Generation of Multiple Primary Input Blocking Patterns for Power Minimization during Scan Testing

Wang-Dauh Tseng
Department of Computer Science & Engineering, Yuan Ze University Chung-li, Taiwan, 32026, ROC
wdtseng@saturn.yzu.edu.tw

Hsu-Yang Lin
Department of Computer & Information Science, Soochow University, Taipei, Taiwan, 10048, ROC
jl@cis.scu.edu.tw

ABSTRACT

In this paper we propose a new approach to generate multiple input control patterns for applying to the primary inputs during shift cycle such that the switching activity occurred in the combinational part of the circuit under test can be suppressed as much as possible. Experiments performed on the ISCAS 89 benchmark circuits show that the proposed approach can always produce better results than the existing approaches.

1. Introduction

Power dissipation in CMOS can be divided into static and dynamic. Leakage current or other current drawn continuously from the power supply causes static dissipation. Dynamic dissipation occurs during output switching because of short-circuit current, and charging and discharging of circuit capacitances. Static dissipation currently has small magnitudes in digital CMOS circuits. Hence for such circuits, the dynamic dissipation is the dominant source of power consumption. Power dissipation during test application is significantly higher than during the circuit's normal operation. This increase can be attributed to the significant correlation that exists between successive vectors applied to a given circuit during system normal operation. In contrast, the correlation between consecutive test patterns generated by an automatic test pattern generator (ATPG) for external testing or by a linear feedback shift register (LFSR) for built-in self test (BIST) can be low. It has reported in [1] that the power consumed during test could be twice as high as the power consumed during normal operation. Hence, it is important to ensure that the heat dissipation during test application does not destroy a chip during test. It is also important to reduce switching activity during test application since excessive switching activity may cause logical error in a fault-free chip leading to an unnecessary loss of yield [2].

Numerous techniques have been proposed to address the problem of reducing test power [3-7]. The majority of these techniques concentrate on reducing the dynamic power dissipation by minimizing the switching activity. In scan-based testing, when scanning in test vectors, typically a much larger percentage of the flip-flops will change values in each clock cycle. The change in the flip-flop will cause other gates in the combinational part of the circuit to switch. The excessive switching activity during scan testing can cause test power to be significantly high. Several techniques in the literature aim at reducing test power through decreasing the number of transitions in the scan chain. To block the scan rippling at the inputs of the internal circuit, the approach in [3] adds additional circuitry to the scan cells to hold the outputs at a constant value during scan. This method reduces substantially the power dissipation during the shifting cycle, but it leads to performance degradation by introducing undesired delay. The approach in [4] partitions the original scan chain into several scan paths and activates these scan paths using different enable lines. Each time only one scan path is activated to restrict the scan rippling. However, since the whole scan chain will be activated when the data is captured, peak power reduction may not be guaranteed. For a scan sequential circuit, each test vector applied to the CUT is composed of primary input part and state input part. The primary inputs are important only at the application cycle when the entire vector is applied; the primary inputs can therefore be set arbitrarily during the shift cycle without affecting test efficiency. The authors in [5], exploiting this observation, propose a scheme which determines the most appropriate updating time of the primary input part of current test vector to reduce test power. The idea behind this scheme is that finding the best primary input change time will lead to higher correlation between consecutive values on the input lines of the combinational part of the circuit. Since this approach requires high computational time, it is infeasible for large sequential circuits. In [6], the authors also exploiting the same observation by identifying an input control pattern for a full scan circuit. They apply the control pattern to the primary inputs of the circuit during scan operation, thereby minimizing switching activity in the combinational part of the circuit. However, the quality of the input control pattern generated by this approach can not be guaranteed because the gate selection criterion which is the kernel of the approach can not exactly predict the number of transitions reduced by blocking a selected gate.

In this paper we propose a new approach to generate an input control pattern for applying to the primary inputs during shift cycle such that the switching activity which occurs in the combinational part of the circuit can be suppressed as much as possible. To achieve this, an impact function based on transition density is developed to measure the impact of transitions at a gate on the switching activity in the combinational part of the

circuit. The gate with higher impact function value has higher probability to cause more transitions and has higher priority to be blocked. The term "input control pattern" is also called "primary input blocking pattern" in this paper to reflect the concept of the proposed approach. Experimental results reported in Section 5 have shown that the proposed approach always produces better results than the approach in [6].

2. Circuit definitions

Consider the full scan sequential circuit shown in Fig.1, comprised of a block of combinational circuit C and a set of m state elements. The primary inputs and outputs of the circuit are $x_1, x_2, \ldots, x_n$, and $z_1, z_2, \ldots, z_k$, respectively. The present state variables, $y_1, y_2, \ldots, y_m$, constitute the state inputs of the combinational circuit. The next state variables, $Y_1, Y_2, \ldots, Y_m$, constitute the state outputs of the combinational circuit. The *forward cone* of line l, $FC(l)$, is defined as the portion of a circuit whose signals are reachable by a forward trace of the circuit topology starting at l. The *primary forward cone* (*PFC*) is defined as the portion of the circuit which is the union of forward cones of all primary inputs. The change of values on primary inputs by applying consecutive test vectors causes switching activity in the primary forward cone. The *state forward cone* (*SFC*) is defined as the portion of a circuit which is the union of forward cones of all state inputs. The change of values on state inputs by shifting a scan vector causes switching activity in the state forward cone. The portion of the circuit defined as the intersection of primary forward cone and state forward cone is referred to as *blocking cone* (*BC*). Any gate in the blocking cone has at least one input in the path which starts from a primary input and one input in the path which starts from a state input. Hence, if one of the inputs of a gate in the blocking cone is assigned to its controlling value by applying binary values to the primary inputs, the gate can be blocked and the transitions at the gate may not be propagated to its forward cone. A controlling value refers to the input logic value of a gate which determines its output logic value, irrespective of the other input values (e.g., 0 for AND and 1 for OR). Not all primary inputs contribute to blocking gates in the blocking cone. Assume that $PI = \{x_i \mid 1 \leq i \leq n\}$ is the set of primary inputs and $SI = \{y_i \mid 1 \leq i \leq m\}$ is the set of state inputs of a circuit. The *independent primary inputs*, $IPI = \{p_i \mid p_i \in PI$ and $FC(p_i) \cap SFC = \phi\}$, is defined as the set of primary inputs whose corresponding forward cones are disjoint with state forward cone and can not be used to block gates in the blocking cone by assigning binary values to them. Contrarily, only those primary inputs that are not in IPI may contribute to blocking gates in the blocking cone and can be defined as *control primary inputs*, $CPI = \{ p_i \mid p_i \in PI - IPI\}$. The *independent state inputs*, $ISI = \{s_i \mid s_i \in SI$ and $FC(s_i) \cap PFC = \phi\}$ is defined as the set of state inputs whose corresponding forward cones are disjoint with primary forward cone. The gates in the forward cone of any state inputs in ISI can not be blocked by assigning binary values to any primary inputs. The set of state inputs that are not in ISI can be defined as *blocking state inputs*, $BSI = \{s_i \mid s_i \in SI - ISI\}$. Fig. 2 shows an example circuit. The primary inputs are $\{x_1, x_2, x_3, x_4\}$, $\{S_1, S_2, S_3, S_4\}$ are scan cells, $\{y_1, y_2, y_3, y_4\}$ are the state inputs, and $\{z_1, z_2\}$ are the circuit outputs. The forward cones of primary input x_2 and y_2 are $\{c_1, c_2, c_4, c_5, c_7, c_8, c_{10}\}$ and $\{c_2, c_3, c_5, c_6, c_7, c_8, c_9, c_{10}, c_{11}\}$, respectively. The blocking cone is $\{c_2, c_5, c_7, c_8, c_9, c_{10}\}$. The independent primary inputs $IPI = \{x_1\}$ and the independent state inputs $ISI = \{y_3\}$. Clearly, control primary inputs $CPI = PI - IPI = \{x_2, x_3, x_4\}$ and blocking state inputs $BSI = SI - ISI = \{y_1, y_2, y_4\}$.

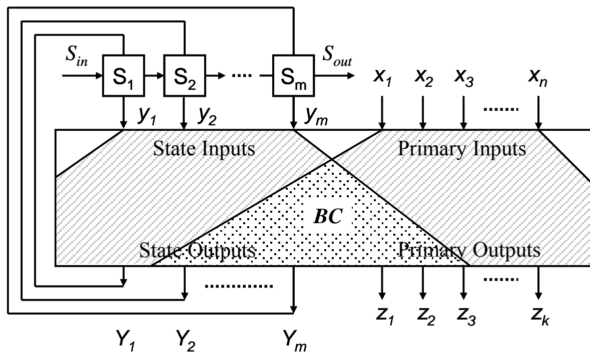

Fig. 1. A full scan sequential circuit.

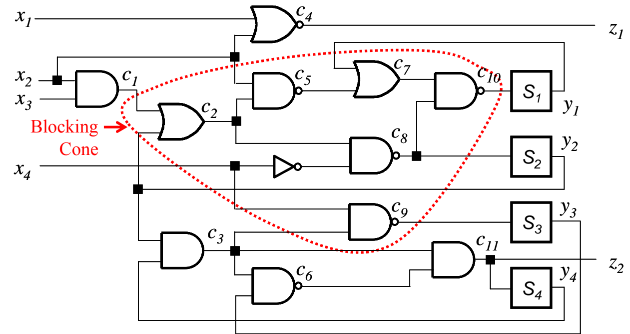

Fig. 2. An example circuit.

3. Impact function

The purpose of the proposed approach is to find a set of binary values for *CPI* such that as many transitions caused by the gates in the blocking cone can be suppressed as possible. However, to block a set of gates simultaneously by assigning controlling values to one of their inputs may result in conflicts in the assignment of binary values to the *CPI*. Depending on the circuit topology, the transitions at some gates cause more transitions than those at other gates. For example, gates with high fanouts may have higher probabilities to propagate transitions than gates with low fanouts and gates in the beginning of a forward cone will propagate more transitions than gates at the end of the forward cone. Hence we need a selection criterion to determine the blocking priority of gates when conflicts arise. The authors in [7] propose a gain function to identify among the primary inputs of the circuit those that cause more transitions at internal circuit lines. The gain function is based on the transition density and measures the impact of reducing the transition density at a selected input on

total switching activity in the circuit. In this section, we develop a selection criterion called impact function which is modified from the gain function to measure the impact of suppressing transitions at a selected gate on total switching activity in the internal circuit.

Consider a CUT with *m* circuit inputs $x_1, x_2, \ldots, x_m$. The *signal probability* $sp(c)$ of a circuit line *c* is defined as the probability that *c* is set to 1:

$$sp(c) = Pr(c = 1) \quad (1)$$

Signal probability can be propagated through logic gates based on simple rules of probability and logic function of the gates. The *transition probability* of a circuit line *c* is the probability of the signal making a transition from one state to another at any time *t* and is denoted by $p_t(c)$. Under the assumption that the values applied to each circuit input are temporally independent, we can write:

$$p_t(c) = 2 \cdot sp(c) \cdot (1 - sp(c)) \quad (2)$$

Let $n_c(T)$ be the number of transitions at a circuit line *c* in a time interval of length *T*. The *transition density* at *c*, i.e. the number of transitions per second at *c*, is defined as:

$$D(c) = \lim_{T \to \infty} \frac{n_c(T)}{T} \quad (3)$$

Let f_i be a function that depends on circuit input x_i. The Boolean difference of f_i with respective to x_i is defined as follows:

$$\frac{\partial f_i}{\partial x_i} = f_i \mid_{x_i=1} \oplus f_i \mid_{x_i=0} \quad (4)$$

where $\oplus$ denotes the exclusive-or operation. The Boolean difference signifies the condition under which output f_i is sensitized to circuit input x_i. If the circuit inputs x_i, $i = 1, \ldots, m$, to the CUT are not spatially correlated, the transition density of a circuit line *c* can be defined in terms of the Boolean difference with respect to each circuit input, $\partial f_i / \partial x_i$, and the transition density of each circuit input, $D(x_i)$, as:

$$D(c) = \sum_{i=1}^{m} P\left(\frac{\partial f_i}{\partial x_i}\right) D(x_i) \quad (5)$$

The Boolean difference $\partial f_i / \partial x_i$ represents the condition for sensitizing circuit input x_i to output f_i as noted in Eq. (4). Therefore, $P(\partial f_i / \partial x_i)$ signifies the probability of sensitizing input x_i to output f_i, while $P(\partial f_i / \partial x_i)D(x_i)$ is the contribution of transitions at output f_i due to circuit input x_i only. Hence, the contribution of transitions at output f_i due to all the circuit inputs is obtained by taking the summation over all the circuit inputs of the CUT.

As shown in Eq. (5), the transition density of a circuit line *c* is the sum of the transitions at each circuit input that sensitize to line *c*. Hence, the portion of the transition density of line *c* due to the transition at a circuit input x_i, is given by

$$D_{x_i}(c) = P\left(\frac{\partial f_i}{\partial x_i}\right) D(x_i) \quad (6)$$

Similarly, the portion of the transition density of line *c* due to the transition at a specific line *k*, can be written by

$$D_k(c) = P\left(\frac{\partial g_i}{\partial k}\right) D(k) \quad (7)$$

where g_i is a function that depends on circuit line *k*. The sum of transition densities of lines in the forward cone of *k*, $FC(k)$, that can be attributed to the transitions at *k* is given by:

$$D_k = \sum_{\forall c \in FC(k)} D_k(c) \quad (8)$$

For the CMOS circuit technology, dynamic power due to the charging and discharging circuit capacitances is the dominant source of power consumption. Hence, the power dissipation in a circuit depends on the load capacitance of internal lines. However, lines connected to more gates are lines with higher parasitic capacitance. If two circuit lines have the same transition density, the one with higher fanout will consume more power than the other one with lower fanout. Load capacitance also depends on the type and the size of the device. For example, a 2-input NOR has more load capacitance than a 2-input NAND, and the load capacitance difference increases as the number of inputs increase. So, two factors, the fanout and device coefficient, are also considered in the impact function as the weights of the transition density. Although wire length also affects power consumption, for simplicity, it is not considered in the impact function. For each circuit line *k* the impact function IMP_k can be expressed as:

$$IMP_k = \sum_{\forall c \in FC(k)} F_c \alpha_c D_k(c) \quad (9)$$

where F_c and α_c are the fanout and device coefficient of circuit line *c*, respectively. The fanout of the lines is defined by circuit topology. The device coefficient can be obtained once the circuit has been synthesized. In this derivation, the Boolean difference of f_i with respect to *c* is derived from the signal probability. However the signal probability $P(\partial f_i / \partial sp_i)$ can easily be calculated using the similar procedure which is used to calculate detection probability [2].

Table 1 illustrates the calculation of impact function value for primary input x_3 in Fig. 2. The first row shows the circuit lines in the forward cone of x_3. The second and third rows show the corresponding fanout and transition density of each circuit line in each column, respectively. The circuit lines and their fanouts included in the forward cone of x_3 can be obtained directly from the circuit structure. However the transition density for each circuit line can be calculated using Eqs. (4) and (6). Take circuit line c_5 for example. As shown in Fig. 2, the Boolean function f_{c5} for circuit line c_5 can be express as $\overline{(x_2x_3 + y_2)x_2}$. By the definition in Eq. (4) the Boolean difference of f_{c5} with respective to x_3 can be expressed as $\frac{\partial f_{c5}}{\partial x_3} = \overline{(x_2x_3 + y_2)x_2}\Big|_{x_3=0} \oplus \overline{(x_2x_3 + y_2)x_2}\Big|_{x_3=1} = x_2\overline{y_2}$. Hence the signal probability for c_5, $f\left(\frac{\partial f_{c5}}{\partial x_3} = x_2\overline{y_2}\right)$, is equal to 1/4. Assume that the signal probability for each bit in a test vector is 1/2. By Eq. (2) the transition probability for x_3 is equal to 2 × 1/2 × (1 - 1/2) = 1/2.

According to Eq. (6), the transition density for c_5 is equal to 1/4 × 1/2 = 1/8. Assuming that the device coefficient is 1 for all type of gates, the impact function value of x_3 can be calculated by Eq. (9) and is equal to 1×1/2 + 2×1/4 + 1×1/8 + 1×1/16 + 2×1/16 + 1×3/32 = 45/32 = 1.4.

Table 1. Calculation of impact function value for primary input x_3.

$FC(x_3)$	c_1	c_2	c_5	c_7	c_8	c_{10}	*IMP*
Fanout	1	2	1	1	2	1	1.4
TD	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{3}{32}$	

4. Proposed approach

To block a gate in the blocking cone means to set *CPI* values that lead to one of the gate's inputs to have its controlling value. This is an instance of the line justification problem, which deals with finding an assignment of primary input values that results in a desired value setting on a specified line in the circuit. Line justification is a recursive process in which the value of a gate output is justified by values of the gate inputs, and so on, until primary inputs are reached. The line justification procedure can be found in the *D-algorithm* which forms the concept of many practical ATPG programs [2]. The proposed approach consists of two basic steps. First, a *blocking table* is established to contain the blocking sub-pattern for each gate to be blocked. The impact function value associated with each sub-pattern is also included in the table. Second, a *combination compatibility graph* (CCG) is constructed from the blocking table. The combination compatibility graph is used to identify a set of combination compatible blocking sub-patterns in which the sum of the impact function values associated with these sub-patterns is maximal.

4.1 Blocking table generation

Starting at a state input, say s_i, in *BSI*, the gates belonging to both the forward cone of s_i and the blocking cone are traversed breadth-first. For convenience, the gates common to the forward cone of a state input and the blocking cone are called *target gates*. Whenever a target gate is visited, the inputs of the gate are justified by its controlling value one at a time. If an assignment of *CPI* values has been found for the gate being visited, the gate number, blocking sub-pattern (an assignment of *CPI* values obtained by justifying a gate input by its controlling value), and impact function value associated with the gate are recorded. If none of *CPI* assignments can be found for the gate, the next target gate is considered. We use a table called *blocking table* to contain these information. Once all target gates in the forward cone of s_i have been visited, the next state input is examined. This process continues until all state inputs in *BSI* have been checked. Table 2 shows the blocking table of the sample circuit in Fig. 2. The first column shows the circuit lines contained in the blocking cone. The second column shows the impact function values associated with the circuit lines and the last column shows the corresponding *CPI* values. The set *CPI* contains only three primary inputs x_2, x_3, and x_4. The notation "X" in the primary input sub-columns represents a don't care term.

Table 2. The blocking table for the circuit in Fig. 2.

Circuit line	*IMP*	*CPI* values		
		x_2	x_3	x_4
c_2	1.15	1	1	X
c_5	0.61	0	X	X
c_7	0.34	0	X	X
c_8	0.56	X	X	1
c_9	0	X	X	0
c_{10}	0	1	1	0

4.2 Primary input blocking pattern generation

The blocking sub-patterns produced by the line justification procedure, in general, partially specified. Two blocking sub-patterns are compatible if they do not specify opposite values for any control primary input. Two compatible blocking sub-patterns b_i and b_j can be combined into one blocking sub-pattern $b_{ij} = b_i \cap b_j$ in which only bit positions where both blocking sub-patterns have X's remain as X's. In all other bit positions, one or both of the blocking sub-patterns have a specified value, which is incorporated into the combined blocking sub-pattern. Clearly, the set of gates blocked by b_{ij} is the union of the sets of gates blocked by b_i and b_j. Thus we can replace b_i and b_j by b_{ij} without loss of blocking rate. Once the blocking table is obtained, the second step is to find a set of compatible sub-patterns such that the sum of impact function values associated with the sub-patterns is maximal. To solve this problem, the blocking table is first transferred into a *combination compatibility graph* (*CCG*). In the CCG, a node appears for each blocking sub-pattern and an edge exists between two nodes if the corresponding two blocking sub-patterns are compatible. The impact function is associated with each node. A clique of a graph is a subset of nodes in which each node is connected to all other nodes of the subset. A clique of the CCG represents a set of blocking sub-patterns that are pairwise compatible. Since the impact function value of a gate represents the amount of switching activity in the combinational circuit that the gate may affect, the problem to find a primary input blocking pattern to maximize the suppression of transitions during shift cycle can be reduced to finding a clique on the CCG such that the sum of the impact function values associated with the nodes in the clique is maximum. This problem is very similar to the *Maximum Clique* (*MC*) problem. The Maximum Clique problem in graphs asks for a clique of maximum size. Finding the maximum clique in a general graph is known to be an NP-complete problem. A rich literature exists on the algorithms for solving this problem. Efficient MC solvers are also available in the public domain. We have develop a program which is based on the algorithm proposed in [8] to find in CCG a clique in which the

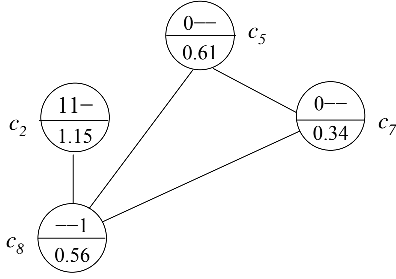

Fig. 3. The combination compatibility graph for Table 2.

sum of impact function values associated with the nodes in the clique is maximal.

Fig. 3 shows a CCG which is derived from Table 2. Since the impact function value for circuit lines c_9 and c_{10} is 0, these two nodes are excluded from Fig. 3. All possible cliques of the CCG are $G_1 = \{c_5, c_7\}$, $G_2 = \{c_5, c_8\}$, $G_3 = \{c_7, c_8\}$, $G_4 = \{c_2, c_8\}$, and $G_5 = \{c_5, c_7, c_8\}$. The total impact function values associated with cliques G_1, G_2, G_3, G_4, and G_5 are 0.95, 1.17, 0.90, 1.71, and 1.51, respectively. Among which G_4 has the largest total impact function value. Hence by merging the blocking sub-patterns in G_4, the primary input blocking pattern can be obtained. The primary input blocking pattern after merging the blocking sub-patterns in G_4 is 111. The detailed algorithm for the generation of primary input blocking pattern is depicted in Fig. 4.

```
Procedure PIBP( )
{
    Set blocking table empty;
    For each gate g_i in BC {
        c = controlling value of g_i;
        For each input l of g_i {
            BP = justify(l,c);          /
            If BP is not empty {
                Insert g_i, IMP_gi and BP into blocking
                table;
                Exit};
        }
    }
Obtain CCG from the blocking table;
Find the clique with maximal total impact function
value from CCG;
Merge the blocking sub-patterns in the clique;
Return the merged pattern;
}
```

Fig. 4. Primary input blocking pattern generation algorithm.

5. Experimental results

To validate the proposed approaches, we have carried out experiments on full scan versions of the ISCAS 89 benchmark circuits. The procedure described in Section 4 was implemented on a 1.5 GHz Pentium IV PC with 512 MB RAM running Linux and using GNU CC version 2.19. The signal probabilities for primary inputs are calculated by counting the number of 0s and 1s for each bit position from the deterministic test sequence. The don't care bits of the generated PIBP are set to zeros. The experimental results are summarized in Table 4. We use average transition count as quantitative measure for power dissipation. The average transition count is defined as the total number of transition count divided by the total number of clock cycles. Only the transition count during shift cycle is considered in our experiments. The efficiency of the input blocking technique very depends on the size of the blocking cone. Hence to evaluate the effectiveness of the PIBP algorithm, we compare the average transition count and the improvement of transition reduction with that of the exhaustive approach. In the second column, The first sub-column *Opt.* shows improved percentage of transition reduction by using the exhaustive approach. The second sub-column *PIBP* shows improved percentage of transition reduction by adopting the PIBP approach. The last column compares the PIBP approach with the CP approach in [6] in terms of *improvement efficiency* which is defined as the ratio between the improved percentages obtained by the PIBP (CP) approach and by the exhaustive approach. From Table 4 we can see that the proposed approach can always produce better results in terms of improvement efficiency than the approach in [6].

Table 4. Comparison of improvement efficiency between PIBP and CP.

Ckt	% Improvement		Efficiency	
	Opt.	*PIBP*	*C-alg.*[11]	*PIBP.*
S27	100	100	100.00	100.00
S208	47.48	47.48	100.00	100.00
S298	8.42	8.42	100.00	100.00
S344	9.21	9.21	55.02	100.00
S349	9.58	9.58	64.99	100.00
S382	28.90	28.90	100.00	100.00
S386	16.90	14.61	57.51	86.43
S400	28.17	28.17	100.00	100.00
S420	28.50	28.50	100.00	100.00
S444	26.97	26.97	100.00	100.00
S526	14.28	14.28	100.00	100.00
S1196	15.19	12.02	69.33	79.13
S1238	80.36	75.31	67.68	93.72
S1423	33.50	33.50	96.51	100.00
s5378	16.72	14.61	NA	87.38
s9234	15.34	12.03	NA	78.42
Avg.	29.97	28.97	86.50	95.32

To make more validation, part of the experiments in Table 4 has been remade by synthesizing each circuit to map to TSMC 0.18μm standard cell library. The power consumption in each circuit is estimated using Synopsys' PowerMill assuming a clock frequency equal to 200 MHz and a power supply voltage of 2.5V. The experimental results are summarized in Table 5. For each circuit we report the average power obtained first without input control technique and second with input control technique. Average power is expressed in milli-Watts. The last column in Table 5 shows the

percentage improvement. As can be seen, the average power of the circuits with input control is, in all cases, lower than that of the circuits without input control. And the percentage improvement measured by the PowerMill for each circuit in Table 5 is close to the percentage improvement obtained by counting the transitions in Table 4.

Table 5. Average power dissipation.

Circuits	Average Power (mW)		% Improv.
	without Input control (A)	with Input control (B)	$\frac{(A)-(B)}{(A)}\times 100$
s298	1.73	1.64	5.20
s344	1.55	1.48	4.52
s420	0.65	0.53	18.46
s526	1.94	1.79	7.73
s1423	9.46	4.37	53.81
s5378	18.89	16.95	10.27

6. Multiple Input Control Pattern

The efficiency of the input control technique very depends on the size of the *BC*. If *BC* is small, less test power improvement is achieved and less computation time is required. If *BC* is big, more power reduction is achieved and more computation time is required. Besides, the input control technique can only reduce the transitions occurred in the combinational part. No transitions in the scan chain can be reduced by this technique. However, the input control technique is independent to a lot of approaches such as scan cell ordering, test vector ordering, and scan chain partitioning, so it can be utilized together with these approaches to further reduce test power.

Unlike the approach in [6] where to find the control pattern only considers the circuit structure, the PIBP approach considers not only the circuit structure but also the signal probabilities for each test vector. Since the signal probabilities for a state input is different when scanning in different test vectors, the impact function values for each circuit line in the blocking cone is also different when applying different test vectors. That is to say, the priority for each gate to be blocked in the blocking cone is different when applying different test vectors. Hence we can always find a PIBP for each test vector and apply the PIBP to the PIs when the test vector is applied. The approach to find a PIBP for each test vector is called *multiple PIBP* (*MPIBP*). Table 6 shows the comparison of test power improvement between single PIBP and multiple PIBP. To exhibit the efficiency of the MPIBP approach, we use exhaustive method to generate the single PIBP. From Table 6 we can see that the MPIBP approach can reduce more power than the single PIBP approach in all cases. It is easy to extend the proposed approach to the MPIBP; however, the MPIBP can not be applied to the approach in [6].

7. Conclusion

We have presented a new input control pattern generation technique for minimizing power dissipation in full scan circuit during test. This technique is based on an impact function to determine the priorities of gates to be blocked. The two main steps in generating the input control pattern are first, to identify a CPI assignment for each gate in the blocking cone and second, to select a set of consistent assignments which can block as many gates as possible. Experimental results have shown that the proposed approach can achieve very high improvement efficiency and always produces better results than the existing approaches.

Table 6. Comparison of single PIBP with multiple PIBP in terms of test power improvement.

Ckt	*l*	% Improvement	
		Single PIBP	*Multiple PIBP*
S208	25	47.48	54.94
S298	22	8.42	14.49
S344	21	9.21	18.53
S349	22	9.58	18.74
S382	26	28.90	40.13
S386	43	16.90	26.43
Average		20.08	28.88

References

[1] Y. Zorian, "A distributed BIST control scheme for complex VLSI devices," *Proc. 11th IEEE VLSI Test Symp.*, 1993, pp. 4-9.

[2] Niraj Jha and Sandeep Gupta, *Testing of Digital Systems*, Cambridge University Press, 2003.

[3] S. Gerstendorfer and H. J. Wunderlich, "Minimized power consumption for scan-based BIST," *Proc. IEEE int'l Test Conf.*, 1999, pp. 77-84.

[4] L. Whetsel, "Adapting scan architectures for low power operation," *Proc. IEEE int'l Test Conf.*, 2000, pp. 863-872.

[5] N. Nicolici, B. M. Al-Hashimi, and A. C. Williams, "Minimisation of power dissipation during test application in full-scan sequential circuits using primary input freezing," *IEE Proceedings on Computers and Digital Techniques*, vol. 147, pp.313-322, Sept. 2002.

[6] T. Huang and K. Lee, "Reduction of Power Consumption in Scan-Based Circuits during Test Application by an Input Control Technique," *IEEE Trans. Computer-Aided Design*, vol. 20, pp. 911-917, July 2001.

[7] S. Wang and S. K. Gupta, "DS-LFSR: a BIST TPG for low switching activity," *IEEE Trans. Computer-Aided Design*, vol. 21, pp. 842-851, July 2002.

[8] R. Boppana and M. M. Halldorsson, "Approximating maximum independent sets by excluding subgraphs," *BIT Numerical Mathematics*, vol. 32, No. 2, pp. 180-196, June 1992.