

Logic BIST With Scan Chain Segmentation

Liyang Lai Janak H. Patel
Coordinate Science Laboratory
University of Illinois at Urbana-Champaign
1308 West Main Street, Urbana, IL 61801-2307
llai1@uiuc.edu
jhpatel@uiuc.edu

Thomas Rinderknecht Wu-Tung Cheng
Mentor Graphics Corp.
8005 S.W. Boeckman Road
Wilsonville, OR 97070-7777
thomas_rinderknecht@mentor.com
wu-tung_cheng@mentor.com

Abstract

This paper presents a novel BIST (Built-In Self Test) scheme with scan chain segmentation. In the scheme, a combination of pseudo random patterns and single-weight patterns have been applied to CUT (Circuit Under Test). Scan chain is partitioned into multiple segments delimited by inverters. When a single weighted pattern is applied to a segmented scan chain, successive segments receive bit patterns with complementary weights. Several segment configurations may be required to achieve full fault coverage. In this scheme the control logic is inside the scan path and built-in self test can be implemented without compromising timing performance of CUT. Experiments show that our scheme can obtain very good fault coverage. Hardware implementation is simple and straightforward.

1. Introduction

Built-in self test [1][2] has been widely adopted for its simple hardware implementation and small design effort. In BIST mode, linear feedback shift register (LFSR) or cellular automaton (CA) can be used for both pseudo-random pattern generation and test response compaction. However, there are random resistant designs for which high fault coverage is hard to achieve with reasonable test length. To overcome this random resistance, two types of techniques have been proposed. One is to modify CUT with test point insertion [3]–[6]. Test point insertion improves random testability by inserting control points and observation points into core logic. However, timing performance of CUT may be compromised by test point insertion.

The other type of techniques involve modification of the pattern generator. This includes WRPT (Weighted Random Pattern Testing) [7]–[17], reseeding [18][19] and pattern mapping [20]–[22]. In reseeding, deterministic test patterns are compressed into cubes and encoded as seeds of a PRPG

(Pseudo Random Pattern Generator); WRPT applies varied signal probability to primary inputs or scan cells to reduce test length; while in pattern mapping, ineffective pseudo-random patterns are mapped to effective deterministic patterns through mapping logic. Techniques that modify the pattern generator are able to improve fault coverage without test point insertion. However, they usually incur considerable memory or logic overhead. In reseeding, explicit on-chip memory has to be dedicated to store multiple seeds; in pattern mapping, mapping logic would be unacceptably large for designs with large number of scan cells; while in WRPT, additional logic is required for multiple weight generation and large memory for storing multiple weight distributions.

In this paper, we propose a novel BIST scheme called *scan chain segmentation*. The scheme combines pseudo random patterns with single-weight weighted patterns. Pseudo random patterns are employed first and they are followed by single-weight weighted patterns. Pseudo random testing is performed using the normal unmodified scan chain. During single-weight weighted random testing, a scan chain is partitioned into multiple segments delimited by inverters. Assume single-weight weighted patterns with weight w are loaded into the scan chain, then after segmentation the scan chain is divided into segments with alternating weights w and $1 - w$. Since control logic for *scan chain segmentation* are inverters along scan path, it poses no timing influence on core logic in both BIST mode and operational mode. Usually more than one segment configuration is required to obtain reasonable fault coverage for BIST application. Scan chain order is taken as is and there is no need to perform re-ordering to do segmentation.

Generally speaking, our BIST scheme can be categorized as a special form of weighted random pattern testing. It distinguishes itself from previous work in several ways. Traditionally, people tend to apply WRPT on the basis of

one weight per primary input or scan cell. There are many methods [9][10][14] which are dedicated to weight distribution optimization. Usually the optimization is based either on detection probability [9] or on deterministic test set [10][11][12] [13][14]. The method of deriving weight distribution from deterministic test set is more promising since it has the potential to obtain complete fault coverage with a small number of patterns. However, with ever-evolving design methodology, fully scannable design has become a common practice and large number of scan cells are very common for industrial designs. It is therefore too expensive to apply WRPT on a one-weight-per-cell basis and the hardware to store multiple weight distributions becomes unacceptably high. As a result, the idea of employing different weights to each scan channel is proposed in [15] which solves the problem of large storage requirement for multiple weight distributions. A method of approximating weight sets by Markov sources [16] was also proposed. In contrast to all previous work, we examine weight distribution of scan cells in a segment-based way or inside scan channel, which stands somewhere between analysis of one weight per input or scan cell and analysis of one weight per scan channel. Single-weight weighted patterns are fed into scan chains and it greatly simplifies the logic for weight generation. In this work, we look into the possibility of partitioning a scan chain into multiple segments such that neighboring segments have complementary weights. Unlike many previous WRPT-based BIST, reordering of the scan chain is not necessary in order to perform segmentation with acceptable inverters. It is noticed that STAR-BIST [23] also introduced inverters in its BIST implementation. The key idea of STAR-BIST is to cover hard-to-detect faults with parent vector and child vectors of certain Hamming distance to the parent. Inverters and scan chain reordering are used to encode multiple deterministic parent vectors after regularity analysis.

The effectiveness of combining pseudo random patterns with weighted patterns has been shown by many [11][12][14]. Our BIST scheme inherits this combination and is divided into multiple phases. In phase 0, pseudo random patterns are applied to CUT such that most easy-to-test faults are picked up in the phase. For each of the following phases, scan chain segmentation is performed with single-weight weighted patterns feeding into scan chain. As a result, a subset of random resistant faults is detected by each segment configuration. A simulated annealing algorithm is developed to determine a good segment configuration. Cost function of annealing schedule is evaluated by fault simulation and thus no structural analysis on netlist is required. Experimental results show that our scheme is able to achieve very good fault coverage with small hardware over-

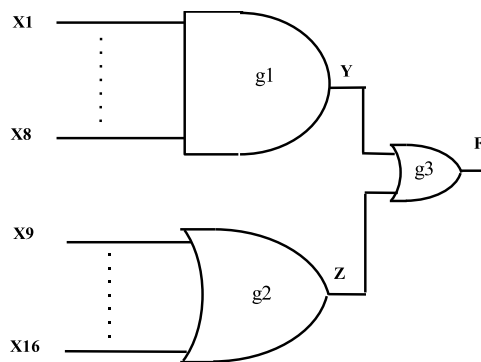


Figure 1: Circuit Diagram of Motivation Example

head.

The remaining paper is organized as follows. Section 2 gives a motivation example for our proposed scheme. It is followed by hardware implementation in section 3. BIST algorithm is described in section 4. Experimental results are presented and analyzed in section 5 and section 6 concludes the paper.

2. Motivation Example

In this section, we'd like to illustrate the basic idea of scan chain segmentation through a simple example. The circuit diagram of this example is shown in Figure 1. It is composed of an 8-input AND gate and an 8-input OR gate with their outputs ORed together. The big AND/OR gates form a typical random resistant structure. Assume boundary scan has been applied on the circuit and the scan chain is ordered by X_1 to X_{16} and then F . As shown in Table 1 the size of the collapsed fault list is 18. Test vectors for each fault are listed under left column and they, as a whole, constitute the complete test set for the circuit. It is easy to tell that faults X_9 to X_{16} s-a-0 are more random detectable than faults X_1 to X_8 s-a-1, Z s-a-1 and Y s-a-0, which are random resistant.

To target this structure, conventional WRPT would apply different weights at each input. In this case, extreme high weight (for example, weight 15/16) would be employed at inputs X_1 though X_8 ; while extreme low weight (for example, weight 1/16) at inputs X_9 to X_{16} . In contrast, our scheme is trying to detect easy-to-test faults first and then target random resistant faults with scan chain segmentation. As a result, pseudo random patterns are applied first to the circuit such that faults X_9 to X_{16} s-a-0 are detected. Scan chain segmentation is used to target the remaining faults X_1 to X_8 s-a-1, Y s-a-0 and Z s-a-1. In Table 1 τ_1 through τ_9 are test vectors for the remaining faults. They have some similar characteristics. That is, in-

Table 1: Complete Test Set and Collapsed Fault List of Fig. 1

Complete Test Set (τ_1 to τ_{17}) $X_1 \dots \dots \dots X_{16}$	Collapsed Fault List
$\tau_1 : 011111111000000000$	X_1 s-a-1, Z s-a-1
$\tau_2 : 101111111000000000$	X_2 s-a-1
$\tau_3 : 110111111000000000$	X_3 s-a-1
$\tau_4 : 111011111000000000$	X_4 s-a-1
$\tau_5 : 111101111000000000$	X_5 s-a-1
$\tau_6 : 111110111000000000$	X_6 s-a-1
$\tau_7 : 111111011000000000$	X_7 s-a-1
$\tau_8 : 111111101000000000$	X_8 s-a-1
$\tau_9 : 111111111000000000$	Y s-a-0
$\tau_{10} : 0xxxxxxx10000000$	X_9 s-a-0
$\tau_{11} : 0xxxxxxx01000000$	X_{10} s-a-0
$\tau_{12} : 0xxxxxxx00100000$	X_{11} s-a-0
$\tau_{13} : 0xxxxxxx00010000$	X_{12} s-a-0
$\tau_{14} : 0xxxxxxx00001000$	X_{13} s-a-0
$\tau_{15} : 0xxxxxxx00000100$	X_{14} s-a-0
$\tau_{16} : 0xxxxxxx00000010$	X_{15} s-a-0
$\tau_{17} : 0xxxxxxx00000001$	X_{16} s-a-0

puts X_1 to X_8 are highly biased to 1 while inputs X_9 to X_{16} are biased toward 0. In order to cover these faults by segmentation, an inverter is placed between X_8 and X_9 while a highly weighted bit stream is fed into scan chain. Assume weighted patterns with weight 15/16 are loaded into scan chain. Because of segmentation, inputs from X_1 to X_8 have weight 15/16 while inputs from X_9 to X_{16} have weight 1/16. Test vectors τ_1 through τ_8 have probability of $(15/16)^{15} \times (1/16) = 0.0237$ to be covered and τ_9 with probability $(15/16)^{16} = 0.3561$. On average, total number of $1/0.0237 = 43$ weighted patterns are required to cover τ_1 through τ_9 and thus the remaining faults.

For this simple example, one segment configuration is able to cover all remaining faults in the circuit. For big designs, usually more than one segment configuration is required to detect different types of random resistant faults. Since there is usually more than one way to detect a fault, a specific deterministic test set is too restrictive to derive a good segment configuration and therefore it is preferable to rely on more accurate information such as fault simulation.

3. Hardware Implementation

The effectiveness of combining uniform and weighted patterns in random testing has been shown by many. Our proposed BIST scheme keeps this combination and assumes test-per-scan configuration for CUT. Of course, the scheme can also be applied to designs with many internal scan chains. Figure 2 presents the hardware implementation for a 3-phase test where a weight of 1/16 is used for all weighted

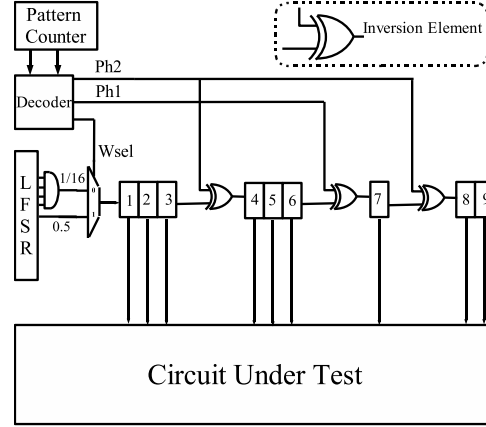


Figure 2: Hardware Implementation of Scan Chain Segmentation

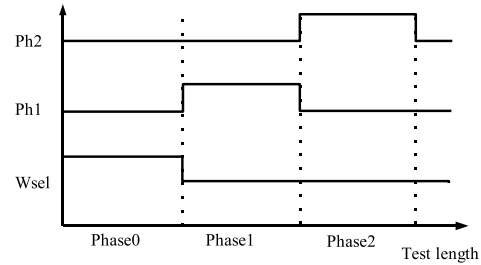


Figure 3: Waveform of Control Signals of Proposed Scheme

patterns. To simplify discussion, a scan chain with length 9 is drawn and the number of each scan cell is marked inside. MISR (Multiple Inputs Signature Register) is not shown in the figure. The decoder takes the outputs of the pattern counter as inputs and produces signals $Wsel$, phase control signals $Ph1$ and $Ph2$. As illustrated in Figure 3 the three outputs of the decoder are non-overlapping signals. $Wsel$ is used for weight selection. In phase 0, $Wsel$ is high such that pseudo random patterns are employed; After that, $Wsel$ switches to low and bit streams with weight 1/16 are applied afterwards. In this figure, a bit stream of weight 1/16 can be produced by simply taking an ANDing of four different taps from LFSR. However, care should be taken to minimize autocorrelation of weighted patterns.

Scan chain segmentation takes place for phases other than 0. To do scan chain segmentation, *inversion elements* are inserted at each position selected as segment boundary. An *inversion element* is simply an XOR gate as shown in the dashed rounded rectangle. It is noticed that inversion elements are placed along the scan path and thus they do not impose timing degradation on the core logic. Depending on the phase control signal, the XOR can selectively invert the weight of the bit stream or simply keep it. As we can see,

in phase 1 the scan chain is partitioned into two segments, namely cells 1-6 and cells 7-9 such that bits at scan cells 1-6 assume weight $1/16$ and those at scan cells 7-9 assume $15/16$. Similarly, in phase 2 three segments are obtained by scan chain segmentation and they are cells 1-3, 4-7 and 8-9 with weights $1/16$, $15/16$ and $1/16$ respectively. If an inversion element is used in several phases, corresponding phase control signals have to be ORed together to control the inversion element.

In short, the working mechanism of our proposed BIST scheme can be summarized as follows. In phase 0, pseudo random patterns are applied to detect most easy-to-test faults. A bit stream with single weight w is loaded into the scan chain thereafter. For each of the following phases, scan chain segmentation takes place such that only two weights w and $1 - w$ can be asserted at any one of the segments. Each segment configuration is used to target a subset of random resistant faults.

To minimize the size of hardware for the control logic, we could keep patterns per phase the same and let it be power of 2. For instance, if patterns per phase for above scheme are 1024, then the decoder only needs to check the first two MSBs (Most Significant Bits) of the pattern counter to generate control signals. As a result, the decoder can be synthesized with minimal hardware. On the other hand, since the phase control signals have to be routed along scan chain, they would inevitably introduce global routing. Care should be taken to minimize the number of phases as well as the number of inversion elements in each segment configuration. It should be noted that phase control signals are not timing critical with respect to scan clock and hence can be routed with slow speed routing.

One minor side effect of our scheme is that when scan-in and scan-out are overlapped, the scan-out is also affected by the inversions. One must use the transformed output response when computing the MISR signature. However, this has no influence on the effectiveness of MISR in any way.

4. BIST Algorithm

The main flow of BIST algorithm is presented in Figure 4. First redundant faults are excluded from the fault list with ATPG tools. Then pseudo random patterns for phase 0 are computed and fault simulated to detect easy-to-test faults. If target FC (fault coverage) is achieved, the algorithm terminates; otherwise, it goes into scan chain segmentation from phase 1 to N . To perform scan chain segmentation for phase p , the weighted random patterns for the phase are computed first. Given the weighted random patterns and a target fault list, the segmentation algorithm gives a segment configuration. Once the segment configuration for phase p is obtained, fault simulation is performed again using trans-

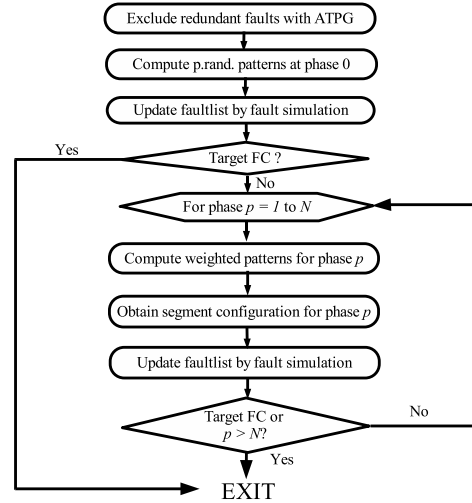


Figure 4: Main Flow of BIST Algorithm

formed weighted random patterns under the segment configuration. The iteration continues until the target fault coverage is achieved or phase limit is reached.

In our BIST algorithm, the key component is the algorithm for scan chain segmentation. The objective of scan chain segmentation is to obtain a segment configuration such that the transformed random pattern set under the configuration can detect as many target faults as possible while the number of inversion elements is minimized. A simulated annealing based algorithm is developed to compute the segment configuration for a given pattern set and target fault list. Fault simulation is performed to evaluate the benefit of each segment configuration. The remaining sections are dedicated to introducing some background knowledge on simulated annealing as well as describing its application on the segmentation problem.

4.1. Background On Simulated Annealing

Simulated annealing (SA)[24] is a probabilistic optimization technique used to solve problems without polynomial time solution. It exploits the analogy between the process that occurs when metal or crystal cools down and resorts to a minimum-energy state and the search for minimum in a general solution space. SA has been successfully applied in various VLSI design problems such as placement and routing [24].

The implementation of SA algorithm is surprisingly simple. It starts from an initial solution S_0 and initial temperature T_0 . As long as the stopping criterion is not satisfied, a neighbor solution is obtained by a random movement operation. Cost difference δC is computed based on a cost function. If the cost decreases, the neighbor solution is ac-

cepted; otherwise it is accepted with probability $e^{-\delta C/T}$. Temperature T is updated at the last iteration of each temperature. By allowing solutions with higher cost to be accepted with certain probabilities, SA is able to perform uphill movement during minimum search procedure and thus avoid being trapped in local minimal regions.

Even though the generic SA algorithm is very simple and straightforward, there is no general rule to determine initial temperature, design cost function, derive neighbor solutions and update temperature, etc. All these steps are crucial to the success of a SA algorithm and usually they, as a whole, are called annealing schedule. Annealing schedule is very problem-specific. We will show the annealing schedule for scan chain segmentation in next subsection.

4.2. Annealing Schedule for Segmentation

Before we take a detailed look at the annealing schedule, let's first go through several definitions and terminologies.

4.2.1. Solution Vector : A segment configuration is represented by a *solution vector*. A *solution vector* is a $\{0, 1\}$ cube with the same length as the scan chain. Whenever there is a $1 \rightarrow 0$ or $0 \rightarrow 1$ transition in *solution vector*, an inversion element must be inserted into the scan chain. A dummy bit 0 is assumed right before the first bit of the scan chain. For example, as shown in Figure 5(a), considering a scan chain with length 8, a *solution vector* 00000000 means the untouched, original scan chain and 00001111 denotes that one inversion element is placed between scan cell 4 and 5; while 11100011 denotes three inversion elements, the first is right before scan cell 1, the second between scan cell 3 and 4 and the last between cell 6 and 7. As a result, a *solution vector* contains the information of the position and number of inversion elements.

By scan chain segmentation, a scan chain is divided into several segments and they are represented by consecutive 1's or 0's in corresponding solution vectors. The 1's or 0's denotes the segment's *polarity*. If a segment consists of consecutive 1's, it's said to have *negative polarity*; while a segment with consecutive 0's assumes *positive polarity*. A segment with *negative polarity* is going to invert the bits during scan chain loading; A segment with *positive polarity* keeps the same bit values as the original scan chain.

4.2.2. Transformed Patterns : Given a solution vector (namely a segment configuration) and a random pattern set, we could easily compute the *transformed patterns* under the solution vector. For example, as shown in Figure 5(b), the original random pattern set is listed at the left and these vectors are loaded into the unmodified scan chain. Under solution vector 00001111, the vectors that loaded into the scan chain becomes the set in the middle of Figure 5(b); likewise, the pattern set corresponding to solution vector 11100011 is

the one at the right of Figure 5(b). The transformed patterns are obtained by simply XORing every vector in the original pattern set with the solution vector.

4.2.3. Pairwise Movement : In simulated annealing, it is essential to define a good movement operator such that neighbor solution is not too "far away" from current solution but "far enough" to allow the annealing procedure to quickly converge to an optimal solution. For our problem, *pairwise movement* is defined to disturb a small segment of the scan chain each time. Pairwise movement consists of two operators. One is to remove a pair of inversion elements; the other is to insert a pair of inversion elements. Let's show how these two operators work through a simple example in Figure 6. First, let's take a look at the operator for inserting a pair of inversion elements in Figure 6(a). A symbol of inverter is used to represent an inversion element. For a scan chain of length 8, 00001111 is the initial solution vector with an inversion element in the middle of the scan chain. A pair of inversion elements with *distance* randomly generated are placed into scan chain randomly. The *distance* of a pair of inversion elements is defined as the number of scan cells between them. And existing inversion element may lie between the newly-inserted pair as shown in option 1 or may lie outside, as in option 2. The distance of the new pair is made variable such that the annealing schedule can choose the appropriate distance as the annealing process goes on. If one of the inversion elements happens to fall onto the position of an existing inversion element, the two inversions cancel out and the number of inversion elements remains the same. The scenario when both inversion elements fall onto existing ones is not allowed and it is considered as part of the options in the procedure of removing a pair.

Figure 6(b) describes the operator for removing a pair of inversion elements. Here 11100011 is the initial solution vector. The basic idea is to invert the *polarity* of a segment in the solution vector. The segment whose polarity is inverted is randomly selected. In most cases it will result in removal of a pair of adjacent inversion elements as shown in Figure 6(b) option 1. If the segment is at the end of scan chain, it may just remove a single inversion element as shown in option 2.

The net effect of both inserting and removing a pair is to flip a portion of consecutive bits in the solution vector. But they have opposite effect on the number of inversion elements. In short, pairwise movement is performed by selectively inserting or removing a pair with certain probability. In our application, 0.5 is selected as the probability for pair insertion and removal. In pair insertion, the distance between the inserted inversion element pair is randomly chosen from 1 to 1/10 of scan chain length.

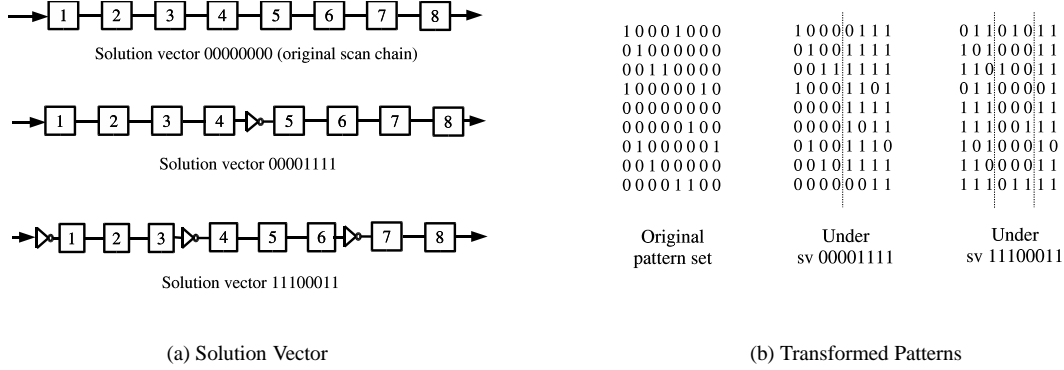


Figure 5: Example of Solution Vector and Transformed Patterns

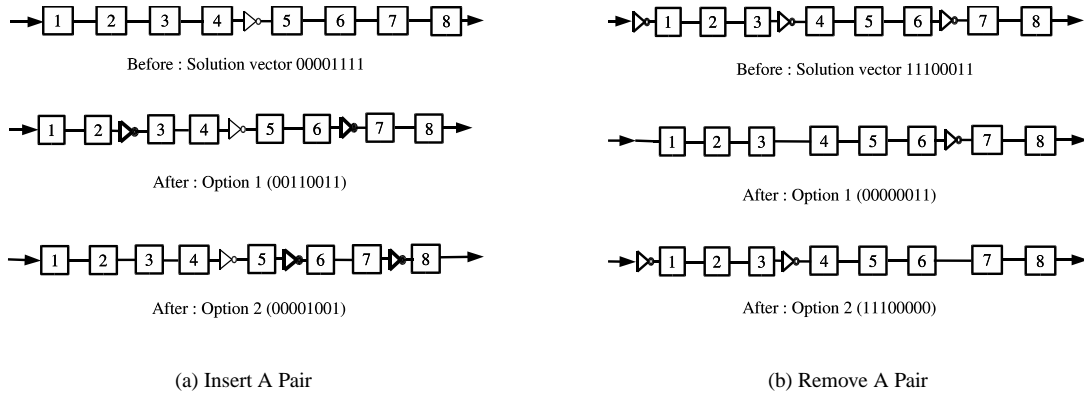


Figure 6: Example of Pairwise Movement

4.2.4. Cost Function : Cost function is another crucial element in the annealing procedure. For our problem, it is required that the cost function should not only reflect the number of undetected faults under a segment configuration but also the number of inversion elements. The more faults detected by a segment configuration, the lower the cost. For two configurations that detect the same number of faults, fewer inversion elements lead to lower cost. As a result, given a random pattern set P and target fault list F , the cost function of a solution vector sv is defined as:

$$C(sv) = \# \text{ of undetected faults by } trans(sv, P) + k \times INV_{new}(sv)$$

where $trans(sv, P)$ denotes the transformed pattern set under solution vector sv . Fault simulation is performed with respect to target fault list F . $INV_{new}(sv)$ is the number of newly introduced inversion elements by solution vector sv and k is a user-defined coefficient to weight $INV_{new}(sv)$. A k value of 1.0 is used in our application. For the inversion elements used by each solution vector, some of them already

exist in previous phases and they are called *reused inversion elements*; others are newly introduced and named by *new inversion elements*. Only *new inversion elements* contribute to cost function such that annealing procedure is biased to choose more *reused inversion elements* than *new inversion elements*. As a result, the total number of inversion elements through all phases should be minimized. In order to reuse an inversion element, the corresponding phase control signals are ORed together and connected to the phase control signal pin of the inversion element.

4.2.5. Initial Solution : In most cases, a random initial solution is sufficient to lead the annealing procedure to an optimal solution. But it may take many iterations. This is especially true when the target fault list is very small or the scan chain is very long. In order to help the simulated annealing algorithm quickly converge to an optimal solution, we provide annealing procedure with the option of using a random initial solution or an initial solution derived from deterministic test set.

For an initial solution based on test set, the test set is gen-

erated by a ATPG tool with random fill turned off. Test vectors with similar characteristics are grouped together. For example, for the test set in Table 1, test vectors τ_1 to τ_9 and τ_{10} to τ_{17} are grouped together. The group with the largest number of vectors are selected. Bit weights are computed based on the vectors in the group. Scan cells are traversed along the scan chain. Whenever there is a weight change from lower than 0.5 to higher than 0.5 or vice versus, an inversion element is inserted. An initial segment configuration is generated in this way.

4.2.6. Initial Temperature and Temperature Updating

: In our annealing schedule, initial temperature is user defined. Temperature is updated by $k_T \times T$ at the last iteration of each temperature, where k_T is temperature decreasing coefficient.

Based on all the above definitions and terminologies, the problem of scan chain segmentation can be formalized as follows.

Formulation 1 *Given a set of weighted random patterns P and target fault list F , find a solution vector sv such that the transformed pattern set $trans(sv, P)$ can detect as many target faults as possible while the number of new inversion elements $INV_{new}(sv)$ is minimal.*

The annealing algorithm for scan chain segmentation is presented in Figure 7. The parameters in the algorithm are listed as follows:

- F : target faultlist.
- P : random pattern set without segmentation.
- T : current temperature of annealing procedure.
- T_{init} : initial temperature.
- T_{low} : lower bound temperature.
- sv : current solution vector.
- sv' : neighbor solution vector.
- IPT : iterations per temperature.
- k_T : temperature decreasing coefficient.

The algorithm starts from an initial solution vector that is randomly generated or derived from deterministic test set. The initial temperature T_{init} is given as an input to the algorithm. At each temperature, IPT iterations are tried to explore the solution space. The temperature is updated with an decreasing coefficient k_T until lower bound temperature T_{low} is reached or no cost reduction in the last IPT iterations. In our application, $T_{init} = 5.0$, $T_{low} = 0.1$, $k_T = 0.97$, $IPT = 500$.

5. Experimental Results

We conducted all experiments on a Linux PC with an Athlon 900MHz processor and 256MB memory. Our proposed algorithm has been applied to ISCAS-85 [25] and

INPUT : faultlist F , random pattern set P , T_{init} , T_{low} , IPT , k_T , etc.
OUTPUT : solution vector

```

T = Tinit;
if (fault list size less than a criterion)
    sv = solution vector based on test set
else
    sv = random solution vector;
Compute cost C(sv) = # of undet. faults by sv
    + k * INVnew(sv);
while (T > Tlow) {
    if (no cost reduction in the last IPT iterations) break;
    for (i = 0; i < IPT; i++) {
        sv' = pairwise_movement(sv);
        Compute cost C(sv');
        δC = C(sv') - C(sv);
        if (δC < 0) {
            sv = sv';
        } else {
            p = e-δC/T;
            if (random(0, 1) < p) sv = sv';
        }
    }
    T = kT × T;
}
Export the solution with lowest cost.

```

Figure 7: SIMULATED ANNEALING FOR SCAN CHAIN SEGMENTATION

ISCAS-89 [26] benchmark circuits. Parameters of annealing schedule were given at the end of Section 4.2. Full scan has been assumed. All fault coverage results are based on single stuck-at fault model and fault coverage is defined as the percentage of irredundant faults. To simplify hardware implementation, the number of patterns used in each phase are forced to be equal. That is, the same number of patterns are used in pseudo-random testing and weighted test for each segmentation. The used numbers of patterns per phase are 1024, 2048, 4096 or 8192 etc. A type I 32-bit LFSR is used for pseudo random pattern generator and the taps come from [2]. The single weight used for scan chain segmentation is listed under column *weight*. Experiments are performed on the effect of different weights on scan chain segmentation. It was discovered that in most cases weight 1/16 seems to generate best results and is used in all the following experiments.

Weighted patterns are generated in a way that autocorrelation between bits is minimized. A phase shifter [27] is inserted between LFSR and CUT and it generates multiple bit streams with interscan channel distance over 1,000,000. Weight 1/16 is simply produced by ANDing four channel

outputs of the phase shifter. ATOM[28] is used to identify redundant faults and a parallel pattern single fault simulator [29] is implemented for fault simulation. It should be noticed that we do not pose any restriction on scan chain ordering and the scan chain is taken as is.

Experimental results on ISCAS 85 benchmark circuits are presented in Table 2. The upper bound of total patterns is set to be $32k$. If a circuit is fully tested in one phase, then only pseudo random patterns are applied and no segmentation is required. The meaning of each column in Table 2 is listed below.

- *Circuit* : name of circuit.
- *Irred. Faults* : number of irredundant faults in collapsed fault list.
- *Red. Faults* : number of redundant faults.
- *# of PI's* : number of primary inputs.
- *# of PO's* : number of primary outputs.
- *weight* : weight applied in segmentation.
- *Pat. Per Phase* : number of patterns per phase.
- *Phases* : total number of phases.
- *Inv. Elements* : total number of inversion elements inserted for scan chain segmentation.
- *Total Patterns* : total number of patterns applied.
- *Run Time* : run time for segmentation procedure.

As shown in the table, all ISCAS 85 circuits can be completely tested within four phases. It is essential that the smallest number of phases and inversion elements are employed since this way hardware overhead is minimal (especially the global routing of phase control signals). In addition, we present the experimental results of several well-known random resistant circuits in Table 3. For all these circuits, weight $1/16$ has been applied in scan chain segmentation. The meaning of each column is as before except the following new columns.

- *Num. of SCs* : number of scan cells which is computed as the sum of primary inputs and flip-flops.
- *Cov. (Det.faults)* : fault coverage and number of detected faults obtained by proposed BIST scheme.
- *Cov. by 32k rand. (Det.faults)* : fault coverage and number of detected faults by 32k pseudo random patterns.
- *Equivalent Gates* : number of additional gates required to implement segmentation scheme in addition to common BIST hardware.

As shown in Table 3 our proposed BIST scheme is able to improve random pattern fault coverage by 2% to 12% with very few inversion elements and phases. The number of *equivalent gates* is computed as the number of equivalent 2-input NAND gates. It includes the logic for generating phase control signals and inversion elements. A 2-input XOR gate is calculated as 2.5 equivalent gates and

an inverter is 0.5. It is noticed that for large circuit such as *s38417* SA algorithm takes very long to run and therefore needs improvement. One way to speedup the SA algorithm is to use sampled fault simulation in the cost evaluation phase.

For comparison, results obtained by [9] and [13] are given in Table 4. [9] employs multiple weight distributions to obtain complete fault coverage. The number of weight distributions is listed under column *weight distr.s*. It should be noticed that for each weight distribution every primary input has been assigned a distinct weight and the actual weights are not given in [9]. In fact many weights computed by [9] may not be feasible or too expensive to implement. In addition the number of patterns for each weight distribution is the smallest pattern count unlike a fixed power of 2 as in most practical implementations. Nevertheless, our method requires far fewer patterns for most random resistant circuits.

In [13] a 3-weight random testing scheme is developed. An expanded test in [13] is a test session when a set of primary inputs are constrained to 0 or 1 by 3-gate modules. In the first expanded test no input is constrained. The number of patterns in each expanded test is the same. As a result, the number of expanded tests under column *exp. tests* could be compared with the number of phases in our proposed method. The number of 3-gate modules under column *3-gate Modules* is compared to that of inversion elements employed in our method. It should be noted that the 3-gate modules are on the functional path and timing of CUT is affected. For each circuit the result with the fewest expanded tests is selected and listed under column [13]. In most cases our method is able to achieve the same complete coverage with far fewer phases and lower gate count. It is especially true for the two well-known random resistant circuits *c2670* and *c7552*.

6. Conclusion

A novel BIST scheme with scan chain segmentation is described in this paper. Scan chain segmentation presents a unique method of employing weighted random patterns. Based on the observation that deterministic test patterns of random resistant faults tend to cluster with extremely high or low weight in different regions of scan chain, a scan chain is partitioned into multiple segments delimited by inversion elements while the scan chain input is fed by a single-weight bit stream. Our BIST scheme works by combining pseudo random patterns with single-weight weighted patterns. Pseudo random patterns are used to cover easy-to-test faults. Several segment configurations are usually required to achieve reasonable fault coverage for BIST application. Complete fault coverage is possible with our BIST scheme. However, many more inversion elements and phases may be

Table 2: Experimental Results of Complete Coverage for ISCAS 85 Benchmarks.

Circuit	Irred. Faults	Red. Faults	# of PI's	# of PO's	weight	Pat. Per Phase	Phases	Inv. Elements	Total Patterns	Run Time (s)
c432	520	4	36	7	–	1024	1	0	1024	–
c499	750	8	41	32	–	1024	1	0	1024	–
c880	942	0	60	26	1/16	1024	2	2	2048	1.1
						2048	2	2	4096	5.9
c1355	1566	8	41	32	1/16	1024	2	1	2048	4.9
						2048	2	0	4096	9.7
c1908	1870	9	33	25	1/16	1024	6	16	6144	19.8
						2048	4	7	8192	20.5
						4096	2	2	8192	9.1
						8192	2	2	16384	18.3
c2670	2630	117	233	140	1/16	1024	4	4	4096	98.5
						2048	3	3	6144	91.6
						4096	3	8	12288	301.9
						8192	3	3	24576	381.9
c3540	3291	137	50	22	1/16	1024	9	24	9216	189.5
						2048	7	19	14336	236.3
						4096	6	18	24576	303.6
						8192	4	14	32768	664.2
c5315	5291	59	178	123	–	4096	1	0	4096	–
c6288	7710	34	32	32	–	1024	1	0	1024	–
c7552	7419	131	207	108	1/16	1024	7	28	7168	423.9
						2048	8	30	16384	975.1
						4096	5	15	20480	846.5
						8192	4	13	32768	2446.8

Table 3: Experimental Results for Random Resistant Circuits.

Circuit	Irred. Faults	Num. of SCs	phases	Total Patterns	Inv. Elements	Cov. (Det.faults)	Cov. by 32k rand. (Det.faults)	Equivalent Gates	Run time (s)
c2670	2630	233	3	6k	3	100.0% (2630)	88.52% (2328)	11.5	91.6
c7552	7419	207	4	32k	13	100.0% (7419)	96.98% (7195)	38.5	2446.8
s9234	6475	247	4	32k	20	98.07% (6350)	93.00% (6022)	60.5	10488.5
s15850	11336	611	4	32k	16	99.25% (11251)	95.57% (10834)	46	11251.5
s38417	31015	1664	4	32k	105	98.72% (30619)	95.51% (29624)	283.5	18561.9

required to achieve complete coverage in random resistant circuits. A simulated annealing algorithm is developed to perform scan chain segmentation. Fault simulation is used to do cost function evaluation.

Experiments on ISCAS benchmark circuits show our method is able to achieve high fault coverage with small number of phases and low hardware overhead. The concept of segmentation could be easily expanded to designs with multiple scan chains.

References

- [1] P. H. Bardell, W. H. McKeeney, and J. Savir, "Built-in Test for VLSI: Pseudorandom Techniques", *John Wiley and Sons*, New York, 1987.
- [2] M. Abramovici, M. A. Breuer and A. D. Friedman, "Digital Systems Testing and Testable Design", *IEEE Press*, New York, 1990.
- [3] A. J. Briers and K.A.E. Totton, "Random Pattern Testability by Fast Fault Simulation", *Proc. of Int'l Test Conf.*, pp.274-281, 1986.
- [4] Y. Savaria, M. Youssef, B. Kaminska and M. Koudil, "Automatic Test Point Insertion for Pseudo-Random Testing", *Proc. of Int'l Symp. on Circuit and Systems*, pp.1960-1963, 1991.
- [5] E. M. Rudnick, V. Chickermane, and J. H. Patel, "An Observability Enhancement Approach for Improved Testability and At-Speed Test", *IEEE Trans. on CAD*, Vol.13, No.8, pp.1051-1056, 1994.
- [6] N. Tamarapalli, J. Rajski, "Constructive Multi-Phase Test Point Insertion for Scan-Based BIST", *Proc. of Int'l Test Conf.*, pp649-658, 1996.

Table 4: Comparison of Results.

Circuit	[9]		[13]			Proposed Method		
	weight distr.s	Patterns	exp. tests (phases)	3-gate Modules	Patterns	phases	Inv. Elements	Patterns
c432	2	1,100	1	0	1,024	1	0	1,024
c499	1	1,200	1	0	1,024	1	0	1,024
c880	3	720	2	13	8,192	2	2	2,048
c1355	1	2,100,000	2	1	4,096	2	0	4,096
c1908	6	21,000	4	0	4,096	2	2	8,192
c2670	5	810,000	15	233	30,675	3	3	6,144
c3540	5	110,000	3	3	12,288	4	14	32,768
c5315	3	11,000	4	0	4,096	1	0	4,096
c6288	1	240	1	0	1,024	1	0	1,024
c7552	6	280,000	36	207	73,728	4	13	32,768

- [7] H. D. Schnurmann, E. Lindbloom, R. G. Carpenter, "The Weighted Random Test Pattern Generation", *IEEE Trans. Comp.*, Vol. C-24, pp. 675-700, July 1974.
- [8] H. J. Wunderlich, "Self Test Using Unequiplorable Random Patterns", *Proc. of Int. Symp. on Fault-Tolerant Computing*, pp.258-263, 1987.
- [9] H. J. Wunderlich, "Multiple Distributions for Biased Random Test Patterns", *Proc. of Int'l Test Conf.*, pp.236-244, 1988.
- [10] J. Waicukauski et. al., "WRP : Method for Generating Weighted Random Patterns", *IBM Journal of Research and Development*, Mar 1989.
- [11] F. Brglez, C. Gloster, and G. Kedem, "Hardware-based weighted random pattern generation for boundary scan, ", *Proc. of Int'l Test Conf.*, pp.264-273, 1989.
- [12] F. Muradali, V. K. Agarwal, and B. NadeauDostie, "A New Procedure for Weighted Random Built-In Self Test", *Proc. of Int'l Test Conf.*, pp.660-669, 1990.
- [13] I. Pomeranz and S. M. Reddy, "3-Weight Pseudo-Random Test Generation Based on Deterministic Test Set for Combinational and Sequential Circuits," *IEEE Trans. on CAD of Integrated Circuits and Systems*, Vol.12, No. 7, pp.1050-1058, 1993.
- [14] J. Hartmann, "On Numerical Weight Optimization for Random Testing", *Proc. of the joint EDAC-EUROASIC Conf.*, pp. 223-230, 1993.
- [15] M. P. Kusko, B. J. Robbins, T. J. Koprowski, W. V. Huott, "99% AC test coverage using only LBIST on the 1 GHz IBM S390 zSeries 900 Microprocessor", *Proc. of Int'l Test Conf.*, pp. 586-592, 2001.
- [16] N. Z. Basturkmen, S. M. Reddy and I. Pomeranz, "Pseudo Random Patterns Using Markov Sources for Scan BIST", *Proc. of Int'l Test Conf.*, pp. 1103-1021, 2001.
- [17] S. Pateras and J. Rajska, "Cube-Contained Random Patterns and their Application to the Complete Testing of Synthesized Multi-Level Circuits", *Proc. of Int'l Test Conf.*, pp. 473-482, 1991.
- [18] S. Hellebrand, J. Rajska, S. Tarnick, S. Venkataraman, and B. Courtis, "Built-In Test for Circuits with Scan Based on Re-seeding of Multiple-Polynomial Linear Feedback Shift Registers", *IEEE Trans. on Comp.*, Vol. C-44, pp. 223-233, Feb. 1995.
- [19] B. Koenemann, "LFSR-Coded Test Patterns for Scan Designs", *Proc. of Euro. Test Conf.*, pp.237-242, 1991.
- [20] M. Chatterjee, and D. K. Pradhan, "A Novel Pattern Generator for Near Perfect Fault-Coverage", *Proc. of Int'l Test Conf.*, pp.417-425, 1995.
- [21] N. A. Touba and E. J. McCluskey, "Transformed Pseudo-Random Patterns for BIST", *Proc. of VLSI Test Symp.*, pp. 410-416, 1995.
- [22] N. A. Touba and E. J. McCluskey, "Altering A Pseudo-Random Bit Sequence for Scan-Based BIST", *Proc. of Int'l Test Conf.*, pp. 167-175, 1996.
- [23] K.-H. Tsai, J. Rajska and M. Marek-Sadowska, "Star Test: The Theory and Its Applications", *IEEE Trans. on Comp. Aided Design of Integrated Circuits and Systems*, Vol.19, No.9, pp. 1052-1064, 2000.
- [24] D. F. Wong, H. W. Leong and C. L. Liu, "Simulated Annealing for VLSI Design", *Kluwer Academic Publishers*, USA, 1988.
- [25] F. Brglez and H. Fujiwara, "A Neutral Netlist of 10 Combinational Benchmark Designs and A Special Translator in Fortran", *Proc. Intl. Symp on Circuits and Systems*, June 1985.
- [26] F. Brglez, D. Bryan, K. Kozminski: "Combinational Profiles of Sequential Benchmark Circuits", *Proc. of Int. Symp. on Circuits and Systems*, pp.1929-1934, 1989.
- [27] J. Rajska, N. Tamarapalli and J. Tyszer, "Automated Synthesis of Large Phase Shifters for Built-In Self-Test", *Proc. of Int'l Test Conf.*, pp. 1047-1056, 1998.
- [28] I. Hamzaoglu and J. H. Patel, "New Techniques for Deterministic Test pattern Generation", *Proc. of VLSI Test Symp.*, pp. 446-452, 1998.
- [29] J. A. Waicukauski, etc, "Fault Simulation for Structured VLSI", *VLSI Systems Design*, Vol. 6, No. 12, pp. 20-32, Dec. 1985.