International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2015): 6.391

Test Pattern Generation for Integrated Circuit Test Using Distance between Vectors

Emad Aldeen Widaat Alla Musa[1], Dr. Abdelrahman Elsharif Karrar[2], Dr. Abu Khari Bin Aain[3]

[1]Faculty of Engineering, Kordofan University, Sudan

[2]College of Computer Science and Engineering, Taibah University, Saudi Arabia

[3]Faculty of Electrical Engineering, University Technology Malaysia, Malaysia

Abstract: ***Random testing is a well-known concept that requires that each test is selected randomly regardless of the test previously applied. In actual practice it takes the form of pseudo-random testing. A well-known method for generating test vector is pseudo-random number generator which is based on LFSR. Recently the anti-random sequence has attained attraction and shown to have better statistical properties compared to pseudo-random numbers. In antirandom testing, in the previous researches, the test vector are generated by using the hamming or Cartesian distance techniques and in the random testing, the test vector are pseudorandom numbers generated in the MATLAB(1).This research aims to develop a test pattern generation (TPG) algorithm using black box and possible with some deterministic approach which could produce comparable fault coverage compared to some of well known TPG methods such as random and anti random, one other purpose of this research to a achieve high fault coverage using less number of test vector with maximum distance. Fault coverage of two test vector is related to the distance between them, two input vector with large distance between them can give greater number of fault detected. The work will be heavily based on understanding and analyzing the effect of test vector distance from one another and fault coverage. The development tools that will be utilized in this research would be Matlab and "Atalanta" fault simulator.***

Keywords: Random testing, fault coverage , Atalanta, Matlab.

1. Introduction

The distance between test vectors play an important role in fault coverage as maximum distance obtained then, high fault coverage is applied, Random test vector approach treats test circuit as a black box.

A typical fault detection curve (1) during fault simulation is shown in Fig.1.1 When simulation begins, a large percentage of faults are detected in a short amount of time. However, as time goes on, the rate at which faults are detected decreases because the test patterns applied detect many faults that have already been detected. If these detected faults are not dropped, extra time is spent on re-simulating these faults but the fault coverage remains the same.

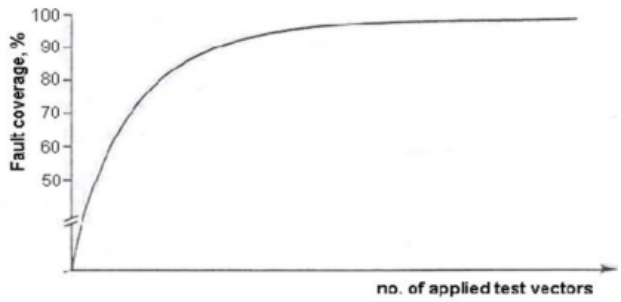

Figure 1.1: Fault Coverage Versus Random Test Vectors

In this research both Matlab language and Atalanta fault simulation software are used to generate better fault coverage with less test vector.

1.1 Problem statement

Achieving more fault coverage with minimum test vector is the main important aspect in IC testing and for this purpose there's a need to generate minimum test vectors with maximum distance between them to improve fault coverage.

1.2 Objectives

The objective of this research are :

- Develop an algorithm using Matlab to reduce test vector and get better fault coverage.
- Develop an algorithm to calculate distance between test vectors to produce vectors with large distance between them which can cover greater number of fault detection.

1.3 Scope

This research is focusing on building an algorithm to reduce test vector and improve the distance between these vectors to achieve better fault coverage.

The tools which is going to be used in this research is the Matlab programming language and Atalanta fault simulator software.

1.4 Research Methodology

The methodology is built on designing an algorithm that covers the following methods to improve fault coverage:

- Generating different test vector with different input pins and studying the per centage of fault coverage for different benchmark circuits.

- Calculating hamming distance and total hamming distance and try to gain maximum distance between test vectors, two input vector with large distance between them can give greater number of fault detected.

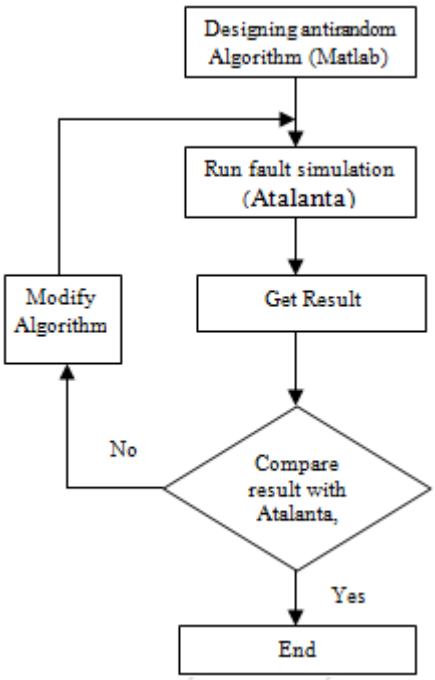

Figure 1.2: Research methodology flowchart

1.5 Random and Antirandom Testing

Random testing is a form of black-box testing which does not require knowledge of the circuit under test. It avoids the problem of deterministic test generation using structural information about the circuit under test1. one more things about black box which it use all test vector for testing purpose unlikely deterministic method which it use a fraction of test vector.

Anti-random testing technique is a variant of random testing. It was proposed by Malayia [2]. As in the case of random testing it is a black-box strategy. This means that it assumes no information about the internal implementation of the object under test. Anti-random testing is based on the idea that test cases have to be selected to have maximum distance from each other.

1.6 Test Pattern Generation Techniques

An automatic test pattern generation (ATPG) and fault simulation technique is used to generate the test patterns, depending upon the desired fault coverage and the specific faults to be tested for, a sequence of test vectors is developed for the circuit under test (CUT). The function of the TPG is to generate these test vectors and apply them to the CUT in the correct sequence.

1.7 Fault coverage and simulator:

Fault coverage refers to the percentage of some type of fault that can be detected during the test High fault coverage is particularly valuable during manufacturing test, and techniques.

The fault simulators which have been studied in this research are Matlab and Atalanta. To check if the test vector is effective there is a need to simulate these test vectors the simulation tools are listed below:

1) Atalanta, is an automatic test pattern generator and a fault simulator for stuck-at faults in combinational circuits. This tool has been developed by researchers from Bradley Department of Electrical and Computer Engineering, Virginia Polytechnic Institute & State University
2) Matlab programming language which is a high-level technical computing language can be used to generate test pattern generator.

1.8 Test Bench Circuit

ISCAS 1995 conference had introduced different combinational benchmark circuits such as c17, c432 and c880. Table 1 provides information about the test bench circuits normally used to verify the effectiveness of test patterns.

Table 1: ISCAS85 Bench circuit

Circuit name	*Input*	*output*	*No. of gates*
C17	5	2	6
c432	36	7	120
C499	41	32	162
C880	60	26	320
C1355	41	32	506
C1908	33	25	603
C2670	233	144	872
C3540	50	22	1179
C5315	178	123	1726
C6288	32	32	2384
C7552	207	108	2636

2. Experiments Results

This section present the results which have been obtained through Matlab programming and Atalanta fault simulator.

2.1 THD of random sequences

In this research, the analysis of constructed random sequence using Matlab programming language is studied. A sample of a random sequence which implemented by Matlab programming language with 15 input bits and 10 test vectors have been constructed and its THD has been calculated. Table 2 and Table 3 show the random sequence and its total Hamming distance respectively.

Table 2: Random test vector generated by Matlab programming language

	Random sequence														
t0	0	0	1	0	1	0	0	0	0	1	0	1	1	0	0
t1	1	1	1	1	1	0	0	0	0	1	1	1	1	0	0
t2	1	1	1	1	0	1	0	1	0	0	0	0	1	0	0
t3	0	1	1	1	1	1	0	0	0	1	0	0	0	0	0
t4	1	1	0	0	1	1	0	0	1	1	0	1	1	1	1
t5	0	0	1	0	0	1	0	0	0	1	0	1	0	0	1
t6	0	1	1	0	1	0	1	1	1	1	1	1	1	0	0
t7	1	1	1	0	1	0	1	0	0	1	1	0	1	1	0
t8	1	1	0	0	1	0	1	0	1	1	0	0	1	1	1
t9	1	1	1	1	0	1	0	1	1	0	1	0	0	0	0

The total Hamming distance for these test vectors is calculated as:
THD for t_0 =0
THD for t_1 =(t_1,t_0) = 4
THD for t_2 = $(t_2,t_1)+(t_2,t_0)$ = 14
THD for t_3 = $(t_3,t_2)+(t_3,t_1)+(t_3,t_0)$ = 15
THD for t_4 = $(t_4,t_3)+(t_4,t_2)+(t_4,t_1)+(t_4,t_0)$ = 31
THD for t_5 =$(t_5,t_4)+(t_5,t_3)+(t_5,t_2)+(t_5,t_1)+(t_5,t_0)$ = 32
THD for t_6 =$(t_6,t_5)+(t_6,t_4)+(t_6,t_3)+(t_6,t_2)+(t_6,t_1)+(t_6,t_0)$ = 44
THD for t_7 =$(t_7,t_6)+(t_7,t_5)+(t_7,t_4)+(t_7,t_3)+(t_7,t_2)+(t_7,t_1)+(t_7,t_0)$ = 47
THD for t_8 =$(t_8,t_7)+(t_8,t_6)+(t_8,t_5)+(t_8,t_4)+(t_8,t_3)+(t_8,t_2)+(t_8,t_1)+(t_8,t_0)$ = 60
THD for t_9 = $(t_9,t_8)+(t_9,t_7)+(t_9,t_6)+(t_9,t_5)+(t_9,t_4)+(t_9,t_3)+(t_9,t_2)+(t_9,t_1)+(t_9,t_0)$ = 74

Table 3: Random test vector with its THD

Test vector	THD
t0	0
t1	4
t2	14
t3	15
t4	31
t5	32
t6	44
t7	47
t8	60
t9	74

2.2 Generation of new sequence (NQ1) from random Sequence

The idea of this experiment is to generate anew sequence NQ1 using rule 90 from the previously generated random sequence. The NQ1 that is going to be generated is based on 5 bits. To produce the NQ1 sequence the random sequences is divided into a group of three bits as:
$(q_0q_1q_2)(q_3q_4q_5)(q_6q_7q_8)(q_9q_{10}q_{11})(q_{12}q_{13}q_{14})$
And every group will present one bit of the NQ1 sequence.
For NQ1_rule90 formula (1) is applied:

$$q_i(t+1) = q_{i-1}(t) \oplus q_{i+1}(t) \quad (1)$$

For NQ1 rule150 formula (2) is applied:

$$q_i(t+1) = q_{i-1}(t) \oplus q_{i+1}(t) \oplus q_i(t) \quad (2)$$

By implementing these equations, two types of the new sequence NQ1 could be generated from the random sequence. Table 4 and Table 5 show the sequence of NQ1 rule 90 and NQ1 rule 150 from random sequence respectively.

Table 4: Random Vs NQ1 rule 90

Random								NQ1_Rule 90								
0	0	1	0	1	0	0	0	1	1	0	0	1	0	0	0	1
1	1	1	1	1	0	0	1	1	1	0	0	0	1	0	0	1
1	1	1	1	0	1	0	0	0	1	0	0	0	0	0	0	1
0	1	1	1	1	1	0	0	0	0	0	0	1	0	0	1	0
1	1	0	0	1	1	0	0	1	1	1	1	1	1	1	0	0
0	0	1	0	0	1	0	0	1	0	0	1	1	1	0	0	1
0	1	1	0	1	0	1	1	1	1	0	0	1	0	0	0	1
1	1	1	0	1	0	1	1	0	1	1	0	0	0	1	1	1
1	1	0	0	1	0	1	1	0	1	1	1	1	1	0	1	1
1	1	1	1	0	1	1	0	1	0	1	1	0	0	1	1	0

Table 5: Random Vs NQ1 rule 150

Random								NQ1_Rule 150								
0	0	1	0	1	0	0	1	0	1	1	0	0	0	0	1	
1	1	1	1	1	0	0	1	1	1	1	0	0	0	1	1	
1	1	1	1	0	1	0	0	0	0	1	0	0	1	0	1	
0	1	1	1	1	0	0	1	0	0	0	0	0	0	1	0	
1	1	0	0	1	0	1	1	0	1	1	1	1	1	0	1	
0	0	1	0	0	0	0	1	0	1	0	0	1	0	0	1	
0	1	1	0	1	1	1	1	1	1	1	0	0	1	1	1	
1	1	1	0	1	0	0	1	1	0	1	1	0	1	0	0	
1	1	0	0	1	0	1	1	0	0	1	1	1	0	1	1	
1	1	1	1	0	1	1	0	1	0	0	0	0	0	1	0	

Table 6: THD for NQ1 rule 90

NQ1_Rule 90					THD
1	0	0	0	1	0
0	1	0	0	1	2
0	0	0	0	1	2
1	0	0	1	0	9
1	1	1	0	0	13
1	1	0	0	1	9
1	0	0	0	1	9
0	0	1	1	1	22
1	0	0	1	0	20
0	0	1	0	0	25

Table 7: THD for NQ1 rule 150

NQ1_Rule 150					THD
1	1	0	0	1	0
1	0	0	1	1	2
1	0	1	0	1	4
0	1	0	1	0	11
0	0	1	0	1	11
1	1	0	0	1	10
0	1	1	1	1	16
1	1	1	0	0	19
0	1	0	1	1	19
1	0	0	1	0	26

The graph in Fig. 2.1 shows the relation between the THD for random and NQ1 rule 90 and 150.

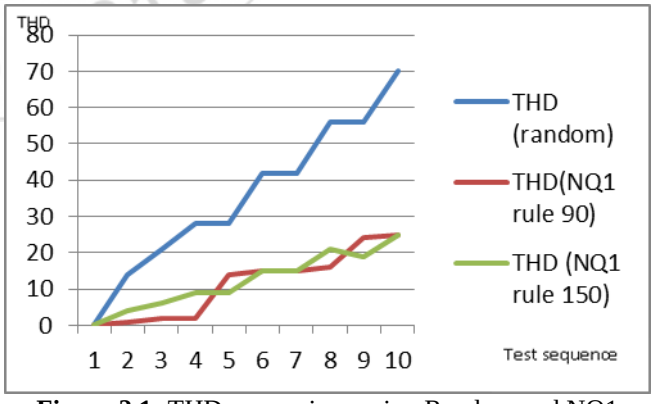

Figure 2.1: THD comparison using Random and NQ1

From the graph, THD (Random) has the biggest distance compared to NQ1 THD (rule 90 and rule 150)

2.4 Calculation of HD

The Hamming Distance for Random number, NQ1 rule 90 and rule 150 have also been calculated.

The Hamming Distance for these test vectors is calculated as:

HD for t_0 =0
HD for t_1 =(t_1,t_0) = 4
HD for t_2 = (t_2,t_1) = 6
HD for t_3 = (t_3,t_2 = 5
HD for t_4 = (t_4,t_3) = 8
HD for t_5 =(t_5,t_4) = 7
HD for t_6 =(t_6,t_5) = 9
HD for t_7 =(t_7,t_6)= 5
HD for t_8 =(t_8,t_7) = 4
HD for t_9 =(t_9,t_8) = 11

Table 8, 9 and 10 show the HD for Random, NQ1 rule 90 and NQ1 rule 150 respectively.

Table 8: Random test vector with its equivalent Hamming Distance

Random	HD
t0	0
t1	4
t2	6
t3	5
t4	8
t5	7
t6	9
t7	5
t8	4
t9	11

Table 9: HD for NQ1 rule 90

NQ1_Rule 90					HD
1	0	0	0	1	0
0	1	0	0	1	2
0	0	0	0	1	1
1	0	0	1	0	3
1	1	1	0	0	3
1	1	0	0	1	2
1	0	0	0	1	1
0	0	1	1	1	3
1	0	0	1	0	3
0	0	1	0	0	3

Table 10: HD for NQ1 rule 150

NQ1_Rule 150					HD
1	1	0	0	1	0
1	0	0	1	1	2
1	0	1	0	1	2
0	1	0	1	0	5
0	0	1	0	1	4
1	1	0	0	1	3
0	1	1	1	1	3
1	1	1	0	0	3
0	1	0	1	1	4
1	0	0	1	0	3

Fig. 3.2 Highlights the comparison of HD between NQ1 rule 90, 150 and Random.

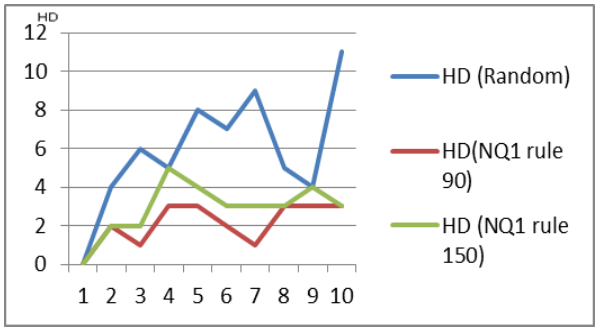

Figure 3.2: HD comparison for Random and NQ1

The graph shows Random technique has the highest HD followed by NQ1 rule 150.

2.5 Generation of new sequence (NQ):

A trial has been made to generate a new test sequence by adding the Random sequence with NQ1 rule150. The lowest 5 bits of 15 bits Random sequence are added to 5 bits NQ1. The new sequence is shown in Table 11.

Table 11: New sequence

	New sequence														
t0	0	0	1	0	1	0	0	0	1	0	0	0	1	0	1
t1	1	1	1	1	1	0	0	0	1	0	0	1	1	1	1
t2	1	1	1	1	0	1	0	1	0	0	1	1	0	0	1
t3	0	1	1	1	1	1	0	0	0	1	0	1	0	1	0
t4	1	1	0	0	1	1	0	0	1	1	1	0	1	0	0
t5	0	0	1	0	0	1	0	0	1	0	0	0	0	1	0
t6	0	1	1	0	1	1	0	0	0	0	0	1	0	1	1
t7	1	1	1	0	1	0	1	0	1	0	1	0	0	1	0
t8	1	1	0	0	1	0	1	0	1	1	1	0	0	1	0
t9	1	1	1	1	0	1	0	1	1	1	0	0	0	1	0

Then the HD for this new test sequence is calculated as:

HD for t_0 =0
HD for t_1 =(t_1,t_0) = 5
HD for t_2 = (t_2,t_1) = 7
HD for t_3 = (t_3,t_2 = 7
HD for t_4 = (t_4,t_3) = 8
HD for t_5 =(t_5,t_4) = 8
HD for t_6 =(t_6,t_5) = 5
HD for t_7 =(t_7,t_6)= 7
HD for t_8 =(t_8,t_7) = 2
HD for t_9 =(t_9,t_8) = 7

Table 12 shows the HD of the new sequence.

Table 12: HD of new sequence

Test vector	HD
t0	0
t1	5
t2	7
t3	7
t4	8
t5	8
t6	5
t7	7
t8	2
t9	7

The comparison between HD of new sequence and Random sequence shows that HD for Random shows higher distance.

Table 13: Random test vector with its equivalent hamming distance

Test vector	HD
t0	0
t1	4
t2	6
t3	5
t4	8
t5	7
t6	9
t7	5
t8	4
t9	11

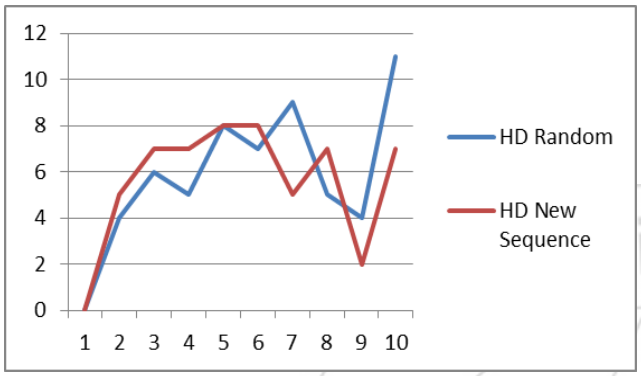

Figure 3.3: HD comparison for Random and new Sequence

By replacing the vector which has higher distance with the ones with lower distance (t6, t8, t9) then anew sequence will appear which is a combination (HD_Col) ,The graph below shows that HD_Col achieved better distance

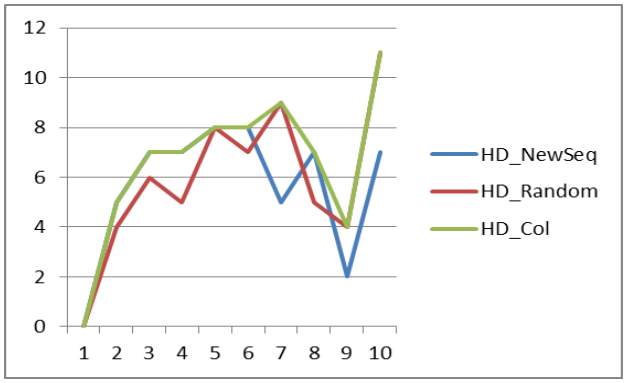

Figure 3.4: HD comparison for Random and new Sequence and the combination (HD_Col)

2.6 Analysis of fault coverage in relation to number of test sequence

This section explains the results of experiments which have been carried out to analyse the relationship between the fault coverage and number of test sequence. The work is based on fault simulation using Atlanta. The test sequence which used in this experiment have been generated using Matlab random sequence. Once the test sequence has been generated it is exported to run in Atlanta for fault simulation.

Table 13 and Fig.3.5 show that a small number of test sequence is only needed to achieve high fault coverage (> 85%). For example test bench C432 has 36 primary inputs (PI). For exhaustive test all 236 (68,719,476,736) would have to be used. However, only 100 test sequence is needed to achieve 88.7% of fault coverage.

Table 14: Fault coverage from random test sequence generated from Matlab.

	No. of input	Max. No. of test	No. of test pattern applied							
			50	100	200	300	400	500	1000	5000
C 432	36	236	83.2	88.7	94.6	95.9	97.3	98.8	99	99.23
C3540	50	250	61.8	72.5	74.3	75.08	75.05	76.5	77.9	83.4
C6288	32	232	99.3	99.5	99.56	99.56	99.56	99.56	99.56	99.56

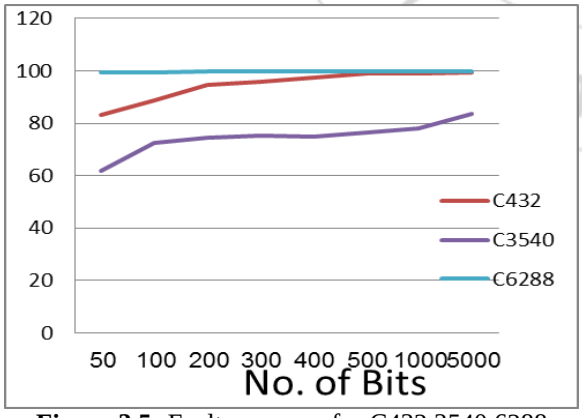

Figure 3.5: Fault coverage for C432,3540,6288

3. Conclusion

The expected result of this research is to develop a test pattern generation (TPG) algorithm using black box with some deterministic approach which could produce comparable fault coverage compared to some well known TPG method such as random and antirandom.

References

[1] Tanveer Ahmed, Liakot Ali , Evaluation of Fibonacci Test Pattern Generator for Cost Effective IC Testing, Journal of Computer Engineering and Informatics(JCEI), Vol. 2 Iss. 2, pp 28-36 ,Apr. 2014
[2] Elcheleburger E.B., and Lindoboloome,'Random-Pattern coverage enhancement and diagnostics for LSSD self-test", volume 27 Issue 3, pp 265-272 May 1983
[3] Shen HuiWu, Sridhar Jandhyala, Yashwant K. Malaiya, and Anura P. Jayasumana," Antirandom Testing A Distance-Based Approach". VLSI design , Yverdon, Switzerland, 2008.
[4] Malaiya, Y.K. ," Antirandom testing: getting the most out of black-box testing",Software Reliability Engineering, Sixth International Symposium on. ISSRE '95, pp 86–95. IEEE Computer Society 1995.
[5] Praveen Kumar Aggarwal, Vandana Yadav, Dr. Arti Noor, "DFT (Design for Testability) Pattern Generation Task for Circuit Under Test",International Journal of Engineering Research and Applications (IJERA) - Vol.

1, Issue 2, pp.190-193, volume 1- issue 2, July-Aug 2011
[6] Ireneusz Mrozek , Vyacheslav N. Yarmolik," Iterative Antirandom Testing",
[7] Journal of electronic Testing: theory and applications, Volume 28 Issue 3, June 2012
[8] DavidH. K. Hoe, JonathanM. Comer, JuanC. Cerda,Chris D.Martinez, andMukul V. Shirvaikar Cellular Automata-Based Parallel Random Number Generators Using FPGAs
[9] Ireneusz Mrozek , Yarmolik ,Antirandom Test Vectors for BIST in Hardware/Software Systems, Journal Fundamenta Informaticae archive , Volume 119 Issue 2, Pages 163-185 ,April 2012
[10] Hortensius, P.D. , Cellular automata circuits for built-in self-test, IBM Journal of Research and Development (Volume:34 , Issue: 2.3), Date of Current Version :06 April 2010
[11] Stavros Athanassopoulo , Christos Kaklamanis, Gerasimos Kalfoutzos, Evi Papaioannou, Cellular Automata: Simulations Using Matlab , The Sixth International Conference on Digital Society 2012,
[12] Chih-Ang Chen, Gupta, S.K., BIST test pattern generators for two-pattern testing-theory and design algorithms, Computers, IEEE Transactions on (Volume:45 , Issue: 3), 06 August 2002
[13] GaneshBabu.J 1, Radhika.P2 , Test pattern generation using pseudorandom BIST , International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering Vol. 2, Issue 5, May 2013
[14] Mayank Shakya and SoundraPandian. K. K, A Power Reduction Technique for Built-In-Self Testing Using Modified Linear Feedback Shift Register , World Academy of Science, Engineering and Technology , Vol:34 , 29-10-2009
[15] Touba, N.A. , Survey of Test Vector Compression Techniques, Design & Test of Computers, IEEE (Volume:23 , Issue: 4), August 2006
[16] Ilker Hamzaoglu, Janak H. Patel, Test set compaction algorithms for combinational circuits, IEEE/ACM