Preprint

A mixed-mode BIST scheme based on folding compression

Huaguo Liang, Sybille Hellebrand and Hans-Joachim Wunderlich

Journal of Computer Science and Technology
Vol. 17(2), March, pp. 203-212

Vol.17 No.2 J. Comput. Sci. & Technol. Mar. 2002

A Mixed-Mode BIST Scheme Based on Folding Compression

LIANG Huaguo (梁华国)[1], Sybille Hellebrand[2] and Hans-Joachim Wunderlich[3]

[1]*Department of Computer and Information, Hefei University of Technology, Hefei 230009, P.R. China*

[2]*Institute of Applied Computer Science, University of Innsbruck, Austria*

[3]*Institute of Computer Science, University of Stuttgart, Germany*

E-mail: huaguolg@informatick.uni-stuttgart.de

Received May 22, 2000; revised February 27, 2001.

Abstract In this paper a new scheme for mixed mode scan-based BIST is presented with complete fault coverage, and some new concepts of folding set and computing are introduced. This scheme applies single feedback polynomial of LFSR for generating pseudo-random patterns, as well as for compressing and extending seeds of folding sets and an LFSR, where we encode seed of folding set as an initial seed of LFSR. Moreover these new techniques are 100% compatible with scan design. Experimental results show that the proposed scheme outperforms previously published approaches based on the reseeding of LFSRs.

Keywords BIST, random pattern testing, LFSR, folding set, encoding seed

1 Introduction

With the development of IC technology at a high speed and embedding different types of cores into single devices (system-on-a-chip), integrated circuit testing is facing many challenges, in which these highly-integrated circuits decrease controllability and observability of components. The scan-based built-in self-test method offers a comprehensive test solution. Therefore BIST (Built-In-Self-Test) technology has become a hot topic in recent years.

For conventional IC testing, a number of powerful BIST techniques have been developed in the past. For example, pseudo-random testing[1] can efficiently generate pseudo-random patterns on chip using linear feedback shift registers (LFSRs). Its structure of pattern generator is simple and hardware overhead is low. But it is difficult to achieve guaranteed levels of fault coverage, especially for circuits containing random-pattern resistant faults. Test point insertion[2–4] and weighted random testing techniques[5–8] have been proposed to address this problem. Test points may be inserted to partition the circuit to introduce extra control or observation points, such that the modified circuit becomes easily testable. Very high fault coverage is the main advantage of this technique. However, this introduces a significant area overhead, and may deteriorate circuit performance during normal operation. In weighted random testing, weight circuits are used to bias a pseudo-random sequence using precomputed weight sets to reduce test length. The combinational logic overhead of weighted random generators may be high and also adversely degrade normal circuit performance.

A multiple seed LFSR scheme[9] uniformly draws patterns from the entire space producible by the LFSR via automatically changing seed circuit and suitably decreases testing time. But the random pattern generator requires four latches and an Exclusive-OR gate for each LFSR stage, and its shift register must be the same length as the scan chain under test circuits, so that the hardware overhead of the generator is rather large for including long scan chain. Moreover, the random change of seed is used to a certain degree, and thus the complete fault coverage cannot be obtained within reasonable testing time for some circuits. To address the issue of achieving high fault coverage with reasonable test application time, common approaches are the mixed-mode BIST, which focus on characterization of complete test sets. LFSRs with multiple seeds or reconfigurable feedback polynomials[10–12] can reduce

Currently LIANG Huaguo is a guest research fellow at the University of Stuttgart, Germany.
This work has been supported by the DFG grant Wu 245/1-3.

the area needed for storing a compact test set and the test length. However, with the increasement of the number of polynomials, the structure of pattern generator becomes complicated, thereby overhead is increased. The deterministic patterns are either stored on chip in a compressed format and expanded during BIST ("store and generate") or directly embedded into an LFSR sequence by "bit-fixing" or "bit-flipping" techniques[13−15]. It has been demonstrated that bit-fixing and bit-flipping can provide high-quality test patterns at low hardware overhead. However, the BIST architecture is extremely tailored to the specific circuit, and a change in the test set requires a resynthesis of the complete BIST hardware.

In this paper, we present a novel test pattern generator using biseed compression. A unique small LFSR is used to generate pseudo-random patterns. Moreover the same LFSR is loaded with seeds that are of both LFSR and folding sets to generate test patterns for difficult-to-test faults, where the output of LFSR which feeds a scan path is modified by a folding controller. We can compact a deterministic test set widely via encoding seeds of folding sets, and apply folding techniques further to reduce the stored number of the test patterns and get a small seed set. The encoding seeds of folding sets are similar to the width compression of test cubes[16], but it is mainly different from the width compression that can be compatible with scan design. For the seeds of folding sets including don't-care bits we can select the degree of feedback polynomial less than $S_{\max}$ maximal number of specified bits in the seed sets[10−12]. Experimental results show that the proposed scheme outperforms previously published approaches based on the reseeding of LFSRs[11].

2 Folding Set and Deterministic BIST

As mentioned in Section 1, most of the known approaches for deterministic or mixed mode BIST are built around basic pattern generators like LFSRs[10−15,21]. In this section, a new type of generator, called folding set generator, is introduced. The generator is similar to pseudo-pattern generator with an LFSR, and a folding controller is added to the output of the generator to modify bit flow and generate folding sets. All deterministic test patterns can be embedded into processes of the folding set generation.

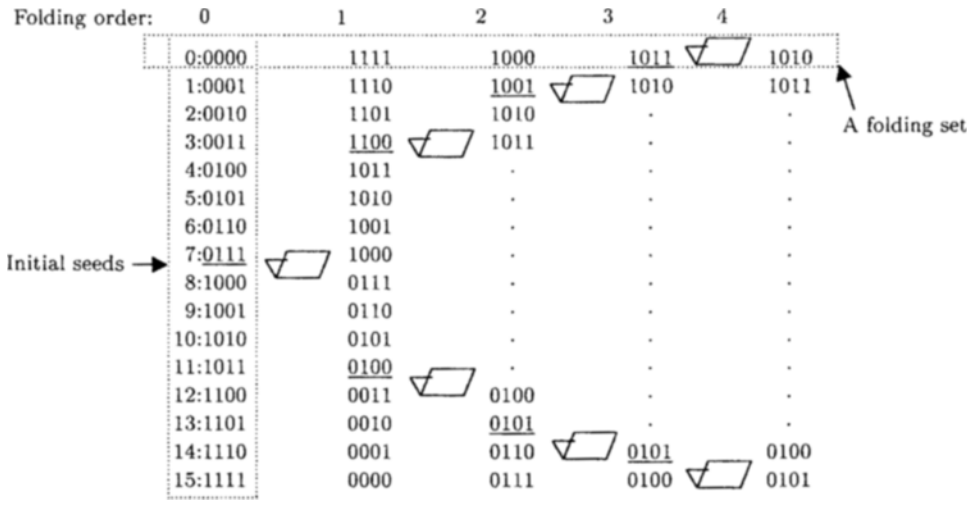

Fig.1. Folding set forming.

Folding set is a new set, which can be formed by half-and-half folding an ordered set $T = \{0,1\}^n$. A simple example is shown in Fig.1. We can use an initial vector $s \in \{0,1\}^n$ to compute a sequence of $n+1$ vectors $\boldsymbol{F}(0,s), \boldsymbol{F}(1,s), \ldots \boldsymbol{F}(i,s), \ldots, \boldsymbol{F}(n,s)$, where for the $\boldsymbol{F}(i,s)$ the initial vector s can be folded i times that the folding compute inverts the odd bits and retains the even bits from the first bit to the i-th bit in s, in which the remaining bits from the i-th bit to the last bit are regarded as a complete bit. A folding set of Fig.1 is shown in Table 1.

Compared to classical generators the folding set sequence produced from a single initial vector (seed) is rather short, and it is very unlikely that a complete set of deterministic vectors can be embedded into a single folding set sequence. However, as it will be shown later, it is generally possible to find a reasonably small number of seeds, such that the union of all the resulting sequences covers a given deterministic test set. This supports an efficient implementation of deterministic BIST, and in particular for scan-based BIST the basic architecture is very simple. As illustrated in Fig.2, the pattern generator consists of a ROM (stored seeds), a shift register and a folding controller.

Table 1. Folding Set Computing $s = 0000$

Folding compute	Folding vectors	Folding times
$F(0,s)$	0000	0
$F(1,s)$	1111	1
$F(2,s)$	1000	2
$F(3,s)$	1011	3
$F(4,s)$	1010	4

To generate a folding set to the CUT during BIST a seed is loaded into the shift register and the T-type flip-flop is reset. The bit and folding counters are initialized as "1" and "0" respectively. If the bit counter value is smaller than the folding counter value, then output of the comparator is "1", so that the state of T flip-flop is changed once after each clock plus, and the seed expanded by LFSR is alternately inverted and shifted into scan path, else the state is remains. As soon as a folding pattern is completely shifted into scan path, the counter and T flip-flop are reset as "1", and the folding counter is activated. This procedure is repeated until the folding counter has cycled through all states and the next seed can be processed.

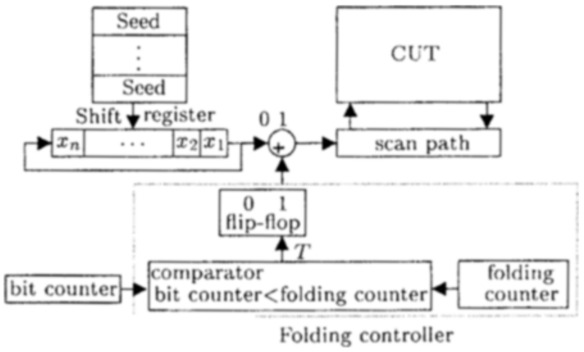

Fig.2. Basic deterministic BIST scheme for reseeding of folding sets.

However, despite its simple and regular structure, the basic architecture of Fig.2 has one serious drawback: the shift register must have the same length as the scan chain, which may be unacceptable for larger circuits. To solve this problem the technique of encoding by seeds provides a good means[10−12]. As proposed in [11,12] the test sets are compressed by systematically exploiting the large number of unspecified bits in deterministic test sets. It is proved that a test pattern $t \in \{0, 1, -\}^n$ with specified bits can be encoded as an initial seed of an LFSR with a very high probability, which is nearly independent of the pattern length n. For decoding such a pattern, the LFSR, which must be implemented anyway for pseudo-random pattern generation, is sufficient. Combining this technique with the architecture of Fig.2, we can encode seeds of folding sets and replace the shift register with a short LFSR, accordingly, the length of the folding seeds to be stored can be shortened considerably. A further reduction of the storage amount for folding seeds is obtained by extending the purely deterministic scheme to a mixed mode scheme which uses pseudo-random patterns to cover the easy-to-detect faults. To simplify the structure of the pattern generator in this scheme, we apply a unique polynomial of LFSR to generate pseudo-patterns and encode the seeds. Moreover, the degree of the LFSR may be smaller than the maximal specified bit $s_{\max}(T) = \max\{s(t)|tT\}$, where T is the deter-

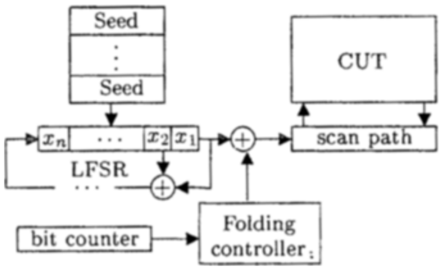

Fig.3. Architecture for mixed mode BIST based on reseeding of folding sets.

ministic test set for the hard-to-detect faults. The resulting architecture, which is the target architecture for the remainder of this paper, is shown in Fig.3.

The generator with an LFSR is merely different from Fig.2. During generating a complete folding set the seed of the LFSR stored in the ROM must be repeatedly loaded after shifting a pattern into the scan path so that for the folding compute of the folding set, the same seed of the folding set can be provided. While the next folding set is computed, a new seed must be loaded. To synthesize and optimize the architecture of Fig.3 for a particular circuit, the following problems have to be solved.

1) A suitable feedback polynomial for a pseudo-random test has to be identified and an efficient encoding scheme for folding set seeds using the same polynomial LFSR has to be determined[17].

2) A minimal number of seeds for the folding sets has to be generated, such that the produced folding set sequences contain the seed-encoded deterministic test set for the hard faults.

For the second problem a more thorough understanding of the nature of folding set sequences is required. Therefore the next section first provides the necessary theoretical background before the complete synthesis procedure and experimental results are presented in the subsequent sections.

3 Folding Sets — Theoretical Background

In order to completely comprehend the folding set, in this section, we introduce formal definitions and some properties of the folding set, and present some new concepts about folding relation, distance and seed. These can help efficiently compact deterministic test patterns.

Definition 1 (Folding Compute). *$x \in \{0,1\}^n$ is a vector $x = x_1x_2\ldots x_i\ldots x_n$. Let $\neg^j x_i$ denote logic NOT j of variable x_i that the logic NOT time is j for the x_i. If j is an even number, we have $\neg^j x_i = x_i$, otherwise $\neg^j x_i = \neg x_i$. Let $F(k,x)$ be a folding compute on k times of the x. Its format is*

$$F(k,x) = \prod_{i=1}^{n} \neg^{\mathrm{inv}(i,k)} x_i \quad (\mathrm{inv}(j,k) = \begin{cases} i, & \text{if } i < k \\ k, & \text{else} \end{cases}) \tag{1}$$

Example 1. $x = 1001, k = 3$, $F(3,x) = \neg^1 1 \neg^2 0 \neg^3 0 \neg^3 1 = 00\underline{10}$

Definition 2 (Folding Set). *If a vector $x \in \{0,1\}^n$ is computed by (1) $n+1$ times to generate $n+1$ vectors, the $n+1$ vectors make up a set. The set is called the folding set. Its format is*

$$F_x = \{x_k | F(k,x), k = 0\ldots n\} \tag{2}$$

Example 2.

$$x = 0110 => F_x = \{0110, \underline{1001}, 1\underline{110}, 11\underline{01}, 110\underline{0}\}$$

Definition 3. (Folding Set Seed). *If $x \in \{0,1\}^n$ is folded to compute $n+1$ times by (1) to generate a folding set F_x, then the x is called a seed of the folding set F_x.*

Definition 4 (Folding Relation). *Assume $x = x_1x_2\ldots x_n$ and $y = y_1y_2\ldots y_n (x \neq y)$ are two elements of a set $T \in \{0,1\}^n$. Let $x \oplus y = z$ of the form $z_1z_2\ldots z_{p-1}z_p\ldots z_{i-1}z_i\ldots z_n$ where $1 \leq i \leq n$, $1 \leq p < i$. If in the z the $z_i \ldots z_n$ bits are the same digital bits (all "0" or "1"), $z_1z_2\ldots z_{p-1} = 00\ldots 0$, and the $z_p\ldots z_{i-1}$ bits are not the same digital adjacent bits (or odd bit is "1" and even bit is "0"), x and y are a pair of folding relation and the relation is denoted as $x\mathrm{F}y$. The bit position i exactly corresponds to the folding computing time in (1).*

Definition 5 (Folding Distance). *If $x, y \in \{0,1\}^n$ are $x\mathrm{F}y$, the bit i in the definition of folding relation is defined as folding distance of the $x\mathrm{F}y$. D denotes the distance.*

Example 3.

Assume $x = 0011, y = 1000$ and $x\mathrm{F}y$. If $z = x \oplus y = 1011$, the bit i is 3 and folding distance D of the $x\text{F}y$ is 3.

Theorem 1. *If $\forall x, y \in \{0,1\}^n$ are a pair of folding relation $x\text{F}y$, then there exists a seed $s \in \{0,1\}^n$ and x, y belong to the folding set F_s.*

Proof. Let $x = x_1x_2\dots x_n$ and $y = y_1y_2\dots y_n (x \neq y)$ be $x\text{F}y$, $z = x \oplus y$ is of the form $(z_1, \dots, z_{p-1}, z_p, \dots, z_{i-1}, z_i, \dots, z_n)$ as defined above, then there are $z_1z_2\dots z_{p-1} = 00\dots0$ and $z_k = z_{k+1} = \dots = z_n$. The folding distance of the $x\text{F}y$ is $D = k$ $(k = 1, \dots, n)$.

We can select a seed

$$s = \prod_{i=1}^{n} \neg^{\text{inv}(i,k)} x_i \tag{3}$$

By (1) we get:

$$F(k, s) = \prod_{i=1}^{n} \neg^{\text{inv}(i,k)} s_i = \prod_{i=1}^{n} \neg^{\text{inv}(i,k)} \neg^{\text{inv}(i,k)} x_i = \prod_{i=1}^{n} \neg^{2*\text{inv}(i,k)} x_i = x \quad \text{by(3)}$$

If z_p is the first non-zero bit in $z_1z_2\dots z_{p-1}z_p\dots z_{k-1}z_k\dots z_n$, where $1 \leq k \leq n$, $1 \leq p < k$, we can get $z_1z_2\dots z_{p-1} = 00\dots0$ and in $z_pz_{p+1}\dots z_{k-1}$ odd bits are "1", even bits are "0" by the definition of folding relation.

Another situation we can compute

$$\begin{aligned}
F(p-1, s) &= \prod_{i=1}^{n} \neg^{\text{inv}(i,p-1)} s_i = \prod_{i=1}^{n} \neg^{\text{inv}(i,p-1)} \neg^{\text{inv}(i,k)} x_i \\
&= \neg^{2*1} x_1 \neg^{2*2} x_2 \dots \neg^{2*(p-1)} x_{p-1} \neg^{p-1+p} x_p \neg^{p-1+p+1} x_{p+1} \dots \neg^{p-1+k-1} x_{k-1} \neg^{p-1+k} x_k \dots \neg^{p-1+k} x_n \\
&= x_1x_2\dots x_{p-1} \neg^{2*p-1} x_p \neg^{2*p} x_{p+1} \dots \neg^{2*p+k-p-2} x_{k-1} \neg^{2*p+k-p-1} x_k \dots \neg^{2*p+k-p-1} x_n \\
&= y
\end{aligned}$$

If the difference $k - p$ is even by the above formula, then $z_{k-1} = 0$ and $z_k \dots z_n = 1\dots1$, else $z_{k-1} = 1$ and $z_k \dots z_n = 0\dots0$ by the definition of folding relation.

As a result of $F(k, s) = x$ and $F(p-1, s) = y$, x and y belong to the folding set of F_s.

From the proof process of the Theorem 1 proof vector x or y can be always gotten via the folding compute of seed s. The seed s of (3) is selected only according to the folding distance of $x\text{F}y$ and vector x or y.

Theorem 2. *If $X = \{x_i | x_i \in \{0,1\}^n, i = 1, \dots, m\}$ is a set and in the set X there exists an element e with which there are folding relations for all other elements and its folding distances with all other elements are the same, then there surely exists a seed $s \in \{0,1\}^n$ and the set X is included in the folding set F_s. We call the element e maximal distance element and the set X the folding subset of the F_s.*

Proof. $\forall x \in X \in \{0,1\}^n (x \neq e)$ there is a pair of folding relation $e\text{F}x$, In terms of Theorem 1 we can obtain a seed $s \in \{0,1\}^n$ and e, x belongs to the folding set F_s. In the proof process of Theorem 1 the seed s can be selected only according to the folding distance of the element e. In the set X all other element folding distances with the element e are the same, then the seeds with all other elements are the same seed s. So the set X is included in the folding set F_s.

Example 4. Set X is $\{x_1 = 0011, x_2 = 1000, x_3 = 1001\}$. In the set X, x_3 is the maximal distance element, as distances of $x_3\text{F}x_1$ and $x_3\text{F}x_2$ are 4, respectively. Seed s generated by element x_3 is "0011". F_s is $\{0011, 1100, 1011, 1000, 1001\}$ and the set $X \subseteq F_s$.

4 The Complete Synthesis Procedure

4.1 Generation of Test Folding Set

4.1.1 Method of Searching Folding Subset

For test set $T \in \{0,1\}^n$ and $T = \{t_0, t_1, \ldots, t_m\}$, according to the judgment condition of folding relation we can search out all folding relations between test vectors in the set T and construct a graph $G(V, E)$. The vertex set V denotes all test vectors of the set T. The edge set E expresses the folding relation between test vectors. The digit on the edge is the folding distance. From G we can get all complete subgraphs. If in the subgraph there exists a maximal distance element, then the subgraph is a folding subset on the basis of Theorem 2. The seed of the folding subset can be generated by the maximal distance element in terms of the proof of Theorem 1.

4.1.2 Cover of Test Set T

Above we can obtain all folding subsets in the set T. From these folding subsets a minimal cover of the set T can be found. The searching minimal cover of the set T can be mapped to solve the problem of set cover. Usually, there are some project methods. Examples are the extraction algorithm[18], the greedy algorithm[19,20] and so on.

Example 5. Test set T is

	x_1	x_2	x_3	x_4
t_1	0	0	1	1
t_2	0	1	1	0
t_3	1	0	0	0
t_4	1	0	0	1
t_5	1	1	1	0

In Fig.3 the folding relation graph of the set T is shown.

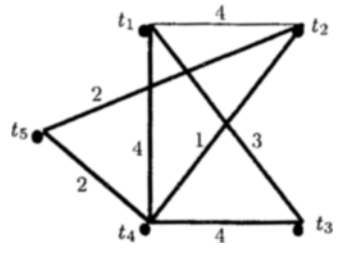

Fig.4. Folding relation graph of set T.

A minimal cover found in Fig.4 is $\{c_1, c_2\}$. The $c_1 = \{t_1, t_3, t_4\}$ and $c_2 = \{t_2, t_4, t_5\}$ are two folding subsets. Their maximal distance elements are $t_4 = 1001$ and $t_5 = 1110$ respectively. Seed s_1 of c_1 is "0011". Seed s_2 of c_2 is "0110". The maximal distance of $\{c_1, c_2\}$ is 4 of c_1.

All test vectors generated are:

$$\boldsymbol{F}_{s1} \Rightarrow \{0011, 1100, 1011, 1000, 1001\}$$
$$\boldsymbol{F}_{s2} \Rightarrow \{0110, 1001, 1110, 1101, 1100\}$$

4.2 Encoding Strategy of Folding Seeds

For a deterministic test set $T \in \{0, 1, -\}^n$, in which not all bits of test patterns are specified, merging the test patterns in the set T into larger folding subsets is with a very high probability. Usually, they contain a very large number of don't-care bits that can be utilized for optimizations and it is easy to extend the folding method for the incompletely specified patterns, moreover the seeds gotten by the folding method can yet include many don't-care bits, with which we can further compress the seed sets to use encoding technique[11,12,17]. Here we have two encoding schemes:

i. At first we find out all folding subsets and select a minimal cover from the deterministic test set T, and then encode the seeds of the folding sets in the minimal cover.

ii. We encode all possible seeds for each pattern of the set T, and then based on the encoded seeds we generate all folding subsets and obtain a minimal cover.

In the first scheme we can obtain a smaller number of folding set seeds than the second one, but the number of don't-care bits in the seed vectors is reduced, as the folding relations among the test patterns in the folding subset restrain some don't-care bits that X are fixed appropriately. Along with the growth of specified bits in the seed vector we must increase the degree of encoding polynomial to satisfy the high probability of encoding, such that the efficiency of compressing seeds is decreased.

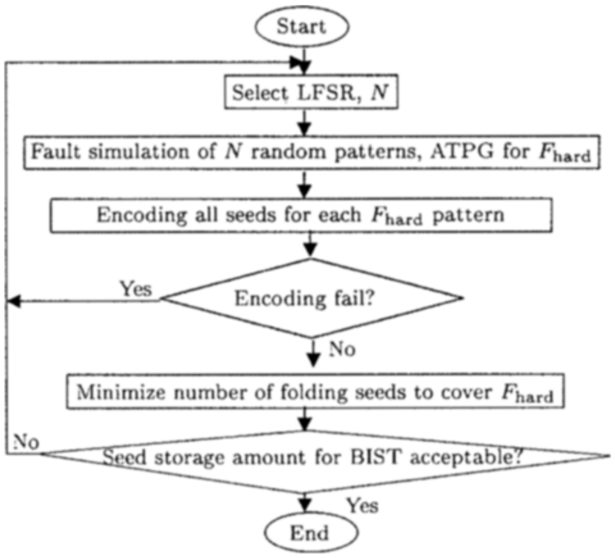
Start
Select LFSR, N
Fault simulation of N random patterns, ATPG for F_{hard}
Encoding all seeds for each F_{hard} pattern
Yes
Encoding fail?
No
Minimize number of folding seeds to cover F_{hard}
No
Seed storage amount for BIST acceptable?
Yes
End

Fig.5. Complete synthesis algorithm.

For encoding seeds of each test pattern of the set T in the second scheme we can decrease significantly the degree of the feedback polynomial (length of LFSR). We use the encoded seeds that all bits are fixed in a seed vector to find all folding subsets. In terms of the folding seed generation method (3) in the proof process of Theorem 1 for an n-bit test vector that is completely specified we may get $n+1$ seeds. Therefore a test vector may be included in $n+1$ folding sets. Moreover if we apply a test vector with s specified bits to generate seeds, we may get $s+1$ seed vectors with s specified bits. The test vector can be generated by the generator in Fig.3 if only we encode a seed vector successfully in the $s+1$ seed vectors. This encoding scheme permits the degree of encoding polynomial to be less than the maximal specified bit $s_{\max}$ with the high probability of encoding. Although in the first scheme there is a smaller number of seeds, the second scheme gains advantage over the first scheme on totally compressing a deterministic test set T. Therefore we select the second scheme as the final encoding scheme.

4.3 Synthesis Procedure Realization

In Fig.5 the complete synthesis procedure for the proposed BIST architecture can be summarized by the flow chart. It basically consists of the following main steps:

1. The feedback polynomial for the LFSR and the desired number N of random patterns are selected. The first N patterns of the LFSR sequence are fault simulated to determine the set F_{hard} of random pattern resistant faults.

2. A set of deterministic test cubes $T \subset \{0,1,-\}^n$ for F_{hard} is generated by ATPG tool, and all possible seed encoding of folding sets are performed applying the algorithms proposed in [17], where the seed width is compressed.

3. The algorithm described above is applied to solving the folding cover problem.

Since the seed storage amount for the proposed BIST scheme may depend on both the choice of the LFSR and the number of random patterns, the complete procedure may be iterated with a different choice of these parameters.

5 Experimental Results

A series of experiments have been performed with the ISCAS-85 and the combinational parts of the ISCAS-89 circuits[22,23]. Only circuits which still have undetected faults after 10000 random patterns are analyzed in further detail. To generate the deterministic test cubes for hard faults a proprietary ATPG tool is used with the option to minimize the number of specified bits.

The first experiment focuses on the basic scheme of Fig.2 without seed encoding. In this case the pattern generator requires a shift register of the same length as the scan path. The LFSR for random pattern generation is chosen of the same degree. Table 2 shows the results.

The names of the circuits and the number of pseudo-primary inputs (lengths of the scan chains) are given in columns one and two. The third column shows the number of folding seeds necessary to guarantee complete fault coverage after 10000 random patterns. The overall number of bits to store the folding seeds is given in the last column labeled with ROM.

Table 2. Results for the Basic Scheme of Fig.2

Circuit	#PPIs	LFSR	Seeds	ROM
s420	34	34	6	204
s641	54	54	2	108
s713	54	54	2	108
s838	66	66	25	1650
s953	45	45	2	90
s1196	32	32	23	736
s1238	32	32	28	896
s5378	214	214	11	2354
s9234	247	247	47	11609
s13207	700	700	27	18900
s15850	611	611	30	18330
s38417	1664	1664	90	149760
s38584	1464	1464	22	32208
c2670	233	233	22	5126
c7552	207	207	43	8901

It can be observed that for all cases a reasonably small number of seeds are sufficient to guarantee complete fault coverage. However, the long scan chains and the long shift registers producing the folding sets lead to a considerable amount of overall storage for larger circuits. To efficiently reduce the storage of seeds relying on seed encoding for folding set seeds with respect to this problem, the experiments are repeated for the scheme of Fig.3. Table 3 shows the results and compares them to the results achieved by competitive approaches published in [11] (HELL95).

Again columns one and two show the names of the circuits and the number of pseudo-primary inputs. The next three columns list the length of LFSRs, the number of encoding polynomials that includes a polynomial of pseudo-pattern generation and the total number of bits to be stored ("ROM(R)") for a mixed-mode BIST scheme based on reseeding of multiple polynomial LFSRs. Columns six, seven, eight and nine are dedicated to the proposed scheme using encoding and folding compression, they contain the length of LFSRs, the number of polynomial, the number of folding seeds and the total number of bits to be stored ("ROM(F)"). The last column presents the reduction as the ration of the storage requirements for the proposed biseed scheme and for the reseeding scheme.

For our comparison we refer to published results for a mixed-mode BIST using 10000 pseudo-random patterns to eliminate the easy-to-detect faults. It can be observed that in almost all circuits the proposed scheme has smaller number of bits to store. Only for the two circuits, s38584 and c7552, the results based on the reseeding of multiple-polynomial LFSRs are superior, but for circuit c7552 the reseeding scheme applies 11 polynomials to encode seeds of LFSRs, therefore these polynomials increase the complexity of encoding and the generator structure. In the proposed scheme for seed encoding and pseudo-pattern generation we only use a feedback polynomial of LFSRs and simplify generator structure. Furthermore, the most of lengths of LFSRs in this scheme are shorter than the reseeding approach, so that the overall hardware cost also contrasts favorably.

Table 3. Comparison of the Proposed Technique to the Reseeding of Multiple-Polynomial LFSRs

		Reseeding HELL95			Biseed FOLDING				
Circuit	#PPIs	LFSR	Poly.s	ROM(R)	LFSR	Poly.s	Seeds	ROM(F)	ROM(F)/ROM(R)
s420	34	20	3	250	18	1	11	198	0.79
s641	54	22	2	183	18	1	4	72	0.39
s713	54	22	2	183	16	1	5	80	0.43
s838	66	36	6	1623	33	1	44	1452	0.89
s953	45	15	4	141	15	1	2	30	0.21
s1196	32	17	4	267	17	1	3	51	0.19
s1238	32	17	4	249	17	1	4	68	0.27
s5378	214	27	3	726	16	1	16	256	0.35
s9234	247	61	8	6923	41	1	124	5084	0.73
s13207	700	24	6	3570	20	1	69	1380	0.39
s15850	611	46	6	6528	30	1	130	3900	0.6
s38417	1664	91	6	24283	42	1	505	21210	0.87
s38584	1464	70	3	3406	49	1	75	3675	1.08
c2670	233	60	5	3412	40	1	37	1480	0.43
c7552	207	100	12	5241	131	1	79	10349	2

6 Conclusions

A new and efficient scheme for the scan-based BIST has been presented. This scheme applies the biseed compression technique, which encodes the folding seeds as the seed of LFSR, considerably reducing the storage amount for the folding seeds. Moreover the simple and regular structure of the pattern generator allows an efficient hardware implementation, such that the overall scheme provides a flexible low cost solution for high-quality BIST.

In this scheme we use a unique polynomial to encode seeds and generate pseudo-patterns, but the polynomial is not the best feedback polynomial for a maximally effective pseudo-pattern test and encoding seed. If we combine an optimum feedback polynomial for pseudo-pattern test with a maximally effective encoding polynomial, we shall obtain better results.

References

[1] Abramovici M, Breuer M, Friedman A. Digital Systems Testing and Testable Design. New York: Computer Science Press (W. H. Freeman and Co.), 1990.

[2] Chen K-T, Lin C-J. Timing driven test point insertion for full-scan and partial-scan BIST. In *Proceedings IEEE International Test Conference,* Washington DC, 1995, pp.506–514.

[3] Savaria Y, Yousef M, Kaminska B, Koudil M. Automatic test point insertion for pseudo-random testing. In *Proceedings of International Symposium on Circuits and Systems,* 1991, pp.1960–1963.

[4] Williams M J Y, Angell J B. Enhancing testability of large-scale integrated circuits via test points and additional logic. *IEEE Transactions on Computers,* January, 1973, C-22(1): 46–60.

[5] Brglez F *et al.* Hardware-based weighted random pattern generation for boundary-scan. In *Proceedings of IEEE International Test Conference,* Washington DC, 1989, pp.264–274.

[6] Strle A, Wunderlich H-J. TESTCHIP: A chip for weighted random pattern generation, evaluation, and test control. In *IEEE Journal of Solid State Circuits,* July, 1991, 26(7): 1056–1063.

[7] Tsai K-H, Hellebrand S, Marek-Sadowska S, Rajski J. STARBIST: Scan autocorrelated random pattern generation. In *Proceedings of ACM/IEEE Design Automation Conference,* Anaheim, CA, June 9–13, 1997.

[8] Wunderlich H-J. Self test using unequiprobable random patterns. In *Proc. IEEE 17th International Symposium on Fault-Tolerant Computing,* FTCS-17, Pittsburgh, 1987, pp.258–263.

[9] Savir J, McAnney William H. A multiple seed linear feedback shift register. *IEEE Transactions on Computers,* February, 1992, 41(2): 250–252.

[10] Hellebrand S, Rajski J, Tarnick S, Venkataraman S, Courtois B. Built-in test for circuits with scan based on reseeding of multiple-polynomial linear feedback shift registers. *IEEE Transactions on Computers,* February, 1995, 44(2): 223–233.

[11] Hellebrand S, Reeb B, Tarnick S, Wunderlich H-J. Pattern generation for a deterministic BIST scheme. In *Proceedings of IEEE/ACM International Conference on CAD-95,* San Jose, CA, November, 1995, pp.88–94.

[12] Koenemann B. LFSR-coded test patterns for scan designs. In *Proceedings of European Test Conference,* Munich, 1991, pp.237–242.

[13] Touba N A, McCluskey E J. Altering a pseudo-random bit sequence for scan-based BIST. In *Proceedings of IEEE International Test Conference,* Washington DC, 1996, pp.167–175.

[14] Kiefer G, Wunderlich H-J. Using BIST control for pattern generation. In *Proceedings IEEE International Test Conference,* Washington DC, November, 1997, pp.347–355.

[15] Wunderlich H-J, Kiefer G. Bit-flipping BIST. In *Proceedings of ACM/IEEE International Conference on CAD-96 (ICCAD96),* San Jose, CA, November, 1996, pp.337–343.

[16] Chakrabarty K, Murray B T. Design of build-in test generator circuits using width compression. *IEEE Trans. CAD,* Oct., 1998, 17: 1044–1051.

[17] Hellebrand S, Wunderlich H-J, Hertwig A. Mixed-mode BIST using embedded processors. *Journal of Electronic Testing: Theory and Applications — JETTA,* February/April, 1998, 12(1/2): 127–138.

[18] Breuer M A. Design Automation of Digital Systems. New Jersey: Computer Science Press, 1972.

[19] Slavik P. A Tight Analysis of the Greedy Algorithm for Set Cover. In *28th Annual ACM STOC'96.*

[20] Chvatal V. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research,* 1979, (4): 233–235.

[21] Li Xiaowei, Paul Y S Cheung, Hideo Fujiwara. 'LFSR-based deterministic TPG for two-pattern testing. *Journal of Electronic Testing: Theory and Application,* Kluwer Academic Publisher, 2000, 16(5): 419–426.

[22] Brglez F *et al.* Accelerated ATPG and fault grading via testability analysis. In *Proceedings IEEE Int. Symp. on Circuits and Systems,* Kyoto, 1985.

[23] Brglez F, Bryan D, Kozminski K. Combinational profiles of sequential benchmark circuits. In *Proc. IEEE Int. Symp. on Circuits and Systems,* 1989, pp.1929–1934.

LIANG Huaguo received the BEng and the MEng degrees in computer science and applications in 1982 and 1989, respectively, from Hefei University of Technology, China. From 1982 to 1998 he worked at Hefei University of Technology, where currently he is an associate professor and also head of the Computer Application Division. Since 1998 he has been working as a guest research fellow at the Division of Computer Architecture, University of Stuttgart, Germany. His research interests include built-in self-test, design automation of digital systems, ATPG algorithms, and distributed control.

Sybille Hellebrand received her diploma in mathematics from the University of Regensburg, Germany in 1986. In 1986 she joined the Institute of Computer Design and Fault Tolerance, University of Karlsruhe, where she received the Ph.D. degree in 1991. Then she was a postdoctoral fellow at the TIMA/IMAG-Computer Architecture Group, Grenoble, France. From 1992 to 1996 she worked as an assistant professor at the University of Siegen, Germany. After a sabbatical stay at Mentor Graphics Corp., Wilsonville, Oregon, she joined the Division of Computer Architecture at the University of Stuttgart, Germany in 1997. Since October 1999 she has been a full professor for Applied Computer Science at University of Innsbruck, Austria. Her main research interests include BIST for high quality applications and synthesis of testable systems.

Hans-Joachim Wunderlich received the Dr. rer. nat. (Ph.D.) degree in computer science from the University of Karlsruhe in 1986. There he was head of a research group on automation of circuit design and test from 1986 to 1991. From 1991 to 1996 he was a full professor for computer science at the University of Siegen. Since October 1996 he has been head of the Division for Computer Architecture at the University of Stuttgart.

He has been a member of the program committee of numerous conferences and a reviewer of research proposals submitted to NSF and NATO. Within the European projects EUROCHIP and EUROPRACTICE he has been a lecturer for courses on VLSI design and test. Prof. Wunderlich is the author and co-author of three books and over 80 papers in the field of test, synthesis, and fault tolerance of digital systems.