DDECS DESIGN & DIAGNOSTICS of ELECTRONIC CIRCUITS & SYSTEMS

8th IEEE Workshop on
Design and Diagnostics of Electronic Circuits and Systems
Sopron, Hungary, April 13 - 16, 2005

VECTOR REPETITION AND MODIFICATION FOR PEAK POWER REDUCTION IN VLSI TESTING

M. Bellos[1], D. Bakalis[1,2], D. Nikolos[1] and X. Kavousianos[3]
[1]*Computer Engineering & Informatics Department, University of Patras, 26500 Patras, Greece*
[2]*ELectronics Laboratory, Physics Department, University of Patras, 26500 Patras, Greece*
[3]*Computer Science Department, University of Ioannina, 45110 Ioannina, Greece*
bellos@ceid.upatras.gr, bakalis@physics.upatras.gr, nikolosd@cti.gr, kabousia@cs.uoi.gr

Abstract*. Excessive peak power dissipation during testing can result in reduced reliability and yield loss due to power and/or thermal constraint violation. In this paper we propose a novel peak power dissipation reduction method based on test vector ordering with vector repetition and vector modification. Experimental results validate that the proposed method achieves favourable peak power dissipation reductions with respect to the test sequence produced by the test vector ordering with vector repetition technique proposed a few years ago.*

1 Introduction

Power dissipation during testing has gained significant attention during the last few years due to the problems arising by elevated average and peak power dissipation. Such problems are decreased reliability and system life cycle, decreased overall yield and increased product costs [1, 2]. Several techniques have been proposed in order to reduce switching activity, consequently the amount of power consumed in a circuit, during testing. Among them post-ATPG test vector ordering techniques have been presented in [3-7]. However, most of the cases target the reduction of average power dissipation, without guaranteeing any reduction in peak power dissipation.

Peak power dissipation reduction was studied in [7, 8]. In [7] a test vector ordering with vector repetition algorithm (TVO_VR), based on the use of minimum spanning trees, was proposed. In [8] the authors attempt to insert a number of vectors between the two vectors (*i*, *j*) that are responsible for a violation in peak power dissipation. More specifically, k vectors are inserted between vectors *i* and *j* in such a manner that the Hamming distance of vectors *i* and *j* is evenly distributed over the k+1 vector pairs created by the insertion of the k vectors. However, the reduction in the Hamming distance of a vector pair cannot guarantee the reduction in the peak power dissipation.

In this work we utilize the TVO_VR algorithm and relax the constraint that the repeated vectors must originate from the initial test set so as to further reduce the peak power dissipation during testing. The proposed approach uses the output of the TVO_VR [3, 7] algorithm as the starting point for further changing the test sequence through modification, copying or removal of the vectors of the vector pair that causes a violation in peak power dissipation.

2 Preliminaries

The TVO_VR algorithm [7] receives as input an initial test set, denoted as *TV*, consisting of *n* test vectors and produces a test sequence of at most 2*n* test vectors. The test sequence produced can be split into the set of initial test vectors, denoted as *FA*, and the set of the vectors the TVO_VR algorithm repeats while trying to reduce the peak power dissipation, denoted as *R*. The vectors of set *R*, henceforth called repeated vectors, do not contribute to the final fault coverage since the vectors of set *FA* contain all the initial vectors of set *TV* and therefore can be modified in order to further reduce the peak power dissipation.

The TVO_VR algorithm provides the maximum reduction in peak power dissipation under the constraints that the test sequence length is shorter that 2*n* and that its test vectors are taken from the initial test set only. Thus using the output of the TVO_VR algorithm and modifying the vectors of set *R* we can further reduce the peak power dissipation.

Before we present the proposed peak power dissipation reduction method let us define the terms of instantaneous (IPD) and peak power dissipation (PPD) [1]. The instantaneous power dissipation is equal to the power dissipated by a pair of consecutive test vectors. If we consider all pairs of the test sequence and their respective instantaneous power dissipation values then the pair whose instantaneous power dissipation is the highest among all other pairs is the one responsible for the peak power dissipation observed at the circuit.

3 Peak power dissipation reduction

3.1 The peak power dissipation reduction method

The aim of the peak power dissipation reduction procedure is to drop peak power dissipation under a user defined threshold (*UDThr*). Our procedure reduces step by step the peak power dissipation until it reaches *UDThr* or until no further reduction is possible.

Before introducing the proposed approach we define the following operators:

a) *modify* vector (MV), which returns true if the modification of a vector reduces the IPD of the two vector pairs it participates in.
b) *copy vector* (CV), which returns true when either a copy of a vector is found in the test sequence or a vector is copied successfully elsewhere in the test sequence
c) *remove vector* (RV), which returns true when the removal of one or both of the vectors of the pair that is responsible for the current peak power dissipation is successful.

A detailed description of the three operators will be given in the following subsections. The proposed algorithm receives as input the test sequence produced by the TVO_VR algorithm and at each iteration it considers the pair of consecutive vectors that is responsible for the current peak power dissipated in the circuit. It then tries to reduce the current peak power dissipation of the pair by applying one or a combination of the above operators. Let (*i*, *j*) be the vector pair that is responsible for the current peak power dissipation in the circuit. The algorithm distinguishes 4 different cases (see Figure 1).

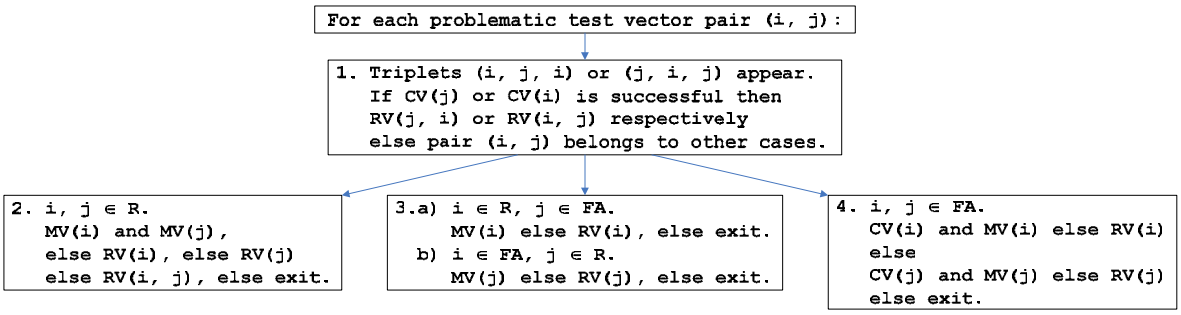

Figure 1: Peak power reduction algorithm

The first case considers the situation where one of triplets (*j, i, j*) and (*i, j, i*) exists. If we assume the zero delay model, the two vector pairs of the triplet exhibit the same IPD in the circuit, which is equal to the current PPD. If we can copy the middle vector elsewhere successfully then the removal of both the two last vectors of the triplet reduces the PPD. If however we fail to find a place to copy the middle vector of the triplet then we consider that pair (*i, j*) belongs to one of the rest cases.

The second case considers that both vectors of the pair belong to set *R*. The MV operator is at first used on both vectors *i* and *j* so as to determine which vector will be modified. Assume that vector *h* (*k*) appears before (after) vector *i* (*j*). Both calls of the MV operator will result in modified versions of vectors *i* and *j* and both the resulting triplets (*h, i′, j*) and (*i, j′, k*). Among the two modified vectors we choose the one that yields the lowest maximum IPD in the triplets above, provided that it is lower than the PPD, caused by pair (*i, j*), that is the one that fulfils the expression below:

$$\min(\max(\mathrm{IPD}(h, i'), \mathrm{IPD}(i', j)), \max(\mathrm{IPD}(i, j'), \mathrm{IPD}(j', k)) < \mathrm{IPD}(i, j)$$

If, however, the MV operator fails to lower the current PPD then we can remove one or both vectors of the pair, by applying the RV operator on vectors *i, j* or on the vector pair, without affecting the fault coverage, since both vectors are members of set *R*. The algorithm exits in case all the above vector removal actions fail to reduce the PPD.

The third case considers that one, for example *i*, of the two vectors is a member of set *FA*. Obviously, we cannot modify *i* but we can apply the MV operator on vector *j*. If the MV operator fails to lower the current PPD, then we can try to remove vector *j* by the use of the RV operator. A failure in vector removal causes the algorithm to exit.

The final case is the one where both vectors belong to set *FA*. In this case both vectors *i, j* are candidates for modification, provided that we can copy them successfully. We apply the CV operator on vector *i* and if it returns true then the MV operator is used on *i* at the initial place. If the MV operator fails then we try to remove vector *i* at the initial place. If this action also fails then we apply the same course of action on vector *j*. If this also fails the algorithm exits.

3.2 Vector modification

The *modify vector* (MV) operator modifies a vector, for example *j*, of pair (*i, j*), so as to reduce its IPD. Obviously the modification of the vector takes into account the presence of the neighboring vectors, that is *i* and *k*, and produces a new vector, say *j′*, such that both values of the IPD of pairs (*i, j′*) and (*j′, k*) are lower than the IPD of pair (*i, j*).

The modification is based on the criterion of reducing the Hamming distances of vector *j* with respect to the neighboring vectors. Specifically, the operator examines vectors *i* and *k* for similar values in corresponding bit positions and assigns the corresponding bit of vector *j* to the same value, thus minimizing the Hamming distance of vector *j* with its neighboring vectors. The IPD of pairs (*i, j′*) and (*j′, k*) is then estimated by simulation and the maximum among them is stored in variable *Current*. The reduction in the Hamming distances may not lead to reduction in IPD and thus we examine whether there is no need to change all the bits of *j*. We scan vector *j* from left to right until we find a changed bit and we restore it to its original value. If this action lowers the IPD of pairs (*i, j′*) and (*j′, k*) below the value of *Current*, then we accept the original bit value, update *Current* and rescan *j* from left to right. Upon reaching the rightmost bit of *j* without performing a bit restoration the operator returns true if the value of *Current* is lower than the IPD of the initial pair (*i, j*).

3.3 Vector copying

The *copy vector* (CV) operator always receives as input the vector pair responsible for the current PPD observed at the circuit, which is stored in variable *Target*, and the vector, say *j*, that must be copied. The operator either finds a copy of *j* or copies *j* elsewhere in the sequence so as to obtain a test sequence where the IPD of the vector pairs the copy of vector *j* participates in is lower than or equal to the current PPD observed in the circuit.

At first, the operator takes advantage of the repeated vectors of the test sequence and checks if a copy of vector *j*, say *j΄*, already exists. If this is the case, then by construction the IPD of vector pairs (*x*, *j΄*) and (*j΄*, *y*), where *x*, *y* are the neighboring vectors of *j΄*, is lower than or equal to *Target*, so the operator returns true.

When a copy of *j* does not exist then we try to copy *j* at a place in the sequence without creating test vector pairs whose IPD is higher than the value of *Target*. The place where vector *j* will be copied is determined by simulation. For each two consecutive vectors (*p*, *q*) of the test sequence we simulate the triplet (*p*, *j*, *q*) and check if it fulfills one of the following two conditions in the order mentioned below:

a) if vectors *p* and *q* belong both to the FA set then the insertion of the chosen vector must not cause IPD higher than the value of *Target* in both pairs formed, that is,

$$\text{IPD}(p, j) < \textit{Target} \text{ and } \text{IPD}(j, q) < \textit{Target} \quad (1)$$

b) if one or both vectors *p* and *q* belong to set *R* then we seek to minimize the sum of the IPD of pairs (*p*, *j*) and (*j*, *q*). Moreover if *p* or *q* is a member of set *FA* then the IPD of the pair formed by this vector and *j* must not be higher than variable *Target*.

$$\min(\ \text{IPD}(p, j) + \text{IPD}(j, q)\ ) \quad (2)$$

$$\text{if } p \in \textit{FA} \text{ or } q \in \textit{FA} \text{ then } \text{IPD}(p, j) < \textit{Target} \text{ or } \text{IPD}(j, q) < \textit{Target} \quad (3)$$

The condition in (a) is stringent since in general there may not be any pairs that abide by (1). Condition (2) may not assure that the IPD of pairs (*p*, *j*) and (*j*, *q*) is lower than the value of *Target*. However, since one or both vectors belong to set *R* we have the potential to reduce the IPD of the above pairs by the use of the MV operator. Once we have found a place to insert the copy of vector *j* we use, if possible, the MV operator on one or both of the vectors that surround the copy of *j*. The operator returns true if the IPD of each of the vector pairs affected by the presence of the copy of *j* is lower than or equal to *Target*.

3.4 Vector removal

The RV operator tries to remove a vector or a pair of vectors. It receives as input the vector *j* or pair of vectors (*i*, *j*) that need to be removed together with a target for IPD that must not be exceeded. For the first case the target is equal to the largest IPD of the pairs vector *j* participates in, while for the second it is equal to the IPD of pair (*i*, *j*). The removal of a vector yields a new test vector pair whose IPD must be lower than the target.

The operator upon removal of a vector or a pair of vectors examines if the IPD of the resulting pair is lower that the given target. If *h* and *k* are the neighboring vectors of pair (*i*, *j*) then the IPD of pair (*i*, *k*) for the case of single vector removal or the IPD of pair (*h*, *k*) for the case of vector pair removal must be lower than the given target. For the case of single vector removal if the IPD of pair (*i*, *k*) is higher than the target then we can try to reduce the IPD by using the MV operator on any of the vectors *i* or *k*, provided that at least one of them belongs to set *R*. The operator returns true if any of the above actions yields a test vector with IPD lower than the given target.

4 Comparison

In our experiments we used test sets, generated by the Synopsys ATPG tools, targeting all single stuck at faults of the ISCAS'85 benchmark circuits. These test sets were used as input to the TVO_VR algorithm [7] in order to produce the test sequences that will be further modified by the peak power reduction (PPR) algorithm.

The TVO_VR algorithm uses a graph, denoted as *TG*, where each edge is assigned a weight using one of the following metrics: a) the number of transitions activated in the CUT after the application of a test vector pair [3, 6, 7] (TRANS), b) the Hamming distances of the test vectors [4] (HAM), and c) the sum of *induced activity function* values for all primary inputs changing from 1 to 0 and vice versa after the application of a test vector pair [5] (IAF). The TVO_VR algorithm produces a minimum spanning tree (MST) of graph *TG* and utilizes the inorder traversal of the MST to produce the test sequence. In our experiments, we consider that set *FA*, consists of the vectors that appear in the produced test sequence in the same order as they appear in the inorder traversal of the MST, and set *R* consists of the remaining vectors.

The TVO_VR algorithm also uses a parameter *peak*, which in [7] is set equal to the largest edge weight of the MST, denoted as *max*. We have to note that when this parameter is set equal to a value lower than max then the peak power dissipation of the test sequence produced remains equal to *max*. In our case we set parameter *peak* equal to the user defined threshold *UDThr*. Since *UDThr* is in general lower than *max*, the peak power dissipation of the test sequence produced by the TVO_VR remains equal to *max*, but we achieve the following advantages: a) the average power dissipation of the test sequence is lower [3], b) there is an increase in the number of repeated vectors which provides a lot of alternatives for the operators of the PPR algorithm and c) the PPR algorithm must handle less vector pairs whose IPD is higher than *UDThr*. In our experiments, we do not have specific values for *UDThr*, so we set it equal to the average weight of the MST edges as proposed in [3].

In Table 1 we compare the PPD achieved by the use of the PPR algorithm against the PPD achieved by the use of the TVO_VR algorithm proposed in [7]. The PPD is estimated by the maximum number of transitions activated in the circuit by a vector pair of the test sequence assuming the zero-delay model, which according to [9] is reasonable for comparisons. As it can be seen from Table 1, the proposed technique can achieve significant savings in peak power dissipation, which can reach up to 43.9% for the case of circuit c499. Even though using the HAM and IAF metrics achieves significant savings we can observe that the actual peak power dissipation values achieved by the use of the TRANS metric are lower than the respective values concerning the other two metrics.

The average power dissipation is also lowered since the PPR algorithm reduces the instantaneous power dissipation of at least one vector pair during each iteration. The extent of average power dissipation reduction is determined by the number of vectors the algorithm modifies and copies.

Table 1: Peak power dissipation reduction achieved

	HAM		IAF		TRANS	
Circuit	**PPD**	**Reduction**	**PPD**	**Reduction**	**PPD**	**Reduction**
c432	113	18.1%	127	14.2%	91	8.1%
c499	129	11.6%	92	43.9%	78	0.0%
c880	280	23.3%	271	21.4%	198	25.6%
c1355	262	18.1%	271	17.1%	259	0.0%
c1908	511	27.9%	511	20.3%	461	10.8%
c2670	615	30.0%	604	26.3%	553	11.7%
c3540	885	24.2%	902	9.2%	746	4.0%
c5315	1596	24.4%	1629	17.5%	1579	4.5%
c6288	1861	11.0%	1861	10.0%	1767	5.4%
c7552	2489	19.3%	2019	20.6%	1928	10.0%

The time needed for the completion of the PPR algorithm is very low. In most cases, the time required for the execution of the PPR algorithm on a 500 MHz Intel® Pentium® III personal computer equipped with 384MB RAM was a few seconds and reached a few minutes for the largest circuits and initial test sets.

5 Conclusions

The use of test vector ordering with vector repetition technique presented in [7] results in test sequences with reduced peak power dissipation when compared to the test vector ordering without vector repetition approach. Even though the method presented in [7] guarantees an optimum solution in peak power dissipation when the vectors of the test sequence are taken from the initial test set, we present a method, which achieves further reduction in peak power dissipation. The proposed method takes advantage of the vectors repeated by the method of [3, 7] and through modification and copying/removal of vectors it achieves significant savings on peak power dissipation.

Acknowledgments

We thank the European Social Fund (ESF), Operational Program for Educational and Vocational Training II (EPEAEK II), and particularly the Program PYTHAGORAS, for funding the above work. An initial version of this work was financially supported by the Public Benefit Foundation “Alexander S. Onassis” via its scholarships programs and by the State Scholarships Foundation of Greece by its Post-Doctoral research scholarships program.

References

[1] P. Girard, "Survey of Low-Power Testing of VLSI Systems", IEEE Design & Test of Computers, pp. 82-92, May-June 2002.

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

[3] V. Dabholkar, et. al., "Techniques for Minimizing Power Dissipation in Scan and Combinational Circuits During Test Application", IEEE Transactions on CAD of Integrated Circuits and Systems, Vol. 17, No. 12, pp. 1325-1333, December 1998.

[4] P. Girard, et. al., "Reducing Power Consumption during Test Application by Test Vector Ordering", Proc. of International Symposium on Circuits and Systems, pp. 296-299, 1998.

[5] P. Girard, et. al., "A Test Vector Ordering Technique for Switching Activity Reduction during Test Operation", Proc. of 9th Great Lakes Symposium on VLSI, pp. 24-27, 1999.

[6] Z. Luo, et. al., "Test Power Optimization Techniques for CMOS Circuits", Proc. of Asian Test Symposium, pp. 332-337, 2002.

[7] V. Dabholkar and S. Chakravarty, "Minimizing Power Dissipation in Combinational Circuits During Test Application", Tech. Rep. 94-10, Dept. of Computer Science and Engineering, University at Buffalo, 1994.

[8] A. Macii and E. Macii, "Peak Power Constrained test sets: generation heuristics and experiments", Proc. of 6th IEEE Int. Conference on Electronics, Circuits and Systems, pp. 925-928, vol.2, 1999.

[9] A. Shen, et. al., "On Average Power Dissipation and Random Pattern Testability of CMOS Combinational Logic Networks", Proc. of International Conference on CAD, pp. 402-407, 1992.