Journal of King Saud University – Computer and Information Sciences 31 (2019) 229–234

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

A heuristic fault based optimization approach to reduce test vectors count in VLSI testing

Vinod Kumar Khera *, R.K. Sharma, A.K. Gupta

Atmel R&D India Pvt Limited, #643, Regus Business Centre, Sector 16A, Noida, Uttar Pradesh, India

ARTICLE INFO

Article history:
Received 2 March 2016
Revised 12 January 2017
Accepted 5 February 2017
Available online 14 March 2017

Keywords:
VLSI testing
Essential fault based test vector optimization
Independent fault based test vector optimization
Test vector count

ABSTRACT

In this work we have proposed a heuristic approach to reduce the test vector count during VLSI testing of standard ISCAS circuits. With the shrinking die-space and increasing circuitry on a single Integrated circuit, the number of test vectors required for testing is also increasing. The number of test vectors directly affects the total testing cost of a circuit. In this work fault based test vector optimization has been proposed. Here, test vectors have been reduced by extracting child test vectors and merging them. The proposed scheme helps in reducing the test vector count and has been tested successfully using single stuck at fault models. The results obtained illustrate the effectiveness of proposed scheme.

1. Introduction

Imperfections and defects in the manufacturing process lead to faults and necessitate testing of the manufactured ICs. The shrinking die space, increasing complexity and enormous clock speeds of application circuits have left behind the Moore's law (International Technology Road Map for Semiconductors, 2013). This trend has led to the increased susceptibility of faults during manufacturing. The level of detection of faults (device, PCB, System, Field operation) plays an important role in deciding cost of testing. So it is extremely important to detect the fault at the device level i.e. at the manufacturing level, as the reliability of manufactured integrated circuit is the unconditional requirement (Hnatek, 1987). With the increasing integration, the test vector count, test power, test time and hence the total cost are growing rapidly (Report, 2001; Chandra, 2002; Athas, 1994; Abramovici, 1994). So, efficient testing schemes are essentially required for reducing test vector count and effective fault detection. This will ensure improved reliability of the manufactured circuits.

Modern automatic test pattern generation (ATPG) tools tend to reduce the test vector count and test power at the cost of increased intra-vector as well as inter-vector switching activities. Here, for single stuck at faults test vector generation, generally, random pattern generation scheme is followed by deterministic test vector generation scheme, as the fault coverage saturates due to presence of random pattern resistant faults in random pattern generation scheme. However these schemes results in increase test cost, due to increased switching. The fundamental objective of the testing is to detect the faults and filter out defective and substandard ICs within acceptable limits of test cost (number of test vectors, time and storage etc.), deterministic don't fulfill the criteria, as there is huge storage requirement. Also, random test vector generation schemes need long test vectors to detect all faults, resulting in increased test cost. This requires efficient scheme which generates optimum number of test vectors for detection of all the faults. Further, while using random test vector generation schemes, we can't take benefit of don't care values for reduction of test vectors. The Scan based reseeding techniques result in more compaction of test vectors, but at the cost of increased area and performance degradation additional multiplexers in the signal path (Abramovici, 1994). Circuits with DFT features have more number of concurrent active blocks in test mode than those during the normal mode operation resulting in increased test power and test cost as well. Test cost reduction can also be realized by using reduced (optimized) set

* Corresponding author.
E-mail address: vinod_kumar.khera@atmel.com (V.K. Khera).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2017.02.001
1319-1578/

of test vectors (Lin, 1995). Researchers have proposed techniques for reducing the test vector count based on static and dynamic compaction schemes. Static compaction is proposed by M. Abramovici et al. in Abramovici (1994) where test vectors are compacted after generation. Dynamic compaction proposed by Goel et al. in Rosales (1979) generates vectors by constraints of previously generated vectors

Schulz (1988) proposed Reverse-order fault (ROF) simulation technique which fails to eliminate redundant test cases. Fault based test vector reduction is also proposed by identifying essential and independent faults. Two faults are said to be Independent if they are not detected by a single vector (Akers, 1987; Pomeranz, 1993). A fault is said to be essential fault if it is detected by only single test vector in test set. Independent fault based and essential fault based techniques have been proposed by Akers (Akers, 1987), Kundu et al in Kundu et al. (2008), Krishna Kumar et al. (2012) and El-Maleh (2007). Compaction techniques based on independent faults have been proposed in Akers (1987), Krishnamurthy (1989). Scan based reseeding techniques results in system performance degradation and increased area due to the additional multiplexers in the signal path (Abramovici, 1994).

These limitations necessitate a technique that generates reduced number of test vectors and reduces inter-vector and intra-vector switching as well. Rest of work is explained in detail in below sections. Section 2 explains in details the proposed scheme 2.1describes the proposed algorithm and Fault based optimization schemes i.e. IFBO AND EFBO are explained in Sections 2.2 and 2.3; Section 3 describes essential test vector identification and, Redundant test vector removal is explained in Section 4. Section 5 enumerates the experimental results.

2. The proposed scheme

Test Vector count is the main component in deciding test cost of a circuit, this leads to the motivation to optimize test vectors count considering both independent and essential faults. The proposed algorithm is as shown below; optimizes test vector count by consideration the combined set of essential and independent faults. The sequence of steps contained in the proposed scheme is shown in Fig. 1.

2.1. Test vector reduction algorithm

- Generation of test vectors
- True Value Simulation (Ti)
- Find Test Vector Detection Count (TVDC) for all faults
- Arrange the faults in ascending order of TVDC
- Find the essential faults
  - o Essential Fault based test vector optimization EFBO() //for reducing intra vector switching & Test Vector
- Independent Fault based test vector optimization: IFBO()
- Other Fault/ Remaining test vector optimization
- Merge test vectors
  - o Substitute don't care bits (X) with suitable bit (0/1) using Genetic Algorithm and Hamming distance//for reducing inter-vector switching we have used GA and Hamming distance based technique so that majority bits in successive test vectors are similar
- Removal of redundant test vectors

*Intra vector switching refers to charging ($0 \rightarrow 1$)/discharging ($1 \rightarrow 0$)/of parasitic capacitance.

*Inter vector switching refers to switching in successive test vectors leading to leakage power.

2.2. Essential fault based test vector optimization (EFBO)

Essential fault of a test vector is a fault that is detected only by a unique test vector in the test set (Krishna Kumar et al., 2012). During EFBO phase, we optimize test vector count based on essential faults. This optimized set of test vectors leads to reduced intra-vector switching and curtailing of test power in circuit under test. Essential test vector detect at least one unique fault which is not detected by other test vectors in the test set. To achieve this, CUD is fault simulated with fully specified set of test vectors generated by ATPG tool. The test vectors are sorted as per their fault detection count (TVDC). Faults with TVDC = 1 are called essential faults. We have used ATLANTA tool for test vector generation and fault simulation. Next step is to find out essential child test vectors, out of the vectors detecting essential faults. Essential child test vectors are the vectors with optimum number of 0 s/1 s and rest are Xs. Essential child vectors are generated by relaxing parent test vector. For the identification of maximum don't care bits and hence identify the essential child vectors we have used the technique as in Kajihara (2004). It spots the primary outputs where there are utmost faults. Next we locate the path to this primary output from this fault by back traversing and find the values on the lines in between (FAN IN lines). So an essential child test vector may contain a combination of 1 s, 0 s and don't care(X) bits. Fig. 2 shows the circuit, essential vectors (T_i) and derived essential child vectors (T_{ci}) (where $1 \leq i \leq 4$) respectively. It detects all the faults as detected by parent essential test vector, but with optimum number of bits. Let the original set of test vectors generated by an ATPG tool be Ti and derived set of essential child test vectors with reduced number of bits be Tc, with a constraint of achieving the same fault coverage. The test vector set as generated by ATPG is fault simulated to detect number of faults detected by each test vectors (TVDC). The detected faults are then sorted in ascending order of their corresponding TVDC count. As discussed earlier that faults with TVDC = 1 are essential faults, next step is to find the essential child vector out of these vectors with TVDC.

There may be identical test child vectors detecting different faults, such test vectors are compatible. Two test vectors Ti and Tj are called compatible test vectors if only if none of the columns contain opposite i.e. ($T_i = T_j$) where i, j are same column bits of Ti and Tj and ($1 \leq i, j \leq n$). We have used the test vector compaction scheme as specified in Navabi (2011) for finding the compatible test vectors. The process of compatible test vectors detection is explained in Fig. 3; we have elaborated the process of finding compatible vectors in Fig. 3(a) and (b) respectively.

The objective of any test generation scheme is to reduce the test vectors and achieve the similar fault coverage, as that of prior to the reduction of test vectors, so after the detection of compatible test vectors, these are merged to reduce the test vector count. The Identified compatible test vectors Ti and Tj can be merged together to a single test vector Tij. The merged test vector Tij now detects faults detected by both Ti and Tj as shown in Fig. 4. This process results in reduced test patterns for a CUD (we have written a fault simulator in C language is written and used for these steps). Remaining bits other than those of child vectors are the don't care (X) bits.

During the whole process of optimization original fault coverage is preserved, and is explained by Table 1 which shows 4 test vectors, their detected faults and the essential child test vectors. Table 1 shows the essential Child test vector extracted from parent vector. The process of finding essential child test vectors is as explained in Section 3.

For clustering of clustered compatible essential test child vectors we have used the fault detection count based process (El-Maleh, 2007), as shown in Fig. 2. The process starts with Fault- simulation of the test vector set as obtained without fault

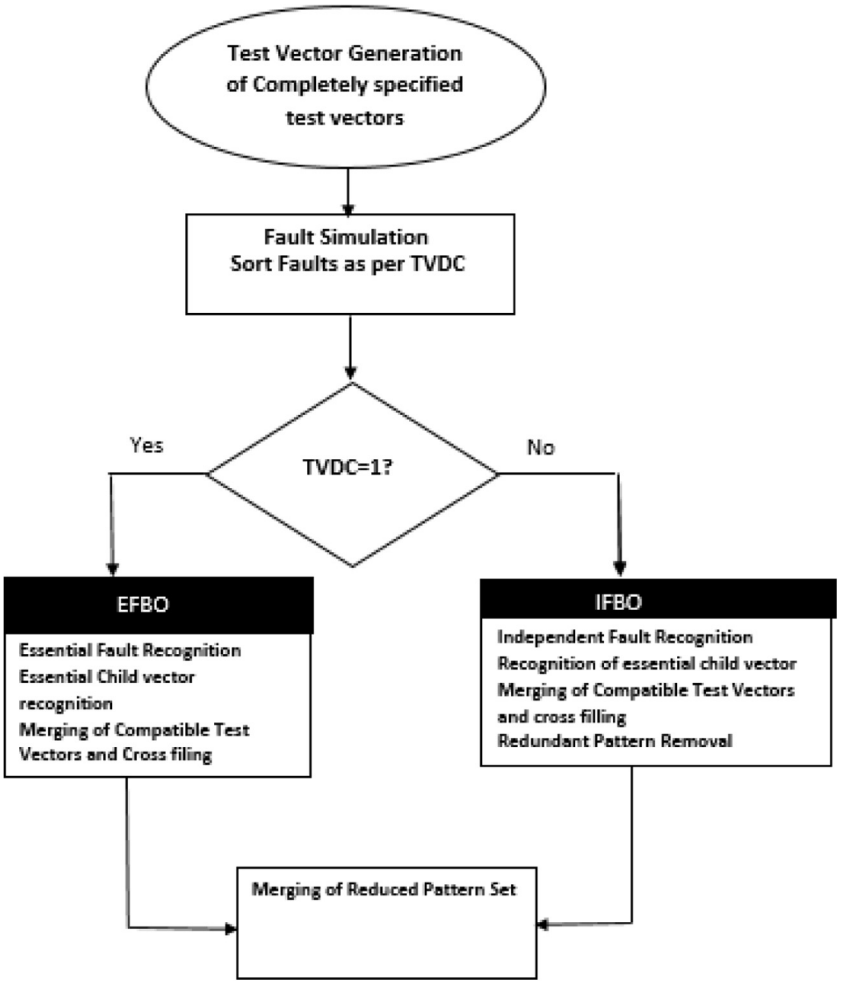
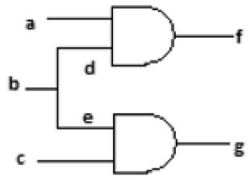


Fig. 1. Test vector reduction scheme.



2(a)

Ti	a	b	c
T1	1	1	1
T2	0	1	0
T3	0	0	0
T4	1	0	1

2 (b)

Ti	a	b	c
T1	X	1	1
T2	0	1	0
T3	0	0	X
T4	1	0	1

2 (c)

Fig. 2. Essential child vector extraction: (a) 3 Input, 2 output circuit; (b) Essential vectors; (c) Essential child vectors.

Ti/Ti+1	0	1	X
0	0	-	0
1	-	1	1
X	0	1	X

Test Vector Ti	Test Vector Ti+1	Compatibility Check
01101X1X	0110X011	Compatible
110X0X10	110X1X10	Incompatible

Fig. 3. Compatibility check: (a) Compatibility rules (b) compatible vectors.

dropping and then all test vectors are arranged with TVDC count, to find essential faults (TVDC = 1). Next clustering is done based on TVDC count. Table 2 shows the clustering Process, column 1 shows essential faults (F1 and F2), here clustering is performed based on TVDC = 1. Next, clustering based on TVDC = 2, is done and the faults F1, F3&F4 and F2, F5 and F8 are clustered bases on compatibility rules in Fig. 3. After this Clustering based on TVDC = 3 is done and we have clustered F1, F3, F4 and F2, F5, F6, F8.

F7 cannot be clustered in existing ones, so a separate bin is made and it was merged with existing fault list.

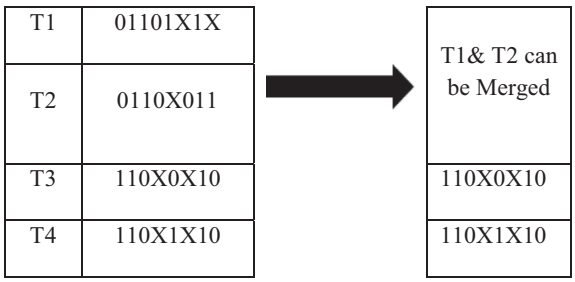


Fig. 4. Compatible test vectors merging.

Table 1
Test vectors, faults and essential child vectors.

Test vector	Faults detected	Essential child test vector
0001111110	F1	0X0XX11XX0
	F4	X0X1111XXX
1011101001	F2	X0X1X01XX1
	F5	XXX1X01001
	F6	1X1XXX10XX
	F8	XX1XXX1XX1
1010111110	F3	X0XXX1X10X
	F5	XX1XX11XX0
	F6	X0X11XX10X
	F7	XX11XX110X
0011110101	F3	0XX1X1XXX1
	F4	00XXX1X1X1
	F6	00XX11XX0X
	F7	X11X1X0X0X
	F8	XXX11X0101

2.2.1. Essential fault based test vector optimization (F, T)

- for every essential fault of Ti
- justify the Primary inputs (PI) to gate and assign true value to Primary output (PO)of this path
- Find and justify the cone for the line of this fault
- Merge PI values fixed at steps a and b
- For every essential child vector T_c (map to compatibility set)
- If there exists no sets; create a new set and map Tc Else-if map the Tc to existing compatible set
- Else create new set and map the Tc to new set // this takes care if no compatible set exists
- Go back to initial Step // repeat until all child vectors are mapped
- Simulate Faults using the set produced by merging compatible sets, drop all those detected and store test patterns in an array

2.3. Independent fault based test vectors optimization (IFBO)

Two faults are said to be independent if they are not detected by a single test vector (Akers, 1987; Pomeranz, 1993). Compatible fault test based compaction is proposed in Akers (1987), Pomeranz (1993), Kundu et al. (2008) and El-Maleh, 2007. In (Akers, 1987) ordering of target fault set using compatible fault set is proposed, whereas (Pomeranz, 1993) focuses on effectiveness of COMPACTEST and shows multiple times improvements. Independent fault sets have been used effectively in Akers (1987) to derive larger independent fault sets. However this technique is suitable for smaller circuits only. Further clustering is proposed using independent faults by El-Maleh in El-Maleh et al. (2002) for reduction of test patterns. Clique utilization for fault clustering is described in Krishna Kumar et al. (2012) to determine the maximum independent fault set. However (Krishna Kumar et al., 2012) is suitable for small network only due to dependency issues. In the proposed technique all faults with TVDC other than 1 are considered for designing Independent graph and independent faults are found as per rules specified in Dsha (1993). Authors have modified the algorithm proposed by El Maleh in El-Maleh (2007) by (i) using genetic algorithm for filling of don't care bits in place of random technique (ii) Don't care detection using extended implication procedure and (iii) recognition of redundant test vectors . Further (El-Maleh, 2007) fails to recognize redundant test vectors. The proposed algorithm has capability to optimize both Independent fault based test vectors as well as compatible fault based test vectors and is with added advantage of removal of redundant test vectors. The algorithm for independent fault based test vector optimization is shown below.

2.3.1. IFBO ( )

- For every independent fault(s), f that is detected by a set of test vector T
- For every test vector t (where t is a member of T):
  - Find the child vector tc from t (as done earlier for essential vectors)
    - For every child vector T_c (map to compatibility set)
    - If there exists no sets; create a new set and map Tc
  - Else-if map the Tc to existing compatible set
  - Else create new set and map the Tc to new set // this takes care if no compatible set exists
- Go back to initial Step // repeat until all child vectors are mapped
- Simulate Faults using the set produced by merging compatible sets, drop all those detected and store test patterns in an array
- Filling the don't care bits
- Remove the redundant test vectors.

2.3.2. Filling Of X-bits

Don't care bits are filled to reduce the dynamic power in the test patterns generated by ATPG. There have been many algorithms proposed in literature to fill don't care bits. In the dynamic test compaction schemes, the unspecified bits in partially specified vector are filled instantly subsequent to their generation with random values, which may result in increased number of transitions

Table 2
Clustering based on TVDC.

Clustering (TVDC = 1)		Clustering (TVDC = 2)		Clustering (TVDC = 3)	
Fault	Child Vector	Fault	Child Vector	Fault	Child Vector
F1	0X0XX11XX0	F1	0X0XX11XX0	F1	0X0XX11XX0
		F3	X0XXX1X10X	F3	X0XXX1X10X
		F4	X0X1111XXX	F4	X0X1111XXX
F2	X0X1X01XX1	F2	X0X1X01XX1	F2	X0X1X01XX1
		F5	XXX1X01001	F5	XXX1X01001
		F8	X0X110X00X	F6	1X1XXX10XX
				F8	X0X110X00X

Table 3
Comparison of test vectors obtained by proposed scheme vs ATALANTA.

Circuit	Test vectors using ATALANTA	Test vectors using proposed scheme	(%) comparitive reduction
C2670	113	96	15.04
C5315	193	61	68.4
c3540	95	63	33.69
s1238	158	117	25.95
s5378f	256	89	65.3
s9234	366	227	37.98
s13207	633	202	68.08
s38584	667	253	62.06
s38417	900	208	76.8
S15850.1	657	99	84.93
s9234	620	195	68.54

(Kundu et al., 2009). The proposed scheme takes care of such transitions and the essential child vectors are processed using genetic algorithm and hamming distance based scheme. In this step don't care (X) bits obtained by merging test vectors of proposed algorithm are replaced with appropriate bit values (1s/0s). As filling of don't care bit always lead to reduction in switching activity, so by using Genetic Algorithm followed by Hamming distance produces the best results (Kundu et al., 2009).

3. Essential child vector extraction

As explained earlier, essential child test vectors are extracted from the parent test vector by relaxing it, with a constraint that, essential child test vector should detect all the faults as detected by parent test vector. Test Relaxation is done so as to obtain partially specified test set (essential child test vectors) that maintains the same fault coverage while maximizing the number of unspecified bits out of fully specified test set. One of easiest method of relaxing the test vectors is to use brute-force technique, where each and every bit is altered to don't care (X) and fault coverage is checked. But the problem here is as the size increase the complexity is O (nm), where n is number of test vectors and m is length of each test vector. Here we have used the extended implication procedure (Kajihara, 2004), so that we may detect maximum don't care bits among a test vector. The brief algorithm is as explained below:

- For every essential fault (TVDC = 1)
- Inject the essential fault;
- Find primary output (PO) to propagate fault
- Find true values for this
- Justify the gate inputs identified above
- Find cone of dependency of injected fault;
- Justify cone of dependency of faulty line;
- Intersect PI values detected by justification of gate inputs and dependency cone.

4. Removal of redundant test vectors

After filling of don't care bits obtained from compatible essential child vectors recognized. We recognize the redundant test vec-

Table 4
% Test Vector reduction over existing techniques.

Circuit	PI	Atalanta (Dsha (1993))	ROF Schulz (1988)	RM El-Maleh et al. (2002)	FCC El-Maleh (2007)	IFC Akers (1987)	Proposed	Proposed over Atalanta	Proposed Over ROF	Proposed Over RM	Proposed over FCC	Proposed over IFC
c2670	233	113	106	100	98	98	96	15.04425	9.433962	4	2.040816	2.040816
C5315	178	193	119	106	80	107	61	68.39378	48.7395	42.45283	23.75	42.99065
c3540	50	95	83	80	75	99	63	33.68421	24.09639	21.25	16	36.36364
s1238	32	158	124	119	116	127	117	25.94937	5.645161	1.680672	−0.86207	7.874016
s15850.1	611	657	456	181	144	142	99	84.93151	78.28947	45.30387	31.25	30.28169
s5378f	214	256	252	145	119	120	89	65.23438	64.68254	38.62069	25.21008	25.83333
s9234	247	620	375	202	198	217	195	68.54839	48	3.465347	1.515152	10.13825
s35932	35	64	63	57	39	47	31	51.5625	50.79365	17.54386	20.51282	34.04255
b17	38	1053	972	819	789	633	599	43.1149	38.3744	22.7106	24.0811	5.37124
s13207	700	633	476	252	238	244	202	68.0884	57.5630	19.8412	15.1260	17.2131

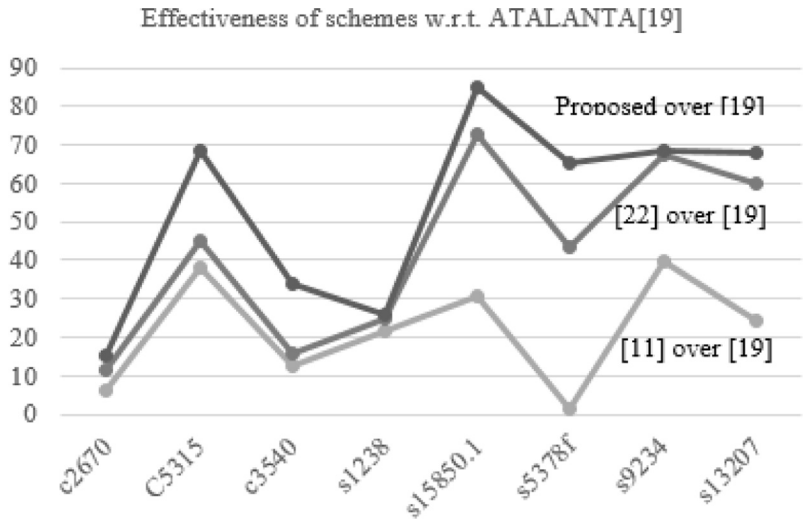


Fig. 5. Effectiveness of proposed scheme and other existing techniques when compared to ATALANTA (Dsha, 1993).

tors. As reduction of test vector count is essential to reduce the test power and hence cost. The test vectors thus obtained after cross filling are fault simulated, all test vectors are arranged in decreasing order of fault recognized. We have used two variables *one_check* and *check* for recognizing the essential and minimal set of test vectors as in Krishna Kumar et al. (2012).

- Calculate one_check for each test pattern
- Patterns with non-zero one_check are copied to the final set Step
- All faults detected by these patterns are dropped from the fault list and also from the fault list of remaining vectors
- While (there are still remaining faults)
  - o Find a pattern that detects the largest number of uncovered faults
  - o Add it to the final set and drop this fault as in Step 3.

5. Results and observations

Algorithm proposed in this work has been successfully applied on fully specified test pattern sets generated by Atalanta (Dsha, 1993). Standard ISCAS circuits have been tested using Atalanta and the scheme proposed in this work. Fully specified test pattern sets for single stuck at fault model are generated using ATPG tool Atalanta (Dsha, 1993); Table 3 shows performance comparison between the proposed scheme and ATPG tool Atalanta (Dsha, 1993). Column 2 and column 3 show test patterns generated by Atalanta with _x option and test patterns generated by proposed method respectively. The reduction (ranging from 15% to 80 %) is shown in column 3 of the Table 2.

Table 4 shows effectiveness in terms of %age of saving done by using the proposed scheme when compared to standard preexisting techniques, here we have compared with the preexisting techniques (Schulz, 1988; Akers, 1987; El-Maleh, 2007; Dsha, 1993; El-Maleh et al., 2002), number of PI in the circuit are shown in column 2, while the column 3–7 show the test patterns generated by preexisting techniques. Column 8 shows the test patterns generated by proposed scheme while in the columns 9–13 we have compared the test pattern %age saving when compared with Atalanta (Dsha, 1993). It has been proved that the proposed scheme has ability to reduce the test pattern more than 80%.

Fig. 5 effectively demonstrates the authenticity of the proposed scheme, it shows the test vector reduction ability of the scheme when all existing schemes are compared to test vectors generated by ATALANTA (Dsha, 1993).

6. Conclusion

We have proposed an effective scheme for reduction of test vector based on fault based test vector optimization scheme. The proposed scheme is based on test vector compaction based on Essential faults based test vector (EFBO) and Independent faults based test vector optimization (IFBO) schemes. The essential child vectors obtained are merged based on compatibility check, resulting in increased number of faults being detected by reduced number of test vectors (compacted set). Experimental results for benchmark ISCAS circuits show that the proposed method achieves considerable reduction in test vector count when compared to preexisting techniques. However the proposed scheme can be improved by combining the proposed scheme with some dynamic compaction scheme, which is our future work.

References

Abramovici, M., 1994. Digital System Testing and Testable Design. IEEE Press, Piscataway- New Jersey.

Akers, S.B., 1987. On the Role of Independent Fault Sets in the Generation of Minimal Test Sets. Inte. Test Conf.

Athas, 1994. Low power digital systems based on adiabatic-switching principles. IEEE Trans. Very Large Scale Integr. VLSI Syst. 2 (4), 398–416.

Chandra, Reduction of SOC Test Data Volume, Scan Power and testing time using Alternate run-length Codes In: ACM/IEEE Design Automation Conference (DAC), 2002.

Dsha, H., 1993. Atalanta: An Efficient ATPG for Combinational Circuits. Virginia Poly. Institute and State University, Virginia.

El-Maleh, A., 2007. Test Vector Decomposition based static compaction based static compaction algorithms for combinational circuits. IET Comput. Digit. 1 (4), 364–368.

El-Maleh, A.A.-S. 2002. An efficient test relaxation technique for combinational and full-scan sequential circuits. In: Proc. of VLSI Test Symposium, Monterey, CA, pp. 53–59.

Hnatek, E.R. 1987. IC Quality – Where Are We. In: Proceedings of the IEEE International Test Conference, pp. 430–445.

International Technology Road Map for Semiconductors, 2013.

S. Kajihara, 2004. XID: Don't care Identification for Test Patterns for combinational Circuits. In: IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems, vol. 23, no. 2.

Krishna Kumar, S., Kundu, S., Chattopadhyay, S., 2012. Customizing completely specified pattern set targeting dynamic and leakage power reduction during testing. Integration VLSI J. 45 (2), 211–221.

Krishnamurthy, B., 1989. Test counting: a tool for VLSI Testing. IEEE Des. Test Comput. 6 (5), 58–77.

Kundu, S., et al., A Novel, Technique to reduce both leakage and peak power during scan testing. In: IEEE Region 10 and Thirs International Conference on Industrial and Information Systems, 2008.

Kundu, S., Chattopadhyay, S., 2009. Efficient don't care filling for power reduction during testing. In: 2009 International Conference on Advances in Recent Technologies in Communication and Computing, Kottayam, Kerala, pp. 319–323.

Lin, C.S., 1995. Test compaction for combinational circuits. IEEE Trans. CAD 14 (11), 1370–1378.

Navabi, Z., 2011. Digital System Test and Testable Design using HDL Models and Architectures. Springer, London.

Pomeranz, 1993. COMPACTEST: A Method to Generate Compact Test Sets for Combinational Circuits. IEEE Trans. CAD 12 (7), 1040–1049.

Report, International Sematech, The International Technology Roadmap for semiconductors (ITRS), 2001.

Rosales, B.C., Test Generation and Dynamic Compaction of Tests. In: International Test Conference, 1979.

Schulz, M.H., 1988. Socrates: A highly Efficient Automatic Test Pattern Generation Scheme. IEEE Trans. Compute. Aided Design Integ. Circ. Syst. 7 (1), 126–137.