MVP: Capture-Power Reduction with Minimum-Violations Partitioning for Delay Testing

Zhen Chen[1*], Krishnendu Chakrabarty[2], Dong Xiang[3]
[1]Department of Computer Science and Technology, [3]School of Software
[1,3]Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, China.
[2]Department of Electrical and Computer Engineering, Duke University, USA.
z-chen07@mails.tsinghua.edu.cn, krish@ee.duke.edu, dxiang@tsinghua.edu.cn

***Abstract*— Scan shift power can be reduced by activating only a subset of scan cells in each shift cycle. In contrast to shift power reduction, the use of only a subset of scan cells to capture responses in a cycle may cause capture violations, thereby leading to fault coverage loss. In order to restore the original fault coverage, new test patterns must be generated, leading to higher test-data volume. In this paper, we propose minimum-violations partitioning (MVP), a scan-cell clustering method that can support multiple capture cycles in delay testing without increasing test-data volume. This method is based on an integer linear programming model and it can cluster the scan flip-flops into balanced parts with minimum capture violations. Based on this approach, hierarchical partitioning is proposed to make the partitioning method routing-aware. Experimental results on ISCAS'89 and IWLS'05 benchmark circuits demonstrate the effectiveness of our method.**

I. Introduction

Power consumption for scan-based test is much higher than that for functional operation due to excessive switching activity during test application. This problem is exacerbated by at-speed response capture in delay-fault testing. Excessive heat may cause permanent damage to the circuit under test, reducing circuit reliability and increasing package cost. High dI/dt due to excessive switching activity may cause supply voltage droops, leading to high gate delay, which in turn may cause good chips to fail tests [16].

Power consumption is therefore a serious problem for at-speed testing, especially for capture cycles. Broadside testing (also called launch-on-capture) is commonly used to detect transition faults. It involves two consecutive at-speed capture cycles, for launching the pattern and for response capture, respectively. Therefore, capture-power reduction has been recognized as a major challenge [13, 17].

Broadside testing needs a vector pair to be applied to the circuit under test. The first test vector is denoted by V_1, and the circuit response to V_1 is the second vector V_2. The response to V_2 is denoted by R. The circuit under test operates at-speed in functional mode for the two capture cycles. The switching activity in the scan chains and resulting capture power are correlated to the Hamming distance between V_1 and V_2, and between V_2 and R [15]. We refer to these quantities as $HD(V_1,V_2)$ and $HD(V_2,R)$, respectively. If $HD(V_1,V_2)$ is high, more transitions will occur in the combinational logic during the capture cycle, leading to high dI/dt, supply-voltage droop, and the associated problem of high gate delays. This results in the failure of good chips during test application, i.e., yield loss. On the other hand, if $HD(V_2,R)$ is high, the large number of transitions in scan cells may lead to excessive power consumption. Therefore, in order to reduce the potential for yield loss, it is important to reduce peak switching activity between V_1 and V_2 (launch cycle) [13]. It is also important to reduce the switching activity between V_2 and R (capture cycle) to prevent excessive power consumption [13].

Recently, many methods have been presented to reduce test power. They can be categorized into pattern-based methods [5, 13, 17] and DFT-based methods [4, 6, 14]. Pattern-based methods include the control of test vectors using post-generation filling [5, 13] or modified ATPG [17]. Since post-generation filling can only reduce the capture power by appropriately filling the unspecified bits in the test cubes, the overall power reduction may be unsatisfactory. Methods based on modified ATPG suffer from high computational complexity and they lead to an increase in test-data volume [13].

DFT-based methods have received much attention for controlling scan power during shifting or response capture [6, 9]. However, many DFT-based methods are applicable only to stuck-at fault testing [10, 14] and they cannot be used to reduce capture power in the launch cycle or capture cycle for broadside testing. Methods that reduce capture power for broadside testing suffer from the drawback of increased test-data volume [9, 18], or significant hardware overhead [8]. A new approach is therefore needed to reduce capture power for broadside testing, with minimum impact on test-data volume and hardware overhead. In this paper, we propose a scan-cell clustering method that can support multiple capture cycles with minimum capture violations. The proposed approach can reduce capture power for broadside testing with minimum test-data overhead.

The rest of this paper is organized as follows. Background and motivation for the proposed approach are presented in Section II. The minimum-violations partitioning method is described in Section III. Section IV describes how the partitioning method can be made layout-aware. Experimental results are presented in Section V. Finally, Section VI concludes the paper.

[*]Zhen Chen carried out this work as a visiting researcher at Duke University (partially supported by the scholarship from China Scholarship Council). This work was supported in part by SRC under Contract no. 1588 and by NSF under Grant no. ECCS-0823835, also supported by NSFC under Grants 60425203 and 60910003, and by 863 Project under Grant 2009AA01Z129.

978-1-4244-8192-7/10/$26.00 ©2010 IEEE

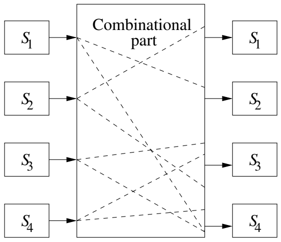

Fig. 1. Output cones of the scan flip-flops.

II. Background and Motivation

A. Background

A simple and efficient way for shift-power reduction is to activate only one scan chain in a shift cycle. Therefore, an interesting question is whether capture power for broadside testing can be reduced in a similar way, for multiple capture cycles.

The methods in [10] and [14] reduce capture power for stuck-at fault testing by using multiple capture cycles. However, they cannot be used to directly reduce capture power for broadside testing, since there are two consecutive capture cycles in broadside testing. The method in [14] replaces some scan flip-flops by pairs of *input-only* and *output-only* flip-flops in order to support multiple capture cycles. However, the output-only flip-flops cannot apply their captured responses to the combinational part in the second capture cycle for broadside testing, since their outputs are disconnected from the original fan-out logic cone. The theoretical basis of the method of [10] is that at most k capture orders are needed to achieve full fault coverage, given that the number of scan chains is k. This claim holds true for a one-time-frame circuit model (stuck-at fault testing), but it is not valid for a two-time-frame circuit model (broadside testing). Traditional broadside testing involves two capture cycles, namely the launch cycle and the capture cycle. The use of additional capture cycles and the use of a subset of flip-flops in each cycle can reduce capture power for broadside testing, but it leads to fault coverage loss. To the best of our knowledge, none of the published methods can use multiple capture cycles to reduce capture power for broadside testing, without fault coverage loss.

In this paper, we attempt to use multiple capture cycles to reduce capture power for broadside testing. However, previous studies [10, 14] demonstrate that, capture violation (explained below) is a major problem if multiple capture cycles are used. Therefore, we must address this problem before considering capture-power reduction for multiple capture cycles.

We next present an example to illustrate capture violations. In Fig. 1, the output cones of four scan flip-flops are shown, and the combinational part is in the middle. In traditional test application, a test vector (v_1, v_2, v_3, v_4) is loaded into the four scan flip-flops and then applied to the combinational part. The test response (r_1, r_2, r_3, r_4) is then captured by the four scan flip-flops. Here, v_1, v_2, v_3, v_4 are the values of flip-flops s_1, s_2, s_3, s_4, respectively. For r_1, r_2, r_3, r_4, the interpretation is similar. According to the structure of the circuit in Fig. 1, the logic function between input and output is: (i) $r_1 = f_1(v_2)$; (ii) $r_2 = f_2(v_1, v_2)$; (iii) $r_3 = f_3(v_1, v_2, v_3, v_4)$; (iv) $r_4 = f_4(v_1, v_3)$.

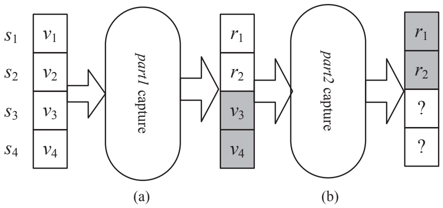

Fig. 2. Test-application scheme with capture violation.

Suppose two capture cycles are used to reduce capture power, with a cluster of scan cells capturing their values in a capture cycle. In order to show a capture violation, let us place s_1 and s_2 into the first cluster (*part*1), and s_3 and s_4 into the second cluster (*part*2). The test-application scheme with these two capture cycles is shown as Fig. 2(a)-(b).

In the test application scheme in Fig. 2(a)-(b), the capture sequence is: (first *part*1, then *part*2). The squares with gray colors represent the corresponding scan flip-flops not clocked in that cycle. After the first capture cycle, as in Fig. 2(a), the test vector (response after the first capture cycle) becomes (r_1, r_2, v_3, v_4). According to equations (iii) and (iv), the response of s_3 and s_4 after the second capture cycle may not be r_3 and r_4 because the values of s_1 and s_2 may have changed (from v_1,v_2 to r_1,r_2), as in Fig 2(b). This situation is referred to as a *capture violation*.

The use of multiple capture cycles may cause scan flip-flops to capture the faulty responses due to capture violations, leading to fault coverage loss. In order to restore the original fault coverage, new test vectors must be generated to cover the untestable faults. Therefore, multiple capture cycles may result in an increase in the test-data volume.

Therefore, the objective of this work is to minimize capture violations. Our premise is that fewer capture violations will lead to less fault coverage loss. We validate this premise on the basis of experimental results presented in Section VI.

B. Motivation

The cause of capture violations is that there is data dependency between two sets of scan flip-flops. Data dependency between them implies that response of one set depends on the test data of the other set. In the topology of the circuit, the data dependency between two scan flip-flops implies that there is a combinational path from one flip-flop to the other. For example, in Fig. 1, *part*2(s_3, s_4) has data dependency with *part*1(s_1, s_2) since there are combinational paths from *part*1 to *part*2 (e.g., s_2 to s_3). Therefore, data dependency determines the amount of capture violations.

In order to reflect the alone relationship between scan flip-flops, a directed graph called "s-graph" is used to represent the dependency relationship. In the s-graph, every node stands for a scan flip-flop and an edge (u, v) denotes the fact that there is a logical path from flip-flop u to flip-flop v in the topology of the circuit. In order to model capture violations, the following definition is given.

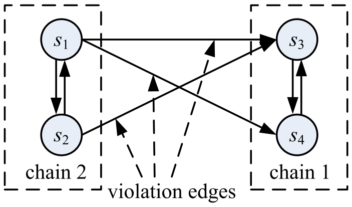

Fig. 3. Example of violation edges in an s-graph.

Definition 1 *Suppose the scan flip-flops are clustered into n scan chains, $sc_1, \ldots, sc_n$, and the capture sequence is given by $sc_n, sc_{n-1}, \ldots, sc_1$.* ***Violation edges*** *are defined as the edges from nodes in sc_i to nodes in sc_j $(i > j)$ in the corresponding s-graph.*

The capture sequence denotes the sequence of scan chains that capture responses in the multiple capture cycles. An example of an s-graph with two capture cycles is shown in Fig. 3. If scan chain 2 captures its response first, followed by scan chain 1, the capture sequence will be (chain 2, chain 1). As a result, the edges from chain 2 to chain 1 are defined as violation edges.

We must therefore minimize the number of violation edges in the s-graph in order to reduce the number of capture violations. Hence, a new partitioning algorithm is needed to obtain minimum-violation edges. The goal is to partition the scan flip-flops into scan chains to minimize the number of violation edges. The proposed partitioning method is referred to as *minimum-violations partitioning* (MVP).

III. Minimum-violations partitioning

In this section, we describe the optimization method that we use for MVP. The method is based on an integer linear programming (ILP) model.

A. ILP model for minimum-violations partitioning

The objective of minimum-violations partitioning is to minimize the number of violation edges in the s-graph. The key constraint is that scan chains must be balanced.

Assume that there are n scan cells and m scan chains. The goal is to cluster scan cells into m balanced scan chains with the least number of violation edges. We define a binary variable S_{ik}, such that S_{ik} = 1 if the scan cell i belongs to the k-th scan chain, and S_{ik} = 0 otherwise. The weight of the edge from scan cell i to j in s-graph is denoted by w_{ij}. If there is a directed edge from scan cell i to j, $w_{ij} = 1$, otherwise, $w_{ij} = 0$. A binary variable V_{ij} is defined, which is set to 1 if the edge from scan cell i to j is a violation edge; otherwise $V_{ij} = 0$. The parameter d represents the difference in length allowed between different scan chains, and it is set by the user.

ILP model for Minimum-Violations Partitioning

Objective:

- Minimize $\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} \cdot V_{ij}$

Subject to:

1 $\forall i, \quad C_i = \sum_{k=1}^{m} k \cdot S_{ik}$;
2 $\forall i, j \quad$ If $C_i - C_j \geq 1$, $V_{ij} = 1$; else, $V_{ij} = 0$;
3 $\forall i, \quad \sum_{k=1}^{m} S_{ik} = 1$;
4 $\forall k, \quad |\sum_{i=1}^{n} S_{ik} - \lceil n/m \rceil| < d$;

Fig. 4. ILP model for minimum-violations partitioning.

The complete ILP model is shown in Fig. 4. The objective is to minimize the number of violation edges. Line 1 introduces a new variable C_i, representing the index of the scan chain that i belongs to. Line 2 determines whether an edge is a violation edge. Assume that the capture sequence is from scan chain m to scan chain 1. According to Definition 1, V_{ij} = 1 if scan cell i belongs to a scan chain with higher index than scan cell j; V_{ij} = 0 otherwise. Line 3 ensures that one scan cell can only be in one scan chain. Line 4 ensures that the length difference between different scan chains should be under a threshold d.

Taking Fig. 3 as an example. The edge from s_1 to s_2 is denoted by (s_1, s_2), and the notations for others are similar. The capture sequence is (chain 2, chain 1). The indexes for s_1, s_2, s_3, s_4 are 1, 2, 3, 4, respectively. If s_1 and s_2 belong to scan chain 2, and s_3 and s_4 belong to scan chain 1, values of C_i will be: C_1 = 2, C_2 = 2, C_3 = 1, C_4 = 1. As a result, (s_1, s_3), (s_1, s_4), (s_2, s_3) are violation edges according to the second constraint in the ILP model, since $C_1 - C_3 \geq 1$, $C_1 - C_4 \geq 1$, $C_2 - C_3 \geq 1$.

Since the constraint "if .. else .." in line 2 is nonlinear, a linearization transformation is needed to ensure the constraint is linear. The following two inequalities are introduced to replace constraint 2 in Fig. 4.

1) $C_i - C_j \geq 1 - M(1 - V_{ij})$
2) $C_i - C_j \leq MV_{ij}$

Here, M is a large positive integer. It can be shown easily that the above inequalities are together equivalent to constraint 2. We can linearize constraint 4 using the inequalities $-d < \sum_{i=1}^{n} S_{ik} - \lceil n/m \rceil$ and $\sum_{i=1}^{n} S_{ik} - \lceil n/m \rceil < d$.

B. Incremental MIP solver

In the previous subsection, the partitioning problem is modeled using ILP. From constraint 2 in Fig. 4, we see that the size of the model (i.e., the number of constraints) is proportional to n^2, where n is the number of scan flip-flops. However, some constraints are redundant and they can be deleted. For the variable V_{ij}, there is no need to define it if there is no edge from scan flip-flop i to j. This is because the objective function of the ILP model is $\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} \cdot V_{ij}$, and the value of V_{ij} is not meaningful if w_{ij} is 0. If there is no edge from i to j in the s-graph, $w_{ij} = 0$, implying that there is no need to define V_{ij}.

After the unnecessary constraints are deleted, the size of the ILP model (number of constraints) becomes proportional to the number of edges in the s-graph. The number of variables in this model is proportional to n. However, ILP is known to be an NP-hard problem [7]. Therefore, simplifications are necessary for large problem instances. It is well known that linear programming (LP) problems can be solved in polynomial time [2]. Therefore, we use LP-relaxation to solve the partitioning problem [12].

In LP-relaxation, integer variables are relaxed to real-valued variables. However, the real value for S_{ik} has no practical meaning. It must be mapped to 0 or 1. Therefore, we use the incremental mixed integer programming (MIP) method to solve the model. This approach does not yield optimal results but it is computationally efficient.

In the LP problem, if we set a subset of variables to be integer type and others to be real type, the resulting MIP problem can also be solved more quickly compared to the ILP problem. Therefore, we set a small subset of variables as integer type in each step and solve the resulting MIP problem. After several iterations, where an upper bound on the number of iterations equals the number of variables in the ILP problem, all the variables will be assigned integer values. From Fig. 4, we can see that values of all other variables in the model depend on the values of the variables referred to by S_{ik}. Thus, if values of all the S_{ik} variables are fixed, all the other variables will also be fixed. Two sets, S_{int} and S_{real}, are defined as follows. The set S_{int} contains all the variables S_{ik} that are assigned as 0 or 1, while S_{real} contains others. Initially, all the S_{ik} variables belong to S_{real}, and $S_{int} = \emptyset$. Once an S_{ik} variable is placed into S_{int}, it will be fixed to be a constant instead of being a variable.

The incremental MIP method involves multiple iterations for solving MIP problem. In each iteration, some S_{ik} variables are set as integers, and the corresponding MIP problem is solved. After solving the problem, the results of these S_{ik} variables must be integer-valued. Hence, they are fixed as 0 or 1 in the next iteration, therefore the number of S_{ik} variables is reduced. The process terminates until all the S_{ik} variables are fixed to integer values. It can be easily shown that, in the worst case, the number of iterations equals the number of variables in the ILP problem.

The procedure for solving the ILP problem using incremental MIP involves three steps. The first step is to solve the corresponding LP problem. The second step is to fix all the S_{ik} that are assigned as 0 or 1 and place them into S_{int}, then, randomly pick a predetermined number of variables from S_{real} and set them as integer type. In this step, the S_{ik} variables that are fixed will be constant values instead of variables in the following iterations. In the third step, the corresponding MIP problem is solved. Steps 2 and 3 are repeated until all the variables are integer-valued. The computation time, i.e., number of iterations, can be traded off with the quality of the solution by the choice of an appropriate number of variables to be made integer in each iteration.

IV. Routing-aware Minimum-violations partitioning (RA-MVP)

In this section, we extend MVP by making it routing-aware and scalable. Although the use of LP-relaxation allow us to solve the ILP model and obtain a near-optimal solution in short time, there are still two problems: (1) the LP solver tends to take more time for larger problem instances; and (2) the scan-chain routing overhead after partitioning may be high since MVP does not consider scan-chain routing. Therefore, we next improve MVP to make it scalable and routing-aware.

In order to apply our method to large circuits, we design the s-graph hierarchically. Instead of considering each scan flip-flop as a node in the s-graph, we consider a small group of scan flip-flops as a node. Such a graph is referred to as a *high-level s-graph*. As a result, the partitioning process consists of two stages: (1) cluster all the scan flip-flops into small groups, and construct a high-level s-graph by considering a group of scan flip-flops as a node; (2) partition the high-level s-graph into two balanced parts with minimum violations. This improved partitioning method is referred to as *hierarchical partitioning*. Based on this approach, MVP can be made scalable, since the size of the model is smaller when we consider a small group of scan flip-flops as a node. The CPU time needed for solving the optimization problem is also reduced.

Routing-aware scan-chain-partitioning methods are based on the principle that neighboring scan flip-flops should be clustered into the same scan chain [1]. In our hierarchical partitioning method, routing awareness can be easily considered in the first stage. When clustering scan flip-flops into small groups, we attempt to place the neighboring flip-flops into the same group. Then, when MVP is executed, the neighboring flip-flops are placed in the same scan chain, leading to less routing overhead.

In the first stage of hierarchical partitioning, neighboring scan flip-flops are clustered into small balanced groups. There are many methods that can be used to achieve this objective [3]. Here, we use a simple method to implement it. We first obtain the layout information of all the scan flip-flops (an x-coordinate and a y-coordinate for each flip-flop) and set the number of rows and columns to be partitioned. Next, we order the scan flip-flops according to their y-coordinates, and cluster them into balanced rows along the y-axis direction. Finally, a similar method is used to cluster scan flip-flops of each row into balanced columns.

After they are clustered into balanced groups, scan flip-flops within the same group are considered as a node in the high-level s-graph, and the weight of edge from group i to group j is the number of all the edges from nodes in group i to the nodes in group j. Next MVP is applied to the high-level s-graph with a smaller number of nodes and edges than before. Based on this approach, the neighboring scan flip-flops tend to be in the same partition, hence, routing overhead is reduced. As stated before, the MVP method is more efficient than before due to the smaller size of high-level s-graph.

V. Experimental results

Experimental results on ISCAS'89 and IWLS'05 benchmark circuits are provided to demonstrate the effectiveness of the proposed method. The test patterns for the experiments were obtained using an in-house broadside test generator. The capture power is estimated using the number of transitions in the scan chain in the capture cycles, since this measure correlates well into the actual capture power [15].

A. Capture violations and test-data volume

In Table I, the relationship between capture violations and increased test-data volume is shown to validate the premise that fewer capture violations lead to less increase in test data in order to maintain the fault coverage. We assume that, to reduce capture power, only one scan chain captures its response in a capture cycle, as in Fig. 2. Due to capture violations, some faults cannot be detected in this case. As a result, new test vectors

TABLE I

RELATIONSHIP BETWEEN THE NUMBER OF CAPTURE VIOLATIONS AND INCREASE IN TEST-DATA VOLUME

Circuit name	RP		Min-cut [11]		MVP (Proposed)	
	No. of violation edges	Increase in test-data volume	No. of violation edges	Increase in test-data volume	No. of violation edges	Increase in test-data volume
s9234	739	47.09%	24	6.12%	12	0.00%
s13207	876	18.27%	68	0.86%	17	0.52%
s15850	1485	15.40%	47	3.42%	2	0.00%
s38417	2999	18.42%	164	0.00%	0	0.00%
s38584	1339	14.17%	502	2.19%	54	1.11%
usb	5528	36.98%	432	5.23%	37	2.16%
pci	16552	31.63%	853	4.22%	0	0.00%

usb: usb_funct; pci: pci_bridge32; RP: Random Partitioning

must be generated to compensate for the fault coverage loss. In this table, three partitioning methods with different numbers of capture violations are shown: (1) random partitioning, which clusters scan flip-flops into two parts randomly; (2) min-cut [11], which attempts to partition nodes into two parts with minimum connections (therefore, the number of violation edges is smaller compared to random partitioning); (3) MVP, the proposed minimum-violations partitioning method.

The second, fourth, and sixth columns show the number of violation edges for the three different methods. The third, fifth, and seventh columns list the percentage increase in the number of test patterns needed to restore the original fault coverage. From the simulation results, we conclude that the increase in test data is less if the number of violation edges is smaller. We also note that MVP is more efficient for scan flip-flop partitioning with a minimum number of capture violations, and it leads to negligible increase in test-data volume.

B. Performance of MVP

In Table II, test power, test-data volume, and partitioning results are presented. The parameters TD, CP, PP represent the test data, average capture power, and peak capture power, respectively. TD is calculated using the formula $TD = \frac{T-T'}{T'} \cdot 100\%$, where T and T' represent test-data volume for the proposed method and for conventional broadside testing, respectively. Parameters CP and PP represent the percentage reduction in average capture power and peak capture power, relative to the results for conventional broadside testing. The parameters $part_1$ and $part_2$ represent the sizes of the two partitions, respectively ($part_3$ and $part_4$ are relevant when there are four partitions). The last column shows the CPU time spent on MVP using a workstation with a 3 GHz CPU and 12 GB memory. For the larger circuits, pci_bridge32 and ethernet, we also present results for four-part partitioning.

From the results, we can see that the capture power reduction is nearly 50% for two scan chains, and 75% for four scan chains. Since only one scan chain is active in a capture cycle, the capture power reduction is indeed as expected. The comparison between $part_1$ and $part_2$ shows that our method partitions the set of scan flip-flops into two balanced parts. For almost all circuits, the number of violation edges is small, hence, the impact of capture violations is low. We also note that the increase in test data is negligible. Therefore, the proposed method can reduce capture power significantly with minimum increase in test-data volume.

Compared to partitioning into two scan chains, capture power reduction is greater when we partition the scan flip-flops into four scan chains. However, the test data increases since the number of violation edges becomes more. Therefore, there is a tradeoff between the capture power reduction and test-data volume.

C. Comparison with other methods

In Table III, our method is compared with the methods in [9] and [18] with respect to peak capture power, average capture power, test-data volume and test application time. The methods in [9] and [18] can reduce capture power for broadside testing, but they increase the test-data volume and test application time considerably. We implemented the methods presented in [9] and [18] to compare than with the proposed technique.

All results presented here are relative to the results for conventional broadside testing. From the comparisons on capture power, we see that the average capture power and peak capture power for MVP are lower than for [9] and [18]. From the comparisons on test-data volume and test time, we see that the increase in test-data volume and test-application time for MVP is much less than for [9] and [18]. Therefore, we conclude that MVP can reduce capture power more effectively and with less test-data volume and test time.

D. Performance of RA-MVP

In Table IV, the routing overheads for MVP and routing-aware MVP (RA-MVP) are presented to show that hierarchical partitioning is routing-aware. The routing method from [3] is used as the metric to evaluate scan-chain length. The second and fourth columns list the percentage of increase in test-data volume. The third and fifth columns show the percentage scan-chain routing overhead. The routing overhead is determined relative to the scan-chain wire length that is obtained after wire-length minimization, without considering capture power for broadside testing. From the comparison, we conclude that RA-MVP leads to much less routing overhead compared to MVP. However, the number of violation edges after partitioning is slightly larger due to less freedom available for partitioning. As a result, test-data volume increases with the reduction of routing overhead. Nevertheless, RA-MVP offers a practical approach for reducing capture power.

VI. CONCLUSIONS

We have presented a new design-for-testability method for reducing capture power in broadside delay testing. We have studied the relationship between the number of capture violations and the increase in test-data volume. Based on this relationship, we have introduced a minimum-violations partitioning method to cluster the scan flip-flops into balanced partitions with minimum number of capture violations. We have extended MVP to make it routing-aware by using hierarchical

TABLE II

PERFORMANCE OF MINIMUM-VIOLATIONS PARTITIONING FOR BROADSIDE TESTING

Circuit name	No. of flip-flops	No. of scan chains	No. of patterns in original test set	No. of additional patterns	*TD* (%)	*CP* (%)	*PP* (%)	$part_1$	$part_2$	$part_3$	$part_4$	No. of capture violations	CPU time for MVP (minutes)
s9234	228	2	567	2	0.35	48.46	43.43	114	114	N/A	N/A	12	0.30
s13207	669	2	666	42	6.31	49.92	46.60	335	334	N/A	N/A	17	0.38
s15850	597	2	526	3	0.57	49.46	42.06	298	297	N/A	N/A	2	10.75
s38417	1636	2	1469	0	0.00	48.24	45.11	818	818	N/A	N/A	0	27.42
s38584	1452	2	1287	19	1.48	47.10	46.50	726	726	N/A	N/A	54	26.30
usb_funct	1746	2	2489	141	5.66	49.40	45.88	823	823	N/A	N/A	37	30.38
pci_bridge32	3359	2	2513	0	0.00	48.91	48.75	1680	1679	N/A	N/A	0	65.88
		4	2513	164	6.51	72.88	69.55	840	840	840	839	16	78.88
ethernet	10544	2	10167	510	5.02	49.83	46.10	5272	5272	N/A	N/A	78	88.12
		4	10167	827	8.13	73.08	71.40	2636	2636	2636	2636	198	99.81

TD: Increase in test-data volume; *CP*: average capture power; *PP*: peak capture power. Data for *CP* and *PP* are relative to the results for conventional "capture power-oblivious" broadside testing.

TABLE III

COMPARISON BETWEEN VARIOUS METHODS IN TERMS OF CAPTURE POWER REDUCTION AND TEST-DATA VOLUME

Circuit	Results for [9]				Results for [18]				MVP (Proposed method)			
name	*PP* (%)	*CP* (%)	*TD* (%)	*TA* (%)	*PP* (%)	*CP* (%)	*TD* (%)	*TA* (%)	*PP* (%)	*CP* (%)	*TD* (%)	*TA* (%)
s9234	27.22	31.20	19.62	19.62	24.82	28.29	34.00	34.00	43.43	48.46	0.35	0.44
s13207	25.64	29.45	25.68	25.68	19.38	22.59	45.11	45.11	46.60	49.92	6.31	6.66
s15850	26.40	29.65	14.69	14.69	22.50	30.06	32.23	32.23	42.06	49.46	0.57	0.65
s38417	34.56	36.78	19.89	19.89	18.98	24.83	28.31	28.31	45.11	48.24	0.00	0.03
s38584	39.97	43.92	29.15	29.15	27.74	33.35	41.72	41.72	46.50	47.10	1.48	1.56
usb_funct	34.88	37.50	31.84	31.84	24.98	31.78	36.20	36.20	45.88	49.40	5.66	6.15
pci_bridge32	37.58	39.96	23.83	23.83	18.98	25.78	38.43	38.43	48.75	48.91	0.00	0.04
ethernet	30.20	34.30	38.46	38.46	27.00	33.10	44.91	44.91	46.10	49.83	5.02	5.09

TD: Increase in test-data volume; *TA*: Increase in test application time; *CP*: average capture power; *PP*: peak capture power. Data for *CP* and *PP* are relative to the results for conventional "capture power-oblivious" broadside testing.

TABLE IV

TRADEOFF BETWEEN MVP AND RA-MVP

	MVP		RA-MVP		
Circuit name	Increase in test-data volume (%)	Routing overhead (%)	Increase in test-data volume (%)	Routing overhead (%)	CPU time for RA-MVP (minutes)
s9234	0.35	19.29	2.62	6.52	0.21
s13207	6.31	40.15	9.27	11.17	0.25
s15850	0.57	31.38	5.70	3.82	3.01
s38417	0.00	46.38	6.77	9.62	13.30
s38584	1.48	40.11	4.76	8.80	15.98
usb_funct	5.66	46.89	12.33	11.50	16.58
pci_bridge32	0.00	47.83	10.07	7.11	21.29
ethernet	5.02	56.37	13.12	9.10	32.02

partitioning. Experimental results for ISCAS'89 and IWLS'05 benchmark circuits show that the proposed method can reduce capture power more effectively, and with less test-data volume, compared to other recent methods.

REFERENCES

[1] D. Berthelot, S. Chaudhuri, and H. Savoj, "An Efficient Linear Time Algorithm for Scan Chain Optimization and Repartitioning," in *Proc. of ITC*, pp. 781-787, 2002.

[2] D. Bertsimas and J. N. Tsitsiklis, "Introduction to Linear Optimization," *Athena Scientific*, 1997.

[3] Y. Bonhomme et al., "Design of routing-constrained low power scan chains," in *Proc. of DATE*, pp. 62-67, 2004

[4] K. M. Butler, "Minimizing power consumption in scan testing: pattern generation and DFT techniques," in *Proc. of ITC*, pp. 355-364, 2004.

[5] A. Chandra and K. Chakrabarty, "Low-power scan testing and test data compression for systems-on-a-chip," *IEEE Trans. on Computer-Aided Design*, vol. 21, no. 5, pp. 597-604, 2002.

[6] J. Chen, C. Yang, and K. J. Lee,"Test pattern generation and clock disabling for simultaneous test time and power reduction," *IEEE Trans. on Computer-Aided Design*, vol. 22, no. 3, pp. 363-370, 2003.

[7] M. R. Gary and D. S. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness," *W. H. Freeman & Co.*, 1990.

[8] P. Girard et al., "Low Power BIST Design by Hypergraph Partitioning: Methodology and Architectures," in *Proc. of ITC*, pp. 652-661, 2000.

[9] H. F. Ko and N. Nicolici, "A Novel Automated Scan Chain Division Method for Shift and Capture Power Reduction in Broadside At-Speed Test," *IEEE Trans. on Computer-Aided Design*, vol. 27, no. 11, pp. 2092-2097, 2008.

[10] K.-J. Lee, S.-J. Hsu, and C.-M. Ho, "Test Power Reduction with Multiple Capture Orders," in *Proc. of Asian Test Symposium*, pp. 26-31, 2004.

[11] G. Karypis et al., "Multilevel Hypergraph Partitioning : Applications in VLSI Domain," Technical Report, Department of Computer Science, University of Minnesota, 1998. Available on the WWW at URL http://www.cs.umn.edu/-karvpis/metis.

[12] P. Raghavan and C. D. Thompson, "Randomized rounding: A technique for provably good algorithms and algorithmic proofs," *Combinatorica*, 7(4), pp. 365-374, 1987.

[13] S. Remersaro et al., "Preferred fill: a scalable method to reduce capture power for scan based designs," in *Proc. of ITC*, paper 32.2, 2006.

[14] P. Rosinger, B. M. Al-Hashimi, and N. Nicolici, "Scan architecture with mutually exclusive scan segment activation for shift and capture power reduction," *IEEE Trans. on Computer-Aided Design*, Vol. 23, no. 7, pp. 1142-1153, July, 2004.

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

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

[17] X. Wen et al., "Low-capture-power test generation for scan-based at-speed testing, in *Proc. ITC*, pp. 1019-1028, 2005.

[18] Z. Zhang et al., "Enhancing Delay Fault Coverage through Low Power Segmented Scan", in *Proc. of ETS*, pp. 21-28, 2006.