The Pseudoexhaustive Test of Sequential Circuits

Hans-Joachim Wunderlich, *Associate Member, IEEE*, and Sybille Hellebrand, *Associate Member, IEEE*

Abstract—**The concept of a pseudoexhaustive test for sequential circuits is introduced in a way similar to that which is used for combinational networks. Using partial scan all cycles in the data flow of a sequential circuit are removed, such that a compact combinational model can be constructed. Pseudoexhaustive test sequences for the original circuit are constructed from a pseudoexhaustive test set for this model. To make this concept feasible for arbitrary circuits a technique for circuit segmentation is presented which provides special segmentation cells as well as the corresponding algorithms for the automatic placement of the cells. Example circuits show that the presented test strategy requires less additional silicon area than a complete scan path. Thus the advantages of a partial scan path are combined with the well-known benefits of a pseudoexhaustive test, such as high fault coverage and simplified test generation.**

I. Introduction

IN [20] and [21] the pseudoexhaustive test was proposed in order to reduce the costs of test pattern generation and test application. For a primary output o of a combinational circuit, the cone C_o is the subcircuit containing all predecessors of o (Fig. 1). A cone is tested by applying all possible patterns at its primary inputs. The total number of all these patterns is smaller than an exhaustive test if the cones are sufficiently small.

An obvious advantage of this test strategy is the high fault coverage: within a cone all combinationally faulty functions are detected. Faulty sequential behavior induced by stuck-open faults can be detected by applying the special pattern sequences described in [28]. Moreover the pseudoexhaustive test sets can be generated by special feedback shift registers [26], [3], [25], which may be used as a self-test technique or for an external low-cost test. A similar approach is possible for CMOS faults [28].

In this paper, we extend the approach to sequential circuits. Using Roth's notation of time frames, a sequential circuit is transformed into a combinational representation [22]. Its size increases linearly with respect to the circuit size if the data-flow graph of the circuit does not contain any cycles [27], [18]. Often the data-flow part of the circuit is acyclic by itself; otherwise flip-flops must be included in a partial scan path [24], [1], [2], [18]. To obtain a pseudoexhaustive test, we generate a pseudoexhaustive test set for the combinational representation and transform these pattern sets into the respective sequences for the original sequential circuit.

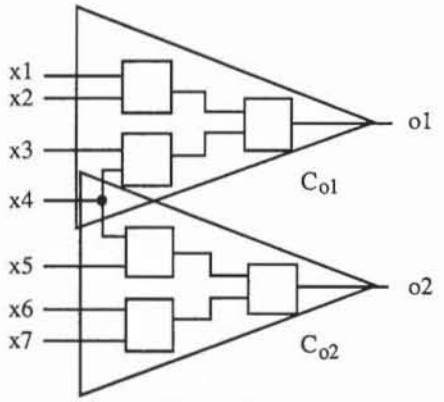

Fig. 1. Cones of outputs $o1$ and $o2$.

A pseudoexhaustive test of the combinational representation is not applicable if a primary output depends on a very large number of primary inputs. In the presented approach, this problem is solved by hardware segmentation, where additional segmentation cells are used to logically disconnect certain circuit lines in the test mode.

A uniform technique is presented integrating the partial scan design and the segmentation of a sequential network in a way similar to that proposed in [13]. Examples show that the additional silicon area needed for the partial scan path and the segmentation cells together is less than the overhead for a complete scan path. As an additional advantage we have complete fault coverage without expensive test pattern generation.

After this introductory section, we sketch some basic graph-theory definitions and facts which are necessary in our approach to circuit modeling. In Section III we present the cells necessary for the design of pseudoexhaustively testable circuits. Besides the well-known LSSD latches, these are the segmentation cells already mentioned. In Section IV we discuss placement algorithms which make a pseudoexhaustive test feasible using a minimum number of these cells.

In Section V we discuss pseudoexhaustive test sequences. The sequences can be generated by linear feedback shift registers (LFSR's) in a way similar to that proposed for combinational networks [5]. Finally, we present some examples.

Manuscript received April 24, 1990. This paper was recommended by Guest Editor J. Rajski.

H.-J. Wunderlich was with the Institute of Computer Design and Fault Tolerance, University of Karlsruhe, Karlsruhe, Federal Republic of Germany. He is now with the University of Siegen, Siegen, Federal Republic of Germany.

S. Hellebrand was with the Institute of Computer Design and Fault Tolerance, University of Karlsruhe, Karlsruhe, Federal Republic of Germany. She is now with the TIM3/IMAG—Computer Architecture Group, Grenoble, France.

IEEE Log Number 9103247.

II. Circuit Modeling and Restrictions

We assume that the sequential circuits are described at gate level, and that the following restrictions are fulfilled.

- The circuits are purely synchronous.
- Only D flip-flops are used.
- The D flip-flops can be augmented according to the rules of either level-sensitive or edge-triggered scan design (LSSD, ETSD).
- In order to control a partial scan path for ETSD circuits, the test signal, T, must block the clock of the unscanned flip-flops. For LSSD, shift clocks and system clocks must be separated.

Such a circuit is modeled by a directed graph $G := (V, E)$ with vertices V and edges $E \subset V^2$. $V := V_C \cup V_S \cup I$ is a disjoint union of combinational vertices, V_C, sequential vertices, V_S, and inputs, I. The outputs of the gates are represented by V_C; the outputs of the flip-flops are represented by V_S; and I contains both the primary and the pseudoprimary inputs. The pseudoprimary inputs correspond to the flip-flops within the scanpath. An example circuit and its circuit graph are shown in Figs. 2 and 3, respectively.

The vertices of this circuit graph are partitioned into the three sets: $V_C = \{6, 8, 9, 12, 13, 15, 16, 18\}$, $V_S = \{7, 10, 11, 14, 17\}$, and $I = \{1, 2, 3, 4, 5\}$.

In general, we have $(v, w) \in E$ if node v is an input of a component, gate, or flip-flop with output node w. The primary outputs are a subset $O \subset V$. For the example circuit, we have $O := \{14, 15, 18\}$.

In the following some basic graph-theory notions are summarized; the corresponding symbols are listed in the Appendix. For a vertex $v \in V$ the set of *direct predecessors* is $pd(v) := \{w \in V \mid (w, v) \in E\}$, and $sd(v) := \{w \in V \mid (v, w) \in E\}$ is the set of *direct successors*. The set of *predecessors* is defined as $p(v) := \{w \in V \mid$ there is a path from w to $v\}$, and $s(v) := \{w \in V \mid$ there is a path from v to $w\}$ is the set of *successors*. Finally a circuit graph G is called consistent if $\{v \in V \mid pd(v) = \emptyset\} = I$ holds. We deal only with consistent circuit graphs.

Only the topology of the storage elements V_S determines the test length. It is described by the so-called S graph.

Definition 1: Let $G := (V, E)$ be a circuit graph with $V := V_C \cup V_S \cup I$ and O. Its S graph $G^S := (V^S, E^S)$ is defined by:

a) $V^S := V_S \cup I$, $I^S := I$ and
$O^S := \{v \in V^S \mid$ there is a path ω from v to an output $o \in O$ in G, and $\omega \cap V^S = \{v\}\} \cup (O \cap V^S)$.

b) $E^S := \{(v, w) \in V^S \times V^S \mid$ there is a path ω from v to w in G, and $\omega \cap V^S = \{v, w\}\}$.

Fig. 4 shows the S graph corresponding to the circuit graph of Fig. 3.

The approach presented is valid for circuits where the S graph does not contain any cycles. Every S graph can be made acyclic by integrating some of the flip-flops into an incomplete scan path. For instance, if in the example circuit flip-flop 17 were a scan path element, then the resulting circuit graph would be acyclic (Fig. 5).

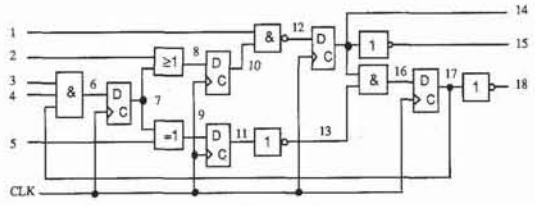

Fig. 2. Example circuit.

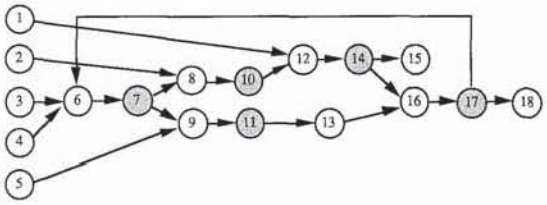

Fig. 3. Circuit graph.

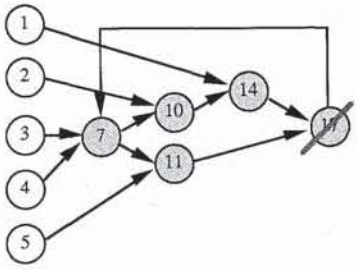

Fig. 4. S graph.

The pseudoexhaustive test of sequential circuits requires the application of pattern sequences instead of single patterns. Using Roth's notation of time frames, copies of the combinational part of the circuit are generated, and the number of time frames corresponds to the length of the test sequences. We modifiy this approach such that at each time step we copy only the small part of the combinational circuit that is actually needed for fault detection. In order to describe our solutions exactly, more graph-theory definitions are required:

Definition 2: Let $G := (V, E)$ be an acyclic graph. The *rank* of G is defined by rank $(G) := \max \{L(\omega) \mid \omega$ is a path in $G\}$, where $L(\omega)$ denotes the length of a path ω.

Definition 3: Let $G := (V, E)$ be an S graph. The function P: $\mathcal{P}(V) \rightarrow \mathcal{P}(V)$, defined by $P(W) := \cup_{w \in W} pd(w)$, is called a backtrace function.

The nodes of a subset $W^t \subset V$ have defined values at time step t if the nodes $W^{t-1} := P(W^t)$ have defined values at time step $t - 1$, and we can use this notation for state backtracing.

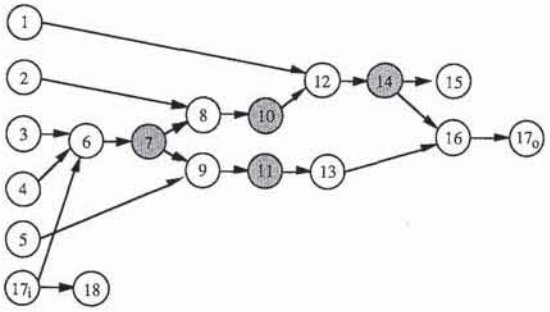

Fig. 5. Acyclic circuit graph.

Observation 1: Let $G := (V, E)$ be an acyclic S graph with rank $(G) = r$. Then $P^r(V) \subset I$.

Corollary: Every state is reachable within r steps if it is reachable at all.

Definition 4: Let $G := (V, E)$ be a circuit graph. Let the corresponding S graph, $G^S := (V^S, E^S)$, be acyclic with rank r. Furthermore let

$$W^r := O^S$$

$$V^r := W^r \cup O \cup \{v \in V \mid \exists\, u \in W^r\ \exists\, w \in O\ (v \text{ is member of a path } \omega \text{ from } u \text{ to } w \text{ and } \omega \cap V^S = \{u\})\}$$

and for $0 \le t < r$,

$$W^t := P(W^{t+1}),$$

$$V^t := \{v \in V \mid \exists\, u \in W^t\ \exists\, w \in W^{t+1}\ (v \neq u \text{ is member of a path } \omega \text{ from } u \text{ to } w \text{ and } \omega \cap V^S = \{u, w\})\} \cup W^t.$$

The combinational representation of G is the graph $\overline{G} := (\overline{V}, \overline{E})$, where

$$\overline{V} := \bigcup_{0 \le t \le r} V^t \times \{t\},$$

$$\overline{E} := \bigcup_{0 \le t \le r} \{((x, t), (y, t)) \mid (x, y) \in V^t \times V^t \cap E\} \cup \bigcup_{0 \le t \le r} \{((x, t), (y, t+1)) \mid x \in V^t \wedge y \in W^{t+1} \wedge (x, y) \in E\}$$

and

$$\overline{V}_C := \bigcup_{0 \le t \le r} \{(x, t) \in \overline{V} \mid x \in V_C \cup V_S\},$$

$$\overline{I} := \{(x, t) \in \overline{V} \mid x \in I\},\ \overline{O} := O \times \{r\}.$$

It should be noted, that all flip-flops are mapped to combinational buffers. The combinational representation for the example circuit graph of Fig. 5 is shown in Fig. 6. This straightforward construction provides our basic theorem.

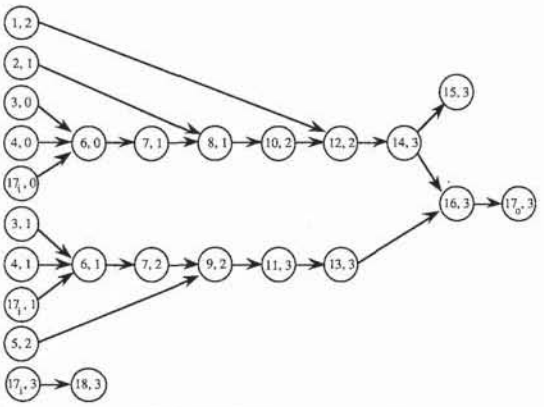

Fig. 6. Combinational representation.

Theorem 1: Let $G := (V, E)$ be an acyclic circuit graph with rank r and let $\overline{G} := (\overline{V}, \overline{E})$ be its combinational representation. A pattern sequence $\langle \langle b_i^t \in \{0,1\} \mid i \in I \rangle \mid 0 \le t \le r \rangle$ detects a fault of a node $v \in V$ exactly at time r if and only if in $\overline{G}$ the corresponding multiple fault of the nodes (v, t), $0 \le t \le r$, if defined, is detected by the pattern $\langle b_i^t \mid (i, t) \in \overline{I} \rangle$.

Theorem 1 justifies the following definition.

Definition 5: Let $G := (V, E)$ be an acyclic circuit graph with rank r and let $\overline{G} := (\overline{V}, \overline{E})$ be its combinational representation. A set B of pattern sequences of length $r + 1$ is called a pseudoexhaustive test for G if the set $\overline{B}$ obtained by mapping each sequence $\langle \langle b_i^t \in \{0,1\} \mid i \in I \rangle \mid 0 \le t \le r \rangle$ onto a pattern $\langle b_i^t \mid (i, t) \in \overline{I} \rangle$ provides a pseudoexhaustive test for $\overline{G}$.

If a cone C_o of the combinational representation has l primary inputs, it is tested exhaustively by 2^l patterns. Each pattern is mapped to a sequence of the maximal length r in the original circuit. Thus the length of the pseudoexhaustive test sequence is bounded by $r \cdot 2^l$. In Section V we discuss some further compactions.

This approach is applicable to all fault models concerning the combinational function of a single node or of multiple nodes. If the design is irredundant, a complete fault coverage is obtained. Faults affecting the topology of the S graph are not guaranteed to be detected. Bridges might connect various cones, but they are hard to detect in purely combinational circuits, too [4].

III. Devices Supporting the Pseudoexhaustive Test

A pseudoexhaustive test is feasible if the corresponding S graph is acyclic and if each cone of the combinational representation has only a limited number of inputs. The first condition can be satisfied by integrating some flip-flops or latches into a partial scan path, P, extending the well-known LSSD rules [9]. This results in the circuit structure of Fig. 7. In order to keep the hardward overhead small, the size of P should be minimal.

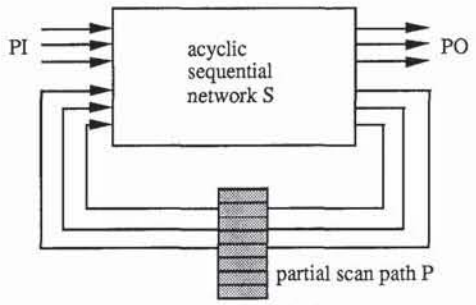

Fig. 7. Partial scan design.

Fig. 8. Cut of a node v.

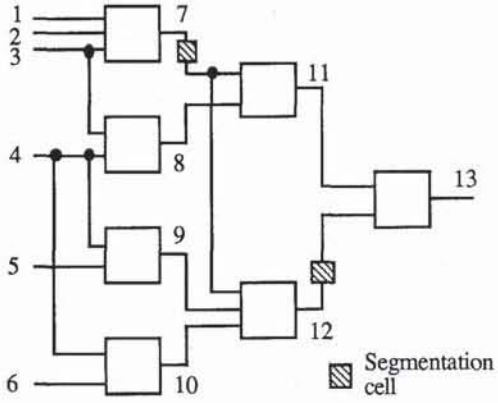

Fig. 9. Hardware segmentation by special cells.

In order to fulfill the second requirement, certain nodes within the sequential network have to be cut such that they are directly accessible. In this way new pseudoprimary inputs, v_i, and outputs, v_o are introduced to replace a cut node v (Fig. 8).

If node v corresponds to a latch in the original network, it is cut by its integration into the partial scan path. For the general case multiplexer partition was originally proposed [20], but this has serious drawbacks with regard to area, speed, test control, and fault coverage [13]. These disadvantages are avoided by the use of segmentation cells (Fig. 9).

In [6] unmodified latches have been proposed for segmentation purposes. But this alters the clocking scheme, and the speed of the entire circuit is slowed down. For this reason, we use the more sophisticated cell shown in Fig. 10. In system mode $S = 1$ is asserted so that D and Q are directly connected. For $S = 0$ the cell works like the usual LSSD L1(L2*) latch with data input D, clock CLK, shift input SDI, and shift clock $A(B)$.

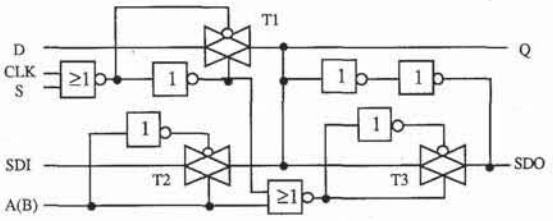

Fig. 10. Segmentation cell.

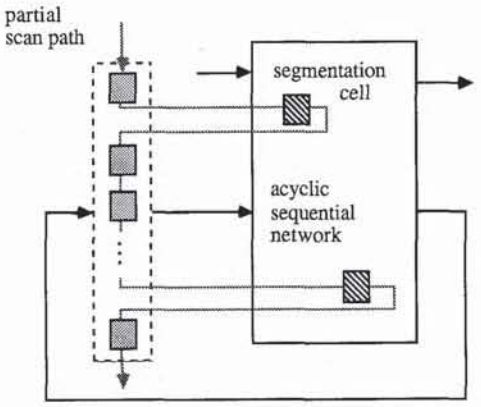

Fig. 11. Integrating segmentation and scan design.

These cells are added to the partial scan path, but they do not affect the system operation of the circuit (Fig. 11).

In the next section, we discuss how to place the directly accessible latches and segmentation cells.

IV. Design Algorithms

The following modifications of a design are required to realize a pseudoexhaustive test strategy:

a) A small number of latches must be integrated into a partial scan path in order to obtain an acyclic S graph.
b) A minimal number of lines within the original circuit must be cut in order to obtain small sets of inputs of the cones within the combinational representation. Since the integration of an existing latch into the scan path requires less hardware overhead than adding a new segmentation cell, cutting nodes corresponding to latches is given preference.

Now we want to describe these tasks in graph-theory terms.

Definition 6: Let $v \in V$. The cut of $G := (V, E)$ in v is the graph $G_{\{v\}} := (V_{\{v\}}, E_{\{v\}})$, where $V_{\{v\}} := \{v_i, v_o\} \cup V \backslash \{v\}$, $v_i, v_o \notin V$, $E_{\{v\}} := \{(x, y) \mid x = v_i \wedge (v, y) \in E) \vee (y = v_o \wedge (x, v) \in E)\} \cup E \backslash \{(x, y) \mid x = v \vee y = v\}$.

One easily verifies that for $v, w \in V$ the cuts are independent of their order, $G_{\{v\}\{w\}} = G_{\{w\}\{v\}}$. Thus we have the following definition.

Definition 7: Let $W := \{w_1, \cdots, w_m\} \subset V$. The *cut* of $G = (V, E)$ along W is the graph $G_W := (V_W, E_W) := G_{\{w_1\}\cdots\{w_m\}}$.

The subproblem of generating acyclic S graphs can now be stated as follows:

Problem FBN (Feedback Node): Let $G = (V, E)$ be an S graph. Find a set $W \subset V$ of minimal cardinality such that $G_W = (V_W, E_W)$ is acyclic.

FBN is known to be NP-complete [17], and heuristics are used in order to obtain good, suboptimal solutions. Let Z_G be the set of all elementary cycles of G. For each cycle $z \in Z_G$, we define $n(z) := \{v \in V \mid v \in z\}$, the set of all nodes of z. Now the scan selection problem is divided into two subproblems:

i) For the S graph $G = (V, E)$, create the set of all elementary cycles Z_G, i.e., all cycles where each node only appears once.
ii) Set $H := \cup_{z \in Z_G} n(z)$. Find a set $W \subset H$ of minimal cardinality such that $\forall_z \in Z_G W \cap n(z) \neq \emptyset$.

These are standard problems of graph theory, and there are well-known solutions. The implemented algorithms are based on methods described in [8] and [16], with additional heuristics used. Alternatively we select a bounded set Z'_G of elementary cycles, solve the hitting set subproblem ii), and select another bounded Z'_G. A detailed description of the implemented algorithm is found in [18] and [27].

The solution of modification b), mentioned at the beginning of this section, is more complicated. First we explain how to segment purely combinational circuits, and then we extend this approach to general combinational representations.

Definition 8: Let $\overline{G} = (\overline{V}, \overline{E})$ be a combinational representation, and let $v \in \overline{V}$. The natural number $d(v) := \mid I \cap p(v) \mid$ is called the dependence level of v.

Now we can state the segmentation problem of combinational circuits exactly:

Problem OCS (Optimal Circuit Segmentation): Let $G := (V, E)$ be a circuit graph of a combinational circuit, and $l \in IN$. Is there a set $W \subset V$ of size $k \leq \mid V \mid$ such that all vertices $v \in V_W$ in $G_W := (V_W, E_W)$ have a dependence level of at most l, i.e., $\mathrm{d}(v) \leq l$ in G_W?

For any cut along W, $\mathrm{d}(v) \leq l$ in G_W can be checked in nearly linear time. Generating and checking all $2^{|V|}$ cuts would take exponential effort. Unfortunately, we cannot expect an algorithm of a better worst-case complexity, since OCS is NP-complete for $l > 2$. A complex proof of this theorem is found in [6], with a shorter treatment in [12] and [29].

As OCS is NP-complete, we refrain from looking for minimal solutions, but present some efficient heuristics. Dealing with general combinational representations is even more complex, since one physical cut of the circuit corresponds to multiple cuts in various time frames.

Definition 9: Let $G = (V, E)$ be an acyclic circuit graph, with $\overline{G} = (\overline{V}, \overline{E})$ its combinational representation and $v \in V$. The *relevant time steps* of v are in the set $T(v) := \{d \mid (v, d) \in \overline{V}\}$. The *equivalent node set* of v is $Q(v) := \{(w, d) \in \overline{V} \mid w = v\} \subset \overline{V}$.

For a subset $W \subset V$ we use the abbreviation $Q(W) := \cup_{w \in W} Q(w)$. Now we can formulate the general segmentation problem:

Problem OCRS: Let $G := (V, E)$ be an acyclic graph, let $\overline{G} = (\overline{V}, \overline{E})$ be its combinational representation, and let $l \in IN$. Is there a set $W \subset V$ of size $k \leq |V|$, such that all vertices $v \in \overline{V}_{Q(W)}$ in $\overline{G}_{Q(W)} = (\overline{V}_{Q(W)}, \overline{E}_{Q(W)})$ have a dependence level of at most l, i.e., $\mathrm{d}(v) \leq l$ in $\overline{G}_{Q(W)}$?

We use some heuristics for an approximate solution of OCRS, applying well-known methods, since OCRS is an instance of a general combinatorial optimization problem:

CO (Combinational Optimization): Let $\mathfrak{F}$ be a set of states, $\mathfrak{F}^* \subset \mathfrak{F}$ be a set of admissible states, and let k: $\mathfrak{F} \rightarrow IR$ be a cost function. Find an admissible state $Z \in \mathfrak{F}^*$ with minimal costs $k(Z) = \min \{k(X) \mid X \in \mathfrak{F}^*\}$.

For OCRS the set of states is $\mathfrak{F} := \mathcal{P}(V)$, since every $Z \subset V$ determines a set of cuts with resulting graph $\overline{G}_{Q(Z)} = (\overline{V}_{Q(Z)}, \overline{E}_{Q(Z)})$. The admissible states are $\mathfrak{F}^* := \{Z \in \mathfrak{F} \mid \forall v \in \overline{V}_{Q(Z)} (d(v) \leq l \text{ in } \overline{G}_{Q(Z)})\}$.

The cost function k: $\mathfrak{F} \rightarrow IR$, $k(Z) := \mid Z \mid$ corresponds to the necessary number of segmentation cells.

We define a heuristical function h: $\mathfrak{F} \rightarrow IR$ to evaluate states:

$$h(Z) := \sum_{v \in V'} \ln(d(v)),$$

$$\text{where } V' := \{v \in \overline{V}_{Q(Z)} \mid d(v) > l \text{ in } \overline{G}_{Q(Z)}\}.$$

This function is an estimation of the number of vertices which have to be cut in the combinational representation. We assume an enumeration $\langle v_1, \cdots, v_n \rangle$ of $\overline{V}$ with $v_i \in pd(v_j) \Rightarrow i < j$.

Definition 10: Let $\overline{G} = (\overline{V}, \overline{E})$ be a combinational representation, and let $v \in \overline{V}$. The *cone* $C(v)$ of v is the subgraph $C(v) := (p(v) \cup \{v\}, (p(v) \cup \{v\})^2 \cap \overline{E})$.

Definition 11: Let $\overline{G} = (\overline{V}, \overline{E})$ be a combinational representation, and $l \in IN$. The first violation $fv \in \overline{V}$ is the node $fv = v_i$, where $i := \min \{j \mid d(v_j) > l\}$.

This definition is illustrated by Fig. 12.

We construct a search graph $\mathcal{S} = (\mathfrak{F}, \mathcal{E})$, where the nodes $Z \in \mathfrak{F} = \mathcal{P}(V)$ define cuts in the sequential network, and an edge $(Z_1, Z_2) \in \mathcal{E}$ exists if and only if

a) $fv \in V \backslash Z_1$ is the first violation in $\overline{G}_{Q(Z_1)}$;
b) $Z_2 := Z_1 \cup \{v\}$, where $Q(v) \cap p(fv) \neq \emptyset$ and $h(Z_1 \cup \{v\})$ is minimal.

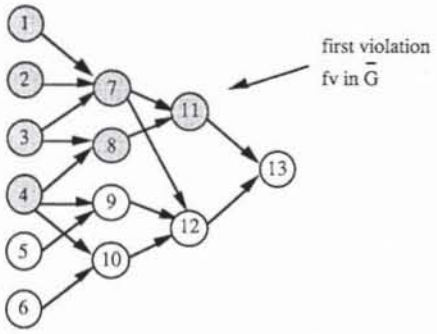

Fig. 12. Example for $l = 3$.

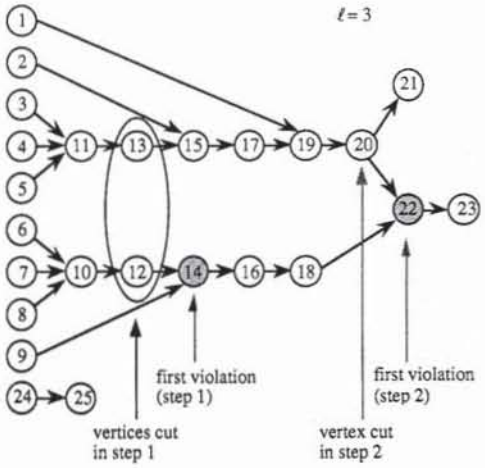

Fig. 13. Steps of the segmentation algorithm for the example circuit of Fig. 2.

The search is started at $Z_0 = \varnothing$. One branches from Z to a $Z_1 \in \{\bar{Z} \mid (Z, \bar{Z}) \in \mathcal{E}\}$, preferably cutting lines corresponding to latches or flip-flops in order to reduce the hardware overhead, until an admissible state Z is reached. The results of this process are presented in Section VI.

For the example circuit of Fig. 2 and $l = 3$, the algorithm requires two steps to determine a solution for OCRS. This is illustrated in Fig. 13, which shows the combinational representation of the circuit with an enumeration according to the signal flow. In the first step, node 14 is the first violation and the two equivalent nodes, 12 and 13, are chosen to be cut. In the second step the first violation is node 22 and the algorithm decides to cut node 20. Nodes 12 and 13 correspond to node 7 in the original circuit, and node 20 corresponds to 14. As both 7 and 14 are flip-flops, they simply have to be integrated into the partial scan path.

Since node 17 has been cut before to guarantee an acyclic S graph, there are altogether three of five flip-flops in the partial scan path. Fig. 14 shows the resulting circuit.

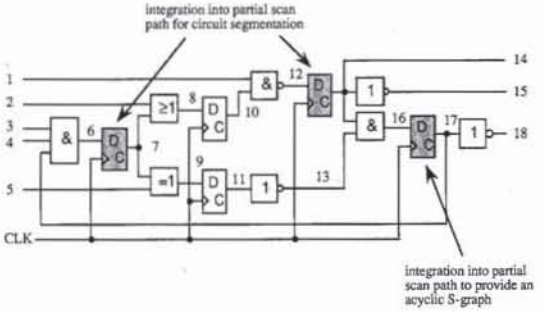

Fig. 14. Resulting circuit after integration of a partial scan path and segmentation.

V. Pseudoexhaustive Test Sequences

A considerable amount of work has already been done in investigating the generation of pseudoexhaustive test sets for combinational circuits. Dependence matrices [15], linear sums [3], cyclic codes [23], [26], and special polynomials for linear feedback shift registers [5] have been proposed. They are all applicable to combinational representations, and little effort is needed to transform the resulting patterns into pseudoexhaustive test sequences. Each pattern $p := \langle b_i \in \{0, 1\} \mid i \in \bar{I} \rangle$ of the combinational representation corresponds to a pattern sequence. We remember $\bar{I} \subset I \times T$, where $T := \{0, \cdots, r\}$; hence the sequence is $S(p) := \langle \langle b_j^t \in \{0, 1\} \mid j \in I \wedge t \in T(j) \rangle \mid t \in T \rangle$.

It should be noted, that it is not for the entire set $I \times T$ that values are defined in $S(p)$. This can be used for pattern compaction.

A further compaction is possible if some sequences have common parts; e.g. the last patterns of sequence $S(p_1)$ are identical to the first of $S(p_2)$. These merged sequences can be generated with the help of linear codes by feedback shift registers, supporting a low-cost external test or a self-test. A complete test chip for generating pseudoexhaustive vectors and sequences externally is presented in [14]. The details are beyond the scope of this paper, which aims at establishing a linear bound on the length of the pseudoexhaustive test sequence.

Let $O \subset V$ be the set of primary and pseudoprimary outputs of the sequential network. Each output function of $o \in O$ is tested by at most 2^l patterns in the combinational representation. Hence the size of the pseudoexhaustive test set is bounded by $|O| \cdot 2^l$. Each pattern is mapped to a sequence of at most length $r := \text{rank } (G)$; hence the entire pseudoexhaustive test sequence is bounded by $r \cdot |O| \cdot 2^l$.

VI. Examples

We discuss three examples: the operation unit of the signal processor (SP) proposed in [7], a multiplier presented in [10], and a PROLOG coprocessor (PP) [11] (see Table I).

The unmodified circuits are very hard to test, which is proven with the help of the program LASAR [19]. Fault

TABLE I
CIRCUIT CHARACTERISTICS

Circuit	Inputs	Outputs	Gates	Flip-flops
SP	83	55	1675	239
MU	43	26	993	193
PP	36	73	1428	136

TABLE II
FAULT COVERAGE BY LASAR AFTER 1 h OF COMPUTING TIME

PP	MU	SP
11.2%	9.8%	8.7%

TABLE III
NUMBER AND PERCENTAGE OF FLIP-FLOPS IN A PARTIAL SCAN PATH IN ORDER TO OBTAIN AN ACYCLIC S-GRAPH

PP	MU	SP
28 (20.6%)	72 (39.3%)	41 (17.2%)

coverages obtained after 3600 seconds of computing time are listed in Table II.

To obtain acyclic S graphs, we first selected a small number of flip-flops using the technique described in [18] and [27] (see Table III).

For these modified circuits, we generated the combinational representation. The representations were segmented by the algorithms described, where the maximal number of inputs to a cone varied from $l = 20$ to $l = 12$. The required test sizes are between a few million and a few thousand patterns, which is competitive with a usual deterministic test.

In Table IV below we distinguish between cuts of flip-flops resulting in additional scan path elements and other cuts requiring more expensive segmentation cells. This table shows significant savings of silicon area compared with the conventional complete scan design. The exact quantification depends on the layout of the LSSD and segmentation cells used. A rough estimation shows savings of approximately 50% for $l = 20$ and $l = 16$. But even for $l = 12$, the hardware overhead is competitive with a conventional scan design, since the larger number of segmentation cells is balanced by the shorter partial scan path.

In all cases the advantages are obvious:

- complete fault coverage with respect to the usual fault models;
- no expensive test pattern generation;
- simple test application.

Also, with respect to the number of necessary segmentation cells, the partial scan design is in most cases superior to the complete scan path. This is because the integration of a complete scan path in general does not provide a pseudoexhaustively testable circuit. Additional segmentation cells are necessary. Table V shows the

TABLE IV
NECESSARY NUMBER OF SEGMENTATION CELLS AND NUMBER OF FLIP-FLOPS IN THE PARTIAL SCAN PATH, IN ORDER TO MAKE THE CIRCUITS PSEUDOEXHAUSTIVELY TESTABLE

	PP	MU	SP
$l = 20$:			
No. segmentation cells	3	7	5
No. additional flip-flops in the scan path	6	6	72
No. overall flip-flops in the scan path (percentage)	34 (25%)	78 (43%)	113 (47%)
$l = 16$:			
No. segmentation cells	6	4	9
No. additional flip-flops in the scan path	12	13	95
No. overall flip-flops in the scan path (percentage)	40 (29%)	85 (46%)	136 (57%)
$l = 12$:			
No. segmentation cells	23	24	16
No. additional flip-flops in the scan path	22	30	131
No. overall flip-flops in the scan path (percentage)	50 (37%)	102 (56%)	172 (72%)

TABLE V
NUMBER OF NECESSARY SEGMENTATION CELLS USING COMPLETE AND PARTIAL SCAN DESIGN, RESPECTIVELY

	PP	MU	SP
$l = 20$:			
Partial scan path	3	7	5
Complete scan path	9	4	4
$l = 16$:			
Partial scan path	6	4	9
Complete scan path	14	7	10
$l = 12$:			
Partial scan path	23	24	16
Complete scan path	40	11	56

number of segmentation cells required for an efficient pseudoexhaustive test based on a complete and on a partial scan design.

In most cases the additional number of segmentation cells increases if all flip-flops are integrated into a complete scan path.

VII. CONCLUSIONS

The new concept of a pseudoexhaustive test of sequential circuits has been introduced. Flip-flops and latches are integrated into an incomplete scan path, such that each possible state of the circuit is reachable within a few steps. More flip-flops and new segmentation cells are added to the partial scan path in order to make a pseudoexhaustive test feasible. Algorithms have been presented for placing these devices automatically. Moreover it has been shown how to transform a pseudoexhaustive test set into a pseudoexhaustive test sequence of a similar size.

The analyzed examples show that a conventional complete scan path without additional testability features requires more hardware overhead than the presented test strategy, which retains all the known benefits of a pseudoexhaustive test.

APPENDIX

$\mathcal{P}(M)$	Power set of the set M.
$G = (V, E)$	circuit graph.
V_C	Set of combinational vertices in V.
V_S	Set of sequential vertices in V.
I	Set of inputs.
O	Set of outputs.
$pd(v)$	Set of direct predecessors of a vertex $v = V$.
$sd(v)$	Set of direct successors of a vertex $v = V$.
$p(v)$	Set of predecessors of a vertex $v \in V$.
$s(v)$	Set of successors of a vertex $v \in V$.
$L(\omega)$	Length of a path ω.
rank (G)	Rank of an acyclic graph G.
$G^S = (V^S, E^S)$	S graph of the graph $G = (V, E)$.
$P: \mathcal{P}(V) \rightarrow \mathcal{P}(V)$	Backtrace function, $P(W) := \cup_{w \in W} pd(w)$.
$\overline{G} = (\overline{V}, \overline{E})$	Combinational representation of the graph $G = (V, E)$.
$G_{\{v\}} = (V_{\{v\}}, E_{\{v\}})$	Cut of the graph $G = (V, E)$ in $v \in V$.
$G_W = (V_W, E_W)$	Cut of the graph $G = (V, E)$ along $W \subset V$.
$d(v)$	Dependence level of a vertex $v \in V$, $d(v) := \lvert I \cap p(v) \rvert$
$T(v)$	Set of relevant times steps of a vertex $v \in V$.
$Q(v)$	Equivalent node set of a vertex $v \in V$.
$Q(W)$	$\cup_{w \in W} Q(w)$.

REFERENCES

[1] V. D. Agrawal, K. Cheng, D. D. Johnson, T. Lin, "Designing circuits with partial scan," *IEEE Design and Test of Computers*, Apr. 1988.

[2] V. D. Agrawal and K. Cheng, "A partial scan method for sequential circuits with feedback," *IEEE Trans. Comput.*, vol. 39, Apr. 1990.

[3] S. B. Akers, "On the use of linear sums in exhaustive testing," in *Proc. Symp. Fault Tolerant Computing*, 1985.

[4] E. C. Archambeau and E. J. McCluskey, "Fault coverage of pseudo-exhaustive testing," in *Proc. Int. Symp. Fault Tolerant Computing*, 1984.

[5] Z. Barzilai, D. Coppersmith, and A. L. Rosenberg, "Exhaustive generation of bit patterns with applications to VLSI self-testing," *IEEE Trans. Computer.*, vol. C-32, Feb. 1983.

[6] S. N. Bhatt, F. R. K. Chung, A. L. Rosenberg, "Partitioning circuits for improved testability," *Advanced Research in VLSI: Proc. 4th MIT Conf.*, Apr. 7–9, 1986.

[7] J. J. LeBlanc, "LOCST: A built-in self-test technique," *IEEE Design & Test*, vol. 1, no. 4, 1984.

[8] N. Christofides and S. Korman, "A computational survey of methods for the set covering problem" in: *Management Sci.*, vol. 21, no. 5, Jan. 1975.

[9] E. B. Eichelberger and T. W. Williams, "A logic design structure for LSI testability," in *Proc. 14th Design Automat. Conf.*, June 1977.

[10] P. Gutberlet, "Entwurf eines schnellen Matrizen-Multiplizierers," Diploma thesis, Institute of Computer Design and Fault Tolerance, University of Karlsruhe, FRG, 1988.

[11] O. Haberl, "Entwurf und Implementierung eines PROLOG-Preprozessors als Standardzellen-Chip mit dem Entwurfssystem VENUS," Diploma thesis, Institute of Computer Design and Fault Tolerance, University of Karlsruhe, FRG, 1987.

[12] S. Hellebrand, "Synthese vollständig testbarer Schaltungen," Ph.D. thesis, Düsseldorf: VDI-Verlag, 1991.

[13] S. Hellebrand and H.-J. Wunderlich, "Tools and devices supporting the pseudo-exhaustive test" in *Proc. European Design Automat. Conf.*

[14] S. Hellebrand, H-J. Wunderlich, and O. F. Haberl, "Generating pseudo-exhaustive vectors for external testing," *Proc. Int. Test Conf.* (Washington), 1990.

[15] F. Hirose and V. Singh, "McDDP, A program for partitioning verification testing matrices," CRC Tech. Rep. No. 81-13, Stanford, 1982.

[16] D. B. Johnson, "Finding all the elementary circuits of a directed graph," *SIAM J. Comput.*, vol. 4, no. 1, Mar. 1975.

[17] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. New York: Plenum Press, 1972.

[18] A. Kunzmann and H.-J. Wunderlich, "An analytical approach to the partial scan problem," *J. Electron. Testing: Theory and Applications*, no. 1, 1990.

[19] *ETS LASAR Users Guide*, Vers. 4.7, Nov. 1985.

[20] E. J. McCluskey and S. Bozorgui-Nesbat, "Design for autonomous test," *IEEE Trans. Comput.*, vol. C-30, Nov. 1981.

[21] E. J. McCluskey, "Verification testing—A pseudoexhaustive test technique," *IEEE Trans. Comput.*, vol. C-33, June 1984.

[22] J. P. Roth, "Sequential Test Generation," *IBM Tech. Disclosure Bull.*, Jan. 1978.

[23] D. T. Tang and C.-L. Chen, "Logic test pattern generation using linear codes," *IEEE Trans. Comput.*, vol. C-33, Sept. 1984.

[24] E. Trischler, "Testability analysis and incomplete scan path," in *Proc. ICCAD*, 1983.

[25] J. G. Udell, Jr., "Test set generation for pseudo-exhaustive BIST," in *Proc. Int. Conf. Computer Aided Design*, 1986.

[26] L.-T. Wang and E. J. McCluskey, "Circuits for pseudo-exhaustive test pattern generation," in *Proc. Int. Test Conf.*, 1986.

[27] H.-J. Wunderlich, "The design of random-testable sequential circuits," in *Proc. 19th Int. Symp. Fault Tolerant Computing* (Chicago), 1989.

[28] H.-J. Wunderlich and S. Hellebrand, "Generating pattern sequences for the pseudo-exhaustive test of MOS-circuits," in *Proc. 18th Int. Symp. Fault Tolerant Computing* (Tokyo), 1988.

[29] H.-J. Wunderlich, *Hochintegrierte Schaltungen. Prüfgerechter Entwurf und Test.* Berlin: Springer-Verlag, 1991.

Hans-Joachim Wunderlich (A'86) received the diploma degree in mathematics from the University of Freiburg, Germany, in 1981 and the Dr. rer. nat. (Ph.D.) degree from the University of Karlsruhe in 1986.

In 1982, he was a consultant at the Fraunhofer Institute of Industrial Engineering, Stuttgart, where he worked in the field of operations research. In 1983, he joined the Institute of Computer Design and Fault Tolerance, University of Karlruhe, where he has headed a research group on automation of circuit design and testing since 1986. His research interests include computer-aided design for testability, test generation, and digital simulation. Currently Dr. Wunderlich is a Full Professor at the University of Siegen, Germany.

Sybille Hellebrand (A'89) received the diploma degree in mathematics from the University of Regensburg, Germany, in 1986. In December 1986 she joined the Institute of Computer Design and Fault Tolerance, University of Karlsruhe, where she received the Ph.D. degree in 1991. At present she is a postdoctoral fellow at the TIM3/IMAG—Computer Architecture Group, Grenoble, France. Her main research interests are self-testable circuits and design for testability.