STARBIST: Scan Autocorrelated Random Pattern Generation

K.H. Tsai[1], S. Hellebrand[2], J. Rajski[3], M. Marek-Sadowska[1]

[1]Department of Electrical and Computer Engineering, University of California, Santa Barbara, CA, USA
[2]Institute of Computer Structures, University of Siegen, Siegen, Germany
[3]Mentor Graphics Corporation, Wilsonville, OR, USA

Abstract. This paper presents a new scan-based BIST scheme which achieves very high fault coverage without the deficiencies of previously proposed schemes. This approach utilizes scan order and polarity in scan synthesis, effectively converting the scan chain into a ROM capable of storing some "center" patterns from which the other vectors are derived by randomly complementing some of their coordinates. Experimental results demonstrate that a very high fault coverage can be obtained without any modification of the mission logic, no test data to store and very simple BIST hardware which does not depend on the size of the circuit.

1 Introduction

The scan-based Built-In-Self-Test (BIST) schemes, which rely on full/partial scan design for testability, use Linear Feedback Shift Registers (LFSRs) as generators of pseudo-random patterns, and employ Multiple-Input Shift Registers (MISRs) as test response compactors, are becoming widely adopted due to their conceptual simplicity and ease of implementation. These schemes naturally extend scan-based test methodology with ATPG generated patterns applied from an ATE to BIST. The additional hardware required to implement the BIST hardware is not large and there are commercially available tools for its automated synthesis. The designers welcome the advantages that BIST offers provided that the quality of the test is as good as that of the state-of-the-art ATPG. However, very high fault coverage usually cannot be easily achieved without addressing the problem of pseudo-random pattern resistant faults.

There are two types of methods proposed to solve this problem, one by modifying the mission logic to make the resistant faults pseudo-random testable, and the other by increasing the ability of the test pattern generators to target pseudo-random pattern resistant faults. Test point insertion is the main technique of the first type [16, 7, 11, 12, 4, 13]. Low hardware overhead and no extra memory needed to achieve very high fault coverage are the main advantages of this technique. However, the modification of the mission logic may not be acceptable by designers due to the possible performance degradation and its impact on the design flow. In a reusable core defined by a netlist such modification of the mission logic may simply not be possible.

The second type of methods, which contain reseeding [8, 6], weighted random patterns [17, 3, 9], pattern mapping and other techniques [1, 5, 14, 10], focus on the characterization of complete test sets. These methods extend the conventional pseudo-random test pattern generators based on LFSRs by biasing the patterns to cover random pattern resistant faults. They achieve very high fault coverage in an acceptable number of patterns, but they need extra memory to store seeds, weight sets or additional logic has to be implemented. The extra memory needed is usually proportional to the size of the circuit and is very expensive for large designs.

In this paper we propose a new BIST structure to generate high quality test patterns without extra memory to store deterministic patterns, seeds or weights. Unlike previous methodologies, this new structure doesn't use the conventional LFSR directly to generate pseudo-random patterns. Our method doesn't modify the mission logic, thus there is no performance degradation and no need for resynthesis. This new approach is based on an experimental observation that a very high fault coverage can be obtained by a small number of clusters of test vectors. Each cluster contains one parent test vector in the center and a number of children derived from the parent by complementing some number of coordinates in a pseudo-random manner. The parent vector is computed by a specialized ATPG algorithm capable of targeting many pseudo-random pattern resistant faults.

The implementation makes use of scan order, polarity between the neighboring cells, and control points inserted between scan cells. With these features the scan chain has the properties of a ROM capable of encoding several parent test vectors. The children are generated by a simple hardware capable of complementing coordinates of parents at random and no additional memory is required to enhance the capabilities of the generator. It is demonstrated experimentally that the entire test information required for high quality test can be encoded in the scan chain.

In the next section we discuss a simple example, well-known in the context of weighted random techniques. We propose the solution to this example which demonstrates the motivations of this paper. The principle of a compressed hierarchical structure of the test patterns is presented in the third section. Section 4 gives a detailed description of the required elements including the control signals and the encoding technique for a fixed number of deterministic patterns. An overview of the complete BIST architecture and the synthesis algorithm follow in sections 5 and 6. Section 7 presents the experimental results, and finally the paper is concluded in section 8.

2 A Motivational Example

In this section the basic principles of the proposed approach will be introduced with the help of the AND-OR circuit shown in Figure 1. This circuit contains random pattern resistant faults and has been used to illustrate basic concepts of weighted random patterns [15]. To test the stuck-at-zero fault at node *y*, all

*This research was supported in part by the NSF grant MIP 9419119

inputs must be set to 1, and if uniformly distributed pseudo-random patterns are applied, the detection probability is 2^{-32}, which results in an unacceptable test length. If weighted random patterns are used, setting each input to 1 with the probability of $31/32$, the same fault can be detected with a probability of $(31/32)^{32}$ = 0.362, implying relatively short test time. At the same time each of the stuck-at-one faults on the inputs of the AND gate is detected with a probability of $1/32 \cdot (31/32)^{31}$ = 0.01168, which means that on average 86 vectors are required to detect it. However, the stuck-at-one fault at the output z requires the complementary weight $1/32$, and to ensure a complete test of the circuit, the two different weights have to be stored for each circuit input.

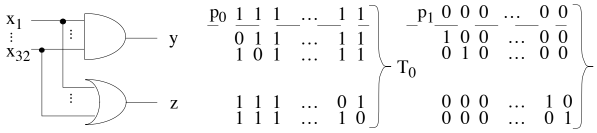

Figure 1: 32-input AND-OR circuit and it's test set.

On the other hand, a complete test is provided by the vectors shown in Figure 1 which exhibit a high degree of regularity. The test set is composed of two parts T_0 and T_1, each of which is characterized by a basic pattern p_0 and p_1, respectively, being referred to as the parent patterns. The remaining patterns $T_i \setminus \{p_i\}$, $i = 1, 2$, can be derived from the parent p_i by complementing a single bit position and are therefore called the children of p_i. Furthermore, p_1 is obtained by complementing p_0. A proper exploitation of this regular structure allows a very simple scan-based BIST as shown in Figure 2. To obtain the targeted parent pattern p_0 of Figure 1, a test point is inserted into the scan chain, such that during scan-in the constant value "1" is available at the input of the scan cell corresponding to x_1. Because of the structure of the test set the approach can rely on a "predict & correct" scheme for the remaining patterns. The exceptions from the parent p_0 are generated by a simple hardware unit and no further storage effort is required.

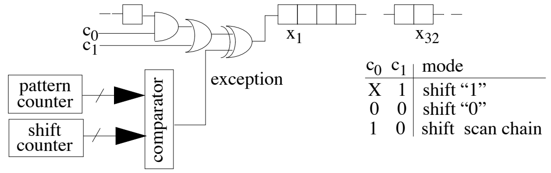

Figure 2: Scan-based BIST for the circuit of Figure 1.

Moreover, randomizing the generation of the children can further simplify the BIST hardware while retaining a high fault coverage with a relatively small number of patterns. If each bit of child is flipped from $p_1(p_0)$ with probability $\alpha = 1/32$, then the probability to generate a particular child within Hamming distance 1 from the parent cube $p_1(p_0)$ is given by $1/32 \cdot (31/32)^{31}$ = 0.01168, and on the average 348 random children provide all patterns within Hamming distance 1. The BIST implementation can be easily generalized to deal with arbitrary parent patterns. If inversions between scan cells are use appropriately, then any given pattern p and its complement $\overline{p}$ can be "hardwired" in the scan chain.

This example illustrates two important properties of complete test sets for many non-trivial circuits. First, there is a small number of clusters, each characterized by a parent cube (or center) which has a large number of highly effective test vectors in its close vicinity. Second, there is a great deal of regularity between the centers.

3 The Star Test Principle

The objective of the proposed approach is to develop a scan-based BIST guaranteeing a very high fault coverage while avoiding control points in the mission logic as well as external test data storage. Furthermore, the complexity of the BIST pattern generator should not depend on the circuit size. The presented solution assumes that a single scan chain will be inserted and that there are no restrictions with respect to the ordering of the cells. Also, it is assumed that the scan cells can be connected with arbitrary inversions between them. In this scenario, the randomized pattern generation scheme of section 2 is adapted as follows: ATPG is used to determine a few powerful parent cubes which can be efficiently combined with randomly derived children. The relations between parent cubes will be exploited to regenerate the whole set of parents from one basic cube encoded in the scan chain. The BIST implementation will then rely on an appropriate scan ordering, interscan test points, the use of inversions between cells and an exception generator.

The proposed strategy of pattern generation will be referred to as star test and is defined more precisely as follows: A test pattern c, which is generated from a pattern t by randomly inverting the components of t with probability α, is called a ***type α (random) child of t.*** The average Hamming distance between an n-bit test pattern t and its type α children is $n \cdot \alpha$, and therefore a set $T^*(t, \alpha, n_c)$ of n_c type α children is called a ***star test around t with n_c type α children***. Finally, a test set T is called a star test, if it is obtained as the union of subsets $P, T_1, \ldots, T_k$, where $P = \{p_1, \ldots, p_k\}$ is a set of deterministic cubes and for each i, $i = 1, \ldots, k$, T_i consists of one or more star tests around p_i.

To elucidate the high fault detection potential of star tests, Figure 3 shows the results of an empirical analysis performed for the benchmark circuit s38417 [3]. In this experiment ATPG was used to generate a compact test set with 70 cubes for the random pattern resistant faults remaining after 32K pseudo-random patterns. Each of the 70 cubes served as parent of 2K children of type 0.5 and 2K children of type 0.25. First, 4K of ordinary pseudo-random patterns were applied and resulted in 89.34% coverage, each parent on average covers an additional 1.04% to 90.38% (light bars). When each parent with its 4K children were applied, they yield average 96.34% coverage (dark bars). In other words, 4K of the vectors derived from a parent cover 5.96% more faults than 4K of pseudo-random patterns.

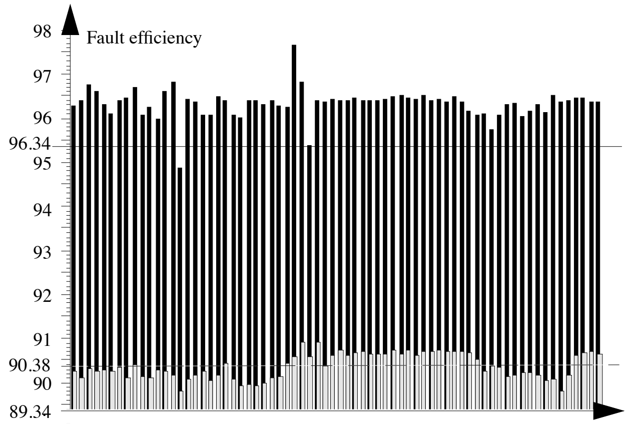

Figure 3: Fault efficiency of star tests compared to conventional pseudo-random patterns for circuit s38417.

The results clearly show that the concept of generating stars around the deterministic cubes is very powerful. The star tests were able to detect a large portion of faults that were neither covered by the same number of pseudo-random patterns nor by the parent deterministic pattern. Furthermore, since only a single test cube together with its children can already provide more than 97% fault efficiency, it is very likely that a few clusters are sufficient to ensure complete or very high fault coverage. Figure 4 shows the ideal structure of a complete star test.

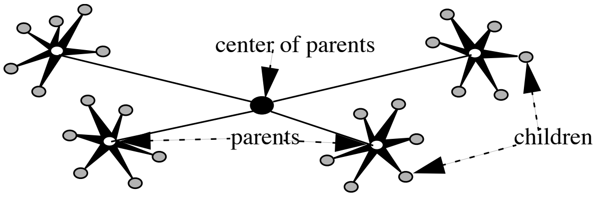

Figure 4: Structure of a star test.

The test set is hierarchically composed of two levels of "stars". The first level star consists of several parent patterns, each of which is obtained by ATPG, and one virtual center p_0 which is determined by the regularity analysis described in section 4. Each of the parent cubes then forms the center of a second level star with children characterized by their flipping probability α.

4 Regularities and Basic Hardware Elements

In this section the correspondence between regular structures in the star tests and the components of an efficient scan-based BIST implementation will be described. Furthermore, it will be explained how these structures can be extracted from a given test set. Since our experiments demonstrate that four deterministic parents are sufficient to achieve very high fault coverage, the pattern generator described in this section will focus on at most four parent patterns.

a) Generation of parent cubes

To study the relations between the parent cubes, the set P is stored as an $m \times n$-matrix with rows corresponding to test cubes and columns corresponding to the inputs of the combinational part of the circuit. Possible entries are "0", "1" or "X" (don't care). The proposed technique identifies columns and submatrices in the matrix P which can be reproduced from one basic cube using simple hardware elements. For this purpose the columns of P are classified into don't care, monotonic, regular and random columns as explained below.

Don't care columns only contain the "X"-entries and therefore pose no encoding problem. Monotonic zero (one) columns are defined as columns with all entries being "0" ("1") or "X" and can be trivially derived from a single cube with the corresponding entry "0" ("1"). If all entries of a column except one are "0" ("1") or "X", this column is referred to as regular column with the background value "0" ("1"), and submatrices consisting of several adjacent regular columns, such that the exceptions are located on the main diagonal are called regular rectangles. Figure 5 shows a regular rectangle with background (1, 1, 1, 0) as an example.

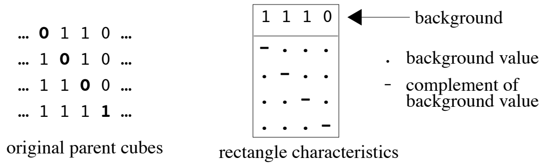

Figure 5: Regular rectangle.

To reproduce this rectangle during BIST the simple hardware structure sketched in Figure 6 is sufficient. The background pattern (1, 1, 1, 0) is "stored" by introducing an inversion between the last two scan cells and by inserting a test point providing a constant "1" (as explained in section 5, the required constant can always be taken from the preceding scan cell). The "envelop" signal ENV_4 ensures that this control point can only be activated during the appropriate shift cycles. Generally, if the scan section to be loaded by this control point has length n and the rectangle corresponds to scan cells k through l, then the envelop signal must be turned on during shift cycles (n+1-l) through (n+1-k). The actual pattern to be loaded is determined by the waveform signal WV_4 which combines the envelop with the signal EQU generating the desired exceptions. The exact timing of both signals is shown in the included diagram.

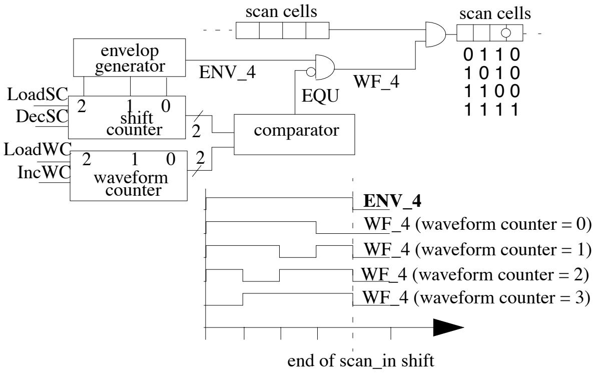

Figure 6: Hardware and timing diagram for a regular rectangle of size four.

Another type of submatrix in P allowing a simple BIST implementation is one whose rows can be partitioned into a regular rectangle and rows representing the inverted background of the regular part as illustrated in Figure 7.

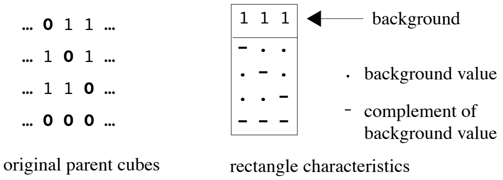

Figure 7: Semi-regular rectangle.

To regenerate such submatrices, which are also referred to as semi-regular rectangles, only a small modification must be added to the architecture of Figure 6 which will be shown later in Figure 11 as part of the complete architecture.

To implement several rectangles their backgrounds can be combined to one large background vector, and one control point is sufficient to reproduce this vector during BIST. The hardware to generate envelop and waveform signals must be modified accordingly. In general, there is a trade-off between the complexity of the hardware necessary to drive one control point and the number of control points inserted. If only rectangles of the same width are combined, then the hardware can be kept simple and the overall costs low.

Rectangles of different widths can be adjusted by inserting additional don't care columns. As an example Figure 8 shows the BIST scheme and the timing diagram for the control signals of four regular rectangles, each of width four. The diagram is similar to the one shown for a single rectangle on Figure 6, except that the envelop signal ENV_16 is enabled for a period of 4·4 = 16 clock cycles and the waveforms repeat periodically. To accomplish this timing, the comparator has only the two least significant bits of the shift counter and the waveform counter (WC) as inputs.

It is certain to define other regular structures which also lead to a relatively simple BIST hardware. Within the framework of this paper, however, we restrict ourselves to the regular and semi-regular rectangles introduced above. Columns of the matrix P which do not fall into the category of monotonic, don't care, regular columns or are not contained in semi-regular rectangles are called random columns. According to our empirical experience (cf. section 7), however, in many cases three or four parent cubes are sufficient, and in this case random columns do not occur. Due to the paper size limitations, we give the following property without a proof.

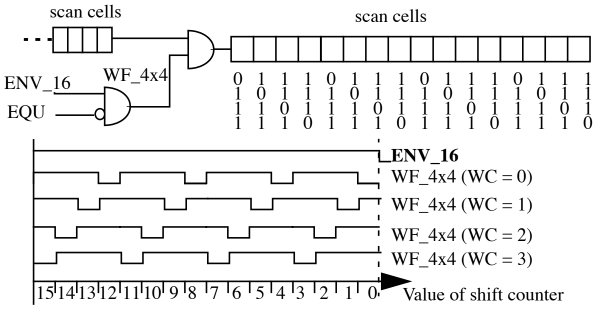

Figure 8: The control logic and timing diagram of four compatible regular rectangles

Property If the matrix P does not have more than four rows and the number of don't care columns is sufficiently high, then a subset of columns can be partitioned into regular and semi-regular rectangles and the remaining columns are either monotonic or don't care columns.

b) Regularity analysis for parent cubes

In general, the matrix P produced by ATPG does not necessarily allow an optimal partitioning into (semi-) regular, monotonic, don't care and random columns. However, since it is assumed that the scan order can be freely determined, it is possible to reorder the columns. For combinational tests the patterns can be applied in an arbitrary order, which also allows row permutations. Furthermore, don't care columns can be inserted to build regular or semi-regular rectangles. To determine the best row and column order we propose a heuristic algorithm proceeding in three steps (cf. Figures 9 and 10).

Original patterns:

```
col. . . . . . . a . b c . . . . . . . d e f g h i j k l m n o p q r s . .
p1  X X X X X X X 1 X X X X X X X X X X X 1 1 0 0 0 1 1 0 0 1 0 0 0 X X
p2  X X X X X X X X X X X X X X X X X X 0 1 1 1 0 0 1 0 0 0 1 1 1 0 0 X X
p3  X X X X X X X X X X X X X X X X X X 0 1 1 0 0 1 1 0 0 0 1 1 1 0 0 X X
p4  X X X X X X X X X 0 1 X X X X X X X 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 X X
```

Step1:

Classify the columns into five different types: regular, semi-regular, monotonic 0's, monotonic 1's and don't care columns.

```
col|e i n h k o q|j l m|b d r s|a c f g p|. . . . . . . . . . . . . . . .
 p1|X 0 0 1 0 0 0|0 1 1|X X 0 0|1 X X 1 1|X X X X X X X X X X X X X X X X
 p2|0 0 0 1 1 1 1|0 0 0|X X 0 0|X X 1 1 1|X X X X X X X X X X X X X X X X
 p3|0 0 0 0 1 1 1|1 0 0|X X 0 0|X X 1 1 1|X X X X X X X X X X X X X X X X
 p4|1 1 1 1 1 1 1|1 1 1|0 0 0 0|X 1 1 1 1|X X X X X X X X X X X X X X X X
       regular   | semi|  0's  |   1's   |          Don't care
```

Figure 9: Original parent cubes and step 1 of the regularity analysis for s1238.

Instead of giving the exact description of the algorithm, we show an example to demonstrate the analysis procedures. In the first step (cf. Figure 9) the columns are grouped into the five types as defined before: regular, semi-regular, monotonic, don't care and random. After collecting this information, the regular columns are rearranged to form regular rectangles (cf. Figure 10). Don't cares in monotonic and in don't care columns may be changed to "0" or "1" to obtain additional regular columns needed for the completion of regular rectangles. Also, don't care entries in the regular columns are assigned to the corresponding background values. Similarly, semi-regular rectangles are built making use of don't care and monotonic columns. The quality of the result may depend on the number of available don't care columns and the distribution of don't cares in monotonic columns. For all the circuits we analyzed the number of don't care columns and monotonic columns are sufficient to provide good results (cf. section 7).

Step2:

Rearrange the order of columns and properly insert don't care columns among (semi-) regular columns, such that they can form (semi-) regular rectangles.

```
col|k . h e o . . i q . . n|l . j m . .|b d r s|a c f g h|. . . . . . . .
 p1|0 X 1 X 0 X X 0 0 X X 0|1 X 0 1 X X|X X 0 0|1 X X 1 1|X X X X X X X X
 p2|1 X 1 0 1 X X 0 1 X X 0|0 X 0 0 X X|X X 0 0|X X 1 1 1|X X X X X X X X
 p3|1 X 0 0 1 X X 0 1 X X 0|0 X 1 0 X X|X X 0 0|X X 1 1 1|X X X X X X X X
 p4|1 X 1 1 1 X X 1 1 X X 1|1 X 1 1 X X|0 0 0 0|X 1 1 1 1|X X X X X X X X
```

Step3:

Replace the don't cares in (semi) regular rectangles and monotonic columns by proper values.

```
bg| 1 1 1 0|1 1 1 0|1 1 1 0|0 0 0|0 0 0|0 0 0 0 1 1 1 1 1 X X X X X X X X
p1| - . . .|- . . .|- . . .|- . .|- . .|. . . . . . . . . . . . . . . . .
p2| . - . .|. - . .|. - . .|. - .|. - .|. . . . . . . . . . . . . . . . .
p3| . . - .|. . - .|. . - .|. . -|. . -|. . . . . . . . . . . . . . . . .
p4| . . . -|. . . -|. . . -|- - -|- - -|. . . . . . . . . . . . . . . . .

   regular rectangles          semi-regular          .  center value
                                rectangles           -  complement of
                                                        background value
```

Figure 10: Steps 2 and 3 of the regularity analysis for circuit s1238.

5 The Complete Architecture

To give a complete description of the STAR-BIST technique, we use the test cubes of the previous example and show how the scan chain and the control signals are connected and synchronized. The scan chain is ordered such that it starts with cells corresponding to regular rectangles. These are followed by cells corresponding to semi-regular rectangles, monotonic and don't care columns as shown in Figure 11.

The parent patterns determined by the regularity analysis of Figure 10 have been rearranged such that all rectangles have width four and can share the same waveform signal. Furthermore, an additional semi-regular rectangle has been constructed from don't cares columns, which makes it possible to combine three regular and three semi-regular rectangles to groups being driven by the same waveform signal WF_4×3. This extra adjustment reduces the area overhead, but will be possible only if the number of don't care or compatible monotonic cells is large enough, which is true for this case. The first control point used to generate the regular columns is inserted at the first scan cell to guarantee a constant scan-in value ("1" in the example). The scan cells are connected together by properly switching between positive and negative polarity to form the center pattern. The waveform signal for the regular rectangles (WF_4×3) generates the parent values when Child_EN is "0". When Child_EN is "1", the *n*-input AND gate fed by the LFSR synthsizes the output signal “1” with probability $\alpha = 2^{-n}$ to generate type α children.

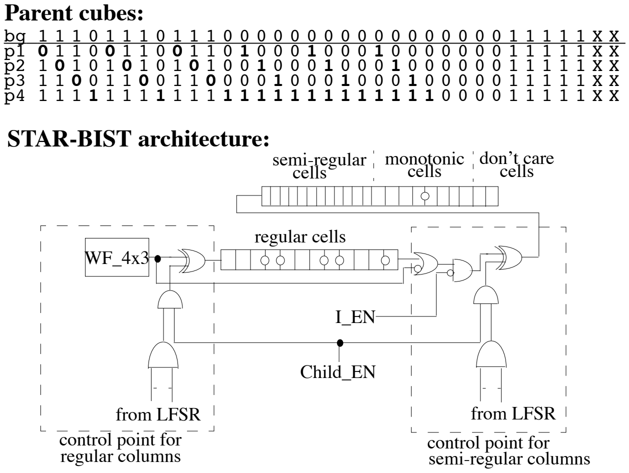
Parent cubes:
bg 1 1 1 0 1 1 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 X X
p1 0 1 1 0 0 1 1 0 0 1 1 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 1 1 1 1 X X
p2 1 0 1 0 1 0 1 0 1 0 1 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 1 1 1 1 1 X X
p3 1 1 0 0 1 1 0 0 1 1 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 1 1 1 1 1 X X
p4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1 1 X X
STAR-BIST architecture:
semi-regular cells
monotonic cells
don’t care cells
regular cells
WF_4x3
I_EN
Child_EN
from LFSR
from LFSR
control point for regular columns
control point for semi-regular columns

Figure 11: The final scan chain and control logic for circuit s1238.

The next control point used to generated semi-regular columns is inserted between the first semi-regular cell and the last regular cell. Since the background value of the last regular cell is “0”, which indicates that the constant “0” is shifted into the semi-regular cells when the first control point is disabled, an OR gate is chosen and the waveform signal (WF_4×3) is complemented to ensure the controlling value being generated at the conflict positions. To guarantee that the last regular cell will always shift the required constant value to the semi-regular cells during the scan-in mode, the regular cells need to be initialized to this value, which can be done by applying extra shift clocks when scanning in the first pattern. This initialization does not make the test application of our technique longer than for a traditional scan-based BIST, since the number of clock cycles for scan-in generally is smaller for the proposed technique than the traditional scan-based BIST. Other logic of the control point is similar to the first one, except one more signal I_EN is added to inverse the scan_in value at the last phase.

6 Synthesis Algorithm

To synthesize the BIST hardware for a star test as described in the previous sections two tasks have to be solved. Firstly, a set P of parent cubes and flipping probabilities $\alpha_1, \ldots, \alpha_k$ must be determined, such that the resulting star test with n_c children of type $\alpha_1, \ldots, \alpha_k$ around each parent achieves a sufficiently high fault coverage. To support a hardware optimal BIST implementation the number of parent cubes should be as small as possible, preferably not exceeding four, because in this case a partitioning into regular structures can be guaranteed (cf. section 4). Secondly, for the set of parents P an appropriate scan ordering must be found allowing to partition the columns of P into (semi-) regular rectangles and monotonic and don’t columns. In doing so, the number of required control points and different envelop and waveform generators is to be minimized. To solve these tasks the algorithm shown in Figure 12 has been developed, which takes as input the combinational part of the target circuit C and a fault list F (if possible all redundant faults are removed from F).

The algorithm starts with fault simulating N pseudo-random patterns to identify the random pattern resistant faults $F_{hard} \subset F$. It should be noted that this step is only performed to calculate F_{hard} and that during the actual BIST no pseudo-random patterns will be applied. If $F_{hard} \neq \varnothing$, the parameters n_c and k for the star test are selected, where k denotes the number of different types of children generated around each cube and n_c is the number of children to be applied of each type. Since patterns generated for random pattern resistant fault usually are very efficient, the set F_{hard} is used as target fault list F_{target} during the first iteration of the ATPG-loop (steps 3 through 6) which follows to determine the set of parent cubes P.

In each iteration of this loop a complete test set T for F_{target} is generated by ATPG, and since we try to find very powerful parent cubes while retaining a high degree of freedom for the remaining synthesis steps, an ATPG tool is used, which can perform dynamic pattern compaction and minimize the number of specified bits. For each cube in T don’t cares entries are then fixed to the majority value found in the corresponding components of the remaining cubes. To find the best flipping probabilities $\alpha_1, \ldots, \alpha_k$ a small sample from T can be used and fault simulated together with different types of children. In general, the best values found in our experiments ranged from $\frac{1}{32}$ to $\frac{1}{2}$. The complexity of the algorithm can be reduced by fixing the values for $\alpha_1, \ldots, \alpha_k$ and skipping step 4 in the following iterations. Our experiments showed only slightly different results when doing so.

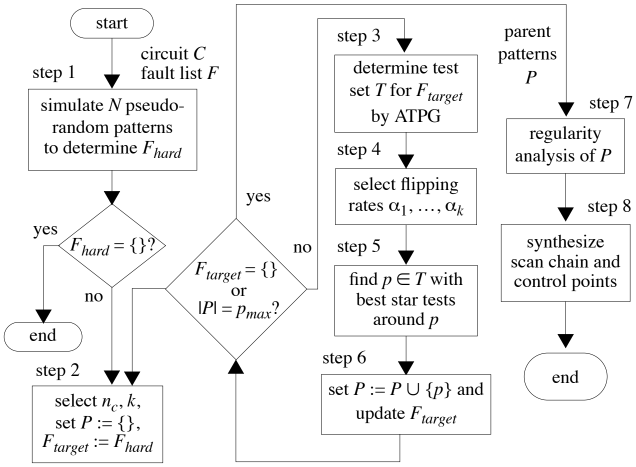
start
step 1
circuit C
fault list F
simulate N pseudo-random patterns to determine F_{hard}
yes
$F_{hard} = \{\}$?
no
end
step 2
select n_c, k, set $P := \{\}$, $F_{target} := F_{hard}$
yes
no
$F_{target} = \{\}$ or $|P| = p_{max}$?
step 3
determine test set T for F_{target} by ATPG
step 4
select flipping rates $\alpha_1, \ldots, \alpha_k$
step 5
find $p \in T$ with best star tests around p
step 6
set $P := P \cup \{p\}$ and update F_{target}
parent patterns P
step 7
regularity analysis of P
step 8
synthesize scan chain and control points
end

Figure 12: Flow chart of the STAR-BIST synthesis algorithm.

After selecting the children’s types, each cube in T together with its $k \cdot n_c$ children is fault simulated against the set F_{target} (in the first iteration against the entire fault set F), and the best cube p is selected and added to P. Then the target fault list is updated to all faults in F which are not covered by the star test determined so far. Steps 3 to 6 are repeated until a sufficiently high fault coverage is reached or the number of parent cubes reaches the user-defined limit p_{max}. As final step, the regularity analysis explained in section 4 is performed for P, which decides the configuration of the scan chain, the control points and the control signals as described in the previous section.

7 Experimental Results

The synthesis algorithm of section 6 has been applied to the random pattern resistant circuits of the ISCAS’85 and ISCAS’ 89 benchmark suite. Table 1 shows the results for a four phase star test ($p_{max} = 4$) with 2K type 0.25 and 2K type 0.5 children in each phase. To characterize the circuits in columns 2 through 4 the number of collapsed irredundant faults and the fault efficiency FE are given. The columns titled φ1 to φ4 show the fault efficiency

FE achieved after each phase of the star test. Note that the star test does not use pseudo-random patterns in advance and targets the entire set of irredundant faults. The results show that the selected types of children 0.5 and 0.25 provide a high quality test for all the circuits. In general, using only the conflict rate 0.25 resulted in a fairly good coverage in the cases we investigated. The type 0.5 children were selected additionally to cover the easy to detect faults, because these patterns can be considered as equivalent to equi-probable pseudo-random patterns. In fact, our results tell us that these type 0.5 children work even better than pseudo-random patterns in detecting the easy faults.

Table 1: Results for a four-phase STAR-BIST (2048 children of type 0.25 and 2048 children of type 0.5 are applied in each phase)

Circuit	#target faults	32Kp.r. patterns	φ1	φ2	φ3	φ4
c2670	2478	97.49%	97.22%	99.27%	99.88%	99.96%
c7552	7417	96.00%	98.50%	99.54%	99.72%	99.81%
s838.1	933	80.72%	80.72%	97.32%	99.25%	99.58%
s5378	4681	99.51%	98.72%	99.72%	99.96%	100%
s9234	6641	90.84%	93.92%	98.67%	99.50%	99.87%
s13207	10340	97.93%	95.23%	98.90%	99.94%	100%
s15850	11564	95.44%	97.57%	99.89%	99.99%	100%
s38417	33039	95.68%	98.09%	99.46%	99.79%	99.94%
s38584	34945	97.49%	96.97%	99.60%	99.93%	99.99%

Though table 1 shows the results of four phases, the fault coverages are already very high after three phases, so the last phase can be eliminated to further reduce one control point (no semi-regular rectangles in three phases star test parents). The length of LFSR used to generate the probability α is 32 bits.

8 Conclusions

This paper presents a novel scan-based BIST technique which gives high fault coverage and no performance degradation in addition to full scan and low area overhead. The test application time is also reduced since the number of the test cubes is significantly smaller than in other existing techniques. The other benefit of STARBIST is that the technique does not modify the mission logic, so no logic resynthesis is required. The control point(s) are added between the scan cells, not inside the mission logic as the current test point insertion techniques do. Though our experiments are based on the stuck at fault model, this technique could be applied to other fault models used for I_{DDQ} test, memory fault test, and delay fault test.

9 Acknowledgment

The authors are very grateful to Rob Thompson for the helps of ATPG in the experiments. The insightful comments from Chien-Chung Tsai and the helps from Nagesh Tamarapalli are also highly appreciated.

References

[1] S. B. Akers, W. Jansz: Test Set Embedding in a Built-in Self-Test Environment; Proc. IEEE Int. Test Conf., Washington D.C., 1989, pp. 257-263

[2] F. Brglez and H. Fujiwara: A Neutral Netlist of 10 Combinational Benchmark Designs and a Special Translator in Fortran; IEEE Int. Symp. on Circuits and Systems (ISCAS), Kyoto, 1985

[3] F. Brglez, D. Bryan and K. Kozminski: Combinational Profiles of Sequential Benchmark Circuits; Proc. IEEE Int. Symp. on Circuits and Systems, 1989, pp. 1929-1934

[4] K.-T. Cheng, C.-J. Lin: Timing-Driven Test Point Insertion for Full-Scan and Partial-Scan BIST; Proc. IEEE Int. Test Conf., Washington, DC, 1995, pp. 506-514

[5] M. Chatterjee and D. K. Pradhan: A New Pattern Biasing Technique for BIST; Proc. of VLSI Test Symp., 1995, pp. 417-425

[6] S. Hellebrand, J. Rajski, S. Tarnick, S. Venkataraman and B. Courtois: Built-in Test for Circuits with Scan Based on Reseeding of Multiple-Polynomial Linear Feedback Shift Registers; IEEE Trans. on Computers, Vol. 44, No.2, February 1995, pp. 223-233

[7] V. S. Iyengar, D. Brand: Synthesis of Pseudo-Random Pattern Testable Designs; Proc. IEEE Int. Test Conf., Washington, DC, 1989, pp. 501-508

[8] B. Koenemann: LFSR-Coded Test Patterns for Scan Designs; Proc. Europ. Test Conf., Munich 1991, pp. 237-242

[9] S. Pateras, J. Rajski: Cube-Contained Random Patterns and their Application to the Complete Testing of Synthesized Multi-level Circuits; Proc. IEEE Int. Test Conf., Nashville, 1991, pp. 473-482

[10] J. Rajski: Spherical pseudo-random pattern testing, apparatus and method; USA patent application, 1996

[11] Y. Savaria, M. Yousef, B. Kaminska, M. Koudil: Automatic Test Point Insertion for Pseudo-Random Testing, Proc. Int. Symp. on Circuits and Systems, 1991, pp. 1960-1963

[12] B. H. Seiss, P. M. Trouborst, M. H. Schulz: Test Point Insertion for Scan-Based BIST; Proc. Europ. Test Conf., Munich 1991, pp. 253-262

[13] N. Tamarapalli, J. Rajski: Constructive Multi-Phase Test Point Insertion for Scan-Based BIST; Proc. IEEE Int. Test Conf., Washington, DC, 1996, pp. 649-658

[14] N. A. Touba and E. J. McCluskey: Synthesis of Mapping Logic for Generating Transformed Pseudo-Random Patterns for BIST; Proc. IEEE Int. Test Conf., Washington, DC, 1995, pp. 674-682

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

[16] M. J. Y. Williams and J. B. Angell: Enhancing Testability of Large Scale Integrated Circuits Via Test Points and Additional Logic; IEEE Trans. on Computers, vol. C-22, 1973, pp. 46-60

[17] H.-J. Wunderlich: Self Test Using Unequiprobable Random Patterns; Proc. IEEE 17th Int. Symp. on Fault-Tolerant Computing, FTCS-17, Pittsburgh 1987, pp. 258-263