On Reducing Peak Current and Power During Test

Wei Li [1] Sudhakar M. Reddy [1]
Dept. of ECE
Univ. of Iowa
Iowa City, IA 52242
{li100, reddy}@engineering.uiowa.edu

Irith Pomeranz [2]
School of ECE
Purdue University
West Lafayette, IN 47907
pomeranz@ecn.purdue.edu

Abstract

This paper presents a Progressive Match Filling (PMF) technique to reduce the peak current and power dissipation during the fast capture cycle in broadside delay fault testing. The proposed method fills the unspecified values (X) in the generated initialization vector such that the resulting launch vector at a minimal Hamming distance from the initialization vector. The proposed method does not require any hardware modification and can be used to obtain any test sets that require two pattern tests. Experimental results show that the proposed method reduces the peak current and power dissipation during the fast capture cycle by 40.59% on average and up to 54.17% for large ISCAS 89 circuits.

1. Introduction

Physical defects that increase the signal propagation delays in circuits can be modeled by delay faults. Gate delay fault model [1] and path delay fault model [2] have been used to model delay defects. The most commonly used gate delay fault model is the transition fault model [1]. In transition fault model, there are two types of transition faults: slow-to-rise (STR) fault and slow-to-fall (STF) fault. A test vector pair <V_I, V_L> is required to detect a transition fault. Two conditions have to be satisfied by <V_I, V_L> to detect a STR (STF) fault. The initialization vector V_I must set the faulty line to 0 (1) to initialize the STR (STF) fault. The launch vector V_L has to launch a rising (falling) transition for the STR (STF) fault and also propagate the fault effect to one or more outputs

Several approaches have been developed to apply two pattern tests to the circuit under test (CUT) via scan chains. Enhanced scan is a scan-based technique that can apply all possible $2^n(2^n-1)$ pairs of test patterns to the CUT, where *n* is the number of circuit inputs from the scan chain. However since enhanced scan uses scan flip-flops that can hold two bits, it requires significant hardware overhead. The broadside approach and the skewed-load approach are used to apply two pattern tests for standard scan designs. In both the approaches, the initialization vector of a two pattern test is shifted into the scan chain in the same manner as a test pattern for a stuck-at fault. In the broadside test approach, the launch vector of a two pattern test is derived from the response of the CUT to the initialization vector. In the skewed-load approach, the launch vector is derived by shifting the initialization vector in the scan chain by one bit.

Broadside test approach is often preferred although its coverage is typically lower than the one achieved with skewed-load approach. One reason for this is that broadside approach does not require a fast scan enable signal and therefore results in less hardware overhead and simpler timing closure.

It is known that power dissipation may be considerably higher in test mode than in the normal operation. While scanning in test vectors as well as during capture cycles, many more flip-flops than during normal circuit operation may change value in each clock cycle, thus causing much higher average and peak power dissipation than in the normal operation mode. This can cause problems both with heat dissipation and with current spikes. Another problem arises when wafer probe testing is done through spring loaded contacts which may not be able to deliver large enough current to meet the peak current requirement of some scan tests and supply voltage drops are caused. This may cause a chip to fail a test even when it is defect-free, thus reducing the yield. The peak current problem is especially important during the fast capture cycle of broadside delay fault testing. Yield loss due to voltage drop caused by higher peak current during test of an ASIC chip has been reported in [3]. Using tests with reduced peak current requirement will help reduce the possibility of such occurrences.

While many techniques have been proposed to reduce power dissipation during scan shifting [4-8], only a few works [9-12] have been reported to reduce the peak current and power dissipation during the capture cycles. All of these techniques [9-12] for reducing the peak current during capture cycles use additional DFT hardware.

Unlike all the previous methods that require additional DFT hardware, we present a Progressive Match Filling (PMF) technique that does not require any additional DFT hardware to reduce the peak current and power dissipation during the fast capture

Proceedings of the IEEE Computer Society Annual Symposium on VLSI
New Frontiers in VLSI Design
0-7695-2365-X/05 $20.00 © 2005 IEEE

cycle in broadside delay fault testing. The proposed method can also be used together with the techniques in [9-12] to further reduce peak current and power during test. Unlike the traditional random filling of the unspecified values (X), the proposed method fills the unspecified values in the generated initialization vector V_I such that the corresponding launch vector V_L has minimal Hamming distance with the initialization vector V_I. The proposed method can be embedded into normal ATPG flow.

The remainder of the paper is organized as follows. Section 2 gives the background on broadside delay testing. In Section 3 we formulate the problem investigated. Section 4 introduces overall flow of the test generation scheme. In Section 5, we describe the proposed progressive match filling. In Section 6 experimental results for large ISCAS89 benchmark circuits are presented. Section 7 concludes the paper.

2. Preliminaries

A CMOS digital circuit dissipates power statically and dynamically. Static power dissipation is mainly due to leakage current. Dynamic power dissipation, on the other hand, occurs when a logic value of a gate changes. Since dynamic power dissipation is much larger than static power [13], it is important to reduce dynamic power dissipation, which is approximated as follows:

$$\sum_j \frac{1}{2} C_L(j) \times V_{dd} \times f$$

where $C_L(j)$ is the load capacitance of the node j and f is the switching frequency of the circuit. Since $C_L(j)$ is proportional to the number of fan-out gates, the power dissipation of node j is dependent on the fan-out number and the switching frequency given a constant V_{dd}.

Large computation time is required to calculate the actual power dissipation. To simplify this, *Weighted Switching Activity* (WSA) defined below is used to represent the power dissipated and current in a circuit when a two-pattern test is applied. WSA was also used to represent instantaneous power in earlier works [5, 12].

Definition 1: The *weighted switching activity* (WSA) of a node is the number of state changes at the node multiplied by (1+node fan-out). The "1" in the formula is to include the output capacitance of the driving gate. The WSA of the entire circuit is obtained by summing the WSA of all the nodes in the circuit.

Definition 2: The *peak power* is the highest value of power dissipation at any given instant of switching activity caused by test application.

Since the peak current is proportional to the peak power dissipation, we use the reduction of peak WSA to represent the reduction of peak current as well in this work.

2.1 Test Application

In broadside test, the launch vector V_L is obtained by capturing the response of the CUT to the initialization vector V_I. Therefore in broadside approach, there are two capture cycles. During the first capture cycle, the scan flip-flops capture the response of the CUT to the initialization vector V_I obtaining the launch vector V_L. During the second capture cycle, the scan flip-flops capture the response of the CUT to the launch vector V_L. The second capture has to run at full system speed to fulfill the requirement of at-speed delay fault testing while the first capture can run at a lower frequency. The first capture is called slow capture and the second capture is called fast capture in [3]. A typical broadside test waveform is given in Figure 1.

The scan enable signal only needs to switch from scan-in mode to functional mode before the slow capture, which implies the scan enable signal need not switch at full system speed. Several dead cycles may be added between the scan-in process and the slow capture cycle to further ensure the safe captures. The peak current requirements during the second capture cycle could cause voltage drops that slow down the circuit and hence good chips may fail at-speed tests [3]. In this work we consider reducing the switching activity during the fast capture cycle thus reducing the peak current and power requirements during this cycle.

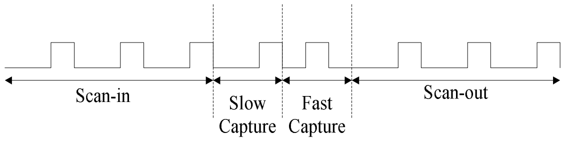

Figure 1. Broadside Waveform

2.2 Earlier Work

As mentioned earlier most of the earlier works which considered reduction of peak current during the fast capture cycle used additional DFT logic [9-12]. A recent work [14] considered a method to reduce average switching activity during the fast capture cycle by filling the unspecified values in test cubes in specific manner. Among the several methods to fill unspecified values it was experimentally found that a method called *repeat fill* reduced the average switching activity during the fast capture cycle. In the repeat fill method the unspecified values (Xs) in the test cube are filled by repeating the binary value found before the start of a run of Xs. For example, consider a test cube 0XXX1XX0XX0XX. The pattern resulting from repeated fill will be 0000111000000. However this method may not insure reduction of peak switching activity. We will present experimental results to show that repeat fill may not reduce the peak switching activity compared to random fill in some designs. Even when it reduces the peak switching activity the method of filling Xs proposed in this work is shown to achieve higher reduction of peak switching activity.

3. Problem Formulation

In this paper, we assume that the sequential circuit under test (CUT), which has *m* primary inputs, p_1, p_2, …, p_m, and *n* state inputs, s_1, s_2, …, s_n, employs full scan and scan input s_i, *i*=1, 2, …, *n*, is driven by a corresponding scan flip-flop D_i during test application. Unlike state inputs driven by a scan chain, we assume that any pattern pairs can be applied to primary inputs in two consecutive cycles.

In broadside delay fault testing, we assume that the initialization vector is given by $V_I = < p_{1I}, p_{2I}, …, p_{mI}, s_{1I}, s_{2I}, …, s_{nI}>$ and the launch vector is given by $V_L = < p_{1L}, p_{2L}, …, p_{mL}, s_{1L}, s_{2L}, …, s_{nL}>$, where $< s_{1L}, s_{2L}, …, s_{nL}>$ is the state output of the CUT in response to the launch vector V_I.

The peak current and power dissipation during the fast capture cycles of broadside delay fault testing can be represented by the weighted switching activity WSA(V_I, V_L). The value of WSA(V_I, V_L) is expected to be proportional to the Hamming distance between V_I and V_L. That is, smaller the Hamming distance between V_I and V_L smaller is the WSA caused during the fast capture cycle. Based on this observation, in the proposed method, we generate the initialization vector V_I such that the Hamming distance between V_I and the resulting launch vector V_L is minimized.

4. Test Generation Flow

Figure 2 gives the overall flow of the proposed test generation method. After a test is generated for a target fault, the Progressive Match Filling (PMF), which will be discussed in next section, is executed to reduce the Hamming distance between the initialization vector V_I and launch vector V_L. Then, a test vector derived after PMF is applied is simulated and all the detected faults are dropped. This process is repeated until all the faults are processed.

5. Progressive Match Filling

As introduced in Section 3, after a test pair is generated for a target transition fault, we fill the unspecified values (X) in the initialization vector V_I to minimize the Hamming distance between V_I and the launch vector V_L derived from V_I. We use Progressive Match Fill (PMF) to achieve this objective.

We represent V_I as $<P_I, S_I>$ where P_I is the set of primary inputs and S_I is the set of state inputs. Similarly, the launch vector V_L is also represented as $<P_L, S_L>$. The steps of PMF are given next.

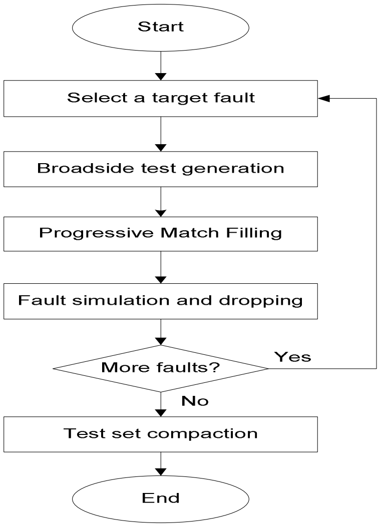
Start
Select a target fault
Broadside test generation
Progressive Match Filling
Fault simulation and dropping
More faults?
Yes
No
Test set compaction
End

Figure 2. Flow of Test Generation

Step 1: We assign the X values in P_I and P_L using the following rules such that the Hamming distance between P_I and P_L is minimized.

- If a primary input is assigned a specified value *a* = 0 or 1, in P_I (P_L) and there is an X value in the corresponding position in P_L (P_I), then we assign the value *a* to this primary input in P_L (P_I).
- If a primary input is unspecified in both P_I and P_L, then we assign the same randomly chosen binary value to this primary input in both P_I and P_L.

Step 2: We search for corresponding state inputs which have unspecified values in S_I and specified values in S_L. If S_L, the state part of the lunch vector, has a specified value *a* (*a* = 0 or 1) in the state variable s_{mL} and the corresponding state variable s_{mI} in the initialization vector S_I has an unspecified value X, then we fill the X in s_{mI} with *a*.

Step 3: We perform logic simulation using the updated initialization vector V_I. If we can find more newly specified values in the next state outputs which can be used to fill the unspecified values in V_I, then we go back to Step 2. Otherwise we go to the next step.

Step 4: If there are still X values in the initialization vector V_I, then we pick N Xs randomly where N is a user specified number and fill them with random binary numbers. Then we go back to Step 3 to do logic simulation with the updated V_I. This procedure continues until the initialization vector V_I is fully specified.

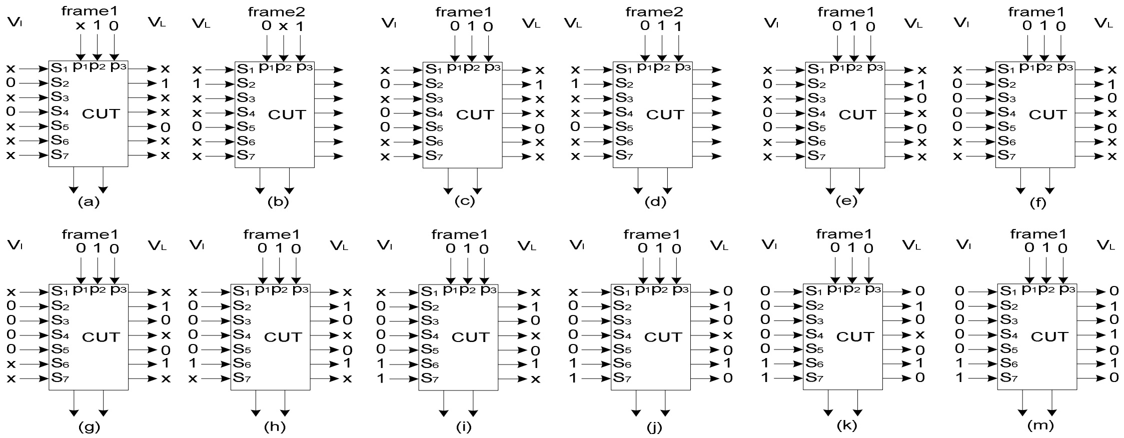

Figure 4. An Example of PMF

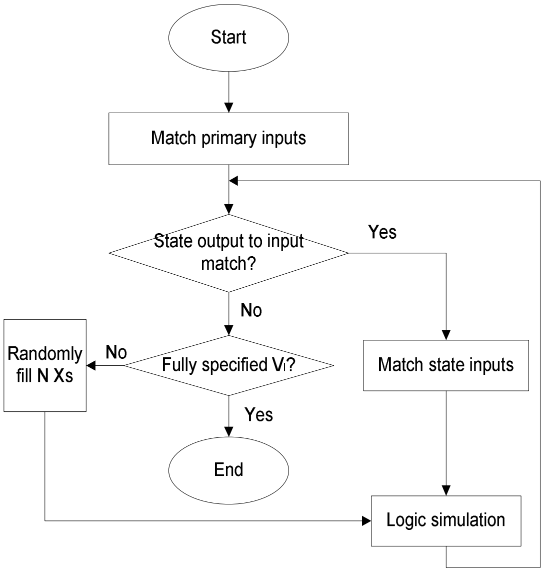

Figure 3. Flow of Progressive Match Filling

Figure 3 gives the flow of the proposed progressive match filling procedure. In order to reduce the run time, we use the following methods.

First, we use event driven logic simulation technique in progressive match filling. Every time after we update the initialization vector V_I, event driven logic simulation re-evaluates the output value of a gate only if at least one of its input changes value and triggers an event.

Second, for some large designs we can set the user-specified parameter N to a large number to fill more Xs at one time. The larger the value of N used, the faster the process converges. For example, after Step 3 we may fill all the remaining Xs in V_I randomly with binary values. This can be achieved by setting the value of N to be the same as the number of state variables (we use $N = \infty$ to represent this). In this case, Step 4 is executed only once. We call the Progressive Match Filling (PMF) with $N = \infty$ as *Non-Aggressive Progressive Match Filling* or *NAPMF*. Similarly, we call the Progressive Match Filling (PMF) with N = 1 as *Aggressive Progressive Match Filling* or *APMF*.

The following example illustrates how the progressive match filling works when N = 1.

Example 1: We use Figure 4 to illustrate PMF when N = 1 is used. Figure 4 depicts a CUT with 3 primary inputs and 7 state inputs. We represent an input vector as $<p_1\ p_2\ p_3\ s_1\ s_2\ s_3\ s_4\ s_5\ s_6\ s_7>$. Figures are labeled with time frames, with frame 1 corresponding to the first capture cycle and frame 2 corresponding to the second (fast) capture cycle of a broadside test. The state component of the initialization vector is shown to the left of the first time frame. The state component of the launch vector is shown at the state outputs of the first and the state inputs of the second time frame, respectively. Figures 4 (a) and (b) show the initialization vector V_I and launch vector V_L in two time frames for a generated test cube (that is the test generated for a target fault but not processed yet by PMF). In this test cube V_I is <X10X0X0XXX> and V_L is <0X1X1XX0XX>. In Step 1 of PMF, we match the primary inputs in V_I and V_L. This sets p_1 of V_I to 0 to match the binary value of p_1 in V_L and sets p_2 of V_L to 1 to match the binary of p_2 in V_I. Thus at the end of Step 1 V_I becomes <010X0X0XXX> and V_L becomes <011X1XX0XX>. During Step 2 we fill Xs in V_I to match specified values in V_L whenever possible. For this example, in Step 2 we fill the unspecified s_5 in V_I with 0 because V_L has a 0 in s_5. Updated V_I and V_L after Step 2 are shown in Figures 4(c) and 4(d). Next, in Step 3 we perform logic simulation with the updated V_I. It causes a newly specified 0 in s_3 of V_L. This is shown in Figure 4(e). This causes execution of Step 2 again and we assign 0 to s_3 in V_I, as shown in Figure 4(f). Next we execute Step 3 again and perform logic simulation with the updated V_I and find one more specified value of 1 for s_6 of V_L, as shown in Figure 4(g). This in turn causes executing Step 2 once more and assignment of 1 to s_6 in V_I as shown in Figure 4(h). Next we do logic simulation and can not find any newly specified value in V_L. Thus, we next execute Step 4 in which we randomly choose an X in V_I and assign a random binary value to it. In the case of the example as shown in Figure 4(i), we assign a 1 to s_7 in V_I. Next, as shown in Figure 4(j) we go back to logic simulation with the updated V_I and find one more specified value of 0 in V_L for s_1. Executing Step 2 again we assign 0 to s_1 in V_I as shown in Figure 4(k). At this point V_I is fully specified and simulating it gives a fully specified V_L also as shown in Figure 4(m).

It is important to note the following point. PMF uses only logic simulation. It attempts to fill the unspecified values in V_I to match the specified values in V_L or fills them randomly. One can determine unspecified values in V_I to match specified values in V_L by performing backward implications. However we chose not to do this as it requires longer run times.

6. Experimental Results

We implemented the proposed method in C language and conducted experiments on large ISCAS89 benchmark circuits. All experiments were conducted on a HP ZX2000 machine with 1GHZ Itanium CPU, 2GB memory and the HP Unix operating system.

Table 1 shows experimental results that are obtained for large ISCAS89 circuits with N = 1 in PMF. That is we used aggressive PMF. Under the heading "WSA_{peak}", we give the peak WSA values during the fast capture cycle achieved with random X-filling method and Aggressive PMF (APMF) method followed by percentage reduction in WSA achieved with the APMF method compared to random X-filling. Similarly, under the heading "WSA_{avg}", we compare the APMF method to random X-filling method in terms of the average of the capture cycle WSA. Finally we give the number of vectors for both random X-filling method and APMF method under the heading "No. of Vec". For a fair comparison, we conducted 10 groups of test generations with random X-filling using different random seeds and reported the average values in the results on random X-filling method.

From Table 1 it can be seen that for every circuit, the peak WSA during the fast capture cycle of broadside delay fault testing using PMF method is much lower than the one achieved using random X filling method. The peak WSA during the fast capture cycle is reduced by 40.59% on average and up to 54.17%. The average of WSA during fast capture cycle is also reduced with the APMF method. The average WSA during the fast capture cycle is reduced by 66.86% on average and up to 86.13%.Compared to the reduction of peak WSA, the reduction of average WSA during the fast capture cycle is even more significant. The numbers of generated test vectors with PMF are always higher than those with random X filling method. The increase of the pattern counts is due to the fact the proposed method minimizes the transitions in the CUT. Therefore, the faults other than the targeted transition fault, will have less chance to be detected, which in turn causes the increase of pattern counts. Since our major concern is the peak WSA and the proposed method reduces the average of WSA during the fast capture cycle even more than the peak WSA, it implies that the proposed method applies too tight a constraint for the generation of all tests. To reduce the pattern counts, one can use the minimized peak WSA as an upper bound to relax the PMF procedure to obtain smaller test sets.

The experimental results on large ISCAS89 circuits obtained with NAPMF method, where $N = \infty$, are given in Table 2. The data is arranged similar to Table 1.

From Table 2, it can be seen that the NAPMF method reduces the peak WSA during the fast capture cycle by 19.63% on average and up to 41.78%. The average of WSA during the fast capture cycle is reduced by 45.34% on average and up to 75.25%. Compared to the APMF method, NAPMF is simpler because it fills all the remaining Xs in V_I after Step 3 with random values, thus avoids executing Step 4 more than once. NAPMF achieves less WSA reduction compared to APMF. The pattern counts for NAPMF are lower than that for APMF, but higher than for the case of filling all Xs in V_I randomly. This shows that there is a tradeoff between WSA reduction, pattern counts and the time complexity of the procedures.

For comparison, in Table 3 we show the experimental results on ISCAS89 circuits obtained using the repeat X-filling method of [14]. The data in Table 3 is arranged similar to that in Tables 1 and 2. It can be seen that the repeat X-filling method reduces the peak WSA for some circuits such as s38417 and s38584 compared to the random X-filling method. However it does not reduce peak WSA for other circuits. Comparing the results given in Table 1, 2 and 3, we can conclude that the proposed PMF method reduces the peak WSA during the fast capture cycle of broadside delay fault testing more effectively than the repeat X-filling method and random X-filling method.

7. Conclusions

In this work, we proposed a progressive match filling technique to reduce the peak current and power dissipation during the fast capture cycle of broadside delay fault test. Unlike the conventional random X filling method where all the X values are filled with binary values randomly, the proposed method fills the X values in a way such that the initialization vector V_I has a minimal Hamming distance with the launch vector V_L. The peak current and power dissipation are reduced with the proposed method. Experimental results show that the proposed method reduces the peak WSA during the fast capture cycle by 40.59% on average and up to 54.17% on large ISCAS89 circuits.

References

[1] J. A. Waicukauski, E. Lindbloom, B. K. Rosen and V. S. Iyengar, "Transition Fault Simulation", IEEE Design & Test of Computers, Vol. 4, No. 2, April 1987.

[2] G. L. Smith, "Model for Delay Faults Based Upon Paths", Proc. ITC, pp. 342-349, 1985

[3] J. Saxena, et. al., "A Case Study of IR-Drop in Structured At-Speed Testing", Proc. ITC. pp.1098-1104, 2003

[4] V. Dabholkar, S. Chakravarty, I. Pomeranz and S. M. Reddy, "Techniques for Minimizing Power Dissipation in Scan and Combinational Circuits During Test Application", IEEE TCAD, pp. 1325-1333, 1998

[5] S. Gerstendorfer and H. J. Wunderlich, "Minimized Power Consumption for Scan-based BIST", Proc. ITC, pp. 77-84, 1999

[6] R. Sankaralingam, R. R. Oruganti and N. A. Touba, "Static Compaction Techniques to Control Scan Vector Power Dissipation", Proc. VTS, pp. 35-40, 2000

[7] S. Kajihara, K. Ishida and K. Miyase, "Test Vector Modification for Power Reduction during Scan Testing", Proc. VTS, pp. 160-165, 2002

[8] A. Chandra and K. Chakrabarty, "Reduction of SOC Test Data Volume, Scan Power and Testing Time Using Alternating Run-length Codes", Proc. DAC, pp 673-678, 2002

[9] P. Girard, L. Guiller, C. Landrault and S. Pravossoudovitch, "Circuit Partitioning for Low Power BIST Design with Minimized Peak Power Consumption", Proc. ATS, pp. 89-94, 1999

[10] P. M. Rosinger, B. M. Al-Hashimi and N. Nicolici, "Scan Architecture for Shift and Capture Cycle Power Reduction", Proc. DFT, pp. 129-137, 2002

[11] K. Lee, S. Hsu and C. Ho, "Test Power Reduction with Multiple Capture Orders", Proc. ATS, pp. 26-31, 2004

[12] W. Li, S. M. Reddy and I. Pomeranz, "On Test Generation for Transition Faults with Minimized Peak Power Dissipation", Proc. DAC, pp. 504-509, 2004

[13] H.Weste, K. Eshraghian, "Plinciples of CMOS VLSI Design: A System Perspective", 2nd ed.,MA:Addison-Wesley, 1992

[14] K. M. Butler, et. al., "Minimizing Power Consumption in Scan Testing: Pattern Generation and DFT Techniques", Proc. ITC, pp 355-363, 2004

Table 1 Experimental Results with Aggressive Progressive Match Filling method (N = 1)

	WSA_{peak}			WSA_{avg}			No. of Vec	
	rand	PMF	red %	rand	PMF	red %	rand	PMF
s5378	5584	2559	54.17%	4412	612	86.13%	393	961
s9234	8258	6033	26.94%	5794	2765	52.28%	614	943
s13207	9879	5456	44.77%	7502	2702	63.98%	766	1689
s15850	11097	7257	34.60%	7492	2686	64.15%	511	1166
s38417	30791	14853	51.76%	24287	6054	75.07%	1830	4344
s38584	39291	26990	31.31%	25497	10316	59.54%	1485	5245
avg	-	-	40.59%	-	-	66.86%	-	-
max	-	-	54.17%	-	-	86.13%	-	-

Table 2 Experimental Results with Non-aggressive Progressive Match Filling method (N = •)

	WSA_{peak}			WSA_{avg}			No. of Vec	
	rand	PMF	red %	rand	PMF	red %	rand	PMF
s5378	5584	3251	41.78%	4412	1092	75.25%	393	824
s9234	8258	7106	13.95%	5794	4151	28.36%	614	823
s13207	9879	8431	14.66%	7502	5310	29.22%	766	1020
s15850	11097	9032	18.61%	7492	4701	37.25%	511	681
s38417	30791	27533	10.58%	24287	13441	44.66%	1830	2410
s38584	39291	32142	18.20%	25497	10879	57.33%	1485	4057
avg	-	-	19.63%	-	-	45.34%	-	-
max	-	-	41.78%	-	-	75.25%	-	-

Table 3 Experimental Results with Repeat Filling Method

	WSA_{peak}			WSA_{avg}			No. of Vec	
	rand	repeat	red %	rand	repeat	red %	rand	repeat
s5378	5584	7744	-38.68%	4412	4564	-3.45%	393	617
s9234	8258	9756	-18.14%	5794	5951	-2.71%	614	853
s13207	9879	10727	-8.58%	7502	5670	24.42%	766	1006
s15850	11097	12797	-15.32%	7492	6399	14.59%	511	962
s38417	30791	25476	17.26%	24287	13465	44.56%	1830	2532
s38584	39291	38348	2.40%	25497	16366	35.81%	1485	4012
avg	-	-	-10.18%	-	-	18.87%	-	-
max	-	-	17.26%	-	-	44.56%	-	-