Generating Pseudo-Exhaustive Vectors for External Testing

S. Hellebrand, H.-J. Wunderlich, O. F. Haberl

Institute of Computer Design and Fault Tolerance
(Prof. Dr.-Ing. D. Schmid)
University of Karlsruhe
P. O. Box 6980, D-7500 Karlsruhe
F. R. Germany

Abstract

In the past years special chips for external test have been successfully used for random pattern testing. In this paper a technique is presented to combine the advantages of such a low cost test with the advantages of pseudo-exhaustive testing, which are an enhanced fault coverage and a simplified test pattern generation.

To achieve this goal two tasks are solved. Firstly, an algorithm is developed for pseudo-exhaustive test pattern generation, which ensures a feasible test length. Secondly, a chip design for applying these test patterns to a device under test is presented. The chip is programmed by the output of the presented algorithm and controls the entire test. The technique is first applied to devices with a scan path and then extended to sequential circuits. A large number of benchmark circuits have been investigated, and the results are presented.

Keywords: pseudo-exhaustive test, built-off test, external low cost test.

1. Introduction

Using built-in self-test (BIST) techniques, a high speed test can be performed without an expensive automatic test equipment, and the test equipment on chip can be used for the fault diagnosis of the system. Significant hardware savings are obtained, if the self-test circuitry is partly implemented externally on a special chip, which performs test control, test pattern generation and evaluation. The basic structure of such a chip is shown in figure 1.

Such an external test chip for generating random patterns was proposed in [3, 8, e.g.]. Higher fault coverages and shorter test times are possible using weighted random patterns; pertinent external test systems were presented in [20, 22]. Practical experience has shown that they allow faster testing than a conventional method. If they are mounted on board together with the circuit to be tested, they can also be used for fault diagnosis later on.

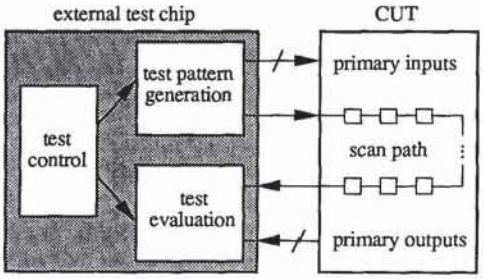


Figure 1: Special chip for a low cost external test (CUT denotes circuit under test).

In this paper, a chip for generating pseudo-exhaustive test patterns is presented. The pseudo-exhaustive test of combinational circuits has been proposed to enhance fault coverage and to simplify test pattern generation [13, 14]. For each output o of a combinational circuit C the cone C(o) is the minimal subcircuit containing all predecessors of o. A pseudo-exhaustive test set for C is a set of test patterns which includes an exhaustive test set for each cone C(o). Within the cones all combinationally faulty functions are detected. A pseudo-exhaustive test set is significantly smaller than an exhaustive test set, if the cones are sufficiently small. To guarantee this property, design algorithms are available [e.g. 11, 15, 17]. The pseudo-exhaustive test of a sequential circuit can be obtained via a pseudo-exhaustive test of an equivalent combinational circuit [24].

1990 International Test Conference CH2910-6/0000/0670$01.00 © 1990 IEEE

In order to combine the advantages of external low-cost testing and pseudo-exhaustive testing, two tasks have been solved:

1) An algorithm has been developed to generate pseudo-exhaustive test sets and to encode them into a compact representation.

2) A chip has been designed and fabricated, which can be programmed with the information obtained in step 1) and generates the corresponding pseudo-exhaustive test set.

The algorithm of step 1) determines groups of cones that can be exhaustively tested in parallel. A pseudo-exhaustive test is provided by the successive test of these test groups. As two arbitrary cones can always be simultaneously tested, we obtain $m \cdot 2^{w-1}$ as an upper bound for the size of the test set. Here m denotes the number of outputs of the circuit and w the maximum number of inputs of the cones. For most conventional pseudo-exhaustive BIST approaches, only smallest upper bounds bigger than $m \cdot 2^w$ can be derived [10].

For each group the test patterns are generated by a switch-matrix which is fed by a w-bit counter. The information generated in step 1) is used to control the switch-matrix. It is stored in an embedded RAM (see fig. 2). The results generated by the new algorithm of step 1) can also be used for the multiplexer or reconfigurable counter design proposed in [19], but by using the programmable chip of figure 2 we get a more flexible solution.

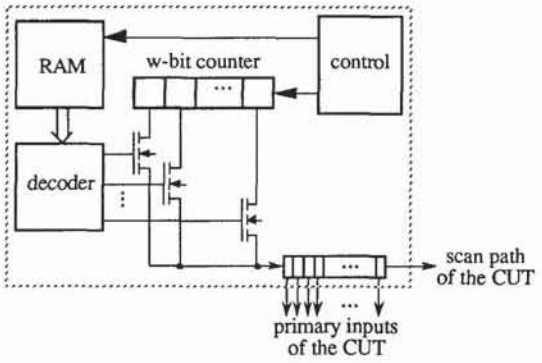


Figure 2: Basic structure of the external test chip.

In section 2 we will start with a formal definition of a pseudo-exhaustive test set. Then the notion of simultaneously testable cones will be made more precise in section 3. As the problem to find a minimal number of test groups will turn out to be np-complete, a fast heuristic will be developed. Subsequently in section 4 we will describe the design of the proposed test chip in more detail. In section 5 we will show, how this approach can be extended to the pseudo-exhaustive test of sequential circuits. Finally we will present some experimental results. It will be demonstrated that, together with the appropriate segmentation tools, our approach yields an effective pseudo-exhaustive test for all well-known ISCAS-benchmark circuits [6].

2. Basic definitions and facts

In the following we only consider circuits with a complete scan path. The extension to sequential circuits is discussed in section 5.

For the purpose of pseudo-exhaustive pattern generation it is sufficient to know for each output the set of inputs it depends on. In [14] this information is represented by a matrix, the so-called dependence matrix. We describe the input-output relationship by a cover of the set of inputs.

Definition 1: Let C be a combinational circuit with a set I of inputs and a set O of outputs. For each $o \in O$ let I(o) denote the set of inputs of the cone C(o). The sets I(o) form a cover $\mathcal{I} := (I(o))_{o \in O}$ of I, i.e. $\bigcup_{o \in O} I(o) = I$. $\mathcal{I}$ is called the *characteristic cover* of C.

For the example circuit of figure 3 the characteristic cover is $\mathcal{I} = (\{1,2,3\}, \{2,3,4\}, \{1,4,5\}, \{1,4,6\})$.

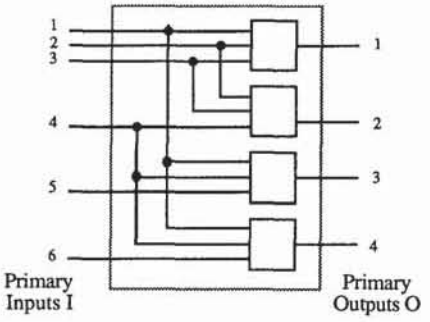


Figure 3: Example circuit.

A characteristic cover $\mathcal{I}$ describes the class of circuits having the same input-output relationships. All circuits of this class can be tested by the same pseudo-exhaustive test. Therefore in the following we define pseudo-exhaustive tests for characteristic covers $\mathcal{I} := (I(o))_{o \in O}$. It is convenient to consider test sets for $\mathcal{I}$ as subsets $T \subset \mathbb{F}_2^I$. Here $\mathbb{F}_2$ denotes the Galois-field of order

2, and for a finite set M we define the vector space $\mathbb{F}_2^M := \{(x_i)_{i\in M} \mid \forall\ i\in M\ x_i\in\mathbb{F}_2\}$. For $M := \{1,...,m\} \subset \mathbb{N}$ we write $\mathbb{F}_2^m$ instead of $\mathbb{F}_2^M$.

A pseudo-exhaustive test set $T \subset \mathbb{F}_2^I$ for $\mathcal{I}$ must include an exhaustive test set for each set I(o). So for each $o \in O$ we have to consider those components of elements in T corresponding to I(o).

Definition 2: Let I be a finite set and $J \subset I$. The mapping $pr_J : \mathbb{F}_2^I \to \mathbb{F}_2^J$, $pr_J(x)_j := x_j$ for $j \in J$, is called the projection from $\mathbb{F}_2^I$ onto $\mathbb{F}_2^J$.

With definition 2 a set $T \subset \mathbb{F}_2^I$ includes an exhaustive test set for I(o), if the projection $pr_{I(o)} : T \to \mathbb{F}_2^{I(o)}$ is surjective. Thus we can define a pseudo-exhaustive test set for $\mathcal{I}$ as follows:

Definition 3: Let $\mathcal{I} := (I(o))_{o\in O}$ be a characteristic cover. A set $T \subset \mathbb{F}_2^I$ is called a pseudo-exhaustive test set for $\mathcal{I}$, if for all $o \in O$ the projections $pr_{I(o)} : T \to \mathbb{F}_2^{I(o)}$ are surjective.

We can immediately construct a pseudo-exhaustive test set $T \subset \mathbb{F}_2^I$ for $\mathcal{I}$, if we define for each $o \in O$ an injective mapping $e_{I(o)} : \mathbb{F}_2^{I(o)} \to \mathbb{F}_2^I$ by

$$e_{I(o)}(x)_i := \begin{cases} x_i & \text{if } i \in I(o) \\ 0 & \text{else} \end{cases}$$

and set $T := \bigcup_{o\in O} e_{I(o)}(\mathbb{F}_2^{I(o)})$. Obviously the projections $pr_{I(o)} : T \to \mathbb{F}_2^{I(o)}$ are surjective by construction of T. This construction can be implemented as a successive test of the sets I(o), e.g. using reconfigurable counters [19]. The size of T is estimated by $2^w \le |T| \le |O|\cdot 2^w$ for $w := \max_{o\in O}|I(o)|$.

To reduce the length of a pseudo-exhaustive test, several techniques of "pattern compaction" have been developed [e.g. 1, 12, 14]. The problem of finding a pseudo-exhaustive test set of minimal size is np-complete [18].

The starting point of the presented approach is the idea of "partitioning dependence matrices" suggested in [14]. Therefore we briefly sketch the underlying concepts: The set I of inputs is partitioned into subsets, which can share the same signal during the test. For the example circuit of figure 3 this is illustrated by figure 4.

Inputs 2 and 6 of the example circuit share the same test signal, and a pseudo-exhaustive test set T is produced by a 5-bit counter. In the following this is made more precise.

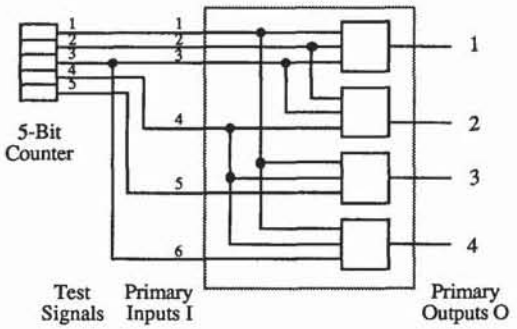


Figure 4: Circuit inputs sharing the same test signal.

Definition 4: Let $\mathcal{I} := (I(o))_{o\in O}$ be a characteristic cover. A set $J \subset I$ is *compatible with* $\mathcal{I}$, if for all $i,j \in J$, $i \neq j$, there is no set I(o) which contains both i and j.

If we have a partition $\mathcal{Z} := (Z_\rho)_{1\le\rho\le r}$ of I into r pairwise disjoint sets compatible with the characteristic cover $\mathcal{I}$, the correspondence between test signals and inputs is described by a mapping $\varphi : I \to \{1,...,r\}$, such that $|\varphi(I(o))| = |I(o)|$ holds for all $o \in O$. Conversely we have the following lemma.

Lemma 1: Let $\mathcal{I} := (I(o))_{o\in O}$ be a characteristic cover. If there is a mapping $\varphi : I \to \{1,...,r\}$, such that $|\varphi(I(o))| = |I(o)|$ for all $o \in O$, then there exists a partition $\mathcal{Z} := (Z_\rho)_{1\le\rho\le r}$ of I into r sets compatible with the characteristic cover.

Proof: The desired partition into r sets is defined by $Z_\rho := \{i \in I \mid \varphi(i) = \rho\}$, $1\le\rho\le r$.

For test pattern generation it is sufficient to consider the *reduced characteristic cover* $\mathcal{I}_\varphi := (\varphi(I(o)))_{o\in O}$ instead of $\mathcal{I}$.

Theorem 1: Let $\mathcal{I} := (I(o))_{o\in O}$ be a characteristic cover, and let $\varphi : I \to \{1,...,r\}$ be a mapping with $|\varphi(I(o))| = |I(o)|$ for all $o \in O$. If $T \subset \mathbb{F}_2^r$ is a pseudo-exhaustive test set for the reduced characteristic cover $\mathcal{I}_\varphi = (\varphi(I(o)))_{o\in O}$ and if $e : \mathbb{F}_2^r \to \mathbb{F}_2^I$ is defined by $e(x)_i := x_{\varphi(i)}$, then $e(T) \subset \mathbb{F}_2^I$ is a pseudo-exhaustive test set for $\mathcal{I}$.

Proof: By construction.

For the example circuit of figure 3 the mapping $\varphi : \{1,...,6\} \to \{1,...,5\}$, defined by $\varphi(i) := i$ for $i = 1,...,5$ and $\varphi(6) := 3$, satisfies the condition of lemma 1 and provides a partition of I into 5 sets compatible with the characteristic cover. The reduced characteristic cover is $\mathcal{I}_\varphi := (\{1,2,3\}, \{2,3,4\}, \{1,4,5\}, \{1,4,2\})$. A pseudo-exhaustive test set for $\mathcal{I}_\varphi$ can be generated

by a 5-bit counter. A pseudo-exhaustive test set for $\mathcal{I}$ according to theorem 1 is obtained, if the counter-stages are connected to the inputs I as shown in figure 4. However, the mapping φ does not provide the smallest possible r. It can be shown that a partition into 4 sets compatible with $\mathcal{I}$ exists, too.

The construction of theorem 1 even provides a pseudo-exhaustive test set for a larger class of circuits.

Definition 5: Let $\mathcal{I} := (I(o))_{o \in O}$ be a characteristic cover, and let $\mathcal{I}_\varphi$ be the reduced characteristic cover according to $\varphi : I \to \{1,...,r\}$. Then $EXP(\mathcal{I}_\varphi) := \{A \subset I \mid |A| = |\varphi(A)| \wedge \exists B \in \mathcal{I}_\varphi\ A \subset \varphi^{-1}(B)\}$ is called the *expanded characteristic cover*.

It can be shown that $\mathcal{I} \subset EXP(\mathcal{I}_\varphi)$, but in general we have $\mathcal{I} \neq EXP(\mathcal{I}_\varphi)$ [23]. By construction the set $e(T) \subset \mathbb{F}_2^I$ derived in theorem 1 is also a pseudo-exhaustive test set for $EXP(\mathcal{I}_\varphi)$.

In order to minimize the test length, the number r of necessary test signals must be minimized. The corresponding decision problem can be stated as follows:

Problem CP (Compatible Partitioning):
Let $\mathcal{I} := (I(o))_{o \in O}$ be a characteristic cover, $r \in \mathbb{N}$. Is there a mapping $\varphi : I \to \{1,...,r\}$, such that $|\varphi(I(o))| = |I(o)|$ holds for all $o \in O$?

Hirose and Singh have shown that this is an np-complete problem [12]. Their proof is a reduction of the graph-coloring problem [9]. For the solution of CP they suggested an exact algorithm as well as a fast heuristic. Another heuristic was proposed in [2].

3. An algorithm for pseudo-exhaustive test generation

If the set of inputs I can be partitioned into r subsets compatible with the characteristic cover $\mathcal{I} := (I(o))_{o \in O}$, only a simple r-bit counter is required to generate a pseudo-exhaustive test set for $\mathcal{I}$. Unfortunately, if it is not possible to keep r below a certain limit (e.g. $r \leq 20$), this technique is no longer feasible, because the test time increases exponentially with r. This problem can be overcome, if the cover $\mathcal{I}$ is divided into groups, such that each group can be tested by a w-bit counter ($w \geq \max_{o \in O} |I(o)|$).

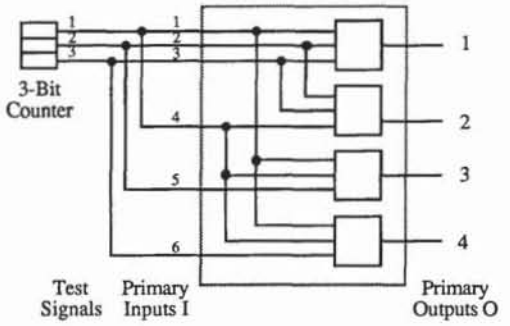
The algorithm to be presented in the sequel identifies groups of input sets, such that the size of the overall test set, which is obtained by a successive test of the groups, is kept small. First we state exactly, when a group of input sets can be tested by a w-bit counter:

Definition 6: Let $\mathcal{I} := (I(o))_{o \in O}$ be a characteristic cover, and let $P \subset O$, $w \in \mathbb{N}$. The system of input sets $\mathcal{I}_P := (I(o))_{o \in P}$ is w-testable, if there is a mapping $\pi_P : I \to \{1,...,w\}$, such that $|\pi_P(I(o))| = |I(o)|$ for all $o \in P$.

For $\mathcal{I}_P$ we write $I_P := \bigcup \mathcal{I}_P$. Considering the characteristic cover of the circuit shown in figure 3, $P := \{1,2\}$ provides a 3-testable group of input sets $\mathcal{I}_{\{1,2\}}$. The function $\pi_P : \{1,...,6\} \to \{1,2,3\}$ can be defined by

$$\pi_P(i) := \begin{cases} i & \text{for } 1 \leq i \leq 3 \\ i - 3 & \text{for } 4 \leq i \leq 6 \end{cases}$$

If $\mathcal{I}_P$ is w-testable, then by theorem 1 we obtain a pseudo-exhaustive test set T for $\mathcal{I}_P$ as the image of $\mathbb{F}_2^w$ under the injective mapping $e_P : \mathbb{F}_2^w \to \mathbb{F}_2^{I_P}$, $e_P(x_i) := x_{\pi_P(i)}$. $\mathbb{F}_2^w$ can be generated by a w-bit counter, and $T = e_P(\mathbb{F}_2^w)$ is applied to $\mathcal{I}_P$ if the counter outputs and the circuit inputs I_P are connected corresponding to π_P. In figure 5 this is demonstrated for the outputs $P = \{1,2\}$ of the example circuit of figure 3:



1	2	3	4	5	6	
1	1	1	1	1	1	
1	1	0	1	1	0	
1	0	1	1	0	1	
1	0	0	1	0	0	
0	1	1	0	1	1	
0	1	0	0	1	0	
0	0	1	0	0	1	
0	0	0	0	0	0	

T

Figure 5: Test generation for a group of input sets.

As an immediate consequence we have the following lemma:

Lemma 2: Let $\mathcal{I} := (I(o))_{o \in O}$ be a characteristic cover, and let $w \in \mathbb{N}$. Furthermore let the set of outputs be partitioned into g subsets $P_1,...,P_g$ such that $\mathcal{I}_{P_\gamma}$ is w-testable for all $\gamma = 1,...,g$. With π_γ denoting the corresponding mapping $I \to \{1,...,w\}$, an injective mapping $e_\gamma : \mathbb{F}_2^w \to \mathbb{F}_2^I$ is defined by $e_\gamma(x)_i := x_{\pi_\gamma(i)}$ for each $\gamma = 1,...,g$.
Then $T := \bigcup_{1 \le \gamma \le g} e_\gamma(\mathbb{F}_2^w) \subset \mathbb{F}_2^I$ is a pseudo-exhaustive test set for $\mathcal{I}$.

Proof: By construction.

Hence a pseudo-exhaustive test can be generated by a w-bit counter if for each group of w-testable input sets the connections between the counter outputs and the inputs I are adapted corresponding to the mapping π_γ. A self-test configuration based on this fact was presented in [19]. The new test generation algorithm to be presented can be used for an efficient design of this structure. The size of the generated test set is $g \cdot 2^w$. Hence we have to reduce the number g of w-testable groups of cones, and solve the following decision problem:

Problem MCP (Multiple Compatible Partitions):
Let $\mathcal{I} := (I(o))_{o \in O}$ be a characteristic cover. Furthermore let $w \ge \max_{o \in O} |I(o)|$ and $g \in \mathbb{N}$. Can O be partitioned into g subsets $P_1,...,P_g$, such that $\mathcal{I}_{P_\gamma}$ is w-testable for all $\gamma = 1,...,g$?

For $g = 1$ this problem specializes to problem CP treated in section 2. As a consequence we have

Theorem 2: Problem MCP is np-complete.

Because of the complexity of problem MCP we refrain from an exact solution. Instead of that we suggest a fast heuristic, which iteratively constructs groups of w-testable cones. It is based on the following theorem:

Theorem 3: Let $\mathcal{I} := (I(o))_{o \in O}$ be a characteristic cover, $w \in \mathbb{N}$. Let $P \subset O$ be a subset of outputs, such that $\mathcal{I}_P$ is w-testable, and let $p^* \in O \setminus P$ with $|I(p^*)| \le w$. If there exists an $o^* \in P$, such that $I(p^*) \cap I_P \subset I(o^*)$, then $\mathcal{I}_{P \cup \{p^*\}}$ is w-testable, too.

Proof: Let $\pi : I \to \{1,...,w\}$ be a mapping, such that $|\pi(I(o))| = |I(o)|$ for each $o \in P$. We construct a mapping $\tilde{\pi} : I \to \{1,...,w\}$ as follows:

i) For $i \in I_P$ we set $\tilde{\pi}(i) := \pi(i)$.

ii) Let $i \in I(p^*)$. For $i \in I(p^*) \cap I_P$ the mapping $\tilde{\pi}$ is already defined and since $I(p^*) \cap I_P \subset I(o^*)$, we have $|I(p^*) \setminus I_P| = |I(p^*)| - |I(p^*) \cap I_P| = |I(p^*)| - |\tilde{\pi}(I(p^*) \cap I_P)| \le w - |\tilde{\pi}(I(p^*) \cap I_P)|$.
Thus we can extend $\tilde{\pi}$ to a mapping $\tilde{\pi} : I_{P \cup \{p^*\}} \to \{1,...,w\}$ with $|\tilde{\pi}(I(p^*))| = |I(p^*)|$.

iii) For $i \in I \setminus I_{P \cup \{p^*\}}$ we set $\tilde{\pi}(i) := 1$. qed.

Theorem 3 provides a simple criterion to decide whether a set $I(p^*)$ can be added to a w-testable group of input sets without disturbing w-testability. For instance, let $P := \{3,4\}$ in the example circuit and set $p^* := \{2\}$. We have $I_P := \{1,4,5,6\}$ and $I(p^*) := \{2,3,4\}$. Since $I_P \cap I(p^*) = \{4\} \subset I(3)$, and since $\mathcal{I}_{\{3,4\}}$ is 3-testable, $\mathcal{I}_{\{2,3,4\}}$ is 3-testable, too. The algorithm of figure 6 uses the criterion of theorem 3 to determine a w-testable system $\mathcal{I}_P$ of input sets.

```
procedure w_testable_group (I,w,I_P)
P := ∅;
choose p* with |I(p*)| = max_{o∈O} |I(o)|;
P := {p*}; O := O \ {p*};
while p* ∈ O exists with I(p*) ∩ I_P ⊂ I(o*)
                    for o* ∈ P and |I(p*)| ≤ w do
   choose such a p* with minimal |I(p*) ∪ I_P|;
   P := P ∪ {p*}; O := O \ {p*};
end;
```

Figure 6: Procedure to determine a w-testable $\mathcal{I}_P \subset \mathcal{I}$.

The input data required are the characteristic cover $\mathcal{I}$ and the value for $w \ge \max_{o \in O} |I(o)|$. In each step we choose p^* such that $|I(p^*) \cap I_P|$ is minimal. Thus we try to maximize the remaining degrees of freedom in succeeding steps of the algorithm. The algorithm of figure 7 uses this procedure to divide the characteristic cover into w-testable groups.

```
procedure w_testable_groups (I,w,g,P)
w := max|I(o)|;
g := 0; P := ∅;
while I ≠ ∅ do
   g := g+1;
   w_testable_group(I,w,I_{P_g});
   P := P ∪ {P_g};
   I := I \ I_{P_g};
end;
```

Figure 7: Algorithm to identify groups of w-testable input sets.

Input of the procedure is the characteristic cover $\mathcal{I}$. The value of w is adjusted to the smallest possible value. The procedure returns the required number g of groups, and a list $\mathcal{P}$ containing the corresponding subsets $P_\gamma \subset O$.

Applied to the example circuit of figure 3 the above algorithm proceeds as follows. The variable w is set to $\max_{o \in O} |I(o)| = 3$. For g=1 first p*= 1 is chosen. As $I(1) \cap I(2) \subset I(1)$ and $|I(1) \cup I(2)|$ is smaller than $|I(1) \cup I(3)|$ and $|I(1) \cup I(4)|$, in the next step p*= 2 is chosen. Since $I(3) \cap I_{\{1,2\}} = I(4) \cap I_{\{1,2\}} = \{1,4\} \not\subset I(1)$ and $\{1,4\} \not\subset I(2)$ the first group of 3-testable input sets cannot be enlarged. For g = 2 the sets I(3) and I(4) are collected in a group and the algorithm stops. The result is a partition into 2 w-testable groups of input sets $\mathcal{I}_{\{1,2\}}$ and $\mathcal{I}_{\{3,4\}}$.

The number g determined by this procedure always satisfies the inequality $g \leq \lceil \frac{|O|}{2} \rceil$, because of a corollary of theorem 3: For two arbitrary outputs p,q ∈ O the system of input sets $\mathcal{I}_{\{p,q\}}$ is w-testable, if $\max\{|I(p)|,|I(q)|\} \leq w$. The procedure requires an effort of $O(|O|^3)$.

Additionally, the complexity of the data to be handled can be reduced using theorem 1. To determine a partition $\mathcal{Z}$ of I into sets compatible with the characteristic cover $\mathcal{I} := (I(o))_{o \in O}$ the heuristic proposed in [12] is used and referred to as "procedure heuristic_CP($\mathcal{I}$,$\mathcal{I}_\varphi$)". Figure 8 shows the corresponding algorithm to identify groups of w-testable input sets.

```
procedure w_testable_groups_2 (I,w,g,P)
w := max|I(o)|;
g := 0; P := ∅;
while I ≠ ∅ do
   g := g+1;
   heuristic_CP(I,I_φ);
   if |⋃ I_φ| ≤ w then
      P := P ∪ {O}; I := ∅;
   else
      w_testable_group(I_φ,I_{P_g});
      P := P ∪ {P_g}; I := I \ EXP(I_φ); O := O \ P_g;
end;
```

Figure 8: Algorithm making use of theorem 1.

Applied to the example circuit of figure 3, for g = 1 the algorithm first determines a reduced characteristic cover $\mathcal{I}_\varphi$ of $\mathcal{I}$. If the cover $\mathcal{I}_\varphi$ derived in section 2 is obtained as a result, the procedure w_testable_group is called with $\mathcal{I}_\varphi := (\{1,2,3\}, \{2,3,4\}, \{1,4,5\}, \{1,4,2\})$. In the first step p*= 1 is chosen. In the second step either 2 or 4 can be chosen for p*, because $|I(1) \cup I(4)| = |I(1) \cup I(2)| < |I(1) \cup I(3)|$. If p*= 2 is chosen, $I_{\{1,4\}} = \{1,2,3,4\}$ and the w-testable group can be enlarged by I(3). But it is not possible to add I(2) to this group. For g = 2 the only remaining set is I(2). The result of the algorithm are the two w-testable groups $\mathcal{I}_{\{1,3,4\}}$ and $\mathcal{I}_{\{2\}}$.

To provide all the information which is needed to generate a pseudo-exhaustive test set for a characteristic cover $\mathcal{I}$, the basic algorithms of figures 6 and 7 must be extended. For each P_γ the corresponding mappings π_γ must be determined. In the procedure w_testable_group of figure 6, π_γ is initialized with $\pi_\gamma(i) := 1$ for all $i \in I$, and in each step of the iterative construction of $\mathcal{I}_{P_\gamma}$ the following procedure is called immediately after the choice of a suitable p*:

```
procedure update_π_γ (P,p*,π_γ)
I* := I(p*) \ I_P; k := |I*|;
choose an enumeration {i_1,...,i_k} = I*;
W* := {1,...,w} \ π(I(p*) ∩ I_P); ℓ := |W*|;
choose an enumeration {j_1,...,j_ℓ} = W*;
for κ := 1 to k do
   π_γ(i_κ) := j_κ;
end;
```

Figure 9: Procedure which updates the mapping π_γ

If the algorithm of figure 8 is used, the required information is described by $\pi_\gamma' := \pi_\gamma \cdot \varphi$.

With these data, a pseudo-exhaustive test set can be constructed according to lemma 2. A possible representation of the data required for test generation is a matrix $TG = (t_{\gamma i})$ with entries $t_{\gamma i} := \pi_\gamma(i)$ ($\pi_\gamma'(i)$ respectively). TG is called the *test generation matrix* of the characteristic cover $\mathcal{I}$.

This way pseudo-exhaustive test sets were derived for a large number of example circuits, in particular for all the ISCAS-benchmark circuits. In all the cases, only a small number g of groups and moderate test lengths were required. The results are discussed in section 6.

4. Design of the external test chip

The required information for test generation can be represented in the compact form of a test generation matrix TG. The information can be stored in an embedded RAM and this way can be used to control the test generation done by a w-bit counter. In the following we describe the design in more detail. The basic components are a RAM, a register to store the number g of w-testable groups of input sets and the number of primary inputs and scan elements, a w-bit counter, a switch matrix controlled by a decoder, a shift register and a test control unit. We omit to describe the process of test evaluation, which is carried out by a signature analysis register SA as seen in figure 10.

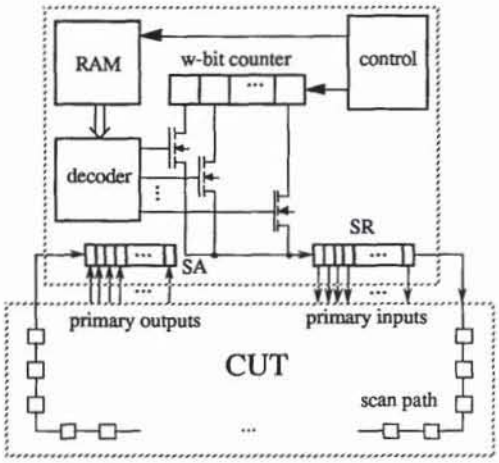


Figure 10: External pseudo-exhaustive test by the developed chip.

As explained above, the RAM contains the test generation matrix TG = $(t_{\gamma i})$. Each row corresponds to a w-testable group of input sets. The test control unit is a finite state machine realizing the following test schedule: For the test of each group of w-testable sets, the w-bit counter cycles through all possible 2^w states. For each state of the counter the primary inputs and the scan path elements of the CUT must be supplied with the contents of the corresponding counter stages. For an input i the element $t_{\gamma i} = \pi_\gamma(i)$ is chosen by the decoder, and the contents of the $t_{\gamma i}$-th stage of the counter are shifted into the shift-register SR. Starting with the scan path elements this is done successively for all inputs. The serial output of the shift-register SR is connected to the scan data input (SDI) of the CUT and the stages of SR are connected to the primary inputs of the CUT.

The test pattern is applied to the CUT, when all bits have been shifted into the scan path. SR then contains the pattern to be applied to the primary inputs.

The counter-overflow is used to indicate that the test of a group of input sets is finished. The test is completed when all the g w-testable groups have been treated. Figure 11 summarizes the test schedule in a short algorithmic description. The set I of inputs of the CUT is regarded as the union PI ∪ PPI of primary inputs (PI) and pseudo-primary inputs (PPI) corresponding to the scan path elements.

```
for γ := 1 to g do
  for c := 0 to 2^w-1 do
    for i ∈ PPI do
      shift the contents of the π_γ(i)-th counter stage into SR;
    for i ∈ PI do
      shift the contents of the π_γ(i)-th counter stage into SR;
    while the scan path is not yet completely loaded do
      shift SR;
    activate the system clock of the CUT;
    switch to the next state of the counter;
end.
```

Figure 11: Test schedule for the external pseudo-exhaustive test.

The procedure of figure 11 yields an overall test time which is directly proportional to $|I| \cdot g \cdot 2^w$.

A further speed-up is possible, if the scan path elements and the primary inputs of the CUT are treated simultaneously, but this requires to divide the test generation matrix into two separate matrices for primary inputs and scan elements. The test control unit becomes more complicated, because the loading of the scan path and of a shift register for the primary inputs has to be synchronized. The chip area occupied by the test control unit increases and the RAM must be made smaller. Because of this trade-off we decided to realize the basic structure shown in figure 10.

For the concrete implementation w = 16 and a 2048 × 8-bit RAM have been chosen, such that all circuits with $|I| \le 512$ and up to 8 w-testable groups of input sets can be dealt with. In order to make our approach feasible for larger circuits the RAM can be supplemented by an external RAM and the shift-register SR can be connected to an external shift-register. The testing time can be reduced by slightly modifying the test schedule. For each group of input sets belonging to a subset $P_\gamma \subset O$ of

outputs, the value $w_\gamma = \max_{o \in P_\gamma} |I(o)| \leq w$ is determined. If $w_\gamma < w$ holds, it is sufficient to cycle through the first 2^{w_γ} states of the counter in order to test the group γ.

The chip has been designed using the VENUS CAD-system for standard cell design [5].

In the next section we describe how our approach can be extended to provide a pseudo-exhaustive test for sequential circuits.

5. Extension to sequential circuits

In [24] the concept of a pseudo-exhaustive test for sequential circuits with an acyclic dataflow was developed. For the test of sequential circuits, pattern sequences have to be applied. A pseudo-exhaustive set of pattern sequences for synchronous circuits without feedback loops is derived from a pseudo-exhaustive test set for an equivalent combinational circuit, the so-called combinational representation. The maximal number of flip-flops on a path from the primary inputs to the primary outputs within the acyclic circuit C is denoted by r. Since C is acyclic, each state is reachable in at most r+1 steps, and it is sufficient to apply pattern sequences of length r+1 in order to detect a combinational fault. To construct the combinational representation of C, Roth's approach of time frames is modified, such that for each time step only a part of the circuit is copied. Thus an equivalent combinational circuit with a set $\bar{I} \subset I \times \{0,...,r\}$ of inputs and characteristic cover $\bar{\vartheta} := (\bar{I}(o))_{o \in O}$ is obtained.

Definition 7: Let C be a sequential circuit with inputs I and outputs O, and let r denote the maximal number of flip-flops on a path from the primary inputs to the primary outputs. Furthermore, let $\bar{\vartheta} := (\bar{I}(o))_{o \in O}$ be the characteristic cover of the combinational representation. A set $T \subset \mathbb{F}_2^{I \times \{0,...,r\}}$ is called a set of pseudo-exhaustive pattern sequences for C, if $\bar{T} := pr_{\bar{I}}(T) \subset \mathbb{F}_2^{\bar{I}}$ is a pseudo-exhaustive test set for $\bar{\vartheta}$.

To apply a pattern sequence $b \in \mathbb{F}_2^{I \times \{0,...,r\}}$ to C, at each time step t, $0 \leq t \leq r$, the pattern $b_t := pr_{I \times \{t\}}(b)$ has to be applied.

In order to compute a pseudo-exhaustive set of pattern sequences for a circuit C, first the characteristic cover $\bar{\vartheta}$ is determined. Then the algorithm presented in section 3 is used to determine a pseudo-exhaustive test set $\bar{T}$ for $\bar{\vartheta}$ and the corresponding test generation matrix TG. With the function

$$e : \mathbb{F}_2^{\bar{I}} \to \mathbb{F}_2^{I \times \{0,...,r\}},\ e(x)_{(i,t)} := \begin{cases} x_{(i,t)} & \text{if } (i,t) \in \bar{I} \\ 0 & \text{else} \end{cases},$$

a set $T := e(\bar{T})$ of pseudo-exhaustive pattern sequences according to definition 7 is obtained.

During test execution the elements of T must be transformed into sequences, i.e. each $b \in T$ must be divided into the patterns $pr_{I \times \{0\}}(b),...,pr_{I \times \{r\}}(b)$ to be applied successively. Since TG is computed by the methods developed in section 3, the pseudo-exhaustive test set $\bar{T}$ for $\bar{C}$ can be generated by the hardware described in section 4.

To construct T and to divide the elements of T into pattern sequences, it is necessary to store how the columns of TG correspond to inputs of C at different time steps. Furthermore, the chip control unit has to distinguish between primary inputs of C and scan elements at each time step. These straightforward extensions are also implemented into the chip reported. The test schedule for this operation mode is sketched in figure 12.

```
for γ := 1 to g do
  for c := 0 to 2^w-1 do
    for t := 0 to r do
      for all scan elements i do
        if (i,t) ∈ Ī then
          shift the contents of the π_γ(i,t)-th counter stage
          into SR
        else
          shift 0 (or the contents of an arbitrary counter
          stage) into SR;
      for all primary inputs i do
        if (i,t) ∈ Ī then
          shift the contents of the π_γ(i,t)-th counter stage
          into SR
        else
          shift 0 (or the contents of an arbitrary counter
          stage) into SR;
      while the scan path is not yet completely loaded do
        shift SR;
      activate the system clock of the CUT;
    switch to the next state of the counter;
end.
```

Figure 12: Test-schedule for sequential circuits.

6. Experimental results

To investigate the efficiency of the algorithm presented in section 3, the ISCAS c-benchmark circuits were segmented [11]. The cones of the modified circuits have at most 16 inputs. Table 1 shows the circuit characteristics. The first column contains the name of the circuit. In the second and third columns the number of inputs and outputs of the modified circuits are listed. These numbers include additional inputs and outputs for segmentation purposes, which are realized by so-called segmentation cells. These cells are part of a scan path. The number of required segmentation cells is listed in the fourth column.

circuit	# inputs	# outputs	# segmentation cells
c432.16	63	34	27
c499.16	49	40	8
c880.16	76	42	16
c1355.16	49	40	8
c1908.16	55	47	22
c2670.16	266	173	33
c3540.16	140	112	90
c5315.16	240	185	62
c6288.16	130	130	98
c7552.16	324	225	117

Table 1: Characteristics of the modified benchmark circuits (w = 16).

First we computed the upper bound $\lceil \frac{|O|}{2} \rceil \cdot 2^w$ for the number of test patterns and compared it to the lower bounds for the pseudo-exhaustive test generation algorithms suggested in [7]. The bounds were computed without reducing the characteristic cover in a preprocessing phase. The results are shown in table 2.

circuit	presented approach: upper bound	cyclic codes [7]: lower bound
c432.16	$1.11 \cdot 10^6$	$1.07 \cdot 10^9$
c499.16	$1.31 \cdot 10^6$	$1.07 \cdot 10^9$
c880.16	$1.38 \cdot 10^6$	$1.07 \cdot 10^9$
c1355.16	$1.31 \cdot 10^6$	$1.07 \cdot 10^9$
c1908.16	$1.57 \cdot 10^6$	$1.07 \cdot 10^9$
c2670.16	$5.70 \cdot 10^6$	$1.07 \cdot 10^9$
c3540.16	$3.67 \cdot 10^6$	$1.07 \cdot 10^9$
c5315.16	$6.09 \cdot 10^6$	$1.07 \cdot 10^9$
c6288.16	$4.26 \cdot 10^6$	$1.07 \cdot 10^9$
c7552.16	$7.41 \cdot 10^6$	$1.07 \cdot 10^9$

Table 2: Comparison of lower bounds for the number of test patterns.

For all circuits the upper bound for the presented approach is better than the lower bound for the technique using cyclic codes suggested in [7]. For the method combining linear feedback shift register and shift registers (LFSR/SR) described in [4] only the trivial lower bound 2^w is known, which also holds for the presented approach. Therefore no real comparison was possible and we also computed the smallest upper bounds for the techniques suggested in [4] and [7]. Table 3 shows the results. In column 2 the bounds computed for the presented test generation algorithm are listed again. Column 3 contains the bounds according to [4] and column 4 contains the upper bound derived in [7].

circuit	presented approach	LFSR/SR [4]:	cyclic codes [7]:
c432.16	$1.11 \cdot 10^6$	$5.37 \cdot 10^8$	$4.13 \cdot 10^{12}$
c499.16	$1.31 \cdot 10^6$	$2.68 \cdot 10^8$	$4.13 \cdot 10^{12}$
c880.16	$1.38 \cdot 10^6$	$5.37 \cdot 10^8$	$6.64 \cdot 10^{13}$
c1355.16	$1.31 \cdot 10^6$	$2.68 \cdot 10^8$	$4.13 \cdot 10^{12}$
c1908.16	$1.57 \cdot 10^6$	$5.37 \cdot 10^8$	$4.13 \cdot 10^{12}$
c2670.16	$5.70 \cdot 10^6$	$8.59 \cdot 10^8$	$5.67 \cdot 10^{15}$
c3540.16	$3.67 \cdot 10^6$	$2.15 \cdot 10^9$	$7.14 \cdot 10^{14}$
c5315.16	$6.09 \cdot 10^6$	$8.59 \cdot 10^9$	$7.14 \cdot 10^{14}$
c6288.16	$4.26 \cdot 10^6$	$2.15 \cdot 10^9$	$7.14 \cdot 10^{14}$
c7552.16	$7.41 \cdot 10^6$	$8.59 \cdot 10^9$	$5.67 \cdot 10^{15}$

Table 3: Comparison of smallest upper bounds for the number of test patterns.

Then we applied the algorithm of figure 6 to the modified benchmark circuits. The results are listed in table 4.

circuit	number of test groups	resulting number of test patterns (presented approach)	resulting number of test patterns ([7])
c432.16	4	$2.63 \cdot 10^5$	$3.61 \cdot 10^{11}$
c499.16	4	$2.63 \cdot 10^5$	$1.47 \cdot 10^{11}$
c880.16	4	$2.63 \cdot 10^5$	n. a.
c1355.16	4	$2.63 \cdot 10^5$	$1.47 \cdot 10^{11}$
c1908.16	5	$3.28 \cdot 10^5$	$3.61 \cdot 10^{11}$
c2670.16	4	$2.63 \cdot 10^5$	n.a.
c3540.16	10	$6.55 \cdot 10^5$	n.a.
c5315.16	9	$5.90 \cdot 10^5$	n.a.
c6288.16	6	$3.93 \cdot 10^5$	n.a.
c7552.16	5	$3.28 \cdot 10^5$	n.a.

Table 4: Results obtained by the algorithm of figure 6 and by the method of [7].

Since suitable tools were not available, not all of the numbers for the other mentioned test generation techniques could be computed. Only for the smaller examples we could determine the smallest possible dimension of an appropriate cyclic code using the code-table in [16].

The resulting test times for all c-benchmark circuits using the presented test chip with a 15 MHz clock are listed in table 5.

circuit	test time [sec]	circuit	test time [sec]
c432.16	1.12	c2670.16	4.67
c499.16	0.87	c3540.16	6.16
c880.16	1.35	c5315.16	9.48
c1355.16	0.87	c6288.16	3.43
c1908.16	1.22	c7552.16	2.86

Table 5: Test time using the presented external test chip.

7. Conclusion

An algorithm and a circuit have been presented for external testing by pseudo-exhaustive patterns. The algorithm divides the cones of the device under test into subsets, which are simultaneously testable. For each group of cones, a minimal pseudo-exhaustive test set can be generated. The union of these test sets leads to a global pseudo-exhaustive test set, the size of which is distinctly smaller than the test lengths derived by the procedures presented earlier.

The circuit described can be programmed by the output of the algorithm and generates the corresponding pseudo-exhaustive test set. It is able to control the entire test application for a device under test if a (partial) scan path is incorporated. Hence the advantages of external testing known for weighted random patterns are combined with the benefits of a pseudo-exhaustive test, for instance a higher and guaranteed fault coverage.

References

[1] Akers, S.B.: On the Use of Linear Sums in Exhaustive Testing; in: Proc. Int. Symposium on Fault Tolerant Computing, 1985, pp. 148-153

[2] Amazonas, J.R.; Strum, M.: Pseudoexhaustive Test Techniques: A New Algorithm to Partition Combinational Networks; in: Proc. 1st European Test Conference, Paris 1989, pp. 392 - 397

[3] Bardell, P.H.; McAnney, W.H.: Self-testing of multichip logic modules; in: Proc. 1982 IEEE Test Conference, pp. 200-204

[4] Barzilai, Z., Coppersmith, D., Rosenberg, A.L.: Exhaustive Generation of Bit Patterns with Applications to VLSI Self-Testing; in: IEEE Trans. on Computers, Vol. c-32, No. 2, February 1983, pp. 190-194

[5] Brenneisen, H.: Entwicklung und Implementierung von Verfahren zur Durchführung pseudo-erschöpfender Tests mithilfe anwendungsspezifischer Schaltungen; Diploma thesis, University of Karlsruhe, 1990

[6] Brglez, F. et al.: Accelerated ATPG and fault grading via testability analysis; in: Proc. IEEE International Symposium on Circuits and Systems, June 1985, Kyoto

[7] Chen, C.L.: Exhaustive Test Pattern Generation Using Cyclic Codes; in: IEEE Trans. on Comp., Vol. 37, No. 2, February 1988, pp. 225 - 228

[8] Eichelberger, E.B.; Lindbloom, E.: Random-Pattern Coverage Enhancement and Diagnosis for LSSD Logic Self-Test; in: IBM J. Res. Develop., Vol. 27, No. 3, May 1983

[9] Garey, M.R.; Johnson, D.S.: Computers and Intractability, A Guide to the Theory of NP-Completeness; W. H. Freeman and Company, San Francisco 1979

[10] Hellebrand, S.: Synthese vollständig testbarer Schaltungen. Ph. D. thesis, University of Karlsruhe, 1990

[11] Hellebrand, S.; Wunderlich, H.-J.: Tools and Devices Supporting the Pseudo-Exhaustive Test; in: Proc. 1st European Design Automation Conference, Glasgow, March 1990

[12] Hirose, F.; Singh, V.: McDDP, A Program for Partitioning Verification Testing Matrices; CRC Technical Report No. 81-13, Stanford, July 1982

[13] McCluskey, E.J.; Bozorgui-Nesbat, S.: Design for Autonomous Test; in IEEE Trans. on Circuits and Systems, Vol. Cas-28, No. 11, November 1981, pp. 1070-1078

[14] McCluskey, E.J.: Verification Testing - A Pseudoexhaustive Test Technique; in: IEEE Trans. on Computers, Vol. c-33, No.6, June 1984, pp. 541-546

[15] Patashnik, O.: Circuit Segmentation for Pseudo-Exhaustive Testing; CRC Technical Report No. 83-14, Stanford, October 1983

[16] Peterson, W.W.; Weldon, E.J.,Jr.: Error-Correcting Codes; MIT Press, Cambridge, Massachusetts, London 1972

[17] Roberts, M.W., Lala, M.Sc.: An Algorithm for the Partitioning of logic circuits; in: IEE Proceedings, Vol. 131,Pt.E., No.4, July 1984, pp. 113-118

[18] Seroussi, G., Bshouty, N. H.: Vector Sets for Exhaustive Testing of Logic Circuits; in: IEEE Trans. on Information Theory, Vol. 34, No. 3, May 1988, pp. 513-522

[19] Udell, J.G.,Jr.: Reconfigurable Hardware for Pseudoexhaustive Test; in: Proc. IEEE International Test Conference, Washington 1988, pp. 522 - 530

[20] Waicukauski, J.A.; Lindbloom, E.; et al.: WRP: A Method for Generating Weighted Random Patterns; in: IBM Journ. of Research and Development, Vol. 33, No. 2, March 1989, pp. 149-161

[21] Wang, L.T.; McCluskey, E.J.: Circuits for Pseudo-Exhaustive Test Pattern Generation; in: Proc. IEEE International Test Conference, 1986, pp. 25-37

[22] Wunderlich, H.-J.: Multiple Distributions for Biased Random Test Patterns; in: Proc. IEEE International Test Conference, Washington 1988, pp. 236-244

[23] Wunderlich, H.-J.: Rechnergestützte Verfahren für den prüfgerechten Entwurf und Test hochintegrierter Schaltungen; Springer-Verlag Berlin Heidelberg New York London Paris Tokyo, 1990

[24] Wunderlich, H.-J.; Hellebrand, S.: The Pseudo-Exhaustive Test of Sequential Circuits; in: Proc. IEEE Int. Test Conference, Washingtion 1989, pp. 19-27