# Layout-Aware Illinois Scan Design for High Fault Coverage

S. Banerjee[1], J. Mathew[1], D. K. Pradhan[1], and S. P. Mohanty[2]
[1]Department of Computer Science
University of Bristol, UK.
[2]University of North Texas, Denton,
TX 76203.

***Abstract*— The Illinois Scan Architecture (*ILS*) consists of several scan path segments and is useful in reducing test application time and test data volume required to test today's high density *VLSI* circuits. However, to achieve high fault coverage with *ILS* architecture one requires judicious grouping and ordering of scan flip-flops for selecting these segments. This may also increase the wiring complexity and cost of the scan chain, as the physical locations of the flip-flops on silicon are determined at an early design stage before scan insertion. In this paper, we propose a scheme of layout-aware as well as coverage-driven *ILS* design. The partitioning of the flip-flops into *ILS* segments is determined by their geometric locations, whereas the set of the flip-flops to be placed in parallel is determined by the minimum incompatibility relations among the corresponding bits of a test set, to enhance fault coverage in broadcast mode. This consequently, reduces the number of test patterns required in serial mode. The proposed methodology reduces test application time significantly, and at the same time, achieves high fault coverage. Experimental results on various benchmark circuits demonstrate the efficacy and versatility of the proposed method.**

## I. Introduction

The controllability and observability of a digital circuit can be increased by various well-known design-for-testability (DfT) techniques. Among them, the serial full scan style is widely used, which transforms a sequential circuit to its combinational parts in test mode. Although this method reduces the cost of test generation and provides high fault coverage, the test application time and power dissipation in test mode become significantly high because of the inherent serial nature of the scan path. It also increases test data volume. As most of the systems now consist of thousands of flip-flops, memory requirement for storing test data in an automatic test equipment (ATE), as well the test time becomes unacceptably high. An ATE with a large storage device slows down its memory access time and the test clock frequency.

The Illinois Scan Architecture (*ILS*) is proposed recently to reduce the test application time and test data volume for the embedded cores [3], [4], [5], [6], [7], [8], [9], [10]. It is applicable to both standalone cores or cores embedded in SOC, and requires a low-cost ATE. The main problem of *ILS* design is to select segments properly so that the objectives of reducing test time/data and that of achieving high fault coverage are satisfied concurrently. Further, none of the earlier works on the *ILS* structure considered the impact of physical design while determining the segments. In a typical design flow, the positions of the flip-flops on silicon are determined by the functional interconnections and other routing constraints, and the scan path is inserted later. While post-insertion of scan paths does not pose a problem for serial scan, it does influence the grouping of flip-flops as segments of *ILS*. Random grouping may increase wiring cost and congestion and degrade the performance. Some layout-aware design of single/multiple scan chains have been considered earlier [11], [12], [13], [14]. A recent work on layout-aware scan cell ordering has been proposed in [15]. to increase the fault coverage for path-delay faults.

In this paper, we propose layout-aware design of *ILS* architecture, which takes care of the physical locations of the flip-flops, and provides a compromise among fault coverage, test application time/test data volume, and wiring cost. The propose algorithm consists of three steps. First, scan cells are grouped based on their geometric proximity to form the different segments of the *ILS*; this reduces wiring cost. Second, the flip-flops in different segments are aligned based on some incompatibility information derived from a test set so that the fault coverage in broadcast mode is enhanced. As a result, the number of additional test patterns required in the serial mode reduces. Finally, some reordering of scan cells concurrently on all the segments, is performed to reduce scan-path length further. Experimental results on various benchmark circuits show that the proposed *ILS* structure provides significant reduction in test application time with high fault coverage while reducing wire length in the scan chains, compared to other structures designed by earlier methods.

## II. The Illinois Scan Architecture

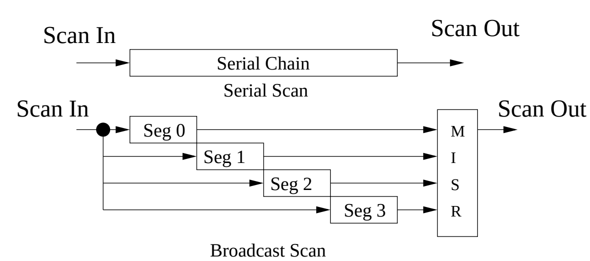


Fig. 1. The basic two modes of the *ILS* architecture

The *ILS* architecture is shown in *Fig. 1*, where the top part shows the original serial scan chain. This mode is known as serial mode. The bottom part of the figure shows the scan chain broken into several segments. These segments are placed in parallel and a single SIP is used to feed the test patterns to these segments. This mode is known as broadcast mode. The outputs of the segments are collected by an MISR. The broadcast mode can be reconfigured into a serial mode by using multiplexers. Since the broadcast mode imposes constraints on test pattern bits many faults may become untestable. To detect these faults, an $ATPG$ is used to generate additional test vectors that are applied under serial mode.

## III. Layout-aware *ILS* structure genaration algorithm

The inputs to the algorithm are,

1) A given set of $FFs$ and their corresponding coordinates in the layout area.
2) A given set of deterministic test vectors

The three basic steps of the algorithm are now described as follows.

### A. Partitioning of flip-flops into ILS segments

In this step, we partition the flip-flops into different groups based on their geometric neighborhood. In other words, the flip-flops nearest to each other are grouped, so that scan path length is minimized. Each group will form a segment of *ILS*, and the number of groups will be equal to the number of required segments in the *ILS* structure, which may be a user-defined parameter. For simplicity, we first assume that the number of flip-flops of the circuit is divisible by the number of segments. In *ILS* structure, an increase in the number of segments will reduce the scan path length and also the fault coverage. Thus, the trade-off may be chosen by the user depending on the requirement and desirability.

The partitioning problem may be expressed formally in graph-theoretic term. A complete graph $G=(V,E)$ is an undirected graph in which every pair of vertices is adjacent, where $V=\{v_1,v_2,..,v_n\}$ be a set of vertices and $E=\{e_1,e_2,..,e_m\}$ be a set of edges. Each vertex represents a scan flip-flop. The weight of the each edge is equal to the distance between the two corresponding flip-flops connected by that edge. A complete graph corresponding to five scan flip-flops is shown in *Figure 2*.

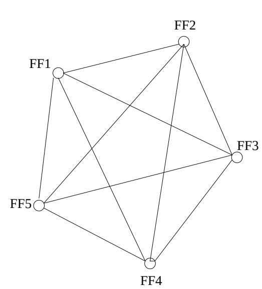


Fig. 2. A complete graph for 5 scan flip-flops

The problem is to partition $V$ into $V_1,V_2,..,V_k$, where $V_i \bigcap V_j = \emptyset, i \neq j$ and $\bigcup_{i=1}^{k} V_i = V$

In this case, one additional constraint is added to make the each segment balanced, which is

$count(V_i) - count(V_j) = 0,\ i \neq j$

where $count(V_i)$ is equal to the number of vertices in the set $V_i$. The above partitioning problem can be solved optimally by using Integer Linear Programming (ILP). Let there be $n_f$ scan flip-flops, which are to be segmented into $n_p$ groups. So the graph $G=(V_{n_f},E)$ will have $n_f$ vertices. The coordinates of a vertex (scan flip-flop) correspond to its center. The distance between the $i$ and $j$ vertices $(w_{ij})$ is equal to the Euclidean distance between them, which is

$$w_{ij} = \sqrt{(x_i-x_j)^2+(y_i-y_j)^2}$$

where $(x_i,y_i)$ and $(x_j,y_j)$ are the coordinates of the $i^{th}$ and $j^{th}$ vertices respectively. Let $x_{ik}$ be a $0-1$ variable defined as follows:

$$x_{ik} = \begin{cases} 1 & \text{if vertex i is in the segment k;} \\ 0 & \text{otherwise.} \end{cases}$$

Similarly $y_{ijk}$ be a $0-1$ variable defined as follows:

$$y_{ijk} = \begin{cases} 1 & \text{if both i,j vertices in the segment k;} \\ 0 & \text{otherwise.} \end{cases}$$

The weight of a segment $k$ can be expressed as $\sum_{i=1}^{n_f}\sum_{j=1}^{n_f} w_{ij}y_{ijk}, i \neq j$. An ILP can be formulated for minimizing the weight:

Minimize $C = \sum_{i=1}^{n_f}\sum_{j=1}^{n_f} w_{ij}y_{ijk}, 1 \leq k \leq n_p, i \neq j$, subject to
1. $\sum_{i=1}^{n_f} x_{ik} = n_f/n_p,\ 1 \leq k \leq n_p$
2. $\sum_{k=1}^{n_p} x_{ik} = 1,\ 1 \leq i \leq n_f$
3. $x_{ik} = 0$ or $1$, $1 \leq i \leq n_f$, $1 \leq k \leq n_p$
4. $y_{ijk} = 0$ or $1$, $1 \leq i \leq n_f$, $1 \leq j \leq n_f$, $i \neq j$, $1 \leq k \leq n_p$

The condition 1 is added to ensure the balance condition with respect to each other. The above cost function can easily be linearized and the resulting ILP model is shown in *Figure 3*.

---

Minimize $C$ subject to
1. $C \geq \sum_{i=1}^{n_f}\sum_{j=1}^{n_f} w_{ij}y_{ijk}, 1 \leq k \leq n_p, i \neq j$
2. $\sum_{i=1}^{n_f} x_{ik} = n_f/n_p,\ 1 \leq k \leq n_p$
3. $\sum_{k=1}^{n_p} x_{ik} = 1,\ 1 \leq i \leq n_f$
4. $x_{ik} = 0$ or $1$, $1 \leq i \leq n_f$, $1 \leq k \leq n_p$
5. $y_{ijk} = 0$ or $1$, $1 \leq i \leq n_f$, $1 \leq j \leq n_f$, $i \neq j$, $1 \leq k \leq n_p$

---

Fig. 3. ILP model for Step 1

### B. Determining the flip-flops to be placed in parallel across the segments

We start with a given test set of the original circuit, and analyze the compatibility relationships among the flip-flops. The proposed algorithm will select one flip-flops from each segment at a time, such that they have minimal incompatibility relationship among themselves with respect to the given test set. All such flip-flops will be placed in parallel (at the same depth) in the *ILS* structure. This will give rise to high fault coverage in broadcast mode.

For example, consider the test set shown in *Fig. 4.a*, which consists of three test vectors. The circuit has six flip-flops; hence each test vector is six-bit long. According to the above description, flip-flop 1, is denoted as $FF1$, flip-flop 2 as $FF2$, and so on. We assume that these six flip-flops are partitioned

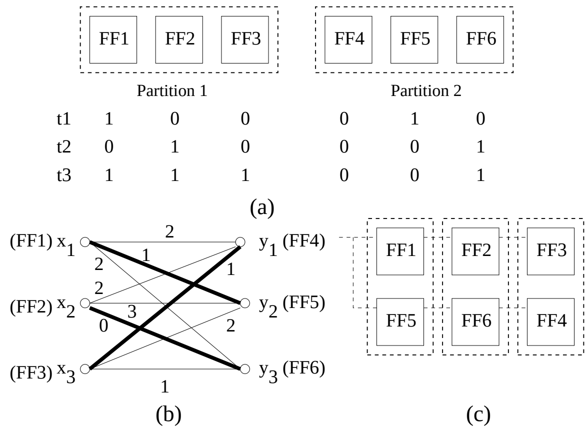


Fig. 4. Example of generation of *ILS* structure

into two segments based on their coordinates. Each segment consists of three flip-flops: $FF1$, $FF2$, and $FF3$ are in segment 1, and $FF4$, $FF5$, and $FF6$ are in segment 2. The ILP structure will have two segments and each segment contains three flip-flops. The next task is to find out three groups of two flip-flops each, by selecting one flip-flop from each segment. The flip-flops belonging to a group will be placed at the same depth in the *ILS* structure. This selection problem based on minimal incompatibility can be solved by finding a perfect matching of a bipartite graph with minimum weight, which is described next.

The incompatibility distance (denoted as $d$) of two flip-flops belonging to two segments is defined as the number of conflicting bits in two corresponding column vectors of the test matrix. For the example in *Fig. 4a*, we have the following distance values:

$d(FF1,FF4) = 2$, $d(FF1,FF5) = 1$, $d(FF1,FF6) = 2$,
$d(FF2,FF4) = 2$, $d(FF2,FF5) = 3$, $d(FF2,FF6) = 0$,
$d(FF3,FF4) = 1$, $d(FF3,FF5) = 2$, $d(FF3,FF6) = 1$.

A graph $G(X,Y)$, called weighted incompatibility bipartite graph (WIBG) is then constructed, whose left set of vertices represents the flip-flops in one segment, and right set of vertices represents the flip-flops in the other segment. As shown in *Fig. 4b*, $X = \{x_1(FF1), x_2(FF2), x_3(FF3)\}$ denotes the set of flip-flops in the segment 1 and $Y = \{y_1(FF4), y_2(FF5), y_3(FF6)\}$ denotes the set of flip-flops in the segment 2. The weight on an edge represents the incompatibility distance between the two vertices connected by the edge. A zero weight denotes a compatible pair. The resultant WIBG of *Fig. 4a* is shown in *Fig. 4b*.

In a WIBG $G(X,Y)$, a non-negative weight $w_{ij}$ is assigned to each edge $x_iy_i$ of $G$. We seek a perfect matching $M$ of graph $G$ that minimizes the total weight $w(M)$. A matching in a graph $G$ is a set of non-loop edge with no shared end points [16]. The vertices incident to the edges of a matching $M$ are saturated by $M$; the others are unsaturated. A perfect matching in a graph is a matching that saturates every vertex. In *Fig. 4b*, the minimum matching consists of three edges i.e. $M = \{x_1y_2, x_2y_3, x_3y_1\}$, and the weight of $M$ is 2. Two vertices or flip-flops connected by an edge of $M$ can be grouped. For example, in *Fig. 4b* three groups of flip-flops have been formed which are $g_1$ ($\{FF1, FF5\}$) connected by the edge $x_1y_2$, $g_2$ ($\{FF2, FF6\}$) connected by the edge $x_2y_3$, and $g_3$ ($\{FF3, FF4\}$) connected by the edge $x_3y_1$. In this way, an *ILS* structure is built where the flip-flops lying at the same depth in the scan structure will have a minimal incompatibility relation. The resultant *ILS* structure without interconnection among the flip-flops is shown in *Fig. 4c*.

To find a perfect matching with minimum weight we replace each weight $w_{ij}$ with $Z - w_{ij}$ for some large number $Z$ (say 100). Then, we can use an algorithm for computing maximum weight perfect matching, which is solvable in polynomial time.

When the number of segments is more than two, we form the groups by considering two segments at a time. After forming the groups of flip-flops between these two segments, another segment will be considered. The process will continue until all the flip-flops in each segment are grouped. For $n$ segments, we have to repeat the process for $(n-1)$ times. An outline of the algorithm (Algorithm_1) is shown in *Fig. 5*.

### *C. ordering of the scan cells*

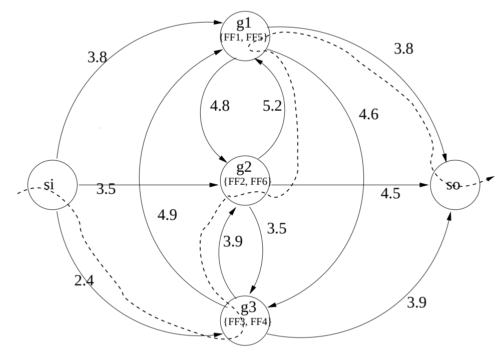


Fig. 6. The WDG for ordering scan cells

Once $FFs$ in each branch and groups of $FFs$ lying at the same depth have been identified, scan cell ordering within a branch or partition is done by using a weighted distance graph ($WDG$). The $WDG$ is a directed graph where each group (obtained from phase-2) are vertices and two directed edges between each two groups i.e., $e_{ij}$ from $g_i$ to $g_j$ and $e_{ji}$ from $g_j$ to $g_i$. In addition to these vertices, $WDG$ also contains one vertex corresponds to scan-in pin (*si*) and one corresponds to scan-out pin (*so*). As outputs of the *ILS* structure are feed either to *MISR* or Compactor so, in the present case we have considered only one scan-out pin. So, the $WDG$ for the present example consists of 5 vertices which are $g_1$, $g_2$, $g_3$, *si*, and *so*. In $WDG$, in addition to the directed edges among the $g_1$, $g_2$ and $g_3$, there will be six more edges. Three edges are due to *si* ($si \rightarrow g_1$, $si \rightarrow g_2$, $si \rightarrow g_3$) and three edges due to *so* ($g_1 \rightarrow so, g_2 \rightarrow so, g_3 \rightarrow so$). The weight of the each edge depends on the average routing distance between the nodes. Let $FF_i$ and $FF_j$ be two scan cells. The distance $d(FF_i, FF_j)$ is the distance between the output port of $FF_i$ and input port of $FF_j$. Similarly, the distance $d(si, FF_i)$ ($d(FF_i, so)$) is the distance between the *si* (output port of $FF_i$) and input port of $FF_i$ (*so*). Now, a group of scan cells should be put in the first level of *ILS* structure if their locations are closer to scan-in pin (*si*).

**Algorithm_1: Generate the *ILS* structure**

Input to the Algorithm_1: (a). The total number of segments $(n_p)$. Let they be $p_1, p_2, \ldots, p_{n_p}$.
(b). The number of flip-flops in each segment.
(c). Flip-flops lying in each segment.
(d). A set of test patterns obtained by an *ATPG* tool with desired level of fault coverage (pilot set).

1. Set two integers i=1 and j=2.
2. Consider the segments $p_i$ and $p_j$.
3. Generate the WIBG $G(X,Y)$ for $p_i$ and $p_j$, based on the information given in $(a)$, $(b)$, $(c)$, and $(d)$.
4. Replace the weight $w_{ij}$ of each edge with $Z - w_{ij}$, where $Z$ is a large number.
5. Find maximum weight perfect matching $M$ of $G(X,Y)$.
6. Group the two flip-flops connected by each edge of the $M$. If $|M| = n_e$, then there will be $n_e$ groups of flip-flops and each group will have two flip-flops.
7. Set $i = i+1$, and $j = j+1$.
8. if $j$ is not equal to $p_{n_p}$, then go to step 1.
9. All the flip-flops in each segment are grouped, i.e., the flip-flops lying at the same depth in the *ILS* structure has been determined.

Fig. 5. Algorithm_1 to generate the *ILS* structure

Similarly, at the last level group of scan cells should be closer to the scan-out pin (*so*). Based on above, weight $w_{g_i \to g_j}$ is the average routing length between scan cells in $g_i$ and $g_j$. Similarly, weight $w_{si \to g_i}$ $(w_{g_i \to so})$ is the average distance from *si* (scan cells in $g_i$) to scan cells in $g_i$ (*so*). The final *WDG* for the *Fig. 4* is shown in *Fig. 6*. After creating *WDG*, the order of the scan cells can be determined by finding a shortest path from *si* to *so*. As here all the weights are positive, we can use Dijkstra's algorithm whose runtime is linear to find out the shortest path. In the present case best path should be $si \to g_2 \to g_3 \to g_1 \to so$ as shown by dotted line in *Fig. 6*. The final *ILS* structure of *Fig. 4* after scan cell reordering is shown in *Fig. 7*.

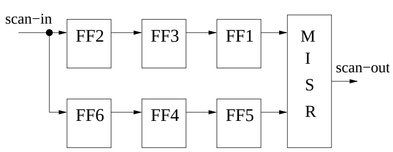


Fig. 7. The final *ILS* structure for *Fig. 4*

## IV. Serial Mode

Once we obtain an *ILS* structure, we rerun the same *ATPG* tool to generate a new set of test vectors with the logical constraints imposed on the secondary inputs of the circuit by segmentation of scan cells. The rationale behind this is as follows: since the same *ATPG* tool is being run on the same circuit-under-test (*CUT*), with some input constraints determined by minimal incompatibility, fault coverage with graceful degradation can be achieved also in the second run. However, in the presence of the constraints, some detectable faults in the original circuit may become untestable or hard-to-test in the broadcast mode. The undetectable faults can now be detected by reconfiguring dynamically the *ILS* structure into the serial mode, and by applying a few additional test patterns. Thus, the first mode is the broadcast mode (BC mode) and second one is the serial scan mode (SS mode). The idea is to apply the major part of the test vectors in BC mode and the remaining part in SS mode. *Fig. 8* illustrates the technique. The switching from BC mode to SS mode is done by a controller, which consists of some logic and a counter that counts the number of test patterns to be applied in the BC mode. The Algorithm_2 described below, is used

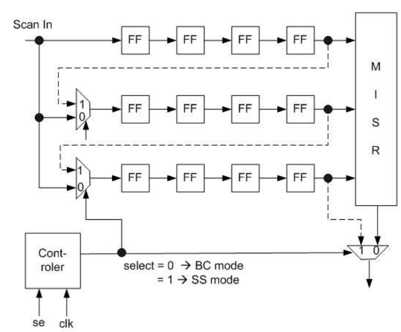


Fig. 8. The *ILS* architecture with dynamic reconfiguration

to determine the complete test patterns $T$ for a circuit. The set $T$ consists of two parts, $T_B$ and $T_S$, i.e., the set of test patterns for the broadcast mode and for the serial scan mode respectively. Algorithm_2 first determines the complete fault list for the circuit. Let the complete fault list be denoted by $f_c$. The constraints of the *ILS* structure are then imposed on the secondary inputs to the CUT. Next, the same *ATPG* tool is run to generate a set of patterns ($T_B$) for the circuit. Incremental fault simulation is performed while generating the test patterns in $T_B$ with the target fault list $f_c$. Let $f_d$ represent the set of currently detectable faults. For each pattern, it identifies the detected faults and $f_d$ is updated. The process continues until all the test vectors in $T_B$ are simulated, and the set of undetectable faults $f_u$ is determined by subtracting $f_d$ from $f_c$.

A few additional test vectors are now generated to handle the faults in $f_u$ by using the serial scan mode. For this purpose, the constraints of the *ILS* structure imposed on the secondary inputs are first removed. Next, the *ATPG* is run with fault list $f_u$ to generate a set of patterns $T_S$ for the circuits. The combined set of test patterns $T$ is finally obtained by the union of $T_B$ and $T_S$. The outline of Algorithm_2 is presented in *Fig. 9*

## V. Experimental Results

The proposed algorithms are implemented in $C$ on a $SUN\ SPARC\ ULTRA-60$ workstation in $SOLARIS$ 5.8 environment and run on several $ISCAS'89$ benchmark circuits. Experiments were performed on each circuit with $0.18\mu m$ digital $CMOS-9$ standard cell library provided by the National Semiconductor. Each circuit is first synthesized by using the Design Analyzer tool of Synopsys. The test patterns are generated by the TetraMax tool from Synopsys. The goal of the experiments is to demonstrate (i) the reduction of the scan path length in the *ILS* structure compared to full scan circuits, and (ii) achieving high fault coverage in broadcast mode.

Results on scan wire length for these benchmark circuits are shown in *Table I*. The back-end phase of these circuits has been carried out by the Talus tool of Magma. The values of the scan wire length are given in $\mu m$. The first column in this table represents the number of segments of the *ILS* structure. The results show that the total wire length of the scan chain decreases when the number of segments in the *ILS* structure increases. When the number of braches in *ILS* structure is 1, the *ILS* structure reduces to a full scan chain structure.

*Table II*, and *Table III* show the fault coverage obtained in the broadcast and serial mode. For each circuit, the columns in these tables show the number of segments in the *ILS*, the number of test patterns and fault coverage obtained in broadcast mode, the number of test patterns in serial mode, the number of cycles required to test the circuit, and total fault coverage. As expected, the fault coverage in the broadcast mode decreases when the number of segments increases. The untestable faults are to be detected by using serial patterns. The test application time is computed as $n_{ILS}+(n_{ILS}+1)T_B+(1+n_f)T_s+T_s$, where $n_{ILS}$ is the number of flip-flops in the longest segment of the *ILS*, $n_f$ is the number of flip-flops in the $CUT$, $T_B$ is the number of test patterns in broadcast mode, and $T_s$ is the number of test patterns in serial mode. The results show that the proposed technique significantly reduces the test application time without degrading the fault coverage.

## VI. Conclusions

The *ILS* architecture is capable of reducing test application time and test data volume significantly. In this paper, we have proposed a layout-aware and coverage-driven design methodology for *ILS* architecture. The technique achieves significant improvements in fault coverage while at same time reduces the scan wire length to a great extent. The proposed design methodology is also suitable for a highly compact test set, i.e., when the flip-flops have very week or no compatibility under a test set.

## References


[1] N. Nicolici and B.M. Al-Hashimi, "Multiple scan chains for power minimization during test application in sequential circuits," IEEE Transactions on Computers, June 2002, vol. 51, pp. 721 – 734.

[2] Md.S. Quasemand and S. Gupta, "Designing reconfigurable multiple scan chains for systems-on-chip," Proc. IEEE VLSI Test Symposium, 25-29 April 2004, pp. 365 – 371.

[3] A. Chandra, F. Ng, and R. Kapur, "Low Power Illinois Scan Architecture for Simultaneous Power and Test Data Volume Reduction," Proc. Design, Automation and Test in Europe, 10-14 March 2008, pp. 462 – 467.

[4] A. Chandra, H. Yan, and R. Kapur, "Multimode Illinois Scan Architecture for Test Application Time and Test Data Volume Reduction," Proc. IEEE VLSI Test Symposium, 6-10 May 2007, pp. 84–92.

[5] M. Sharma, J.H. Patel, and J. Rearick, "Test data compression and test time reduction of longest-path-per-gate tests based on Illinois scan architecture," Proc. 21st VLSI Test Symposium, 27 April-1 May 2003, pp. 15 – 21.

[6] H.F Ko and N. Nicolici, "Functional Illinois scan design at RTL," Proc. IEEE International Conference on Computer Design: VLSI in Computers and Processors, 11-13 Oct. 2004, pp. 78 – 81.

[7] F.F Hsu, K.M. Butler, and J.H. Patel, "A case study on the implementation of the Illinois Scan Architecture," Proc. International Test Conference, 30 Oct.-1 Nov. 2001, pp. 538 – 547.

[8] M.A. Shah and J.H. Patel, "Enhancement of the Illinois scan architecture for use with multiple scan inputs," Proc. IEEE Computer society Annual Symposium on VLSI, 19-20 Feb 2004, pp. 167 – 172.

[9] A.R. Pandey and J.H. Patel, "Reconfiguration technique for reducing test time and test data volume in Illinois Scan Architecture based designs," Proc. IEEE VLSI Test Symposium, 28 April-2 May 2002, pp. 9 – 15.

[10] A.R. Pandey and J.H. Patel, "An incremental algorithm for test generation in Illinois scan architecture based designs," Proc. Design, Automation and Test in Europe Conference and Exhibition, 4-8 March 2002, pp. 368 – 375.

[11] Po-Chang Tsai and Sying-Jyan Wang, "Multi-Mode Segmented Scan Architecture with Layout-Aware Scan Chain Routing for Test Data and Test Time Reduction," Proc. Asia and South Pacific Design Automation Conference, 20-23 Nov. 2006, pp. 225 – 230.

[12] Sying-Jyan Wang, Kuo-Lin Peng, and Katherine Shu-Min Li, "Layout-Aware Scan Chain Reorder for Skewed-Load Transition Test Coverage," Proc. Asian Test Symposium, Nov. 2006, pp. 169 – 174.

[13] P. Gupta, A.B. Kahng, and S. Mantik, "Routing-aware scan chain ordering," Proc. Asia and South Pacific Design Automation Conference, 21-24 Jan. 2003, pp. 857 – 862.

[14] D. Berthelot, S. Chaudhuri, and H. Savoj, "An efficient linear time algorithm for scan chain optimization and repartitioning," Proc. International Test Conference, 7-10 Oct. 2002, pp. 781 – 787.

[15] P. Gupta, A.B. Kahng, I.I. Mandoiu, P.A.R. Sharma, and J.H. Patel, "Layout-aware scan chain synthesis for improved path delay fault coverage," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, July 2005, vol. 24, pp. 1104 – 1114.

[16] Douglas B. West, "Introduction to Graph Theory," Prentice-Hall of India, New Delhi, 2005.

**Algorithm_2: To generate the test patterns for the circuit and to compute test application time**

Input to the Algorithm_2: Complete information about the $ILS$ structure

1. Find the set of complete faults $f_c$ in the circuit.
2. Insert the input constraints by considering the $ILS$ structure.
3. Run the $ATPG$ incrementally to generate the set of test patterns $T_B$.
4. Use incremental fault simulation for each pattern of $T_B$ to compute the detectable faults. The detected faults are stored in $f_d$.
5. Compute the set of undetectable or hard-to-detect (HTD) faults $f_u = f_c - f_d$.
6. Set the circuit in serial scan chain mode.
7. Generate the set of serial test patterns $T_S$ for the target faults $f_u$ (using the $ATPG$).
8. Compute $T = T_B \bigcup T_S$.
9. Compute the test application time/test data volume for the complete test set $T$.

Fig. 9. Algorithm_2 to generate the complete set of test patterns

TABLE I

TOTAL WIRE LENGTH OF THE SCAN PATH W.R.T. NUMBER OF SEGMENTS

| Number of segments in $ILS$ | $s1423$ $WL$ $(\mu m)$ | $s5378$ $WL$ $(\mu m)$ | $s9234$ $WL$ $(\mu m)$ | $s13207$ $WL$ $(\mu m)$ | $s15850$ $WL$ $(\mu m)$ | $s35932$ $WL$ $(\mu m)$ |
|---|---|---|---|---|---|---|
| 1 | $9.1E+3$ | $5.1E+4$ | $7.5E+4$ | $2.3E+5$ | $1.2E+5$ | $5.0E+5$ |
| 2 | $8.5E+3$ | $4.1E+4$ | $6.1E+4$ | $1.4E+5$ | $0.9E+5$ | $4.3E+5$ |
| 4 | $7.5E+3$ | $3.6E+4$ | $5.1E+4$ | $0.8E+5$ | $0.3E+5$ | $3.2E+5$ |
| 6 | $7.1E+3$ | $2.9E+4$ | $4.8E+4$ | $9.6E+4$ | $8.8E+4$ | $2.8E+5$ |
| 8 | $5.7E+3$ | $2.0E+4$ | $3.9E+4$ | $9.2E+4$ | $8.2E+4$ | $2.5E+5$ |

TABLE II

RESULTS ON TEST CYCLE REDUCTION USING PROPOSED ALGORITHMS FOR $s9234$ AND $s13207$

| #segments in $ILS$ | $s9234$ | | | | | $s13207$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Broadcast mode | | Serial mode | total test cycles | total coverage (%) | Broadcast mode | | Serial mode | total test cycles | total coverage (%) |
| | #Test patterns | Fault coverage | #Test patterns | | | #Test patterns | Fault coverage | #Test patterns | | |
| 1 | - | - | 276 | 63432 | 98.25 | - | - | 412 | 276709 | 99.02 |
| 2 | 264 | 80.05 | 70 | 46468 | 98.12 | 384 | 79.70 | 72 | 178268 | 97.01 |
| 4 | 253 | 71.28 | 94 | 36232 | 97.29 | 367 | 70.20 | 134 | 119009 | 97.57 |
| 6 | 248 | 65.23 | 97 | 31903 | 96.31 | 323 | 61.24 | 110 | 120472 | 97.01 |
| 8 | 218 | 51.01 | 104 | 30395 | 96.17 | 315 | 56.33 | 128 | 112973 | 96.34 |

TABLE III

RESULTS ON TEST CYCLE REDUCTION USING PROPOSED ALGORITHMS FOR $s15850$ AND $s35932$

| #segments in $ILS$ | $s15850$ | | | | | $s35932$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Broadcast mode | | Serial mode | total test cycles | total coverage (%) | Broadcast mode | | Serial mode | total test cycles | total coverage (%) |
| | #Test patterns | Fault coverage | #Test patterns | | | #Test patterns | Fault coverage | #Test patterns | | |
| 1 | - | - | 330 | 197937 | 98.68 | - | - | 165 | 287013 | 98.54 |
| 2 | 354 | 82.11 | 47 | 134847 | 97.03 | 215 | 78.08 | 17 | 217960 | 95.81 |
| 4 | 349 | 66.05 | 73 | 96751 | 97.17 | 189 | 61.96 | 46 | 163531 | 95.86 |
| 6 | 315 | 60.64 | 88 | 84821 | 96.09 | 185 | 58.76 | 52 | 145389 | 93.04 |
| 8 | 288 | 57.81 | 83 | 71907 | 96.22 | 110 | 49.14 | 68 | 143276 | 93.55 |