A Technique to Reduce Peak Current and Average Power Dissipation in Scan Designs by Limited Capture

Seongmoon Wang Wenlong Wei
NEC Labs., America, Princeton, NJ
{swang,wwei}@nec-labs.com

Abstract— In this paper, a technique that can efficiently reduce peak and average switching activity during test application is proposed. The proposed method does not require any specific clock tree construction, special scan cells, or scan chain reordering. Test cubes generated by any combinational ATPG can be processed by the proposed method to reduce peak and average switching activity without any capture violation. Switching activity during scan shift cycles is reduced by assigning identical values to adjacent scan inputs and switching activity during capture cycles is reduced by limiting the number of scan chains that capture responses. Hardware overhead for the proposed method is negligible. The peak transition is reduced by about 40% and average number of transitions is reduced by about 56-85%. This reduction in peak and average switching activity is achieved with no decrease in fault coverage.

I. Introduction

Scan (full or partial) is widely used as a de facto standard design-for-testability technique to develop high quality test patterns for complex sequential circuits in short test development time. Finite state machines are often implemented in such a manner that patterns representing successive states are highly correlated to reduce transitions between successive clock cycles. Using scan allows the automatic test equipment (ATE) to apply any pattern to the state inputs during test application and hence correlation between patterns that are consecutively applied to state inputs decreases. Furthermore, since each test pattern is applied through scan chains by a series of shift operations, the test pattern applied at the state inputs of the circuit at the current cycle represents the shifted values of the test pattern that was applied at the previous cycle. This decreases correlation between test patterns that are applied at consecutive cycles. Excessive switching activity due to low correlation between consecutive test patterns can cause several problems.

Since heat dissipation in a CMOS circuit is proportional to switching activity, excessive switching activity can permanently damage the circuit under test (CUT). High temperature can hurt circuit's reliability by accelerating metal migration. High switching activity also causes large IR drop, which is given by $I \times R$, where I is current flow and R is resistance of power rail. To test a bare die, power must be supplied during test application through probes that typically have higher inductance than power and ground pins of a circuit package. Hence, the bare die under test will experience higher power/ground noise that is given by Ldi/dt, where L is the inductance of power and ground lines and di/dt is the rate of change of current flowing in power and ground lines. Power/ground noise exacerbates voltage drop during test application. In consequence, excessive power/ground noise can erroneously change logic states of circuit lines causing some good dies to fail the test, leading to unnecessary loss of yield.

If the number of scan elements in the longest scan chain is m, then a complete scan input pattern is loaded into scan chains through m cycles of scan shift operations. Scan inputs can continuously have transitions during shift cycles. Then scan elements are configured into their normal mode to capture the response to the scanned in pattern. Hence a capture cycle occurs every $m + 1$ cycles, where $m \gg 1$. Hence switching activity during capture cycles does not significantly contribute to increase in the chip temperature. However, as described above, excessive switching activity can cause high power/ground noise, which increases signal propagation delay and can even flip logic states of circuit lines.

In this paper, a technique that can effectively reduce peak current and average heat dissipation during test application is presented. Switching activity during shift cycles is reduced by assigning identical values to adjacent scan inputs and peak current during capture cycles is reduced by limiting the number of scan chains that capture responses. Even though excessive switching activity during test application has been addressed in the industry, commercial automatic test pattern generator (ATPG) tools that reduce switching activity during test application are not widely used and most test patterns are still generated by regular ATPG tools that do not consider switching activity. The technique presented in this paper can post-process test patterns generated by any ATPG tool to reduce switching activity during test application. The proposed technique does not require changing existing scan chain structure.

The rest of this paper is organized as follows. Section II introduces prior work. Section III describes techniques to reduce transitions during shift cycles. Techniques to reduce transitions during capture cycles are described in Section IV. Experimental results are given in Section V. Section VI has conclusions.

II. Prior Work

A number of papers have been published to tackle the problem of excessive switching activity during test application. Most papers focus on reducing average power dissipation and

1-4244-0630-7/07/$20.00 ©2007 IEEE.

cannot handle peak current (especially during capture cycles). Test pattern generators for built-in self-test that can reduce peak current during shift cycles are proposed in [1, 4, 10]. However, these techniques cannot be used in deterministic testing environment, where test patterns are generated by an ATPG or manually. Techniques to reduce peak current during capture cycles in deterministic testing environment are proposed in [5, 6, 7, 8, 11]. The techniques proposed in [5, 6, 7] can reduce switching activity during capture cycles as well as scan shift cycles. In [5, 6, 8], each scan chain is partitioned into p scan sub-chains and each scan sub-chain c_i is driven by a separate sub-clock clk_i, where $i = 1, 2, \ldots, p$. In each clock cycle, only one sub-clock among the p sub-clocks is activated. Therefore only the scan flip-flops that are driven by the activated sub-clock can cause transitions. Typically, clock trees are globally optimized for all clock domains to minimize area and clock skew. Hence, if clock trees are constructed only to reduce switching activity during test application, the resulting clock trees may not meet clock skew requirement or be suboptimal.

If scan sub-chains capture in sequence, the part of test pattern loaded in a set of scan sub-chains that captures at an earlier phase is corrupted (overwritten) by the captured response. This problem, which occurs due to capturing scan chains in sequence, is referred to as *capture violation* [5]. To cope with the capture violation problem, [6] identifies dependency relations between scan flip-flops and constructs strongly connected graphs (SCGs) according to the dependency relations. All scan flip-flops in an SCG should capture in the same sub-clock cycle not to cause capture violation. If there are many scan flip-flops that talk to each other, there will be several large SCGs. Large SCGs are broken by replacing selected scan flip-flops by special flip-flops that can hold two bits. This incurs additional area overhead and may entail performance degradation. Scan flip-flops are reordered to minimize the number of special flip-flops. This may entail large routing overhead. Instead of using special scan flip-flops, [5] resolves the capture violation problem by ATPG techniques. In order to avoid capture violations, the (sequential) ATPG should be able to handle p different time frames, where p is the number of sub-clocks, even for combinational faults such as stuck-at faults. If there are many sub-chains, i.e., $p \gg 1$, then it will be very difficult to achieve high fault coverage and test generation time will be very long because the ATPG should handle large number of sequential depths. Moreover since specifying a single input in a later time frame may require specifying large number of scan inputs in the first time frame, this technique increases the size of test set. In [8], all scan sub-chains capture at the same time in every capture cycle to avoid any capture violation. Hence [8] cannot reduce peak current during capture cycles.

The ATPG-based method proposed in [11] reduces peak current during capture cycles for circuits with regular scan chains. Their ATPG exploits don't cares (X's) that exist in test patterns to reduce instantaneous current during capture cycles. The achieved reduction in peak current during capture cycles is not significant. Furthermore, since [11] reduces peak current only during capture cycles, it requires an additional technique to reduce switching activity during shift cycles. The technique proposed in [7] assigns X's that exist in pre-computed test cubes to reduce peak current during test application. A *test cube* is a test pattern that is not fully specified. Since X's in test cubes are assigned to reduce switching activity during all different types of test cycles: cycles to load a test pattern into the scan chains, cycles to capture the response to the test pattern, and cycles to unload the response, achieving enough reduction in peak current is very difficult. (Assigning X's to reduce switching activity during capture cycles can increase switching activity during shift cycles or vice versa.)

The proposed method requires no special clock trees or scan flip-flops. Further, it does not require reordering scan flip-flops, which may increase routing overhead. The only modification required by the proposed method to the existing design is controlling scan chains by multiple scan enable signals rather than a single scan enable signal. Unlike [7] where don't cares that exist in test cubes are utilized to reduce switching activity during both scan shift cycles and capture cycles, in this paper, don't cares in test cubes are utilized to reduce switching activity only during shift cycles and switching activity during capture cycles is reduced by capturing limited number of scan chains. Hence the proposed method can achieve larger reduction in peak and average switching activity. Since the proposed method does not use sequential capture, which causes capture violations, capture violations never occur in the proposed method. To our best knowledge, no previously published papers reduce peak current during capture cycles by capturing limited number of scan chains. Instead of generating test patterns by a special ATPG that can reduce switching activity, the proposed method post-processes test patterns to reduce switching activity during test application and hence is applicable to test patterns generated by any ATPG tool.

III. Reducing Transitions during Shift Cycles

In this paper we assume that the sequential CUT is implemented in CMOS and employs full-scan. Even though the proposed technique can be extended to level sensitive scan latch design (LSSD) with a few modifications, we assume that scan chains are constructed with muxed scan flip-flops.

Typically, a test cube generated by an ATPG tool has large number of X's. The faults that a test cube targets can be detected independent of the binary values assigned to those X's. Assigning identical binary values to the adjacent scan inputs that are assigned X's can reduce transitions at scan inputs during shift cycles. This technique is commonly used to reduce switching activity during test application [7, 9]. Since test pattern compaction merges several test cubes into one test cube [3], highly compacted test cubes can have large numbers of care bits. If test cubes do not have enough X's, significant reduction in switching activity may not be achieved by filling X's with identical binary values. If a test cube t_j does not have enough care bits, then we reverse-compact t_j into two test cubes t_{j1} and t_{j2} each of which has far fewer care bits to increase numbers of X's. In order to minimize the number of

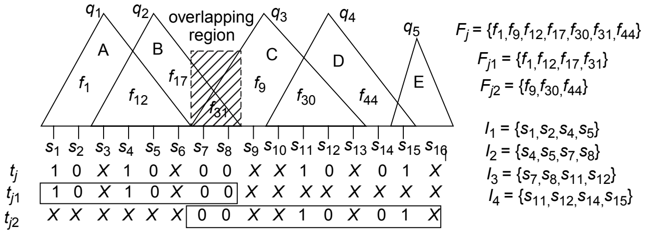
q_1
q_2
overlapping region
q_3
q_4
q_5
A
B
C
D
E
f_1
f_{12}
f_{17}
f_{31}
f_9
f_{30}
f_{44}
$F_j = \{f_1, f_9, f_{12}, f_{17}, f_{30}, f_{31}, f_{44}\}$
$F_{j1} = \{f_1, f_{12}, f_{17}, f_{31}\}$
$F_{j2} = \{f_9, f_{30}, f_{44}\}$
$I_1 = \{s_1, s_2, s_4, s_5\}$
$I_2 = \{s_4, s_5, s_7, s_8\}$
$I_3 = \{s_7, s_8, s_{11}, s_{12}\}$
$I_4 = \{s_{11}, s_{12}, s_{14}, s_{15}\}$
s_1 s_2 s_3 s_4 s_5 s_6 s_7 s_8 s_9 s_{10} s_{11} s_{12} s_{13} s_{14} s_{15} s_{16}
t_j 1 0 X 1 0 X 0 0 X X 1 0 X 0 1 X
t_{j1} 1 0 X 1 0 X 0 0 X X X X X X X X
t_{j2} X X X X X X 0 0 X X 1 0 X 0 1 X

Fig. 1. Reverse-compacting a Test Cube

care bits in t_{j1} and t_{j2}, the following should be satisfied. First, numbers of care bits in the partitioned test cubes t_{j1} and t_{j2} should be balanced, i.e., the number of care bits in t_{j1} should be close to that of care bits in t_{j2}. Second, the overlap of care bits between t_{j1} and t_{j2} should be minimized. In other words, if input s_i is specified in t_{j1}, then it should not be specified in t_{j2}, and vice versa. Note that the bi-partitioned test cubes t_{j1} and t_{j2} together should be able to detect all the faults that are detected by the original test cube t_j. The proposed reverse-compaction procedure is described in the following.

1. Identify a set of all outputs Q_j at which at least one fault in the set of faults that are detected by test cube t_j, which is denoted by F_j, is observed.
2. For every output q_a in Q_j, find the set of inputs I_a that are in the fanin cone of output q_a and specified in t_j.
3. Initialize all bits in two test cubes t_{j1} and t_{j2} with X's (note that both t_{j1} and t_{j2} have the same width as t_j).
4. Select output q_{first} at which the largest number of faults in F_j are observed and mark the faults that are observed at q_{first} from F_j. Remove q_{first} from Q_j. For every input s_i in I_{first}, which drives q_{first} (note that all inputs in I_{first} are specified), if s_i is assigned v, where $v = 0$ or 1, in t_j, set the corresponding input s_i to v in t_{j1}. Remove I_{first}.
5. If the number of care bits in t_{j1} is greater than that of care bits in t_{j2}, then $t_d \leftarrow t_{j2}$ and $t_o \leftarrow t_{j1}$. Otherwise $t_d \leftarrow t_{j1}$ and $t_o \leftarrow t_{j2}$. Select an output q_b from Q_j whose input set I_b contains the fewest inputs that are already specified in t_o among all outputs. This will minimize the overlap in specified bits between t_{j1} and t_{j2}. Remove q_b from Q_j and I_b and mark all faults from F_j that can be observed at the selected output q_b. For every scan input s_i in I_b, if s_i is assigned a binary value v in t_j, then set the corresponding input s_i to v in t_d. If there are no unmarked faults in F_j, then exit. Repeat Step 5.

Example 1: The example circuit shown in Figure 4 consists of 5 circuit cones, A, B, C, D, and E, and 16 scan inputs, $s_1, s_2, \ldots, s_{16}$. Assume that test cubes that have more than 6 specified bits are reverse-compacted to increase the number of don't cares. Hence test cube t_j shown in Figure 4, which has 10 specified bits, is now reverse-compacted. Faults $f_1, f_9, f_{12}, f_{17}, f_{30}, f_{31}$, and f_{44} can be detected by t_j and these faults can be observed at outputs q_1, q_2, q_3, and q_4. Hence $Q_j = \{q_1, q_2, q_3, q_4\}$. The list of inputs that drive output q_1 and are specified in test cube t_j is $I_1 = \{s_1, s_2, s_4, s_5\}$. Similarly, $I_2 = \{s_4, s_5, s_7, s_8\}$, $I_3 = \{s_7, s_8, s_{11}, s_{12}\}$, and $I_4 = \{s_{11}, s_{12}, s_{14}, s_{15}\}$. First, t_{j1} and t_{j2} are both initialized to $XX...X$. The output at which the most faults can be observed is selected first. q_2 and q_3 can observe the most faults, 3 faults. Assume q_2 is selected. q_2 is removed from Q_j, i.e., $Q_j = \{q_1, q_3, q_4\}$. The faults that can be observed at q_2, f_{12}, f_{17}, and f_{31}, are marked in F_j. Since q_2 is selected, the values that are assigned to $I_2 = \{s_4, s_5, s_7, s_8\}$ in t_j are copied to t_{j1} to update t_{j1} to $XXX10X00X...X$. Then q_2 and I_2 are removed. Since the number of specified bits in t_{j1}, 4, is greater than that of specified bits in t_{j2}, 0, $t_d \leftarrow t_{j2}$ and $t_o \leftarrow t_{j1}$. Next we choose q_4 since I_4 contains no inputs that are specified in $t_o (= t_{j1})$. The values that are assigned to $I_4 = \{s_{11}, s_{12}, s_{14}, s_{15}\}$ in t_j are copied to $t_d (= t_{j2})$ to update it to $X...XX10X01X$. The faults that can be observed at q_4, f_{30} and f_{44}, are marked (now only faults f_1 and f_9 are not marked in F_j). q_4 and I_4 are removed. Since the number of specified bits in t_{j1} is not greater than that of specified bits in t_{j2}, $t_d \leftarrow t_{j1}$ and $t_o \leftarrow t_{j2}$. Since I_1 contains no inputs that are specified in t_o, I_1 is selected next. $t_d \leftarrow t_{j1}$ is updated to $10X10X00XX...X$ by coping the values assigned to $I_1 = \{s_1, s_2, s_4, s_5\}$ in t_j to t_{j1}. Fault f_1, which can be observed at q_1, is marked in F_j (note that the other fault f_{12} that can be observed at q_1 is already marked). q_1 and I_1 are removed. Since the number of specified bits in t_{j1} is greater than that of specified bits in t_{j2}, $t_d \leftarrow t_{j2}$. Since only I_3 remains, the values assigned to $I_3 = \{s_7, s_8, s_{11}, s_{12}\}$ in t_j are copied to $t_d (= t_{j2})$ to make t_{j2} $X...X00XX10X01X$. After I_3 and q_3 are removed, there are no more outputs left in Q_j. Now two bi-partitioned test cubes $t_{j1} = 10X10X00XX...X$ and $t_{j2} = X...X00XX10X01X$ are obtained. □

If either of the bi-partitioned test cubes has still too many specified bits, then the test cube is further divided into another pair of test cubes. This is repeated until the number of specified bits in every test cube in the set is smaller than a predefined number. According to our extensive experiments (see Table I), very few test cubes need to be reverse-compacted even if highly compacted test cubes are used. Hence increase in test set sizes due to reverse-compaction is not significant.

Note that the proposed technique modifies only test patterns, i.e., input stimuli, to reduce switching activity during scan shift cycles. Once the response to a test pattern is captured, then the response captured in scan flip-flops is scanned out by a series of shift operations, causing transitions at scan inputs. Even though the proposed method does not modify responses to reduce switching activity, the transitions caused by responses being scanned out are automatically taken care of for the following reason. In the proposed method, only selected scan chains capture responses in each capture cycle (see Section IV) and the other scan chains hold test patterns, which were already modified for low switching activity during shift cycles. Since only the selected scan chains scan out responses and the other scan chains scan out the test patterns modified for low switching activity during shift cycles, the number of transitions caused by responses being scanned out is small.

IV. Reducing Transitions during Capture Cycles

In [6, 5, 8], since scan chains capture in sequence, their main concerns are to avoid capture violation. In contrast, in the pro-

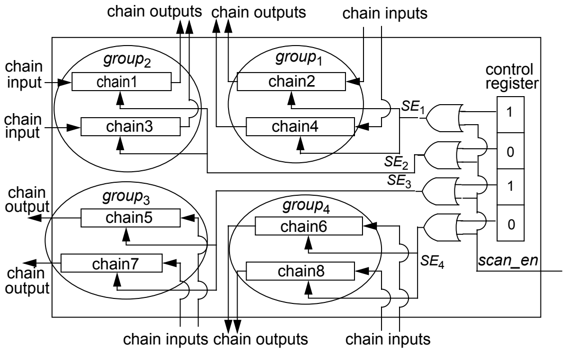

Fig. 2. Scan Architecture of the Proposed Method

posed method, capture violations never occur. However, since only limited number of scan chains capture in each capture cycle, the major concern of the proposed method is to minimize decrease in fault coverage (or increase in test pattern count if additional test patterns are applied to make up for the decrease in fault coverage) due to observing only part of responses. In short, although they may look similar, the proposed method and [5, 6, 8] are fundamentally different in nature.

In this paper, scan chains in the circuit are first clustered into G groups (clustering does not require modification of existing scan structure and is independent of test patterns). A group scan enable signal SE_i is connected to all scan chains that belong to the same group $group_i$. In each capture cycle, scan chains in the selected C scan chain groups, where $C \leq G$, capture the response and scan chains in the other $G - C$ groups continue shift operations. The C groups of scan chains that are selected to capture in the capture cycle for a test pattern is called the *capture groups* for the test pattern. Since only selected scan chains capture in each capture cycle and the scan chains that are not selected continue shifting the test pattern, which is modified to reduce switching activity during shift cycles (see Section III), we can reduce switching activity during capture cycles.

Figure 2 illustrates an example scan architecture implementing the proposed technique. Scan chains in each group $group_i$ are controlled by group scan enable signal SE_i, where $i = 1, 2, 3,$ and 4. If the i-th bit of the control register is loaded with 1, then the corresponding scan enable signal SE_i stays at 1 even in capture cycles (when the external scan enable signal $scan\_en$ is set to 0). Since the capture control register is loaded with 1010 in the example shown in Figure 2, group scan enable signals SE_1 and SE_3 are always 1 and scan chains only in $group_2$ and $group_4$ capture in the capture cycle, i.e., $group_2$ and $group_4$, are the capture groups. The constituent flip-flops of the control register can be distributed across the chip to minimize routing for the group scan enable signals.

Since only a limited number of scan chains capture during each capture cycle, some fault effects that could be detected if all scan chains capture the response in every capture cycle may not be observed. This may result in decrease in fault coverage. However if we select capture groups for each test pattern carefully, then we can minimize or eliminate decrease in fault coverage due to not capturing all scan chains. In the following, we present an efficient algorithm to select capture groups that always guarantees the same fault coverage that can be achieved by capturing all scan chains.

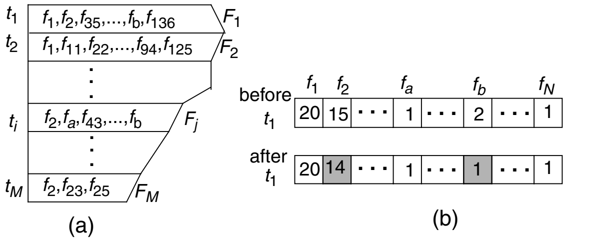

Fig. 3. (a) n-detection Target Fault Lists (b) Detection Count Table

First, we assign X's in test cubes to minimize switching activity during shift cycles (see Section III). Now all test patterns are fully specified. Next we conduct n-detection fault simulation with these fully specified test patterns and identify a set of faults detected by each test pattern t_i, where $i = 1, 2, \ldots, M$, where M is the number of test patterns in the set. A detection count dc_j is assigned to each fault f_j, where $j = 1, 2, \ldots, N$, where N is the number of faults in the fault list. If fault f_j is detected by test pattern t_i during the fault simulation, dc_j is incremented by 1. The set of faults that are detected by test pattern t_i is called the *target fault list* of t_i and denoted by F_i. If we detect all faults in the target fault list F_i of every test pattern t_i, where $i = 1, 2, \ldots, M$, by capturing only limited number of scan chains, we can achieve the same fault coverage that can be achieved by observing all scan chains.

Figure 3 (a) shows example target fault lists for M test patterns after 20-detection fault simulation. Figure 3 (b) shows the corresponding detection count table for the faults in the fault lists shown in Figure 3 (a). The detection count table shows that f_1 is detected by 20 or more test patterns. On the other hand, faults f_a and f_N are detected by only one test pattern ($dc_a = dc_N = 1$) and f_b is detected by only 2 test patterns ($dc_b = 2$). Faults that are detected by only one test pattern are called *single detection faults* and faults that are detected by only two test patterns are called *double detection faults*.

For each test pattern t_i, where $i = 1, 2, \ldots, M$, we determine the capture groups. If we cluster scan chains in the circuit into G groups and select C scan chain groups, then there are $\binom{G}{C}$, where $\binom{G}{C} = \frac{G\times(G-1)\times\ldots\times(G-C+1)}{C\times(C-1)\times\ldots\times 1}$, different combinations of C scan chain groups to choose. If f_j is a single detection fault, then there is only one test pattern in the entire test set that detects the fault. Hence, in order not to lose fault coverage, all single detection faults in the target fault list of every test pattern should be captured in the selected C groups.

Larger G gives more combinations of capture groups even if the same fraction of scan chain groups are selected to capture. For example, if $G = 12$ and 1/3 of scan chain groups are selected to capture in each capture cycle, i.e., $C = 4$, then there are $(12 \times 11 \times 10 \times 9)/(4 \times 3 \times 2 \times 1) = 495$ different combinations of capture groups to choose. In contrast, if $G = 6$ and also 1/3 of scan chains are selected to capture, then there are only $6 \times 5/2 = 15$ combinations of capture groups to choose. If there are more combinations of capture groups to

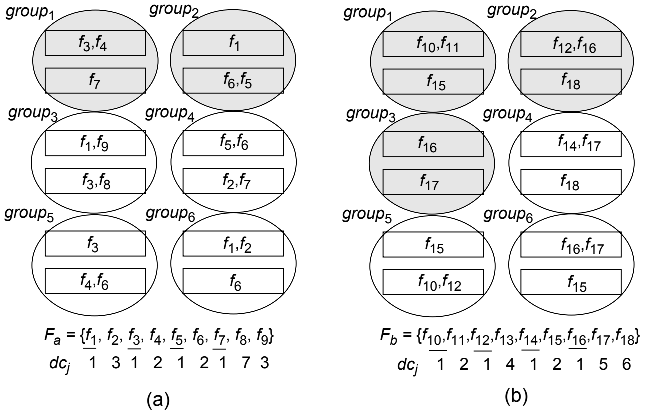

Fig. 4. Selecting Capture Groups

choose, then there will be obviously higher chance that at least one combination of capture groups detects all single detection faults for the test pattern. Hence, large G is more favorable than small G. However, large G increases time complexity of the procedure to select capture groups due to the large search space. According to our experiments, the number of groups need not be more than 12 for any design. Hence hardware overhead for the control register and extra data for the control register to be stored in the ATE are negligible.

If there are more than one combination of C scan chain groups for a test pattern that can detect all single detection faults in its target fault list, then the combination of C scan chain groups that can detect the most double detection faults is selected. If a double detection fault is not selected for observation, then it becomes a single detection fault. For example, in Figure 3 (b), originally fault f_b is a double detection fault. However if the capture groups selected for t_1, which is one of the two test patterns that can detect f_b, include no scan chains that can capture fault effects of f_b, then f_b becomes a single detection fault.

Example 2: Figure 4 illustrates the algorithm for selecting capture groups for test patterns. The circuit has 12 scan chains, which are clustered into 6 groups. The scan chains are denoted by rectangles and the faults that can be captured in each scan chain are shown inside the corresponding rectangle. Figure 4 (a) gives the faults that can be detected by test pattern t_a; the target fault list for t_a is $F_a = \{f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9\}$. dc_j, where $j = 1, 2, \ldots, 9$, below F_a are the detection counts for the faults in F_a. The detection counts show that f_1, f_3, f_5, and f_7 (underscored) are single detection faults and f_4 and f_6 are double detection faults. Assume that we can capture two capture groups in each capture cycle without exceeding peak current limit, i.e., $C = 2$. Selecting $group_1$ and $group_2$ as the capture groups for t_a can observe $f_1, f_3, f_4, f_5, f_6, f_7$, detecting all the 4 single detection faults and also the 2 double detection faults. Selecting $group_3$ and $group_4$ as the capture groups can also detect all single detection faults for t_a but detects only one double detection fault, f_6, we select $group_1$ and $group_2$ as the capture groups for t_a. □

For some test patterns, it may not be possible to detect all single detection faults in their target fault lists by capturing any combination of C scan chain groups. However capturing a few more scan chain groups in addition to the C scan chain groups may make all single detection faults observed. Let $peak\_limit$ be the maximum number of transitions allowed for a test pattern at any shift or capture cycle. The number of transitions that occur when extra scan chain groups capture should not exceed $peak\_limit$. If capturing only C scan chain groups cannot detect all single detection faults for test pattern t_i, then we temporarily select some extra scan chain groups in addition to the C scan chain groups to detect all single detection faults. Then we compute the number of transitions to be caused by t_i during the capture cycle and all shift cycles when those temporarily selected scan chain groups and the C scan chain groups are allowed to capture. If the number of transitions does not exceed $peak\_limit$ at the capture cycle or any shift cycle for t_i (since more than C scan chain groups capture responses, the number of transitions caused by the response being scanned out can be larger than capturing only C scan chain groups), then we permanently select those extra scan chain groups along with the C scan chain groups and capture them during the real capture cycle for t_i. Otherwise, we search a combination of $2C$ scan chain groups that detect all single detection faults in the target fault list of t_i. During test application, t_i is applied to the scan chains twice and, in each of the two capture cycles, different C scan chain groups are selected to capture. If this still does not detect all single detection faults in the target fault list, then we select extra scan chain groups in addition to the $2C$ scan chain groups to detect all single detection faults. If this exceeds $peak\_limit$ in any test cycle, then we select all $3C$ scan chain groups (since we always choose $3C \geq G$, capturing $3C$ scan chain groups detects all single detection faults). The test pattern is applied to the scan chains three times capturing different C scan chain groups in each of the 3 capture cycles. The flow chart shown in Figure 5 summarizes the overall algorithm for selecting capture groups.

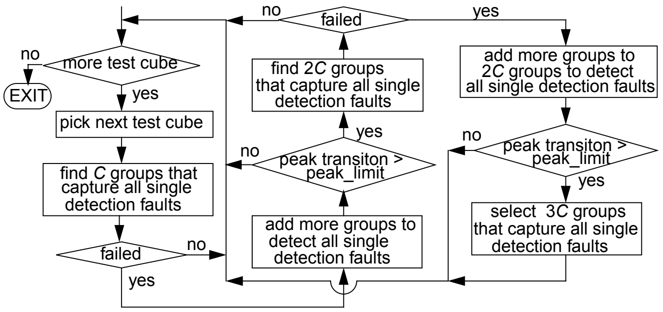
no
failed
yes
no
more test cube
EXIT
yes
pick next test cube
find C groups that capture all single detection faults
failed
no
yes
find 2C groups that capture all single detection faults
yes
no
peak transiton > peak_limit
add more groups to detect all single detection faults
add more groups to 2C groups to detect all single detection faults
no
peak transiton > peak_limit
yes
select 3C groups that capture all single detection faults

Fig. 5. Flow Chart for Capture Group Selection

Example 3: Figure 4 (b) gives the target fault list $F_b = \{f_{10}, f_{11}, f_{12}, f_{13}, f_{14}, f_{15}, f_{16}, f_{17}, f_{18}\}$ for test pattern t_b and faults each scan chain can capture. F_b has 4 single detection faults, $f_{10}, f_{12}, f_{14}, f_{16}$. None of combinations of two scan chain groups can capture all single detection faults for t_b. However, capturing 3 groups, $group_1, group_2$, and $group_3$, can observe all double detection faults as well as all single detection faults. Now we check by simulation if capturing $group_1, group_2$, and $group_3$ exceeds $peak\_limit$ at any scan shift cycle or capture cycle. If the number of transitions computed by simulation does not exceed $peak\_limit$ in any scan shift or capture cycle, then the scan chains in those three groups will capture the response during the capture cycle for t_b. Otherwise, $2C$ groups (since $C = 2$, $2C$ gives 4 groups) are selected as the capture groups for test pattern t_b. During test application, the same test pattern t_b will be applied to the circuit twice and in the capture cycle for each application of t_b, different C groups will capture. Assume that $group_1, group_2, group_3$, and $group_4$ are selected as the capture groups. Also assume that $group_1$ and $group_2$ capture the response in the capture cycle for the first application of t_b. Then $group_3$ and $group_4$ will capture in the capture cycle for the second application. □

V. Experimental Results

We conducted experiments on the largest ISCAS 89 [2] benchmark circuits and three industrial circuits D1, D2, and D3 (the number next to each circuit name in Table I is the component count of the circuit) to demonstrate feasibility of our idea. The results are reported in Table I. Results under the heading *Traditional* show results obtained by using traditional scan testing method (assigning random binary values to all don't cares and capturing all scan flip-flops during every capture cycle) on highly compacted test cube sets. The column *FC %* shows fault coverage. The number of patterns that are stored in the ATE memory is reported in the column *# pat*. The column *avg tran* reports the average number of transitions for the entire test cycles. The column *peak over* lists the maximum number of transitions during entire test cycles (including both shift and capture cycles) while the column *peak capt* shows the maximum number of transitions only during all capture cycles.

Results for the proposed method are reported in the columns under the heading *Proposed*. We used $n = 20$ (the maximum number of detections), $G = 12$ (the number of scan chain groups), and $C = 4$ (the number of capture groups) for every benchmark and industrial circuit. Since it is not possible to determine real $peak\_limit$ (maximum allowed switching activity that will not cause any adverse effect on the chip under test) without analyzing real chips under test, we computed $peak\_limit$ by the following procedure in the experiments. We first collected a set of test patterns whose all single detection faults can be detected by capturing C scan chain groups. Then we computed the maximum number of transitions caused by each test pattern in the set during shift and capture cycles. $peak\_limit$ was defined as the maximum among those maximum numbers of transitions caused by individual test patterns. Fault coverage achieved by the proposed method is exactly same as that achieved by the traditional method for every circuit and hence omitted.

Results clearly show that both peak and average switching activity can be significantly reduced by the proposed method. About 36-46% reduction in overall peak transition (column *peak over red%*) is achieved. Average reduction in peak transitions during capture cycles (column *peak capt red%*) is about 30% for ISCAS circuits and even larger for the industrial circuits. Reduction in average numbers of transitions for the entire test cycles is in a range of 56-85%. If we use a smaller number for C, then we will achieve even higher reduction in both peak and average transitions. A straightforward solution to reduce switching activity during scan shift cycles is to reduce the speed (frequency) of the test clock. Let us compare results obtained by the proposed approach with those that can be obtained by the straightforward approach. About 75 % reduction in the average number of transitions is obtained for s13207 by the proposed method. In order to achieve 75 % reduction in average power dissipation by reducing the clock speed, the speed of test clock should be reduced by a factor of 4. This will increase test application by about factor of 4. In contrast, the increase in test application time for the proposed method is only 16.6%. Recall that the straightforward approach cannot reduce peak current during either scan shift cycles or capture cycles.

Since very few test cubes were reverse-compacted (see Section III), the increase in the number of test patterns due to reverse-compaction is minor for most circuits. Numbers in parentheses in the column *# pat* give increases in numbers of test patterns over the original test patterns in per cent. Numbers of test patterns increased about 0.5-12%. Note that the increase in test pattern counts for all industrial circuits is almost **negligible**. The column *test time (inc%)* gives increase in overall test application time in per cent (note that if all single detection faults cannot be detected by capturing only C scan chains, then we repeatedly apply the same pattern and capture different C scan chains in each capture cycle). Results show that increase in test application time is not significant (average about 18% for ISCAS circuits and 1-22 % for industrial circuits). The column *# com* shows the number of distinct combinations of C scan chains that are selected to capture for each set of test patterns.

Results obtained by the proposed method are compared also with results of two recent publications, [11] and [6]. For all benchmark circuits except s5378, the proposed technique achieves significantly higher reduction in average numbers of transitions than [11]. The proposed method achieves larger reduction also in peak capture cycle transitions than [11]. Note that [11] reduces the peak capture cycle transition only by 3.0% for s5378 and 1.7% for s9234 while the proposed method reduces 33.1% and 28.9% respectively. Results for several different segment architectures are presented in [6]. Since we used $G = 12$ and $C = 4$, i.e., only about 1/3 of scan chains capture in each cycle, we compare results of the proposed method with three segment architecture results of [6]. For most circuits, reduction in the peak shift cycle transition achieved by the proposed method is larger than that achieved by [6]; the proposed method achieves on an average 40% reduction while [6] achieves 32% reduction. Reduction in peak capture cycle transitions achieved by both methods are close. [6] achieves higher reduction in the average number of transitions than the proposed method. In [6], scan flip-flops are reordered to minimize the number of special flip-flops that are inserted to break dependency between scan chains. Inserting special flip-

TABLE I
EXPERIMENTAL RESULTS

CKT	*Traditional*					*Proposed*						*[11]*				*[6] 3 segments*		
name	*FC %*	*# pat*	*avg tran*	*peak over*	*peak capt*	*# pat (inc%)*	*test time inc%*	*avg red %*	*peak over red%*	*peak capt red%*	*# com*	*FC %*	*# pat*	*avg tran red%*	*peak capt red%*	*avg tran red%*	*peak shift red%*	*peak capt red%*
s5378	98.96	138	607	2381	1976	170(18.8)	23.9	55.8	36.6	33.1	46	99.13	112	56.1	3.0	71.6	43	39
s9234	94.13	216	2889	4357	3529	235 (8.1)	12.0	60.0	38.9	28.9	124	93.48	141	40.9	1.7	69.0	19	17
s13207	98.27	199	2807	6812	5084	218 (9.5)	16.6	74.7	46.1	38.5	122	98.46	263	63.6	19.3	69.4	44	41
s15850	96.92	165	3791	7741	4689	181 (9.7)	17.6	68.9	42.2	12.9	125	96.68	124	53.0	23.5	69.6	23	26
s38417	99.53	189	16562	19302	13771	211(11.6)	21.7	62.6	40.1	28.0	203	99.47	102	32.8	21.1	90.9	28	20
s38584	96.44	191	6505	19402	14148	217(13.6)	18.3	65.1	35.9	41.2	152	95.85	124	40.2	13.7	92.4	36	35
D1 172K	99.12	565	46538	61750	46538	573(1.4)	22.1	73.7	45.8	34.3	501							
D2 223K	99.17	983	79627	122013	105291	1021(3.9)	6.4	84.7	41.2	31.9	276							
D3 98K	99.58	415	26913	69673	43663	417(0.5)	1.2	80.7	3.8	50.5	154							

flops increases hardware overhead. Since typically scan chains are routed to minimize routing overhead, reordering scan flip-flops to minimize the number of special flip-flops will increase routing overhead and sometimes not be possible due to severe routing congestion. In contrast, the proposed method does not require specific order of scan flip-flops. Since only extra logic that is required to implement the proposed method is a small control register, hardware overhead is negligible.

VI. CONCLUSION

In this paper, a technique that can efficiently reduce peak and average switching activity during test application is proposed. The peak transition is reduced by about 40% and average number of transitions is reduced by about 56-85%. This reduction in peak and average switching activity is achieved without any decrease in fault coverage or substantial increase in test pattern counts. The proposed method can reduce switching activity during both scan shift and capture cycles.

Unlike [6, 5], which require dividing each scan chain into multiple sub-chains and driving each scan sub-chain with a separate clock tree, the proposed method does not require any specific clock tree construction or scan chain routing. Test cubes generated by any ATPG can be processed by the proposed method to reduce peak and average switching activity without any capture violation. Hardware overhead for the proposed method is negligible. Further, the hardware for the proposed method can be implemented without detailed knowledge of the design and need not be redesigned for last minute design changes. Another advantage of the proposed method is that reduction in switching activity is adjustable depending on the desired level of switching activity.

REFERENCES

[1] N. Z. Basturkmen, S. M. Reddy, and I. Pomeranz. A Low Power Pseudo-Random BIST Technique. In *Proceedings IEEE International Conference on Computer Design*, pages 468–473, 2002.

[2] F. Brglez, D. Bryan, and K. Kozminski. Combinational Profiles of Sequential Benchmark Circuits. In *Proc. of International Symposium on Circuits and Systems*, pages 1929–1934, 1989.

[3] J.-S. Chang and C.-S. Lin. Test Set Compaction for Combinational Circuits. *IEEE Trans. on Computer-Aided Design of Integrated Circuit and System*, Vol. 14(11):1370–1378, November 1995.

[4] P. Girard, L. Guiller, C. Landrault, and S. Pravossoudovitch. A Test Vector Inhibiting Technique for Low Energy BIST Design. In *Proceedings VLSI Testing Symposium*, pages 407–412, 1999.

[5] K.-J. Lee, S.-J. Hsu, and C.-M. Ho. Test Power Reduction with Multiple Capture Orders. In *Proceedings Asian Testing Symposium*, pages 26–31, 2004.

[6] P. Rosinger, B. M. Al-Hashimi, and N. Nicolichi. Scan Architecture with Mutually Exclusive Scan Segment Activation for Shift-and Capture-Power Reduction. *IEEE Trans. on Computer-Aided Design of Integrated Circuit and System*, Vol. 23(7):1142–1153, July 2004.

[7] R. Sankaralingam and N. A. Touba. Controlling Peak Power During Scan Testing. In *Proceedings VLSI Testing Symposium*, pages 153–159, 2002.

[8] J. Saxena, K. M. Butler, and L. Whetsel. An Analysis of Power Reduction Techniques in Scan Testing. In *Proceedings IEEE International Test Conference*, pages 670–677, 2001.

[9] S. Wang and S. K. Gupta. An Automatic Test Pattern Generator for Minimizing Switching Activity during Scan Testing Activit. *IEEE Trans. on Computer-Aided Design of Integrated Circuit and System*, Vol. 21(8):954–968, Aug. 2002.

[10] S. Wang and S. K. Gupta. LT-RTPG: A New Test-Per-Scan BIST TPG for Low Switching Activity. *IEEE Trans. on Computer-Aided Design of Integrated Circuit and System*, Vol. 25(10):1565–1574, Aug. 2006.

[11] X. Wen, Y. Yamashita, S. Kajihara, L.-T. Wang, K. Saluja, and K. Kinoshita. On Low-Capture Power Test Generation for Scan Testing. In *Proceedings VLSI Testing Symposium*, pages 265–270, 2005.