On the Generation of Scan-Based Test Sets with Reachable States for Testing under Functional Operation Conditions

Irith Pomeranz*
School of ECE
Purdue University
W. Lafayette, IN 47907

Abstract

Design-for-testability (*DFT*) for synchronous sequential circuits allows the generation and application of tests that rely on non-functional operation of the circuit. This can result in unnecessary yield loss due to the detection of faults that do not affect normal circuit operation. Considering single stuck-at faults in full-scan circuits, a test vector consists of a primary input vector U and a state S. We say that the test vector consisting of U and S relies on non-functional operation if S is an unreachable state, i.e., a state that cannot be reached from all the circuit states. Our goal is to obtain test sets with states S that are reachable states. Given a test set C, the solution we explore is based on a simulation-based procedure to identify reachable states that can replace unreachable states in C. No modifications are required to the test generation procedure and no sequential test generation is needed. Our results demonstrate that the proposed procedure is able to produce test sets that detect many of the circuit faults, which are detectable using scan, and practically all the sequentially irredundant faults, by using test vectors with reachable states. The procedure is applicable to any type of scan-based test set, including test sets for delay faults.

Categories & Subject Descriptors: B.8.1 Reliability, Testing, and Fault-Tolerance

General Terms: Reliability

Keywords: functional tests, reachable states, scan design.

* Research supported in part by NSF Grant No. CCR-0098091 and in part by SRC Grant No. 2001-TJ-950.

DAC 2004, June 7-11, 2004, San Diego, California, USA

1. Introduction

Design-for-testability (*DFT*) for synchronous sequential circuits [1]-[11] allows tests to be generated and applied to the circuit with *DFT* logic, which may not be applicable to the circuit without *DFT*. Such tests use non-functional operation of the circuit, enabled by the *DFT* logic, to detect faults. This has several implications. (1) Faults detected during non-functional operation may not affect the functional operation of the circuit. If a fault can only be detected during non-functional operation, the fault is undetectable or possibly even redundant, and its detection may cause unnecessary yield loss. Considering delay faults, faults that can only be detected during non-functional operation do not affect the circuit performance during normal operation. This, again, can lead to unnecessary yield loss. This problem is discussed in [12]. (2) Non-functional operation may result in a higher power dissipation than normal operation. It may be necessary to avoid this situation to preserve battery power or to avoid damage due to a higher power dissipation than the circuit is designed for.

To simplify the discussion we consider single stuck-at faults in synchronous sequential circuits with full-scan (our results are applicable to other fault models, including delay faults as discussed later). A test vector for a full-scan circuit consists of a subvector U that specifies the values of the primary inputs, and a subvector S that specifies the values of the present-state variables. We denote a test vector by $U \cdot S$. We say that a test vector $U \cdot S$ can be applied during normal circuit operation if S is reachable from all the circuit states. We verify this property by checking whether S is reachable from the all-unspecified state, i.e., by checking whether it is possible to find a primary input sequence that takes the circuit from the all-unspecified state to S. If S is not reachable from the all-unspecified state, then S may not be reachable from the initial state of the circuit during normal operation. For the reasons discussed above, it is better to detect faults using test vectors that include reachable states.

To obtain test vectors whose state subvectors are reachable states, a possible solution is to perform test generation such that only reachable states are used in every test vector. However, this requires modifications to the test generation procedure, including identification of reachable states during test generation. It is also possible

to identify reachable states in advance; however, the number of reachable states may be too large to allow this process to be carried out effectively. Performing sequential test generation is another way to ensure the use of reachable states during test application. In either case, the resulting test generation process will be more complex.

The solution we explore in this work does not require any modifications to the test generation procedure, and does not require any sequential test generation. The proposed procedure accepts a test set C generated for the full-scan circuit and produces a new test set CR (where the R stands for reachable). Using a simulation-based process, the procedure checks for every test vector $U \cdot S \in C$ whether S is a reachable state. During this process, the procedure also collects a set of fully-specified reachable states Ψ that are as close to S as possible. If S is reachable then $U \cdot S$ is included without modification in the new test set CR. If S is unreachable, then the states in Ψ are used for defining new test vectors, which are included in CR instead of $U \cdot S$. The resulting test set CR contains only test vectors with reachable states.

The fault coverage achieved by CR may be lower than the fault coverage of C for several reasons. (1) Some faults are only detectable using test vectors with unreachable states. An example is shown in Figure 1 using the state diagram of a synchronous sequential circuit. The fault changes the output value under the state-transition from S_1 to S_2, which is marked with a dashed line in Figure 1. The fault can only be detected using a test vector that has S_1 as its state subvector. However, S_1 is not a reachable state since it has no incoming state-transitions from S_2, S_3, S_4 and S_5. Such a fault is undetectable in the non-scan sequential circuit. (2) The proposed procedure does not perform an exhaustive search for reachable states. Thus, it is possible that the state S in a test vector $U \cdot S$ would be reachable but the procedure would not identify this fact. As a result, faults detected by $U \cdot S$ may not be detected by CR. (3) The proposed procedure only considers test vectors of the form $U \cdot P$ (where P is a state) to replace a test vector of the form $U \cdot S$, i.e., it only replaces the state subvector but not the primary input subvector. It is possible that some faults detected by $U \cdot S$ would not be detectable by test vectors of the form $U \cdot P$ but would be detectable by test vectors of the form $V \cdot P$ for a different primary input subvector V.

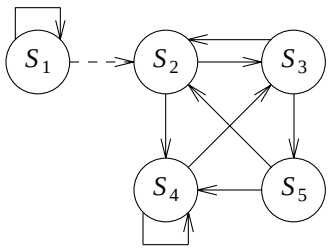

Figure 1: An example state diagram

We investigate experimentally the extent to which the proposed procedure is able to produce single stuck-at test sets with reachable states. Our results demonstrate that the proposed procedure is able to produce test sets that detect many of the circuit faults, which are detectable using scan, and practically all the sequentially irredundant faults, by using test vectors with reachable states. We stress that the proposed procedure does not change the structure of the test set, and that each test vector still has the form $U \cdot S$. However, the states S are reachable states after application of the proposed procedure.

During test application, it is possible to use only CR to ensure that only faults that affect the functional operation of the circuit are detected. For added reliability, it is also possible to define a test set that consists of CR followed by tests out of C to detect the faults that cannot be detected by CR. Such a test set would detect as many faults as possible using reachable states and the remaining faults using unreachable states. We denote this test set by CmR (mR stands for maximal reachable).

Although we consider only stuck-at test sets with test vectors of the form $U \cdot S$, the proposed procedure is also applicable to test sets with two-pattern tests used for detecting delay faults. In this case, the proposed procedure can be applied to the state included in the first pattern of every two-pattern test of a broadside test set. In the case of skewed-load, the state of the second pattern needs to be a reachable state as well. It is also applicable when a scan operation is followed by a sequence of primary input vectors applied in functional mode. In this case, the proposed procedure can be applied to the scan-in states.

It is important to note the following point. In [12], tests that consist of a scan operation followed by several primary input vectors are used to obtain broadside transition fault tests that are better at ensuring that fault detection occurs during functional operation. This is based on the intuition that a longer sequence of primary input vectors applied after the scan operation is more likely to cause the circuit to enter its functional operation mode (what we call here its reachable state space). The example of Figure 1 shows that this is not necessarily true. Starting from state S_1 it is possible to apply a primary input sequence of arbitrary length that will leave the circuit in an unreachable state, S_1. In more complex state diagrams there are even more opportunities to stay in an unreachable state after a (long) sequence of primary input vectors. The use of reachable states to define tests that use functional operation of the circuit resolves this issue.

The paper is organized as follows. In Section 2 we describe the simulation-based procedure for finding reachable states. In Section 3 we describe the procedure for modifying a test set C into a test set CR with reachable states. Experimental results are given in Section 4.

2. Finding reachable states

In this section we describe the procedure we use for checking whether a state S, which is part of a combinational test vector $U \cdot S$, is reachable in the sequential circuit. The same procedure is used for collecting a set of fully-specified states Ψ, which are reachable in the sequential circuit and have the smallest possible Hamming distances from S. If S is not reachable, the states in Ψ will

be used for replacing S to form new combinational test vectors that potentially detect the same faults but contain reachable states. We demonstrate the procedure by considering ISCAS-89 benchmark circuit $s27$.

A combinational test set C for $s27$ is shown in Table 1. The circuit has four primary inputs and three state variables. Each combinational test vector in C has the form $U_i \cdot S_i$, where U_i is the four-bit subvector applied to the primary inputs, and S_i is the three-bit subvector applied to the present-state variables.

Table 1: Test set for $s27$

i	U_i	S_i
0	0000	011
1	1001	010
2	0100	110
3	0111	001
4	1101	011
5	1010	000

Considering the test vector $U_1 \cdot S_1 = 1001 \cdot 010$, we need to check whether $S_1 = 010$ is reachable. We use a simulation-based procedure to compute an input sequence that brings the circuit from the all-unspecified state xxx to $S_1 = 010$, or to a state which is as close to S_1 as possible. We initially set $P = xxx$. Starting from P, we construct the input sequence by considering at most L_{MAX} consecutive time units, where L_{MAX} is a preselected number. At every time unit u, we apply a preselected number $N_{RAND} = 5$ of random vectors $V_0, \cdots, V_4$. We then select one of the vectors, V_r, to be included in the input sequence. The selection is based on an analysis of the next-states obtained under $V_0, \cdots, V_4$. The selected next-state becomes the present-state at time unit $u+1$.

At time unit $u = 0$ we have $P = xxx$ and we consider the five random vectors V_r, $0 \le r \le 4$, shown in Table 2(a). Under column Q_r we show the next-state obtained when V_r is applied to the primary inputs and the circuit is in present-state $P = xxx$, $0 \le r \le 4$. For every next-state Q_r we compute the following parameters.

Table 2: Constructing an input sequence for S_1

(a) $u=0$

r	V_r	Q_r	*n_spec*	*new_state*	*dist*
0	0100	0x1	2	1	2
1	0101	0x1	2	1	2
2	1101	101	3	1	3
3	1100	101	3	1	3
4	0100	0x1	2	1	2

(b) $u=1$

r	V_r	Q_r	*n_spec*	*new_state*	*dist*
0	0110	000	3	1	1
1	0110	000	3	1	1
2	0000	001	3	1	2
3	0101	001	3	1	2
4	1010	100	3	1	2

The first parameter we compute is the number of specified values in Q_r, denoted by *n_spec* (Q_r). Since the combinational test vectors in C are fully specified, our first priority is to bring the circuit to a fully-specified state. We achieve this goal by selecting a state that has the highest value of *n_spec* (Q_r) [13].

The second parameter is denoted by *new_state* (Q_r). We set *new_state* $(Q_r) = 1$ if Q_r was not reached before along the input sequence being computed. Otherwise, *new_state* $(Q_r) = 0$. We prefer Q_r to be a new state since this maximizes our ability to explore the space of reachable states.

The third parameter we compute is the Hamming distance between S_1 and Q_r, denoted by *dist* (S_1, Q_r). The Hamming distance is incremented by one for every bit where S_1 is different from Q_r. This includes bits where Q_r is unspecified (S_1 is completely specified). We prefer to select a state for which *dist* (S_1, Q_r) is minimum as a heuristic to guide us towards reaching S_1. In addition, if S_1 is not reached, it is advantageous to obtain states that are as close to it as possible.

The parameters *n_spec* (Q_r), *new_state* (Q_r) and *dist* (S_1, Q_r) for the five vectors considered at time unit $u = 0$ are shown in Table 2(a). We note that V_2 and V_3 have the highest numbers of specified values. They both result in new next-states that have the same distance from S_1. Therefore, we arbitrarily select V_2 with $Q_2 = 101$. This becomes the present-state P at time unit $u = 1$.

At time unit $u = 1$ we have $P = 101$ and we consider the five random vectors V_r, $0 \le r \le 4$, shown in Table 2(b). All the next-states have the same number of specified values and they are all new states. We select between V_0 and V_1 arbitrarily to use V_0 with *dist* $(S_1, Q_0) = 1$.

At time unit $u = 2$ we have $P = 000$ and we consider five random vectors V_r, $0 \le r \le 4$. We find that under $V_2 = 1011$, the next state is $Q_2 = 010$, equal to S_1. Therefore, the procedure terminates with the indication that S_1 is reachable. The three fully-specified present-states reached during this process are included in the set Ψ. We have $\Psi = \{101, 000, 010\}$.

Applying the same process to the other test vectors in Table 1 we find that S_0 is reachable at the end of a primary input sequence of length 18, S_3 is reachable at the end of a sequence of length 2, S_4 is reachable at the end of a sequence of length 9, and S_5 is reachable at the end of a sequence of length 2. The only state that cannot be reached is S_2. For S_2 we obtain an input sequence of length $L_{MAX} = 20$ that visits the states shown in Table 3 (we only show in Table 3 fully-specified new states when they are reached). Under column u we show the time unit. Under column V_u we show the input vector at time unit u. Under column P_u we show the present-state at time unit u. We then show the three parameters related to P_u. The states reached in Table 3 are included in the set Ψ. They will be used later to define new test vectors that will replace $U_2 \cdot S_2$.

We refer to the procedure demonstrated above as Procedure 1. The complexity of Procedure 1 is determined by the need to simulate up to $N_{RAND} L_{MAX}$ input vectors. Parallel pattern simulation can be used to speed up Procedure 1.

Table 3: Input sequence for $S_2 = 110$

u	V_u	P_u	*n_spec*	*new_state*	*dist*
1	1101	100	3	1	1
2	0110	101	3	1	2
3	0011	000	3	1	2
4	0101	010	3	1	1
5	0010	011	3	1	2
10	1111	001	3	1	3

3. Obtaining test vectors with reachable states

In this section we describe the procedure that accepts a test set C where some of the states may be unreachable and produces a test set CR where all the states are reachable. As before, a test vector is represented as $U \cdot S$.

The test set CR is obtained as follows. For every test vector $U_i \cdot S_i \in C$, if S_i is reachable then $U_i \cdot S_i$ is copied to CR. If S_i is not reachable, we add to CR a set of test vectors $R(U_i \cdot S_i)$ to replace $U_i \cdot S_i$. The set $R(U_i \cdot S_i)$ may contain test vectors that do not detect any new faults. As a result, the number of test vectors in CR may be larger than necessary. We remove unnecessary test vectors from CR by performing fault simulation with fault dropping. A test vector that does not detect any fault during this process is removed from CR.

We first consider the example of $s27$. We then describe the general procedure.

3.1. Example

For the test set of $s27$ shown in Table 1, we found earlier that the states S_0, S_1, S_3, S_4 and S_5 are reachable. The corresponding test vectors $U_i \cdot S_i$ for $i = 0,1,3,4,5$ are copied from C to CR. For $U_2 \cdot S_2$ we define a set of test vectors $R(U_2 \cdot S_2)$ as follows.

Every test vector in $R(U_2 \cdot S_2)$ has the form $U_2 \cdot P_u$, i.e., we maintain the primary input subvector U_2 of $U_2 \cdot S_2$ in every test vector included in $R(U_2 \cdot S_2)$. For P_u, we use the states included in Ψ during the application of Procedure 1 to S_2. These states are the present-states shown in Table 3.

We consider the states P_u in Ψ by order of increasing distance $dist(S_2, P_u)$. Thus, we start with $P_1 = 100$ and $P_4 = 010$ at distance 1; we then consider $P_2 = 101$, $P_3 = 000$ and $P_5 = 011$ at distance 2; and finally we consider $P_{10} = 001$ at distance 3. For every P_u we add to $R(U_2 \cdot S_2)$ the test vector $U_2 \cdot P_u$. We obtain the set of test vectors $R(U_2 \cdot S_2) =$ {0100·100, 0100·010, 0100·101, 0100·000, 0100·011, 0100·001}. Instead of $U_2 \cdot S_2$, we include the set $R(U_2 \cdot S_2)$ in CR. The resulting test set CR is shown in Table 4.

Table 4: The test set CR for $s27$

i	U_i	S_i	i	U_i	S_i
0	0000	011	6	0100	010
1	1001	010	7	0100	101
2	0111	001	8	0100	000
3	1101	011	9	0100	011
4	1010	000	10	0100	001
5	0100	100			

By ordering the test vectors in $R(U_i \cdot S_i)$ as we do, we ensure that earlier test vectors are more likely to detect the faults detected by $U_i \cdot S_i$, while later test vectors can be dropped by performing fault simulation. The upper bound on the number of test vectors included in CR before dropping any test vectors is $|C| L_{MAX}$ (for every original test vector in C we may include in CR up to L_{MAX} test vectors).

Performing fault simulation with fault dropping for the test set CR shown in Table 4, we find that test vectors $U_6 \cdot S_6, \cdots, U_{10} \cdot S_{10}$ do not detect any new faults. We end up with a test set CR that includes the first six test vectors in Table 4.

3.2. Procedure

The procedure for constructing CR from C is given next as Procedure 2.

Procedure 2: Constructing the test set CR

(1) Let $C = \{U_i \cdot S_i : 0 \le i < n\}$ be the given test set. Set $CR = \phi$.
(2) For $i = 0,1, \cdots, n-1$:
  (a) Call Procedure 1 with S_i.
  (b) If $S_i \in \Psi$, add $U_i \cdot S_i$ to CR. Else:
    (i) Set $min\_dist = \min\{dist(S_i, P_u) : P_u \in \Psi\}$.
    (ii) For $dist = min\_dist, min\_dist+1, \cdots, N_{SV}$ (where N_{SV} is the number of state variables):
      For every $P_u \in \Psi$, if $dist(S_i, P_u) = dist$, add $U_i \cdot P_u$ to CR.
(3) Let $CR = \{U_i \cdot S_i : 0 \le i < m\}$. Let F be the set of faults detected by C. For $i = 0,1, \cdots, m-1$:
  (a) Simulate F under $U_i \cdot S_i$ with fault dropping.
  (b) If $U_i \cdot S_i$ does not detect any fault in F, remove $U_i \cdot S_i$ from CR.

The complexity of Procedure 2 is determined by the fact that Procedure 1 is called $|C|$ times. Procedure 2 also performs fault simulation with fault dropping of CR.

4. Experimental results

The results of the application of Procedure 2 to single stuck-at faults in ISCAS-89 and ITC-99 benchmark circuits are reported in this section. We only consider circuits for which synchronizing sequences can be computed starting from the all-unspecified initial state using three value logic.

For all the circuits we use a compact test set C_0 as the test set C. The test set C_0 is obtained by the dynamic compaction procedure of [14] for ISCAS-89 benchmark circuits. For ITC-99 benchmark circuits it is selected out of a large set of random vectors and compacted by reverse order fault simulation.

To provide additional flexibility in constructing CR, we also use an incompletely specified test set C_x obtained as follows. We start from the compact test set C_0. For every fault f, we find the first test $U_i \cdot S_i$ in C_0 that detects f. This test is obtained by fault simulation with fault dropping of C_0. We then consider each fault f. Let f be detected by $U_i \cdot S_i \in C_0$. Considering only S_i, we unspecify the bits of S_i one at a time. If f is still detected by $U_i \cdot S_i$ after unspecifying bit j of S_i, we keep bit j unspecified; otherwise, we restore the original value of bit

j under S_i. At the end of this process we have a new test $U_i \cdot \hat{S}_i$ for f, where $\hat{S}_i$ is incompletely specified. We add $U_i \cdot \hat{S}_i$ to C_x and drop from consideration all the faults that it detects. The test set C_x is typically larger than C_0 since fewer faults are detected by each test.

When Procedure 1 is applied to check whether an incompletely specified state S_i is reachable, the distance between $\hat{S}_i$ and a state Q_r reached during Procedure 1 is defined over the specified bits of $\hat{S}_i$. As a result, the search for a reachable state concentrates on fewer specified bits, and it is more effective. When C is incompletely specified, we allow incompletely specified states Q_r to be included in Ψ. As a result, CR may be incompletely specified as well.

As parameters for Procedure 1 when C is the compact test set C_0 we use $L_{MAX} = 1000$ and $N_{RAND} = 100$. When C is the incompletely specified test set C_x, it is easier to reach the states of C_x or states close to them and we use $L_{MAX} = 200$ and $N_{RAND} = 100$.

Procedure 2 reduces the size of CR by fault simulation with fault dropping that eliminates unnecessary tests. To reduce the size of CR further, we also apply to CR forward-looking reverse-order fault simulation [15]. This is a static compaction procedure similar to reverse order fault simulation but more effective in removing unnecessary tests.

The results are shown in Tables 5 and 6. Part (a) of every table contains information about numbers of (detected) faults while part (b) contains information about test set sizes. The cases where we use the test set C_x for C are marked with x's. In all other cases we use a conventional, compact test set C_0.

In part (a) of every table, after the circuit name we show the number of state variables and the number of single stuck-at faults. Under column *detected* we show the following numbers of detected faults. Under subcolumn *seq* we show the number of faults detected by a test sequence generated for the non-scan sequential circuit. For ISCAS-89 benchmark circuits the test sequences are generated by a test generation procedure that can detect all or almost all the sequentially detectable faults. Under subcolumn *orig*(C) we show the number of faults detected by C. Under subcolumn *mod*(CR) we show the number of faults detected by CR. In the last column we show the run time of Procedure 2 in seconds on a Sun B1000 workstation (this includes the run time of the calls to Procedure 1).

In part (b) of every table, under column *orig*(C) we show the number of test vectors in C, and the number of test vectors in C that have reachable states. Under column *mod*(CR) we show the number of test vectors in CR before removing unnecessary test vectors, the number of test vectors in CR after removing unnecessary test vectors (this is the number of *effective* test vectors in CR), and the number of test vectors in CR after forward-looking reverse-order fault simulation. The last column will be explained later. The following points can be seen from Tables 5 and 6.

Table 5: Experimental results for ISCAS-89

(a) Numbers of faults

			detected			
				orig	mod	
circuit	s.v.	flts	seq	(C)	(CR)	time(sec)
s208	8	215	150	215	155	62.25
s298	14	308	273	308	273	81.57
s344	15	342	335	342	335	68.45
s382	21	399	378	399	376	154.11
s400	21	421	395	415	392	156.07
s420	16	430	204	430	209	200.45
s526	21	555	463	554	460	415.48
s641	19	467	408	467	408	174.81
s1196	18	1242	1239	1242	1241	5346.29
s1423	74	1515	1457	1501	1434	732.81
s1423x	74	1515	1457	1501	1457	2429.45
s5378	179	4603	3659	4563	3428	10769.49
s5378x	179	4603	3659	4563	3681	28353.72
s35932	1728	39094	35110	35110	35110	14141.35

(b) Test set sizes

	orig(C)		mod(CR)			comb
circuit	tst	rch	all	eff	flr	(CmR)
s208	27	1	426	32	19	38
s298	24	1	974	43	25	37
s344	15	0	2769	37	17	20
s382	25	0	12406	61	28	38
s400	24	0	13264	61	32	42
s420	43	1	374	25	17	57
s526	50	0	13538	93	51	81
s641	22	0	6617	58	28	42
s1196	138	1	136590	192	135	136
s1423	26	0	25948	174	70	92
s1423x	676	300	61051	136	71	105
s5378	100	0	96910	251	80	180
s5378x	1625	644	195859	207	97	565
s35932	13	0	13000	321	52	52

x - incompletely specified test set

(1) The compact test set C_0, when used as C, typically includes very few test vectors with reachable states. The incompletely specified test set C_x includes large numbers of reachable states. This is due to the fact that the incompletely specified states in C_x are easier to reach.

(2) The number of test vectors included in CR is typically high, sometimes very high, compared to the number of test vectors in C. However, by dropping unnecessary test vectors it is possible to obtain a test set which is in most cases smaller than C.

(3) The number of faults detected by CR is typically higher than the number of faults detectable in the non-scan circuit. The detectable faults of the non-scan circuit constitute a subset of the faults that affect the functionality of the circuit. The other subset of faults that affect the circuit functionality are undetectable faults, which are not redundant (partially detectable faults) [16]. The number of faults detected by CR is smaller than the number of faults detectable by the test vectors in C, which may have unreachable states.

(4) The use of C_x instead of C_0 for the test set C can result in a test set CR that detects significantly more faults. This is due to the fact that the states in C_x are easier to reach and hence the states contained in Ψ are closer to the states in C_x. Consequently, the tests in CR

Table 6: Experimental results for ITC-99
(a) Numbers of faults

circuit	s.v.	flts	detected seq	detected orig (C)	detected mod (CR)	time (sec)
b03	30	452	334	452	349	150.20
b04	66	1346	1168	1344	1248	1667.63
b04x	66	1346	1168	1344	1260	702.94
b05	34	1816	-	1729	1128	2037.02
b06	9	202	186	202	190	38.41
b07	51	1183	-	1153	935	776.37
b08	21	489	-	489	485	253.35
b08x	21	489	-	489	487	38.70
b09	28	420	339	420	394	126.22
b10	17	512	467	512	489	282.72
b10x	17	512	467	512	490	70.33
b11	30	1089	997	1078	1042	1547.31

(b) Test set sizes

circuit	orig(C) tst	orig(C) rch	mod(CR) all	mod(CR) eff	mod(CR) flr	comb (CmR)
b03	31	0	5604	55	33	52
b04	62	0	61998	150	47	69
b04x	643	469	34635	101	54	135
b05	83	0	280	68	40	102
b06	20	0	87	22	16	19
b07	69	0	344	49	32	85
b08	50	0	7506	90	50	53
b08x	228	207	3965	74	44	46
b09	32	0	1746	49	35	44
b10	57	1	1471	88	47	65
b10x	223	165	3207	65	46	67
b11	77	0	62259	145	70	86

x - incompletely specified test set

are closer to the tests in C_x, resulting in the detection of more faults.

After *CR* is applied, to achieve a higher level of reliability and for diagnosis, certain test vectors from *C* may be applied as well. The number of test vectors from *C* that need to be added to *CR* in order to achieve complete fault coverage is computed by performing fault simulation with fault dropping of *CR* followed by *C*. Tests in *C* that do not detect any new faults are dropped. The resulting test set detects as many faults as possible using reachable states and the remaining faults using unreachable states. We denote this test set by *CmR*.

We report on the size of *CmR* in the last column of Tables 5(b) and 6(b). It can be seen that the size of *CmR* is typically not significantly higher than the size of *C*. Thus, it is possible to maximize the detection of faults using reachable states without increasing the test set size significantly.

5. Concluding remarks

Scan allows the generation and application of test vectors that use unreachable states of the circuit, i.e., states that the circuit cannot reach starting from certain initial states. Faults detected by such test vectors may not affect the functional operation of the circuit (or its performance). Such test vectors can also result in higher power dissipation than the circuit is designed for. We described a procedure that accepts a test set *C* whose test vectors may include unreachable states, and produces a test set *CR* whose test vectors include only reachable states. Using a simulation-based process, the procedure checks for every test vector in *C* with state subvector *S* whether *S* is a reachable state. During this process, the procedure also collects a set of reachable states Ψ that are as close to *S* as possible. If *S* is reachable, the test vector is included without modification in *CR*. If *S* is unreachable, then the states in Ψ are used for defining new test vectors that will be included in *CR* instead of the original test vector. Our results demonstrated that the proposed procedure is able to produce test sets that detect many of the circuit faults, which are detectable using scan, and practically all the sequentially irredundant faults, by using test vectors with reachable states. The procedure is applicable to the initial scan-in state of every test in a scan-based test set for any fault model.

References

[1] M. J. Y. Williams and J. B. Angell, "Enhancing Testability of Large Scale Integrated Circuits via Test Points and Additional Logic", IEEE Trans. on Computers, 1973, pp. 46-60.

[2] E. B. Eichelberger and T. W. Williams, "A Logic Design Structure for LSI Testability", in Proc. 14th Design Autom. Conf., June 1977, pp. 462-468.

[3] S. M. Reddy and R. Dandapani, "Scan Design Using Standard Flip-Flops", IEEE Design & Test, Feb. 1987, pp. 52-54.

[4] M. Abramovici, P. S. Parikh, B. Mathew and D. G. Saab, "On Selecting Flip-Flops for Partial Reset", in Proc. Intl. Test Conf., 1993, pp. 1008-1012.

[5] V. Chickermane, E. M. Rudnick, P. Banerjee and J. H. Patel, "Non-Scan Design-for-Testability Techniques for Sequential Circuits", in Proc. 30th Design Autom. Conf., 1993, pp. 236-241.

[6] P. Parikh and M. Abramovici, "On Combining Design for Testability Techniques", in Proc. 1995 Intl. Test Conf., Oct. 1995, pp. 423-429.

[7] D. K. Das and B. B. Bhattacharya, "Testable Design of Non-Scan Sequential Circuits using Extra Logic", in Proc. Asian Test Symp., Nov. 1995, pp. 176-182.

[8] I. Pomeranz and S. M. Reddy, "On the Use of Fully Specified Initial States for Testing of Synchronous Sequential Circuits", IEEE Trans. on Computers, Feb. 2000, pp. 175-182.

[9] X. Dong, X. Yi and H. Fujiwara, "Non-Scan Design for Testability for Synchronous Sequential Circuits Based on Conflict Analysis", in Proc. Intl. Test Conf., 2000, pp. 520-529.

[10] H. Fujiwara, "A New Class of Sequential Circuits with Combinational Test Generation Complexity", IEEE Trans. on Computers, Sept. 2000, pp. 895-905.

[11] M. Abramovici, X. Yu and E. M. Rudnick, "Low-Cost Sequential ATPG with Clock-Control DFT", in Proc. 39th Design Autom. Conf., June 2002, pp. 243-248.

[12] J. Rearick, "Too Much Delay Fault Coverage is a Bad Thing", in Proc. Intl. Test Conf., Oct. 2001, pp. 624-633.

[13] V. D. Agrawal, K. T. Cheng, and P. Agrawal, "A Directed Search Method for Test Generation Using Concurrent Simulator," IEEE Trans. on Computer-Aided Design, Feb. 1989, pp. 131-138.

[14] S. Kajihara, I. Pomeranz, K. Kinoshita and S. M. Reddy, "Cost-Effective Generation of Minimal Test Sets for Stuck-at Faults in Combinational Logic Circuits", IEEE Trans. on Computer-Aided Design, Dec. 1995, pp. 1496-1504.

[15] I. Pomeranz and S. M. Reddy, "Forward-Looking Fault Simulation for Improved Static Compaction", IEEE Trans. on Computer-Aided Design, October 2001, pp. 1262-1265.

[16] I. Pomeranz and S. M. Reddy, "Classification of Faults in Synchronous Sequential Circuits", IEEE Trans. on Computers, Sept. 1993, pp. 1066-1077.