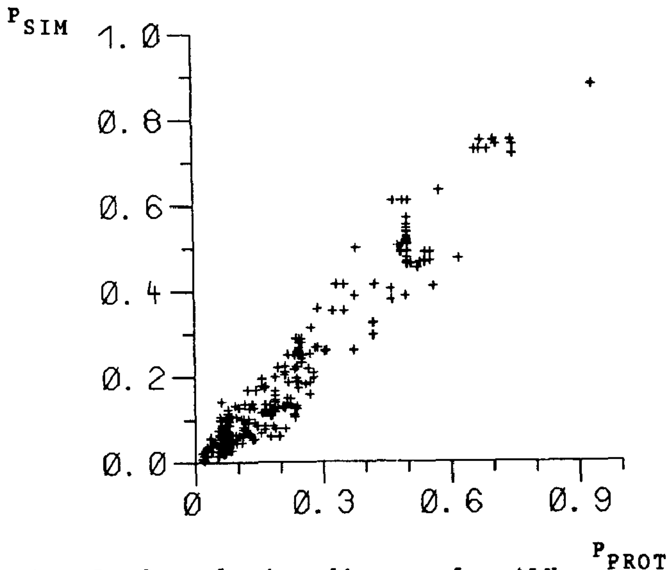


Fig. 5: Correlation diagram for ALU

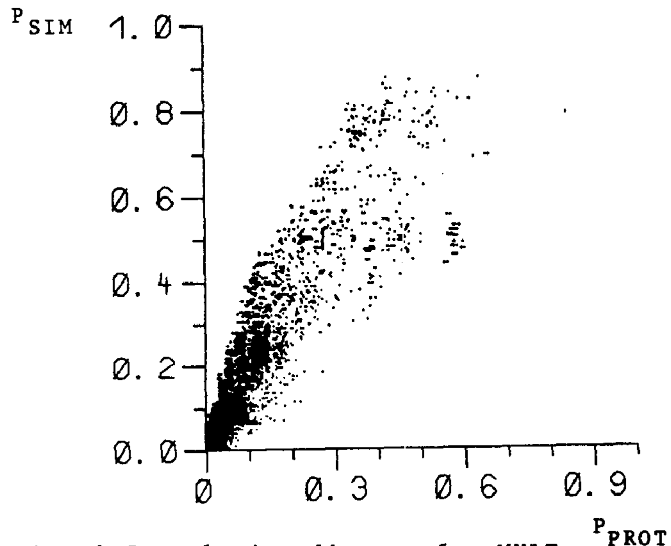


Fig. 6 Correlation diagram for MULT

5.0 Necessary testlengths

Let F be a set of stuck-at faults. For the sake of simplicity we assume that fault detection is statistically independent, - in other words, the probability p_f that a pattern detects f is the same, whether this pattern detects some other faults too or not.

The probability that n patterns detect all faults in F is

$$(3) \quad p_F := \prod_{f \in F} (1-(1-p_f)^N).$$

For a given p_F this formula can be used to approximate N by an iteration very fast. The necessary number of patterns N to detect all faults in F with probability p_F requires only knowledge of the fault detection probability p_f. But unfortunately there are only estimations s_f for each p_f available. These estimations may result in a systematic over-estimation of p_F. For compensation in [AgJa84] a weighting factor is proposed. PROTEST does not need such a factor, because its estimations were systematically higher than p_f in all investigated examples.

PROTEST also uses formula 3 to solve the following problem: F consists of all faults of the circuit, for each d $\in$ [0,1] F_d are the d*100 % faults with highest detection probilities and e is the required probability that a pattern set detects whole F_d. Then PROTEST determines the necessary size N of the pattern set. Table 2 shows the results for MULT and ALU:

	d	e	N
ALU	0.98	0.98	212
MULT	0.98	0.98	962

Table 2: Size of test sets

Several random pattern sets of the required size were created for both the ALU and the MULT. With all those sets fault simulation had reached a coverage of 99,9 - 100 %.

There are known circuits which can be very poorly tested by random patterns. Some of those circuits were also analyzed by PROTEST. DIV is the combinatorial part of a 16 bit divider and COMP is the connection of 16 slightly modified SN7485 comparators to a cascaded 24 bit word comparator (Fig. 7).

In order to test those circuits PROTEST proposed a number of test patterns listed in table 3.

d	e	N(DIV)	N(COMP)
1.0	0.95	499 960	292 808 220
	0.98	614 590	355 083 821
	0.999	966 967	556 622 443
0.98	0.95	491 827	247 522 478
	0.98	608 900	309 063 047
	0.999	965 591	510 127 655

Table 3: Size of test sets

These large pattern sets cause random pattern testing to become uneconomical. But by modifying the pattern sets slightliy, the number of patterns can be reduced drastically.

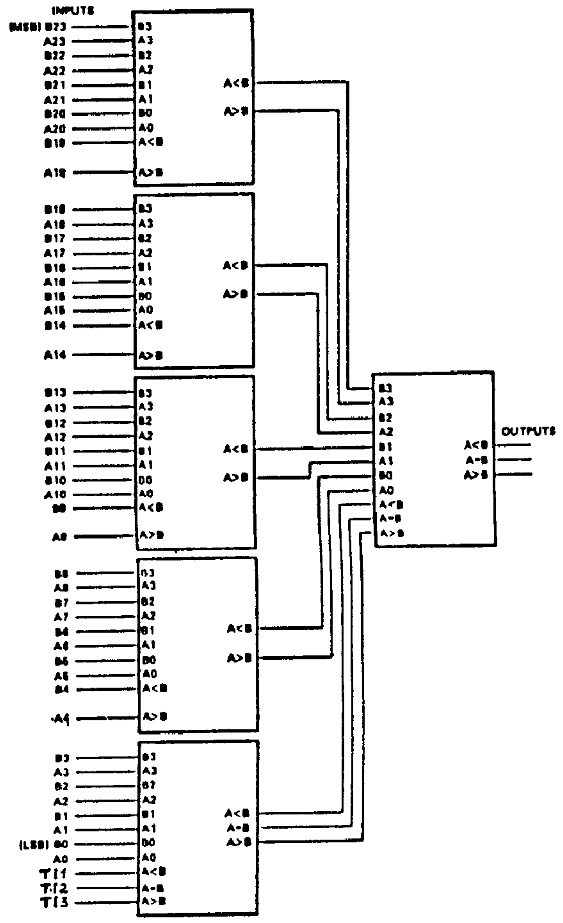


Fig. 7: 24 bit comparator COMP

6.0 Optimized input signal probabilities

In Table 3 the large number of pattern sets are calculated under the assumption that the primary inputs of the circuit are all stimulated by logical "1" with the probability $p_i = 0.5$. But an arbitrary tupel $X := \langle p_i \mid i \in I \rangle \in [0,1]^I$ determines pattern sets which have an arbitrary probability $p_i \in [0,1]$. X determines a probability $p_f(X)$ of detection of each fault f.

With some natural number N

$$J_N(X) := \prod_f (1-(1-p_f(X))^N)$$

is an estimation of the probability that N realizations of X detect the whole F. For every combinational circuit $J_N : [0,1]^I \longrightarrow [0,1]$ is a real function which can be maximized by a tupel $\langle p_i \mid i \in I \rangle$, thus leading to maximal fault detection. N is only a numerical parameter.

PROTEST includes an optimizing procedure, which finds a local maximum of J_N. The procedure works according to the hill climbing principle also used in many other fields [Nils80]. Table 4 shows the optimal signal probabilities PROTEST proposed for each primary input of COMP.

A0	0.63	B0	0.56	A1	0.69	B1	0.75
A2	0.38	B2	0.38	A3	0.25	B3	0.31
A4	0.13	B4	0.13	A5	0.94	B5	0.88
A6	0.88	B6	0.88	A7	0.88	B7	0.88
A8	0.88	B8	0.94	A9	0.94	B9	0.94
A10	0.88	B10	0.88	A11	0.88	B11	0.94
A12	0.88	B12	0.88	A13	0.88	B13	0.94
A14	0.94	B14	0.94	A15	0.94	B15	0.94
A16	0.88	B16	0.88	A17	0.94	B17	0.94
A18	0.94	B18	0.88	A19	0.94	B19	0.94
A20	0.94	B20	0.88	A21	0.94	B21	0.88
A22	0.94	B22	0.94	A23	0.94	B23	0.88
TI1	0.63	TI2	0.63	TI3	0.63		

Table 4: Optimized signal probabilities at the primary inputs

It is remarkable how much the optimal input probabilities differ from the conventionally used value of 0.5. Table 5 shows the number of patterns which are necessary using these probabilities.

d	e	N(DIV)	N(COMP)
1.0	0.95	6 066	8 932
	0.98	6 969	10 284
	0.999	10 063	14 911
0.98	0.95	5 097	6 828
	0.98	5 780	7 767
	0.999	8 052	10 893

Table 5: The necesarry size of optimized test sets.

The testlength using the optimized input signal probabilities was reduced by several orders of magnitude. This result was validated by fault simulation.

Each circuit has been simulated with two random pattern sets of size 12 000. One of the random pattern sets stimulated each primary input with logical "1" with the commonly used probability of 0.5. The other pattern set was created according to the optimized signal probabilities. Table 6 shows for both pattern sets, how the fault coverage increases with the number of used pattern.

For both circuits fault simulation shows that conventional random pattern test yields very insufficient results whereas the pattern sets proposed by PROTEST detect nearly all faults.

Pattern	DIV		COMP	
count	not optim.	optim.	not optim.	optim.
10	16.8	26.1	32.1	44.5
100	56.5	66.3	70.4	72.7
1000	69.1	94.6	75.8	95.4
2000	71.4	98.5	76.5	97.2
3000	73.2	99.0	77.2	98.3
4000	74.7	99.1	79.6	99.4
5000	76.8	99.1	80.0	99.4
6000	77.2	99.4	80.4	99.4
7000	77.2	99.4	80.4	99.5
8000	77.2	99.6	80.5	99.5
9000	77.2	99.7	80.5	99.5
10000	77.2	99.7	80.6	99.7
11000	77.2	99.7	80.6	99.7
12000	77.2	99.7	80.7	99.7

Table 6: Fault detection by simulation of random patterns

7.0 Implementation

PROTEST consists of different programs written in standard PASCAL. They compile a structure description language for circuits, they generate test pattern sets and they carry out the algorithms described above. It runs on a SIEMENS 7561 computer, a machine with approximately 2.4 MIPS. The performance of the analysis program is dependent on both the size of the circuit and the number of joining points.

In table 7 the CPU time and the resulting size of a test set are listed for some circuits. The transistor count for these circuits is based on a CMOS library.

Transistor count	estimated size of a test set	CPU time (seconds)
368	594	0.4
1 274	1 798 000	0.7
2 496	1 120 000 000	1.0
26 450	32 160	23.0
47 936	8 284 000	41.0

Table 7: CPU time for the analysis by PROTEST

The optimization of the input signal probabilities is more CPU intensive. Here the effort depends on the number of the primary inputs, too.

Table 8 shows the CPU time needed to optimize the input signal probabilities.

Transistor count	Inputs	optim. test set	CPU-time
368	11	267	6.4
1 274	32	6 264	49.0
2 496	48	43 010	152.0
26 450	32	1 778	2181.0

Table 8: CPU time for the optimization by PROTEST

8.0 Some applications

The optimal input signal probabilities calculated by PROTEST are used to design non-linear feedback shift registers (NLFSR), which generate such optimal pattern sequences [KuWu84]. Similar the wellknown BILBO [Much81] the NLFSR are used for self-testing. Such an NLFSR reaches a higher fault detection probability in shorter test time, generating minimal hardware overhead compared to the standard BILBO.

This self testing strategy is used in the Karlsruhe synthesis system CADDY (Carlsruher Digital Design System) [CKR84],[CKR84a],[SCHD84]. It uses several design for testability features like conventionel BILBOs and NLFSR optionally. When using BILBOs the system gets the necessary testlength from PROTEST. When using NLFSR it also obtains the optimal input signal probability.

The use of PROTEST also reduces the computing time of ordinary ATPGs for combinational circuits or circuits containing scan features. Most ATPG first use fault simulation by random patterns, and second when this becomes inefficient, they use other procedures like the D-algorithm. Computing time for fault simulation is drastically reduced by using optimized pattern sets. (For the optimized patterns the calculation of table 7 needed only a quarter of the computing time needed for the not optimized patterns.). Additionally the number of faults which are to be treated by the more expensive second procedure decreases.

9.0 Summary:

The CAD tool PROTEST has been presented. The testability measure given by PROTEST is the random pattern testability of combinational circuits. It finds less testable faults, computes the random pattern test length and determines an optimized input signal probability for each primary input.
This optimization leads to a drastic reduction of the necessary random pattern set. The results were all validated by fault simulation. The design system at the university of Karlsruhe integrates PROTEST as the key tool to achieve design for testability.

References:

[AgAg75] P. Agrawal, V.D. Agrawal; Probabilistic Analysis of Random Test Generation Method for Irredundant Combinational Logic Networks; IEEE Trans. on Comput., Vol. C-24, No. 7, July 1975

[AgJa84] S.K. Jain, V.D. Agrawal; STAFAN: An Alternative to Fault Simulation; Proc. of 21st Design Automation Conference, 1984

[AgMe82] Agrawal, V. D., Mercer, M. R.; Testability Measures - What Do They Tell Us; International Test Conference, 1982, pp. 391 - 396

[BDS84] J. Savir, G.S. Ditlow, P.H. Bardell; Random Pattern Testability; IEEE, Trans. Comp., Vol. C-33, No. 1, Jan. 1984

[CKR84a] R. Camposano, A. Kunzmann, W. Rosenstiel; Automatic Data Path Synthesis from DSL Spezifications; IEEE, Proc. of Int. Conf. on Computer Design, ICCD, 1984

[EiWi77] E.B. Eichelberger, T.W. Williams; A logic design structure for LSI testability; Proc. 14th Design Automation Conference, pp. 462-468, June 1977

[Goel81] P. Goel; An implicit enumeration algorithm to generate tests for combinational logic circuits; IEEE Trans. on Comp., Vol. C-30, no. 3, March 1981

[Hart80] R.W. Hartenstein; VLSI-Bausteine in geringen Stueckzahlen fuer Spezialanwendungen; Elektronische Rechenanlagen, Heft 4, 1980

[HeLe83] J.H. Heckmaier, D. Leisengang; Fehlererkennung mit Signaturanalyse; Elektron. Rechenanl. 25, 1983, H. 3, 109 - 116

[KuWu84] A. Kunzmann, H.-J. Wunderlich; Steigerung der Effizienz beim Test mit Zufallsmustern Report 19/84 at the Faculty for Informatics, University of Karlsruhe, October 1984

[Much81] J. Mucha; Hardware Techniques for Testing VLSI Circuits Based on Built-In Test; Proc. COMPCON 81, Feb. 1981

[MuSa81] E.I. Muehldorf, A.D. Savkar; LSI Logic Testing - An Overview; IEEE Trans. on Computers, Vol. C-30, No.1, January 1981

[Nils80] N.J. Nilsson; Principles of Artificial Intelligence Tioga Publishing Company, Palo Alto, California, 1980;

[SCHD84] D. Schmid et al.; Automatischer Entwurf hochintegrierter Schaltungen aus Beschreibungen der Schaltungsfunktion; Inforamtik-Fachberichte 88, 14. GI-Jahrestagung, Braunschweig Okt. 1984, Springer-Verlag