ATPG for Heat Dissipation Minimization for Scan Testing

Seongmoon Wang and Sandeep K. Gupta

CENG 96-22

Department of Electrical Engineering - Systems
University of Southern California
Los Angeles, California 90089-2562
(213) 740-2251

October 1996

ATPG for Heat Dissipation Minimization for Scan Testing

Seongmoon Wang and Sandeep K. Gupta

University of Southern California Electrical Engineering–Systems Los Angeles CA90089-2562

Abstract

An ATPG technique is proposed that reduces heat dissipation during testing sequential circuits that have full-scan. The objective is to permit safe and inexpensive testing of low power circuits and bare dies that would otherwise require expensive heat removal equipment for testing at high speeds. The proposed ATPG exploits all possible don't cares that occur during scan shifting, test application, and response capture to minimize switching activity in the CUT. Furthermore, an ATPG that minimizes the number of state inputs that are assigned specific binary values, has been developed. In order to guide the ATPG to generate test vectors that have maximum number of don't cares at state inputs, new cost: controllability and observability cost functions have been defined. These cost functions are used to guide backtrace and to select objectives from D-frontier. Don't cares at primary inputs during scan shifting and capture are used to block gates that may cause transition during scan shifting and don't cares at state inputs are assigned binary values that cause the minimum number of transitions.

The proposed algorithm has been implemented and the generated tests are compared with those generated by a simple PODEM implementation for full scan versions of ISCAS 89 benchmark circuits. Tests generated by the proposed ATPG decreased the average number of transitions during test from 22% to 89% with higher reductions occuring in circuits that have more primary and state inputs. These reductions in circuit transitions were achieved with a reasonable increase in the time complexity of the ATPG (factor of 2 to 3).

1 Introduction

The main objective of traditional test development has been the attainment of high fault coverage. As the techniques have matured and this objective has been attained, other objectives have become important. We believe that reducing the heat dissipation during test application is rapidly becoming another objective of the test development process. In this paper, we present a new *automatic test pattern generator* (*ATPG*) algorithm that generates tests for *full scan circuits* that minimize heat dissipation in the circuit during their application via the scan chain. Of course, the tests generated by the proposed ATPG achieve high fault coverage.

The importance of heat dissipation consideration during test development is already influencing the design of practical test methodologies. For example, it is reported in [Zor93] that one of the major considerations in test scheduling has been the fact that the heat dissipated during test application can be significantly higher than that during the circuit's normal operation (sometimes 100-200% higher).

There are two main reasons for excessive heat dissipation during test application. Firstly, the correlation between consecutive test vectors is often significantly lower than that between consecutive vectors applied to a circuit during its normal operation. The fact that a significant correlation exists between consecutive vectors during the normal operation of a circuit is what has motivated several architectural concepts, such as cache memories, and is central to their effectiveness. This is even more true for the high speed systems for processing of digital audio and video signals, where the inputs to most modules change relatively slowly. In contrast, the correlation between consecutive test vectors generated by an automatic test pattern generator (ATPG) is very low, since a test is generated for a given target fault without any consideration of the previous vector in the test sequence.

The use of design-for-testability (DFT) techniques can further decrease the correlation between successive test vectors. Consider a finite state machine that is implemented as a sequential circuit by using techniques for state assignment and logic design that minimize power dissipation [TPCD94]. One of the characteristics of the circuit that reduces power dissipation is the high correlation between the vectors representing successive states. However, the use of a scan based DFT technique can significantly decrease the correlation between consecutive vectors, since the values applied at the state inputs of the circuit represent shifted values of test vectors and circuit responses and have no particular temporal correlation. Hence, in such circuits, the application of ATPG generated vectors via scan can cause heat dissipation that is significantly higher than that during the circuit's normal operation. This is even more true for built-in self-test (BIST) techniques which typically employ random pattern generators (RPGs), such as linear feedback shift registers (LFSRs),

to generate test sequences where the correlation between consecutive vectors is provably small [Gol82].

The second reason is tied to trends such as circuit miniaturization for portability and high performance (smaller chips can be placed closer, decreasing interconnect delays). This is achieved by using circuit designs that decrease power dissipation and also by reducing the package size to aggressively match the average heat dissipation during a circuit's normal operation [Zor]. Hence, to ensure non-destructive testing of such a circuit, it is necessary to generate tests that will cause heat dissipation that is comparable to the heat dissipated during the circuit's normal operation.

Why then has the heat dissipation problem during testing not been considered in the past? There are two recent developments that are only beginning to bring this issue to light. In the past, the tests were typically applied at rates much lower than a circuit's normal clock rate (since only the coverage of stuck-at faults was deemed to be important and slow testers provided an inexpensive way of testing). However, in recent years, aggressive timing has made it essential for the tests to identify slow chips via delay testing. This is especially important for the growing number of circuits that are being manufactured for use in MCMs and must be tested and sold as bare die. For such circuits delay testing is not only highly desired but almost imperative – a fact reflected in the extensive demand for performance certified dies [Kee92, Par92]. Consequently, circuits are now tested at higher clock rates – if possible, at the circuit's normal clock rate (called at-speed testing). Consequently, the heat dissipation during test application is on the rise and is fast becoming a problem that requires close attention.

This paper describes an ATPG algorithm that generates tests that minimize heat dissipation during test application via scan chain. Heat dissipation can be reduced by decreasing the number of transitions in the circuit during test application. In combinational circuits, transitions can be reduced by reducing transitions between each pair of successive vectors [WG94]. However, in scan circuits, a large number of transitions occur in the circuit when test data is scanned into, and test response is scanned out of, the flip-flops between the application of successive vectors. Therefore, techniques to reduce transitions during scan in and scan out are required to reduce heat dissipation in a sequential circuit equipped with scan. The proposed ATPG reduces transitions by generating suitable tests and utilizing the large number of don't cares that exist during scan testing.

The tests generated by this ATPG can be used for at-speed testing of chips and bare die without running the risk of damaging the device under test by excessive heat dissipation. In case of bare die testing for MCMs, the use of tests generated by the proposed ATPG can obviate the need for expensive heat removal equipment that may be required otherwise. (It should be noted that the objective of the proposed ATPG is to ensure that the heat

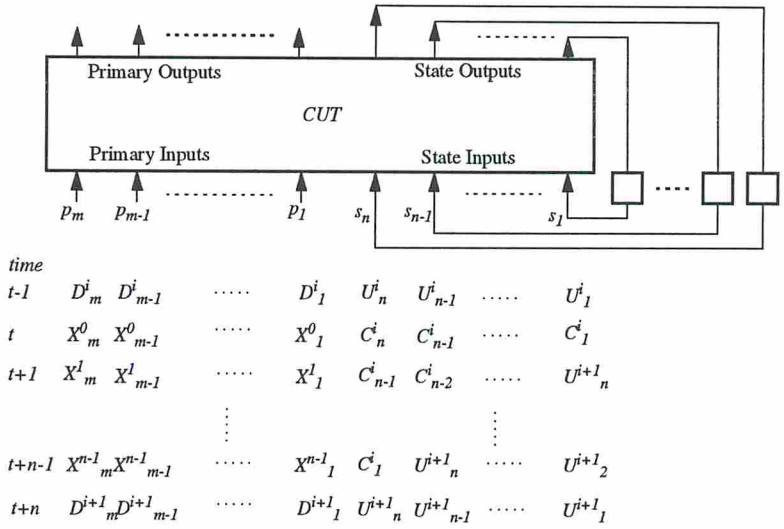


Figure 1: Application of Tests via Scan

dissipation during test application is low enough to ensure safe and non-destructive at-speed testing of the device under test without the need for expensive heat removal equipment during test application. The reduced power consumption during testing itself is of little consequence since test time is an insignificant part of a circuit's operational life.) Finally, to test a bare dice, power must be supplied during the period of test through probes. The proposed tests will reduce the power and ground noise, caused due to the high inductance of probes, by reducing the number of transitions in the circuit. This will prevent unnecessary loss of yield caused entirely due to the limitations of probing.

2 Scan Based Testing

In this paper, we assume that the sequential circuit under test (*CUT*) has full-scan that employs a single scan chain for test application. We also use the *single-stuck-at* fault model. In a such scenario, traditionally, an ATPG for combinational circuits is used to generate *combinational test vectors* by considering only the combinational part of the scan circuit under test. The resulting combinational test vectors are subsequently *edited* to obtain *scan tests* that are applied using the scan chain as described in the following. However, the above two step process can not be used for generating tests for scan circuits if the heat dissipation during test application is to be minimized. This is due to the fact that the heat dissipation is a function of both the combinational test vectors as well as how they are applied using the scan chain. The objective of the following is to develop an ATPG algorithm that minimizes heat dissipation during the test application, while still obtaining high fault coverage, by

considering both these steps together.

Figure 1 describes the test application via scan for a CUT that has m primary inputs and n state inputs. The combinational ATPG generates a set of combinational test vectors, each of which is a binary $m+n$ tuple and must be applied to the m primary inputs $(p_1, p_2, ..., p_m)$ and the n state inputs $(s_1, s_2, ..., s_n)$ during test application. The bits of a test vector that are applied to the primary and state inputs will be referred to as its *primary input* and *state input parts*, respectively. Assume that a test vector, $V^i = D^i_m$, D^i_{m-1},..., D^i_1, U^i_n, U^i_{n-1},..., U^i_1, is applied to the CUT through primary inputs and state inputs at time $t-1$, and the state part of the CUT response to V^i, say C^i_n, C^i_{n-1},..., C^i_1, is captured in the scan register at time t. Subsequently, the state input part (U^{i+1}_n, U^{i+1}_{n-1},..., U^{i+1}_1) of the next combinational vector V^{i+1} is shifted in during the next n cycles ($t+1$, $t+2$, ..., $t+n$) while the test response (C^i_n, C^i_{n-1},..., C^i_1) is shifted out. This process of shifting in the state part of a test vector by using the scan chain will be referred to as *scan shifting*. Note that no specific values need to be applied to the primary inputs at times $t, t+1, t+2, ..., t+n-1$. This is depicted by the vector X^j_m, X^j_{m-1},..., X^j_1 that is applied to primary inputs at time $t+j$, where $j = 0, 1, 2, .., n-1$. Finally, the primary input part of V^{i+1}, D^{i+1}_m, D^{i+1}_{m-1},..., D^{i+1}_1, is applied at $t+n$, and the response of the CUT to the combinational test vector V^{i+1} is captured into the flip-flops at $t+n+1$. This is repeated until all test vectors are applied.

The above discussion shows that the switching activity in a CUT during the application of a scan based test depends not only on correlation between the two consecutive combinational test vectors V^i and V^{i+1}, but also on how the tests are applied. Even though the vectors V^i and V^{i+1} may cause a minimum number of transitions in the CUT when applied in *two consecutive clock cycles* to the combinational part of the circuit, a significant number of transitions may occur in the circuit if they are applied to a sequential circuit via its scan chain. Hence, the sequence of test vectors that minimize heat dissipation in a combinational circuit may not be suitable for minimizing heat dissipation in testing the corresponding sequential circuit via scan. The test generation process must generate a combinational test vector ($V^{i+1} = D^{i+1}_m$, D^{i+1}_{m-1},..., D^{i+1}_1, U^{i+1}_n, U^{i+1}_{n-1},..., U^{i+1}_1) to minimize transitions in the circuit during its application. While the primary input part of the combinational test vector is only applied during one clock cycle, any value assigned to a state input by the ATPG is applied successively at several state inputs.

Also, a combinational test vector generated by a fault oriented *ATPG* (*e.g.*, V^{i+1}) is not fully specified. The target fault can hence be detected independent of the binary value assigned to these unspecified inputs. In addition to the don't cares in combinational vector V^{i+1}, the n m-tuples depicted as X^j_m, X^j_{m-1},..., X^j_1 ($j = t, t+1$, ..., $t+n-1$) in Figure

1, that are applied to the primary inputs during scan shifting, are not specified by the test vector V^{i+1}. This implies that all primary inputs during scan shifting can be treated as *don't cares* as well.

Don't cares in the state input part have different characteristics from those at primary inputs during scan shifting. Primary input don't cares are fully controllable, *i.e.* completely independent binary m tuples can be assigned to these don't cares during each scan shift cycle. On the other hand, only one binary n tuple is shifted into the scan register for each test vector. Consequently, a single binary value must be assigned to each don't care in the state input part of the combinational test vector. Furthermore, each bit of the state input part of the combinational vector is applied at several state inputs of the circuit during the scan shifting. The order in which each value is applied to the inputs is determined by the order of the flip-flops in the scan chain.

In summary, the value assigned to a primary input by the ATPG only affects the number of transitions during the *test clock cycle* when the test is applied. On the other hand, the value assigned at any state input can cause transitions during the *entire scan shifting* as well as during the *test clock*. Finally, all the primary inputs can be assigned arbitrary values during the entire scan shifting to minimize the number of transitions. Based on these considerations, the proposed ATPG algorithm first generates test vectors where only a minimum number of state inputs are assigned binary values. This enables a greater reduction in heat dissipation by allowing careful assignment of the state input don't cares — via the analysis of the transitions during all scan clocks.

The primary inputs are used to minimize the number of gates that can possibly have transitions due to scan shifting. This is achieved by assigning each primary input don't care a value that implies the *controlling value* at the inputs of the gates that are fed by that primary input as well as state input(s) to *block* the transitions caused by shifting of the scan chain contents. (The controlling value of a gate is the binary value which, when applied to any input of a gate, determines the output value of that gate *independent of the values applied at the other inputs of the gate.*) In the following, the considerations and a procedure for the assignment of don't cares to the primary inputs will be discussed first. Subsequently, the ATPG procedure that generates tests with minimum number of specified state inputs will be described, followed by the discussion of the overall test generation strategy.

3 Assignment of Primary Input during Scan Shifting

Consider a primary input p_j of a circuit. Associated with the input p_j are the don't care values, X_j^0, X_j^1,..., X_j^{n-1}, during scan shifting (*i.e.* at times $t, t+1,..., t+n-1$). One possible

strategy is to assign a *single* binary value to all the don't cares associated with each primary input p_j that is *fixed for all vectors.* There are two main advantages of such a strategy. Firstly, an appropriate binary assignment can be determined once for a given circuit and used for each combinational vector (V^i, V^{i+1}, ...), thereby reducing the run time complexity of the overall ATPG. More importantly, such binary values can be *implicitly stored* in an intelligent automatic test equipment, thereby drastically decreasing the test data volume. However, in some cases, it may be possible to decrease the number of transitions by assigning different values to a primary input during different scan clocks. The appropriate assignment for such input don't cares will then be determined by taking into account the specific test vector being applied, *i.e.* the content of the flip-flops. In the following, it will first be shown that a fixed value can be assigned to a large number of primary inputs in a manner that guarantees the minimization of the number of transitions, independent of the specific sequence of combinational vectors generated by ATPG. Techniques to identify such inputs and an appropriate binary assignment are presented. Finally, a technique, that takes into account the specific test vector being applied, to assign binary values to the remaining primary inputs is presented.

3.1 Notations and Definitions

3.1.1 Basic Circuit Definitions

If a line l of a circuit C is driven by a gate g, then l is said to be a *fanout* of g. The *transitive fanout* of a line l includes all lines reachable from l via a sequence of fanouts. A *path* is a sequence of consecutive circuit lines. The *inversion parity* of a path is the number of inverting gates along the path, modulo-2. That is, if the inversion parity of a path is 1, then the path is said to have *odd inversion parity*, otherwise the path is said to have *even inversion parity* [ABF90]. The *forward cone* of line l is the set of gates and circuit lines in the transitive fanout of l. During test generation, each line l in a circuit is assigned one of three values (0, 1, X). (Initially, all lines are assigned X.) A path along which all lines have unknown values (X) is called *an x-path.* The assignment of a binary value at l_1 can imply a desired value at l_2 only if there exists at least one x-path from the line l_1 to the line l_2.

Primary inputs that are not assigned any binary value (1 or 0) are used to *block* transitions at the outputs of gates that may have transitions due to shifting of the state input part of the vector into (and the test response out of) the scan chain.

Definition 1 *C = combinational part of a sequential CUT with binary values assigned to some or none of its lines*
$PI = \{p_i \mid p_i$ is a primary input of C that is not assigned a binary value$\}$

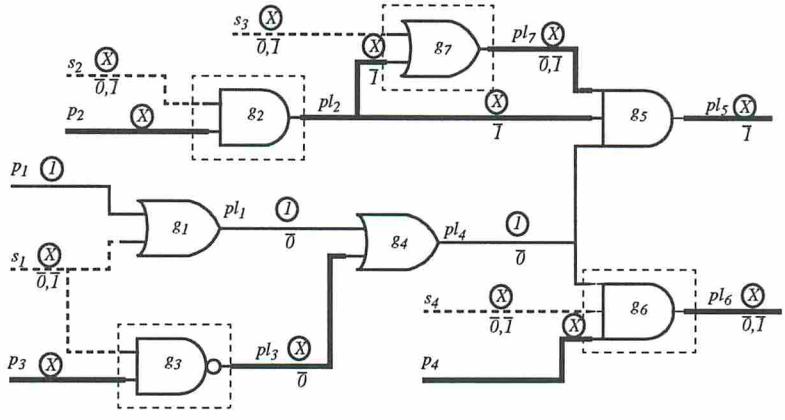


Figure 2: Example Circuit C

$SI = \{s_i \mid s_i$ *is a state input of* $C\}$
$TPI = \{pl \mid pl$ *is a line in the transitive fanout of any* $p \in PI\}$
$TPI^C = \{pl \mid pl$ *is a line not in the transitive fanout of any primary input*$\}$
$TPI' = \{pl \mid pl \in TPI$ **and** $\exists$ *an x-path from any* $p \in PI$ *to* $pl\}$

3.1.2 Blocking Objectives

If none of the inputs of a gate are assigned the controlling value of the gate, then the gate is called *unblocked gates*. If a transition propagates to the output of a gate g, then new transitions are caused at each fanout of g. Furthermore, transitions at the fanouts may cause more transitions at the output of the gates driven by the fanouts, and so on. The transition caused by a state input should hence be blocked as early (close to the state input) as possible, to prevent it from propagating into the forward cone of the state input. In this paper, we try to block all gates that are driven by a state input *directly* and a transitive fanout of a primary input. Such gates, that have at least one input in TPI' and at least one input in TPI^C, are called *blocking objectives* and are members of the set G. Set Gp_i contains the blocking objectives $g \in G$ that are in the forward cone of primary input p_i; gates in Gp_i are called *blocking objectives with respect to* p_i.

Definition 2 $C =$ *combinational part of a sequential CUT with binary values assigned to some or none of its lines*
$UB = \{ub \mid ub$ *is a gate with none of its inputs assigned the gate's controlling value*$\}$
$G = \{g \mid g \in UB$ *with at least one input in* TPI' *and at least one in* $TPI^C\}$

$Gp_i = \{g \mid g \in G$ **and** $\exists$ *an x-path from* p_i *to at least one input of* $g\}$, $\forall p_i \in PI$

Example 1 Figure 2 shows the combinational part of a sequential circuit that has primary inputs p_1, p_2, p_3, and p_4 and state inputs s_1, s_2, s_3, and s_4. All state inputs belong to SI, hence $SI = \{s_1, s_2, s_3, s_4\}$. Suppose that primary input p_1 is assigned a 1 to block gate g_1. Circled 1 or X at a line denotes the current value assigned to the line. With p_1 assigned a 1, PI contains only p_2, p_3 and p_4. The lines that are contained in TPI, $p_2, p_3, p_4, pl_3, pl_5, pl_6, pl_7$, are denoted by thick lines and the lines that are in TPI^C, s_1, s_2, s_3, s_4, are denoted by dotted lines. All elements of TPI, except pl_4, which is assigned a binary value (no x-path can exist from any primary input to pl_4), are also members of TPI'. □

3.1.3 Independent Inputs

If a primary input p_i can not be used to block transitions caused by a scan inputs during scan shifting (*i.e.* if $G'p_i = \phi$) and any transition caused by a scan input does not propagate to any gates in the forward cone of p_i, then the primary input p_i is called *independent.* Independent primary inputs are assigned binary values that minimize transitions based on the binary value assigned to the inputs in the preceding time frame. Appropriate values for such inputs are determined during each test clock (clock $t-1$ in Figure 1) by using the *don't care assignment* algorithm described in [WG94]. The determined values are maintained until the application of next test.

3.2 Single Input Conflict Free Assignment

If the assignment of a binary value at a particular primary input, say p_i, can not imply the non-controlling value at any input of the gates in Gp_i (which are fed by state input(s) as well as p_i) for any combination of values assigned to other inputs, then the assignment is always helpful to block a maximum number of gates *independent of the test vector* and *the values assigned to other inputs.* Such an assignment is said to be a *conflict free* assignment. Whether the don't care at a primary input p_i can be assigned a binary value in a conflict free manner is identified during *conflict free analysis* defined precisely in the following.

A primary input p_i is conflict free if a binary assignment at p_i can be made such that it helps block all gates in Gp_i. Let g_a be any gate that is fed by a transitive fanout of p_i and belongs to Gp_i. Let $\{xp_1, xp_2, ..., xp_h\}$ be the set of x-paths from p_i to the inputs of $g_a \in Gp_i$. Let $\{\Pi_1, \Pi_2, ..., \Pi_h\}$ be the parity of the x-path xp_i $(i = 1, 2, .., h)$. Finally let c_a be the controlling value of g_a.

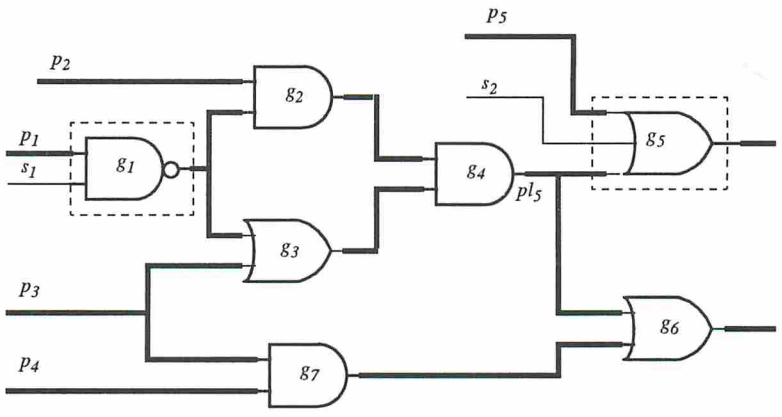


Figure 3: Circuit $C1$

Lemma 1 The assignment of primary input p_i is *conflict free* if and only if all $\Pi_i \oplus c_a$ $(i = 1, 2, ..., n)$ are identical for all $g_a \in Gp_i$. □

Example 2 In the circuit $C1$ shown in Figure 3, lines s_1 and s_2 are state inputs and p_1, p_2, p_3 and p_4 are primary inputs. Assume that all lines in the circuit are currently assigned X. Gates in dashed boxes (g_1 and g_5) currently belong to G. The NAND gate g_1 is fed by a primary input p_1 and a state inputs s_1. The OR gate g_5 is fed by a internal line pl_5 and a state inputs s_2. Both g_1 and g_5 are also members of Gp_1. The assignment at p_1 is conflict free if it helps block all gates in Gp_1, *i.e.* g_1 and g_5. All x-paths (p_1, g_1, g_2, g_4, pl_5 and p_1, g_1, g_3, g_4, pl_5) from p_1 to pl_5 have odd parity and the controlling value of g_5 is 1. Hence, p_1 needs to be set to 0 to block the transition at the output of g_5. Also p_1 needs to be set to 0 to block the transitions at the output of g_1. Since both these requirements can be simultaneously satisfied, assigning 0 to p_1 is always good in terms of blocking all gates in Gp_1. The assignment of 0 to p_1 is hence conflict free.

Assume that we replace the NAND gate g_2 in $C1$ with an AND gate. The parity of the path g_1, g_2, and g_4 then becomes even. Additionally, assume that p_3 has already been assigned a 1 to block the transition at the output of g_3. Thus, there exists only one x-path (p_1, g_1, g_2, g_4, pl_5) from pl_5 to p_1. Since the path parity is even, a 1 must be assigned pl_5 to block g_5 and, finally, a 1 is required at p_1 when backtraced along the above x-path. This would result in a conflict with the value 0 that is required to block g_1. Hence, p_1 can not be assigned in a conflict free manner in the modified circuit. □

The above definition of conflict free assignment can be expanded in three ways. Firstly, assigning a binary value to a primary input may set lines on one or more x-paths to binary values, consequently removing some lines from TPI'. This may also remove some

gates from the list of blocking objectives. Consequently, primary inputs that were previously not conflict free can be assigned in a conflict free manner after some primary input assignments. Furthermore, additional primary inputs may become *independent* of scan inputs after conflict free assignments are made.

Secondly, a gate $g \in G$ is blocked by assigning appropriate value to the primary inputs such that the assignment can imply the controlling value of g at one or more inputs of g. All scan inputs $s_i \in SI$ are assumed to be uncontrollable during the assignment of primary input don't cares, since the values of the scan inputs are not known during this analysis, which is performed *before* the tests are generated. Under these conditions, there may exist gates none of whose inputs can be set to the gates' controlling value by assigning any combination of binary values to the primary input. If all inputs of a gate g that belongs to TPI' are uncontrollable to the controlling value of g, then the gate g is *unblockable*. Transitions at the output of unblockable gates can not be blocked by assigning *any combination of binary values to primary inputs.* Uncontrollable gates are identified by performing uncontrollability analysis [AKR91]. The analysis starts by assigning all scan inputs to $\bar{1}$, $\bar{0}$ where the assignment of $\bar{1}$ ($\bar{0}$) assigned to a line l denotes that the line l is uncontrollable for 1 (0) [AKR91]. Next these values are propagated to all internal lines according to the propagation rules defined in [AKR91]. Unblockable gates can not be blocked by *any combination of primary inputs.* More inputs may be assigned conflict free values when G' is used instead of G during the conflict free assignment analysis.

Finally, if an x-path exists from primary input p_i to internal line pl_j (pl_j is a transitive fanout of p_i) and pl_j is controllable to the non-controlling value of the gate that is driven directly by pl_j, then the x-path is called an *influence x-path* with respect to p_i and pl_j. Influence x-paths are subset of x-paths and identified by performing uncontrollability analysis [AKR91]. When only influence x-paths (instead of all x-paths) are considered for the conflict free assignment analysis, more primary inputs can be assigned in conflict free manner by removing unnecessary requirements imposed on the conflict free primary inputs.

Example 3 A $\bar{1}$ or/and $\bar{0}$ below the circuit lines in Figure 2 denotes the result of the uncontrollability analysis. Scan inputs are assigned $\bar{1}$, $\bar{0}$ and these values are propagated to internal lines. p_1 in the circuit C is assigned 1 and other primary inputs are currently unknown. p_2 can not be assigned a binary value in a conflict free manner because the gates in $Gp_2 = \{g_2, g_7\}$ have different controlling values. However, since pl_2 is uncontrollable to 1 (since it has been assigned $\bar{1}$ by uncontrollability analysis), no input of g_7 can be assigned 1 (the controlling value of g_5). Therefore, g_7 is unblockable. When the unblockable gate g_7 is removed from the set Gp_2, p_2 can be assigned a 0 to block g_2, in a conflict free manner. □

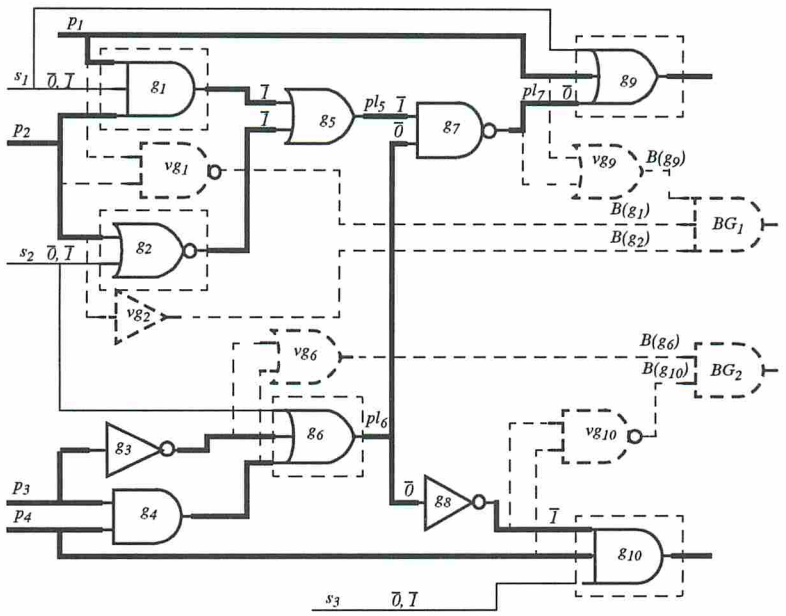


Figure 4: Example Circuit $C2$

3.3 Multiple Input Conflict Free Assignment

Even when no more conflict free primary input assignments can be found by considering each primary input individually, additional conflict free primary input assignments can be found by considering multiple primary inputs. Let $B(g_j)$ denote the necessary and sufficient condition to block $g_j \in G$ $(j = 1, 2, 3, ..., h)$ in terms of the suitable values of primary inputs. Thus each $B(g_j)$ is the function of primary inputs p_i $(i = 1, 2, ..., m)$. Additionally, let BG be the set of all $B(g_j)$ $(j = 1, 2, 3, ..., h)$. If a combination of primary input assignments satisfies all $B(g_j)$, then the combination is said to be a *conflict free multiple input assignment.* Typically, each $B(g_j)$ is a function of a subset of PI. The set of conditions BG can hence be partitioned into groups $(BG_k \subset BG)$, where all gates in each group BG_k depend on a subset of primary inputs PI_k and can be blocked independently of the primary inputs in other groups PI_j $(j \neq k)$.

Assume that currently none of the primary inputs of the circuit in Figure 4 is assigned a binary value. Thus gates g_1, g_2, g_6, g_9, and g_{10} belong to G and are denoted by dashed boxes. The conditions for blocking all gates $g_i \in G$ are given by:

$$
\begin{aligned}
B(g_1) &= \bar{p_1} + \bar{p_2} \\
B(g_2) &= p_2 \\
B(g_9) &= p_1 + p_2(\bar{p_1} + \bar{p_2}) = p_1 + \bar{p_1}p_2 \\
B(g_6) &= \bar{p_3} + p_3p_4 \\
B(g_{10}) &= \bar{p_4} + p_3p_4 + \bar{p_3}
\end{aligned} \tag{1}
$$

The above expressions are obtained by identifying the necessary and sufficient conditions for blocking each gate g_j ($g_j \in G$). In the above expressions, each primary input appears in its inverted ($\bar{p_i}$) as well as non-inverted (p_i) form. This implies that any single input can not be assigned in a conflict free manner. The expressions $B(g_1)$, $B(g_2)$, and $B(g_9)$ are dependent on only p_1 and p_2 while $B(g_6)$ and $B(g_{10})$ are functions of only p_3 and p_4. Thus the five expressions can be partitioned into two groups, $BG_1 = \{B(g_1), B(g_2), B(g_9)\}$ and $BG_2 = \{B(g_6), B(g_{10})\}$. Corresponding to the sets BG_1 and BG_2 are partitions of primary inputs $PI_1 = \{p_1, p_2\}$ and $PI_2 = \{p_3, p_4\}$. If a combination of primary input values satisfies all conditions in a group (*e.g.* BG_i evaluates to 1) the combination is said to be a conflict free multiple primary input assignment. When $p_1 = 0$ and $p_2 = 1$, then all expressions in BG_1 ($B(g_1)$, $B(g_2)$, and $B(g_9)$) become 1. Hence, the assignment $p_1 = 0$ and $p_2 = 1$ is a conflict free multiple input assignment. $B(g_6)$ and $B(g_7)$ can be also set to 1 by assigning $p_3 = 1$ and $p_4 = 1$. This assignment is also conflict free multiple input assignment. All gates in G are hence blocked by assigning $p_1 = 0$, $p_2 = 1$, $p_3 = 1$, and $p_4 = 1$.

Since obtaining expressions for the necessary and sufficient conditions for each gate in G is hard, conflict free multiple inputs are determined without obtaining expressions. Firstly, the primary inputs are partitioned into independent groups. For example, the conditions to block gates in BG_1 depend only on the inputs p_1 and p_2 and the conditions to block gates (g_6, and g_{10}) in BG_2 depend only on the inputs p_3 and p_4. Therefore circuit $C3$ has two independent groups of primary inputs $Group_1 = \{g_1, g_2, g_9\}$ $Group_2 = \{g_6, g_{10}\}$. In this paper, primary inputs can be partitioned into each independent group by identifying *strongly connected components* [CLR90].

Next, a virtual gate vg_i is added for each gate g_i ($g_i \in G$). Inputs of vg_i are connected to inputs pl ($pl \in TPI'$) of g_i if the controlling value of g_i can indeed be applied to pl via primary input assignment. The gate type of vg_i is determined such that the output of vg_i is set 1 if and only if g_i is blocked. This implies that vg_i is a NAND gate if g_i is a AND/NAND gate. And it is an OR gate if g_i is an OR/NOR gate. The expression at the output of vg_i corresponds to $B(g_i)$. In the circuit $C2$ (Figure 4), The AND gate g_1 has two inputs p_1 and p_2, in TPI'. Two input NAND gate vg_1 is then added for g_1, since both inputs p_1 and p_2 are controllable to 0 (the controlling value of g_1). p_1 and p_2 are connected to two inputs of vg_1.

$$
\begin{aligned}
B(g_1) &= \bar{p_1} + \bar{p_2} \\
B(g_2) &= p_2 \\
B(g_9) &= p_1 + p_2(\bar{p_1} + \bar{p_2}) = p_1 + \bar{p_1}p_2 \\
B(g_6) &= \bar{p_3} + p_3p_4 \\
B(g_{10}) &= \bar{p_4} + p_3p_4 + \bar{p_3}
\end{aligned} \tag{1}
$$

The above expressions are obtained by identifying the necessary and sufficient conditions for blocking each gate g_j ($g_j \in G$). In the above expressions, each primary input appears in its inverted ($\bar{p_i}$) as well as non-inverted (p_i) form. This implies that any single input can not be assigned in a conflict free manner. The expressions $B(g_1)$, $B(g_2)$, and $B(g_9)$ are dependent on only p_1 and p_2 while $B(g_6)$ and $B(g_{10})$ are functions of only p_3 and p_4. Thus the five expressions can be partitioned into two groups, $BG_1 = \{B(g_1), B(g_2), B(g_9)\}$ and $BG_2 = \{B(g_6), B(g_{10})\}$. Corresponding to the sets BG_1 and BG_2 are partitions of primary inputs $PI_1 = \{p_1, p_2\}$ and $PI_2 = \{p_3, p_4\}$. If a combination of primary input values satisfies all conditions in a group (*e.g.* BG_i evaluates to 1) the combination is said to be a conflict free multiple primary input assignment. When $p_1 = 0$ and $p_2 = 1$, then all expressions in BG_1 ($B(g_1)$, $B(g_2)$, and $B(g_9)$) become 1. Hence, the assignment $p_1 = 0$ and $p_2 = 1$ is a conflict free multiple input assignment. $B(g_6)$ and $B(g_7)$ can be also set to 1 by assigning $p_3 = 1$ and $p_4 = 1$. This assignment is also conflict free multiple input assignment. All gates in G are hence blocked by assigning $p_1 = 0$, $p_2 = 1$, $p_3 = 1$, and $p_4 = 1$.

Since obtaining expressions for the necessary and sufficient conditions for each gate in G is hard, conflict free multiple inputs are determined without obtaining expressions. Firstly, the primary inputs are partitioned into independent groups. For example, the conditions to block gates in BG_1 depend only on the inputs p_1 and p_2 and the conditions to block gates (g_6, and g_{10}) in BG_2 depend only on the inputs p_3 and p_4. Therefore circuit $C3$ has two independent groups of primary inputs $Group_1 = \{g_1, g_2, g_9\}$ $Group_2 = \{g_6, g_{10}\}$. In this paper, primary inputs can be partitioned into each independent group by identifying *strongly connected components* [CLR90].

Next, a virtual gate vg_i is added for each gate g_i ($g_i \in G$). Inputs of vg_i are connected to inputs pl ($pl \in TPI'$) of g_i if the controlling value of g_i can indeed be applied to pl via primary input assignment. The gate type of vg_i is determined such that the output of vg_i is set 1 if and only if g_i is blocked. This implies that vg_i is a NAND gate if g_i is a AND/NAND gate. And it is an OR gate if g_i is an OR/NOR gate. The expression at the output of vg_i corresponds to $B(g_i)$. In the circuit $C2$ (Figure 4), The AND gate g_1 has two inputs p_1 and p_2, in TPI'. Two input NAND gate vg_1 is then added for g_1, since both inputs p_1 and p_2 are controllable to 0 (the controlling value of g_1). p_1 and p_2 are connected to two inputs of vg_1.

The output of vg_1 evaluates to 1 when p_1 or p_2 is assigned 0. A buffer vg_2 (*i.e.* a 1-input OR gate) is added for g_2 that has only one input p_2 in TPI'. The output of vg_2 evaluates to 1 when p_1 is assigned 1. Two input OR gate vg_9 is added for g_9 because the controlling value of g_9 is 1 and two inputs in TPI' of g_9 are both controllable to 1.

The output of all virtual gates vg_i in the same group is connected to a k input AND gate (k is the number of virtual gates in the group). In Figure 4, vg_i ($i = 1, 2, 6, 9, 10$) are virtual gates for g_i respectively. A strongly connected group $Group_1$ contains three gates (g_1, g_2, and g_9). The inputs of three input AND gate BG_1 are connected to the outputs of corresponding virtual gates (vg_1, vg_2, and vg_9). Virtual gates vg_1, vg_2, and vg_9 depend on primary inputs p_1 and p_2. Hence if there exist a combination of primary inputs p_1 and p_2 that implies 1 at the output of BG_1, the combination of primary inputs is multiple input conflict free assignment. Finally, ATPG techniques are used to search for primary input assignments that imply 1 at the output of the k input AND gate. This is repeated for each group.

3.4 Iterative improvement heuristic algorithm

3.5 Iterative Improvement Heuristic

The primary inputs that are not identified as independent, or assigned conflict free (single/multiple) values, are assigned binary values to maximize blocking. This is achieved by using Kernighan and Lin [KL70] iterative improvement bipartitioning algorithm (or, *K-L algorithm*). Associated with each gate $g_a \in G$ is a weight $w(g_a)$, the number of lines in the forward cone of g_a, that are not yet assigned binary values. The objective of this algorithm is the maximization of the following function:

$$F(X_j) = \sum_a B(g_a) \times w(g_a) \tag{2}$$

where X_j is the vector applied to the primary inputs and $B(g_a)$ is a function that evaluates to 1 if g_i is blocked otherwise it evaluates to 0. Primary inputs are divided into two partitions, Ψ_0 and Ψ_1, such that Ψ_0 (Ψ_1) contains primary inputs assigned 1(0). The primary inputs that are already assigned binary values by conflict free assignments are assigned to Ψ_0 or Ψ_1, depending on the value assigned. Initially, the other primary inputs are placed into $\Psi_0(\Psi_1)$ arbitrarily. In each iteration, a primary input p_i, that was not assigned a binary value by conflict free assignment, is moved from Ψ_v to $\Psi_{\bar{v}}$ ($v = 0, 1$). In other words, the bit corresponding to p_i in the initial vector X_{init} is flipped (denoted by $X_{p_i \leftarrow \overline{p_i}}$) and the gain

$(\Delta F(X_{p_i \leftarrow \overline{p_i}}))$ due to flipping is calculated, where $\Delta F(X_{p_i \leftarrow \overline{p_i}})$ is given by:

$$\Delta F(X_{p_i \leftarrow \overline{p_i}}) = F(X_{init}) - F(X_{p_i \leftarrow \overline{p_i}}). \tag{3}$$

The migration is repeated until $F(X_{p_i \leftarrow \overline{p_i}})$ does not increase any longer.

4 Scan Input Assignment to Minimize the Number of Transitions

The contents of the scan register during scan shifting are determined by the values captured in response to the test vector applied and the new test being scanned in. The captured value is always fully specified. In contrast, the next test vector generated may have don't cares. The switching activity in the CUT during test application can be reduced by carefully assigning these don't cares. Hence, combinational test vectors with the maximum number of don't cares in their state input part are more suitable for minimizing transitions in circuit lines. An existing implementation of PODEM [Goe81] has been modified to generate combinational test vectors that have minimum number of specified bits at the state inputs. New cost functions: controllability cost, observability cost, and test generation cost, are defined to direct the ATPG to generate such tests. These cost functions are calculated only once during the entire test generation process in a preprocessing step.

The controllability cost $Cv(l)$ is the minimal number of state inputs that need to be assigned to set the line l to a desired value v. In order to detect the stuck-at-v fault at line l, first the target fault is activated by setting l to $\bar{v}$. Subsequently the activated fault effect is propagated to one of primary or state outputs. Therefore, generating a test vector consists of many *line-justifications*. In PODEM [Goe81], a value v at a line l is justified by mapping v to input (primary and state) assignments by backtracing to inputs. The controllability cost functions guides the ATPG to select the backtrace paths that require minimum number of state input assignments, whenever there is a choice of several paths to backtrace from the target line to the inputs. The controllability cost is given by

$$Cv(l) = \begin{cases} 0 & \text{if } l \textit{ is a primary input} \\ 1 & \text{if } l \textit{ is a state input} \\ |\bigcup_{l_j} \{\ p \mid p \textit{ are min state inputs required to set } l_j \textit{ to } \bar{c_a}\ \}| & \text{if } v = \bar{c_a} \oplus i_a \\ \min_j\{Cc(l_j)\} & \text{if } v = c_a \oplus i_a \end{cases} \tag{4}$$

where l_j are the inputs of the gate g_a with output line l, and c_a and i_a are the controlling value and inversion of g_a, respectively.

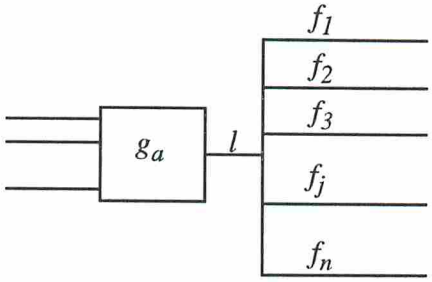
Assume that a scan register consists of n flip-flops, $s_1, s_2, ..., s_n$, and the state input part of the vector, U_1^{j+1}, U_2^{j+1},..., U_n^{j+1}, is scanned into the scan register through s_1 and scanned out of s_n. The value U_n^{j+1} for s_n reaches the destination after it passes through $n-1$ flip-flops. Thus, in the worst case it may cause transition at all n flop-flops and the combinational circuit during the scan-in. In contrast, the value U_1^{j+1} for s_1 reaches the destination in one cycle. Hence, only s_1 and the gates in its forward cone may switch during scan-in of U_1^{j+1}. To take this difference between the state inputs s_q and s_r $(q \neq r)$ into account, the controllability cost function can be modified by assigning a weight $q \times C$ to the state input s_q, where C is a constant and q is the position of the flip-flop s_q in the scan register.

During test generation, all gates whose output values are currently unknown and at least one of whose gate inputs has the fault effect belong to *D-frontier* [Goe81]. In the proposed ATPG, a gate that is likely to need the least number of state input assignments to propagate the fault effect at its input is selected from D-frontier repeatedly, until the fault effect reaches one or more primary or scan outputs. The observability cost function described below serves as a selection criterion to achieve this objective. A gate in the D-frontier whose input with fault effect has minimum observability cost is selected each time. The observability cost function $O(l)$ indicates the minimum number of state inputs that need to be assigned binary values to propagate a value at line l to an observation point. Observability cost is calculated for every line in the CUT, starting from primary and state outputs and traversing the circuit backward toward primary and state inputs. The observability cost is given by

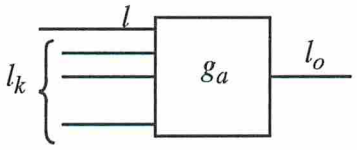
$$O(l) = \begin{cases} 0, \text{ if } l \textit{ is an observation point} \\ \min_j\{O(f_j)\}, \text{ if } l \textit{ is a fanout stem} \\ |\bigcup_{l_k \neq l} \{\ p \mid p \textit{ are min state inputs required to set } l_k \textit{ to } \bar{c}_a\ \}| + O(l_o), \textit{otherwise} \end{cases} \tag{5}$$

where f_j are fanout branches of line l and l_k are inputs of gate g_a that is driven by l, and l_o is the output of g_a (as shown in Figure 5).

The objective of the proposed ATPG is to generate a test vector that has the maximum number of unspecified state inputs. As described earlier, in order to generate the vector to detect the *stuck-at-*$\bar{v}$ fault at line l, the fault should be activated by setting l to v, then the fault effect should be propagated to one or more outputs. The controllability cost and observability cost correspond to the former and the latter, respectively. Thus a test vector that specifies boolean values at a minimum number of state inputs is obtained in *fanout free circuits* by using the two proposed cost functions for (a) selection of objective, and (b) directing the backtrace procedures in PODEM [Goe81]. In circuits with fanouts,



(a) Fanout Branches



(b) A Gate g_a with Output l_o and Inputs l and l_k

Figure 5: Gate Model for Computing Observability Cost Functions

a test vector that is generated by the proposed ATPG may not have minimum number of specified states because only the approximate value of both their cost functions is computed. Furthermore, even if the exact value of cost functions is computed, conflicts between multiple line justifications may necessitate specification of values at additional state inputs.

In general, a test vector generated by the proposed ATPG has many don't cares at state inputs which can be assigned to minimize the switching activity in the circuit. Again this can be performed by using the K-L algorithm [KL70]. This procedure is similar to that used to find the primary input pattern that can block the most gates (Section 3.5), except that the objective function and the gain are defined somewhat differently.

Since the time complexity of the K-L algorithm is mainly determined by the number of state inputs, the run time for circuits that have many state inputs (say, greater than 50) may not be acceptable. A simple heuristic can be used for these circuits instead of the K-L algorithm without increasing the number of transitions significantly. Assume that n consecutive scan flip-flops in a scan chain are assigned don't cares in a test vector and they are flanked by two flip-flops s_i and s_j that are assigned the same binary value v in the test vector. The simple heuristic assigns v to all these don't cares. If s_i and s_j are assigned different values, v and $\overline{v}$, in the test vector, then the simple heuristic chooses a value randomly and assigns it to these don't cares. Experimental results with these two state input don't care assignment algorithms show that tests generated by the proposed

ATPG which uses the simple heuristic cause, on an average, the same number of transitions as those generated by the proposed ATPG which uses the K-L algorithm. While the run time is lower for the simple heuristic, the length of the test sequence obtained is typically higher. This is due to the fact that the simple heuristic tries to assign the same binary values to neighboring scan flip-flops. Many scan inputs are hence assigned the same binary value in most of the test vectors resulting in lower fault coverage. Hence, to achieve the same fault coverage, longer test sequences are required. Enhancements to reduce test sequence length are under investigation. One of these enhancements is to find a boundary that divides a set of neighboring state inputs, which are assigned don't care values and flanked by different binary values v and $\overline{v}$, into two subsets; where one subset will be assigned v and the other subset $\overline{v}$. This boundary will change for every test vector, depending on values assigned to state and primary inputs. This will give more variation to the values assigned to state inputs, thereby increasing fault coverage for a given test length.

Tests for some faults need many state inputs to be specified. Typically, any assignment of don't care state inputs does not reduce transitions effectively, if the number of don't cares is very low. However, these don't cares can be assigned to detect other faults which also need specific values at many state inputs. Such an assignment can reduce the number of tests in the sequence without increasing the average number of transitions. If the number of specified state inputs in a test generated using the abovementioned ATPG is greater than a predefined number (*e.g.* 80%), the generated test is discarded and the fault is moved to a *high cost fault list*, that is initially empty. Target faults are taken from the regular fault list until the regular fault list is empty. After the regular fault list is empty, target faults are selected from the high cost fault list randomly and a test for the fault is generated by the normal PODEM, rather than the proposed ATPG. If the test has any don't care at state inputs, the normal PODEM selects a secondary fault and tries to generate a test for the fault. If the ATPG fails to generate the test, it selects another fault and tries to generate the test for that fault. This procedure is repeated until the ATPG tries all faults in the high cost fault set or all the don't cares are assigned binary values.

The proposed test generation algorithm can now be described as follows.

1. Perform uncontrollability analysis (Section 3.2).

2. Assign appropriate binary values to primary input don't cares that can be assigned in a conflict free manner, either one at a time or as a group (Section 3.2, 3.3). These assignments are used for all vectors and during all scan shifting clocks.

3. Identify all independent primary inputs (Section 3.1.3). Assign binary values to these inputs once for each vector, as described in Section 3.1.3.

4. Apply the bipartitioning algorithm to find an assignment for the remaining primary input don't cares during scan shifting to block most gates in the CUT (Section 3.5).

5. a) If the regular fault list is empty, then go to Step 6.

   b) Select a target fault from the regular fault list and generate a combinational test V^{i+1} using the proposed ATPG that assigns minimum number of state inputs. If the generated test has fewer don't cares in state input part than a predefined number, then move the target fault to the high cost fault list and go to Step 5 a).

   c) Assign remaining unspecified primary inputs (which are independent primary inputs identified in Step 3) according to their previous values using the procedure outlined in [WG94]. (The values assigned at this time are kept constant during scan shifting of vector V^{i+1}.)

   d) Apply either the simple heuristic described above or the K-L algorithm to assign binary values to the remaining state input don't cares to minimize the number of transitions.

   e) Perform fault simulation and drop detected faults from the fault list and go to Step 5 a).

6. a) If there are no more undetected faults in the high cost fault list, then exit. Otherwise, select a target fault from the high cost fault list.

   b) Generate a combinational test for the selected target fault by using a normal PODEM.

   c) If the generated test has any don't cares in state input parts, select a secondary target fault from the high cost fault list and generate a test for it by assigning binary value to don't cares. Repeat this step until there are no remaining don't cares in the state input part or all faults in the high cost fault list are tried.

   d) Go to Step 6 a).

5 Experimental Results

Table 1 shows the experimental results for full scan versions of ISCAS89 sequential benchmark circuits. The experiments were performed on a Sparcstation 4 with 32Mbytes of memory. Table 1 compares the results obtained by the normal implementation of PODEM (columns labeled *Norm.*) and those obtained by the proposed ATPG (columns *labeled*

Table 1: Experimental Results

Circuit	*Fault Coverage*		*Avr. # Transitions*		*# Test Vectors*		*Effect.*	*Test Gen. Time (sec)*	
	Norm.	*Prop.*	*Norm.*	*Prop.*	*Norm.*	*Prop.*	*Reduct.*	*Norm.*	*Prop.*
s208	100	100	40.1	14.3(64)	38	55	.51	.2	.5
s298	100	100	105.4	46.0(56)	47	68	.63	.3	.6
s344	100	100	109.5	50.0(54)	30	72	1.1	.3	.7
s386	100	100	112.5	83.0(26)	82	86	.77	.7	2.4
s420	100	100	68.0	14.2(79)	65	114	.37	.6	1.7
s444	99.1	99.1	135.5	43.6(67)	38	92	.78	.5	1.2
s510	100	100	141.1	114.4(19)	69	80	.94	.9	2.9
s526	100	100	178.4	74.9(58)	78	142	.76	1.1	2.4
s641	100	99.7	206.6	22.9(89)	83	136	.21	3.1	2.7
s713	97.5	97.0	221.4	24.2(89)	69	118	.19	11.4	13.2
s820	100	100	283.0	184.5(35)	146	192	.86	3.8	12.2
s832	99.8	99.5	286.9	185.5(35)	156	184	.76	4.4	12.2
s838	100	100	106.5	13.6(87)	131	234	.22	3.2	9.6
s953	100	100	205.7	27.8(86)	114	161	.19	4.1	6.6
s1196	100	99.5	346.8	47.0(86)	180	219	.16	8.6	12.3
s1423	98.6	98.1	477.0	122.2(74)	85	237	.71	10.1	14.9
s1488	100	100	500.9	393.8(21)	164	188	.90	11.4	35.1
s1494	99.9	99.5	506.4	403.9(21)	159	185	.93	11.5	38.1
s5378	99.3	99.0	1545.8	321.5(79)	315	739	.49	158	401
s9234	93.1	93.1	2785.0	716.1(74)	351	936	.68	613	1467

Prop.). Fault coverages, average number of transitions, total number of test vectors in test sequences, and test generation times are presented. The number of transitions was counted under zero delay model. Zero delay model is used exclusively by test generators and fault simulators for stuck-at faults. Under the zero delay model, no hazards are considered at the circuit lines and the heat dissipation is assumed to be mainly due to $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions at the circuit lines. The use of zero delay model is justified by the observation that the heat dissipation estimated under this model has a high correlation with that under the general delay model [SGDK92]. Effective Reduction column (labeled *Effect. Reduct.*) shows the total number of transitions in each CUT during entire scan based test when the test vectors generated by the proposed ATPG are applied, as a fraction of those occuring due to the application of the tests generated by the normal PODEM. The simple heuristic was used to assign binary values to unspecified state inputs. In normal PODEM, all don't cares were assigned randomly. Note that the fault coverage obtained by using two ATPGs are *almost identical.*

The results demonstrate that the tests generated using the proposed ATPG decreased

the average number of transitions by 21% to 89%. The numbers shown in parenthesis (under the column labeled *Avr. # Transitions — Prop.*) denote the percentage increase/decrease over the results obtained by the normal PODEM. These results clearly show that the switching activity during test application can be reduced significantly by using the proposed ATPG. It should be noted that larger reductions occur in circuits that have more primary and state inputs (s641, s713, s838, and s913). This is due to the fact that in such circuits, more primary inputs can be assigned binary values to block more gates that may otherwise have transitions. Furthermore, for such circuits, most of test vectors generated by the proposed ATPG have many don't cares in state input part. If these don't cares are carefully assigned to minimize transitions, only a small portion of state inputs will have transitions during scan shifting resulting in lower average number of transitions. The number of transitions also depends on the circuit structure. Note that s420 and s510 have the same number of primary inputs and state inputs. However the results of two circuits are significantly different. This is due to the difference in the structures of these circuits. 14 primary input can be assigned in a conflict free manner in s420 while only 5 primary inputs can be assigned in a conflict free manner in s510.

The number of vectors in test sequences increased by a factor of 1 to 2.8. To take into account both the reduction in average number of transitions and the increase in test length, we present the effective reduction in the Table. This shows that the reduction in the average number of transitions are more than sufficient to compensate for the increase in the test sequence for all circuits except s344. For s344, only two primary input don't cares can be assigned a binary value to block transitions at state inputs and all other primary inputs are independent of state inputs. The results for these circuits can be improved at the cost of longer run time by using the K-L algorithm to assign state input don't cares. The effective reductions for s344, when the K-L algorithm is used instead of the simple heuristic, is 0.56 while the run time is 43.8 seconds. This improvement in the effective reduction is due to the decrease in test sequence length (46) rather than the decrease in average number of transitions (44.7). The K-L algorithm outperformed significantly the simple heuristic in the circuits (such as s344) which have small number of state inputs and very few of whose primary inputs can be used to block transitions at state inputs. For these circuits, the run time of the proposed ATPG using the K-L algorithm is within a few tens of seconds higher (even though it is an order of magnitude longer) than that of the simple heuristic.

Finally, the run time of the proposed ATPG (with the simple heuristic) is a factor of about 2 to 3 higher than that of the the normal PODEM, for the most circuits. Considering that the test sequence length also increases in a similar proportion, these data clearly demonstrate that the time complexity of the operation to reduce transitions in the CUT is very low.

6 Conclusion

An ATPG technique is proposed that reduces heat dissipation during testing sequential circuits that have full-scan. The objective is to permit safe and inexpensive testing of low power circuits and bare dies that would otherwise require expensive heat removal equipment for testing at high speeds. The proposed ATPG exploits all possible don't cares that occur during scan shifting, test application, and response capture to minimize switching activity in the CUT. Furthermore, an ATPG that minimizes the number of state inputs that are assigned specific binary values, has been developed. In order to guide the ATPG to generate test vectors that have maximum number of don't cares at state inputs, new cost: controllability and observability cost functions have been defined. These cost functions are used to guide backtrace and to select objectives from D-frontier. Don't cares at primary inputs during scan shifting and capture are used to block gates that may cause transition during scan shifting and don't cares at state inputs are assigned binary values that cause the minimum number of transitions.

The proposed algorithm has been implemented and the generated tests are compared with those generated by a simple PODEM implementation for full scan versions of ISCAS 89 benchmark circuits. Tests generated by the proposed ATPG decreased the average number of transitions during test from 22% to 89% with higher reductions occuring in circuits that have more primary and state inputs. These reductions in circuit transitions were achieved with a reasonable increase in the time complexity of the ATPG (factor of 2 to 3). We believe that the increase in run time is mainly due to the longer test sequence length and are investigating enhancements to reduce the test sequence length.

References

[ABF90] M. Abramovici, M. A. Breuer, and A. D. Friedman. *Digital Systems Testing and Testable Design.* Computer Science Press, New York, N.Y., 1990.

[AKR91] M. Abramovici, J. J. Kulikowski, and R. K. Roy. The Best Flip-Flops to Scan. In *Proceedings IEEE International Test Conference*, pages 166–173, October 1991.

[CLR90] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithm.* The MIT Press, Cambirdge M.A., 1990.

[Goe81] P. Goel. An Implicit Enumeration Algorithm to Generate Tests for Combinatio nal Logic Circuits. *IEEE Trans. on Computers*, Vol. C-30(3), March 1981.

[Gol82] S. W. Golomb. *Shift Register Sequences.* Aegean Park Press, Laguna Hills, C.A., 1982.

[Kee92] D. C. Keezer. Bare Die Testing and MCM Probing Techniques. In *Proceedings-Multi Chip Module Conference*, pages 20–23, March 1992.

[KL70] B. W. Kernighan and S. Lin. An Efficient Heuristic Procedure for Partitioning Graphs. *Bell System Techniacl Journal*, 49:291–307, Febrary 1970.

[Par92] R. H. Parker. Bare Die Test. In *ProceedingsMulti Chip Module Conference*, pages 24–27, March 1992.

[SGDK92] A. Shen, A. Ghosh, S. Devadas, and K. Keutzer. On Average Power Dissipation and Random Pattern Testability of CMOS Combinational Logic Networks. In *Proceedings IEEE International Conference on Computer-Aided Design*, pages 402–407, November 1992.

[TPCD94] C.-Y. Tsui, M. Pedram, C.-A. Chen, and A. M. Despain. Low Power State Assignment Targeting Two- and Multi-level Logic Implementation. In *Proceedings IEEE International Conference on Computer-Aided Design*, pages 82–87, 1994.

[WG94] S. Wang and S. K. Gupta. ATPG for Heat Dissipation Minimization During Test Applicaion. In *Proceedings IEEE International Test Conference*, pages 250–258, October 1994.

[Zor] Y. Zorian. Private Communication.

[Zor93] Y. Zorian. A Distributed BIST Control Scheme for Complex VLSI Devices. In *Proceedings VLSI Testing Symposium*, pages 4–9, 1993.