Pattern-Directed Circuit Virtual Partitioning for Test Power Reduction

Qiang Xu
Dept. of Computer Science & Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
qxu@cse.cuhk.edu.hk

Dianwei Hu
Dept. of Computer Science & Technology
Tsinghua University
Beijing 100084, China
hdw@mails.tsinghua.edu.cn

Dong Xiang
School of Software
Tsinghua University
Beijing 100084, China
dxiang@tsinghua.edu.cn

Abstract

For a large circuit under test (CUT), it is likely that some test patterns result in excessive power dissipations that exceed the CUT's power rating. Designers may resort to low-power automatic test pattern generation (ATPG) tools to solve this problem, which, however, usually leads to larger test data volume and requires extra computational effort, even if such tools are available. Another method is to partition the circuit into multiple subcircuits and test them separately. Unfortunately, this usually involves rerunning the time-consuming ATPG for each partitioned subcircuit and solving the problem of how to achieve an acceptable fault coverage for the glue logic between subcircuits. In this paper, we propose a novel low-power virtual test partitioning technique without the above-mentioned shortcomings. The basic idea is to partition the circuit in such way that the faults in the glue logic between subcircuits can be detected by patterns with low power dissipation that are applied at the entire circuit level, while the patterns with high power dissipation can be applied within a partitioned subcircuit without loss of fault coverage. Scan chain routing cost has also been considered during the partitioning process. Experimental results show that the proposed technique is very effective in reducing test power.

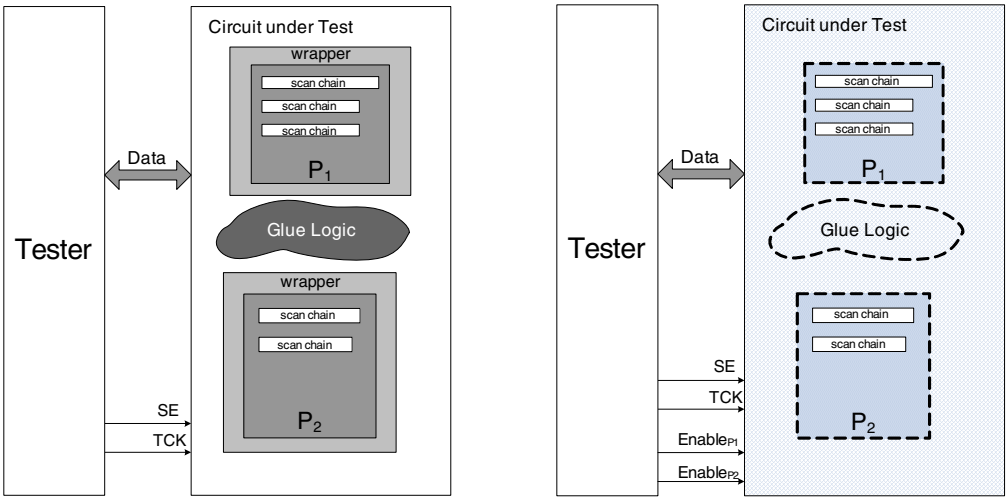
1 Introduction

One of the major concerns in seeking a reliable test strategy for today's very large scale integration (VLSI) integrated circuit (IC) is that, an IC's power dissipation during testing can be significantly higher than that during normal operation [8, 13]. This brings the following problems: (i) the accumulated effect of test power dissipation can generate elevated test heat that requires more expensive package or causes permanent damage to the circuit under test (CUT); (ii) the excessive instantaneous test power dissipation can result in large voltage drop that causes circuit to malfunction in test mode only and thus lead to yield loss [20]. Therefore, reducing power consumption has become an important objective of today's test development process.

Most of the prior research in low power testing has been focused on scan-based circuits as full-scan is the most widely adopted test strategy in the industry. In a full-scan circuit, all the internal flip-flops (FFs) are replaced by scan-FFs and operate in two modes during test: shift mode and capture mode. In shift mode, scan-FFs form scan chains, through which test stimuli/responses can be shifted in/out so that we are able to control/observe all the internal memory elements. In capture mode, scan-FFs operate as functional FFs such that the test stimuli are applied to the combinational portion of the circuit and the test responses are stored into these FFs themselves in the next clock cycle. It is possible that the test power consumption exceeds the circuit's power rating in both shift mode and capture mode. Techniques based on scan chain manipulation (e.g, [1, 2, 16, 26]) are very effective in reducing scan shift power, but does not help in reducing scan capture power. There are also some other approaches that reduce the switching activities of the CUT by taking advantage of the 'don't-care' bits in test cubes, e.g., the low-power automatic test pattern generation (ATPG) techniques in [6, 22] and the test vector manipulation methodologies in [7, 15, 17, 25]. Some of them are able to reduce scan shift power while the others are helpful in reducing scan capture power.

Even after applying the above techniques, it is possible that there remains some patterns that exceed the circuit's power rating if the CUT is large. One of the solutions in such cases is to rerun low power ATPG for the faults that were detected by those problematic patterns [19, 23]. However, even if such ATPG tools are available, they generally result in larger test data volume and are computationally expensive. Another solution is to partition the original circuit into multiple subcircuits and test them separately through clock gating [9]. This not only significantly reduces the power consumed in the logic part, but also reduces the power consumed in clock tree, which is a major contributor to test power consumption [14]. Partitioning the circuit for test, however, usually involves rerunning the time-consuming ATPG process for the partitioned subcircuits and solving the problem of how to achieve an acceptable fault cover-

1-4244-1128-9/07/$25.00



(a) Conventional Partitioning Scheme

(b) Proposed Virtual Partitioning Scheme

Figure 1. Circuit partitioning for test power reduction.

age for the glue logic between subcircuits. *An interesting question is whether partitioning for test can be conducted without the above-mentioned shortcomings?*

To tackle the above problem, in this paper, we propose a pattern-directed "*virtual*" partitioning technique[1] that exploits given test patterns to optimally direct the partitioning process (for test only), such that the patterns with high power dissipation can be applied within a partition to reduce test power without fault coverage loss while all the rest of the faults are covered by low-power test patterns applied at the entire circuit level (i.e., not partitioned). Since scan chains are built within each partitioned subcircuits, the partitioning strategy has implications on the scan chain routing cost, which has also been taken into account during the partitioning process. Please note that the proposed methodology can be applied at the core level and is hence compatible with the various test scheduling techniques presented in the literature (e.g., [5, 11, 27]), which involves scheduling groups of cores to be tested at different time in order to meet given power constraints.

The organization of the rest of the paper is as follows. Section 2 gives the preliminaries and formulates the problem that we address in this paper. The proposed pattern-directed circuit virtual partitioning algorithms are detailed in Section 3. Next, Section 4 takes scan chain routing cost into consideration during the partitioning process. Experimental results on several ISCAS'89 and ITC'99 benchmark circuits are shown in Section 5. Finally, Section 6 concludes this paper.

[1]The "virtual" here means that no extra design for test (DFT) logic need to be placed between the partitioned subcircuits and not all test patterns are applied to the partitioned subcircuits.

2 Preliminaries and Problem Formulation

There are numerous options to partition the circuits into roughly-equal sized subcircuits. The traditional method is to partition the circuit based on its logic function and/or layout position. After the circuit is partitioned (say, into two subcircuits P_1 and P_2), in addition to properly testing the subcircuits, it is also important to achieve an acceptable fault coverage for the glue logic between P_1 and P_2. One way to do this is to add wrappers to the two subcircuits and test the glue logic using boundary scan external test, as shown in Fig. 1(a). Testing glue logic in this manner usually does not exceed the circuit's power constraint as only the wrapper cells and the glue logic need to be enabled (i.e., the internal logics of the partitioned subcircuits can be kept quiescent). However, this method requires extra design efforts and usually leads to nontrivial timing/area overhead to the design.

In the above partitioning methodology, all the test patterns are applied with reduced power consumption (i.e., applied for a subcircuit or for the glue logic only). This is, however, unnecessary because the power consumptions for a CUT's test patterns generally vary significantly. For those patterns with their test power consumptions within the CUT's power rating, it is safe to apply them at the entire circuit level. Only those test patterns with high-power dissipation need to be applied within a partitioned subcircuit. As a result, without introducing wrappers, we can divide the entire test process into two test sessions: (i) the session with only one subcircuit's internal flip-flops enabled in test mode while the other partition is disabled through clock gating to reduce test power, in which we test the internal faults in the

partitioned subcircuits and (ii) the session with the flip-flops in multiple subcircuits enabled in test mode to test the rest of the faults, including the ones in glue logic.

Oftentimes only a few bits in a test pattern are essential to detect all the faults covered by it, denoted as "care-bits" of the pattern. Therefore, as long as the partitioned subcircuits contain all the "care-bits" of the high-power test patterns applied in the first test session, there is no fault coverage loss for the proposed methodology. Consider an example circuit containing 100 flip-flops with two test patterns violating its power rating and suppose one of them is able to detect all faults that are covered by this pattern by using the first 30 flip-flops only; while the other pattern can be applied with the last 40 flip-flops only without fault coverage loss, then *only when applying the two high-power test patterns* we can treat the circuit as two partitioned subcircuits (say, one contains the first 50 flip-flops while the other contains the other 50 flip-flops) and disable one of them during test application to save test power. For the other 98 test patterns, we simply apply them at the entire circuit level and they are able to cover all the rest of the faults (including the ones between the two partitioned subcircuits).

While the proposed test application scheme seems to be similar to the one in [18] that reduces test power by disabling some scan chains for certain test patterns, the motivations behind the two problems and the methodologies are significantly different. In [18], the scan chains in the CUT are similarly divided into two sets (say, set A and set B), but unlike the proposed approach in this paper, only set B is disabled from time to time during test application while set A operates normally all the time. In addition, [18] divides the CUT during the ATPG process, while the proposed method starts with given test patterns. In addition to the above, perhaps the most important difference between our approach and the one in [18] is that [18] does not take the power consumption of the test pattern into consideration and it is very likely that the high-power patterns are applied with both sets of scan chains enabled and hence the CUT's power rating can be violated.

We mainly consider scan capture power during the partitioning process because the excessive scan shift power problem can still be dealt with scan chain reordering techniques (e.g., [4, 10]) and/or using lower scan shift frequencies after partitioning (at the cost of longer testing time), while the scan capture power is determined once the partitions are fixed. At the same time, for at-speed testing, problems are more likely to arise during scan capture than during scan shift due to the higher capture frequency [3]. Moreover, partitioning the circuit into two roughly-equal sized subcircuits and shifting test patterns to them at different time (if necessary) are able to reduce the scan shift power significantly as well.

To apply the above proposed session-based test strategy, the only hardware modification is to let the CUT be able to enable only one partition during the first test session. As shown in Fig. 1(b), two additional input signals $Enable_{P1}$ and $Enable_{P2}$ (controlled by the external tester) are introduced, which are used to clock gating the two subcircuits when they are deactivated. In the CUT's first test session, only one of the signals is set as logic '1' during the scan capture phase to save test power; while in the second test session both signals are activated to capture the test responses at the entire circuit level.

For the testing time penalty with the above session-based test strategy, the two subcircuits can be shifted concurrently at an appropriate frequency so that the circuit's average power constraint is not violated. Therefore, only one or two capture cycles (depending on whether it is stuck-at test or delay test) are increased for each test pattern in the second test session.

We assume given test sets with all bits specified for full-scanned circuits in this paper, which has been generated either directly from ATPG tools or by applying various kinds of X-filling techniques on test cubes (e.g., random filling or low-power filling as shown in [17, 25]) after ATPG. The *pattern-directed circuit virtual partitioning problem* considered in this paper is formulated in the following:

Problem P_{par}: Given test sets with all bits specified, partition the CUT into roughly equal-sized subcircuits to reduce the circuit's maximum scan capture power consumption as much as possible without regenerating test patterns and without fault coverage loss.

For the sake of simplicity, we discuss how to partition the CUT into two subcircuits only in this paper. The proposed approach, however, can be easily extended to support partitioning the circuit into any number of subcircuits.

3 Pattern-Directed Partitioning

For a particular high-power problematic pattern, as long as all its care-bits in both the stimulus and the response are put into the same partitioned subcircuits, then this test pattern can be applied within that partition with less test power and at the same time without loss of fault coverage. Based on the above observation, the original problem P_{par} can be translated into how to partition the circuit such that the "care-bits" of as many as possible high-power patterns belong to a single partition.

We can model this problem as a hypergraph partitioning problem with each vertex corresponding to a scan-FF and hyperedges corresponding to the high-power problematic test patterns are added by connecting their care bits (vertices). Our objective is to partition the circuit into roughly equal-sized subcircuits without any hyperedge cuts (a hyperedge cut means that the corresponding pattern cannot be

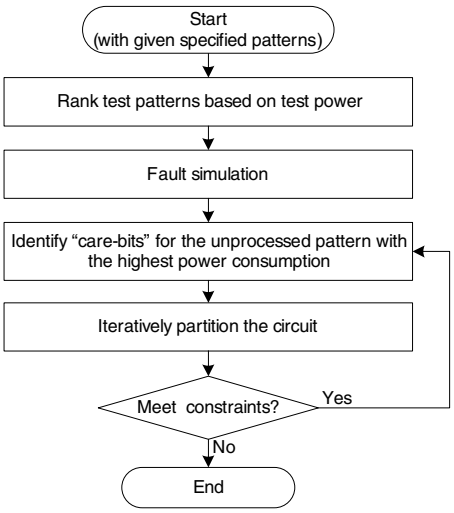


Figure 2. Design flow for pattern-directed circuit virtual partitioning.

applied within a partitioned subcircuit without fault coverage loss). Hypergraph partitioning problem is a well-researched problem and it is possible to use public tools (e.g., the *hMetis* package [21]) to solve our problem. However, because we need to add the hyperedges to the hypergraph first before running the hypergraph partitioning procedure, determining the maximum number of hyperedges added to the hypergraph that results in no hyperedge cut in this manner is a "try-and-error" process. Therefore, we propose a more efficient and effective heuristic by taking the specific characteristics of our problem into account. The basic idea is based on the following observation: if the care-bits of two test patterns have intersection, all their care-bits have to be put in the same partitioned subcircuit. Therefore, instead of partitioning the circuit as a whole, the proposed heuristic processes the high-power test patterns one by one and iteratively groups their care-bits until a pre-defined partitioning constraint is violated.

The proposed design flow for this problem is shown in Fig. 2. We first rank the test patterns based on their scan capture power values, which can be obtained from power simulation or calculated according to a scan capture power model. For the sake of simplicity, in this paper, we select to use the scan capture transition count of the test patterns[2] as the mechanism to rank these test patterns.

In order to effectively apply more high-power patterns within a single partition, we would like to let the high-power patterns to have as few care-bits as possible. Therefore, if a fault is detected by multiple test patterns, we will let the test pattern with the lowest scan capture power to cover it

[2]The scan capture transition count of a test pattern is defined as the total number of transitions of all circuit nodes in scan capture mode, including both FFs and combinational gates.

Algorithm 1 - IdentifyCareBits

INPUT: v, $C_{existing}$
OUTPUT: C_v, $C_{existing}$

1. Initialize $C_v = \emptyset$;
2. **for** every f detected by v {
3. Identify C_r for f;
4. **for** $c_i \in C_r$ {
5. $C_{s\_i} = LimitedImplication(c_i, f)$;
6. $C_{f\_i} = C_{s\_i} \bigcup \{c_i\}$;
7. $cost_i = |C_{f\_i} \bigcup C_v \bigcup C_{existing}|$;
8. $C_f = \min_{cost\_i}\{C_{f\_i}\}$;
. }
9. $C_v = C_v \bigcup C_f$;
. }
10. $C_{existing} = C_{existing} \bigcup C_v$;
11. return C_v, $C_{existing}$;

Figure 3. Procedure for identifying care-bits of a test pattern.

(we do not consider *N-detect* here for the sake of simplicity in discussion). This is done by running fault simulation for the test patterns in a non-decreasing scan capture power order (i.e., the pattern with the lowest scan capture power first). After this step, we guarantee that every pattern only detects faults that are not covered by patterns with lower scan capture power.

Next, we mark the care-bits of the test patterns in a reverse order (i.e., the pattern with the highest scan capture power first). The procedure for this task is shown in Fig. 3, which takes the currently-processing test vector v and the care bits that have already been marked by previous-processed patterns $C_{existing}$ as inputs and outputs C_v, the care-bits of v and updates $C_{existing}$ after applying v. After initializing C_v (Line 1), for every fault f that is detected by pattern v, its *test response care-bits* C_r are first identified, which are the bits that differentiate the correct response from the erroneous one (Line 3). We are able to detect a fault as long as one bit of the test response is different from the correct one. Therefore, when multiple bits are different for a particular fault (i.e., $|C_r| > 1$), we need to pick just one of them as the test response care-bit for this fault. To find out the best choice, we iteratively try every $c_i \in C_r$ in the inner loop of the procedure and select the one with the minimum cost (Lines 4-8). In each iteration, we first mark the *test stimuli care-bits* that activate f, which can be identified by searching the fan-ins of c_i. For stuck-at test or skewed-load test that requires one capture cycle, we complete this duty by looking at only the first-level fan-ins of the test response care-bit; while for broadside test with two capture cycles, we need to look for its two-levels' fan-ins. We reuse the "limited implication" procedure in [12] to obtain the test stimuli care-bits C_{s_i} in

our fully-specified test pattern v for fault f (Line 5). The care-bits used to detect fault f with test response care-bit c_i can then be obtained by the aggregation of C_{s_i} and $\{c_i\}$ (Line 6). As we want to have as few newly-introduced care-bits as possible after applying test pattern v, Line 7 calculates the cost of each candidate as the total number of care-bits when C_{f_i} is introduced. We then select the one with the minimum cost and add these bits into C_v (Lines 8-9). After applying the above procedure for every faults detected by v, the procedure updates the care-bits that are marked by all processed patterns $C_{existing}$ and returns the care-bits C_v for test pattern v and the updated $C_{existing}$ (Lines 10-11).

The maximum number of test response care-bits for a test vector is the number of faults it needs to detect (*i.e.*, $|F_{detected}|$), which is usually a small number for the high-power patterns after conducting the reverse fault simulation (see Fig. 2). By avoiding to select the care bits in the test response with a large number of fan-ins, the corresponding test stimuli care-bits can be also limited to a small number. This lays a solid foundation for us to have as many as possible test patterns to be applied within a partitioned subcircuit, as discussed in the following paragraphs.

As shown in Fig. 4, the procedure *IterativePartition* takes the given test patterns $Pattern_{set}$ and the size difference constraint for the two partitioned subcircuits d_{size} (in percentage) as inputs and outputs the partitioned subcircuits P_1 and P_2. Line 1 sorts $Pattern_{set}$ in non-increasing test power order (based on scan capture transitions here). Next, in Line 2, we calculate the maximum allowed size of a partitioned subcircuit $Size_{max}$ based on the size difference constraint d_{size}, which equals the total number of scan-FFs N_{FF} divided by $2+d_{size}$. Next, we initialize the *care-bits groups* and $C_{existing}$ before further processing (Lines 3-4). Inside the loop (Lines 5-17), we work on the highest unprocessed pattern $pattern_{max}$ in each iteration. Line 6 sets a temporary care-bit group P_{temp} to be the care-bits of $pattern_{max}$, identified using the procedure shown in Fig. 3. Next, we search all the existing care-bits groups to find out whether they have intersection with P_{temp}. If they do, all of them need to be merged with P_{temp} so that these test patterns can be applied in the same partitioned subcircuit (Lines 7-9). If the size of P_{temp} is smaller than the maximum allowed size $Size_{max}$ after this merging process, we shall update the care-bits groups (Line 11). Otherwise, we are not able to apply this pattern within a partition and the procedure needs to generate the partitioned subcircuits P_1 and P_2 and terminates (Lines 12-17), in which the maximum care-bits group is set as P_1 in the beginning (Line 12) and the rest of the care-bits groups are merged together to form P_2 (Lines 13-15). There might be still some scan-FFs that are not assigned to any partition. They are then evenly distributed to P_1 and P_2 in Line 16. Finally, the procedure returns P_1 and P_2, the roughly-equal sized subcircuits after partitioning.

Algorithm 2 - IterativePartition

INPUT: $Pattern_{set}$, d_{size}
OUTPUT: P_1, P_2

1. Sort $Pattern_{set}$ in non-increasing power order;
2. Calculate $Size_{max} = \lceil \frac{N_{FF}}{2+d_{size}} \rceil$;
3. Initialize $N_{groups} = 1$; $P_{group\_1} = \emptyset$;
4. Initialize $C_{existing} = \emptyset$;
5. **for** unprocessed $pattern_{max} \in Pattern_{set}$ {
6. $P_{temp} = IdentifyCareBits(pattern_{max}, C_{existing})$;
7. **for** i from 1 to N_{groups} {
8. **if** $(P_{temp} \bigcap P_{group\_i} \neq \emptyset)$ {
9. $P_{temp} = P_{temp} \bigcup P_{group\_i}$;
. }
. }
10. **if** $(|P_{temp}| < Size_{max})$ {
11. update P_{groups}, N_{groups};
. } **else** {
12. $P_1 = \max\{P_{groups}\}$;
13. **for** i from 1 to N_{groups} {
14. **if** $(P_{group\_i} \neq P_1)$ {
15. $P_2 = \bigcup_i P_{group\_i}$;
. }
. }
16. Distribute unassigned scan-FFs and update P_1, P_2;
17. **break**;
. }
. }
18. return P_1, P_2;

Figure 4. Procedure for pattern-directed circuit virtual partitioning.

4 Routing-Aware Partitioning

As scan chains are built within partitioned subcircuits, how to partition the CUT has a significant impact on the scan chain routing cost. In this section, we take this cost factor into consideration and revise the heuristics shown in the previous section to be routing-aware.

First of all, when identifying care-bits for a test pattern, we should take care not to let the care bits spread all over the CUT. We model the spreadness of a set of care-bits C_{set} (denoted as S_{set}) as the sum of the Euclidian distance from the position of every care bit $c_i \in C_{set}$, $P_i = (x_i, y_i)$, to the mid point position of C_{set}, calculated as $P_m = (\frac{\sum_i x_i}{|C_{set}|}, \frac{\sum_i y_i}{|C_{set}|})$. That is, $S_{set} = \sum_i distance(P_i, P_m)$. We then revise the cost function used in procedure $IdentifyCareBits$ as follows:

$$cost_i = w \times |C_{f\_i} \bigcup C_v \bigcup C_{existing}| + S_{C_v \cup C_{f\_i}} \quad (1)$$

, in which $S_{C_{set} \cup C_{f\_i}}$ denotes the spreadness of the current care-bits of the test vector and w is a weighting factor to scale the two cost terms to be in similar range.

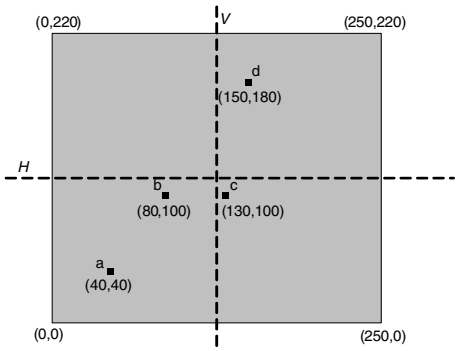


Figure 5. Care-bits distribution cost.

We should also consider scan chain routing cost when grouping the care-bits of different test patterns during the iterative partitioning process. That is, we would like the scan-FFs within the partitioned subcircuits to be physically adjacent. From this perspective, it is preferable that the CUT are partitioned either horizontally (i.e., divided into left and right region) or vertically (i.e., divided into top and bottom region) so that the two subcircuits are clustered at different side. We try to distribute the care-bits in both manners and select the one with smaller routing cost, which is modeled to be the sum of the distance from those care-bits that do not belong to the physical region it is assigned to the middle line of the CUT. For example, suppose one test pattern contains four care-bits as illustrated in Fig. 5, if the care-bits of the CUT are to be distributed in a horizontal manner, then this test pattern should be assigned to the left region and the cost of the distribution is sum of the distance from c to line V and the one from d to line V, which equals $(130-125)+(150-125)=30$; if the care-bits of the CUT are to be distributed in a vertical manner, however, then this test pattern should be assigned to the bottom region and the cost of the distribution is the distance from d to line H, which equals $180-110=70$. Therefore, if currently there is only this single care-bits group, we should select to use horizontal distribution.

The routing-aware procedure $RA-IterativePartition$ is shown in Fig. 6. Compared to the original heuristic in Fig. 4, it takes an additional input, a user-defined routing constraint $R_{constraint}$ (representing the total scan chain length), and functions differently starting from Line 10, which initializes the routing cost to be the routing constraint before the following loop that distributes care-bits (Lines 11-18). Both horizontal and vertical distribution manners (i.e., R_{scheme}) are tried in this loop and the one with the minimum cost without violating the partition size constraint is selected. At first, Line 12 generates the two temporary partitions P_{1_t} and P_{2_t} with the new care-bits group P_{temp} according to the current distribution manner. The procedure will break the loop and try the other distribution manner if the current one violates the size constraint (Line 13). Next we calculate the distribution cost as discussed in the above paragraph in Line 14, and if it is not larger than $R_{constraint}$, we select the distribution with smaller cost and update P_{groups}, N_{groups}, P_1 and P_2 (Lines 15-18). Once we hit the condition that both P_1 and P_2 do not change when trying to group the care-bits of the current pattern (Line 19), it means the current pattern needs to be applied at the entire circuit level and there is no need to try the rest of the low-power patterns. We hence distribute the unassigned scan-FFs based on the selected routing scheme, update P_1 and P_2 and then terminate the loop (Lines 20-22). Finally, Line 23 returns the two partitioned subcircuits.

By considering scan chain routing cost during both care-bits selection and iterative partitioning processes, the total scan chain length is expected to be significantly reduced.

Algorithm 3 - RA-IterativePartition

INPUT: $Pattern_{set}$, d_{size}, $R_{constraint}$
OUTPUT: P_1, P_2

1. Sort $Pattern_{set}$ in non-increasing power order;
2. Calculate $Size_{max} = \lceil \frac{N_{FF}}{2+d_{size}} \rceil$;
3. Initialize $N_{groups} = 1$; $P_{group\_1} = \emptyset$;
4. Initialize $C_{existing} = \emptyset$;
5. **for** $pattern_{max} \in Pattern_{set}$ {
6. $P_{temp} = IdentifyCareBits(pattern_{max}, C_{existing})$;
7. **for** i from 1 to N_{groups} {
8. **if** $(P_{temp} \bigcap P_{group\_i} \neq \emptyset)$ {
9. $P_{temp} = P_{temp} \bigcup P_{group\_i}$;
. }
. }
10. Initialize $R_{cost} = R_{constraint}$;
11. **for** $R_{scheme} \in \{HORIZONTAL, VERTICAL\}$ {
12. Generate P_{1_t} and P_{2_t} based on R_{scheme};
13. **if** $(|P_{1\_t}| > Size_{max}$ **OR** $|P_{2\_t}| > Size_{max})$ **continue**;
14. calculate R_{new};
15. **if** $(R_{new} \leq R_{cost})$ {
16. $R_{cost} = R_{new}$;
17. update P_{groups}, N_{groups};
18. $P_1 = P_{1\_t}$; $P_2 = P_{2\_t}$;
. }
. }
19. **if** (P_1 and P_2 do not change in this iteration) {
20. Distribute unassigned scan-FFs based on R_{scheme};
21. update P_1 and P_2;
22. **break**;
. }
. }
23. return P_1, P_2;

Figure 6. Procedure for routing-aware pattern-directed circuit virtual partitioning.

N_{PI}: Number of primary inputs; N_{PO}: Number of primary outputs; N_{FF}: Number of flip-flops; N_v: Number of test vectors;
P_{ct}: Maximum scan capture transitions of the original circuit; l_{orig}: Scan chain length of the original circuit; N_{FF_P1}: Number of flip-flops in subcircuit P_1;
N_{FF_P2}: Number of flip-flops in subcircuit P_2; N_{v_part}: Number of test patterns that can be applied within a partitioned subcircuit;
P_{ct_part}: Maximum scan capture transitions after partitioning; l_{part}: Scan chain length after partitioning;
$\Delta N_{v\_part} = \frac{N_{v\_part}}{N_v} \times 100\%$; $\Delta P_{ct\_part} = \frac{P_{ct\_part}}{P_{ct}} \times 100\%$; $\Delta l_{part} = \frac{l_{part}}{l_{orig}} \times 100\%$; $T_{run}(s)$: Computational time;

Circuit	N_{PI}	N_{PO}	N_{FF}	N_v	P_{ct}	l_{orig} (μm)	N_{FF_P1}	N_{FF_P2}	N_{v_part}	ΔN_{v_part} (%)	P_{ct_part}	ΔP_{ct_part} (%)	l_{part} (μm)	Δl_{part} (%)	$T_{run}(s)$
s9234	36	39	211	148	1321	6298.42	106	105	46	31.08	560	42.39	8102.82	128.65	1.27
s13207	62	152	638	242	1162	15175.82	322	316	75	30.99	519	44.66	21480.14	141.54	8.23
s15850	77	150	534	125	1972	13128.5	267	267	34	27.20	1004	50.91	22859.98	174.12	8.24
s35932	35	320	1728	16	12999	42984.48	884	844	6	37.50	7792	59.94	53029.68	123.37	12.57
s38417	28	106	1636	165	1537	39210.6	818	818	65	39.39	592	38.52	55920.48	142.62	50.31
s38584	38	304	1426	169	4983	43449.12	724	702	63	37.28	2192	43.99	62935.62	144.85	47.06
b17	37	97	1415	2421	3006	34357.40	710	705	293	12.10	1720	57.22	48161.95	140.18	84.29
b18	37	23	3320	4659	15990	92570.94	1661	1659	686	14.72	8857	55.39	121282.7	131.02	161.36
average										**28.78**		**49.13**		**142.52**	

Table 1. Experimental results for stuck-at tests without considering scan chain routing cost.

Circuit	N_{PI}	N_{PO}	N_{FF}	N_v	P_{ct}	l_{orig} (μm)	N_{FF_P1}	N_{FF_P2}	N_{v_part}	ΔN_{v_part} (%)	P_{ct_part}	ΔP_{ct_part} (%)	l_{part} (μm)	Δl_{part} (%)	$T_{run}(s)$
s9234	36	39	211	148	1321	6298.42	105	106	32	21.62	642	48.60	6519.48	103.51	1.12
s13207	62	152	638	242	1162	15175.82	319	319	50	20.66	559	48.11	17722.52	116.78	9.29
s15850	77	150	534	125	1972	13128.5	267	267	31	24.80	1094	55.48	14533.20	110.69	8.45
s35932	35	320	1728	16	12999	42984.48	864	864	4	25.00	9161	70.47	43307.22	100.75	13.28
s38417	28	106	1636	165	1537	39210.6	818	818	42	25.45	711	46.26	44542.08	113.60	54.34
s38584	38	304	1426	169	4983	43449.12	713	713	25	14.79	3419	68.61	48506.48	111.64	48.87
b17	37	97	1415	2421	3006	34357.40	707	708	187	7.72	1910	63.54	38817.90	112.98	87.11
b18	37	23	3320	4659	15990	92570.94	1660	1660	509	10.93	9640	60.29	99074.14	107.03	162.8
average										**18.87**		**57.67**		**109.62**	

Table 2. Experimental results for stuck-at tests when considering scan chain routing cost.

Circuit	N_{PI}	N_{PO}	N_{FF}	N_v	P_{ct}	l_{orig} (μm)	N_{FF_P1}	N_{FF_P2}	N_{v_part}	ΔN_{v_part} (%)	P_{ct_part}	ΔP_{ct_part} (%)	l_{part} (μm)	Δl_{part} (%)	$T_{run}(s)$
s9234	36	39	211	355	3215	6298.42	107	104	79	22.25	1921	59.75	8777.34	139.36	2.53
s13207	62	152	638	342	3378	15175.82	320	318	82	23.98	2056	60.86	23235.96	153.11	10.17
s15850	77	150	534	227	3583	13128.5	267	267	57	25.11	1996	55.71	20680.22	157.52	9.00
s35932	35	320	1728	79	19353	42984.48	864	864	11	13.92	11826	61.11	57361.26	133.45	18.91
s38417	28	106	1636	227	11102	39210.6	818	818	49	21.59	7811	70.36	52344.16	133.49	26.93
s38584	38	304	1426	428	10202	43449.12	724	702	16	3.74	7578	74.28	62408.5	143.64	21.66
b17	37	97	1415	1762	3968	34357.40	707	708	387	21.96	5298	66.76	44686.84	130.10	142.95
b18	37	23	3320	1037	18648	92570.94	1660	1660	173	16.68	11168	59.89	132083.60	142.78	168.2
average										**18.65**		**63.59**		**141.68**	

Table 3. Experimental results for broad-side tests without considering scan chain routing cost.

Circuit	N_{PI}	N_{PO}	N_{FF}	N_v	P_{ct}	l_{orig} (μm)	N_{FF_P1}	N_{FF_P2}	N_{v_part}	ΔN_{v_part} (%)	P_{ct_part}	ΔP_{ct_part} (%)	l_{part} (μm)	Δl_{part} (%)	$T_{run}(s)$
s9234	36	39	211	355	3215	6298.42	106	105	52	14.65	2021	62.86	6424.22	110.00	2.77
s13207	62	152	638	342	3378	15175.82	319	319	55	16.08	2370	70.16	16510.34	108.80	11.19
s15850	77	150	534	227	3583	13128.5	267	267	41	18.06	2227	62.15	15487.34	117.97	9.26
s35932	35	320	1728	79	19353	42984.48	864	864	9	11.39	12006	62.04	47392.84	110.26	19.98
s38417	28	106	1636	227	11102	39210.6	818	818	27	11.89	8043	72.45	47131.92	120.20	27.34
s38584	38	304	1426	428	10202	43449.12	713	713	11	2.57	7876	77.20	47829.1	110.08	23.11
b17	37	97	1415	1762	3968	34357.40	707	708	280	15.89	5609	70.68	36712.12	106.85	146.34
b18	37	23	3320	1037	18648	92570.94	1660	1660	144	13.89	12804	68.66	107751.16	116.40	173.11
average										**13.05**		**68.28**		**112.57**	

Table 4. Experimental results for broad-side tests when considering scan chain routing cost.

5 Experimental Results

To analyze the effectiveness of the proposed solution, experiments are carried out for several ISCAS'89 benchmark circuits and two big ITC'99 benchmark circuits. For ISCAS'89 circuits, the fully-specified test patterns used in our experiments are the low-capture-power patterns in [24] (for stuck-at tests) and [25] (for broad-side delay tests); while for ITC'99 circuits, they are generated with a commercial ATPG tool. In addition, we generate the scan cell placement

with a commercial physical design tool and make use of the algorithms in [2] to route our scan chains for scan chain routing cost comparison. Finally, the size difference constraint d_{size} for the partitioned subcircuits is set to be 15% in our experiments; while the weighting factor w in Eq. 1 is set as 200 to match the two cost factors. Since no other techniques targeting scan capture power reduction starts with fully-specified test patterns, the experimental results is compared with the original unpartitioned circuit only.

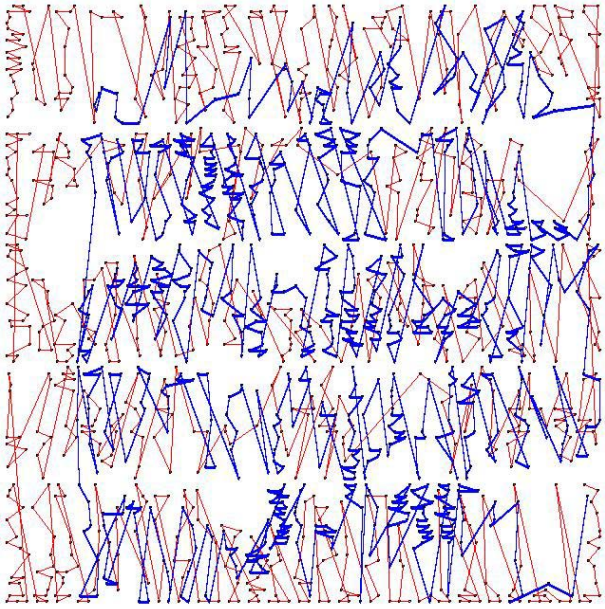
Tables 1, 2, 3 and 4 show the experimental results of our pattern-directed circuit partitioning algorithm with and without considering scan chain routing cost, for stuck-at faults and broad-side transition faults, respectively, in which N_{PI}, N_{PO}, N_{FF}, N_v, P_{ct}, and l_{orig} denote the number of primary inputs, the number of primary outputs, the number of flip-flops, the number of test vectors, the maximum scan capture transitions for the original circuit, and the scan chain length for the original circuit, respectively. N_{FF_P1} and N_{FF_P2} are the number of flip-flops in the two subcircuits $P1$ and $P2$ after virtual partitioning. N_{v_part}, P_{ct_part} and l_{part} are the number of test patterns that can be applied within a partitioned subcircuit, the maximum scan capture transitions after partitioning and the total scan chain length after partitioning, respectively. ΔN_{v_part}, ΔP_{ct_part} and Δl_{part} are computed as $\Delta N_{v\_part} = \frac{N_{v\_part}}{N_v} \times 100\%$, $\Delta P_{ct\_part} = \frac{P_{ct\_part}}{P_{ct}} \times 100\%$ and $\Delta l_{part} = \frac{l_{part}}{l_{orig}} \times 100\%$, respectively. Finally, $T_{run}(s)$ represents the computational time spent on the proposed iterative partitioning algorithm (including the care-bits selection procedure) using a Sun Blade 1000 workstation with 1GB memory, from which we can observe they are quite small even for the two large ITC'99 circuits.

As can be observed from Tables 1, for stuck-at faults, without considering scan chain routing cost, in average the proposed partitioning strategy is able to apply about 29 percent of high-power patterns within a partitioned subcircuit, and the number of scan capture transitions after partitioning is only about half of that of the original circuit (less than 40% for s38417). When considering scan chain routing cost, as shown in Table 2 the number of high-power patterns that can be applied within a subcircuit is reduced to be about 19% in average, but the scan capture transitions can still be kept less than 60% of that of the original circuit. For broadside tests, as we need to search two level fan-ins when identifying test stimuli care-bits, the number of care-bits for each test pattern is usually higher when compared to the one for stuck-at tests. As a result, the percentage of broadside patterns that are able to be applied within a partition is smaller, which is around 13% for the case that considers scan chain routing cost or is close to 19% without considering scan chain routing cost in average. The number of scan capture transitions after partitioning is less than 70% of that of the original circuit in both cases, which is still a significant improvement. From Tables 3 and 4, we can also observe that less than 4% of the broad-side patterns can be applied within a partitioned subcircuit for s38584. This is mainly because the high-power patterns in this particular circuit have a large number of care-bits even after applying reverse fault simulation in our design flow. However, at the same time, because these few problematic patterns have much larger scan capture transitions when compared to the other low-power patterns, we are still able to achieve around 25% test power reduction for s38584 in both cases.

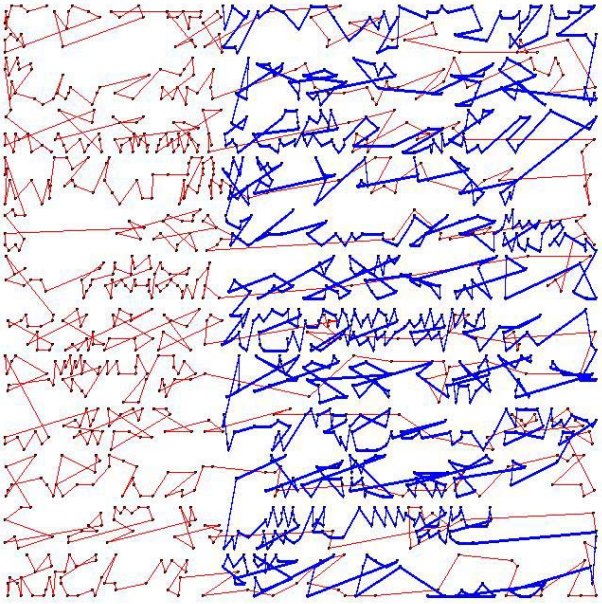
Although the proposed pattern-directed circuit virtual partitioning technique does not require to rerun ATPG and add test wrappers to ensure no fault coverage loss, it is not entirely cost-free, especially from the scan routing cost point of view. To compare the scan chain length before and after partitioning, we use the routing-constrained scan chain design algorithm in [2], which first divides the circuit into several sub-regions and routes them separately and then connect them in a "snake-like" way. We assume there exist two scan chains for the original circuit and each partitioned subcircuit contains a single scan chain. It can be observed from Tables 1-4 that, when compared to the unpartitioned circuit, in average the total scan chain length is incremented for about 40% using the proposed $IterativePartition$ algorithm without considering scan routing cost; while for the routing-aware algorithm $RA-IterativePartition$, it increases for only about 10%. Figures 7 and 8 compare the scan chain designs of s38417 for stuck-at tests and s38584 for broadside tests. As can be observed from these figures, when considering scan routing cost during virtual partitioning, most of the scan-FFs belonging to the same partitioned subcircuit are clustered in the same physical region, which effectively reduces the total scan chain length.

6 Conclusion

Partitioning the circuit into multiple subcircuits and test them separately is an effective technique to reduce test power consumption. In this paper, we propose to utilize the available test patterns to direct the circuit partitioning process, in such way that the faults in the glue logic between subcircuits are detected by patterns with low power dissipation applied at the entire circuit level, while the patterns with high power dissipation are applied within a partitioned subcircuit without loss of fault coverage. Since scan chains need to be built within partitioned subcircuits, we have also demonstrated how to effectively consider scan chain routing cost during the partitioning process. Experimental results on benchmark circuits show that the proposed technique is able to reduce the scan capture transitions significantly for both stuck-at tests and broadside delay tests at an acceptable DFT cost.

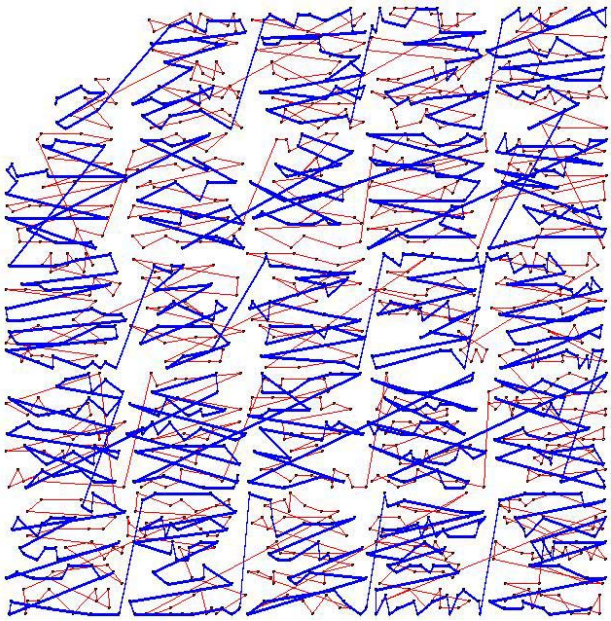


(a) with the *IterativePartition* algorithm

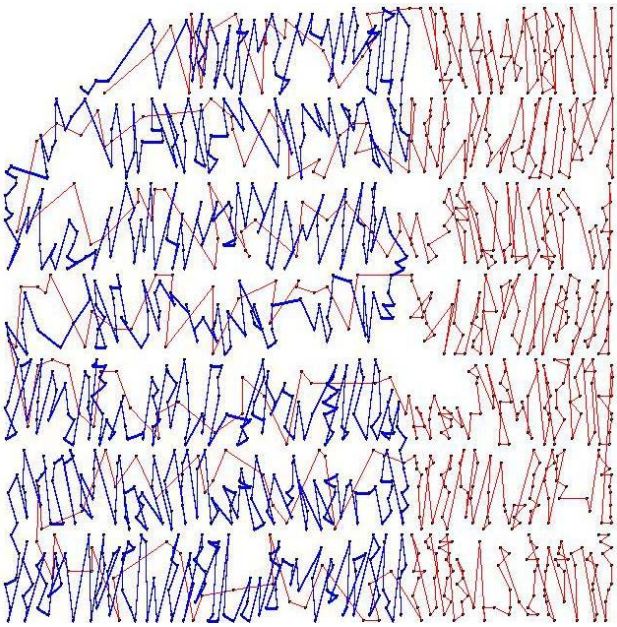


(b) with the $RA-IterativePartition$ algorithm

Figure 7. Comparison of the scan chain routing for stuck-at tests of s38417.



(a) with the *IterativePartition* algorithm



(b) with the $RA-IterativePartition$ algorithm

Figure 8. Comparison of the scan chain routing for broad-side tests of s38584.

7 Acknowledgement

The authors wish to thank Dr. Kohei Miyase and Dr. Xiaoqing Wen from Kyushu Institute of Technology for providing the test patterns in [24, 25] and Dr. Yu Hu from Chinese Academy Science for generating the benchmark circuits' scan cell placement.

References

[1] Y. Bonhomme, P. Girard, L. Guiller, C. Landrault, and S. Pravossoudovitch. A Gated Clock Scheme for Low Power Scan Testing of Logic ICs or Embedded Cores. In *Proceedings IEEE Asian Test Symposium (ATS)*, pages 253–258, Nov. 2001.

[2] Y. Bonhomme, P. Girard, L. Guiller, C. Landrault, and S. Pravossoudovitch. Power-Driven Routing-Constrained Scan Chain Design. *Journal of Electronic Testing: Theory and Applications*, 20(6):647–660, December 2004.

[3] K. M. Butler, J. Saxena, A. Jain, T. Fryars, J. Lewis, and G. Hetherington. Minimizing Power Consumption in Scan Testing: Pattern Generation and DFT Techniques. In *Proceedings IEEE International Test Conference (ITC)*, pages 355–364, Oct. 2004.

[4] S. Chakravarty and V. Dabholkar. Two Techniques for Minimizing Power Dissipation in Scan Circuits During Test Application. In *Proceedings IEEE Asian Test Symposium (ATS)*, pages 324–329, 1994.

[5] R. M. Chou, K. K. Saluja, and V. D. Agrawal. Scheduling tests for VLSI systems under power constraints. *IEEE Transactions on VLSI Systems*, 5(2):175–184, June 1997.

[6] F. Corno, P. Prinetto, M. Rebaudengo, and M. Sonza-Reorda. A test pattern generation methodology for low power consumption. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 453–460, 1998.

[7] V. Dabholkar, S. Chakravarty, I. Pomeranz, and S. M. Reddy. Techniques for minimizing power dissipation in scan and combinational circuits during test application. *IEEE Transactions on Computer-Aided Design*, 17(12):1325–1333, December 1998.

[8] P. Girard. Survey of Low-Power Testing of VLSI Circuits. *IEEE Design & Test of Computers*, 19(3):80 – 90, May-June 2002.

[9] P. Girard, L. Guiller, C. Landrault, and S. Pravossoudovitch. Low Power BIST Design by Hypergraph Partitioning: Methodology and Architectures. In *Proceedings IEEE International Test Conference (ITC)*, pages 652–661, October 2000.

[10] P. Girard, C. Landrault, S. Pravossoudovitch, and D. Severac. Reducing power consumption during test application by test vector ordering. In *Proceedings International Symposium on Circuits and Systems (ISCAS)*, pages 296–299, 1998.

[11] V. Iyengar and K. Chakrabarty. Precedence-Based, Preemptive, and Power-Constrained Test Scheduling for System-on-a-Chip. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 368–374, Marina del Rey, CA, May 2001.

[12] K. Miyase and S. Kajihara. XID: Don't Care Identification of Test Patterns for Combinational Circuits. *IEEE Transactions on Computer-Aided Design*, 23(2):321–326, February 2004.

[13] N. Nicolici and B. M. Al-Hashimi. *Power-Constrained Testing of VLSI Circuits*. Kluwer Academic Publishers, 2003.

[14] B. Pouya and A. L. Crouch. Optimization trade-offs for vector volume and test power. In *Proceedings IEEE International Test Conference (ITC)*, pages 873–881, Oct. 2000.

[15] S. Remersaro, X Lin, Z Zhang, S.M. Reddy, I. Pomeranz and J. Rajski. Preferred Fill: A Scalable Method to Reduce Capture Power for Scan Based Designs. In *Proceedings IEEE International Test Conference (ITC)*, Oct. 2006.

[16] P. M. Rosinger, B. M. Al-Hashimi, and N. Nicolici. Scan Architecture with Mutually Exclusive Scan Segment Activation for Shift- and Capture-Power Reduction. *IEEE Transactions on Computer-Aided Design*, 23(7):1142–1153, October 2004.

[17] R. Sankaralingam, R. R. Oruganti, and N. A. Touba. Static Compaction Techniques to Control Scan Vector Power Dissipation. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 35–40, 2000.

[18] R. Sankaralingam, B. Pouya, and N. A. Touba. Reducing Power Dissipation During Test Using Scan Chain Disable. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 319–324, 2001.

[19] R. Sankaralingam and N. A. Touba. Controlling Peak Power During Scan Testing. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 153–159, 2002.

[20] J. Saxena, K. M. Butler, V. B. Jayaram, and S. Kundu. A Case Study of IR-Drop in Structured At-Speed Testing. In *Proceedings IEEE International Test Conference (ITC)*, pages 1098–1104, Oct. 2003.

[21] N. Selvakkumaran and G. Karypis. Multi-Objective Hypergraph Partitioning Algorithms for Cut and Maximum Subdomain Degree Minimization. In *Proceedings International Conference on Computer-Aided Design (ICCAD)*, pages 726–733, 2003.

[22] S. Wang and S. K. Gupta. ATPG for heat dissipation minimization during test application. *IEEE Transactions on Computers*, 47(2):256–262, February 1998.

[23] X. Wen, et al. A New ATPG Method for Efficient Capture Power Reduction during Scan Testing. In *Proceedings IEEE VLSI Test Symposium (VTS)*, May 2006.

[24] X. Wen, et al. On Low-Capture-Power Test Generation for Scan Testing. In *Proceedings IEEE VLSI Test Symposium (VTS)*, pages 265–270, May 2005.

[25] X. Wen, et al. Low-Capture-Power Test Generation for Scan-Based At-Speed Testing. In *Proceedings IEEE International Test Conference (ITC)*, pages 1019–1028, November 2005.

[26] L. Whetsel. Adapting Scan Architectures for Low Power Operation. In *Proceedings IEEE International Test Conference (ITC)*, pages 863–872, Oct. 2000.

[27] Q. Xu and N. Nicolici. Resource-Constrained System-on-a-Chip Test: A Survey. *IEE Proceedings, Computers and Digital Techniques*, 152(1):67–81, January 2005.