ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 3, March 2015

Test Compaction of Logic Blocks by using Fault Identification Method

R.Vishnu Vardhan[1], M.Sathiskumar [2]

PG Scholar, VLSI Design, P.A. College of Engineering and Technology, Coimbatore, India [1]

Head of the Department, PG-ES, P.A. College of Engineering and Technology, Coimbatore, India [2]

Abstract: An arbitrary design implemented into a field-programmable gate array (FPGA). FPGA contains many logical blocks. Fault equivalence and fault dominance method are used to detect the fault with minimum time period. An approach provides transparent scan to share tests among different logic blocks whose primary inputs and outputs are included in scan chains even if the blocks have different numbers of state variables. The transparent-scan sequences based on tests for one logic block could detect faults in other logic blocks, with different numbers of state variable. It uses n number of test configuration instead of 2^n number of test configuration by test code algorithm. Transparent scan enhances the ability to produce a compact test set for a group of logic blocks. The procedure obtains a set of transparent-scan sequences for a group of logic blocks from compacted test sets for the logic blocks in the group. From this set, it chooses a subset that finds all the target faults, which are propagated by the complete set by using Modelsim and area is obtained by using the XILINX ISE 8.1 software.

Keywords: Full-scan circuits, test compaction, test generation, transparent scan, Field-programmable gate array (FPGA) testing.

I. INTRODUCTION

An approach to test application called transparent scan, the scan-select and scan-chain inputs of a scan circuit are considered as inputs of the sequential circuit in the same way as the primary inputs, and the scan-chain outputs are considered as outputs in the same way as the primary outputs. A test sequence under transparent scan specifies the values for all the inputs without distinguishing between them based on the types. The corresponding output sequence specifies values for all the outputs, again, without distinguishing between them based on their types. Faults are allowed to be detected during all the clock cycles of a transparent-scan sequence. In general, fault coverage is computed by sequential fault simulation of the transparent-scan sequence. This view of the test application process does not require any modifications to the scan design or the design of the circuit.. test data compression that consists of test vector compression on the input side and response compaction on the output side[3]-[4]. Test vector compression has been an active area of research. testing system-on-chips by applying huge amounts of test data, which is stored in the test memory and then transferred to the chip under test during test application[12]. Therefore, practical techniques, such as test compaction and compression, are used to reduce the amount of test data. The problem of compacting a set of test sequences for sequential circuits was modeled with the help of a covering matrix, where the test sequences can be modeled as columns with variable cost to reflect the cost (number of vectors) of covering selected subsets of circuit faults [6]. D-algorithm that is shown to be ineffective for the class of combinational logic circuits that is used to implement error correction and translation (ECAT) functions [11]. PODEM algorithm is a new test generation algorithm for combinational logic circuits. More patterns may be obtained than from standard ATPG programs [15]. However, fault coverage is much higher in all irredundant multiple as well as single stuck faults are detected and rectified. The test patterns are easily generated algorithmically either by program or hardware and the application of set covering models to the compaction of test sets, which can be used with heuristic test set compaction procedure[8]. For this purpose, recent and highly effective set covering algorithms are used. Compaction refers to a reduction in the test application timing, while at-speed testing points to the application of primary input sequences that contribute to the detection of delay defects [1]. The proposed procedure generates an initial test set that has a low test application time and consists of long sequences of primary input vectors applied consecutively, Experimental evidence suggests that the size of computed test sets can often be reduced by using set covering models and algorithms[14]. The test compression method consumes large area and the fault identification is a time consuming process is observed from the previous techniques. All of the previous works are mainly focused on test compression, to get the minimum transition in the fault dictionary.

In this case, the scan-select and scan-chain input sequences are such that the conventional test set is applied to the circuit by applying the transparent scan sequence. When a logic block is embedded in a designing, access to its primary inputs and primary outputs, as well as the state variables, for the purpose of test application may be available only serially through scan chains. Fig. 1 illustrates such a logic block *Bi* with a single scan chain. The scan chain is marked with dashed lines. The scan-select input of Bi is denoted by $SCSEL_i$, its scan-chain input by $SCINP_i$, and its scan-chain output by $SCOUT_i$. Under the model of Fig. 1, a transparent-scan sequence for Bi specifies values only for $SCSEL_i$ and $SCINP_i$. The

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 3, March 2015

output sequence specifies only the corresponding values of $SCOUT_i$.

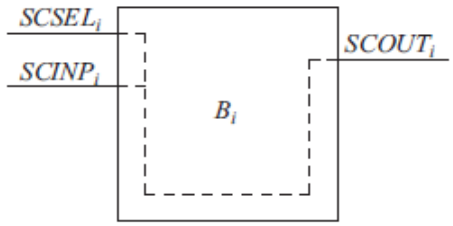

Fig. 1. Logic block of test compaction

II. PROPOSED METHOD

Transparent-scan sequences were allowed to assign arbitrary values to the scan-select input. In this case, the scan-select and scan-chain input sequences are such that the conventional test set is applied to the circuit by applying the transparent scan sequence. However, the scan-select sequence of the final transparent-scan sequence obtained is allowed to be different from the initial sequence. The scan-chain input and primary input sequences are also allowed to change relative to the initial sequences. The goal is to achieve test compaction for a single logic block using a single transparent scan sequence, and changing the sequences of the various inputs contributes to test compaction.

TABLE I
TRANSPARENT SCAN SEQUENCES

u	$T_{i,0}(u)$	$T_{i,1}(u)$	$T_{i,2}(u)$
0	11	11	11
1	11	10	10
2	10	11	10
3	10	10	11
4	0x	0x	0x
5	1x	1x	1x
6	1x	1x	1x
7	1x	1x	1x
8	1x	1x	1x

The transparent-scan sequences shown in Table I apply these tests to the circuit. Considering si, 0 = 0011, clock cycles 0 to 3 of Ti, 0 are scan clock cycles, and they have values T_i, 0(u, 0) = 1 for $0 \leq u \leq 3$. These clock cycles are used for loading the test 0011 into the scan chain. The values of the scan-chain input, T_i, 0(u, 1) for $0 \leq u \leq 3$, correspond to this test assuming that scan chains are shifted to the right. Clock cycle 4 is a functional clock cycle with T_i, 0(4, 0) = 0. This clock cycle is used for capturing the circuit response to 0011 in the scan chain. The scan-chain input value T_i,0(u, 1) can be determined arbitrarily, and it is marked with an "x" in Table I. Clock cycles 5 to 8 are scan clock cycles with T_i,0(u, 0) = 1 for $5 \leq u \leq 8$. They allow the response of the circuit to 0011 to be scanned out and observed. The values of the scan-chain input T_i,0(u, 1), for $5 \leq u \leq 8$, can be determined arbitrarily. They can be used for overlapping the test with the next test. For example, a two-pattern broadside test for a logic block with ki state variables would have two functional clock cycles between two scan subsequences of length k_i.

For the transparent-scan sequences considered in this paper, it is also possible to use combinational fault simulation instead of sequential fault simulation. This can be achieved by applying the following process.

1) The present state for the functional clock cycle can be computed without logic or fault simulation based on the values of the scan-chain input during the scan clock cycles that define the scan-in operation at the beginning of the test.

2) Combinational fault simulation is required for the functional clock cycle.

3) Based on the fault effects that are propagated to the flipflops during the functional clock cycle, and the number of scan clock cycles for the scan-out operation at the end of the test, it is possible to compute which fault effects will reach an output.

The size of the test set impacts the test storage requirements and time for test application, especially for the circuits using scan design. The test application time is directly proportional to the product of the number of test patterns and the number of scan cells in the longest scan chain. This necessitates generation of small test sets. The complexity of the compaction process plays an important role in test compaction. There are computation-intensive procedures proposed in the literature that produce minimal size test sets close to the lower bound. For instance, in tests are generated repeatedly which can detect several faults at the same time so as to replace previous tests found.

A. Static Compaction

Static compaction is applied as a post processing step to already generated test sets, to reduce the test set size further and therefore is independent of the test generation method. Static compaction is performed after all the patterns are generated and this is independent of test generation. The complexity of the compaction process plays an important role in test compaction. There are computation-intensive procedures proposed in the literature that produce minimal size test sets close to the lower bound. For instance, in tests are generated repeatedly which can detect several faults at the
Same time so as to replace previous faults found. Though these methods produce small test sets, they are not suitable to large designs.

B. Dynamic Compaction

Dynamic compaction is interlinked within the test generation process where a test cube is generated for a fault and the generated test cube is added as constraints to the next targeted fault. The advantage of dynamic compaction and static compaction is that it reduces the time required for post-processing step for compacting test patterns. The dynamic compaction begins with a fault which is on top of previously
Ordered fault list, called as primary fault. The primary fault is targeted for test generation and if a test is generated for the fault, another fault called secondary fault

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 3, March 2015

is picked and a test generation for the fault is attempted The test generation tries to generate a test for the secondary fault with the primary input values and scan cell values specified by previously generated test vector.
Test generation under the proposed approach, which eliminates the distinction between scan operations and application of primary input vectors, can be done as follows. The circuit for which test generation is carried out is Cscan. This circuit has two extra primary inputs compared to the original circuit C: the scan-in input scan_inp, and the scan-select input scan_sel. It also has an extra primary output, the scan output scan_out. The procedure will produce a test sequence where scan_sel and scan_inp are used as conventional primary inputs, and fault effects may be observed on scan_out. An example of such a test sequence is shown in Table II.

TABLE II
TEST SEQUENCE

Sequence	a1	a2	a3	a4	Scan_sel	Scan_inp
0	0	0	1	0	0	0
1	1	1	0	1	0	0
2	0	0	1	0	0	0
3	0	0	0	0	0	0
4	0	0	0	1	0	0
5	0	0	0	0	1	0
6	0	0	0	0	0	0
7	0	0	0	0	1	0
8	0	0	0	1	0	0
9	1	0	0	0	0	0
10	0	0	0	1	0	0
11	0	0	0	0	0	1
12	0	0	0	1	0	0
13	0	0	0	0	1	0
14	0	0	0	0	1	1
15	0	0	0	1	0	0
16	1	0	0	0	1	0
17	0	0	0	1	0	0
18	0	0	0	0	1	1
19	0	0	0	0	1	0
20	0	0	0	0	0	0
21	0	1	0	0	0	0
22	0	0	1	0	0	0
23	1	0	0	1	0	0
24	0	0	0	0	0	0

This sequence was generated for s 27scan, which is the scan version of ISCAS-89 benchmark circuit s 27. The circuit has four primary inputs labeled a1,a2,a3,a4. It has three state variables. It is interesting to note that scan is applied for a single time unit at time unit 5, 7 and 16. In addition, it is applied for two consecutive time units at time units 13, 14 and 18, 19. Thus, all the scan operations are limited scan operations with one and two shifts of the scan chain, and there is never a complete scan operation that takes three shifts of the scan chain.

C. Fault Identification Method

In order to detect all faults in the fault list, faults must be sensitized using a set of single-term functions and test vectors shown in Fig. 2. These single-term functions are implemented in all LUTs used in the user design. The single-term functions implemented in the user LUTs correspond to a test configuration which detect the interconnect faults sensitized in that test configuration. The objective is to come up with a minimum number of test configurations such that all faults in the fault list are sensitized and, hence, detected in at least one test configuration. Testing for bridging faults has always been a challenging issue, particularly for ASICs. This is mainly due to the fact that finding an appropriate fault list for bridging faults is not as straightforward as that for stuck-at faults. The number of all possible single stuck-at faults in a circuit is linear with the size of the circuit whereas the number of all pairwise bridging faults is quadratic with the size of the circuit. Activating all possible faults (stuck-at, open, and pairwise bridging faults) for M nets performed using only$[\log_2(M+2)]$ test vectors. These vectors are columns of binary representations of numbers 1 to using bits and called test codes.

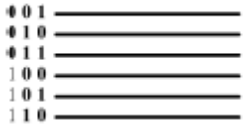

Fig. 2. Logarithmic test set to activate all faults for six wires

D. Controllability

Controllability refers to the ability to apply test patterns to the inputs of a sub circuit via the primary inputs of the circuit. To enhance the controllability of a circuit, the state that cannot be controlled from its primary inputs has to be reduced. These conditions are ensured by the enforcement of certain design rules, particularly pertaining to the clocks that evoke state changes in the network. Scan refers to the ability to shift into or out of any state of the network.

E. Observability

Observability refers to the ability to observe the response of a sub circuit via the primary outputs of the circuit or at some other output points. To enhance the Observability, output of the gate must be separately observed. Latches are used in pairs; each has a normal input data, output data and clock for system operation. For testing operation, the two latches form a master/slave pair with one scan input and scan output and non-overlapping scan clocks A and B which are held low during system operation but cause the scan data to be latched when high pulsed during scan. If the trace is shorted to another signal or if the trace signal is open, the correct signal value does not show up at the destination pin, indicating a fault. The purpose of the testing is to identify the presence of the defects in the circuit. By understanding that, the circuit has defects if it observes incorrect behavior. Fault simulation has to determine the fault coverage for a specified set of test

DOI 10.17148/IJARCCE.2015.4391

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 3, March 2015

vectors applied to a CUT, fault simulation is carried out. For each fault expected in the CUT (excluding redundant faults), the output produced when a test vector is applied to a faulty circuit differs from the output produced in a fault –free circuit. Thus, fault simulation that can be used for this purpose, some commercial and others academic. It then selects a subset of these sequences that is sufficient for detecting all the target faults that are detected by S_0, S_1,. . ., S_{n-1}. For $0 \leq i < n$ and for $0 \leq j < m_i$, the procedure translates si, j into a transparent-scan sequence $T_{i,j}$ as described in the previous section. It then adds $T_{i,j}$ to T . It is possible to use a set covering procedure in order to select a subset of T that detects all the faults in F.

However, a set covering procedure requires information about all the sequences from T that detect every fault in F. In the context of transparent scan, the two steps proceed as follows.

Step 1 selects a subset $Tsel_1 \subseteq T$ by identifying faults from F that are detected by unique sequences in T . If a fault $f \in F$ has only one transparent-scan sequence $T_{i,j} \in T$ that detects it, $T_{i,j}$ must be included in the selected subset of transparent scan sequences. In this case, $T_{i,j}$ is included in $Tsel_1$. Step 2 selects additional transparent-scan sequences as necessary to produce a subset $Tsel_2$ that detects all the faults in F. The details of the two steps are described next. Step 1 performs two-detection fault simulation of the faults in F under the transparent-scan sequences in T. The number of detections of a fault $f \in F$ is equal to the number of sequences from T that detect the fault. The two-detection fault simulation procedure drops a fault from further simulation after it finds two transparent-scan sequences in T that detect the fault. It stores the number of times a fault $f \in F$ is detected during this process in a variable denoted by ndet (f). It stores the index of the first transparent-scan sequence that detects a fault $f \in F$ in a variable denoted by first (f). If, at the end of this process, a fault $f \in F$ has ndet (f) = 1, the sequence with index first (f) must be selected. This sequence is included in Tsel1.

The expectation is that the sequences in $Tsel_1$ will detect most of the faults in F. Tsel1 is guaranteed to detect a fault f with ndet (f) = 1. For a fault f with ndet (f) = 2, there are two or more options for transparent-scan sequences in T that detect it. Such a fault is likely to be detected by $Tsel_1$. With a small number of faults that are not detected by $Tsel_1$, Step 2 uses fault simulation with fault dropping to select additional sequences so as to detect all the faults in F

Step 2 starts by assigning $Tsel_2 = Tsel_1$. It performs fault simulation with fault dropping of F under the transparent-scan sequences in $Tsel_2$ in order to remove from consideration faults that are already detected. It then performs fault simulation with fault dropping of F under $T-Tsel_2$. A fault $f \in F$ is detected by a transparent-scan sequence vectors as a logic module.

F. Fault Equivalence Method

It is possible that two or more faults produce same faulty behavior for all input patterns are called equivalent faults. Any single fault from the set of equivalent fault set can represent the whole set. In this case, much less than k×n fault tests are required for a circuit with n signal line removing equivalent faults from entire set of faults is called fault collapsing that significantly decreases the number of faults to check.

G. Fault Dominance Method

Fault F is called dominant to F' if all tests of F' detects F. In this case, F can be removed from the fault list. If F dominates F' and F' dominates F, then these two faults are equivalent. Two faults are functionally equivalent if they produce identical faulty functions or two faults are functionally equivalent if those cannot distinguish them at primary outputs (PO) with any input test vector. For instance, tests are generated repeatedly which can detect several faults at the same time so as to replace previous faults found.

III. SIMULATION RESULTS

A. Fault Equivalence Output for Test Compaction

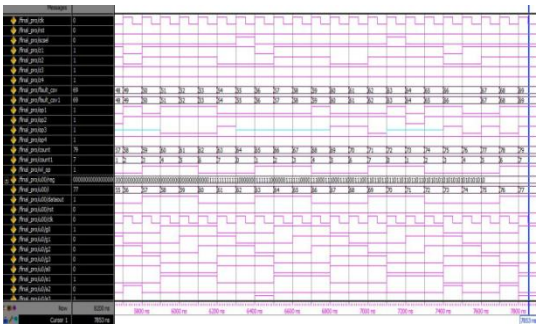

Fig. 3. Fault Equivalence Output for Test Compaction

Figure 3 finds the total numbers of faults that are present in the circuit. This method increases the fault identification and the total number of faults can be identified. This test compaction also comprises of two set of fault coverage. Here , both the fault coverage identifies the equal number of faults present in the circuit.

B. Fault Dominance Output for Test Compaction

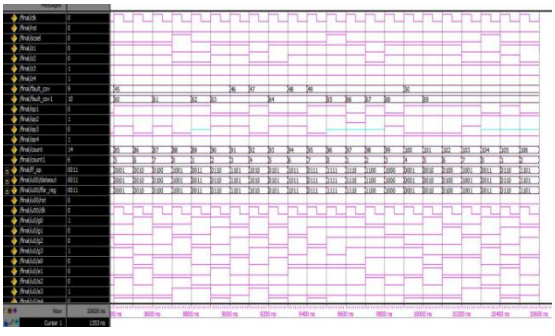

Fig. 4. Fault Dominance Output for Test Compaction

Figure 4 shows the faults that are dominated by another set of fault in the test compaction. There are two set of fault coverage in the test compaction. One set of fault coverage is for the logic block used and another set of fault coverage is for the dominance circuit. Second set of fault coverage identifies the total number of fault within minimum time period.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 3, March 2015

C. Fault identification in s27 benchmark circuit

Fig. 5. Fault Identification Output in s27 Benchmark Circuit

Figure 5 shows the result of the fault identification of s27 benchmark circuit. Stuck –at fault in the circuit can be identified by the values stored in the latch. Both the Stuck –at 1 and Stuck – at 0 can be identified. Stuck –at 1 is obtained for the values of latch as 1 and Stuck –at 0 is obtained for the values of latch as 0

D. Fault identification in s208 benchmark circuit

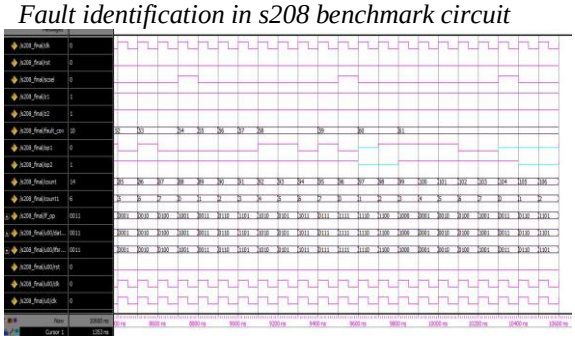

Fig. 6. Fault Identification Output in s208 Benchmark Circuit

Figure 6 shows the result of s208 benchmark circuit from which the fault coverage is obtained. Comparing to the previous techniques fault coverage of 61 is obtained in this benchmark circuit with minimum number of test vectors is obtained. The fault coverage value is obtained during the transition in the LFSR values. During the linear propagation of the shift register, fault coverage is identified.

E. Comparison of Fault Coverage and Test Vectors for Different Methods:

Table III shows comparison of test vector and fault coverage using test code algorithm. The proposed technique uses test code algorithm method with reduced number of test vectors and also it provide high fault coverage compare to the existing method.

Fault Equivalence and Dominance method is 20% more efficient than PODEM and Boolean Difference method. For the proposed method, by using the fault equivalence and fault dominance method the number of test vectors used is minimum and the fault coverage is high. From the comparison, Boolean Difference method is 12% efficient than PODEM algorithm, Fault Equivalence and Dominance method is 20% more efficient than PODEM and Boolean Difference method.

TABLE III
COMPARISON OF FAULT COVERAGE AND TEST VECTORS FOR DIFFERENT METHODS

Benchmark circuits	Boolean Difference Method **[El-Maleh A. and Osais (2004)]**		Fault Equivalence and Dominance Method**[PHASE –I WORK]**		Fault Identification Method **[PHASE –II WORK]**	
	Test Vector	FC	Test Vector	FC	Test Vector	FC
s1428	100	50	120	52	120	64
s298	160	61	160	63	150	70
s27	120	50	100	50	100	67
s208	100	52	100	61	90	72
s27-s208	110	60	80	69	80	81

IV. CONCLUSION

Fault identification method used in the test compaction, it minimizes the total number of test vectors used in the circuit. This project describes a test compaction procedure under transparent scan for groups of logic blocks whose primary inputs and outputs are scanned. Using test code algorithm it reduces 2^n number of test vector sequence to n number of test vector. Experimental results showed that transparent-scan sequences based on tests for one logic block could detect faults in other logic blocks, with different number of state variables. This allowed a reduced number of transparent-scan sequences to be used for the group. Transparent-scan sequences of logic blocks with higher numbers of state variables typically detected faults of logic block with smaller numbers of state variables. This was the main contributor to the reduction in the number of transparent scan sequences for the group using test code algorithm.

REFERENCES

[1] Barnhart C. and Brunkhorst V. (2001), 'OPMISR: The foundation for compressed ATPG vectors,' in Proc. Int. Test Conf., pp. 748–757.
[2] Boateng K.O. and Nakata T.M. (2001), 'A method of static compaction of test stimuli,' in Proc. Asian Test Conf., pp. 137–142
[3] Chang J.S. and Lin C.S. (1992), 'Test set compaction for combinational circuits,' in Proc. Asian Test Conf., pp. 20–25.
[4] Dimopoulos M. and Linardis P. (2003), 'Accelerating the compaction of test sequences in sequential circuits through problem size reduction,' IEEE Trans. On Comput. Aided Design, Vol. 22, No. 10, pp. 1443–1449.
[5] Dimopoulos M. and Linardis P. (2004), 'Efficient static compaction of test sequence sets through the application of set covering techniques,' in Proc. Design, Int. Test Conf., pp. 194–199.
[6] El-Maleh A.H. and Osais Y.E. (2003), 'Test vector decomposition-based static compaction algorithms for combinational circuits,' IEEE Trans. On Comput. Aided Design, Vol. 8, No. 4, pp. 430–459.
[7] Flores P.F. and Marques-Silva J.P. (2000), 'On applying set covering models to test set compaction,' IEEE Trans. on Comput. Aided Design, Vol. 22, No. 10, pp. 1443–1449
[8] Hamzaoglu I. and Patel J.H. (1998), 'Test set compaction algorithms for combinational circuits,' in Proc. Int. Conf. Comput. Aided Design, pp. 283–289
[9] Hochbaum D.S. (1996), 'An optimal test compression procedure for combinational circuits,' IEEE Trans. on Comput. Aided Design, Vol. 15, No. 10, pp. 1294–1299.
[10] Kajihara S. and Reddy S.M. (1995), 'Cost-effective generation of minimal test sets for stuck-at faults in combinational logic circuits,' IEEE Trans. On Comput. Aided Design, Vol. 14, No. 12, pp. 1496–1504.
[11] Koenemann B. and Wheater D. (2001), 'A Smart BIST variant guaranteed encoding,' in Proc. Asian Test Conf., pp. 325–330.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 3, March 2015

[12] Lee K. and Huang C. (1998), 'Using a single input to support multiple scan chains,' in Proc. Int. Conf. Comput. Aided Design, pp. 74–78.
[13] Pomeranz I. and Reddy S.M. (1998), 'A new approach to test generation and test compaction for scan circuits,' in Proc. Int. Conf. Compt. Aided Design, pp. 283–289
[14] Pomeranz I. and Reddy S.M. (1991), 'COMPACTEST: A method to generate compact test sets for combinational circuits,' in Proc. Int. Test Conf., pp. 194–203
[15] Rajski J. and Tsai K.H. (2002), 'Embedded deterministic test for low cost manufacturing test,' in Proc. Int. Test Conf., pp. 301–310.